# Covariate Shift Adaptation in High-Dimensional and Divergent Distributions

**Felipe Maia Polo**
University of São Paulo
Advanced Institute for Artificial Inteligence (AI2)
felipemaiapolo@gmail.com

**Renato Vicente**
University of São Paulo
Experian DataLabs LatAm
rvicente@usp.br

## 1 Introduction[1]

In real world applications of supervised learning methods, training and test sets are often sampled from the distinct distributions and we must resort to domain adaptation techniques. One special class of techniques is Covariate Shift Adaptation[2], which allows practitioners to obtain good generalization performance in the distribution of interest when domains differ only by the marginal distribution of features. Traditionally, Covariate Shift Adaptation is implemented using Importance Weighting which may fail in high-dimensional settings due to small Effective Sample Sizes (ESS).

In this paper, we propose (i) a new connection between ESS, high-dimensional settings and generalization bounds and (ii) a simple, general and theoretically sound approach to combine feature selection and Covariate Shift Adaptation. The new approach yields good performance with improved ESS. Existing solutions to the same problem are not prone to interpretability [13] or general in terms of suitable hypothesis classes [10, 17].

From now on, we denote the source/train joint distribution as $Q_{\mathbf{x},\mathrm{y}}$ and a different target/test joint distribution by $P_{\mathbf{x},\mathrm{y}}$. Features and labels are sampled from the same measurable space $(\mathcal{X} \times \mathcal{Y}, \Sigma)$, with $Q_{\mathrm{y}|\mathbf{x}} = P_{\mathrm{y}|\mathbf{x}}$ and $Q_{\mathbf{x}} \neq P_{\mathbf{x}}$.

## 2 Effective Sample Size (ESS), Covariate Shift Adaptation and Generalization Bounds

In this section we assume the true importance function, a.k.a. density ratio, is known up to a constant. If ESS is a concept borrowed from Monte Carlo literature, how we should transpose the importance of ESS to the Covariate Shift Adaptation framework is an important question. In the following we show that there is a close relationship between the above definition of ESS and generalization bounds under importance weighting in covariate shift correction.

Consider two probability distributions $P_{\mathbf{x}}$ and $Q_{\mathbf{x}}$ over $\mathcal{X} \subseteq \mathbb{R}^d$, $P_{\mathbf{x}} \ll Q_{\mathbf{x}}$, with probability density functions $p_{\mathbf{x}}$ and $q_{\mathbf{x}}$, respectively. Suppose we have a random sample $\{\mathbf{x}_i\}_{i=1}^n$, independently sampled from the distribution $Q_{\mathbf{x}}$, and we define the weights $\mathrm{w}_i = w(\mathbf{x}_i) \propto p_{\mathbf{x}}(\mathbf{x}_i)/q_{\mathbf{x}}(\mathbf{x}_i)$. Assuming $0 < \mathbb{E}_{\mathbf{x}\sim Q_{\mathbf{x}}}\left[w(\mathbf{x})^2\right] < \infty$ and using the most common formulation for the ESS [11, 7], i.e. $\text{ESS} := (\sum_{i=1}^n \mathrm{w}_i)^2/(n\sum_{i=1}^n \mathrm{w}_i^2)$, it is possible to show that $\text{ESS} \xrightarrow{a.s.} \text{ESS}^* = 1/d_2(P_{\mathbf{x}}||Q_{\mathbf{x}})$ when $n \to \infty$. The quantity $d_2(P_{\mathbf{x}}||Q_{\mathbf{x}})$ equals $\exp\left[D_2(P_{\mathbf{x}}||Q_{\mathbf{x}})\right]$, where the quantity in the exponent is the Rényi Divergence of order 2 of $P_{\mathbf{x}}$ from $Q_{\mathbf{x}}$ [16, 2].

It is very interesting how Rényi Divergence naturally emerges when working with ESS. It is a keypoint to understand that, when calculating the Effective Sample Size, we are actually approximating a

[1]This is an abstract of [9] and our package to make feature selection can be found in `https://github.com/felipemaiapolo/infoselect`.

[2]See [12, 14] for an introduction.

quantity inversely proportional to the exponential of Rényi Divergence of order 2 of $P_{\mathbf{x}}$ from $Q_{\mathbf{x}}$. The result above essentially tells us that a lower ESS is related to more divergent distributions $P_{\mathbf{x}}$ and $Q_{\mathbf{x}}$. Another important result is that given two joint probability distributions $P_{\mathbf{x}_1,\mathbf{x}_2}$ and $Q_{\mathbf{x}_1,\mathbf{x}_2}$ over $\mathcal{X} \subseteq \mathbb{R}^d$, with joint probability density functions $p_{\mathbf{x}_1,\mathbf{x}_2}$ and $q_{\mathbf{x}_1,\mathbf{x}_2}$, we have that $D_2(P_{\mathbf{x}_1,\mathbf{x}_2}||Q_{\mathbf{x}_1,\mathbf{x}_2}) \geq D_2(P_{\mathbf{x}_1}||Q_{\mathbf{x}_1})$. That is, the Rényi Divergence (and its exponential) does not decrease with the number of variables (dimensions). The last result also tells us that ESS$^*$ non-increases with the number of dimensions, what indicates potential problems.

Going deeper, it is evident the dimensionality of the problem may play an important role considering the Theorem 3 proved by [2]. According to the result shown by [2], a larger ESS$^*$ leads to a tighter generalization bound of an importance weighted learning algorithm. In consequence, the rationale behind using ESS as an heuristic for diagnosis of Covariate Shift Adaptation becomes clearer. Furthermore, as long as ESS$^*$ is inversely connect with the number of dimensions, the generalization bounds get tighter when we discard some variables and everything else is held constant. Thus, it seems that performing a smart feature selection before Covariate Shift Adaptation by maintaining only the essential information about the labels[3] should be a good procedure, as long as we have a larger ESS$^*$, tighter generalization bounds and approximately the same potential performance for our models.

## 3 Variable Selection for Covariate Shift Adaptation

Here we propose a feature selection approach prior to covariate shift correction. Working with a good subset of features potentially enables a greater ESS and better generalization. Right after feature selection, the covariate shift adaptation is carried via importance weighting using off-the-shelf methods for density ratio estimation.

Theorem 1 from [13] supports the idea of using Sufficient Dimensionality Reduction (SDR) in order to make dimensionality reduction while having a solution for the problem of covariate shift adaptation in high-dimensions. We adapt that idea to make sense of feature selection prior to covariate shift adaptation.

**Variable Selection via Sufficient Dimensionality Reduction (SDR)**: Given a set of features $\mathbf{x}$ and a target variable y, the objective of SDR [4, 15] is to find a matrix $\boldsymbol{M} \in \mathbb{R}^{d\times d'}$, with $d' < d$ and $\boldsymbol{M}^\top \boldsymbol{M} = \boldsymbol{I}_{d'}$, such that $\mathrm{y} \perp\!\!\!\perp \mathbf{x} \mid \boldsymbol{M}^\top \mathbf{x}$. That is, the representation $\boldsymbol{M}^\top \mathbf{x}$ is sufficient for y. Usually, $\boldsymbol{M}$ is assumed to be dense, but we focus in the case where the matrix $\boldsymbol{M}$ is sparse and each column of it is given by zeros, except for one entry set as 1 to create a feature selector, as it is done in [4].

An interesting way to face the problem of Sufficient Dimensionality Reduction is using the concept of Mutual Information [15]. Recall that the Mutual Information between y and a random vector $\mathbf{x}'$, sampled from $Q_{\mathbf{x}',\mathrm{y}}$, with probability density function $q_{\mathbf{x}',\mathrm{y}}$, is given by $I(\mathrm{y};\mathbf{x}') = \mathbb{E}_{Q_{\mathbf{x}',\mathrm{y}}}\left[\log v(\mathbf{x}',\mathrm{y})\right]$, where $v = q_{\mathbf{x}',\mathrm{y}}/[q_{\mathbf{x}'}q_{\mathrm{y}}]$ is a density ratio. It is possible to show that if we consider a random vector $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ and a random variable y that have joint distribution $Q_{\mathbf{x},\mathrm{y}}$ and p.d.f. $q_{\mathbf{x},\mathrm{y}}$, then $I(\mathrm{y};\mathbf{x}) \geq I(\mathrm{y};\mathbf{x}_1)$ and $I(\mathrm{y};\mathbf{x}) = I(\mathrm{y};\mathbf{x}_1)$ iff $\mathrm{y} \perp\!\!\!\perp \mathbf{x}|\mathbf{x}_1$. Therefore, if we find a subset of features $\mathbf{x}_1$ from original set $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ that $I(\mathrm{y};\mathbf{x})$ is close to $I(\mathrm{y};\mathbf{x}_1)$, then working with this subset can be sufficient for our purposes.

Testing all the possibilities of subsets $\mathbf{x}_1$ quickly becomes impracticable with bigger number of dimensions even if distributions that generated the data are known. Thus, we adopt a greedy strategy called "*Forward Selection*" [5, 9]. A generally more precise but slower algorithm in this case is *Backward Elimination* [5]. In words, *Forward Selection* starts choosing the feature that has the largest estimated mutual information with the target variable and, at each subsequent step, we select the feature that marginally maximizes the estimated mutual information. We repeat the process until we reach a stop criterion, e.g. mutual information marginal growth is low.

An efficient alternative for estimating mutual information to perform feature selection for regression and classification problems is the use of Gaussian Mixture Models (GMMs) [3, 6]. Even though we consider GMM for mutual information estimation and feature selection, we acknowledge that there are other options that would fit our methodology.

[3]We might think the true function we are trying to learn depending only on some subset of features.

## References

[1] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning for differing training and test distributions. In *Proceedings of the 24th international conference on Machine learning*, pages 81–88, 2007.

[2] Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Advances in neural information processing systems*, pages 442–450, 2010.

[3] Emil Eirola, Amaury Lendasse, and Juha Karhunen. Variable selection for regression problems using gaussian mixture models to estimate mutual information. In *2014 International Joint Conference on Neural Networks (IJCNN)*, pages 1606–1613. IEEE, 2014.

[4] Kenji Fukumizu, Francis R Bach, and Michael I Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5(Jan):73–99, 2004.

[5] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[6] Tian Lan, Deniz Erdogmus, Umut Ozertem, and Yonghong Huang. Estimating mutual information using gaussian mixture model for feature ranking and selection. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 5034–5039. IEEE, 2006.

[7] Art B. Owen. *Monte Carlo theory, methods and examples*. ., 2013.

[8] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[9] Felipe Maia Polo and Renato Vicente. Covariate shift adaptation in high-dimensional and divergent distributions. *arXiv preprint arXiv:2010.01184*, 2020.

[10] Sashank Jakkam Reddi, Barnabas Poczos, and Alex Smola. Doubly robust covariate shift correction. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[11] Christian P Robert, George Casella, and George Casella. *Introducing monte carlo methods with r*, volume 18. Springer, 2010.

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

[13] Petar Stojanov, Mingming Gong, Jaime G Carbonell, and Kun Zhang. Low-dimensional density ratio estimation for covariate shift correction. *Proceedings of machine learning research*, 89:3449, 2019.

[14] Masashi Sugiyama and Motoaki Kawanabe. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.

[15] Taiji Suzuki and Masashi Sugiyama. Sufficient dimension reduction via squared-loss mutual information estimation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 804–811, 2010.

[16] Tim van Erven and Peter Harremoës. Rényi divergence and kullback-leibler divergence. *arXiv preprint arXiv:1206.2459*, 2012.

[17] Fulton Wang and Cynthia Rudin. Extreme dimension reduction for handling covariate shift. *arXiv preprint arXiv:1711.10938*, 2017.

## 4 Supplementary Material: Experiments

### 4.1 A Toy Model Experiment

We present a toy model in order to gain some intuition about the concepts presented. Assume there are two joint distributions of features and labels $P_\lambda$ and $Q$ with densities $p_\lambda$ and $q$, being the case that $Q$ describes the source/training population and that $P_\lambda$ describes the target/test population. Moreover, we assume we are facing the classical covariate shift problem, that is, $p_\lambda(y|\boldsymbol{x}) = q(y|\boldsymbol{x}) = p(y|\boldsymbol{x})$ but $p_\lambda(\boldsymbol{x}) \neq q(\boldsymbol{x})$, plus the fact that we cannot sample the labels from the test population. Finally, consider $q(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\mathbf{0}, \boldsymbol{I}_d)$ and $p_\lambda(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\lambda \cdot \mathbf{1}, \boldsymbol{I}_d)$, for $\lambda \neq 0$, with $d$ indicating the number of dimensions. Suppose $p(y|\boldsymbol{x}) = \mathcal{N}(y|100 \cdot x_1, 1)$, that is, $y$ depends on $\mathbf{x}$ only through its first coordinate $\mathrm{x}_1$. First we calculate $D_2(P_\lambda||Q)$ and ESS$^*$ as functions of $d$ and then simulate how the predictive power of a decision tree regressor deteriorates as ESS$^*$ decreases and $d$ increases. We train the trees by minimizing the empirical error weighted by the true weighting function $w$ in the training set, also imposing a minimum of 10 samples per leaf as a regularization strategy. We choose to work with decision trees since they are robust against irrelevant features, thus it is reasonable to expect that great part of performance deterioration is not due to noisy features.

The first step to calculate $D_2(P_\lambda||Q)$ is to calculate its exponential:

$$\begin{aligned}
d_2(P_\lambda||Q) &= \mathbb{E}_{\mathbf{x}\sim P_\lambda}\left[\frac{p_\lambda(\mathbf{x})}{q(\mathbf{x})}\right] \\
&= \mathbb{E}_{\mathbf{x}\sim P_\lambda}\left\{\frac{\exp[-\frac{1}{2}(\mathbf{x}-\lambda\mathbf{1})^\top(\mathbf{x}-\lambda\mathbf{1})]}{\exp[-\frac{1}{2}\mathbf{x}^\top\mathbf{x}]}\right\} \\
&= \exp\left(-\frac{d\lambda^2}{2}\right)\cdot\mathbb{E}_{\mathbf{x}\sim P_\lambda}\left[\exp\left(\lambda\sum_{j=1}^{d}\mathrm{x}_j\right)\right] \\
&= \exp(d\lambda^2)
\end{aligned}$$

The last equality is true since $\exp(\lambda\sum_{j=1}^{d}\mathrm{x}_j) \sim \text{LogNormal}(d\lambda^2, d\lambda^2)$. Then, $D_2(P_\lambda||Q) = d\lambda^2$ and ESS$^* = \exp(-d\lambda^2)$. Figure 1 depicts our results:

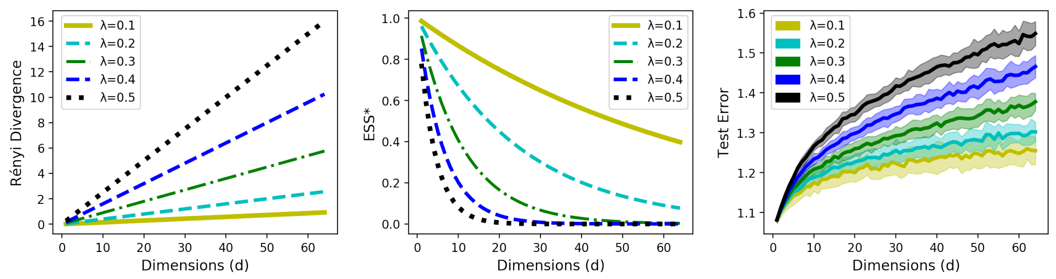

Figure 1: (i) We plot the Rényi Divergence of the target dist. $P_\lambda$ from the source dist. $Q$ as a function of the number of features. Both distributions are normal with the same covariance matrix but located $\sqrt{d\lambda^2}$ units apart from each other, i.e. the divergence also depends on $|\lambda|$; (ii) We plot the ESS$^*$ as a function of $d$ and also varying $\lambda$. As expected, ESS$^*$ exponentially decays in $d$ as long as the divergence is linearly related with $d$; (iii) In 50 simulations for each pair $(\lambda, d)$, we observe how decision trees' performances deteriorate due to low ESS.

The figure above depicts the behavior of Rényi Divergence and ESS$^*$ as functions of $d$. We also vary the value for $\lambda$. When $|\lambda|$ is bigger, the divergence between the source and target distributions also increases. An interesting fact is that the divergence between the distributions is not noticeable by only looking at marginals. Finally, to check how large $d$ affects performance of a regressor we, for each $d$, (i) sample 50 training and test sets, (ii) train the trees on the training set minimizing the weighted empirical error and (iii) assess the regressors on the test sets. The third plot of Figure 1, represents the average root-mean-square error $\pm$ standard deviations across samples. Clearly the regressor deteriorates as the divergence between domains grows larger.

### 4.2 Experiments with real data

For the following experiments, 10 regression tasks datasets with no missing values have been selected[4]. Each experiment consisted of (i) artificially causing covariate shift, (ii) estimating the weights, (iii) correcting the shift by the importance method, and finally (iv) assessing the performance of the predictors as well as the effective sample size. Besides regressions tasks, we have also performed classification tasks by binarizing the target variables using their medians as a threshold. We have used the same datasets for both regression and classification experiments to make performance comparisons easier. In each one of 10 datasets, we have performed the following pre-processing steps: (i) we kept up to 8,000 data points per dataset, (ii) augmented the number of features up to $40$ features where the new features are consisted by independent Gaussian noise and (iii) standardized each column in every dataset.

Right after the pre-processing steps, the following procedure have been used to create divergent training and test sets. For each one of the datasets, we have sampled a sequence of vectors uniformly from $[-1,1]^d$ and have projected the data points onto the subspace generated by each vector, resulting in only one feature $\boldsymbol{x}_i^{(j)}$ per sample $i$ for each subspace/simulation $j$. For each $\boldsymbol{x}_i^{(j)}$, we have calculated the score $s_{ij} = \Phi\big([x_i^{(j)} - \text{median}(\boldsymbol{x}^{(j)})]/\sigma_j\big)$, which is the probability that the data point $i$ from simulation $j$ is selected to the training set. According to that score, we randomly allocate each data point in either the training or test set in simulation $j$. The constant $\sigma_j$ is adjusted until the effective sample size is less than $0.01$. For each one of the training/test sets, we fit two decision trees: one in the training set and one in a subset of the test set. Then, we test both decision trees in the unused portion of the test set and compare their performance according to the Mean Squared Error for regression and Classification Error (1 - Accuracy) for classification. We have selected the 100 simulations in which decision trees trained in the test sets did best, relatively to the training set tree. We chose Decision Trees because they are fast to train and robust against irrelevant features. Thus, the noisy features added in the datasets are not likely to directly affect predictive power, but only making the effective sample size smaller.

To make feature selection, we have combined our selection algorithm with Gaussian Mixture Models to estimate the mutual information between a subset of features and the target variable[5]. Our stopping criteria used in our selection algorithm is that we should stop selecting features when the marginal improvement in the empirical mutual information in less than $1\%$ relative to the last level or when we select the first $15$ features. To estimate the weighting function for covariate shift correction, we have used the probabilistic classifier approach [1, 14] with Logistic Regression model with a quadratic polynomial expansion of the original features. Some of our benchmarks use Adaptive Weighting [12, 14], which is an attempt to make ESS higher. Adaptive Weighting elevates the raw weights to the power of a flattening parameter $\gamma \in [0,1]$, where $\gamma$ is chosen by Importance-Weighted Cross-Validation [14]. The optimal $\gamma$ gives a good balance between bias and variance for the risk estimation.

We work with four basic scenarios for training the models. Firstly, we use the whole set of features and no weighting method. In the second scenario we use the whole set of features and make use of 'true' weights $(1-s_{ij})/s_{ij}$. In the third, we use the whole set of features and estimate the weights using the probabilistic classifier approach. Finally, we use only selected features and estimate the weights using the probabilistic classifier approach. In the last three scenarios, we use both raw weights and their flattening version, i.e. we also use the Adaptive Weighting method, choosing the flattening parameters by a validation scheme.

Table 1 shows, for each and every one of the datasets employed, (i) the original number of features, (ii) the augmented number of features, (iii) the average number of selected features ($\pm$ std. deviation) for the regression experiments and (iv) for the classification experiments.

[4]From `www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html` and `https://archive.ics.uci.edu/ml/datasets.php`.

[5]More details on hyperparameter tuning can be found in the next section

| Dataset | Original | Augment. | Selected (Reg) | Selected (Class) |
|---|---|---|---|---|
| abalone | 7 | 40 | $3.93 \pm 1.26$ | $11.94 \pm 4.57$ |
| ailerons | 40 | 40 | $4.92 \pm 0.52$ | $3.82 \pm 0.68$ |
| bank32nh | 32 | 40 | $10.00 \pm 1.84$ | $13.19 \pm 1.69$ |
| cal housing | 8 | 40 | $5.53 \pm 1.14$ | $6.71 \pm 4.51$ |
| cpu act | 21 | 40 | $10.01 \pm 1.14$ | $2.61 \pm 0.79$ |
| delta ailerons | 5 | 40 | $3.92 \pm 0.42$ | $3.70 \pm 0.67$ |
| elevators | 18 | 40 | $7.96 \pm 0.79$ | $12.91 \pm 2.24$ |
| fried delve | 10 | 40 | $4.48 \pm 0.50$ | $5.00 \pm 0.00$ |
| puma32H | 32 | 40 | $1.98 \pm 0.14$ | $11.76 \pm 4.63$ |
| winequality | 11 | 40 | $9.56 \pm 1.09$ | $14.00 \pm 0.00$ |

Table 1: Average Numbers of features ($\pm$ std. deviation) - in this table we compare the numbers of original, augmented and selected (for regression and classification tasks) features. It is possible to note that, on average, we select small subsets of features, even smaller than the original set.

| | | All feat. | All feat. (True Weights) | | All feat. (Estimated Weights) | | Selected feat. (Estimated Weights) | |
|---|---|---|---|---|---|---|---|---|
| | Dataset | Unweighted | Raw | Adapt. | Raw | Adapt. | Raw | Adapt. |
| Regression | abalone | 1.00 | $1.37 \pm 0.21$ | $1.05 \pm 0.13$ | $1.22 \pm 0.17$ | $0.99 \pm 0.06$ | $\mathbf{0.91 \pm 0.05}$ | $\mathbf{0.91 \pm 0.05}$ |
| | ailerons | 1.00 | $1.02 \pm 0.13$ | $0.98 \pm 0.07$ | $0.98 \pm 0.10$ | $0.97 \pm 0.06$ | $\mathbf{0.87 \pm 0.10}$ | $0.89 \pm 0.13$ |
| | bank32nh | 1.00 | $1.27 \pm 0.12$ | $1.03 \pm 0.10$ | $1.19 \pm 0.09$ | $1.01 \pm 0.06$ | $0.97 \pm 0.05$ | $\mathbf{0.94 \pm 0.04}$ |
| | cal housing | 1.00 | $1.52 \pm 0.25$ | $1.03 \pm 0.16$ | $1.38 \pm 0.21$ | $0.98 \pm 0.10$ | $0.85 \pm 0.08$ | $\mathbf{0.84 \pm 0.07}$ |
| | cpu act | 1.00 | $0.55 \pm 0.62$ | $0.48 \pm 0.52$ | $0.58 \pm 0.64$ | $0.59 \pm 0.52$ | $\mathbf{0.15 \pm 0.22}$ | $0.22 \pm 0.30$ |
| | delta ailerons | 1.00 | $1.37 \pm 0.14$ | $1.05 \pm 0.12$ | $1.26 \pm 0.10$ | $1.00 \pm 0.04$ | $\mathbf{0.91 \pm 0.03}$ | $\mathbf{0.91 \pm 0.04}$ |
| | elevators | 1.00 | $1.09 \pm 0.16$ | $0.97 \pm 0.09$ | $1.04 \pm 0.14$ | $0.98 \pm 0.08$ | $0.84 \pm 0.15$ | $\mathbf{0.83 \pm 0.11}$ |
| | fried delve | 1.00 | $1.56 \pm 0.20$ | $1.09 \pm 0.12$ | $1.39 \pm 0.12$ | $1.02 \pm 0.06$ | $\mathbf{0.88 \pm 0.09}$ | $\mathbf{0.88 \pm 0.09}$ |
| | puma32H | **1.00** | $2.11 \pm 0.99$ | $1.07 \pm 0.14$ | $1.45 \pm 0.19$ | $1.02 \pm 0.06$ | $1.02 \pm 1.07$ | $1.02 \pm 1.06$ |
| | winequality | 1.00 | $1.31 \pm 0.12$ | $1.06 \pm 0.10$ | $1.23 \pm 0.09$ | $1.02 \pm 0.07$ | $0.95 \pm 0.04$ | $\mathbf{0.94 \pm 0.03}$ |
| Classification | abalone | 1.00 | $1.24 \pm 0.15$ | $1.02 \pm 0.16$ | $1.16 \pm 0.14$ | $0.97 \pm 0.11$ | $1.00 \pm 0.12$ | $\mathbf{0.92 \pm 0.10}$ |
| | ailerons | 1.00 | $1.03 \pm 0.22$ | $0.93 \pm 0.16$ | $1.00 \pm 0.17$ | $0.91 \pm 0.14$ | $\mathbf{0.84 \pm 0.13}$ | $0.86 \pm 0.13$ |
| | bank32nh | 1.00 | $1.22 \pm 0.10$ | $1.04 \pm 0.10$ | $1.17 \pm 0.09$ | $1.00 \pm 0.07$ | $0.97 \pm 0.07$ | $\mathbf{0.94 \pm 0.05}$ |
| | cal housing | 1.00 | $1.39 \pm 0.20$ | $1.02 \pm 0.15$ | $1.32 \pm 0.17$ | $0.97 \pm 0.11$ | $0.90 \pm 0.17$ | $\mathbf{0.88 \pm 0.16}$ |
| | cpu act | 1.00 | $1.07 \pm 0.13$ | $\mathbf{0.95 \pm 0.10}$ | $1.03 \pm 0.12$ | $0.97 \pm 0.11$ | $0.98 \pm 0.12$ | $0.97 \pm 0.12$ |
| | delta ailerons | 1.00 | $1.32 \pm 0.29$ | $0.94 \pm 0.13$ | $1.21 \pm 0.22$ | $0.92 \pm 0.11$ | $\mathbf{0.83 \pm 0.09}$ | $\mathbf{0.83 \pm 0.08}$ |
| | elevators | 1.00 | $1.06 \pm 0.13$ | $0.97 \pm 0.10$ | $1.03 \pm 0.12$ | $0.95 \pm 0.09$ | $\mathbf{0.88 \pm 0.11}$ | $0.89 \pm 0.09$ |
| | fried delve | 1.00 | $1.31 \pm 0.16$ | $1.04 \pm 0.10$ | $1.22 \pm 0.13$ | $1.02 \pm 0.09$ | $0.83 \pm 0.05$ | $\mathbf{0.82 \pm 0.05}$ |
| | puma32H | **1.00** | $1.65 \pm 0.55$ | $1.01 \pm 0.10$ | $1.19 \pm 0.14$ | $1.01 \pm 0.08$ | $1.05 \pm 0.38$ | $1.02 \pm 0.35$ |
| | winequality | 1.00 | $1.16 \pm 0.10$ | $1.02 \pm 0.11$ | $1.11 \pm 0.09$ | $1.00 \pm 0.09$ | $1.03 \pm 0.09$ | $\mathbf{0.97 \pm 0.07}$ |

Table 2: Average Test Errors ($\pm$ std. deviation) - here we compared the predictive performance of decision trees in the test set of 100 different simulations for each dataset. We have four basic scenarios: (i) whole set of features and no weighting method; (ii) whole set of features and use of 'true' weights; (iii) whole set of features and estimated weights; (iv) selected features and estimated weights. In the last three scenarios, we use both raw weights and their flatter version ("Adapt."). The numbers reported are the (relative) MSE and classification error averages and their std. deviations.

From Table 1, it is possible to note that, on average, we select small subsets of features, even smaller than the original set. The small number of selected features for some datasets is probably due to the nature of the selection method, allowing the discarding of highly redundant features even though they are relevant separately. It seems that using Gaussian Mixture Models to make feature selection usually works better for regression compared to classification tasks. Among other factors, that can be due to the loss of information when binarizing the target variable in some cases.

In Table 2, we see predictors' average test errors ($\pm$ std. deviation), with all errors relative to the first scenario. From Table 2, it is noticeable that our feature selection approach and posterior weighting, combined or not with the Adaptive Weighting method, systematically outperforms all the other benchmarks, especially the pure weighting method when the whole set of features is used. Even the benchmarks that used true weights are often beaten by large margins. That suggests that the degradation in the model performances is mainly due to low effective sample size instead of a difficulty in estimating the weighting function. Let us directly evaluate how feature selection affects ESS looking at Figure 2.

In Figure 2, one can see the distribution of Effective Sample Sizes in all the weighted approaches, calculated in the entire set of experiments. It is possible to notice how small the ESSs can be by adopting the pure weighting strategy. Using the Adaptive Weighting method, without prior feature selection, yields very high ESS in exchange for higher biases. The feature selection itself allows higher ESSs and, when combined with Adaptive weighting, it yields the highest ESSs with less pronounced biases.

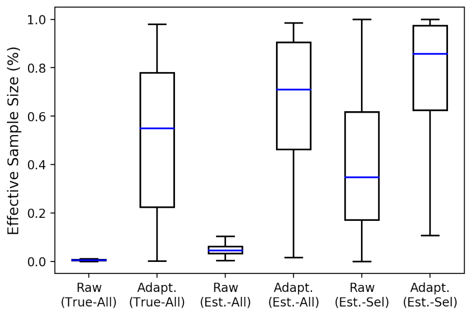

Figure 2: Effective Sample Size distributions across all experiments. Notice higher ESSs can be achieved by a prior feature selection stage. We use both raw weights and their flatter version ("Adapt.") in a combination of scenarios which includes all/selected features and true/estimated weights.

#### 4.2.1 Hyperparameters

In the experiments section, we tune four hyperparameters: (i) $l1$ regularization parameter used to train the logistic regression model when estimating $w$, (ii) the minimum number of samples per leaf in each regression/classification tree, (iii) the flattening parameter $\gamma$ used to make the weighting function flatter and (iv) number of GMM components. We use the Scikit-Learn [8] implementations to train the logistic regressions, regression/classification trees and GMMs. First, we choose the $l1$ logistic regression regularization parameter $C$ from values in $[10^{-4}, 5]$, in order to minimize the log loss in a holdout dataset. Second, we choose the minimum number of samples per leaf in each regression/classification tree from values in $[5, 15, 25, 40, 50]$, in order to minimize the mean squared error or classification error within a 2-fold cross-validation procedure. Third, we choose $\gamma$ from $[.1, .2, .3, .4, .5, .6, .7, .8, .9]$ using Importance-Weighted Validation [14] in a holdout dataset. Finally, we maximize the log-likelihood in a holdout dataset to choose the number of GMM components, varying the possible number of components within the list $[1, ..., 15]$.