Multiply Robust Estimation for Local Distribution Shifts with Multiple Domains

Steven Wilkins-Reeves [1] **Xu Chen** [2] **Qi Ma** [2] **Christine Agarwal** [2] **Aude Hofleitner** [2]

Abstract

Distribution shifts are ubiquitous in real-world machine learning applications, posing a challenge to the generalization of models trained on one data distribution to another. We focus on scenarios where data distributions vary across multiple segments of the entire population and only make local assumptions about the differences between training and test (deployment) distributions within each segment. We propose a two-stage multiply robust estimation method to improve model performance on each individual segment for tabular data analysis. The method involves fitting a linear combination of the based models, learned using clusters of training data from multiple segments, followed by a refinement step for each segment. Our method is designed to be implemented with commonly used off-the-shelf machine learning models. We establish theoretical guarantees on the generalization bound of the method on the test risk. With extensive experiments on synthetic and real datasets, we demonstrate that the proposed method substantially improves over existing alternatives in prediction accuracy and robustness on both regression and classification tasks. We also assess its effectiveness on a user city prediction dataset from Meta.

1. Introduction

Machine learning models are often trained, evaluated and monitored across multiple segments in both scientific research and industrial applications (Perone et al., 2019; Wang et al., 2019; McMahan et al., 2013; Chen et al., 2023). Due to the heterogeneous nature of data distributions, the optimal model and its performance may differ among these segments. For example, in the context of the technology industry, it is crucial to evaluate model performance across different countries, end devices and age groups, as user behaviors can significantly vary across these segments (McMahan et al., 2013). In addition to the heterogeneity of data distributions across segments, it is also a common issue where the distribution of the data available for model training differs from that of the target population for model deployment (Storkey, 2009), primarily due to sampling bias (Bethlehem, 2010; González-Bailón et al., 2014).

While building domain[1]-specific models for all segments and then adjusting for sampling bias may be a straightforward solution, it presents at least three challenges: (1) unreliable model training on segments with an insufficient number of labels, (2) excessive computation when training individual models with a large number of segments, and, (3) a potential loss of information shared across different segments. To address these challenges, we aim to propose a flexible modeling framework for general supervised learning scenarios on tabular data, with the goal of improving model performance on each individual domain; however, we also discuss how our methods could be adopted to image and text data.

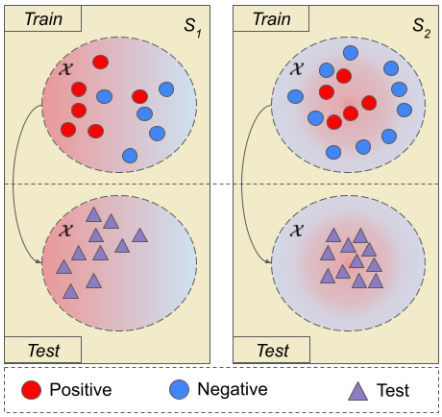

Figure 1. Schematic representation of the problem setup with two segments $\{S_1, S_2\}$. We observe both features and responses for the training data and only features for the test domain. We don't make any global assumptions of distribution shifts but only shifts within a segment.

Towards that end, we operate under the setting that the

[1]Department of Statistics, University of Washington, Seattle, USA [2]Central Applied Science, Meta, Menlo Park, USA. Correspondence to: Steven Wilkins-Reeves <stevewr@uw.edu>, Xu Chen <xuchen2@meta.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

[1]We use 'domain' and 'segment' interchangeably hereafter.

observed data in each domain adheres to the standard supervised learning setting. Specifically, we observe a training set and a test (or deployment) set in each domain. The training set consists of samples with a set of features and corresponding responses (X, Y), while the test set only contains samples with features X', and our objective is to predict their corresponding responses Y'.

To make accurate predictions of Y' under distribution shift, it is necessary to assume that the distribution of the training data P and that of the test data Q share a persistent property, a component that is identical for P and Q, to allow the model learned from the training data to be generalized to the test data. In classic domain adaptation literature, the two fundamental persistent property assumptions are covariate shift (Shimodaira, 2000) and label shift (Saerens et al., 2002). In this work, we adopt a local assumption that such persistent property only holds within each segment, without positing any restrictive global assumptions on how data distributions are different across segments. We refer to this assumption as local distribution shift. The problem setup we have described above is visualized in Figure 1.

Under the local distribution shift assumption, we propose a novel two-stage estimation method to enable both information sharing across segments and adaptive adjustments for each individual segment. In the first stage, we learn base models by utilizing the training data from multiple domains and then determine the optimal linear combination of these base models for each segment. In the second stage, we further refine the model from the first stage with a regularization to achieve an effective bias-variance trade-off. Inspired by the doubly robust estimation for covariate shift (Reddi et al., 2015) and multiple robustness methods used in causal inference and missing data problems (Han & Wang, 2013; Chan, 2013), we refer to the proposed method as 'multiply robust' estimation because it leverages all base models to create a domain-specific final model with enhanced predictive power and robustness.

Our main contributions are as follows.

- We introduce a relaxed framework for domain adaptation, by assuming the persistent properties only hold locally within each segment of the training and test data. We do not make any assumptions about how the data distributions may vary across different segments.
- We propose the multiply robust estimation method for tabular data analysis that is widely applicable
  - for both covariate shift and label shift scenarios.
  - in general supervised learning scenarios, including regression, binary and multi-class classifications.
  - with commonly used models such as linear models, tree-based models and more.
- From a theoretical viewpoint, we establish guarantees on the generalization bound for the proposed method on the test risk. From an empirical perspective, we demonstrate that the proposed method substantially outperforms over existing alternatives in terms of both prediction accuracy and robustness, through comprehensive numerical experiments on both synthetic and real datasets.

The rest of the paper is organized as follows. We review the related work and highlight the novelty of the proposed method in Section 2. In Section 3, we outline the problem formulation and assumptions along with notations to facilitate the introduction of the proposed method. We propose the multiply robust estimation method in Section 4 and provide pragmatic strategies for implementation. In Section 5, we justify the method with theoretical guarantees on generalization bound. We evaluate the empirical performance of the method with extensive experiments on both simulated and real datasets and compare it against several competing methods in Section 6. We conclude and discuss several future research directions in Section 7.

2. Related Work

We divide the related work broadly into two categories: single-source domain adaptation and multi-source domain adaptation.

Single-source domain adaptation. This category of research focuses on scenarios where all training data originates from a single-source domain. The two most common assumptions are covariate shift (Shimodaira, 2000) and label shift (Saerens et al., 2002). A substantial body of literature proposes weighting based methods to adjust the training data to align it with the target population. Some prominent examples include Kullback-Leibler importance estimation (Sugiyama et al., 2008, KLIEP), kernel mean matching (Gretton et al., 2009, KMM), penalized risk minimization (Nguyen et al., 2010, PRM) and discriminative learning based methods (Bickel et al., 2009) for covariate shift, and maximum likelihood label shift (Saerens et al., 2002, MLLS) and black box shift estimation (Lipton et al., 2018, BBSE) for label shift. These weighting methods can be adopted to adjust local distribution shifts in each segment, as we shall elaborate in Section 4.2.2.

Despite the wide usage of weighting correction, it may introduce high variance due to the potential extreme weights (Reddi et al., 2015). For variance reduction, a doubly robust (DR) method is proposed by incorporating a shrinkage towards the model trained without weights. Applying DR in our setting requires combining training data from all

segments. Therefore DR is sub-optimal because it is unable to effectively adjust for any potential distribution shifts across segments, even when domain-specific features are incorporated into the modeling (Section 6).

Multi-source domain adaptation. The work in this area assumes the presence of multiple source domains and aims to make predictions on a new target domain (Mansour et al., 2008). A line of work in the avenue seeks to learn domain-invariant representations for all domains using deep learning (Zhao et al., 2018; Xu et al., 2018). However, such representation can be challenging to learn without strong assumptions on the shared properties across segments (Zhu et al., 2019). To address this difficulty, Zhu et al. (2019) proposes MFSAN to extract both global and domain-specific features. Another research direction leverages optimal transport to learn a dictionary of data distributions and then approximate the target population using a mixture of the elements in the dictionary (Montesuma & Mboula, 2021; Montesuma et al., 2023, DADIL). We select two representatives, MFSAN and DADIL, from the two categories for comparison with the proposed method in numerical experiments in Section 6.

3. Problem Formulation

Supervised learning. We consider the standard framework for supervised learning. Let $X \in \mathcal{X} \subset \mathbb{R}^d$ denote the covariates (features), $Y \in \mathcal{Y} \subset \mathbb{R}$ denote the outcome of interest (label), and $f : \mathcal{X} \to \mathcal{Y}$ denote a prediction model, where $f \in \mathcal{F}$ for some function space $\mathcal{F}$.

Multi-Segment Unsupervised domain adaptation. In the setting of domain adaptation, we consider N segments that span the whole population. Let $S \in \mathcal{S} = \{1, 2, ..., N_S\}$ denote the segment index, $\{Y_i, X_i, S_i\}_{i=1}^{n_{tr}}$ and $\{Y'_j, X'_j, S'_j\}_{j=1}^{n_{te}}$ be the training and testing sets sampled independently from population probability distributions P and Q respectively, where $P \neq Q$. Our goal is to construct segment-specific predictors $f^{(s)} : \mathcal{X} \to \mathcal{Y}$ which perform well on each segment's test distribution $Q^{(s)} \equiv Q(y, x|S = s)$ according to the segment-conditional risk $R_Q[f|s] = \mathbb{E}_{(X,Y)\sim Q^{(s)}}[\ell(Y, f(X))]$, where $\ell(\cdot, \cdot) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is a loss function. We consider unsupervised domain adaptation where the test labels Y'_js are unobserved. We similarly define $P^{(s)} \equiv P(y, x|S = s)$.

Local distribution shifts. Classical assumptions include covariate shift (Shimodaira, 2000) where $P(y|x) = Q(y|x),\ P(x) \neq Q(x)$ and label shift (Saerens et al., 2002) where $P(x|y) = Q(x|y),\ P(y) \neq Q(y)$. In our context, we adopt a relaxed version of these shifts where either covariate shift or label shift holds within each segment throughout this work. These assumptions include:

- Local covariate shift: $P^{(s)}(y|x) = Q^{(s)}(y|x)$
- Local label shift: $P^{(s)}(x|y) = Q^{(s)}(x|y)$.

We highlight the motivation for these relaxed assumptions with a toy example. Consider a scenario where the manifestation of symptoms X (such as fever) is contingent upon the status of an influenza infection Y, and we wish to develop a classifier for this year's flu optimized for both children ($S = 0$) and adults ($S = 1$) that exhibit differences in symptoms of infection $P(x|y, s = 0) \neq P(x|y, s = 1)$. Here, P represents the training distribution derived from historical data, while Q signifies the test distribution for the current flu season. Across various flu seasons, the prior propensity of contracting the flu may vary, expressed as $P(y|s) \neq Q(y|s)$. However, the distribution of symptoms given a flu status remains consistent, i.e., denoted by $P(x|y, s) = Q(x|y, s)$, thereby illustrating a local label shift as opposed to a global shift.

4. Multiply Robust Estimation

In this section we introduce our proposed multiply robust estimation method. This approach allows for information sharing across segments and adaptive adjustments for each domain, resulting in enhanced model performance on each specific domain. An overview of the proposal is as follows. We first cluster the segments into several groups and train a base model on each cluster. This enables information sharing across segments with similar data distributions and particularly beneficial for segments with a limited number of samples to leverage strength from others. To adjust for local distribution shifts, we estimate the importance weights to match the training data distribution to test data for each segment. We then learn a domain-specific model by finding the best linear combination of the base models, which we refer as stage 1 estimators. With the stage 1 estimators as 'priors', we further refine the estimators using weighted training samples. The pseudo-code of this procedure is presented in Algorithm 1.

We draw inspirations from doubly robust method (Reddi et al., 2015) and multiply robust methods used in causal inference and missing data problems (Han & Wang, 2013; Chan, 2013). We refer our proposal as 'multiply robust' because the predictive power of any base model can be leveraged in stage 1 estimators. If the linear combination is sub-optimal, further refinement is achieved with stage 2 estimators.

The key innovation of the proposed method is the two-stage estimation, which we delve into the details in Section 4.1. In Section 4.2, we provide pragmatic implementation strategies for the other components in Algorithm 1 including segment clustering and weighting. We discuss the computational complexity of the proposed method in Section 4.3.

Remark 4.1. In applications that involve image and text data,

Algorithm 1 Multiply Robust Estimator

INPUTS: Training data: $\{Y_i, X_i, S_i\}_{i=1}^{n_{tr}}$. Testing data: $\{X'_j, S'_j\}_{j=1}^{n_{te}}$. Weight estimator: $\mathcal{W} : (\{Y_i, X_i\}_{i=1}^{n_{tr}}, \{X'_j\}_{j=1}^{n_{te}}) \rightarrow \hat{w}$. Base Model Training Algorithm $\mathcal{H} : \{Y_i, X_i\} \rightarrow h_m$. Segment Clustering Algorithm $\mathcal{C}_M : \{Y_i, X_i, S_i\} \rightarrow \{\mathcal{S}_m\}_{m=1}^M, \mathcal{S}_m \subset \mathcal{S}$. Base training split $\varsigma \in [0, 1]$

Step 1: Randomly partition training indices into N_{base}, N_{tune} such that $|N_{base}|/|N| = \varsigma$

Step 2: Cluster the segments $\mathcal{C}_M(\{Y_i, X_i, S_i\}_{i=1}^{n_{tr}}) = \{\mathcal{S}_m\}_{m=1}^M$ into M clusters and a final group partition which includes all clusters $\mathcal{S}_{M+1} = \mathcal{S}$ where each $\mathcal{S}_m \subseteq \mathcal{S}$ are subsets of the set of segments.

Step 3: For each m, train a base model using the segments $h_m = \mathcal{H}(\{Y_i, X_i\}_{i \in N_{base}: S_i \in \mathcal{S}_m})$

for $s \in \mathcal{S}$ **do**

Step 4a: Estimate the weights for segment s, $\widehat{w}_s = \mathcal{W}(\{Y_i, X_i\}_{i=1:S_i=s}^{n_{tr}}, \{X'_j\}_{j=1:S'_j=s}^{n_{te}})$

Step 4b: (Stage 1 estimator). Using the tuning segment data $\{Y_i, X_i\}_{i \in N_{tune}: S_i = s}$, learn a linear combination of the base classifiers $\widehat{f}^{1,(s)} = \sum_{m=1}^M \widehat{\beta}_m^{(s)} h_m$.

Step 4c: (Stage 2 estimator). Refine a second model $\widehat{f}_{MR}^{(s)}$ using the training data within the segment $\{Y_i, X_i\}_{i \in N_{tune}: S_i = s}$ weighting each sample using the weights learned in step 4a, and penalizing the difference of $\widehat{f}_{MR}^{(s)}$ to $\widehat{f}^{1,(s)}$

end for

OUTPUT: Collection of refined estimators $\{\widehat{f}_{MR}^{(s)}\}_{s \in \mathcal{S}}$

it is common to exploit large-scale pre-trained models such as VGG-16 (Simonyan & Zisserman, 2014). A straightforward and widely adopted approach to applying our method to text and image data is to utilize the embeddings of a large pre-trained model, treating them as the feature vector. Subsequently, our method can be applied, as outlined in Algorithm 1, employing a regularized, high-dimensional linear model for both the base-model and the stage 2 estimator classes. While addressing label shift issues typically involves straightforward application of BBSE, caution should be exercised when dealing with high-dimensional embeddings, necessitating careful selection of an appropriate covariate shift weight estimation method.

4.1. Two Stage Domain-adaptive Estimation

We now elaborate the two-stage estimation method. Let $\{h_m\}_{m=1}^M, h_m : \mathcal{X} \rightarrow \mathcal{Y}$ denote a set of base models and $\mathbf{h}$ the vector collection of them. Let $R[f|\mathbf{Y}, \mathbf{X}, , s; \alpha]$ denote the α-weighted empirical training risk of segment s,

$$R[f|\mathbf{Y}, \mathbf{X}, s; \alpha] = \frac{1}{n_s} \sum_{i \in N_s} \alpha(Y_i, X_i) \ell(Y_i, f(X_i))$$

where N_s denotes the indices of the data corresponding to segment s: $N_s \subset \{1, 2, \ldots n\}, n_s = |N_s|$, and $\mathbf{Y}, \mathbf{X}$ denote the entire training data. We abbreviate the unweighted empirical risk $R[f|\mathbf{Y}, \mathbf{X}, s; \mathbf{1}] = R[f|\mathbf{Y}, \mathbf{X}, s]$. We let $\mathcal{B} \subseteq \mathbb{R}^M$ denote a convex subset for the parameters. Lastly, let $\Omega[f, g]$ denote a penalty function for the deviation of f and g (for example $\|f - g\|_{\mathcal{F}}$ for a function space $\mathcal{F}$, or $\|\beta_f - \beta_g\|_2$ for the parameters of a linear model), where $\Omega[f] = \Omega[f, 0]$. Given a set of estimated weights on the test set $\widehat{w}_s$, and a regularization parameter λ'_s[2]. Steps 4b and 4c are presented as follows

$$\widehat{\beta}^{(s)} = \text{argmin}_{\beta \in \mathcal{B}} R[\beta^\intercal \mathbf{h} | \mathbf{Y}, \mathbf{X}, s]$$

(Stage 1 Estimator)

$$\widehat{f}_{MR}^{(s)} = \text{argmin}_{f \in \mathcal{F}} R[f|\mathbf{Y}, \mathbf{X}, s; \widehat{w}_s] + \lambda'_s \Omega[\widehat{\beta}^{(s)\intercal} \mathbf{h}, f]$$

(Stage 2 Estimator)

4.1.1. Stage 1 Estimators

When $\mathcal{B} = \mathbb{R}^M$, the stage 1 estimator is a simple linear combination of base models. For example, if the loss function is the mean square loss this can be computed using linear regression or logistic regression when the ℓ is the cross entropy loss. In practice, the performance of the model at this stage is influenced by the quality of the base models. Therefore, as with all machine learning problems, it is crucial to select a method that is suitable for the specific prediction task at hand, including kernel methods, ensemble methods, and others.

To avoid overfitting in the first stage, one may consider restrictions on $\mathcal{B}$. Our default restriction is to consider the unit ball of coefficients $\beta \in \mathcal{B}_1 = \{\beta : \|\beta\|_2 \leq 1\}$, which includes convex combinations of the original model set. The unit ball restriction can be simply implemented through a successive application of ridge regression. for a specific regularization parameter λ

$$\widehat{\beta}_\lambda = \text{argmin}_{\beta \in \mathbb{R}^M} \sum_{i \in N_s} \ell(y_i, \beta^\intercal \mathbf{h}(X_i)) + \frac{\lambda}{2} \|\beta\|^2$$
$$= \text{argmin}_{\beta: \|\beta\|_2 \leq \nu} \sum_{i \in N_s} \ell(Y_i, \beta^\intercal \mathbf{h}(X_i))$$

there is a corresponding ν corresponding to the constrained optimization problem. We can find the corresponding λ via a binary search to find the smallest value of $\lambda \leq \lambda_{max}$ such that $\|\widehat{\beta}_\lambda\|_2 \leq 1$, for some preset λ_{max}.

4.1.2. Stage 2 Estimators

The stage 2 estimator requires penalization towards a base model. In the case of the mean square loss and a form of the penalty $\Omega[f, g] = \Omega[f - g]$, then this can be computed by fitting a regularized model fit to the residuals of the stage 1 model, $\delta Y_i = Y_i - \widehat{\beta}^{(s)\intercal} \mathbf{h}(X_i)$. For other loss functions,

[2]This can be any type of regularization, such as restricting the number of trees in an ensemble, or penalizing the coefficients in a linear model.

such as those involved in classification, one can include a base margin $\widehat{\delta}_i = \widehat{\beta}^{(s)\intercal}\mathbf{h}(X_i)$ then one can learn a model $\tilde{g}$ such that $f = \tilde{g} + \widehat{\delta}_i$ by penalizing the complexity of $\tilde{g}$, through $\Omega[\tilde{g}]$. This functionality is available in popular packages such as XGBoost (Chen & Guestrin, 2016, XGB) and can be similarly implemented in many common function classes such as SVMs, tree-based methods and neural networks. In the case of XGBoost $\Omega[\tilde{g}]$ involves penalties on the complexity of the second stage model, i.e. the number of trees, tree depth etc. We illustrate a cross validation procedure for our method in Appendix C.1.

4.2. Training Base Models and Learning Weights

To complete the understanding of Algorithm 1, we discuss pragmatic strategies for the remainder of the steps on obtaining base models and importance weights. For the former, since we use off-the-shelf methods for training the base models, we focus on segment clustering below.

4.2.1. Segment Clustering for Base Model Training

The trade-off in training base models is that, it is desirable to train models on more segments to reduce variance, while the variability across base models is also crucial to provide a richer function set for the stage 1 estimator. We balance the trade-off by first clustering the segments into groups $\mathcal{S}_m \subset \mathcal{S}$, based on the similarity of covariates and responses, and then training one base model using all the training data within each segment cluster $\{Y_i, X_i\}_{i:S_i \in \mathcal{S}_m}$.

One can cluster the segments with domain knowledge, such as a continent when the domains are countries. When external knowledge is not available to define clusters of segments, we cluster the segments into groups where the joint training distribution $P^{(s)}(y, x)$ are similar for $s \in \mathcal{S}_m$. A common choice to measure the similarity between two distributions P and Q is Maximum Mean Discrepancy (MMD) (Smola et al., 2006). With empirical measures $\widehat{P}, \widehat{Q}$ and corresponding samples $\{u_i\}_{i=1}^{m_1}$ and $\{v_i\}_{i=1}^{m_2}$, MMD can be computed explicitly in (1) where $k(\cdot, \cdot)$ is a positive definite, symmetric kernel function.

$$D_{MMD}(\widehat{P}, \widehat{Q}) = \frac{1}{m_1(m_1-1)} \sum_{i=1}^{m_1} \sum_{j \neq i}^{m_1} k(u_i, u_j) \tag{1}$$
$$+ \frac{1}{m_2(m_2-1)} \sum_{i=1}^{m_2} \sum_{j \neq i}^{m_2} k(v_i, v_j) - \frac{2}{m_1 m_2} \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} k(u_i, v_j)$$

We use Equation (1) to compute a distance matrix between segments and perform hierarchical clustering as described in Algorithm 2. The choice of kernel may be determined by the datatype. For example, we choose the delta kernel $k_y(y, y') = 1 - I(y = y')$ for categorical dimensions, where I is the indicator function, and choose the Gaussian kernel for continuous dimensions. The joint kernel $k_{y,x}$ can be computed as the product of the k_x and k_y kernels, $k_{y,x}((y, x), (y', x')) = k_y(y, y')k_x(x, x')$. In our experiments, we generally adhere to a rule of thumb where we select the largest M in Algorithm 2 that avoids generating singleton clusters, thereby ensuring a reasonably large function class in the base models. However, we note that generating an excessive number of distinct clusters could potentially restrict information sharing across segments, thereby diminishing the overall performance. One could also treat M as a tuning parameter and select it through cross validation.

Algorithm 2 Clustering Segments for Base Model Learning

1: **Input:** A kernel function: $k(\cdot, \cdot)$ and a training dataset $\{(Y_i, X_i, S_i)\}_{i=1}^{n_{tr}}$, number of desired clusters M.
2: **for** each segment pair $(s, s') \in \mathcal{S}^2$ **do**
3: Compute $\widehat{D}_{ss'} = D_{MMD}(\widehat{P}^{(s)}(y, x), \widehat{P}^{(s)}(y, x))$ where $\widehat{P}^{(s)}(y, x)$ corresponds to the empirical distribution of training data (Y, X) in segment s.
4: **end for**
5: Apply hierarchical agglomerative clustering to $\widehat{D}$ until M clusters are obtained.

4.2.2. Importance Weight Estimation

Lastly, we discuss options for addressing local distribution shifts. In classical covariate or label shift, one can weight the observations in the training distribution to match the test distribution. These sample weights can be used on the training distribution to debias the empirical training risk when the goal is minimizing the test risk. These weights, known as the importance weights, under covariate shift are defined by the ratio of densities (when X is continuous), $w(x) = \frac{dQ(x)}{dP(x)}$ (also known as the Radon-Nikodym derivative), and under label shift are defined by the ratio of class frequencies $w(y) = \frac{Q(y)}{P(y)}$.

Under our local covariate and label shift assumptions, we can define the analogous importance weights conditional on a segment s.

$$w_s(x) = dQ^{(s)}(x)/dP^{(s)}(x) \text{ local covariate shift}$$
$$w_s(y) = Q^{(s)}(y)/P^{(s)}(y) \text{ local label shift}$$

We can then estimate the weights using the methods discussed in Section 2 in each segment, such as PRM or KMM for covariate shift or BBSE for label shift.

4.3. Computational Complexity

The major computation burden in the proposed multiply robust method occurs at the segment clustering and 2-stage estimation. In Algorithm 2 for segment clustering, the distance matrix computation requires $\mathcal{O}(n_{tr}^2)$ operations, which is typically the most computationally expensive part of the pro-

cedure. The agglomeration clustering step has complexity $\mathcal{O}(MN_S^2)$ which tends to be much smaller as $N_S \ll n_{tr}$.

Suppose the complexities of training stage 1 and stage 2 estimators are $c_1(n_s)$ and $c_2(n_s)$ for segment s in step 4 of Algorithm 1. The total algorithmic complexity is $\sum_{s=1}^{N_S} c_1(n_s) + c_2(n_s)$. For example, if we adopt linear regression in stage 1 and XGB in stage 2, then $c_1(n_s) = \mathcal{O}(n_s M^2 + M^3)$ and $c_2(n_s) = \mathcal{O}(TD_T d n_s + p n_s \log(n_s))$ resulting in the total complexity scaling log-linearly with n_{tr}. Here T is the total number of trees and D_T is the maximum tree depth and d is the dimension of X. The base model training can be analyzed similarly as the stage 2 estimator, but for training M models. It is worth noting that each of the stage 1 and 2 estimators can be computed in parallel, also allowing for further algorithmic improvements in practice.

5. Theoretical Analysis

We now present a theoretical justification for the proposed multiply robust estimation method. As our goal is to construct a model that is adaptive to each segment with high predictive accuracy, we provide a performance guarantee by establishing a generalization bound on the test risk. Our proof leverages results from Ostrovskii & Bach (2021) to establish the convergence rate of the stage 1 model. We integrate a modified version of the risk minimization technique presented in Reddi et al. (2015). Notably, our methods are adaptable to any weight estimator, allowing us to separate the bound based on the quantities associated with the weight estimator $\widehat{w}_s$ and the function estimation of $\widehat{f}^{(s)}_{MR}$. The assumptions herein, though restrictive, are used to provide intuition on the performance of the algorithm and are not necessary for the algorithm to be effective in practice.

Recall the training risk for segment s, $R_P[f|s] = \mathbb{E}_{(Y,X)\sim P^{(s)}}[\ell(Y, f(X))]$ and denote the corresponding test risk $R_Q[f|s] = \mathbb{E}_{(Y,X)\sim Q^{(s)}}[\ell(Y, f(X))]$, and $\ell(y, \varrho)$ is a loss function which is convex in ϱ for each y, such as the mean square loss for regression or cross entropy loss for classification. Our goal is to minimize the test risk $R_Q[f|s]$ for any given segment s. We focus on one segment to establish the theory, which can then be applied to all of the segments.

Suppose a set of base models $\mathbf{h} = \{h_m\}_{m=1}^M$ and the candidate model f are contained in a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{F}$ equipped with the norm $\|\cdot\|_{\mathcal{F}}$. Let $\beta^{*(s)} = \text{argmin}_{\beta\in\mathcal{B}} R_P[\beta^\intercal \mathbf{h}|s]$ be the optimal linear combination and denote the following quantities derived from the base models

$$
\begin{aligned}
Z_{mi} &= h_m(X_i), \quad H = \mathbb{E}[\nabla^2_\beta \ell(Y, Z^\intercal \beta^{*(s)})|s] \\
G &= \mathbb{E}[\nabla_\beta \ell(Y, Z^\intercal \beta^{*(s)}) \nabla_\beta \ell(Y, Z^\intercal \beta^{*(s)})^\intercal |s] \\
M_{eff} &= \text{Tr}\left(H^{-1}G\right), \quad M_F = \max_m \|h_m\|_{\mathcal{F}} \\
\Sigma &= \mathbb{E}[ZZ^\intercal|s], \quad \text{Where } Y_i, X_i \sim_{iid} P^{(s)}(y,x).
\end{aligned}
$$

Let $\ell'(y, \varrho) = \frac{d}{d\varrho}\ell(y, \varrho)$ and similarly for higher order derivatives. Assume that each $Q^{(s)}$ is absolutely continuous with respect to $P^{(s)}$ such that there exists a set of bounded importance weights $w_s(x, y) = \frac{dQ^{(s)}(x,y)}{dP^{(s)}(x,y)}$.

Assumption 5.1 (Regularity Conditions)**.** We assume that the following regularity conditions hold:

$$|\ell(y, \varrho)| \le L, \quad |\ell(y, \varrho) - \ell(y, \varrho')| \le L|\varrho - \varrho'| \tag{2}$$

$$|\ell'''(y, \varrho)| \le |\ell''(y, \varrho)|, \quad \text{for all } \varrho \in \mathbb{R} \tag{3}$$

$$Z_{mi}, \ell'(Y, Z^\intercal \beta^{*(s)})Z, \ell''(Y, Z^\intercal \beta^{*(s)})^{1/2} Z \sim \text{subG} \tag{4}$$

$$\Sigma \preceq \rho H \text{ for some } \rho > 0 \tag{5}$$

$$\tilde{\lambda}_1 = \lambda_{min}(H) > 0 \tag{6}$$

$$w_s(y, x) \le \eta \text{ for all } x \in \mathcal{X}, y \in \mathcal{Y} \tag{7}$$

where $\xi \sim$ subG means that the ξ is sub-Gaussian (i.e., $\mathbb{E}[\exp(s(\xi - \mathbb{E}[\xi]))] \le \exp(\sigma^2 s^2/2)$ for some $\sigma > 0$ and for all $s \in \mathbb{R}$), and $\preceq$ denotes the semidefinite ordering.

Remark 5.2. In Assumption 5.1, Equation (2) is a simple boundedness and Lipschitz continuous property for the loss function. Equation (3) is the 2-self concordance property and Equations (4) and (5) are standard regularity assumptions for the finite sample convergence of parametric models as illustrated in Ostrovskii & Bach (2021) and satisfied for many convex loss functions such as those arising from exponential families. Equation (6) ensures that the optimal $\beta^{*(s)}$ is identifiable, as no model will be a linear combination of the others, and Equation (7) ensures that the true importance weights are bounded.

For each segment, let $f^{*(s)}_{q,\nu_{q,s}}$ denote the minimizer of the constrained test risk

$$f^{*(s)}_{q,\nu_{q,s}} = \text{argmin}_{f\in\mathcal{F}:\|f\|_{\mathcal{F}} \le \nu_{q,s}} R_Q[f|s] \tag{8}$$

where the target function class is bounded since an unregularized solution may not have a bounded norm. Let $\widehat{w}_s$ denote an estimate of the segment-specific weights w_s.

Theorem 5.3 (Generalization Bound For a Multiply Robust Estimator)**.** *Suppose that Assumption 5.1 holds. Then denote the 2-stage estimator illustrated in Algorithm 1,*

$$
\begin{aligned}
\widehat{\beta}^{(s)} &= \textit{argmin}_{\beta\in\mathcal{B}} R[\mathbf{h}^\intercal \beta|Y, X, s] \\
\widehat{f}^{(s)}_{MR} &:= \textit{argmin}_{f\in\mathcal{F}} R[f|Y, X, s; \widehat{w}_s] + \lambda'_s \|f - \widehat{\beta}^{(s)\intercal}\mathbf{h}\|^2_{\mathcal{F}} \\
&= \textit{argmin}_{f\in\mathcal{F}:\|f - \widehat{\beta}^{(s)\intercal}\mathbf{h}\| \le \nu'} R[f|Y, X, s, \widehat{w}_s]
\end{aligned}
$$

where $\nu' = \|\beta^{*(s)\intercal}\mathbf{h} - f^{*(s)}_{\nu_{q,s}}\|_{\mathcal{F}} + C\frac{M_F}{\tilde{\lambda}_1}\sqrt{\frac{M\log(e/\delta)}{n_s}}$ *and* C *is a universal constant. For* $n \geq M_{eff}M\log(eM/\delta)$ *then with probability at least* $1-2\delta$

$$R_Q[\widehat{f}^{(s)}_{MR}|s] \leq R_Q[f^{*(s)}_{\nu_{q,s}}|s] + \Delta_{MR,w_s,s} + \Delta_{MR,f,s} \quad (9)$$

where $\Delta_{MR,w_s,s}$ *is the bound related to the learning of the correct weights, and* $\Delta_{MR,f,s}$ *is the functional complexity part of the bound.*

$$\Delta_{MR,w_s} := 2L\frac{\|\widehat{w}_s - w_s\|_1}{n_s}$$

$$\Delta_{MR,f,s} := 2\eta L\left(\frac{2\nu'\sqrt{tr(\mathbf{K}_s)}}{n_s} + 3\sqrt{\frac{\log(2/\delta)}{2n_s}} + \frac{\|\widehat{\beta}^{(s)\intercal}\mathbf{h}(\mathbf{X}_s)\|_2}{n_s}\right)$$

where $\|\widehat{w}_s - w_s\|_1 = \sum_{i\in N_s}|\widehat{w}_s(X_i,Y_i) - w_s(X_i,Y_i)|$ *and* $\mathbf{K}_s$ *is the Gram matrix corresponding to the kernel function* $k(\cdot,\cdot)$ *associated with the space* $\mathcal{F}$, $\mathbf{K}_{s,ij} = k(X_i,X_j) : i,j \in N_s$, *and* $\|\widehat{\beta}^{(s)\intercal}\mathbf{h}(\mathbf{X}_s)\|_2 = \sum_{i\in N_s}(\widehat{\beta}^{(s)\intercal}\mathbf{h}(X_i))^2$

Remark 5.4 (Error decomposition). We split the components Δ_{MR,w_s} and $\Delta_{MR,f,s}$ so that the estimation rate of our method is separated from that of learning the importance weights. We discuss the estimation rates for different weighting methods in Appendix B.4. In particular, PRM and BBSE methods can guarantee estimation of the correct weights, at rates of $\Delta_{MR,w_s,s} = \mathcal{O}(n_s^{-1/4})$ and $\mathcal{O}(n_s^{-1/2})$ respectively.

Remark 5.5 (Comparison with global estimators). Unless the minimizers $f^{*(s)}_{\nu_{q,s}}$ are identical across all segments s, any single competing estimator $\tilde{f} \in \mathcal{F}$ with $\|\tilde{f}\|_{\mathcal{F}} \leq \nu_{q,s}$ will be asymptotically sub-optimal across every group. This includes global estimators that are trained on all segments, such as the standard doubly robust (DR) estimator we compare in Section 6 (Reddi et al., 2015).

Remark 5.6 (Comparison with doubly robust estimator). An alternative competitor one could suggest is to apply the DR estimator separately over every segment. Instead of employing a linear combination for the first stage, the DR learns an estimate $\widehat{f}^{(s)}_{p,\nu_{p,s}} \in \mathcal{F}$ trained on the unweighted empirical risk. Our comparison focuses on the functional complexity component of our bounds. They include a term analogous to our ν' in their bound, which we denote as ν'_{DR}. However, they demonstrate $\nu'_{DR} = \|f^{*(s)}_{\nu_{p,s}} - f^{*(s)}_{\nu_{q,s}}\|_{\mathcal{F}} + \mathcal{O}(n_s^{-1/4})$, where $f^{*(s)}_{\nu_{p,s}}$ represents the population optimizer of the unweighted problem. In contrast, our first stage estimator is expressed as $\nu' = \|\beta^{*(s)\intercal}\mathbf{h} - f^{*(s)}_{\nu_{q,s}}\|_{\mathcal{F}} + \mathcal{O}(n_s^{-1/2})$, suggesting an improved rate for segments with small samples. Moreover if $\beta^{*(s)\intercal}\mathbf{h}$ proves to be a better approximation through similar segments to $f^{*(s)}_{\nu_{q,s}}$ then we can improve on the constant factor as well.

Remark 5.7 (Curse of dimensionality). In this analysis, it's important to acknowledge that our risk minimization is centered around comparing the estimator to the population minimizer within the function class of the Reproducing Kernel Hilbert Space (RKHS), thereby circumventing the curse of dimensionality. However, if one aims to minimize $R_Q[\cdot|s]$ across a broader function class we would expect to observe a dependence on the dimension of the features. Additionally, under covariate shift, the curse of dimensionality could potentially affect the estimation of the weights Δ_{MR,w_s}. Nonetheless, our theoretical findings offer valuable insights into our method's behavior.

6. Experiments

We evaluate the methods on both simulated and real data. We use **mean squared error** (MSE) for regression tasks and **cross entropy** (CE) for classification tasks for evaluation. Standard errors are reported in parentheses in the following tables. More implementation details such as weights estimation and model hyperparameters are included in Appendix C.2. The code is available at github.com/facebookresearch/multiply_robust.

We compare the proposed multiply robust method (MR) against several existing alternatives. We include a doubly robust[3] model (DR) and a version of DR that simply includes the segments as hot features, which we refer to as DR-SF. We also compare two multi-source domain adaptation methods MFSAN (Zhu et al., 2019) and DADIL (Montesuma et al., 2023). Since tree ensemble methods have demonstrated superior performance on tabular data learning (Prokhorenkova et al., 2018; Shwartz-Ziv & Armon, 2022), we also include XGB as a baseline and use XGB to construct base models for both MR and DR.

In summary, we find that the proposed MR consistently outperforms the competitors. In particular, MR achieves the lowest test losses on the majority of segments (see results in Figure 3 and Appendix C) on all datasets. Since XGB and DR do not account for the distributional heterogeneity across segments, their performances are typically sub-optimal. The improvement of DR-SF over DR in most cases suggests that including one-hot segment feature helps the model adapt to different segments, though with limited capability. The performance of MFSAN and DADIL are sensitive to hyperparameter settings. Even after fine-tuning (see Appendix C.3 for more details), they still cannot achieve comparable performance to MR and DR-SF. The issue with MFSAN lies in its attempt to align the predictions made by all the classifiers learned from different sources (Equation (6) in their paper). This alignment penalty in

[3]Though doubly robust methods were introduced to adjust for covariate shift, they can also handle label shift.

MFSAN could potentially hinder its performance, as it contradicts the goal of leveraging the heterogeneity of data from various sources. On the other hand, the dictionary learning approach, which is the core of DADIL, assumes that the data distribution in each target domain can be well represented by a mixture of learned distributions. However, this assumption is restrictive and may not hold in real-world applications. Even if this assumption holds, effectively learning the dictionary of moderate to high dimensional densities becomes challenging due to the curse of dimensionality. In addition, it is worth noting that MFSAN and DADIL require significantly more computation time compared to other competing methods even with the use of GPU[4] (Table 2).

6.1. Simulations

We first consider a simple covariate shift regression simulation. We simulate a response $y = \alpha_{0s} + \alpha_{1s}^\intercal x + \gamma_s x_2(1 + \sin(3x_1)) + \epsilon_i$ where $\epsilon_i \sim N(0, 0.3)$. The covariates $x = [x_1, x_2, x_3, x_4]^\intercal \in \mathbb{R}^4$ are sampled from $x \sim \mathcal{N}(\mathbf{0}_4, I_4)$ and $\mathcal{N}(\mathbf{1}_4, 0.3^2 I_4)$ for training and testing data, respectively. To construct segments, we set $\alpha_{0s} \in [-2, 2]$, evenly spaced for $s \in \{1, 2, \ldots, 20\}$ and $\alpha_{1s} = -\alpha_{0s}\mathbf{1}_4$.

We experiment with different sample sizes from 1000 to 10000 and report the average MSE with 100 repetitions for each. The results are summarized in Figure 2. In addition to the performance comparison summarized above, we observe in this example that the test loss of MR and DR-SF significantly decreases as sample size becomes larger. This aligns our understanding that the differences in distributions across segments can be learned and adjusted for more effectively with more data. This also complies with the generalization bound in Theorem 5.3 which is tighter as n_s becomes larger. However, the other the test losses for four methods does not exhibit a clear decreasing trend.

6.2. Public Datasets

We consider two classification datasets (`Adult` and `Banking`) and two regression datasets (`Apartment` and `Online News`) from the UCI repository (Dua & Graff, 2017). We also include a multiclass `customer` segmentation dataset (Vetrivel, 2021) from Kaggle. We include summary statistics, the segments used for partitioning, and the process by which we construct and correct for dataset shifts in Appendix C.2.

We report the test loss relative to XGB for all of the methods in Table 1. We observe that MR outperforms the competing models on all datasets, on binary, multi-classification and regression tasks with smallest standard errors on 4 out of 5 datasets. The results suggest the effectiveness of MR in terms of both predictive power and robustness. We include the detailed test losses on all segments for all datasets in Appendix C.5.

[4]Details on computing environment are in Appendix C.3

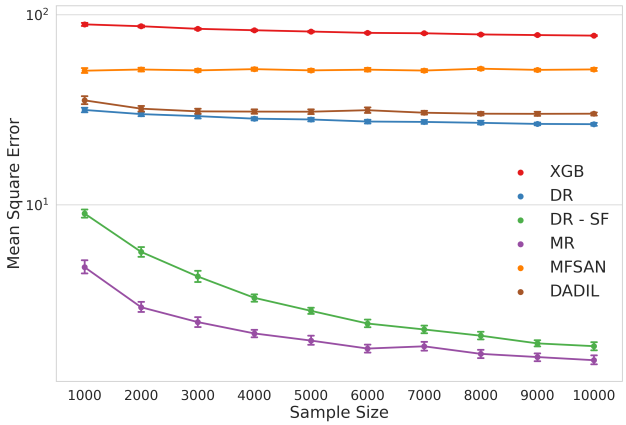

Figure 2. Performance of the proposed MR method against the competitors in the simulation study. Error bars indicate 99% confidence intervals.

6.3. Meta User City Prediction

We consider a dataset utilized for user city prediction from Meta. The observational unit of the dataset is a (user, city) pair and the features are mostly based on the interaction between a user and a city. A user could have multiple candidate cities and thus be associated with multiple (user, city) pairs. The labels are collected from the users, based on responses to a survey. The responses $y \in \{0, 1\}$ indicate whether a given city is the primary location of a given user. These predictions are then fed through downstream applications for multiple use cases.

The labeled data consists of a curated training dataset and a test dataset, where there are approximately 50,000 users in the training set, 12,000 users in the test set and approximately 50 features. On average there are 5 candidate cities per user in the training set and 11.5 in the test set. As a result, the proportion of positive examples is 0.2 in the training set, compared to a lower proportion of 0.08 in the test set. The training set is derived from survey responses, which typically present fewer candidate cities. Consequently, a local label shift occurs due to the increased prevalence of negative examples in the test set, stemming from the larger pool of candidate cities per user. In this study, we segment users based on their country of origin. We consider a total of 21 segments, with 20 countries that are selected based on factors such as population size, user engagement level, and growth potential. The remaining users from other countries are grouped together as a single segment.

We assess each method's performance using test CE relative to the production model. We also include the **Brier score**, a measure of calibration of a classifier (Brier, 1950). As

Table 1. Performance of each of the methods on distribution shifted test sets of the UCI and Kaggle datasets.

MODELS	BINARY CLASSIFICATION [CE]		MULTICLASS [CE]	REGRESSION [MSE]	
	ADULT	BANKING	CUSTOMER	APARTMENT	ONLINE NEWS
XGB	1.000 (0.026)	1.0 (0.03)	1.0 (0.009)	1.0 (0.013)	1.0 (0.023)
DR	0.953 (0.027)	0.976 (0.03)	0.978 (0.009)	1.0 (0.013)	1.011 (0.023)
DR-SF	0.948 (0.027)	0.971 (0.03)	1.025 (0.01)	0.774 (0.01)	0.994 (0.023)
MR	**0.901 (0.028)**	**0.925 (0.03)**	**0.834 (0.008)**	**0.524 (0.008)**	**0.949 (0.023)**
MFSAN	1.232 (0.028)	1.01 (0.032)	0.853 (0.008)	1.873 (0.02)	1.129 (0.023)
DADIL	3.285 (0.038)	1.347 (0.038)	1.679 (0.016)	2.895 (0.03)	1.121 (0.023)

shown in Table 2, our multiply robust method consistently outperforms others in CE and Brier score. The CE results by country are also visualized in Figure 3, demonstrating improved performance compared to the production model across nearly all segments.

Table 2. Relative performance of various competing methods on location dataset with production model as baseline, and running time (in minutes) comparison. Both MFSAN and DADIL are implemented on GPU, whereas the remaining methods are run on CPU.

MODELS	MODEL PERFORMANCE		RUNTIME
	CE	BRIER	
XGB	0.974 (0.012)	0.978 (0.014)	0.09
DR	0.93 (0.012)	0.935 (0.013)	0.26
DR-SF	0.917 (0.011)	0.927 (0.013)	0.36
MR	**0.852 (0.012)**	**0.864 (0.013)**	5.17
MFSAN	1.36 (0.015)	1.401 (0.017)	47.36
DADIL	1.879 (0.021)	1.974 (0.019)	59.86

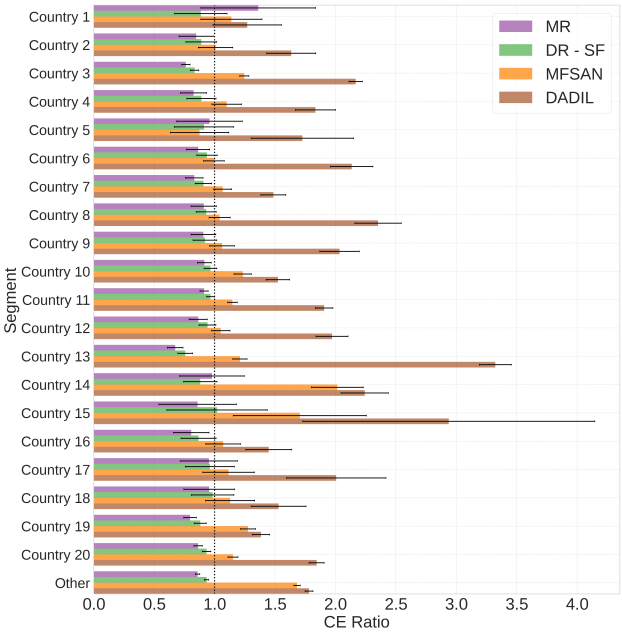

Figure 3. Test CE relative to the production model (black dotted line) by country. Error bars denote 99% confidence intervals.

7. Conclusion and Future Work

Existing assumptions for domain adaptation can be overly restrictive when applied to an entire population. To address this, we introduce and study a domain adaptation framework that allows for the relaxation of classical distribution shift assumptions to hold only within segments of the data. This presents a statistical challenge in refining a predictor across all segments of our population.

To address this challenge, we present a method to refine models across population segments. We provide theoretical justification and empirical evidence demonstrating the enhanced performance of our method compared to existing state-of-the-art domain adaptation techniques. Our approach is easily implementable using off-the-shelf algorithms.

Potential avenues for future work include exploring methods of jointly learning each of the refined models, by incorporating segmentation clustering, treating this as a multitask learning problem. Extensions to image and text data could also be considered, possibly utilizing pre-trained models in place of the base models in Algorithm 1. These problems however would involve further consideration of the trade-off between domain invariance properties and refinement for each segment. Lastly, our framework of studying a segmented distribution shift may have further connections with algorithmic fairness, as we avoid the problem of large segments dominating the training of the model. However, recent work explores the impact of distribution shift on fairness metrics and may be worth further exploration in segmented settings (Jiang et al., 2024).

Acknowledgements

The authors thank Sai Sumanth Miryala, Tao Wu, Xiaolu Xiong, and Andy Yu from the People Data Platform team for their support and feedback throughout this research.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. Journal of Machine Learning Research, 3(Nov):463–482, 2002.

Bethlehem, J. Selection bias in web surveys. International statistical review, 78(2):161–188, 2010.

Bickel, S., Brückner, M., and Scheffer, T. Discriminative learning under covariate shift. Journal of Machine Learning Research, 10(9), 2009.

Brier, G. W. Verification of forecasts expressed in terms of probability. Monthly weather review, 78(1):1–3, 1950.

Chan, K. C. G. A simple multiply robust estimator for missing response problem. Stat, 2(1):143–149, 2013.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794, 2016.

Chen, X., Marazopoulou, K., Lee, W., Agarwal, C., Sukumaran, J., and Hofleitner, A. Binary classifier evaluation on unlabeled segments using inverse distance weighting with distance learning. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 3877–3888, 2023.

Dua, D. and Graff, C. Uci machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

González-Bailón, S., Wang, N., Rivero, A., Borge-Holthoefer, J., and Moreno, Y. Assessing the bias in samples of large online networks. Social Networks, 38: 16–27, 2014.

Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., Schölkopf, B., et al. Covariate shift by kernel mean matching. Dataset shift in machine learning, 3(4): 5, 2009.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. The Journal of Machine Learning Research, 13(1):723–773, 2012.

Han, P. and Wang, L. Estimation with missing data: beyond double robustness. Biometrika, 100(2):417–430, 2013.

Jiang, Z. S., Han, X., Jin, H., Wang, G., Chen, R., Zou, N., and Hu, X. Chasing fairness under distribution shift: A model weight perturbation approach. Advances in Neural Information Processing Systems, 36, 2024.

Lipton, Z., Wang, Y.-X., and Smola, A. Detecting and correcting for label shift with black box predictors. In International conference on machine learning, pp. 3122–3130. PMLR, 2018.

Mansour, Y., Mohri, M., and Rostamizadeh, A. Domain adaptation with multiple sources. Advances in neural information processing systems, 21, 2008.

McMahan, H. B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D., et al. Ad click prediction: a view from the trenches. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1222–1230, 2013.

Montesuma, E. F. and Mboula, F. M. N. Wasserstein barycenter for multi-source domain adaptation. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 16785–16793, 2021.

Montesuma, E. F., Mboula, F. N., and Souloumiac, A. Multi-source domain adaptation through dataset dictionary learning in wasserstein space. arXiv preprint arXiv:2307.14953, 2023.

Nguyen, X., Wainwright, M. J., and Jordan, M. I. Estimating divergence functionals and the likelihood ratio by convex risk minimization. IEEE Transactions on Information Theory, 56(11):5847–5861, 2010.

Ostrovskii, D. M. and Bach, F. Finite-sample analysis of m-estimators using self-concordance. Electronic Journal of Statistics, 15:326–391, 2021.

Perone, C. S., Ballester, P., Barros, R. C., and Cohen-Adad, J. Unsupervised domain adaptation for medical imaging segmentation with self-ensembling. NeuroImage, 194: 1–11, 2019.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. Catboost: unbiased boosting with categorical features. Advances in neural information processing systems, 31, 2018.

Reddi, S., Poczos, B., and Smola, A. Doubly robust covariate shift correction. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 29, 2015.

Saerens, M., Latinne, P., and Decaestecker, C. Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. Neural computation, 14(1):21–41, 2002.

Shimodaira, H. Improving predictive inference under covariate shift by weighting the log-likelihood function. Journal of statistical planning and inference, 90(2):227–244, 2000.

Shwartz-Ziv, R. and Armon, A. Tabular data: Deep learning is not all you need. Information Fusion, 81:84–90, 2022.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

Smola, A. J., Gretton, A., and Borgwardt, K. Maximum mean discrepancy. In 13th international conference, ICONIP, pp. 3–6, 2006.

Storkey, A. When training and test sets are different: characterizing learning transfer. Dataset shift in machine learning, 30(3-28):6, 2009.

Sugiyama, M., Suzuki, T., Nakajima, S., Kashima, H., Von Bünau, P., and Kawanabe, M. Direct importance estimation for covariate shift adaptation. Annals of the Institute of Statistical Mathematics, 60:699–746, 2008.

Vetrivel, P. Kaggle: Customer segmentation, 2021. URL https://www.kaggle.com/datasets/vetrirah/customer/.

Wang, T., Johnson, T. S., Shao, W., Lu, Z., Helm, B. R., Zhang, J., and Huang, K. Bermuda: a novel deep transfer learning method for single-cell rna sequencing batch correction reveals hidden high-resolution cellular subtypes. Genome biology, 20(1):1–15, 2019.

Xu, R., Chen, Z., Zuo, W., Yan, J., and Lin, L. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 3964–3973, 2018.

Zhao, H., Zhang, S., Wu, G., Moura, J. M., Costeira, J. P., and Gordon, G. J. Adversarial multiple source domain adaptation. Advances in neural information processing systems, 31, 2018.

Zhu, Y., Zhuang, F., and Wang, D. Aligning domain-specific distribution and classifier for cross-domain classification from multiple sources. In Proceedings of the AAAI conference on artificial intelligence, volume 33, pp. 5989–5996, 2019.

A. Additional Implementation Details

A.1. Methods for Learning Importance Weights

We can apply standard techniques for learning the importance weights of the function class. Firstly, we consider learning covariate shift weights through discriminative learning, i.e. learning a probabilistic classifier that discerns between the weights (Bickel et al., 2009). Let $T = \{0, 1\}$ denote whether a sample is within the training or test distribution letting $P(\cdot) = \mathbb{P}(\cdot|T = 0)$ and $Q(\cdot) = \mathbb{P}(\cdot|T = 1)$, then the importance weights can be expressed up to proportion as

$$w(x) = \frac{dQ(x)}{dP(x)} \propto \frac{\mathbb{P}(T = 1|x)}{\mathbb{P}(T = 0|x)}.$$

Hence, one can estimate a probabilistic classifier to discriminate between $T = 0$ and $T = 1$, $\widehat{\mathbb{P}}(T = 1|x)$, which can then be used to construct a plug-in estimate

$$\widehat{w}(x) = \frac{\widehat{\mathbb{P}}(T = 1|x)}{\widehat{\mathbb{P}}(T = 0|x)}$$

Secondly, we consider Penalized Risk Minimization (Nguyen et al., 2010). This method involves the estimation as follows:

$$\begin{aligned}
\widehat{w}_s(x) &= \widehat{g}_s(x) \\
\widehat{g}_s &= \text{argmin}_{g \in \mathcal{G}} \frac{1}{n_{s,tr}} \sum_{i \in N_s 1} g(X_i) \\
&- \frac{1}{n_{s,test}} \sum_{j \in N'_s}^{n} \log(g(X'_j)) + \frac{\gamma_n}{2} \widetilde{I}^2(g)
\end{aligned}$$

where $\mathcal{G}$ is a reproducing Kernel Hilbert space, N_s and N'_s refer to the indices corresponding to training and test indices in segment s and $\widetilde{I}$ is a non-negative measure of complexity for g such that $\widetilde{I}(w) < \infty$ (for example, $\widetilde{I}(g) = \|g\|_{\mathcal{F}}$).

Another common approach to estimating importance weights is via Kernel Mean Matching (KMM) (Gretton et al., 2012). Given a positive definite kernel function $k(\cdot, \cdot) \to \mathbb{R}$ let $G_{ij} = k(X_i, X_j)$ and let $\kappa_i = \frac{n_{tr}}{n_{te}} \sum_{j=1}^{n_{te}} k(X_i, X'_j)$. Then kernel mean matching computes a set of weights $\{w_i\}_{i=1}^{n_{tr}}$

$$\min_{\mathbf{w}} \frac{1}{n_{tr}^2} \mathbf{w}^\intercal G \mathbf{w} - \frac{2}{n_{tr}^2} \kappa^\intercal \mathbf{w}$$

$$\text{such that } w_i \in [0, \eta] \text{ and } \left| \sum_{i=1}^{n_{tr}} w_i - n_{tr} \right| \leq n_{tr} \epsilon$$

where a good choice of ϵ is $\mathcal{O}(\eta/\sqrt{n_{tr}})$. For further details see (Gretton et al., 2009).

Lastly, when correcting for label shift one can use the Black Box Shift Estimation (BBSE) of (Lipton et al., 2018). This involves using a predictor $h(\cdot)$, and denote $\widehat{Y} = h(X)$. The corresponding importance weights $\tilde{\mathbf{w}} \in \mathbb{R}^K$ where $\tilde{w}(y) = \frac{Q(y)}{P(y)}$, can be defined through a linear equation of the expected confusion matrix under the training distribution $\mathbf{C}$, where $\mathbf{C}_{ij} = P(\widehat{Y} = i, Y = j)$ and the predicted class frequencies on the test distribution $\mu_{Q,h} \in \mathbb{R}^K$ where $\mu_{Q,h,j} = Q(\widehat{Y} = j)$. We differentiate the importance weights for each class $\tilde{\mathbf{w}} \in \mathbb{R}^K$ from $\mathbf{w} \in \mathbb{R}^n$, the weight vector for each training observation using the tilde annotation. The importance weights can then be defined by the solution to the following:

$$\mu_{Q,h} = \mathbf{C}\tilde{\mathbf{w}}$$

Using the training labels, one can estimate $\widehat{\mu}_{Q,h}$ and $\widehat{\mathbf{C}}$ using empirical means. As long as $\mathbf{C}, \widehat{\mathbf{C}}$ are invertible, then one can estimate the importance weights of the class

$$\widehat{\tilde{\mathbf{w}}} = \widehat{\mathbf{C}}^{-1} \widehat{\mu}_{Q,h}$$

B. Proofs of Theorem 5.3 and supporting Lemmas

In order to prove Theorem 5.3 we first introduce a series of supporting lemmas, after which we proceed with the proof of the theorem in Appendix B.3.

B.1. Supporting Lemmas

We first introduce a useful Lemma for proving the main theorem of our paper.

Lemma B.1 (Best Linear Combination As A Prior)**.** *Let $\{h_m\}_{m=1}^M$ be a collection of base models contained in some function space $h_m \in \mathcal{F}$. Let s denote the target segment of interest, and let $N_s \subset \{1, 2, n_{tr}\}$ be the set of indices of the training data corresponding to segment s with $|N_s| = n_s$. Let $\beta \in \mathcal{B} \subset \mathbb{R}^M$ denote the parameters for a linear combination of h_m. Let $\mathcal{B}$ be a convex set in the parameter space. Let $\beta^{*(s)}$ denote the optimal linear combination satisfying $\beta^{*(s)} = argmin_{\beta\in\mathcal{B}} R_P[\mathbf{h}^\intercal\beta|s]$ and the empirical estimate $\widehat{\beta}^{(s)} = argmin_{\beta\in\mathcal{B}} R[\mathbf{h}^\intercal\beta|\mathbf{Y}, \mathbf{X}, s]$. Denote the following random variables and parameters derived from the loss function and the base predictors:*

$$
\begin{aligned}
Z_{mi} &= h_m(X_i), \quad H = \mathbb{E}[\nabla^2_\beta \ell(Y, Z^\intercal\beta^{*(s)})|s] \\
G &= \mathbb{E}[\nabla_\beta \ell(Y, Z^\intercal\beta^{*(s)})\nabla_\beta \ell(Y, Z^\intercal\beta^{*(s)})^\intercal|s] \\
M_{eff} &= Tr\left(H^{-1}G\right), \quad M_F = \max_m \|h_m\|_{\mathcal{F}} \\
\Sigma &= \mathbb{E}[ZZ^\intercal|s], \quad \textit{Where } Y_i, X_i \sim_{iid} P^{(s)}(y, x).
\end{aligned}
$$

Assume that the conditions of Assumption 5.1 hold. Then for all $n \geq \rho M_{eff} M \log(eM/\delta)$

$$
\|\mathbf{h}^\intercal\beta^{*(s)} - \widehat{\beta}^{(s)\intercal}\mathbf{h}\|_{\mathcal{F}} \leq C\frac{M_F}{\tilde{\lambda}_1}\sqrt{\frac{M M_{eff}\log(e/\delta)}{n_s}} \tag{10}
$$

For a universal constant C

Lemma B.1 follows from a simple application of the results from (Ostrovskii & Bach, 2021). The main idea we exploit is that learning a simple linear combination is easier than learning in the larger function class $\mathcal{F}$. Since we assume that the total number of base estimators is not increasing, then M_{eff}, M_F, M and $\tilde{\lambda}_1$ are all constants and we recover the stage 1 model at a rate of $n_s^{-1/2}$. This therefore, allows us to learn this prior faster than that of the stage 2 model.

B.2. Proof of Lemma B.1

Proof. Recall $h_m : \mathcal{X} \to \mathcal{Y}$ are the set of base-models, and let $\mathbf{h}$ denote the vector of such functions. Denote a linear combination of these with weights β such that $\mathbf{h}^\intercal\beta = \sum_{m=1}^M h_m\beta_m$.

The first step is an immediate application of Theorem 1.1 of (Ostrovskii & Bach, 2021). This theorem is applied as follows. Let ρ denote the minimum value such that $\Sigma \preceq \rho H$. Then for all $n_s \geq \rho M_{eff} M \log(eM/\delta)$

$$
R[\mathbf{h}^\intercal\widehat{\beta}^{(s)}|s] - R[\mathbf{h}^\intercal\beta^{*(s)}|s] \lesssim (\widehat{\beta}^{(s)} - \beta^{*(s)})^\intercal H^2(\widehat{\beta}^{(s)} - \beta^{*(s)}) \lesssim \frac{M_{eff}\log(e/\delta)}{n_s}
$$

where $\lesssim$ denotes less than or equal to, up to a universal constant C.

Further let $\tilde{\lambda}_1$ denote the smallest eigenvalue of H. Then with probability at least $1 - \delta$

$$
\|\widehat{\beta}^{(s)} - \beta^{*(s)}\|_2^2 \lesssim \frac{M_{eff}\log(e/\delta)}{\lambda_1^2 n_s}.
$$

Therefore, for any norm associated with a function space $\|\cdot\|_{\mathcal{F}}$,

$$\begin{aligned}\|\mathbf{h}^\intercal \beta^{*(s)} - \mathbf{h}^\intercal \widehat{\beta}^{(s)}\|_{\mathcal{F}} &\leq \sum_{m=1}^{M} \|h_m(\beta_m^{*(s)} - \widehat{\beta}_r)\|_{\mathcal{F}} \\ &= \sum_{m=1}^{M} \|h_m(\beta_m^{*(s)} - \widehat{\beta}_r)\|_{\mathcal{F}} \\ &\leq \|\widehat{\beta}^{(s)} - \beta^{*(s)}\|_1 \sup_{m\in\{1,2,\ldots,M\}} \|h_m\|_{\mathcal{F}}\end{aligned}$$

Therefore our proof is complete after applying $\|\widehat{\beta}^{(s)} - \beta^{*(s)}\|_1 \leq \sqrt{M}\|\widehat{\beta}^{(s)} - \beta^{*(s)}\|_2$.

$$\|\mathbf{h}^\intercal \beta^{*(s)} - \mathbf{h}^\intercal \widehat{\beta}\|_{\mathcal{F}} \lesssim \frac{M_F}{\tilde{\lambda}_1}\sqrt{\frac{M M_{eff}\log(e/\delta)}{n_s}}$$

□

We next consider the following lemma, first introduced in (Reddi et al., 2015) with its few line proof for completeness.

Lemma B.2 (Lemma 2 of (Reddi et al., 2015))**.** *Suppose $f^*_{q,\lambda_{q,s}}$ denote the population maximizer as in equation (8). The following bound holds with probability at least $1-\delta$*

$$\sup_{f:\|f-f_0\|\leq\nu} |R[f|\mathbf{Y},\mathbf{X},s;w_s] - R_Q[f|s]| \leq \eta L\left(\frac{2\nu}{n_s}\sqrt{tr(\mathbf{K}_s)} + 3\sqrt{\frac{\log(2/\delta)}{n_s}} + \frac{1}{n_s}\sqrt{\sum_{i\in N_s} f_0^2(X_i)}\right)$$

Proof. This proof simply uses the standard Rademacher complexity bounds for kernel methods.

Let $\widehat{\mathcal{R}}_{n_s}(\mathcal{F})$ denote the empirical Rademacher average of a function class $\mathcal{F}$

$$\widehat{\mathcal{R}}_{n_s}(\mathcal{F}) = \mathbb{E}\left[\sup_{f\in\mathcal{F}} \frac{2}{n_s}\left|\sum_{i\in N_s}\sigma_i f(x_i)\right|\right]$$

where σ_i are independently drawn uniform $\{\pm 1\}$ random variables.

Using Lemma 22 of (Bartlett & Mendelson, 2002), and in combination with the boundedness of the loss function. It can be shown that with probability $1-\delta$

$$\sup_{f\in\mathcal{F}} |R[f|\mathbf{Y},\mathbf{X},s] - R_P[f|s]| \leq L\widehat{\mathcal{R}}_n(\mathcal{F}) + 3L\sqrt{\frac{\log(2/\delta)}{2n}}$$

due to the L-Lipschitz and L-bounded loss function ℓ Furthermore, in this setting, if the empirical distributions are weighted by some function $w_s(y,x) \leq \eta$. Then

$$\sup_{f\in\mathcal{F}} |R[f|\mathbf{Y},\mathbf{X},w_s] - R_P[f]| \leq L\eta\widehat{\mathcal{R}}_n(\mathcal{F}) + 3L\eta\sqrt{\frac{\log(2/\delta)}{2n}}$$

If $\mathcal{F} = \{f : \|f - f_0\| \leq \nu\}$, then $\widehat{\mathcal{R}}_n(\mathcal{F}) = \frac{2\nu}{n}\sqrt{\text{tr}(\mathbf{K})} + \frac{1}{n}\sqrt{\sum_{i=1}^n f_0^2(X_i)}$ and the proof is complete. □

Lastly, we include an upper bound on the weights.

Lemma B.3. *Let $\widehat{w}_s$ be an estimator of the true importance weights. Then*

$$\sup_{f\in\mathcal{F}} |R[f|\mathbf{Y},\mathbf{X},s,\widehat{w}_s] - R[f|\mathbf{Y},\mathbf{X},s,w_s]| \leq L\frac{\|\widehat{w}_s - w_s\|_1}{n_s}$$

Proof. The proof is a straightforward application of Holder's inequality.

$$\begin{aligned}\sup_{f\in\mathcal{F}}|R[f|\mathbf{Y},\mathbf{X},s,\widehat{w}_s]-R[f|\mathbf{Y},\mathbf{X},s,w_s]| &= \sup_{f\in\mathcal{F}}|\frac{1}{n_s}\sum_{i\in N_s}\ell(Y_i,f(X_i))(\widehat{w}_s(X_i,Y_i)-w_s(X_i,Y_i))| \\ &\leq \sup_{f\in\mathcal{F}}\max_{i\in N_s}|\ell(Y_i,f(X_i))|\frac{\|\widehat{w}_s-w_s\|_1}{n_s} \\ &\leq L\frac{\|\widehat{w}_s-w_s\|_1}{n_s}\end{aligned}$$

□

B.3. Proof of Theorem 5.3

In this proof, we will rely on several lemmas, which we include subsequently in Appendix B.1.

Proof. Recall the two-stage multiply robust estimator

$$\begin{aligned}\widehat{f}^{(s)}_{MR} &= \text{argmin}_{f\in\mathcal{F}}R[f|\mathbf{Y},\mathbf{X},s;\widehat{w}_s]+\lambda'\|\mathbf{h}^\intercal\widehat{\beta}^{(s)}-f\|_{\mathcal{F}} \\ &= \text{argmin}_{f:\|\mathbf{h}^\intercal\widehat{\beta}^{(s)}-f\|_{\mathcal{F}}\leq\nu'}R[f|\mathbf{Y},\mathbf{X},s;\widehat{w}_s]\end{aligned}$$

where the constraint value ν' denotes the corresponding value to the regularization parameter λ'.

We first draw on Lemma B.1 which states with probability at least $1-\delta$, $\|\mathbf{h}^\intercal\widehat{\beta}^{(s)}-f\|_{\mathcal{F}}\leq\|\beta^{*(s)\intercal}\mathbf{h}-f\|_{\mathcal{F}}+C\frac{M_F}{\tilde{\lambda}_1}\sqrt{\frac{RM_{eff}\log(e/\delta)}{n_s}}=\nu'$

Next, let B denote the bound from Lemma B.2 which holds with probability at least $1-\delta$

$$\sup_{f:\|f-\mathbf{h}^\intercal\widehat{\beta}\|\leq\nu'}|R[f|\mathbf{Y},\mathbf{X},s;w_s]-R_Q[f|s]|\leq B(n_s,\delta)$$

$$B(n_s,\delta):=\eta L\left(\frac{2\nu'}{n_s}\sqrt{\text{tr}(\mathbf{K}_s)}+3\sqrt{\frac{\log(2/\delta)}{n_s}}+\frac{1}{n_s}\sqrt{\sum_{i\in N_s}(\mathbf{h}^\intercal\widehat{\beta})^2(X_i)}\right)$$

Furthermore, let $C(n_s,\widehat{\beta})$ denote the bound from Lemma B.3 denoting the statistical error from estimating the importance weights:

$$|R_Q[f|\mathbf{Y},\mathbf{X},s,\widehat{w}_s]-R_Q[f|\mathbf{Y},\mathbf{X},s,w_s]|\leq C(n_s,\widehat{w}_s):=L\frac{\|\widehat{w}_s-w_s\|_1}{n_s}$$

Using these two bounds, we can define a simple bound on the generalization error, with probability at least $1-\delta$.

$$\begin{aligned}R_Q[\widehat{f}^{(s)}_{MR}|s] &\leq R_Q[\widehat{f}^{(s)}_{MR}|\mathbf{Y},\mathbf{X},s,w_s]+B(n_s,\delta) \\ &\leq R[\widehat{f}^{(s)}_{MR}|\mathbf{Y},\mathbf{X},s,\widehat{w}_s]+B(n_s,\delta)+C(n_s,\widehat{w}_s) \\ &\leq R[f^{*(s)}_{\nu_{q,s}}|\mathbf{Y},\mathbf{X},s,\widehat{w}_s]+B(n_s,\delta)+C(n_s,\widehat{w}_s) \\ &\leq R[f^{*(s)}_{\nu_{q,s}}|\mathbf{Y},\mathbf{X},s,w_s]+B(n_s,\delta)+2C(n_s,\widehat{w}_s) \\ &\leq R[f^{*(s)}_{\nu_{q,s}}|s]+2B(n_s,\delta)+2C(n_s,\widehat{w}_s)\end{aligned}$$

The third inequality follows from the fact $R[\widehat{f}^{(s)}_{MR}|\mathbf{Y},\mathbf{X},s,\widehat{w}_s]\leq R[f^{*(s)}_{\nu_{q,s}}|\mathbf{Y},\mathbf{X},s,\widehat{w}_s]$

Therefore, with probability at least $1-2\delta$

$$R_Q[\widehat{f}^{(s)}_{MR}|s]\leq R[f^{*(s)}_{\nu_{q,s}}|s]+2\eta L\left(\frac{2\nu'}{n_s}\sqrt{\text{tr}(\mathbf{K}_s)}+3\sqrt{\frac{\log(2/\delta)}{n_s}}+\frac{1}{n_s}\sqrt{\sum_{i\in N_s}f_0^2(X_i)}\right)+2L\frac{\|\widehat{w}_s-w_s\|_1}{n_s} \tag{11}$$

Where

$$\nu' = \|\mathbf{h}^\intercal \beta^{*(s)} - f^{*(s)}_{\nu_{q,s}}\| + C\frac{M_F}{\tilde{\lambda}_1}\sqrt{\frac{MM_{eff}\log(e/\delta)}{n_s}}$$

for a universal constant C.

In order for our method to be effective $\nu_{MR} = \|\mathbf{h}^\intercal \beta^{*(s)} - f^{*(s)}_{\nu_{q,s}}\|_{\mathcal{F}} \ll \|f^{*(s)}_{\nu_{q,s}}\|_{\mathcal{F}} = \nu_{q,s}$, however this is not necessary for developing the bound in the first place. In other words, the best linear combination of the base models trained on the unweighted segment, should be close to the optimizer of the weighted set in the function class $\mathcal{F}$. The idea here is that if we have a candidate set of prior functions, then finding the best linear combination is generally going to be a much easier statistical task than learning a function class regularized towards 0. □

B.4. Bounds on estimating the importance weights

Our upper bound in Theorem 5.3 relies on estimating the importance weights $\|\widehat{w}_s - w_s\|_1$. We focus on two upper bounds, one for covariate shift and one for label shift.

Lemma B.4 (Lemma 8 of (Reddi et al., 2015))**.** *Suppose that $\widehat{w}_s$ is an estimator for w_s obtained by Penalized Risk Minimization (as in (Nguyen et al., 2010)). Suppose $\gamma_n = cn_s^{-2/(2+\tau)}$ for some $\tau > 0$. Let $\eta : w_s \leq \eta$ denote the bound of the magnitude of the true importance weights. Then with probability at least $1-\delta$*

$$\frac{1}{n_s}\|\widehat{w}_s - w_s\|_1 \leq \sqrt{\gamma_n \eta} + \eta\sqrt[4]{\frac{8}{n_s}\log(2/\delta)} \tag{12}$$

We note that in general, learning the importance weights under covariate shift non-parametrically is a statistically challenging task. An alternative approach may be to propose some parametric model for w_s, which can help to reduce the final complexity.

A second approach we consider is the black box shift estimation (BBSE) for label shift, where in this case the outcome $y \in \{1, 2, \ldots, K\}$ (Lipton et al., 2018). It can be shown that the importance weights can be estimated at a $n^{-1/2}$ using the output of a classifier $\widehat{y} = h(x)$. For simplicity, assume that there are an equal number of train and test samples $n_{train} = n_{test} = n$.

Theorem B.5 (Theorem 3 from (Lipton et al., 2018))**.** *Let $\mathbf{C}$ denote the expected confusion matrix computed under the training distribution $\mathbf{C}_{i,j} = P(\widehat{Y} = i, Y = j)$ where $\mathbf{C}$ is invertible where σ_{min} it's smallest eigenvalue. Then there is a universal constant C such that for all $n > 80\log(n)\sigma_{min}^{-2}$. Then with probability at least $1 - \frac{5K}{n^{10}}$.*

$$\sum_{y\in\{1,2,\ldots,K\}} (\widehat{\tilde{w}}_s(y) - \tilde{w}_s(y))^2 \leq \frac{C}{\sigma_{min}^2}\left(\frac{(\sum_y(\tilde{w}_s(y))^2 + K)\log(n)}{n}\right)$$

It follows from a simple application of the Cauchy-Schwarz inequality that we can upper bound $\|\widehat{w}_s - w_s\|_1/n$

$$\begin{aligned}
\|\widehat{w}_s - w_s\|_1/n &= \frac{1}{n}\sum_{i\in N}|\widehat{\tilde{w}}_s(y_i) - \tilde{w}_s(y_i)| \\
&= \sum_{y\in\{1,2,\ldots,K\}} \frac{n_k}{n}|\widehat{\tilde{w}}_s(y) - \tilde{w}_s(y)| \\
&\leq \sqrt{\sum_{y\in\{1,2,\ldots,K\}}(\frac{n_k}{n})^2 \sum_{y\in\{1,2,\ldots,K\}}(\widehat{\tilde{w}}_s(y) - \tilde{w}_s(y))^2} \\
&\leq \sqrt{\sum_{y\in\{1,2,\ldots,K\}}(\widehat{\tilde{w}}_s(y) - \tilde{w}_s(y))^2} \\
&\leq \frac{\sqrt{C}}{\sigma_{min}\left(\sum_y \tilde{w}^2(y) + K\right)}\sqrt{\frac{\log(n)}{n}}
\end{aligned}$$

where n_k is the number of observations in N with outcome $y = k$. Therefore, we can effectively learn the weights.

C. Additional Empirical Details

C.1. Data splitting and cross validation

The primary reason for data splitting is to avoid overfitting that can occur when the same data is used for training the base models $h_m{}_{m=1}^{M}$ and subsequently fitting the stage 1 estimator. Alternatively, one may split the training data and preserve a holdout set (e.g. 20%) which will be used for estimating β, then use use the fine-tuning dataset to estimate $\widehat{f}^{(s)}$. We provide a visualization in Figure 4. This splitting procedure can be simply generalized to a K-fold versions, as well as grouped versions for data types that are clustered by observational unit (for example, the multiple candidate cities for each userid in our location dataset). Cross-validation details for the numerical experiments are included in Appendix C.2 and Appendix C.3.

Given a training and test dataset, we constructed a cross-validation procedure as follows:

1. Split the training data into K folds.
2. For each k in $1, 2, \ldots K$:
   (a) Use the kth fold as the validation fold, and combine the remaining folds to form the train fold.
   (b) Fit the MR estimator using algorithm 1 on the train fold, splitting it in a base train and a tune fold in the process.
   (c) Compute the importance weights on the validation fold to match the test data.
   (d) For each segment, evaluate the fine-tuned MR estimator on the weighted validation fold.

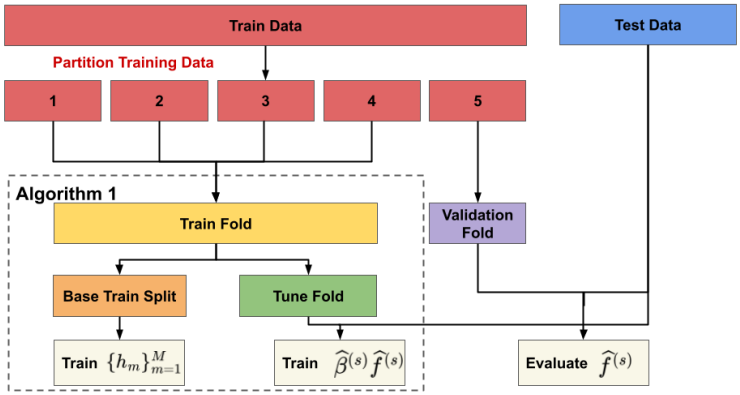

Figure 4. Data splitting for the multiply robust estimator. We use a holdout set for refining the linear combination, and train the second stage model on the whole training fold segment.

C.2. Real data experiment details and constructed distribution shifts.

We include a brief set of summary statistics for each of the datasets in our application along with descriptions of the constructed shifts in Table 3. We include the `Apartment` price and `Online News` popularity datasets for regression, `Banking` marketing campaign and `Adult` census datasets for binary classification, and the `customer` segmentation dataset (Vetrivel, 2021) for multi-classification.

In the regression datasets, we construct a covariate shift by picking the feature most correlated with the outcome, and creating a propensity for sampling into the train and test set respectively. In the Apartment price dataset, listings over 2000 square feet are selected into the test set with probability 0.8 with the remaining having probability 0.2. Similarly, if the average number of shares for articles by articles with the same key words is greater than 6000, then we select these into the test set with probability 0.8, and otherwise 0.2 respectively. In the case of label shift, we first construct an 80%, 20% split into a train and test set, then followup by sub-sampling the positive cases in the test set by $1/2$, thereby creating a change in the rate of positive classes in the test set. For the multiclass dataset, we resample the test fold according to the rates $(0.4, 0.1, 0.1, 0.4)$ for each of the 4 classes respectively.

For the experiments on the location platform dataset, we refer to the set of parameters in Appendix C, otherwise we refer to the default parameters of the models in Appendix C. All cross-validation procedures are $K = 5$ fold.

Table 3. Summary statistics for UCI datasets

DATASET	SAMPLE SIZE	NUMBER OF COVARIATES
ADULT	32561	15
BANKING	45211	16
CUSTOMER	10695	8
APARTMENT	87257	6
ONLINE NEWS	39644	51
DATASET	**DISTRIBUTION SHIFT**	**SEGMENTS**
ADULT	LAB. SHIFT	WORK CLASS (7)
BANKING	LAB. SHIFT	OCCUPATION (12)
CUSTOMER	LAB. SHIFT	ANONYMOUS GROUP (8)
APARTMENT	COV. SHIFT	STATES (30)
ONLINE NEWS	COV. SHIFT	DAY OF WEEK (7)
DATASET	**TASK**	**OUTCOME (POSITIVITY RATE)**
ADULT	CLASS.	INCOME $\geq$\$50K (0.24)
BANKING	CLASS.	SUBSCRIPTION (0.116)
CUSTOMER	MULTICLASS.	CONSUMER GROUP [A,B,C,D] (0.244, 0.230, 0.244, 0.281)
APARTMENT	REG.	PRICE (\$)
ONLINE NEWS	REG.	LOG(SHARES)

When correcting for covariate shift, we consider learning a Logistic Regression classifier, then use the predicted odds ratio as the importance weights. We this approach as it is simple to scale to the dataset sizes we consider as opposed to KMM or PRM which scale quadratically with training sample size in their complexity. When correcting for label shift, we use an XGB classifier trained on the unweighted training set as a base model and apply the BBSE method for label shift (Lipton et al., 2018).

C.3. Computing Environment and Tuning Parameters

C.3.1. COMPUTING ENVIRONMENT

Both MFSAN and DADIL are implemented on NVIDIA Tesla P100-SXM2 GPUs with 16 GB of memory. All other methods are run on Intel Core Processor (Broadwell) Model 61 with 36 cores.

C.3.2. TUNING PARAMETERS

XGB Tuning Parameters
When cross validation is not used to pick the tuning parameters we use the default parameters for XGB. The default for the refinement step stacked XGB are: `max_depth: 2, n_estimators: 25`.

Tuning parameters for the clustered segments are by default set to `colsample_bytree: 1.0, learning_rate: 0.1, max_depth: 3, n_estimators: 200, subsample: 0.8` while the base model is trained with the default parameters.

Cross Validation Tuning Parameters
All experiments involving XGB for base models were tuned with the following `n_estimators:` [10, 25, 50, 200, 300, 500], `max_depth`: [2, 3, 5, 7], `subsample`: [0.8, 1.0], `colsample_bytree`: [0.8, 1.0]. The `learning_rate` was kept at 0.1. For second stage estimators (refinement), we grid search over `learning_rate`: [0.001,0.01,0.1], `n_estimators:` [0, 10, 25, 50], `max_depth`: [0, 1, 3], `subsample`: [0.8, 1.0], `colsample_bytree`: [0.8, 1.0]. All methods were tuned using 5 fold cross-validation.

Tuning parameters for MFSAN The parameters of MFSAN were selected via grid search, while the search space are defined as `batch_size`: [32, 64, 128, 256], `learning_rate`: [0.0001, 0.0005, 0.001, 0.005], `adaptation_factor`: [0.1, 0.2, 0.3, 0.4, 0.5], `n_iterations`: [200, 300, 400, 500]. We uses *adam* optimizer throughout the experiments, and adapted the size of the sub-networks within MFSAN by different sizes of datasets to avoid overfitting.

Tuning parameters for DADIL The parameters of DADIL were selected via grid search, while the search space are defined as `batch_size`: [64, 128, 256], `learning_rate`: [0.001, 0.01, 0.1, 0.5], `n_components`: [6, 7, 8, 9, 10], `n_iterations`: [100, 150, 200]. We uses *adam* optimizer throughout the experiments.

C.4. Learned Clusters by Segment

After computing a distance matrix using Algorithm 2 we apply Ward's linkage for hierarchical agglomerative clustering and select the threshold that maximizes the number of base models M before isolating a single segment.

Table 4. Segment Clusters of the `Adult` Dataset

CLUSTER	WORK CLASS
0	['OTHER' 'PRIVATE' 'SELF-EMP-INC']
1	['FEDERAL-GOV' 'LOCAL-GOV' 'NEVER-WORKED' 'SELF-EMP-NOT-INC']

Table 5. Segment Clusters of the `Banking` Dataset

CLUSTER	JOB CATEGORY
0	['ADMIN.' 'BLUE-COLLAR' 'SERVICES' 'STUDENT']
1	['ENTREPRENEUR' 'MANAGEMENT' 'SELF-EMPLOYED' 'TECHNICIAN' 'UNEMPLOYED']
2	['HOUSEMAID' 'RETIRED' 'UNKNOWN']

Table 6. Segment Clusters of the `Apartment` Dataset

CLUSTER	COUNTRY
0	['AZ' 'FL' 'GA' 'NC' 'NV' 'TN' 'TX' 'VA']
1	['CA' 'MA' 'NJ']
2	['CO' 'IL' 'MD' 'OH' 'PA' 'WA']
3	['LA' 'MO' 'NE']

Table 7. Segment Clusters of the `Online News` Dataset

CLUSTER	DAY OF THE WEEK
0	['SATURDAY' 'SUNDAY']
1	['MONDAY' 'TUESDAY' 'WEDNESDAY' 'THURSDAY' 'FRIDAY']

Table 8. Segment Clusters of the `Customer` Dataset

CLUSTER	ANONYMIZED CATEGORY
0	['CAT_1' 'CAT_3' 'CAT_5' 'CAT_6' 'CAT_7' 'UNKNOWN']
1	['CAT_2' 'CAT_4']

Table 9. Base Model Clustering of Location Platform Dataset

CLUSTER	COUNTRIES
0	['COUNTRY 1' 'COUNTRY 7' 'COUNTRY 12' 'COUNTRY 18' 'OTHER']
1	['COUNTRY 2' 'COUNTRY 3' 'COUNTRY 6' 'COUNTRY 13' 'COUNTRY 16' 'COUNTRY 17' 'COUNTRY 20']
2	['COUNTRY 4' 'COUNTRY 9' 'COUNTRY 14']
3	['COUNTRY 5' 'COUNTRY 8' 'COUNTRY 10' 'COUNTRY 11']
4	['COUNTRY 15' 'COUNTRY 19']

C.5. Model Performance Across Segments

We include the performance of each of the models across the segments considered for each of the real datasets. We bold the top-performing model on each of the segments. We also calculated the area under the ROC curve, area under the precision-recall curve, and the F1 score for classification problems, however, their results closely align with production or unweighted models and are thus omitted for brevity. For classification tasks, Brier scores gave similar results to CE and are also omitted to avoid repetition.

Table 10. Apartment Prices. Partition by State

SEGMENT	XGB	DR	DR-SF	MR	MFSAN	DADIL
AZ	1.0 (0.062)	1.004 (0.063)	**0.378 (0.032)**	0.392 (0.03)	3.404 (0.173)	3.801 (0.167)
CA	1.0 (0.029)	0.996 (0.03)	0.388 (0.017)	**0.386 (0.018)**	0.669 (0.031)	0.708 (0.032)
CO	1.0 (0.061)	0.993 (0.063)	**0.784 (0.061)**	0.832 (0.062)	2.105 (0.09)	4.509 (0.216)
FL	1.0 (0.06)	1.029 (0.063)	0.981 (0.068)	**0.939 (0.06)**	2.311 (0.102)	3.84 (0.169)
GA	1.0 (0.044)	0.994 (0.045)	0.847 (0.038)	**0.673 (0.032)**	2.293 (0.094)	5.077 (0.22)
IL	1.0 (0.077)	0.998 (0.078)	1.055 (0.089)	**0.828 (0.078)**	2.18 (0.188)	1.631 (0.138)
LA	1.0 (0.056)	0.992 (0.059)	0.766 (0.05)	**0.294 (0.038)**	2.921 (0.176)	4.053 (0.204)
MA	1.0 (0.036)	0.997 (0.037)	1.187 (0.041)	**0.417 (0.02)**	0.64 (0.03)	0.727 (0.045)
MD	1.0 (0.059)	1.013 (0.06)	1.29 (0.071)	**0.696 (0.053)**	1.78 (0.07)	3.969 (0.191)
MO	1.0 (0.078)	1.003 (0.078)	0.792 (0.063)	**0.379 (0.031)**	3.59 (0.205)	2.735 (0.127)
NC	1.0 (0.037)	1.003 (0.038)	0.749 (0.031)	**0.417 (0.023)**	2.722 (0.085)	6.82 (0.171)
NE	1.0 (0.051)	1.008 (0.055)	0.738 (0.044)	**0.135 (0.014)**	2.871 (0.158)	3.421 (0.149)
NJ	1.0 (0.043)	0.995 (0.043)	1.208 (0.048)	**0.506 (0.027)**	0.948 (0.041)	1.054 (0.063)
NV	1.0 (0.051)	0.977 (0.051)	0.709 (0.041)	**0.413 (0.028)**	3.778 (0.15)	8.172 (0.346)
OH	1.0 (0.048)	0.991 (0.052)	0.771 (0.051)	**0.422 (0.024)**	3.267 (0.098)	4.19 (0.105)
PA	1.0 (0.098)	1.018 (0.103)	0.93 (0.091)	**0.794 (0.106)**	2.742 (0.209)	2.433 (0.191)
TN	1.0 (0.102)	1.01 (0.104)	0.838 (0.101)	**0.596 (0.096)**	2.519 (0.175)	3.751 (0.222)
TX	1.0 (0.03)	1.009 (0.031)	0.753 (0.023)	**0.612 (0.022)**	3.81 (0.084)	5.975 (0.136)
VA	1.0 (0.037)	0.993 (0.038)	1.012 (0.038)	**0.965 (0.036)**	2.233 (0.076)	3.466 (0.129)
WA	1.0 (0.055)	1.004 (0.062)	1.443 (0.08)	**0.44 (0.039)**	1.052 (0.076)	1.057 (0.122)
OVERALL MSE	1.0 (0.013)	1.0 (0.013)	0.774 (0.01)	**0.524 (0.008)**	1.873 (0.02)	2.895 (0.03)

Table 11. Online News Popularity Dataset. Partition by day of the week.

SEGMENT	XGB	DR	DR-SF	MR	MFSAN	DADIL
MONDAY	1.0 (0.064)	1.02 (0.064)	0.987 (0.063)	**0.965 (0.063)**	1.164 (0.064)	1.143 (0.064)
TUESDAY	1.0 (0.056)	1.011 (0.056)	0.984 (0.056)	**0.943 (0.057)**	1.134 (0.054)	1.129 (0.051)
WEDNESDAY	1.0 (0.05)	1.008 (0.05)	0.99 (0.053)	**0.955 (0.05)**	1.134 (0.052)	1.175 (0.054)
THURSDAY	1.0 (0.054)	1.016 (0.054)	1.013 (0.056)	**0.94 (0.052)**	1.12 (0.047)	1.116 (0.047)
FRIDAY	1.0 (0.053)	0.994 (0.051)	0.993 (0.053)	**0.984 (0.057)**	1.189 (0.064)	1.161 (0.06)
SATURDAY	1.0 (0.085)	1.009 (0.085)	0.994 (0.085)	**0.926 (0.077)**	1.035 (0.092)	1.007 (0.087)
SUNDAY	1.0 (0.078)	1.022 (0.079)	1.011 (0.08)	**0.873 (0.067)**	0.974 (0.084)	0.905 (0.066)
OVERALL MSE	1.0 (0.023)	1.011 (0.023)	0.994 (0.023)	**0.949 (0.023)**	1.129 (0.023)	1.121 (0.023)

Table 12. Adult Census dataset. Partition by Work Class

SEGMENT	XGB	DR	DR-SF	MR	MFSAN	DADIL
PRIVATE	1.0 (0.033)	0.955 (0.033)	0.968 (0.034)	**0.917 (0.036)**	1.229 (0.035)	3.582 (0.049)
FEDERAL-GOV	**1.0 (0.156)**	1.003 (0.164)	1.01 (0.133)	1.001 (0.142)	1.087 (0.146)	2.181 (0.168)
STATE-GOV	1.0 (0.149)	0.941 (0.149)	0.921 (0.148)	**0.845 (0.139)**	1.366 (0.158)	3.027 (0.163)
LOCAL-GOV	1.0 (0.095)	0.93 (0.095)	0.909 (0.098)	**0.859 (0.098)**	1.237 (0.098)	3.013 (0.116)
SELF-EMP-INC	1.0 (0.098)	**0.952 (0.096)**	1.017 (0.101)	0.966 (0.105)	1.158 (0.102)	2.319 (0.179)
SELF-EMP-NOT-INC	1.0 (0.074)	0.943 (0.074)	0.831 (0.071)	**0.796 (0.068)**	1.253 (0.08)	2.729 (0.091)
OTHER	1.0 (0.135)	0.949 (0.142)	0.92 (0.136)	**0.909 (0.157)**	1.289 (0.137)	3.172 (0.13)
OVERALL CE	1.0 (0.027)	0.953 (0.027)	0.948 (0.027)	**0.901 (0.028)**	1.232 (0.028)	3.285 (0.038)

Table 13. Banking Promotion dataset. Partition by Occupation Category

SEGMENT	XGB	DR	DR-SF	MR	MFSAN	DADIL
ADMIN.	1.0 (0.1)	**0.981 (0.101)**	1.02 (0.098)	1.018 (0.102)	1.096 (0.113)	1.46 (0.113)
BLUE-COLLAR	1.0 (0.083)	0.969 (0.084)	0.963 (0.081)	**0.875 (0.083)**	1.075 (0.093)	1.558 (0.11)
ENTREPRENEUR	1.0 (0.209)	0.979 (0.208)	0.979 (0.199)	**0.933 (0.2)**	0.945 (0.175)	1.266 (0.23)
HOUSEMAID	1.0 (0.159)	0.955 (0.157)	0.825 (0.147)	**0.806 (0.167)**	1.085 (0.197)	1.436 (0.229)
MANAGEMENT	1.0 (0.06)	0.977 (0.06)	0.972 (0.058)	**0.935 (0.059)**	0.984 (0.061)	1.256 (0.085)
RETIRED	1.0 (0.084)	0.979 (0.084)	0.923 (0.078)	**0.856 (0.074)**	0.989 (0.091)	1.241 (0.096)
SELF-EMPLOYED	1.0 (0.138)	0.978 (0.139)	0.938 (0.141)	**0.924 (0.14)**	0.956 (0.125)	1.214 (0.193)
SERVICES	1.0 (0.111)	**0.967 (0.112)**	1.028 (0.114)	0.969 (0.126)	1.152 (0.129)	1.669 (0.154)
STUDENT	1.0 (0.16)	0.98 (0.16)	1.043 (0.157)	0.912 (0.142)	**0.846 (0.123)**	1.215 (0.147)
TECHNICIAN	1.0 (0.072)	0.976 (0.072)	0.957 (0.071)	**0.927 (0.075)**	1.0 (0.079)	1.251 (0.085)
UNEMPLOYED	1.0 (0.153)	0.992 (0.158)	1.013 (0.168)	1.001 (0.179)	**0.895 (0.138)**	1.34 (0.199)
UNKNOWN	1.0 (0.297)	0.966 (0.302)	1.06 (0.332)	0.987 (0.3)	**0.93 (0.281)**	1.994 (0.417)
OVERALL CE	1.0 (0.03)	0.976 (0.03)	0.971 (0.03)	**0.925 (0.03)**	1.01 (0.032)	1.347 (0.038)

Table 14. Customer Segmentation Dataset. Partition by anonymized feature

SEGMENT	XGB	DR	DR-SF	MR	MFSAN	DADIL
CAT_1	1.0 (0.081)	0.984 (0.08)	1.001 (0.081)	0.78 (0.061)	**0.736 (0.055)**	1.876 (0.147)
CAT_2	1.0 (0.041)	0.997 (0.041)	1.053 (0.043)	**0.771 (0.03)**	0.793 (0.03)	1.668 (0.069)
CAT_3	1.0 (0.029)	0.991 (0.03)	1.021 (0.03)	**0.822 (0.023)**	0.917 (0.025)	1.594 (0.048)
CAT_4	1.0 (0.025)	0.975 (0.025)	0.948 (0.024)	**0.807 (0.02)**	0.958 (0.023)	1.581 (0.041)
CAT_5	1.0 (0.082)	0.964 (0.079)	0.997 (0.081)	0.924 (0.074)	**0.854 (0.067)**	1.743 (0.146)
CAT_6	1.0 (0.012)	0.973 (0.012)	1.039 (0.012)	0.846 (0.01)	**0.83 (0.009)**	1.703 (0.021)
CAT_7	1.0 (0.061)	0.997 (0.062)	1.002 (0.061)	0.972 (0.055)	**0.932 (0.052)**	1.673 (0.104)
UNKNOWN	1.0 (0.082)	1.0 (0.083)	1.032 (0.085)	**0.673 (0.053)**	0.764 (0.059)	1.817 (0.15)
OVERALL CE	1.0 (0.009)	0.978 (0.009)	1.025 (0.01)	**0.834 (0.008)**	0.853 (0.008)	1.679 (0.016)

Multiply Robust Estimation

Table 15. User City Dataset. Relative Cross Entropy to the Production model presented for each of the country segment. We include MR - LC for our method which only includes the first stage estimators without the second stage fine-tuning.

COUNTRY	XGB	DR	SF	DR-SF	MR	MR - LC	MFSAN	DADIL
COUNTRY 1	0.944 (0.222)	0.898 (0.217)	0.961 (0.24)	**0.887 (0.218)**	1.362 (0.475)	1.550 (0.512)	1.139 (0.253)	1.270 (0.283)
COUNTRY 2	0.921 (0.138)	0.876 (0.129)	0.938 (0.134)	0.890 (0.128)	**0.847 (0.142)**	1.041 (0.153)	1.008 (0.142)	1.635 (0.203)
COUNTRY 3	0.957 (0.035)	0.911 (0.032)	0.855 (0.037)	0.835 (0.035)	**0.763 (0.035)**	0.842 (0.037)	1.246 (0.038)	2.168 (0.056)
COUNTRY 4	0.951 (0.128)	0.912 (0.122)	0.940 (0.131)	0.891 (0.121)	**0.826 (0.108)**	0.876 (0.111)	1.101 (0.123)	1.835 (0.167)
COUNTRY 5	0.950 (0.258)	0.953 (0.259)	0.919 (0.259)	**0.913 (0.246)**	0.958 (0.271)	1.120 (0.222)	0.876 (0.242)	1.727 (0.424)
COUNTRY 6	0.962 (0.091)	0.910 (0.084)	0.989 (0.092)	0.937 (0.085)	**0.864 (0.097)**	0.991 (0.099)	0.995 (0.086)	2.135 (0.177)
COUNTRY 7	0.923 (0.068)	0.920 (0.068)	0.956 (0.072)	0.908 (0.069)	**0.832 (0.073)**	0.943 (0.075)	1.067 (0.074)	1.486 (0.104)
COUNTRY 8	0.975 (0.085)	0.924 (0.085)	0.990 (0.085)	0.931 (0.083)	**0.911 (0.105)**	1.089 (0.107)	1.042 (0.088)	2.353 (0.192)
COUNTRY 9	0.988 (0.109)	0.951 (0.102)	0.949 (0.101)	0.920 (0.098)	**0.908 (0.103)**	1.013 (0.103)	1.062 (0.104)	2.035 (0.164)
COUNTRY 10	0.996 (0.056)	0.946 (0.053)	1.010 (0.056)	0.966 (0.052)	**0.916 (0.058)**	1.017 (0.056)	1.234 (0.072)	1.524 (0.097)
COUNTRY 11	0.995 (0.035)	0.954 (0.034)	0.995 (0.033)	0.966 (0.032)	**0.914 (0.034)**	1.042 (0.032)	1.148 (0.040)	1.908 (0.072)
COUNTRY 12	0.965 (0.072)	0.932 (0.071)	0.967 (0.072)	0.942 (0.07)	**0.865 (0.075)**	0.937 (0.067)	1.051 (0.076)	1.972 (0.133)
COUNTRY 13	0.995 (0.052)	0.963 (0.050)	0.763 (0.060)	0.757 (0.061)	**0.674 (0.065)**	0.751 (0.065)	1.211 (0.059)	3.324 (0.133)
COUNTRY 14	1.004 (0.147)	0.994 (0.134)	**0.831 (0.153)**	0.882 (0.139)	0.978 (0.268)	1.036 (0.273)	2.016 (0.215)	2.243 (0.196)
COUNTRY 15	1.051 (0.472)	1.099 (0.465)	1.177 (0.485)	1.021 (0.416)	**0.86 (0.322)**	0.956 (0.269)	1.705 (0.551)	2.938 (1.209)
COUNTRY 16	1.048 (0.168)	0.963 (0.150)	1.016 (0.179)	0.868 (0.145)	**0.807 (0.147)**	0.920 (0.141)	1.071 (0.145)	1.448 (0.191)
COUNTRY 17	**0.900 (0.192)**	0.937 (0.207)	0.948 (0.206)	0.962 (0.204)	0.953 (0.237)	1.013 (0.251)	1.116 (0.215)	2.007 (0.412)
COUNTRY 18	1.024 (0.193)	0.984 (0.177)	0.981 (0.173)	0.984 (0.177)	**0.956 (0.21)**	1.311 (0.284)	1.129 (0.204)	1.531 (0.227)
COUNTRY 19	0.960 (0.054)	0.915 (0.051)	0.917 (0.052)	0.882 (0.05)	**0.797 (0.053)**	0.874 (0.05)	1.277 (0.062)	1.384 (0.071)
COUNTRY 20	0.970 (0.039)	0.932 (0.038)	0.977 (0.037)	0.934 (0.036)	**0.864 (0.035)**	0.983 (0.035)	1.153 (0.042)	1.845 (0.063)
OTHER	0.975 (0.020)	0.926 (0.019)	0.990 (0.019)	0.934 (0.018)	**0.859 (0.019)**	0.972 (0.018)	1.683 (0.027)	1.781 (0.033)

22