Versatile Decision Trees for Learning over Multiple Contexts

Reem Al-Otaibi[1,2], Ricardo B.C. Prudêncio[3], Meelis Kull[1], and Peter Flach[1]

[1]Intelligent System Laboratory, Computer Science, University of Bristol
Bristol, United Kingdom
{ra12404, meelis.kull, peter.flach}@bristol.ac.uk
[2]King Abdulaziz University, Saudi Arabia
[3]Informatics Center, Universidade Federal de Pernambuco, Brazil
rbcp@cin.ufpe.br

Abstract. Discriminative models for classification assume that training and deployment data are drawn from the same distribution. The performance of these models can vary significantly when they are learned and deployed in different contexts with different data distributions. In the literature, this phenomenon is called dataset shift. In this paper, we address several important issues in the dataset shift problem. First, how can we automatically detect that there is a significant difference between training and deployment data to take action or adjust the model appropriately? Secondly, different shifts can occur in real applications (e.g., linear and non-linear), which require the use of diverse solutions. Thirdly, how should we combine the original model of the training data with other models to achieve better performance? This work offers two main contributions towards these issues. We propose a Versatile Model that is rich enough to handle different kinds of shift without making strong assumptions such as linearity, and furthermore does not require labelled data to identify the data shift at deployment. Empirical results on both synthetic shift and real datasets shift show strong performance gains by achieved the proposed model.

Keywords: Versatile Model; Decision Trees; Dataset Shift; Percentile; Kolmogorov-Smirnov Test

1 Introduction

Supervised machine learning is typically concerned with learning a model using training data and applying this model to new test data. An implicit assumption made for successfully deploying a model is that both training and test data follow the same distribution. However, the distribution of the attributes can change, especially when the training data is gathered in one context, but the model is deployed in a different context (e.g., the training data is collected in one country but the predictions are required for another country). The presence of such *dataset shifts* can harm the performance of a learned model. Different kinds of dataset shift have been investigated in the literature [10]. In this work we focus on shifts in continuous attributes caused by hidden transformations from context to another. For instance, a diagnostic test may have different resolutions when produced by different laboratories, or the average temperature may change from city to city. In such cases, the distribution of one or more of the covariates in X changes. This problem is referred as a covariate observation shift [7].

We address this problem in two steps. In the first step, we build Decision Trees (DTs) using percentiles for each attribute to deal with covariate observation shifts. In this proposal, if a certain percentage of training data reaches a child node after applying a decision test, the decision thresholds in deployment are redefined in order to preserve the same percentage (60%) of deployment instances in that node. In the original learned DT, the learned threshold in a decision node corresponds to the 60^{th} percentile of the *training* data. The updated threshold in deployment will be the 60^{th} percentile of the *deployment* data.

The percentile approach assumes that the shift is caused by a monotonic function preserving the ordering of attribute values but ignoring the scale. For some shifts it may be more appropriate to assume a transformation from one linear scale to another. We therefore develop a more general and versatile DT that can choose between different strategies (percentiles, linear shifts or no shift) to update the DT thresholds for each deployment context, according to the shifts observed in the data.

The rest of the paper is organised as follows. Section 2 presents the dataset shift problem and the existing approaches addressing it. In Section 3 we introduce the use of percentiles and the versatile model proposed in our work. Section 4 presents the experiments performed to evaluate the versatile model on both synthetic and non-synthetic dataset shifts, and Section 5 concludes the paper.

2 Dataset Shift

We start by making a distinction between the *training* and *deployment* contexts. In the training context, a set of labelled instances is available for learning a model. The deployment context is where the learned model is actually used for predictions. These contexts are often different in some non-trivial way. For instance, a model may be built using training data collected in a certain period of time and in a particular country, and deployed to data in a future time and/or in a different country. A model built in a training context may fail in a deployment context due to different reasons: in the current paper we focus on performance degradation caused by *dataset shifts* across contexts.

A simple solution to deal with shifts would be to train a new classifier for each new deployment context. However, if there are not enough available labelled instances in the new context, training a new model specific for that context is then unfeasible as the model would not sufficiently generalise. Alternative solutions have to be applied to reuse or adapt existing models, which will depend on the kind of shift observed.

Shifts can occur in the input attributes, in the output or both. Dataset shift happens when training and deployment joint distributions are different [10], i.e.:

$$P_{tr}(Y,X) \neq P_{dep}(Y,X) \tag{1}$$

A shift can occur in the output, i.e., $P_{tr}(Y) \neq P_{dep}(Y)$, while the conditional probability distribution remains the same $P_{tr}(X|Y) = P_{dep}(X|Y)$. This is referred in the literature as the prior probability shift and can be addressed in different ways (e.g., [5]).

In our work we are mainly concerned with situations where the marginal distribution of a covariate changes across contexts, i.e.: $P_{tr}(X) \neq P_{dep}(X)$. Given a change in the marginal distribution, we can further define two different kinds of shifts depending on whether the conditional distribution of the target also changes between training and deployment. In the first case, the marginal distribution of X changes, while the conditional probability of the target Y given X remains the same:

$$\begin{array}{l} P_{tr}(X) \neq P_{dep}(X) \\ P_{tr}(Y|X) = P_{dep}(Y|X) \end{array} \tag{2}$$

For instance, the smoking habits of a population may change over time due to public initiatives but the probability of lung cancer given smoking is expected to remain the same [12]. In the same problem, a labelled training set may be collected biased to patients with bad smoking habits. Again, the marginal distribution in deployment may be different from training while the conditional probability is the same. The above shift is referred in the literature by different terms such as simple covariate shift [12] or sample selection bias [15]. A common solution to deal with simple covariate shifts is to modify the training data distribution by considering the deployment data distribution. A new model can then be learned using the shift-corrected training data distribution. This strategy is adopted by different authors using importance sampling which corrects the training distribution using instance weights proportional to $P_{dep}(X)/P_{tr}(X)$. Examples of such solutions include Integrated Optimisation Problem IOP [3], Kernel Mean Matching [6], Importance Weighted Cross Validation IWCV [14] and others.

In this paper we focus on the second kind of shift in which both the marginal and the conditional distributions can change across contexts, i.e.:

$$\begin{array}{l} P_{tr}(X) \neq P_{dep}(X) \\ P_{tr}(Y|X) \neq P_{dep}(Y|X) \end{array} \tag{3}$$

This is a more difficult situation that can be hard to solve and requires additional assumptions. A suitable assumption in many situations is that there is a hidden transformation of the covariates $\Phi(X)$ for which the conditional probability is unchanged across contexts, i.e.:

$$\begin{array}{l} P_{tr}(X) \neq P_{dep}(X) \\ P_{tr}(Y|X) = P_{dep}(Y|\Phi(X)) \end{array} \tag{4}$$

This is defined in [7] as a *covariate observation shift*. For instance, in prostate cancer detection, shifts can be observed in data from different laboratories due to differences in their equipments and resolution of diagnostic tests [9]. A mapping between attributes can be performed to correct the existing differences in data [9]. As another example [7], suppose that in an image recognition problem, pictures are taken by a camera with two different colour adjustments settings, thus representing two different contexts. This can result in a shift in the covariates. The conditional probability, however, may be the same given an invariant hidden raw camera representation. Finally, a sensor used to detect an event may degrade over time. Such degradation can be seen as a transformation function in the sensor outputs that causes a covariate observation shift.

Previous authors dealt with covariate observation shifts by finding a transformation function Φ to correct the deployment data [1]. Once transformed or 'unshifted' using

Φ, the deployment data is given as input to the model learned in the training context. Finding a linear transformation is a natural choice in this approach. In [1], for instance, the authors adopted Stochastic Hill Climbing to find the best linear transformation to apply in the given deployment data. In that work, labelled deployment instances are required in order to evaluate the suitability of a candidate linear map. The parameters of the linear transformation are then optimised to maximise the accuracy obtained by the learned model on the labelled deployment instances (once transformed). A similar idea was proposed in [9], using genetic programming techniques to find more complex transformation functions (both linear and non-linear). As [1], it requires that labelled instances are available in the deployment data to evaluate the adequacy of the transformation functions.

In summary, we emphasise that it can be difficult to recognise or distinguish between the different kinds of shifts that may occur in a dataset. It can be simple in some cases to identify a shift in the covariates, relying on a sufficient number of unlabelled instances in the deployment context. On the other hand, verifying a shift in the conditional probabilities $P(Y|X)$ is not possible if there are only unlabelled instances in deployment or may be unreliable if the number of labelled instances in deployment is low. Additionally, suppose that we have evidence that a change is caused by a transformation in a covariate. Trying to detect a linear transformation to apply in the deployment data may be counter-productive if the true transformation is non-linear instead. Finally, applying a shift-aware method in a deployment context that did not actually change compared to the training context may be detrimental as well. These considerations motivated our proposal of a more sophisticated approach that can adapt to different kinds of dataset shifts under different assumptions.

3 Versatile Decision Trees

In this work, we propose different strategies to build Decision Trees (DTs) in the presence of covariate observation shifts. We make two main contributions. First, we propose a novel approach to build DTs based on percentiles (see Sections 3.1 and 3.2). The basic idea is to learn a conventional DT and then to replace the internal decision thresholds by percentiles, which can deal with monotonic shifts. Secondly, we propose a more general Versatile Model (VM) that deploys different strategies (including the percentiles) to update the DT thresholds for each deployment context, according to the shifts observed in the data (see Section 3.3). The shifts are identified by applying a non-parametric statistical test.

3.1 Constructing Splits Using Percentiles

We consider an example using the diabetes dataset from the UCI repository, which has 8 input attributes and 768 instances. Suppose we train a decision stump and the discriminative attribute is the Plasma glucose concentration attribute, which is a numerical attribute. Suppose the decision threshold is 127.5, meaning that any patient with plasma concentration above 127.5 will be classified as diabetic, otherwise classified as non-diabetic as seen in Figure 1 (left). If there is no shift in the attribute from training to

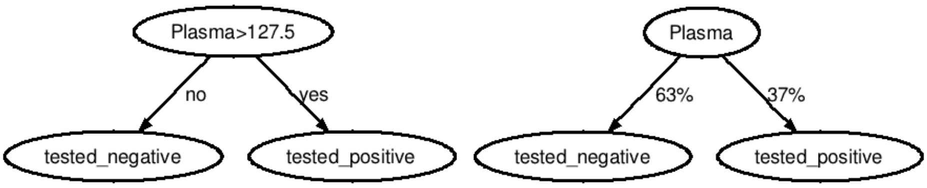


Fig. 1: Two types of models; on the left is the model using a fixed threshold while on the right is the model using percentiles. For each deployment context, the decision tree is deployed in such a way that the deployment instances are split to the leaves in the same percentile amounts of 63% and 37%.

deployment, the decision node can be directly applied, i.e., the threshold 127.5 is maintained to split data in deployment. However, if the attribute is shifted in deployment, the original threshold may not work well.

In the current work, we propose to adopt the percentiles[1] of continuous attributes to update the decision thresholds for each deployment context. Back to the example, instead of interpreting the data split in an absolute sense, we will interpret it in terms of ranks: 37% of the training examples with the *highest* values of Plasma reach the right child, while 63% of the training examples with the *lowest* values of Plasma in turn reach the left child (see the right side of Figure 1). We can say that the data was split at the 63^{th} percentile in training. Given a batch set of instances in deployment to classify, the DT can apply the same split rule: the 37% of the examples in deployment with the highest values of Plasma are associated to the right child, while 63% of the examples with the lowest values of Plasma in deployment are associated to the left child. The decision threshold in deployment is updated in such a way that the percentage of instances split to each child is maintained. In this proposal, it is assumed that the shift is caused by a *monotonic* transformation Φ. Such functions when applied to an input attribute preserve the order of its original values. Different from the previous work [1] the transformation function in the versatile DT is not explicitly estimated, but it is implicitly treated by deploying the percentiles.

Formally, let $\mathscr{L} = \{c_1, \ldots, c_L\}$ be the set of class labels in a problem. Let th_{tr} be the threshold value applied on a numerical attribute X in a decision node. In the previous example $th_{tr} = 127.5$ for the Plasma attribute. Let $n_{left}^{c_l}$ be the number of training instances belonging to class c_l that are associated to the left child node after applying the decision test, i.e., for which $X \leq th_{tr}$. The total number of instances n_{left} associated to this node is:

$$n_{left} = \sum_{c_l \in \mathscr{L}} n_{left}^{c_l} \tag{5}$$

Let $R_{tr}(th_{tr}) = 100 * n_{left}/n$ be the percentage of training instances in the left node, where n is the total number of training instances. Then, th_{tr} is the percentile associated to $R_{tr}(th_{tr})$ for the attribute X. In the above example: $R_{th}(127.5) = 63\%$ and th_{tr} is the 63^{th} percentile of Plasma in the training data. Then the threshold adopted in deployment

[1] Percentile is the value below which a given percentage of observations in a group is observed.

is defined as:

$$th_{dep} = R_{dep}^{-1}(R_{tr}(th_{tr})) \tag{6}$$

In the above equation, the threshold th_{dep} is the attribute value in deployment that, once adopted to split the deployment data, maintains the percentage of instances in each child node: $R_{dep}(th_{dep}) = R_{tr}(th_{tr})$.

3.2 Adapting for Output Shifts

The percentile rule can be adapted to additionally deal with shifts in the class distribution across contexts. Figure 2 illustrates a situation where the prior probability of the positive class was 0.5 in training and then shifted to 0.6 in deployment. In Figure 2(a) we observe a certain number of positives and negatives internally in each child node, which is used to derive the percentiles. If a shift is expected in the target, the percentage of instances expected in deployment for each child node may change as well. For instance, a higher percentage of instances may be observed in the right node in deployment because the probability of positives has increased and the proportion of positives related to negatives in this node is high. In our work, we estimate the percentage of instances in each child node according to the class ratios between training and deployment.

Let $P_{tr}^{c_l}$ and $P_{dep}^{c_l}$ be the probability of class c_l, respectively in training and deployment. $P_{dep}^{c_l}$ can be estimated using available labelled data in deployment or just provided as input. There is a prior shift related to this class label when $P_{tr}^{c_l} \neq P_{dep}^{c_l}$. For each instance belonging to c_l observed in training we expect to observed $P_{dep}^{c_l}/P_{tr}^{c_l}$ instances of c_l in deployment. The number of instances associated to the left child node in deploy-

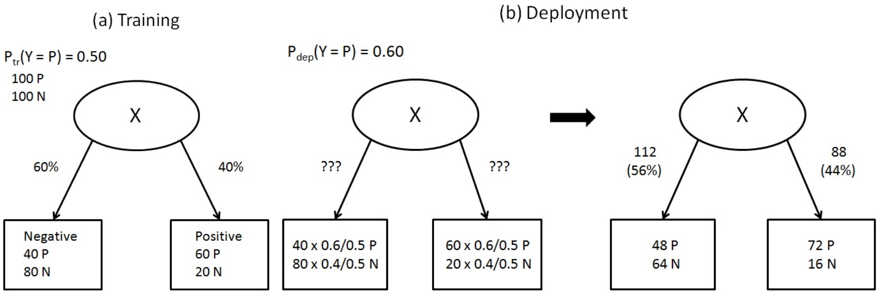


Fig. 2: Example of DT with percentiles when a shift is identified in the class distribution. Part (a) illustrates the percentiles of each leaf for the training context, with prior probability equal to 0.5. Part (b) illustrates the correction of the percentiles for a new deployment context where the prior probability is 0.6. The correction of performed using the ratios of $0.6/0.5$ and $0.4/0.5$ respectively for the positive and negative instances (left side - (b)). Corrected number of instances expected at each leaf resulted in new estimated percentiles (right side - (b))

.

ment is then estimated by the following equation:

$$\hat{n}_{left} = \sum_{c_l \in \mathscr{L}} n^{c_l}_{left} \frac{P^{c_l}_{dep}}{P^{c_l}_{tr}} \tag{7}$$

The percentile is then computed using the corresponding percentage: $R_{tr}(th_{tr}) = 100 * \hat{n}_{left}/n$. In Figure 2(b), the class ratios of $0.6/0.5$ and $0.4/0.5$ are respectively adopted to correct the number of positive and negative instances in each node. In the left node, for instance, the expected number of positive and negative instances is respectively 48 and 64, resulting in 112 instances. The percentage to be adopted in deployment is now 56%, instead of 60% if no correction is performed. The 56^{th} percentile in the deployment data is then adopted as the decision threshold.

3.3 Versatile Model for Decision Trees

By adopting percentiles in the DT, we are assuming a monotonic transformation Φ across contexts. In this sense, our work is more general compared to the previous work [1] that assumes a linear transformation. Monotonic shifts can not only cover the linear case but also a broad range of non-linear monotonic transformations (e.g., piecewise linear transformations). Even the case where there is no shift can be seen as a monotonic transformation when Φ is the identity function. Despite this generality, the use of percentiles has limitations too. First, percentile estimates (either in training or deployment) can be inaccurate when there is few or sparse data for estimation. Also, it may be worth trying alternative methods if the assumptions made by these methods about the context shifts are actually met. For instance, if we expect the shift to be linear we might be better off fitting an explicit linear transformation between training and deployment.

In this section, we propose a versatile decision tree model that employs different strategies to choose the decision threshold according to the shift observed in deployment. Algorithm 1 presents the proposed versatile model, which receives as input the original threshold applied on an attribute, the training and the deployment data of that attribute and returns the appropriate threshold to adopt in deployment. This *versatile model* (VM) can be described in three steps:

1. Initially a statistical test is applied to verify whether the distribution of the attribute differs between training and deployment. In this step, we aim to avoid dealing with shifts when they do not really exist, which could lead to overfitting. In our implementation, the non-parametric Kolgomorov-Smirnoff (KS) test was adopted[2]. If there is no shift in the attribute, the versatile DT is applied using the original thresholds learned in the training context, i.e. $th_{dep} = th_{tr}$.
2. If a shift is detected by the previous test, a linear transformation is fitted and applied to the attribute in deployment, aiming to correct a potential linear shift. In our

[2] We employed the KS test on the values of the attribute: training X_{tr} and deployment X_{dep}. The KS test tests the null hypothesis that the empirical cumulative distribution functions of X_{tr} and X_{dep} are identical against the alternative hypothesis that the two distributions are different.

Algorithm 1 Versatile Model Threshold Selection Algorithm

```
Input: training attribute vector $X_{tr} = (x_1, \ldots, x_n)$ with $n$ the number of training instances (i.e., a column of the data matrix),; deployment attribute vector $X_{dep} = (x'_1, \ldots, x'_m)$; decision threshold in training $th_{tr}$ for attribute $X_{tr}$.
Output: deployment decision threshold $th_{dep}$.
/* Test for no shift. Null hypothesis $H_0 : F(X_{tr}) = F(X_{dep})$ */
$p_{value}$=Kolmogorov-Smirnov($X_{tr}$,$X_{dep}$)
if $p_{value}$ <0.05 then
  /* Reject $H_0$, $X_{dep}$ is shifted. Try a linear transformation */
  ($X^u_{dep}, \alpha, \beta$)= Linear Transformation($X_{tr}$,$X_{dep}$)
  /* Test corrected shift. Null hypothesis $H_0 : F(X_{tr}) = F(X^u_{dep})$ * /
  $p_{value}$=Kolmogorov-Smirnov($X_{tr}$,$X^u_{dep}$)
  if $p_{value}$ <0.05 then
    /* Reject $H_0$, $X^u_{dep}$ is still shifted. Use the percentile. */
    $th_{dep} = R^{-1}_{dep}(R_{tr}(th_{tr}))$
  else
    /* Accept $H_0$, $X^u_{dep}$ is not shifted. Use the linearly corrected threshold */
    $th_{dep} = \alpha \cdot th_{tr} + \beta$
  end if
else
  /* Accept $H_0$, $X_{dep}$ is not shifted. Use training threshold. */
  $th_{dep} = th_{tr}$
end if
Return $th_{dep}$
```

implementation, α and β parameters were estimated based on the change in mean and standard deviation of the attribute in training and deployment (see Algorithm 2). We then apply the KS test again to compare the distribution of the transformed attribute in deployment and the attribute in the training data. If no shift is observed now, we assume that the linear transformation applied was adequate. The versatile DT is then deployed with a threshold $th_{dep} = \alpha \cdot th_{tr} + \beta$.

3. Finally, if the second test indicates that there is still a shift in the attribute (i.e., the shift was not corrected using the linear transformation), then the percentiles are deployed, assuming a non-linear monotonic shift. In this case the adopted threshold is: $th_{dep} = R^{-1}_{dep}(R_{tr}(th_{tr}))$.

4 Experimental Results

The VM combines 3 strategies for defining the DT thresholds in deployment: original thresholds, linear transformations, and monotonic transformations using percentiles. In the experiments, each strategy was individually compared to the VM, respectively named as Original Model (OM), (α, β) and Perc. Additionally, (α, β) and the Percentile methods were combined with the KS test, referred in the experiments as KS+(α, β) and KS+Perc, respectively. In the former, linear transformation is applied to all shifted

Algorithm 2 Linear Transformation

```
Input: training attribute vector $X_{tr} = (x_1, \ldots, x_n)$; deployment attribute vector $X_{dep} = (x'_1, \ldots, x'_m)$.
Output: ‘Unshifted’ deployment attribute vector $X^u_{dep}$ and corresponding parameters $\alpha, \beta$.
/* Estimate the mean and standard deviation of X in training and deployment */
$\mu_{tr} = \frac{1}{n}\sum_{i=1}^{n} x_i$ , $\sigma_{tr} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu_{tr})^2}$
$\mu_{dep} = \frac{1}{m}\sum_{i=1}^{m} x'_i$, $\sigma_{dep} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x'_i - \mu_{dep})^2}$
/* Estimate $\alpha$ and $\beta$ considering that: $\sigma_{dep} = \alpha\sigma_{tr}$ and $\mu_{dep} = \alpha\mu_{tr} + \beta$ */
$\alpha = \frac{\sigma dep}{\sigma_{tr}}$
$\beta = \mu_{dep} - \alpha \cdot \mu_{tr}$
/* Produce unshifted deployment data $X^u_{dep}$ according to $\alpha$ and $\beta$ */
$X^u_{dep} = \emptyset$
for $i = 1$ to $m$ do
    $x^u_i = \frac{(x'_i - \beta)}{\alpha}$
    $X^u_{dep} = X^u_{dep} \cup x^u_i$
end for
Return $X^u_{dep}, \alpha, \beta$
```

attributes, whereas, in the latter, Percentiles are utilised. In both approaches, the original model was applied if there is no shift detected by the KS test.

The first set of experiment applies synthetic shifts to UCI datasets to analyse the performance of the shift detection approach adopted by the VM. We inject two types of shifts into the deployment data to test the VM: a non-linear monotonic transformation and linear shifts with different degrees (see Sections 4.1 and 4.2). In Section 4.3 we report on experiments with actual context changes occurring in real-world datasets.

4.1 Generating Synthetic Shifts

In these experiments, linear transformations were applied to numerical attributes in order to simulate shifts between two contexts. Two parameters, α and β, were adopted in each simulation to perform the linear transformation $X_{dep} = \alpha \cdot X_{tr} + \beta$. Let μ_{tr} and σ_{tr} be the mean and standard deviation of attribute X in training. When X is shifted using the parameters α and β, the mean and standard deviation of the transformed variable become

$$\mu_{dep} = \alpha \cdot \mu_{tr} + \beta$$
$$\sigma_{dep} = \alpha \cdot \sigma_{tr}$$

It is useful to re-parametrise α and β as follows.

$$\alpha = 2^{\varphi}$$
$$\beta = (1 - 2^{\varphi}) \cdot \mu_{tr} + \gamma \cdot \sigma_{tr}$$

If φ is negative the attribute values are compressed across contexts, and if φ is positive the values are stretched. The mean is shifted by γ times the standard deviation in train-

Table 1: Values used in the experiments for φ and γ in order to generate the synthetic linear shifts.

φ	γ	Effect
0	0	unshifted data (original)
0	1	μ_{dep} shifted to right
0	-1	μ_{dep} shifted to left
1	0	stretch data
-1	0	compress data
1	1	μ_{dep} shifted to right and stretch the data
1	-1	μ_{dep} shifted to left and stretch the data
-1	1	μ_{dep} shifted to right and compress the data
-1	-1	μ_{dep} shifted to left and compress the data

ing: $\mu_{dep} = \mu_{tr} + \gamma \cdot \sigma_{tr}$. Table 1 shows the values used in the experiments for φ and γ.

To create non-linear monotonic shifts we use the following transformation:

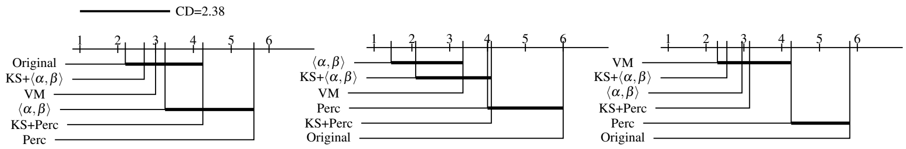
$$X_{dep} = \sigma_{tr} \cdot \left(\frac{X_{tr} - \mu_{tr}}{\sigma_{tr}} \right)^3 + \mu_{tr} \tag{8}$$

We use a cubic rather than a square transformation to ensure monotonicity. In order to preserve the mean and standard deviation of the data we first convert the attribute values to z-scores, apply the cubic transformation and then restore the mean and standard deviation.

4.2 Results of the Synthetic Shifts

We selected 10 datasets from UCI [8] and KEEL [2] with all numerical (real-valued as well as integer-valued) attributes. Each dataset was randomly partitioned into 5 folds. Using 4 folds for training and the 5^{th} fold for deployment, after shifting according to each set of parameters in Table 1. The same shift is applied to all attributes in each dataset. Results are averaged over 5 cross-validation runs for each dataset. Table 2 reports the average accuracy of 5 different runs for all used methods in 4 cases: unshifted, linear shift, non-linear and mixture shift data. Performance of these methods applied to linear shifts is the average of 8 degrees of linear shift as reported in Table 1. We conducted the Friedman test based on the average ranks for all datasets in order to verify whether the differences between algorithms are statistically significant [4]. At significance level 0.05 the Friedman test gives significance for all experiments except the non-linear shifts, so we show critical difference diagrams based on the Nemenyi post-hoc test for the former in Figure 3. We proceed to discuss the results of each experiment.

Unshifted data Unsurprisingly, the original model is the best when there is no shift from training to deployment, but the CD diagram demonstrates that the Versatile Model is not significantly worse. Percentiles don't work well in this case, confirming the need for a multi-strategy approach.



(a) unshifted data (b) linear shift (c) mixture shift

Fig. 3: Critical Difference diagrams using pairwise comparisons for those experiments where the Friedman test gives significance at 0.05.

Linear shifts Estimating a linear shift is the right thing to do here so it is not surprising that $\langle\alpha,\beta\rangle$ performs strongest, with KS+$\langle\alpha,\beta\rangle$ trailing slightly behind as the KS test may sometimes fail to detect the shift. The original model is significantly outperformed by all context-sensitive models except the percentiles. The Versatile Model is a good representative of the context-sensitive models.

Non-linear shift Here the Versatile Model outperforms all other methods in terms of the average ranks, but not significantly. Notice that, while the original model performs worst, there are 2 datasets where the original model performs best: in these datasets many attribute values are in the range $[-1,1]$ where the cubic transformation has less effect.

Mixture shift The aim of this experiment was to test how well the Versatile Model deals with a mixture of different shifts: one-third of the attributes was shifted linearly, one-third non-linearly, and one-third remained unchanged. The results demonstrate that the Versatile Model derives a clear advantage from the ability of being able to distinguish between these different kinds of shift and adapt its strategy.

4.3 Results on Non-Synthetic Shifts

The aim of this experiment is to evaluate the Versatile Model on real dataset shift and compared it the with state-of-art covariate shift solvers Integrated Optimisation Problem (IOP) [3] and Kernel Mean Matching (KMM) [6]. IOP and KMM algorithms were retrieved from [11] and run using default parameters.

Diabetes Our first benchmark is a dataset of 4 different ethnic groups of diabetes patients [13]. The original dataset consists of 47 attributes and 101 766 instances. Each instance corresponds to a unique patient diagnosed with diabetes. The features describe the diabetic encounters such as diagnoses, medications, and number of visits in the year preceding the encounter. We rank features using information gain ratio then we select the best 8 numerical features. The classification task is whether the patient was readmitted to the hospital. The values of the readmission attribute are two: “yes” or “no”. In the original dataset, the classes are: readmitted within 30 days “<30”, readmitted after 30 days “>30” or no.

Table 2: Cross-validated classification accuracy for both unshifted, linear shift, non-linear shift and mixture shift. The numbers between brackets are ranks. VM is the Versatile Model, OM is the original model, $\langle \alpha, \beta \rangle$ corresponds to a linear shift, Perc corresponds to a percentile shift, and KS+… indicates that the Kolmogorov-Smirnov test is used for detecting the shift.

	VM	OM	$\langle \alpha, \beta \rangle$	KS+$\langle \alpha, \beta \rangle$	Perc	KS+Perc
			unshifted data			
Phoneme	0.851(3)	**0.856(1)**	0.846(5)	0.854(2)	0.819(6)	0.850(4)
Bupa	0.631(3)	**0.632(1.5)**	0.619(5)	**0.632(1.5)**	0.578(6)	0.625(4)
Appendicitis	0.846(5.5)	0.849(4)	**0.853(1)**	0.846(5.5)	0.851(2.5)	0.851(2.5)
Pima	**0.728(1.5)**	0.725(4)	**0.728(1.5)**	0.727(3)	0.711(6)	0.721(5)
Breast-w	**0.947(3)**	**0.947(3)**	**0.947(3)**	**0.947(3)**	0.820(6)	**0.947(3)**
Magic	0.821(4)	**0.834(1)**	0.830(3)	0.833(2)	0.773(6)	0.814(5)
Threenorm	0.674(2.5)	**0.682(1)**	0.673(4)	0.674(2.5)	0.635(6)	0.671(5)
Ringnorm	0.735(2.5)	0.731(4.5)	**0.744(1)**	0.735(2.5)	0.678(6)	0.731(4.5)
Ionosphere	0.893(2.5)	**0.894(1)**	0.851(4)	0.893(2.5)	0.825(5.5)	0.825(5.5)
Sonar	0.752(2.5)	**0.754(1)**	0.739(5)	0.752(2.5)	0.716(6)	0.746(4)
Average	0.787(3)	**0.790(2.2)**	0.783(3.25)	0.789(2.7)	0.740(5.6)	0.778(4.25)
			linear shift			
Phoneme	0.825(3)	0.660(6)	**0.846(1.5)**	**0.846(1.5)**	0.819(4.5)	0.819(4.5)
Bupa	0.601(3)	0.558(6)	**0.619(1.5)**	**0.619(1.5)**	0.578(4.5)	0.578(4.5)
Appendicitis	0.844(4.5)	0.776(6)	**0.853(1)**	0.844(4.5)	0.851(2)	0.846(3)
Pima	0.726(3)	0.624(6)	**0.728(1.5)**	**0.728(1.5)**	0.711(4.5)	0.711(4.5)
Breast-w	0.820(4)	0.782(6)	**0.947(1.5)**	**0.947(1.5)**	0.820(4)	0.820(4)
Magic	0.761(5)	0.579(6)	**0.830(1.5)**	**0.830(1.5)**	0.773(3.5)	0.773(3.5)
Threenorm	0.672(2.5)	0.606(6)	**0.673(1)**	0.672(2.5)	0.635(4.5)	0.635(4.5)
Ringnorm	**0.744(1.5)**	0.608(6)	**0.744(1.5)**	0.743(3)	0.678(4.5)	0.678(4.5)
Ionosphere	0.810(5)	0.694(6)	**0.851(1.5)**	**0.851(1.5)**	0.825(3.5)	0.825(3.5)
Sonar	**0.739(2)**	0.624(6)	**0.739(2)**	**0.739(2)**	0.716(4.5)	0.716(4.5)
Average	0.754(3.35)	0.651(6)	**0.783(1.45)**	0.781(2.1)	0.740(4)	0.740(4.1)
			non-linear shift			
Phoneme	**0.819(2)**	0.746(4)	0.720(5.5)	0.720(5.5)	**0.819(2)**	**0.819(2)**
Bupa	**0.594(1)**	0.506(6)	0.571(4.5)	0.571(4.5)	0.578(2.5)	0.578(2.5)
Appendicitis	0.847(4.5)	0.240(6)	0.849(3)	0.847(4.5)	**0.851(1.5)**	**0.851(1.5)**
Pima	0.715(3)	0.478(6)	**0.728(1.5)**	**0.728(1.5)**	0.711(4.5)	0.711(4.5)
Breast-w	0.820(4)	0.464(6)	**0.916(1.5)**	**0.916(1.5)**	0.820(4)	0.820(4)
Magic	0.761(3)	0.398(6)	0.744(4.5)	0.744(4.5)	**0.773(1.5)**	**0.773(1.5)**
Threenorm	0.651(2.5)	**0.671(1)**	0.607(6)	0.635(4.5)	0.635(4.5)	0.651(2.5)
Ringnorm	0.698(2.5)	**0.731(1)**	0.667(6)	0.680(4)	0.678(5)	0.698(2.5)
Ionosphere	**0.825(2)**	0.820(4)	0.781(5.5)	0.781(5.5)	**0.825(2)**	**0.825(2)**
Sonar	**0.744(2)**	0.478(6)	**0.744(2)**	**0.744(2)**	0.716(4.5)	0.716(4.5)
Average	**0.747(2.65)**	0.553(4.6)	0.732(4)	0.736(3.8)	0.740(3.2)	0.744(2.75)
			mixture shift (unshifted, linear shift, non-linear)			
Phoneme	**0.828(1)**	0.749(6)	0.787(5)	0.789(4)	0.819(3)	0.823(2)
Bupa	**0.605(1)**	0.551(6)	0.595(2)	0.594(3)	0.578(5)	0.592(4)
Appendicitis	0.843(4.5)	0.718(6)	0.847(2.5)	0.843(4.5)	**0.851(1)**	0.847(2.5)
Pima	0.710(5)	0.512(6)	**0.727(1)**	0.724(2)	0.711(4)	0.712(3)
Breast-w	0.935(3.5)	0.797(6)	**0.947(1.5)**	**0.947(1.5)**	0.819(5)	0.935(3.5)
Magic	**0.805(1.5)**	0.510(6)	0.802(3)	**0.805(1.5)**	0.773(5)	0.785(4)
Threenorm	**0.672(1)**	0.647(4)	0.635(5.5)	0.653(2)	0.635(5.5)	0.649(3)
Ringnorm	**0.739(1)**	0.674(6)	0.728(2.5)	0.720(4)	0.678(5)	0.728(2.5)
Ionosphere	0.843(3.5)	0.792(6)	0.843(3.5)	**0.865(1)**	0.825(5)	0.848(2)
Sonar	**0.740(1)**	0.631(6)	0.737(3)	0.738(2)	0.716(4)	0.712(5)
Average	**0.772(2.3)**	0.658(5.8)	0.764(2.95)	0.767(2.55)	0.740(4.25)	0.763(3.15)

Table 3: Classification accuracy for Diabetes dataset. Symbols denote ethnic groups as follows: African-American (AA), Asian (A), Caucasian (C), Hispanic (H). X-Y denotes trained on X, deployed on Y.

	A-AA	A-C	A-H	AA-A	AA-C	AA-H	C-A	C-AA	C-H	H-A	H-AA	H-C
# shifted	6	5	4	6	5	4	5	5	4	4	4	4
VM	0.569	0.529	**0.576**	**0.653**	**0.530**	**0.590**	0.645	0.546	0.588	0.624	**0.565**	**0.564**
OM	**0.574**	**0.538**	0.554	0.642	0.526	0.587	0.641	**0.566**	**0.595**	**0.642**	0.562	0.563
IOP	0.526	0.499	0.547	0.500	0.494	0.463	0.520	0.488	0.469	0.519	0.509	0.452
KMM	0.467	0.499	0.419	0.352	**0.530**	0.474	**0.647**	0.557	0.580	0.400	0.442	0.507

In our experiment, we split the dataset in 4 subsets according to the “ethnic group” the patient belongs to. There are 4 different groups: Caucasian, African American, Asian, and Hispanic. We evaluate the performance of models trained on ethnic group X and deployed on ethnic group Y, denoted by X-Y. Table 3 shows the performance of the Versatile Model against the original model, IOP and KMM. We also report the number of shifted attributes according to the KS test. The Versatile Model wins most often, followed by the original model.

Heart Our next benchmark is the heart disease dataset. We split it into two subsets according to gender: male and female. In this dataset there are 5 continuous attributes, 3 of them are indicated as shifted between gender according to KS test, which are age, heart rate and serum cholesterol. Table 4 shows the performance of versatile method against the original model, IOP and KMM. In both contexts the VM has the best accuracy among all three methods including the original model.

Table 4: Classification accuracy for Heart dataset, with contexts by gender (F: Female, M: Male).

	M-F	F-M
# shifted	3	3
VM	**0.735**	**0.568**
OM	0.712	0.557
IOP	0.703	0.500
KMM	0.724	0.540

Bike Sharing This dataset [8] contains the hourly and daily count of rental bikes between years 2011 and 2012 in addition to weather information. It contains 4 continuous attributes: actual and apparent temperature in Celsius, humidity and wind speed. The classification task is whether there is a demand in this period of time or not. In order to evaluate the shift effects, we split the dataset as proposed in [1] to obtain the 4 seasons datasets. According to KS, all these 4 attributes are detected as shifted except in 3 cases. First, between Summer-Spring, wind speed is not shifted. Second, in both

Table 5: Classification accuracy for Bike Sharing dataset, with contexts by season (Sp: Spring, S: Summer, A: Autumn, W: Winter).

	Sp-S	Sp-A	Sp-W	S-Sp	S-A	S-W	A-Sp	A-S	A-W	W-Sp	W-S	W-A
# shifted	3	4	4	3	3	4	4	3	3	4	4	3
VM	**0.641**	**0.558**	**0.601**	0.519	**0.579**	0.601	**0.602**	**0.543**	0.556	0.646	0.565	0.526
OM	0.538	0.468	0.544	0.607	0.547	0.612	0.574	0.521	0.528	0.718	**0.657**	**0.558**
IOP	0.489	0.468	0.533	**0.635**	0.510	**0.657**	0.585	0.534	0.522	0.658	0.630	0.510
KMM	0.559	0.468	0.522	**0.635**	0.521	0.651	0.585	0.521	**0.589**	**0.690**	0.521	0.521

Table 6: Classification accuracy for AutoMPG dataset, with contexts by origin (U: USA, E: Europe, J:Japan).

	U-E	U-J	E-U	E-J	J-U	J-E
# shifted	4	4	4	3	4	3
VM	**0.676**	**0.759**	**0.873**	**0.772**	**0.780**	0.647
OM	0.544	0.607	0.670	0.746	0.747	**0.691**
IOP	0.558	0.493	0.400	0.417	0.600	0.441
KMM	0.558	0.582	0.600	0.582	0.400	0.485

Summer-Autumn and Autumn-Winter, humidity is not shifted. The performance of Versatile Model and others are shown in Table 5. Again we note the solid performance of the Versatile Model.

AutoMPG AutoMPG dataset [8] concerns the consumption in miles per gallon of vehicle from 3 different regions: USA, Europe and Japan. It contains 4 numerical attributes: displacement, horsepower, weight and acceleration. All these input attributes have been detected as shifted between regions using KS test. This dataset has been binarised according to the mean value of the target. We split the dataset as proposed in [1] to obtain the 3 regions datasets. The performance of Versatile Model and others are shown in Table 6. The VM outperforms all three methods and has only one loss against the original model.

Finally, we report the result of a Friedman test and post-hoc analysis on all non-synthetic shifts. Figure 4 demonstrates that the Versatile Model outperforms all others, significantly so except for the original model.

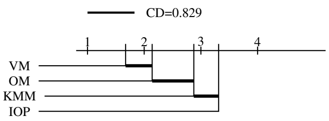


Fig. 4: Critical Difference diagram using pairwise comparisons for non synthetic shift. Average ranks as follows: VM=1.671, OM=2.140, KMM=2.875 and IOP= 3.312. The Friedman test gives significance at 0.05.

5 Conclusion

We proposed a model for adapting to covariate observation shift using unlabelled deployment data. The proposed model is called the Versatile Model and is a Decision Tree model with enhanced splits. The main idea of the VM is that it captures more information about the context during the training process in order to be able to adapt this model for deployment contexts. The VM trains a classifier over the available data and then adapts some of its decisions according to the (usually unlabelled) deployment data. We use a non-parametric test to choose among different strategies to update the decision thresholds in a DT. The VM does not make any strong assumptions such as linear transformation between contexts. Furthermore, it does not need any tuning parameters to adjust the model. Finally, empirical results on both synthetic shift and real dataset shift show strong performance gains by achieved the proposed methods.

This work opens up many avenues for future work. One direction is to adapt the VM to other predictive problems, such as regression. Another direction is to assume that the deployment data is partially labelled and utilise this knowledge in the VM.

Acknowledgment

Reem is a PhD student who is sponsored by King Abdulaziz University, Saudi Arabia. Ricardo Prudencio is financially supported by CNPq (Brazilian Agency) and did this work during a visit to the University of Bristol. This work was partly supported by the REFRAME project granted by the European Coordinated Research on Long-term Challenges in Information and Communication Sciences and Technologies ERA-Net (CHISTERA), and funded by the Engineering and Physical Sciences Research Council in the UK under grant EP/K018728/1.

References

1. Ahmed, C.F., Lachiche, N., Charnay, C., Braud, A.: Reframing continuous input attributes. In: Tools with Artificial Intelligence (ICTAI), 2014 IEEE 26th International Conference on. pp. 31–38 (Nov 2014)
2. Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S.: Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. Multiple-Valued Logic and Soft Computing 17(2-3), 255–287 (2011)
3. Bickel, S., Brückner, M., Scheffer, T.: Discriminative learning under covariate shift. Journal of Machine Learning Research 10, 2137–2155 (Dec 2009)
4. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30 (Dec 2006)
5. Elkan, C.: The foundations of cost-sensitive learning. In: In Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence. pp. 973–978 (2001)
6. Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., Schlkopf, B.: Covariate shift by kernel mean matching. In: Joaquin Quiñonero-Candela, Masashi Sugiyama, A.S., Lawrence, N.D. (eds.) Dataset Shift in Machine Learning, pp. 131–160. MIT Press (2009)
7. Kull, M., Flach, P.: Patterns of dataset shift. In: First International Workshop on Learning over Multiple Contexts (LMCE) at ECML-PKDD 2014. Nancy, France (September 2014)

8. Lichman, M.: UCI machine learning repository (2013), http://archive.ics.uci.edu/ml
9. Moreno-Torres, J.G., Llorí, X., Goldberg, D.E., Bhargava, R.: Repairing fractures between data using genetic programming-based feature extraction: A case study in cancer diagnosis. Inf. Sci. 222, 805–823 (Feb 2013)
10. Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. Pattern Recognition 45(1), 521–530 (Jan 2012)
11. Moreno-Torres, J.G., Raeder, T., Aláiz-Rodríguez, R., Chawla, N.V., Herrera, F.: Tackling dataset shift in classification: Benchmarks and methods. http://sci2s.ugr.es/dataset-shift, accessed: 2015-03-30
12. Storkey, A.J.: When training and test sets are different: Characterising learning transfer. In: Joaquin Quiňonero-Candela, Masashi Sugiyama, A.S., Lawrence, N.D. (eds.) Dataset Shift in Machine Learning, chap. 1, pp. 3–28. MIT Press (2009)
13. Strack, B., DeShazo, J.P., Gennings, C., Olmo, J.L., Ventura, S., Cios, K.J., Clore, J.N.: Impact of hba1c measurement on hospital readmission rates: analysis of 70,000 clinical database patient records. BioMed research international 2014, 781670 (2014), http://europepmc.org/articles/PMC3996476
14. Sugiyama, M., Krauledat, M., Müller, K.R.: Covariate shift adaptation by importance weighted cross validation. Journal of Machine Learning Research 8, 985–1005 (Dec 2007)
15. Zadrozny, B.: Learning and evaluating classifiers under sample selection bias. In: In International Conference on Machine Learning ICML04. pp. 903–910 (2004)