Addressing Underestimation Bias in CBR through Case-Base Maintenance

William Blanzeisky and Pádraig Cunningham

School of Computer Science
University College Dublin
Dublin 4, Ireland

Abstract. The knowledge containers perspective on CBR suggests that bias and fairness issues can be addressed in a number of different ways. In this paper we assess the use of case-base maintenance to ensure fairness. We present FairCBM, a strategy for removing cases that are causing biased classifications. We evaluate this strategy on five different datasets and show that it is effective for ensuring fairness with minimal impact on classification accuracy. FairCBM is also evaluated against an alternative metric learning strategy (similarity knowledge container); the evaluation shows that both strategies are equally effective. FairCBM has the benefit that it is quite transparent; by comparison the metric learning strategy is more opaque.

1 Introduction

Algorithmic bias and fairness has become a major concern in machine learning research in recent years and case-based reasoning (CBR) is not an exception in this regard. Recently Blanzeisky *et al.* [6] presented FairRet a metric learning strategy to help ensure fairness in CBR. From a knowledge containers perspective, this is just one way that bias and fairness can be addressed in CBR. Richter's knowledge container perspective on CBR includes four knowledge containers, vocabulary, case-base, similarity and adaptation [20]. Whereas metric learning addresses the similarity knowledge container, in this paper we consider tackling bias in the case-base itself.

Addressing fairness by removing cases causing bias rather than through tweaking the similarity metric has some advantages. There is merit in removing cases that are causing problems and this is quite transparent compared with adjustments to the similarity metric that can be quite opaque. CBR research on case-base maintenance (CBM) has a long history, the relevant aspects of which are reviewed in section 2.2.

Our algorithm for removing cases causing bias is presented in section 3. The basic idea is to identify the cases that cause biased predictions; these can be ranked by the number of biased predictions they cause. Then cases above a threshold count can be removed. This threshold is set through a cross-validation process on the training data.

An evaluation of this algorithm on five datasets is presented in section 4. The relevant background research is reviewed in section 2 and the paper finishes with some conclusions and discussion of future work in section 5.

2 Related Research

This research is informed by other work relating to bias in machine learning (ML) and case-base manitenance in CBR. The relevant work in these areas is discussed in the following subsections.

2.1 Bias in ML

ML models play an increasingly important role in making decisions that impact people's lives, including those related to employment, credit, housing, and criminal justice. The issue of ensuring fairness in ML models is critical, as biased or unfair models can perpetuate existing inequalities and discrimination, with minority groups being particularly vulnerable. Several measures of unfairness have been proposed in recent years, each emphasizing different aspects of fairness [8]. In general, an ML model is considered 'fair' if it is not inclined to award desirable outcomes $Y = 1$ (e.g., loan approval/job offers) preferentially to one side of a protected category $S = 0$ (e.g. female/protected group). Disparate Impact (DI_S) is one of the accepted measures of unfairness [12]:

$$\mathrm{DI}_S \leftarrow \frac{P[\hat{Y} = 1|S = 0]}{P[\hat{Y} = 1|S = 1]} < \tau \tag{1}$$

It is the ratio of desirable outcomes $\hat{Y}$ predicted for the protected minority $S = 0$ compared with that for the majority $S = 1$. However, this measure is independent of what is actually in the training data.

In this paper, we are particularly interested in the bias introduced by the algorithms themselves, usually referred to as *underestimation*. Underestimation bias arises when the distribution of predictions from an algorithm is not in line with the true underlying distribution of the data. This happens when the algorithm focuses on strong signals in the data thereby missing more subtle phenomena. Hence, the classifier accentuates bias that might be present in the data and underestimates the infrequent outcome for the minority group. An underestimation score (US_S) in line with DI_S (see Equation 1) that compares predicted and actual outcomes for the protected minority would be [5]:

$$\mathrm{US}_{S=0} \leftarrow \frac{P[\hat{Y} = 1|S = 0]}{P[Y = 1|S = 0]} \tag{2}$$

This is the ratio of desirable outcomes predicted by the classifier for the protected group compared with what is actually present in the data [5]. If $\mathrm{US}_{S=0} < 1$ the classifier is under-predicting desirable outcomes for the minority. It is worth noting that when $\mathrm{US}_{S=0} = 1$ the classifier may still be biased against the minority group; it is faithful to the data but there may still be a poor DI_S score.

An alternative underestimation score that considers divergences between overall actual and predicted distributions for all groups S is the underestimation index (UEI) based on the Hellinger distance [17] :

$$\mathrm{UEI} = \sqrt{1 - \sum_{y,s \in D} \sqrt{P[\hat{Y} = y, S = s] \times P[Y = y, S = s]}} \tag{3}$$

Here y and s are the possible values of Y and S respectively. This Hellinger distance is preferred to KL-divergence because it is bounded in the range [0,1] and KL-divergence has the potential to be infinite. UEI = 0 indicates that there is no difference between the probability distribution of the training samples and the prediction made by a classifier (no underestimation). [17] refer to underestimation as the state in which a learned model is not fully converged due to the finiteness of the size of a training dataset - i.e., the learned classifier may lead to more unfair determinations than that observed in the training sample distribution. Although this notion is useful when quantifying the extent to which a model's prediction deviates from the training samples, it does not directly tell us how the protected group is doing since it is an aggregate score across all protected attributes S and outcomes Y.

Bias mitigation strategies in machine learning (ML) can be broadly classified into three categories depending on the stage at which fairness measures are applied. These stages include the pre-processing, in-processing, and post-processing steps, each of which has its unique set of approaches towards ensuring fairness.

Pre-processing Techniques: In this category, the focus is on the data, which is often the root of bias issues. The strategies involve transforming the dataset to correct imbalances or unrepresentative aspects before feeding it into the ML model. A variety of methods have been proposed, including disparate impact repair strategies as proposed by Feldman et al. [12], and probabilistic mappings designed to maintain individual and group fairness, as suggested by Dwork et al [11]. Other techniques involve the re-labeling or perturbation of data to eliminate undesirable biases [16, 24]. Our proposed strategy, FairCBM, falls under this category, addressing underestimation bias by removing cases from the training data that contribute to biased classifications.

In-processing Techniques: These approaches aim to reduce bias during the model-building process by adjusting the algorithm's objective function to account for fairness measures [17, 22]. This is often achieved by enforcing a fairness constraint into the algorithm's optimization function, thereby transforming algorithmic bias into a multi-objective optimization problem (MOOP) [19, 2, 22, 13, 4]. Some strategies for handling the non-convex optimization problem that arises include leveraging recent advances in convex-concave programming or using a majorization-minimization procedure [23]. Other specific strategies include modifying decision trees' splitting criteria and pruning strategies [1] or adding a latent variable representing the unbiased label to the Bayesian model [7]. The FairRet method proposed by Blanzeisky et al. [6] also belongs to this category. It aims to correct underestimation in case-based reasoning (CBR) systems by including underestimation as an additional criterion in the ML optimization process.

Post-processing Techniques: The final category acknowledges that the ML model's output might be biased towards specific subgroups within the protected attribute. To mitigate this, transformations are applied to the model's output to ensure fairness, such as setting different thresholds for different subgroups [9].

Although pre- and post-processing techniques can effectively mitigate bias without explicitly modifying the ML optimization process, they may have legal implications and reduce the model's interpretability [8]. Despite the challenges, these approaches, along with in-processing methods, continue to play a crucial role in addressing algorithmic bias and fostering fairness in ML.

2.2 Case-Base Maintenance

As pointed out in the IJCAI 2018 review paper by Juarez *et al.* [15] Case-Base Maintenance has a long history dating back to the Condensed Nearest Neighbor paper by Hart in 1968 [14]. There can be two motivations for CBM, to improve the efficiency of the retrieval process by removing redundant cases [3] or to improve the *competence* of the system by removing noisy cases [10].

Juarez *et al.* point out that, when the objective is to improve system competence, the Smyth-Keane-McKenna competence model [21, 18] has stood the test of time. This competence model is based on two key concepts, the Coverage Set and the Reachability Set. The Coverage Set (CS) of a case is the set of other cases that this case can be used to solve. In the same way, the Reachability Set (RS) of a case is the set of other cases that solve the case in question. In classification scenarios, *solves* simply means that one case is an appropriate retrieval for another, i.e. they are in the nearest neighbor set and have the same class. These can be expressed formally as follows:

$$CS(c, C) = \{c' \in C \mid c' \in NN(c, C) \wedge c \text{ solves } c'\} \tag{4}$$

$$RS(c, C) = \{c' \in C \mid c \in NN(c', C) \wedge c' \text{ solves } c\} \tag{5}$$

where C is the case-base and $NN(c, C)$ are the nearest neighbors of c. This model was extended by Delany and Cunningham [10] to include the concept of a Liability Set (LS), this is the set of cases that are misclassified by c.

$$LS(c, C) = \{c' \in C \mid c \text{ misclassifies } c'\} \tag{6}$$

They present an algorithm for deleting cases that are causing misclassifications and are not required otherwise, i.e. the cases in their CS will be correctly classified even if they are deleted. Our algorithm for deleting cases that result in biased classifications is based on this idea - see section 3.

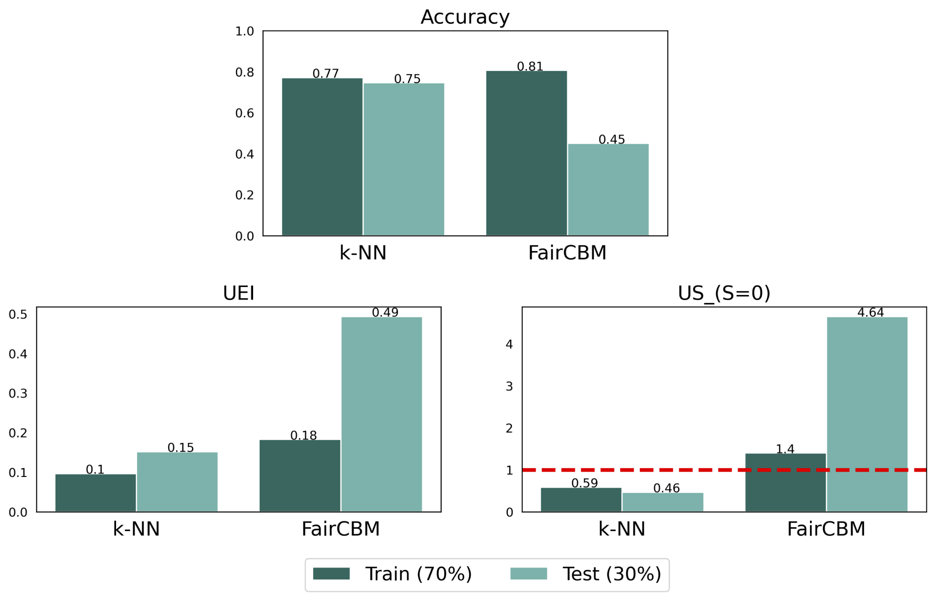


Fig. 1: These charts illustrates the performance of FairCBM with a naive policy of deleting all cases that contribute to biased classification ($|LS_{\text{FairCBM}}| > 0$) on Titanic dataset. The baseline k-NN is biased as we can see from the UEI and $US_{S=0}$ scores. The naive FairCBM policy removes 189 ($\sim 37\%$) samples from the training set. In terms of the $US_{S=0}$ score, this overshoots slightly on the training data and very significantly on unseen test data. The UEI scores are actually worse than the k-NN baseline on train and test data.

3 FairCBM: CBM Strategy to Ensure Fairness

This section introduces FairCBM, a Case-Base Maintenance strategy to address unfairness in CBR systems. Our focus is on situations where the model under-predicts desirable outcomes for minority groups (i.e., loan approvals for females), which can occur when the model prioritizes overall accuracy over fairness. To combat this, we propose FairCBM as a solution to maintain the case-base and reduce bias in the model.

FairCBM builds upon the Liability Set (LS) concept introduced by Delaney and Cunningham [10], but takes a novel approach to *tune* the amount of case deletion. Rather than removing cases that result in incorrect classifications, FairCBM identifies and removes cases that contribute to bias. This approach seeks to establish a case-base that is fair to all groups S while maintaining accuracy.

Focusing only on underestimation for the minority group, FairCBM first constructs a liability set LS_{FairCBM} to identify a set of cases that lead to biased classifications for the minority group:

$$LS_{\text{FairCBM}}(c, C) = \{c' \in Q \mid c \text{ misclassifies } c'\} \tag{7}$$

where $Q = \{q \in C \mid S_q = 0 \wedge Y_q = 1\}$, i.e. Q represents the set of cases in the minority group having desirable outcomes.

A naive strategy to mitigate underestimation is to delete all cases that contribute to biased classifications ($|LS_{\text{FairCBM}}| > 0$). However, initial experiments on this strategy showed worsened overall model performance and overshooting in underestimation ($\text{US}_{S=0} > 1$) on the test set. Figure 1 clearly illustrates this phenomenon on the Titanic dataset. Deleting all cases with non-empty LS removes 189 (∼37%) cases. This causes accuracy to drop to 45% from 75% while $\text{US}_{S=0}$ raises to 4.64 which is above the target of 1.

This is likely due to the limited number of samples for desirable outcomes for the minority group in the dataset. In most cases, cases that lead to the biased classification ($|LS_{\text{FairCBM}}| > 0$) are samples for undesirable outcomes for the minority group ($S = 0|Y = 0$). Removing all of these cases may result in the model not learning as effectively for these samples, leading to overestimation. This highlights the importance of tuning. In the next subsection, we show how a cross-validation strategy can be used to select an appropriate threshold for the case deletion process to avoid overshooting.

3.1 Tuning

Given the importance of the case deletion process to the overall model performance, determining the appropriate deletion threshold τ is key. Cases causing at least τ biased classifications will be removed. In a hold-out-test scenario, we initially divide the initial training dataset into training and validation sets (as depicted in Figure 2). Following this, we construct the liability set from the training set employing a leave-one-out method to identify biased classification. The validation set is then used to set the threshold τ for case deletion. We incrementally increase τ where $\tau \in Set(|LS_{\text{FairCBM}}(q, C)|)$ and select τ according

Table 1: The impact of different values of τ on the validation set.

τ	**Accuracy**	**UEI**	**US_S=0**	**# of cases deleted**
1	0.454	0.572	3.939	142 (46.4%)
2	0.605	0.232	2.576	112 (36.6%)
3	0.722	0.075	1.485	79 (25.8%)
4	0.722	0.067	1.273	45 (14.7%)
5	0.722	0.063	1.030	30 (9.8%)
6	0.702	0.075	0.970	26 (8.5%)
7	0.702	0.088	0.727	19 (6.2%)
8	0.707	0.099	0.636	16 (5.2%)
9	0.727	0.138	0.394	9 (2.9%)
10	0.732	0.154	0.333	10 (1.3%)

to its performance on the validation set. Upon identifying the optimal τ, we construct a new liability set, using the leave-one-out approach on the combined training and validation sets. Subsequently, we carry out the case deletion based on the optimized τ.

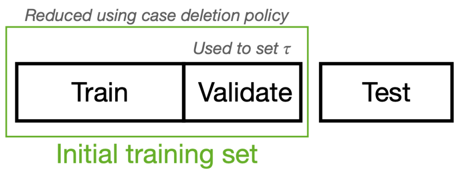


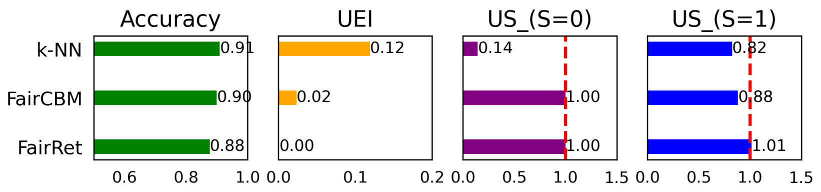
Fig. 2: The strategy for setting the deletion threshold τ. The initial training set is divided into Train and Validate sets and the validation set is used to find the optimal value for τ. Then this threshold is used when case deletion is applied in the initial training set.

An example of this method in operation on the Titanic dataset can be seen in Table 1. When $\tau = 1$ (all cases with non-empty LS are deleted) 142 cases are deleted. This reduces accuracy to 45% and we have almost 4-fold overestimation. By relaxing the threshold we can reduce this overshooting; we see that ($\tau = 5$) is enough to eliminate underestimation on the validation set so this is selected as the threshold.

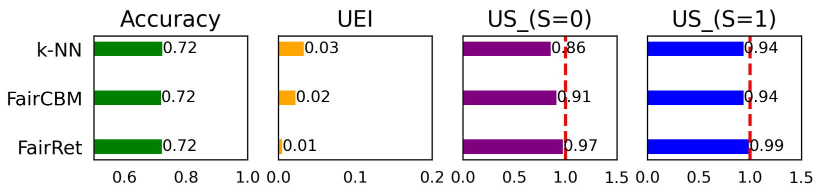
Following the selection of the optimal value τ, we perform the case deletion process on the initial training data before assessing the models performance on the hold-out test set. Notably, the validation process for identifying the optimal value of τ uses only the training data.

Table 2: Summary details of all the datasets used in the experiments. The sensitive group is chosen to have a significant discrepancy between the majority and minority groups. For consistency, the positive class indicates the minority outcomes. In some cases, this may require some pre-processing.

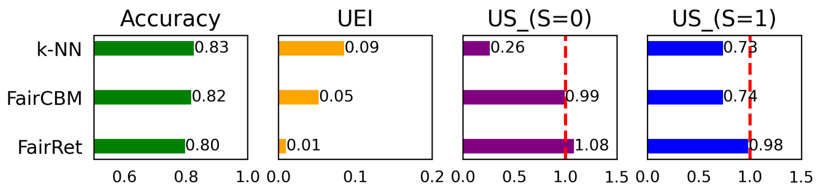
Dataset	**# of Instances**	**# of Attributes (cat./bin./num.)**	**Class ratio (+:-)**	**Sensitive group**	**Target class**
Bike Sharing	731	5:3:2	1:3.55	working day	usage
Synthetic	5,000	0:1:2	1:1.48	gender	admit
red. Adult	48,842	0:1:6	1:3.03	gender	income
Titanic	891	4:1:2	1:1.61	sex	survived
Bank Marketing	45,211	6:4:7	1:21.64	age	subscription



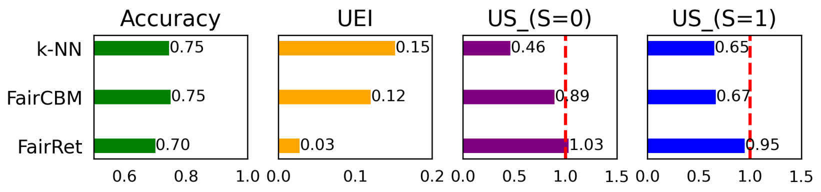
(a) Bike Sharing



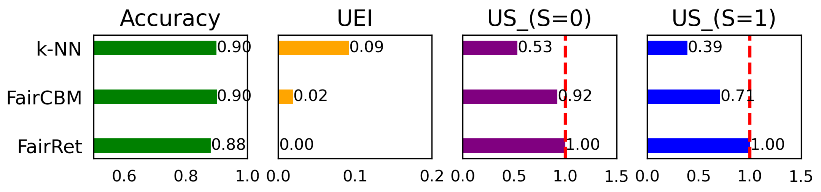
(b) Synthetic



(c) Adult



(d) Titanic



(e) Bank Marketing

Fig. 3: An evaluation of the remediation strategies on five datasets. It is clear that both FairCBM and FairRet are effective in mitigating underestimation, but FairRet takes a bigger hit on the accuracy likely due to optimizing for UEI which is a broader criterion than $US_{S=0}$, whereas FairCBM focuses on $US_{S=0}$ only.

4 Evaluation

In this section, we evaluate FairCBM on one synthetic and four real-world datasets commonly used in the fairness literature. In each dataset, we choose a binary feature as the sensitive attribute S on which the model should be fair. Summary statistics for these datasets are provided in Table 2.

To illustrate the effectiveness of the proposed strategy, we compare FairCBM against standard k-NN and FairRet [6] the metric learning strategy to address underestimation mentioned in section 2. The main results are plotted in Figure 3:

- First it is clear that underestimation is an issue when standard k-NN is used. This is evident from the high UEI and low $US_{S=0}$ scores. This is not surprising since these models are optimized solely on accuracy, without consideration for how the inaccuracies distribute.
- Secondly, we can see that FairRet is effective in fixing underestimation, as measured by $US_{S=0}$ and UEI, compared to standard k-NN.
- This good performance on fairness by FairRet comes at a price in accuracy; for instance, for the Adult dataset, accuracy falls by 5%.
- Turning to FairCBM, we see that it is very effective in bringing $US_{S=0}$ close to 1. It is less effective in fixing UEI. Again, this is not surprising given that the FairCBM algorithm focuses on $US_{S=0}$ only and does not consider UEI which is a wider criterion.

In summary, when it comes to addressing underestimation, FairCBM performs equally well compared to FairRet specifically in relation to the minority group $US_{S=0}$. However, it falls short in terms of the broader UEI criterion. On the positive side, FairCBM excels at preserving accuracy, with a minimal decrease of only 1% observed across the five datasets.

5 Conclusions

In this paper, we introduce FairCBM, a case-base maintenance strategy designed to enhance fairness in Case-Based Reasoning (CBR) systems. This strategy works by eliminating cases that lead to underestimations of favorable outcomes for minority groups, such as loan approvals for females. Experimental results show that FairCBM performs on par with the metric learning strategy by Blanzeisky et al. [6] in terms of mitigating underestimation. An outstanding feature of FairCBM is its transparency, which stands in stark contrast to the relatively opaque nature of the metric learning approach. Looking forward, we aim to explore the possibility of integrating both strategies to enhance the effectiveness in rectifying bias.

Acknowledgements

This work was funded by Science Foundation Ireland through the SFI Centre for Research Training in Machine Learning (Grant No. 18/CRT/6183) with support from Microsoft Ireland. We wish to thank Sarah Jane Delany for comments on an earlier draft of this paper.

References

1. Discrimination aware decision tree learning. pp. 869–874 (2010). https://doi.org/10.1109/ICDM.2010.50
2. Agarwal, A., Beygelzimer, A., Dudík, M., Langford, J., Wallach, H.: A Reductions Approach to Fair Classification. 35th International Conference on Machine Learning, ICML 2018 **1**, 102–119 (3 2018), http://arxiv.org/abs/1803.02453
3. Aha, D.W., Kibler, D., Albert, M.K.: Instance-based learning algorithms. Machine learning **6**, 37–66 (1991)
4. Bechavod, Y., Ligett, K.: Penalizing Unfairness in Binary Classification (6 2017), http://arxiv.org/abs/1707.00044
5. Blanzeisky, W., Cunningham, P.: Algorithmic factors influencing bias in machine learning. In: Kamp, M. (ed.) Machine Learning and Principles and Practice of Knowledge Discovery in Databases. pp. 559–574. Springer International Publishing, Cham (2021)
6. Blanzeisky, W., Smyth, B., Cunningham, P.: Algorithmic bias and fairness in case-based reasoning. In: Case-Based Reasoning Research and Development: 30th International Conference, ICCBR 2022, Nancy, France, September 12–15, 2022, Proceedings. pp. 48–62. Springer (2022)
7. Calders, T., Verwer, S.: Three naive Bayes approaches for discrimination-free classification. In: Data Mining and Knowledge Discovery. vol. 21, pp. 277–292. Springer US (9 2010). https://doi.org/10.1007/s10618-010-0190-x
8. Caton, S., Haas, C.: Fairness in machine learning: A survey. arXiv preprint arXiv:2010.04053 (2020)
9. Corbett-Davies, S., Goel, S.: The measure and mismeasure of fairness: A critical review of fair machine learning (2018)
10. Delany, S.J., Cunningham, P.: An analysis of case-base editing in a spam filtering system. In: Advances in Case-Based Reasoning: 7th European Conference, ECCBR 2004, Madrid, Spain, August 30-September 2, 2004. Proceedings 7. pp. 128–141. Springer (2004)
11. Dwork, C., Hardt, M., Pitassi, T., Reingold, O., Zemel, R.: Fairness through awareness. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. p. 214–226. ITCS '12, Association for Computing Machinery, New York, NY, USA (2012). https://doi.org/10.1145/2090236.2090255
12. Feldman, M., Friedler, S.A., Moeller, J., Scheidegger, C., Venkatasubramanian, S.: Certifying and removing disparate impact. In: proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining. pp. 259–268 (2015)
13. Goh, G., Cotter, A., Gupta, M., Friedlander, M.: Satisfying Real-world Goals with Dataset Constraints. Tech. rep. (2016)
14. Hart, P.: The condensed nearest neighbor rule (corresp.). IEEE transactions on information theory **14**(3), 515–516 (1968)
15. Juarez, J.M., Craw, S., Lopez-Delgado, J.R., Campos, M.: Maintenance of case bases: Current algorithms after fifty years. In: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18. pp. 5457–5463. International Joint Conferences on Artificial Intelligence Organization (7 2018)

16. Kamiran, F., Calders, T.: Data preprocessing techniques for classification without discrimination. Knowledge and Information Systems **33**(1), 1–33 (10 2012). https://doi.org/10.1007/s10115-011-0463-8, https://link.springer.com/article/10.1007/s10115-011-0463-8
17. Kamishima, T., Akaho, S., Asoh, H., Sakuma, J.: Fairness-aware classifier with prejudice remover regularizer. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 35–50. Springer (2012)
18. McKenna, E., Smyth, B.: Competence-guided case-base editing techniques. In: Advances in Case-Based Reasoning: 5th European Workshop, EWCBR 2000 Trento, Italy, September 6–9, 2000 Proceedings 5. pp. 186–197. Springer (2000)
19. Quadrianto, N., Sharmanska, V.: Recycling Privileged Learning and Distribution Matching for Fairness. Tech. rep. (2017)
20. Richter, M.: Introduction–the basic concepts of cbr. Case-Based Reasoning Technology: From Foundations to Applications, LNAI **1400** (1998)
21. Smyth, B., Keane, M.: Remembering to forget: a competence-preserving case deletion policy for case-based reasoning sytems. In: Proc. of 14th International Joint Conference on Artificial Intelligence. pp. 377–382 (1995)
22. Woodworth, B., Gunasekar, S., Ohannessian, M.I., Srebro, N.: Learning non-discriminatory predictors. In: Conference on Learning Theory. pp. 1920–1953. PMLR (2017)
23. Zafar, M.B., Valera, I., Rodriguez, M.G., Gummadi, K.P.: Fairness Constraints: Mechanisms for Fair Classification. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017 (7 2015), http://arxiv.org/abs/1507.05259
24. Zliobaite, I., Kamiran, F., Calders, T.: Handling conditional discrimination. In: Proceedings - IEEE International Conference on Data Mining, ICDM. pp. 992–1001 (2011). https://doi.org/10.1109/ICDM.2011.72