# Proceedings of the 3rd International Workshop on Learning to Quantify (LQ 2023)

Mirko Bunse, Pablo González,
Alejandro Moreo, and Fabrizio Sebastiani (eds.)

# Preface

The 3rd International Workshop on Learning to Quantify (LQ 2023 – `https://lq-2023.github.io/`) was held in Torino, IT, on September 18, 2023, as a satellite workshop of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD 2023). While the 1st edition of the workshop (LQ 2021 – `https://cikmlq2021.github.io/`) had to be an entirely online event, LQ 2023 (like the 2nd edition LQ 2022 – `https://lq-2023.github.io/`) was a hybrid event, with presentations given in-presence, and both in-presence attendees and remote attendees. The workshop was the second part (Sep 18 afternoon) of a full-day event, whose first part (Sep 18 morning) consisted of a tutorial on Learning to Quantify presented by Alejandro Moreo and Fabrizio Sebastiani. The LQ 2023 workshop consisted of the presentations of seven contributed papers, and a final collective discussion on the open problems of learning to quantify and on future initiatives.

The present volume contains five of the seven contributed papers that were accepted for presentation at the workshop (the authors of the other two papers decided not to have their paper included in the proceedings). Each contributed paper was submitted as a response to the call for papers, was reviewed by at least three members of the international program committee, and was revised by the authors so as to take into account the feedback provided by the reviewers.

We hope that the availability of the present volume will increase the interest in the subject of quantification on the part of researchers and practitioners alike, and will contribute to making quantification better known to potential users of this technology and to researchers interested in advancing the field.

Mirko Bunse
Pablo González
Alejandro Moreo
Fabrizio Sebastiani

# Table Of Contents

# LQ 2023 Program Committee

Mirko Bunse, University of Dortmund, DE (co-Chair)
González, University of Oviedo, ES (co-Chair)
Alejandro Moreo, Consiglio Nazionale delle Ricerche, IT (co-Chair)
Fabrizio Sebastiani, Consiglio Nazionale delle Ricerche, IT (co-Chair)

Rocío Alaíz-Rodríguez, University of León, ES
Gustavo Batista, University of New South Wales, AU
Juan José del Coz, University of Oviedo, ES
Andrea Esuli, Consiglio Nazionale delle Ricerche, IT
Alessandro Fabris, Max Planck Institute for Security and Privacy, DE
Cèsar Ferri, Universitat Politècnica de València, ES
George Forman, Amazon Research, US
Wei Gao, Singapore Management University, SG
Rafael Izbicki, Federal University of São Carlos, BR
André G. Maletzke, Universidade Estadual do Oeste do Paraná, BR
Marco Saerens, Catholic University of Louvain, BE
Dirk Tasche, Swiss Financial Market Supervisory Authority, CH

# Sponsors

The work of Alejandro Moreo and Fabrizio Sebastiani has been supported by the SoBigdata++ project, funded by the European Commission (Grant 871042) under the H2020 Programme INFRAIA-2019-1, and by the AI4Media project, funded by the European Commission (Grant 951911) under the H2020 Programme ICT-48-2020. The authors' opinions do not necessarily reflect those of the European Commission.

# qunfold: Composable Quantification and Unfolding Methods in Python

Mirko Bunse[0000−0002−5515−6278]

Artificial Intelligence Unit, TU Dortmund University, 44227 Dortmund, Germany
mirko.bunse@cs.tu-dortmund.de

**Abstract.** We present qunfold, a Python package that implements several quantification and unfolding methods. A unique capability of qunfold is the composition of novel methods from existing and easily customized loss functions and data representations. Moreover, this package leverages a powerful optimization back-end to yield state-of-the-art performances for all compositions. This paper introduces the common usage patterns for qunfold, revisits the technical background of the package, and empirically demonstrates the resulting performance.

**Keywords:** Quantification · Unfolding · Unconstrained optimization · Multi-class classification · Software

## 1 Introduction

Many quantification methods, i.e., many methods for the supervised estimation of class prevalences [12], can be described as a combination of a loss function and a data representation [5,11]. This observation motivates implementations that make this combination explicit, to provide a high amount of code quality (separation of concerns, reusability) and to establish an opportunity of composing new methods from existing components.

The value of such compositions comes from the specific characteristics that each component introduces; unfolding methods, for instance, address an order among the classes through regularization [6]. Hence, the name qunfold mingles the terms "quantification" and "unfolding" to hint at the package's capability of composing new methods from existing loss functions and data representations. The diversity of quantification use cases, including political sciences, market research, epidemiology, and others [8] calls for this capability.

Implementations of quantification methods have to define another aspect in addition to the loss function and data representation: the numerical optimization algorithm through which the loss is minimized. This additional aspect is vital in the multi-class setting, where an exhaustive search of all class prevalences is not feasible. A recent proposal is to employ a soft-max operator to ensure legitimate and accurate results with unconstrained optimization algorithms [4], including those well-tested algorithms that a standard numpy/scipy stack provides.

Our Python package `qunfold`[1] leverages these recent developments to provide a highly performant and composable implementation of many existing quantification and unfolding methods. The package, which is released under an open-source license, is designed for meeting the following goals:

- focus on methods (disregarding data loading, evaluation protocols, etc.)
- easy composition of new methods
- high prediction performance due to a powerful optimization routine
- easy extendability through API design and through automatic differentiation
- compliance with the conventions established by `scikit-learn`
- detailed documentation and high test coverage

These goals partially differ from the goals of QuaPy [16], the current state-of-the-art implementation for all aspects of quantification, including the acquisition of data and the evaluation of methods. QuaPy provides a large collection of quantification methods, but does not allow to recompose them. We provide a thin wrapper for `qunfold` methods, which allows users to combine the functionalities of QuaPy and our package.

We introduce the usage of `qunfold` in Sect. 2 and revisit its conceptual background in Sect. 3. Sect. 4 demonstrates the performance of our package before Sect. 5 concludes with prospective extensions.

## 2 Usage

The package is easily installed via `pip` and its quantification methods are used like classifiers from `scikit-learn`. These design choices result in a seamless access for newcomers of quantification, as illustrated in Listing 1.

```
from qunfold import ACC # Adjusted Classify and Count
from sklearn.ensemble import RandomForestClassifier

acc = ACC( # use OOB predictions for training the quantifier
    RandomForestClassifier(oob_score=True)
)
acc.fit(X_trn, y_trn) # fit to training data
p_hat = acc.predict(X_tst) # estimate a prevalence vector $\hat{\mathbf{p}} \in \mathbb{R}^C$
```

Listing 1: A minimal example where the quantification method Adjusted Classify and Count (ACC) [12] predicts the class prevalences of a testing sample.

Beyond the existing methods of quantification and unfolding, users have the opportunity to compose new methods from existing loss functions and data representations. This opportunity also includes the combination of multiple loss

[1] https://github.com/mirkobunse/qunfold

```
# the ACC loss, regularized with strength 0.01 for ordinality
loss = TikhonovRegularized(LeastSquaresLoss(), 0.01)

# the original data representation of ACC with 10-fold cross-validation
transformer = ClassTransformer(CVClassifier(LogisticRegression(), 10))

# the ordinal variant of ACC, wrapped for being used in QuaPy
ordinal_acc = QuaPyWrapper(GenericMethod(loss, transformer))
```

Listing 2: The ordinal variant [6] of ACC is composed of the original ACC loss, a regularization term, and the original data representation of ACC. Finally, this variant is wrapped for being used in QuaPy.

terms, like regularizers, through a `CombinedLoss` type. Listing 2 conveys, as an example of composition, the creation of an ordinal variant of ACC.

The creation of novel data representations only requires implementing the `fit_transform` and `transform` methods of the `AbstractTransformer` type. Novel loss functions are easily implemented through `jax` [13], a package which automatically differentiates the loss while complying to the well-known `numpy` API. All of the above aspects are thoroughly documented, illustrated through examples, and tested in a continuous integration pipeline.

## 3 Background

We intend to predict $\mathbf{p} \in \mathbb{R}^C$, the class prevalences of an unlabeled data sample $\mathrm{D} \in \mathcal{X}^m$. For this purpose, we have a labeled training set $\bigcup_{i=1}^{C} \mathrm{D}_i$ where $\mathrm{D}_i$ contains the data items of the $i$-th class. The composition of quantification methods is enabled through the observation [5,11] that many methods estimate $\mathbf{p}$ by solving a system of linear equations

$$\begin{aligned} \mathbf{q} &= \mathbf{M}\mathbf{p} \qquad (1) \\ \text{where} \quad [\mathbf{q}]_i &= \frac{1}{|\mathrm{D}|} \sum_{\mathbf{x} \in \mathrm{D}} [f(\mathbf{x})]_i \end{aligned}$$

is a mean embedding of D, which employs a feature transformation $f : \mathcal{X} \to \mathbb{R}^F$. Here, the matrix $\mathbf{M} \in \mathbb{R}^{F \times C}$ with entries

$$[\mathbf{M}]_{ji} = \frac{1}{|\mathrm{D}_j|} \sum_{\mathbf{x} \in \mathrm{D}_j} [f(\mathbf{x})]_i$$

represents the mean embedding of each class in the training set.

Finding a solution $\hat{\mathbf{p}}$ for Eq. 1 requires the minimization of a loss function $\mathcal{L} : \mathbb{R}^C \to \mathbb{R}$, which reflects the goodness of $\hat{\mathbf{p}}$. Hence, quantification methods of the above kind are defined through a loss function and a feature transformation.

### 3.1 Unconstrained Soft-Max Optimization

Given a loss function and a feature transformation, a recent proposal [4] for solving Eq. 1 is

$$\begin{aligned} \hat{\mathbf{p}} \; &= \; \mathrm{softmax}\big(\mathbf{l}^*\big) \\ \text{where} \quad \mathbf{l}^* &= \operatorname*{arg\,min}_{\mathbf{l} \in \mathbb{R}^C} \mathcal{L}\big(\, \mathrm{softmax}(\mathbf{l}); \mathbf{q}, \mathbf{M}\,\big) \end{aligned} \tag{2}$$

is a vector of latent quantities. Here, the output of the soft-max operator is $[\mathrm{softmax}(\mathbf{l})]_i = \exp([\mathbf{l}]_i) / (\sum_{j=1}^C \exp([\mathbf{l}]_j))$, which ensures that any $\hat{\mathbf{p}}$ is a legitimate estimate of class prevalences. We regard an estimate as being legitimate if it represents a probability density function, i.e., if $[\hat{\mathbf{p}}]_i \geq 0 \; \forall \, i$ and $\sum_i [\hat{\mathbf{p}}]_i = 1$. The latent quantities can be interpreted as scaled log-probabilities of the classes.

In `qunfold`, we establish the uniqueness of $\mathbf{l}^*$ by fixing the first dimension to the constant value $[\mathbf{l}]_1 = 0$. Thereby, we minimize $\mathcal{L}$ only over $(n-1)$ actual variables in $\mathbf{l}$. This reduction of dimensionality comes without sacrificing the optimality of $\mathbf{l}^*$; it only defines the scaling of the latent quantities.

### 3.2 Out-of-Bag Training of Quantifiers

The estimation of $\mathbf{M}$ (see Eq. 1) requires a labeled training set. In case of a supervised feature transformation $f$, like the `ClassTransformer` of ACC, this estimation should not use the same training data as $f$; otherwise, biases of $f$ will hardly be corrected by the quantification method. One possibility of diversifying the training data of $f$ and $\mathbf{M}$ is through cross-validation [12]. Here, $f$ is trained with the training folds and $\mathbf{M}$ is trained with the test predictions. We implement this training strategy in the `CVClassifier` class (revisit Listing 2).

In addition, we implement a similar technique which builds on bagging estimators [3]. Here, $f$ is trained with the training folds and $\mathbf{M}$ is trained with the out-of-bag predictions of the estimator. The advantage of bagging over cross-validation is that bagging ensembles, like random forests, can be trained at no extra cost. For this strategy, the bagging classifier can be used directly, without the need for a meta-classifier (revisit Listing 1).

### 3.3 Existing Loss Functions and Feature Representations

Our package implements several existing methods in terms of their loss functions and feature representations, which are listed in Tab. 1. The modular design of our package enables compositions of novel methods from the existing components.

In case of HDx and HDy [14], we have replaced the original loss function with a surrogate loss that is better suited for numerical optimization. The original loss, which is the average of feature-wise (or class-wise) Hellinger distances, is problematic because it lacks twice differentiability and, hence, complicates numerical optimizations. As a twice differentiable surrogate, we therefore employ the average of squared Hellinger distances. This `HellingerSurrogateLoss` behaves similar to the original loss, while facilitating numerical optimizations.

**Table 1.** Existing methods in terms of their loss functions and feature transformations.

| method | loss function | feature transformation |
|---|---|---|
| ACC [12] | `LeastSquaresLoss()` | `ClassTransformer(*, is_probabilistic=False)` |
| PACC [1] | `LeastSquaresLoss()` | `ClassTransformer(*, is_probabilistic=True)` |
| HDx [14] | `HellingerSurrogateLoss()` | `HistogramTransformer(*)` |
| HDy [14] | `HellingerSurrogateLoss()` | `HistogramTransformer(*, preprocessor=ClassTransformer(*))` |
| EDx [15] | `EnergyLoss(*)` | `DistanceTransformer(*)` |
| EDy [7] | `EnergyLoss(*)` | `DistanceTransformer(*, preprocessor=ClassTransformer(*))` |
| RUN [2] | `TikhonovRegularized( BlobelLoss(), *)` | any `AbstractTransformer` |
| CC [12] | `None` | `ClassTransformer(*, is_probabilistic=False)` |
| PCC [1] | `None` | `ClassTransformer(*, is_probabilistic=True)` |

## 4 Performance

We evaluate the performance of our package on the public data set [9] of the LeQua2022 competition [10]. This dataset, which contains 28 classes, constitutes a gold-standard benchmark for multi-class text quantification. We employ the vectorial representation of the data and a logistic regression classifier with $C \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}\}$, where the best $C$ is chosen separately for each quantification method on hold-out validation samples. A classifier of this kind obtained winning performances during the competition [17]. For HDy, we optimize the number of bins per class on hold-out data over $B \in \{2, 4, 6\}$. We compare the results of `qunfold` and QuaPy in terms of the mean absolute error (AE) and in terms of the mean relative absolute error (RAE). We omit HDx, EDx, CC, and PCC because we deem these methods unsuitable for text quantification. We also omit several methods that are available in QuaPy but not (yet) in `qunfold`.

**Table 2.** Performance comparison between `qunfold` and QuaPy. The performances are measured in terms of two error metrics, AE and RAE. The performance of the best implementation, for each method and metric, is printed in **boldface**. An asterisk (∗) indicates that a method is missing from QuaPy v.0.1.7.

| method | AE (↓) | | RAE (↓) | |
|---|---|---|---|---|
| | QuaPy | `qunfold` | QuaPy | `qunfold` |
| ACC | 0.0190±0.0045 | **0.0164±0.0046** | 1.5380±1.4460 | **1.2553±1.1763** |
| PACC | 0.0197±0.0050 | **0.0119±0.0034** | 1.7070±2.0091 | **0.9594±0.8342** |
| HDy | 0.0163±0.0042 | **0.0143±0.0041** | 1.3634±1.2062 | **1.1319±1.0803** |
| EDy | ∗ | **0.0125±0.0036** | ∗ | **1.1856±1.1080** |
| RUN | ∗ | **0.0165±0.0046** | ∗ | **1.2305±1.1478** |

The results of our evaluation are displayed in Tab. 2. They convey that the methods from qunfold beat the corresponding implementations from QuaPy, which is the state-of-the-art package for quantification. We attribute this outcome to the powerful soft-max optimization technique that our package leverages. The current version of Quapy[2], in contrast, employs the pseudo-inversion method for ACC and PACC and constrained optimization for HDy, all of which have been shown to yield inferior performances [4]. We note, however, that our soft-max optimization is computationally more expensive than the pseudo-inverse method.

## 5 Conclusion

We have presented qunfold, a highly performant and composable implementation of several quantification and unfolding methods. This implementation leverages two recent findings: first, that many quantification methods consist of a loss function and a data representation, which can be reassembled in arbitrary ways; second, that these methods can be optimized through a soft-max operator. Further improvements of our implementation are a surrogate loss for HDx and HDy and an optional out-of-bag training of quantifiers. These features lead to performances that beat QuaPy, the current state-of-the-art implementation for quantification methods. We recommend qunfold to anyone who is looking for composability or for strong baseline methods.

In the future, we are planning to extend our package with additional loss functions and data representations. We also conceive novel features, like ensembling and automatic compositions of methods, as valuable extensions.

**Acknowledgements** This work was partly funded by the Federal Ministry of Education and Research of Germany and the state of North Rhine-Westphalia as part of the Lamarr Institute for Machine Learning and Artificial Intelligence.

We also thank the reviewers of our LQ 2022 publication [4] for pointing out that the solution of Eq. 2 is unique if $[\mathbf{l}]_1 = 0$ is fixed. This observation has substantially improved our implementation.

## References

1. Bella, A., Ferri, C., Hernández-Orallo, J., Ramírez-Quintana, M.J.: Quantification via probability estimators. In: Int. Conf. on Data Mining. pp. 737–742. IEEE (2010). https://doi.org/10.1109/ICDM.2010.75
2. Blobel, V.: Unfolding methods in high-energy physics experiments. Tech. rep., CERN (1985). https://doi.org/10.5170/CERN-1985-009.88, https://cds.cern.ch/record/157405/files/p88.pdf
3. Breiman, L.: Bagging predictors. Mach. Learn. **24**(2), 123–140 (1996). https://doi.org/10.1007/BF00058655

[2] https://github.com/HLT-ISTI/QuaPy/releases/tag/0.1.7

4. Bunse, M.: On multi-class extensions of adjusted classify and count. In: Int. Worksh. on Learn. to Quantify: Meth. and Appl. pp. 43–50 (2022), https://lq-2022.github.io/proceedings/CompleteVolume.pdf
5. Bunse, M.: Unification of algorithms for quantification and unfolding. In: Worksh. on Mach. Learn. for Astropart. Phys. and Astron. pp. 459–468. Gesellschaft für Informatik e.V. (2022). https://doi.org/10.18420/INF2022_37
6. Bunse, M., Moreo, A., Sebastiani, F., Senz, M.: Ordinal quantification through regularization. In: Europ. Conf. on Mach. Learn. and Knowl. Discov. in Databases. pp. 36–52. Springer (2023). https://doi.org/10.1007/978-3-031-26419-1_3
7. Castaño, A., González, P., González, J.A., del Coz, J.J.: Matching distributions algorithms based on the earth mover's distance for ordinal quantification. IEEE Trans. on Neur. Netw. and Learn. Syst. pp. 1–12 (2022). https://doi.org/10.1109/tnnls.2022.3179355
8. Esuli, A., Fabris, A., Moreo, A., Sebastiani, F.: Learning to Quantify, The Inform. Retr. Series, vol. 47. Springer (2023). https://doi.org/10.1007/978-3-031-20467-8
9. Esuli, A., Moreo, A., Sebastiani, F.: Learning to quantify: LeQua 2022 datasets (2021). https://doi.org/10.5281/zenodo.6546188
10. Esuli, A., Moreo, A., Sebastiani, F.: LeQua@CLEF2022: Learning to quantify. In: Adv. in Inform. Retr. pp. 374–381. Springer (2022). https://doi.org/10.1007/978-3-030-99739-7_47
11. Firat, A.: Unified framework for quantification (2016), http://arxiv.org/abs/1606.00868
12. Forman, G.: Quantifying counts and costs via classification. Data Mining and Knowl. Discov. **17**(2), 164–206 (2008). https://doi.org/10.1007/s10618-008-0097-y
13. Frostig, R., Johnson, M.J., Leary, C.: Compiling machine learning programs via high-level tracing. Syst. for Mach. Learn. **4**(9) (2018), http://github.com/google/jax
14. González-Castro, V., Alaíz-Rodríguez, R., Alegre, E.: Class distribution estimation based on the Hellinger distance. Inform. Sci. **218**, 146–164 (2013). https://doi.org/10.1016/j.ins.2012.05.028
15. Kawakubo, H., du Plessis, M.C., Sugiyama, M.: Computationally efficient class-prior estimation under class balance change using energy distance. IEICE Trans. Inform. Syst. **99-D**(1), 176–186 (2016). https://doi.org/10.1587/transinf.2015EDP7212
16. Moreo, A., Esuli, A., Sebastiani, F.: QuaPy: A Python-based framework for quantification. In: Int.Conf. on Inform. and Knowl. Management. pp. 4534–4543. ACM, New York, NY, USA (2021). https://doi.org/10.1145/3459637.3482015
17. Senz, M., Bunse, M.: DortmundAI at LeQua 2022: Regularized SLD. In: Conf. and Labs of the Eval. Forum. vol. 3180, pp. 1911–1915. CEUR (2022), http://ceur-ws.org/Vol-3180/paper-152.pdf

# MC-SQ: A Highly Accurate Ensemble for Multi-class Quantification (Extended Abstract)

Zahra Donyavi[1], Adriane Serapião[1,2], and Gustavo Batista[1]

[1] School of Computer Science and Engineering, University of New South Wales,
UNSW, Sydney, Australia, 2052
{z.donyavi, g.batista}@unsw.edu.au
[2] São Paulo State University
adriane.serapiao@unesp.br

**Abstract.** Quantification research proposes methods to estimate the class distribution in an independent sample. Many areas, such as epidemiology, sentiment analysis, political research and ecological surveillance, rely on quantification methods to estimate aggregated quantities. For instance, epidemiologists are often concerned with the dynamics of the number of disease cases across space and time. Thus, while classification predicts individual subjects, quantification is the class of methods that directly estimate the number of cases. Quantification is a thriving research area, and the community has proposed several approaches in the last decade. Nevertheless, most quantification research has focused on binary-class quantifiers, expecting these approaches to extend to multi-class using the one-versus-all (OVA) approach. However, enough empirical evidence indicates that OVA multi-class quantifiers' performance is subpar. This paper has two main contributions. First, we demonstrate why OVA quantifiers are doomed to underperform in multi-class settings due to a distribution shift they cannot handle. Second, we propose a new class of quantifiers based on ensemble learning that boosts the performance of the base quantifiers in the binary and, more importantly, multi-class settings. In one of the most comprehensive experimental setups ever attempted in quantification research, we show that our ensembles are the best-performing quantifiers compared with 33 state-of-the-art (single and ensemble) quantifiers and rank first in a recent quantification competition.

**Keywords:** Quantification · prevalence estimation · class probability estimation · ensembles · multi-class · machine learning.

## 1 Introduction

*Quantification* is the Machine Learning task that proposes methods to estimate the class distribution in an independent sample [8]. It finds applications in areas where we are more interested in understanding the behavior of groups than predicting individual cases. One well-known example is sentiment analysis, in which we often want to understand trends, such as the percentage of users making positive comments about a personality, brand, or product in a given period.

*Classify & Count* (CC) is the simplest quantifier. It is a direct application of classification to solve quantification problems. However, despite its simplicity, CC is a biased quantifier. Forman [9] reveals that CC contains a systematic bias. For an imperfect classifier, the CC method will underestimate the true proportion of positives $\hat{p}$ in a test set for $\hat{p} > p^*$ and overestimate for $\hat{p} < p^*$, where $p^*$ is the particular proportion at which the CC method estimates correctly. This flaw has motivated a thriving community of researchers to develop novel quantifiers that provide accurate class estimates for the whole spectrum of class distributions.

So far, the quantification community has heavily focused on developing binary quantifiers. The idea is that those binary quantifiers can be extended to multi-class problems using the *one-versus-all* (OVA) approach. An OVA quantifier performs independent binary quantifications for each class versus all others and then normalizes the final estimates to sum to 100%.

However, recent empirical evidence has shown that OVA quantifiers' performance is subpar in multi-class problems [28]. Even more worrisome, multi-class quantifiers perform better than OVA quantifiers but just by a small margin. In this paper, we make two contributions to multi-class quantification. ($i$) For the first time, we explain why OVA quantifiers underperform in multi-class problems. ($ii$) We propose a simple ensemble approach that boosts the performance of existing multi-class quantifiers.

As contributions of this paper, we show that modeling a multi-class quantification problem with a set of OVA datasets induces a distribution shift in $p(\mathbf{x}|y)$. However, existing quantification methods assume that $p(\mathbf{x}|y)$ is constant. Therefore, these methods are doomed to perform poorly in multi-class settings. This finding will sound counter-intuitive to a significant part of the Machine Learning community since OVA is one of the *de-facto* approaches to converting binary classifiers into multi-class.

We show that a simple ensemble can significantly improve the performance of existing quantifiers. In a comprehensive empirical comparison with 33 state-of-the-art quantifiers and 40 datasets, our proposals are the best-performing quantifiers for both binary and multi-class datasets. In addition, our methods rank first in a recent quantification competition.

This paper is organized as follows. Section 2 introduces the basic concepts and the notation used throughout this paper. Section 3 presents the related work, briefly describing the 33 quantifiers used in our experiments. Section 4 discusses the limitations of using the OVA approach for multi-class quantification. Section 5 describes the ensemble approach that constitutes our main technical contribution. Section 6 presents the experimental results in both multi-class and binary quantification settings as well as the LeQua 2022 competition. Finally, Section 7 concludes our work and presents directions for future work.

## 2 Background

This section introduces the mathematical notation and fundamental concepts employed throughout this work.

A *dataset* is a collection of $N$ examples such that $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$. Each $\mathbf{x}_n \in \mathcal{X}$ is a vector with $M$ attributes, and $y_n \in \mathcal{Y} = \{1, 2, \ldots, C\}$ is the class label associated with $\mathbf{x}_n$.

We can create a predictive model with the dataset $\mathcal{D}$. The primary goal of *classification* is to predict the class label of each example using the covariates. Hence, the classifier is a predictive model $h_c$ trained from $\mathcal{D}$ such that:

$$h_c : \mathcal{X} \rightarrow \{1, 2, \ldots, C\}$$

In this paper, we are interested in quantification problems. We define a *quantifier* as a supervised model learned from a dataset $\mathcal{D}$ to estimate the class prevalence in a test sample. Therefore, the quantifier is a function $h_q$ such that:

$$h_q : \mathcal{S} \rightarrow [0, 1]^C$$

where $\mathcal{S}$ represents all possible sets of samples under the representation $\mathcal{X}$. From an unlabeled set $\mathbf{S} \in \mathcal{S}$, $h_q$ outputs a vector $\hat{\mathbf{p}} = \langle \hat{p}_n \rangle_{n=1}^{C}$, where $\hat{p}_n$ is the estimate of the proportion of the class $n$, such that $\sum_{n=1}^{C} \hat{p}_n = 1$. The aim is to estimate the predicted ratios $\hat{\mathbf{p}}$ as close as possible to the true ratios $\langle p(n) \rangle_{n=1}^{C}$ of the unlabeled set $\mathbf{S}$.

Comparing the functions $h_c$ and $h_q$, we can notice the similarities and differences in classification and quantification. These two tasks use the same data representation, a labeled tabular dataset $\mathcal{D}$, to induce their models. However, the objectives are distinct. While a classifier outputs a class label for each input instance, a quantifier outputs a class distribution estimate for a given *sample* of examples.

In both classification and quantification, the examples are *independent* of each other. Thus, the occurrence of one instance does not change the probability of the other instances. However, training and test samples are not *identically distributed* in quantification problems, as we expect that the class distribution will change.

Let us introduce one example to make these ideas more concrete. In the case of sentiment analysis, we can create a dataset of, say, tweets and label them as $\{\oplus, \ominus, \odot\}$, representing the positive, negative and neutral classes, respectively. A classifier will take a single tweet as input and output a unique class label. In contrast, a quantifier will take a set of tweets, such as the tweets from the last 24 hours that match the search criteria, and will output a vector $\hat{\mathbf{p}} = \langle \hat{p}_\oplus, \hat{p}_\ominus, \hat{p}_\odot \rangle$. In this example, $\hat{p}_\oplus$ is the estimated percentage of users expressing positive sentiment.

Two final observations about quantifiers. First, we can trivially convert the class probability estimates into counts by multiplying these probabilities with the test sample size. Thus, quantifiers are also known as *counters*. Second, the test sample size can vary according to the application. In the example of tweet sentiment analysis, we can have a test sample with tweets from the last hour, day, week, or month. Thus, it is essential to consider different test sample sizes when assessing quantifiers [19].

We conclude this section by defining a scorer as several quantifiers use them as an intermediate step in their computation. A *scorer* is a model induced from $\mathcal{D}$ such that:

$$h_s : \mathcal{X} \rightarrow \mathbb{R}^C$$

A scorer outputs a vector $\mathbf{s} = \langle s_n \rangle_{n=1}^{C}$ of real values called *scores* for a given input example. Each score $s_n$ has a positive correlation with the posterior probability of the class $y_n$, i.e., $p(y_n|\mathbf{x})$. Accordingly, a higher $s_n$ value means an increased chance for an example belonging to the class $y_n$.

## 3 Related Work

This section reviews all existing quantification algorithms in the literature. Due to lack of space, we briefly describe the 29 single quantifiers and one ensemble approach and provide relevant references for readers interested in further details. We organize this section according to the taxonomy proposed by [12], resulting in three groups of methods:

**Classify, count & correct:** These methods use a classifier to classify each instance and then count them by the class label. They often include an additional step that applies a correction to the counts.

**Distribution matching:** These methods parametrically model the training distribution and later search for the parameters that provide the best match against the test distribution.

**Adaptations of classification algorithms:** These methods adapt classification algorithms, transforming them into quantifiers.

We conclude this section by describing the only ensemble quantification approach in the literature.

### 3.1 Classify, Count & Correct

is a class of methods that count the classes using a classifier and apply a correction factor to obtain the final estimate.

**CC (Multi-class):** Classify & Count (CC) uses a classifier to count the class predictions for each label. Forman [9] shows that CC is a biased quantifier.

**ACC (Binary):** Adjusted Classify & Count (ACC) corrects the output of the CC method by employing the following correction factor:

$$p_{ACC}(y = \oplus|\mathbf{S}) = \frac{p_{CC}(y = \oplus|\mathbf{S}) - fpr}{tpr - fpr} \tag{1}$$

where $p_{CC}(y = \oplus|\mathbf{S})$ is the positive class prevalence provided by CC in the test set $\mathbf{S}$, and $fpr$ is the false-positive and $tpr$ is the true-positive rates often estimated in the training set using cross-validation.

**PCC and PACC (Binary):** Probabilistic Classify & Count (PCC) and Probabilistic Adjusted Classify & Count (PACC) [3] assume that probabilities have richer information than the label predictions of the classifier. PCC is a counterpart of the CC method, averaging the probabilities to estimate the class prevalence. Similar to ACC, PACC corrects the estimate of PCC using Equation 1. Since the class distribution influences the calibration of the classifiers, PCC and PACC approaches suffer from a *chicken-and-egg* problem [9].

**GACC and GPACC (Multi-class):** The Generalized Adjusted Classify & Count (GACC) and Generalized Probabilistic Adjusted Classify & Count (PACC) are multi-class generalizations of ACC and PACC, respectively [7]. These methods build the following system of equations and solve it via constrained least-squares regression:

$$p(h_c(\mathbf{S}) = n) = \sum_{i=1}^{C} p(h_c(\mathbf{S}) = n|y = i)p(y = i)$$

for $n = 1 : C$. As $P(h_c(\mathbf{S}) = n|y = i)$ is unknown, we estimate it using cross-validation in the training data.

**FM (Multi-class):** Friedman's method (FM) [11] also builds a system of equations. However, unlike GPACC, FM only considers a subset of the test instances with probabilities above the training class prevalences.

**X, MAX, T50 (Binary):** These methods search for different classification thresholds aiming for more reliable estimates for $fpr$ and $tpr$ [10]. X selects the threshold value where the difference between $1 - tpr$ and $fpr$ is minimal. MAX chooses the threshold value that maximizes the denominator in Equation 1. T50 selects the threshold where $tpr \approx 50\%$.

**MS (Binary):** Median Sweep (MS) [10] returns the median of several applications of the ACC method for a range of classification thresholds. Each threshold estimates the $tpr$ and $fpr$ using cross-validation and then applies ACC correction. We use a variant with a subset of the thresholds that produce a denominator in Equation 1 greater than 0.25.

### 3.2 Distribution Matching

is a class of methods that parametrically model the training distribution and then search for the parameter that best matches the training and test distributions.

**FMM (Binary):** Forman's Mixture Method (FMM) [8] models the positive and negative class distributions using cumulative distribution functions (CDFs). As modeling $p(\mathbf{S}|y)$ is often difficult, this method uses the score distribution, i.e., $P(h_s(\mathbf{S})|y)$, which is more amenable since it is a set of unidimensional real values. FMM models the training scores from the positive and negative classes independently, as well as the test scores using CDFs. Then, it compares the test CDF with a mixture of positive and negative class CDFs while varying a mixture parameter. Forman uses the Probability-Probability plot to measure the difference between the training and test CDFs and returns the parameter whose curve produces the minimum difference as the positive class prevalence.

**HDx and HDy (Binary):** Gonzalez-Castro et al. [13] propose a mixture method similar to FMM that uses histograms to represent data distributions and the Hellinger Distance (HD) to compare those histograms. A weighted sum of the positive and negative class histograms provides a mixture that is compared with the test histogram. HDy uses scores to represent the distributions. Conversely, HDx operates over each feature independently and averages the HD values. The following equation describes the search performed by HDy:

$$p_{\text{HDy}}(y = \oplus | \mathbf{S}^{\oplus}, \mathbf{S}^{\ominus}, \mathbf{S}^{\odot}) = \\ \underset{0 \leq \alpha \leq 1}{\arg\min} \left\{ \text{HD}\left( \alpha H[\mathbf{S}^{\oplus}] + (1-\alpha) H[\mathbf{S}^{\ominus}], H[\mathbf{S}^{\odot}] \right) \right\}$$

where HD is the Hellinger distance and $H[\cdot]$ is a transformation of scores into a histogram representation, and $\mathbf{S}^{\oplus}$, $\mathbf{S}^{\ominus}$, and $\mathbf{S}^{\odot}$ are the positive, negative and test scores, respectively.

**DyS (Binary):** Distribution y-Similarity (DyS) [18] is a framework of mixture models method for binary quantification, based on HDy, that supports the use of different distance measures besides HD.

**ED (Multi-Class):** Similar to HDx, Energy Distance Minimization (ED) uses the actual features of the input space to model the distributions. But instead of HD, ED tries to minimize the Energy distance measure as described in [17].

**Readme (Multi-class):** Readme [15] is similar to HDx, as it also operates directly over features instead of using a classifier. Readme models the feature distribution by counting co-occurrences. Thus, for continuous attributes, this method requires feature discretization. Only a subset is used in an optimization problem solved by general least-squares regression.

**EMQ (Multi-class):** The Expectation-Maximization Quantifier (EMQ) [27] uses the well-known Expectation-Maximization (EM) algorithm to adjust the output of probabilistic classifiers for changes in the class distribution.

### 3.3 Adaptations of Classification Algorithms

is a class of methods that adapt an existing classification algorithm to quantification.

**QT and QT-ACC (Multi-Class/Binary):** Quantification trees (QT) [20] is a quantification method based on a decision tree algorithm. The main difference between QT and classification trees is the node-splitting criterion. QT employs a criterion suitable for the quantification task instead of a measure based on information theory used for classification tasks. QT-ACC is similar to QT with the additional application of the ACC correction (Equation 1).

- **PWK (Multi-class):** The Proportion-Weighted $k$-Nearest Neighbor algorithm (PWK) [2] is an adaptation of the $k$-Nearest Neighbor (NN) algorithm to quantification using a weighting scheme which applies less weight on neighbors from the majority class.
- **CDE (Binary):** The Class Distribution Estimation (CDE) [30] applies the cost-sensitive classification principle to update the classifier according to the class distribution change between the training and test sets. CDE is an iterative algorithm that re-trains the classifier according to the cost ratio calculated using the distribution mismatch ratio with the previous iteration's estimate.
- **SVM-Q and SVM-K (Binary):** These methods use the SVM-perf implementation of Support Vector Machines (SVM) optimized for multivariate loss functions [16]. SVM-Q uses the Q-measure [1], and SVM-K uses the Kullback-Leibler Divergence [5].

### 3.4 Ensembles for quantification.

An ensemble is a set of individually trained models whose predictions are combined to forecast novel instances, often providing more accurate results than their base models [23].

Ensembles are extremely common in classification, but quantification has not dedicated much attention to this research venue. To the best of our knowledge, the use of ensembles in quantification is restricted to two articles [25,26].

In these papers, the authors explore the drift in $p(y)$ as a factor to generate diversity for the base classifiers. Therefore, they propose to train each base classifier using a different class prevalence. They sample the dataset using random sampling with replacement to vary $p(y)$ while ensuring that $p(\mathbf{x}|y)$ remains constant, a common assumption in quantification learning. The proposal uses the same pair of base classifier and quantifier for all samples and aggregates the individual predictions in a final predicted class prevalence.

We refer to this method as the *class-prevalence ensemble* (CPE) to avoid confusion with the approach proposed in this paper.

## 4 Multi-class Quantification

Forman [9] was the first to advocate using OVA for multi-class problems. An OVA quantifier performs independent binary quantifications for each class versus all others and then normalizes the final estimates to sum to 100%.

More recently, Schumacher et al. [28] have assessed 29 existing binary and multi-class quantifiers in a comprehensive evaluation involving 40 datasets. They conclude that binary quantifiers allied with OVA "*showed mediocre performance in the multi-class case.*"

What is intriguing here is why this is the case and which factors can make an accurate binary quantifier inaccurate for a multi-class problem transformed into a binary-class dataset with OVA. Schumacher et al. [28] hypothesize that the issue comes with the OVA normalization step. This section demonstrates that the OVA quantifiers perform poorly in multi-class settings due to a change in $p(\mathbf{x}|y)$.

We offer an intuitive explanation for the OVA issues using an example. Suppose we have a multi-class problem with only three classes with red, green, and blue labels. Blue is chosen as the positive ($\oplus$) class during one of the OVA executions, while red and green receive the negative label ($\ominus$) (see Figure 1). As we have a quantification problem, we expect the prevalence of red and green classes to vary independently, i.e., reds' prevalence can increase while greens' decreases and vice-versa. However, as the OVA quantifier sees the instances of these two classes as a single negative class, these prevalence drifts lead to a change in $p(\mathbf{x}|y)$. An intuitive way to realize this is to notice that an increase in reds' prevalence leads to a more complex separation of the positive and negative classes since the red class is closer to the blue class. In contrast, increasing green prevalence leads to an easier separation.

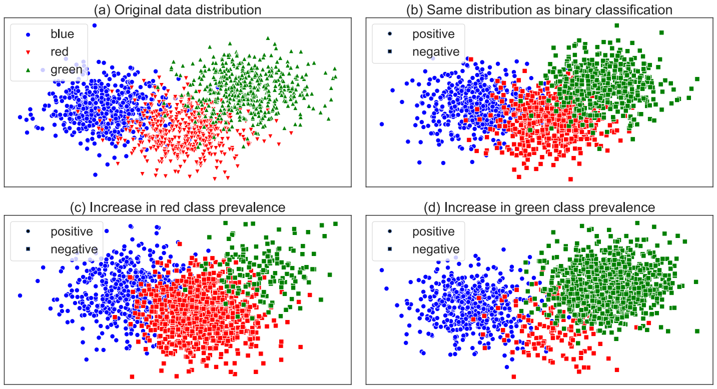

**Fig. 1.** A hypothetical three-class dataset $(a)$ transformed into a binary-class problem $(b)$ with class blue as positive. The change in the prevalence of the classes red and green causes a concept drift in $p(\mathbf{x}|y = \ominus)$, making the binary classes harder $(c)$ or easier $(d)$ to discriminate.

Suppose we characterize quantification as a $Y \to X$ problem [6]. The joint probability distribution is factored as $p(\mathbf{x}, y) = p(\mathbf{x}|y)p(y)$. We expect that the class distribution, $p(y)$, will change, as this is the primary motivation of the quantification. However, the quantification literature assumes that the conditional distribution $p(\mathbf{x}|y)$ remains constant. For instance, classify, count and adjust methods estimate the class errors $(p(h_c(\mathbf{S}) = n|y = i))$ on the training set, and the distribution matching methods try to model $p(\mathbf{S}|y)$ or $p(h_s(\mathbf{S})|y)$ using an approximation with training data.

A change in $p(\mathbf{x}|y)$ and $p(y)$ for $Y \rightarrow X$ problems is hardly addressed in the literature, as this problem is so complex it is considered impossible to solve [21]. Therefore, OVA quantification approaches are doomed to underperform, as observed in the literature [28].

## 5 Proposed Approach

This section presents an ensemble method that is our main technical contribution. We start discussing our main requirements:

**Multi-class** As we have discussed in Section 4, a binary-class method will not perform well on multi-class problems. Thus, the solution must be intrinsically multi-class since it will naturally apply to any number of labels.

**Accurate** One of the conclusions of experimental comparisons such as [28] is that multi-class quantification is a difficult problem, as both OVA and multi-class quantifiers perform poorly. Thus, we look for a significantly more accurate solution than current single and ensemble methods.

**Simple** The method must be simple as our primary motivation is to demonstrate the limitation of the current OVA approach and the directions for future research in multi-class quantification. We hope the community will further develop these ensemble approaches by looking for more complex (and hopefully accurate) variations.

**Hyperparameter-free** Our approaches must not add new hyperparameters beyond those inherited from the base classifiers and quantifiers. Our performance improvement should not originate from an extensive hyperparameter search.

Figure 2 illustrates our proposal. It consists of an ensemble of $n$ pairs of classifier and quantifier. We vary the base classifier to provide diversity and fix the base quantifier. Therefore, we name our approach *multiple-classifier, single-quantifier*, or MC-SQ.

We set the number of pairs of classifier-quantifier as seven to eliminate parameters. We employ the following classifiers in our experiments: Random Forest (RF), Nave Bayes (NB), Gradient Boosting (GB), Support Vector Machines (SVM)[3], Linear Discriminant Analysis (LDA), Light Gradient Boosting Machines (LGBM), and Logistic Regression (LR). The motivation for selecting those algorithms is that they represent different learning paradigms and are often shortlisted as the most successful approaches in Machine Learning.

Finally, in our experiments, we employ the quantifiers EMQ, FM, GACC and GPACC as these were shortlisted as the best performing multi-class quantifiers in [28]. We provide further details in the next section.

[3] The SVM implementation in sklearn [24] uses one-versus-one to implement multi-class classifiers. This does not impact our ensembles; they only use the scores these classifiers provide.

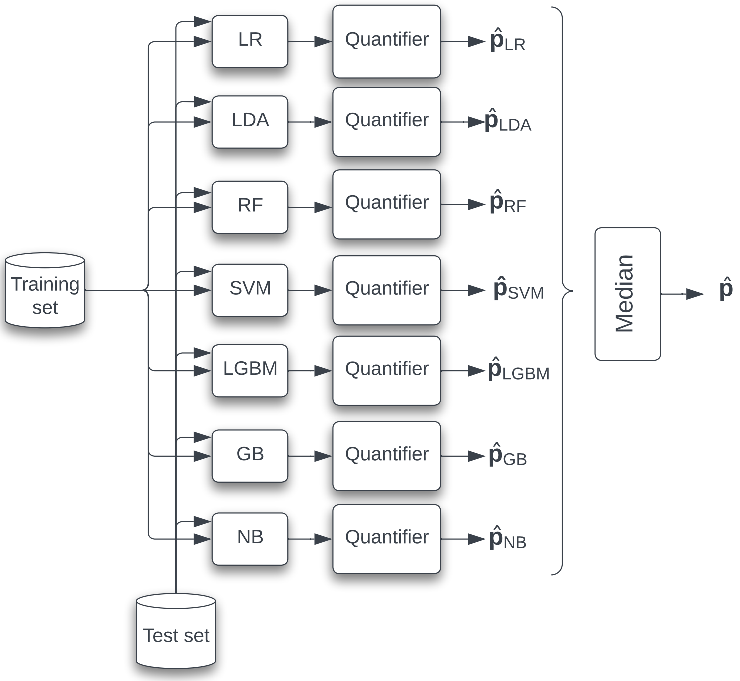

**Fig. 2.** Schematic of the proposed Multi-Classifier, Single-Quantifier ensemble approach.

## 6 Experimental Evaluation

This section details the experimental setup and assessment results. We include some ablation studies that provide insights into our method's design decisions.

### 6.1 Experimental Setup.

We are strongly committed to reproducible research. Therefore, we decided to use the same experimental design as [28], allowing direct comparison with those results. For the base quantifiers, we use the implementation provided in their paper[4]. Also, we created a paper website to store code, figures, tables, and detailed results perpetually[5].

We compare the results obtained by our ensemble methods with the single quantifiers and the class-prevalence ensembles from [26]. We use the ensemble

[4] The only exception is the HDy method which we found to differ significantly from the method described in [13]. In this case, we use our implementation.

[5] `https://sites.google.com/view/mc-sq`.

implementation provided in QuaPy [22]. As suggested in [22], we produce 50 different training samples with various distributions and apply Linear Regression as the base classifier to get scores for the 50 samples. A base quantifier is applied over the scores, producing 50 quantifiers for each class. The predicted prevalence is the normalized (to sum to 100%) average of the prevalences for each class label. To generate comparable results, we execute our ensembles and the class-prevalence ensembles over the same set of base quantifiers.

The experiments involve 40 benchmark datasets, 23 binary and 17 multi-class. We briefly describe the datasets' main characteristics on the paper's website. We use Absolute Error (AE), Equation 2, as the primary measure to assess our results. AE has several attractive features. For instance, it is easy to interpret and restrained in the interval $[0, 2]$ independently of the number of classes [29].

$$AE(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{C} \sum_{n=1}^{C} |\hat{p}_n - p_n| \tag{2}$$

The experimental setup follows the Artificial-Prevalence Protocol (APP) [14]. It consists of splitting a classification dataset into training and test sets. The test set class prevalence is artificially manipulated through sub-sampling, creating multiple test set samples. The idea is to create test samples with class prevalences that differ significantly from the training class distribution. We train the quantifiers with the training set, assess them in each test sample, and report the average AE across all test sets. We refer to [28] for further details about the experimental setup.

### 6.2 Experimental Results.

Table 1 shows the numerical results for the multi-class datasets. The proposed MC-SQ methods are the best-performing methods. The last row shows the average performance across all datasets[6]. MC-SQ provides a tremendous improvement over the base quantifiers: 22% for EM, 38% for GACC, 25% for GPACC, and 31% for FM.

Figure 3 provides the CD diagram for the results in Table 1. The four proposed ensembles (MC-SQ) occupy the five top-ranking positions. MC-SQ with the base quantifier FM outperforms with statistical significance all existing quantifiers but its sibling MC-SQ ensembles with GPACC and EM as base quantifiers.

Due to a lack of space, we have presented the numerical results for binary datasets on the paper's website. Figure 4 provides the CD diagram for the results in this table. The comparison involves a total of 34 approaches, as we also include DyS as a base quantifier for both ensemble approaches. We decided to include DyS with Tøpsoe distance, which is one of the best-performing binary quantifiers [28].

[6] We understand that computing average AE across datasets can be misleading, but it is often the only way to compare average performance improvement.

**Table 1.** Experimental results for multi-class datasets. Our proposal, the Multiple-classifier, Single-quantifier (MC-SQ) ensemble, is among the best-performing approaches.

| | Single quantifiers | | | | | | | | | Class-Prevalence Ensembles | | | | **MC-SQ Ensembles** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Dataset | readme | ED | CC | PWK | QF | EM | GACC | GPACC | FM | EM | GACC | GPACC | FM | EM | GACC | GPACC | FM |
| bike | 0.201 | 0.176 | 0.368 | 0.315 | 0.638 | 0.082 | 0.113 | 0.073 | 0.102 | 0.117 | 0.101 | 0.096 | 0.104 | 0.096 | 0.068 | **0.059** | 0.065 |
| blog | 0.180 | 0.290 | 0.588 | 0.422 | 0.547 | 0.196 | 0.360 | 0.236 | 0.285 | 0.256 | 0.238 | 0.249 | 0.264 | 0.167 | 0.173 | **0.115** | 0.122 |
| conc | 0.432 | 0.457 | 0.915 | 0.480 | 0.662 | 0.498 | 0.486 | 0.473 | 0.510 | 0.410 | 0.407 | 0.381 | 0.389 | 0.256 | 0.275 | 0.266 | **0.245** |
| cond | 0.129 | 0.093 | 0.343 | 0.213 | 0.431 | 0.059 | 0.155 | 0.066 | 0.088 | 0.085 | 0.078 | 0.064 | 0.074 | 0.054 | 0.054 | **0.045** | 0.047 |
| contra | 0.424 | 0.434 | 0.833 | 0.572 | 0.675 | 0.396 | 0.600 | 0.515 | 0.512 | 0.409 | 0.468 | 0.411 | 0.402 | **0.391** | 0.470 | 0.424 | 0.419 |
| craft | 0.412 | 0.274 | 0.752 | 0.442 | 0.763 | 0.191 | 0.296 | 0.190 | 0.190 | 0.271 | 0.264 | 0.206 | 0.218 | 0.225 | 0.186 | 0.168 | **0.156** |
| diam | 0.117 | 0.209 | 0.784 | 0.404 | 0.501 | 0.214 | 0.197 | 0.098 | 0.118 | 0.183 | 0.196 | 0.110 | 0.100 | 0.042 | 0.029 | **0.027** | 0.027 |
| drugs | 0.338 | 0.238 | 0.465 | 0.407 | 0.600 | 0.218 | 0.256 | 0.199 | 0.181 | 0.229 | 0.250 | 0.252 | 0.259 | 0.204 | 0.206 | 0.181 | **0.163** |
| ener | 0.331 | 0.169 | 0.879 | 0.439 | 0.925 | 0.131 | 0.273 | 0.115 | 0.129 | 0.161 | 0.225 | 0.120 | 0.130 | 0.158 | 0.108 | **0.084** | **0.084** |
| fifa | 0.221 | 0.278 | 0.481 | 0.384 | 0.432 | 0.127 | 0.313 | 0.181 | 0.216 | 0.198 | 0.182 | 0.202 | 0.211 | 0.117 | 0.145 | 0.111 | **0.104** |
| news | 0.446 | 0.245 | 0.827 | 0.471 | 0.917 | 0.221 | 0.498 | 0.335 | 0.376 | **0.246** | 0.288 | 0.249 | 0.238 | 0.260 | 0.325 | 0.261 | 0.268 |
| nurse | 0.263 | 0.049 | 0.138 | 0.213 | 0.399 | 0.022 | 0.023 | 0.019 | 0.020 | 0.027 | 0.016 | 0.017 | 0.018 | 0.015 | 0.011 | 0.013 | **0.009** |
| thrm | 0.471 | 0.470 | 1.042 | 0.511 | 0.827 | 0.494 | 0.780 | 0.629 | 0.663 | 0.323 | 0.409 | 0.337 | 0.382 | 0.330 | 0.344 | 0.321 | **0.302** |
| turk | 0.489 | 0.356 | 0.976 | 0.622 | 0.834 | **0.277** | 0.525 | 0.342 | 0.392 | 0.365 | 0.402 | 0.338 | 0.348 | 0.432 | 0.408 | 0.315 | 0.339 |
| vgame | 0.364 | 0.424 | 0.590 | 0.418 | 0.589 | 0.322 | 0.520 | 0.460 | 0.474 | 0.375 | 0.358 | 0.364 | 0.371 | **0.315** | 0.397 | 0.391 | 0.358 |
| wine | 0.428 | 0.440 | 0.965 | 0.496 | 0.613 | 0.757 | 0.656 | 0.575 | 0.605 | 0.414 | 0.416 | 0.371 | 0.388 | **0.340** | 0.440 | 0.449 | 0.431 |
| yeast | 0.474 | **0.289** | 0.878 | 0.295 | 0.526 | 0.613 | 0.567 | 0.408 | 0.413 | 0.546 | 0.448 | 0.401 | 0.411 | 0.353 | 0.450 | 0.476 | 0.482 |
| Mean | 0.336 | 0.288 | 0.696 | 0.418 | 0.640 | 0.284 | 0.389 | 0.289 | 0.310 | 0.272 | 0.279 | 0.245 | 0.253 | 0.221 | 0.241 | 0.218 | **0.213** |

Similarly to the multi-class case, MC-SQ with FM quantifier is also the best quantifier for binary datasets. The CD diagram shows that MC-SQ with FM outperforms all existing quantifiers but the Median Sweep (MS) with a significant statistical difference. These results are evidence of the performance of the ensemble approaches for quantification, as the MS algorithm can be framed as an ensemble approach.

### 6.3 Ablation Study: Number of Base Classifiers.

A relevant parameter for our ensembles is the number of base classifier-quantifier pairs. In our experimental results, we fixed this number to seven. However, it is unclear if we could improve performance using a different number of pairs.

We executed experiments with all possible combinations of the number of classifiers and averaged the results, grouping them by the number of base classifiers. Figure 5 shows the CD diagram for this experiment. The ensembles with seven classifiers obtain the best results but with diminishing returns and no statistically significant difference compared to six base pairs.

### 6.4 Case Study: The LeQua2022 Competition.

Recently, Esuli, Moreo and Sebastiani [4] organized the LeQua 2022 competition for quantification learning. The competition released a large dataset of product reviews from Amazon.

The competition was organized into four streams: T1-A and B released tabular datasets consisting of binary and multi-class problems. Similarly, T2-A and B released textual binary and multi-class datasets. In this section, we focus on the T1-B task as we do not want the feature extraction methods to influence the methods' performance. We are primarily interested in multi-class quantification.

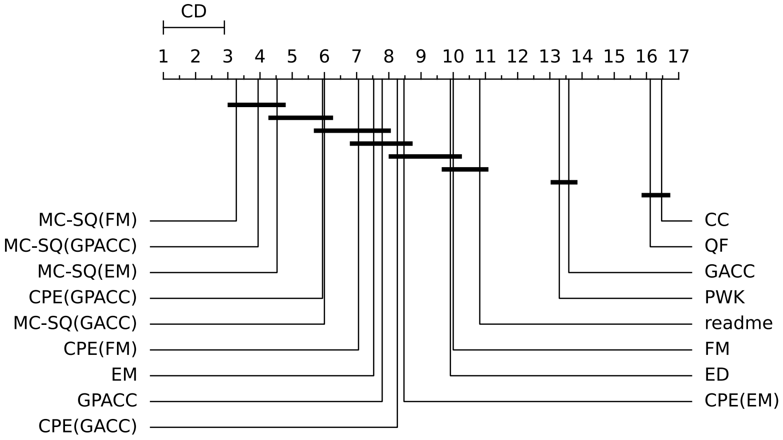

**Fig. 3.** CD diagram for multi-class datasets.

The released dataset has 28 classes with 20,000 training instances, and the competitors also had access to 1,000 development samples of 1,000 examples each. Finally, all methods were assessed in a hidden test set consisting of 5,000 test samples of 1,000 examples each.

Our methods use the default parameters. We assessed our ensembles with four base quantifiers: EM, FM, GACC and GPACC, using the development set and chose the best-performing one, GPACC as our representative. Finally, we assessed MC-SQ GPACC in the test set. Table 2 summarizes the results, with our proposal ranked first.

The competition uses Relative Absolute Error (RAE) as the main assessment criterion. RAE is defined as:

$$RAE(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{C} \sum_{n=1}^{C} \frac{|\hat{p}_n - p_n|}{p_n}$$

## 7 Conclusion and Future Work

In this paper, for the first time, we clarified the shortcomings of OVA quantification approaches in a multi-class context. We concluded that using OVA causes a distribution shift in $p(\mathbf{x}|y)$, which contradicts a common assumption of quantification methods.

We proposed an accurate multi-class ensemble method for quantification that naturally works for binary and multi-class problems. MC-SQ is a simple and parameter-free ensemble method that uses seven classifiers and the same base quantifier. We investigated its performance through extensive experiments

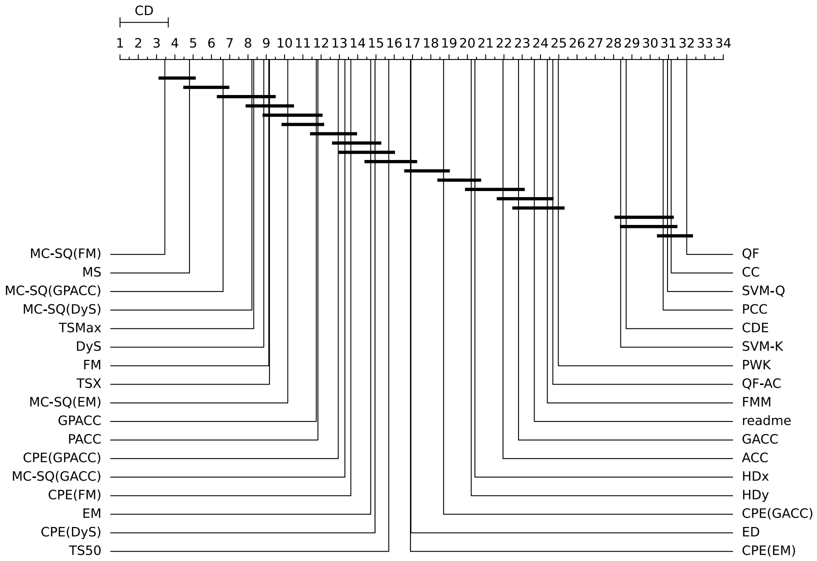

**Fig. 4.** CD diagram for binary datasets.

showing that MC-SQ is the best-performing quantifier for binary and multi-class problems.

In future work, we plan to investigate other ensemble variations, such as methods that use more than one quantification approach.

## Acknowledgments

The authors would like to thank FAPESP (Process 2021/12278-0) for the financial support.

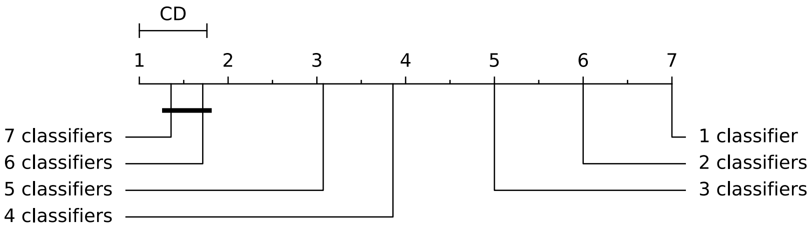

**Fig. 5.** CD diagram for the number of classifier-quantifier pairs.

**Table 2.** Results of LeQua2022 Task T1B, including our proposal MC-SQ GPACC ranked first.

| Methods | RAE |
|---|---|
| MC-SQ GPACC | **0.861** |
| UniDortmund | 0.880 |
| UniOviedo(Team1) | 0.884 |
| UniOviedo(Team2) | 1.114 |
| KULeuven | 1.178 |
| SLD | 1.182 |
| PACC | 1.305 |
| ACC | 1.421 |
| CC | 1.894 |
| PCC | 2.265 |
| MLPE | 4.577 |

## References

1. Barranquero, J., Díez, J., del Coz, J.J.: Quantification-oriented learning based on reliable classifiers. Pattern Recognit **48**(2), 591–604 (2015)
2. Barranquero, J., González, P., Díez, J., del Coz, J.J.: On the study of nearest neighbor algorithms for prevalence estimation in binary problems. Pattern Recognit **46**(2), 472–482 (Feb 2013)
3. Bella, A., Ferri, C., Hernández-Orallo, J., Ramirez-Quintana, M.J.: Quantification via probability estimators. In: ICDM. pp. 737–742. IEEE (2010)
4. Esuli, A., Moreo, A., Sebastiani, F.: LeQua@CLEF2022: Learning to quantify. In: ECIR. pp. 374–381. Springer (2022)
5. Esuli, A., Moreo Fernández, A., Sebastiani, F.: A recurrent neural network for sentiment quantification. In: CIKM. pp. 1775–1778. ACM (2018)
6. Fawcett, T., Flach, P.A.: A response to webb and ting's on the application of roc analysis to predict classification performance under varying class distributions. Mach Learn **58**(1), 33–38 (2005)
7. Firat, A.: Unified framework for quantification. arXiv preprint arXiv:1606.00868 (2016)
8. Forman, G.: Counting positives accurately despite inaccurate classification. In: ECML. pp. 564–575. Springer (2005)
9. Forman, G.: Quantifying counts and costs via classification. Data Min Knowl Discov **17**(2), 164–206 (2008)
10. Forman, G., Kirshenbaum, E., Suermondt, J.: Pragmatic text mining: minimizing human effort to quantify many issues in call logs. In: SIGKDD. pp. 852–861. ACM (2006)
11. Friedman, J.H.: Class counts in future unlabeled samples (2014), https://jerryfriedman.su.domains/talks/HK.pdf
12. González, P., Castaño, A., Chawla, N.V., Coz, J.J.D.: A review on quantification learning. CSUR **50**(5), 1–40 (2017)
13. González-Castro, V., Alaiz-Rodríguez, R., Alegre, E.: Class distribution estimation based on the hellinger distance. Information Sciences **218**, 146–164 (2013)
14. Hassan, W., Maletzke, A.G., Batista, G.: Pitfalls in quantification assessment. In: CIKM Workshops. ACM (2021)

15. Hopkins, D.J., King, G.: A method of automated nonparametric content analysis for social science. Am J Pol Sci **54**(1), 229–247 (2010)
16. Joachims, T.: A support vector method for multivariate performance measures. In: ICML. pp. 377–384 (2005)
17. Kawakubo, H., Du Plessis, M.C., Sugiyama, M.: Computationally efficient class-prior estimation under class balance change using energy distance. IEICE Trans Inf Syst **99**(1), 176–186 (2016)
18. Maletzke, A., dos Reis, D., Cherman, E., Batista, G.: Dys: a framework for mixture models in quantification. In: AAAI Conference. vol. 33, pp. 4552–4560 (2019)
19. Maletzke, A.G., Hassan, W., dos Reis, D.M., Batista, G.E.: The importance of the test set size in quantification assessment. In: IJCAI. pp. 2640–2646 (2020)
20. Milli, L., Monreale, A., Rossetti, G., Giannotti, F., Pedreschi, D., Sebastiani, F.: Quantification trees. In: ICDM. pp. 528–536. IEEE (2013)
21. Moreno-Torres, J.G., Raeder, T., Alaiz-Rodríguez, R., Chawla, N.V., Herrera, F.: A unifying view on dataset shift in classification. Pattern Recognit **45**(1), 521–530 (2012)
22. Moreo, A., Esuli, A., Sebastiani, F.: Quapy: a python-based framework for quantification. In: CIKM. pp. 4534–4543. ACM (2021)
23. Opitz, D., Maclin, R.: Popular ensemble methods: An empirical study. J Mach Learn Res **11**, 169–198 (1999)
24. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. J Mach Learn Res **12**, 2825–2830 (2011)
25. Pérez-Gállego, P., Quevedo, J.R., del Coz, J.J.: Using ensembles for problems with characterizable changes in data distribution: A case study on quantification. Inf Fusion **34**, 87–100 (2017)
26. Préz-Gállego, P., Castano, A., Ramón Quevedo, J., José del Coz, J.: Dynamic ensemble selection for quantification tasks. Inf Fusion **45**, 1–15 (2019)
27. Saerens, M., Latinne, P., Decaestecker, C.: Adjusting the outputs of a classifier to new a priori probabilities: a simple procedure. Neural Comput **14**(1), 21–41 (2002)
28. Schumacher, T., Strohmaier, M., Lemmerich, F.: A comparative evaluation of quantification methods. arXiv preprint arXiv:2103.03223 (2021)
29. Sebastiani, F.: Evaluation measures for quantification: An axiomatic approach. Inf Retr J **23**(3), 255–288 (2020)
30. Xue, J.C., Weiss, G.M.: Quantification and semi-supervised classification methods for handling changes in class distribution. In: KDD. pp. 897–906. ACM (2009)

# Measuring Fairness under Unawareness via Quantification (Extended Abstract)

Alessandro Fabris[1,2], Andrea Esuli[3], Alejandro Moreo[3], Fabrizio Sebastiani[3]

[1]Dipartimento di Ingegneria dell'Informazione
Università di Padova
35131 Padova, Italy
E-mail: fabrisal@dei.unipd.it

[2]Max Planck Institute for Security and Privacy
44799 Bochum, DE
E-mail: alessandro.fabris@mpi-sp.org

[3]Istituto di Scienza e Tecnologie dell'Informazione
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
E-mail: {alejandro.moreo,fabrizio.sebastiani}@isti.cnr.it

**Abstract.** Models trained by means of supervised learning are increasingly deployed in high-stakes domains, and, when their predictions inform decisions about people, they inevitably impact (positively or negatively) on their lives. As a consequence, those in charge of developing these models must carefully evaluate their impact on different groups of people and ensure that sensitive demographic attributes, such as race or sex, do not result in unfair treatment for members of specific groups. For doing this, awareness of demographic attributes on the part of those evaluating model impacts is fundamental. Unfortunately, the collection of these attributes is often in conflict with industry practices and legislation on data minimization and privacy. For this reason, it may be hard to measure the *group fairness* of trained models, even from within the companies developing them. In this work, we tackle the problem of measuring group fairness under unawareness of sensitive attributes, by using techniques from *quantification*. We identify five important factors that complicate the estimation of fairness under unawareness and formalize them into five different experimental protocols under which we assess the effectiveness of different estimators of group fairness. We also consider the problem of potential model misuse to infer sensitive attributes at an individual level, and demonstrate that quantification is suitable for decoupling the (desirable) objective of measuring group fairness from the (undesirable) objective of inferring sensitive attributes of individuals.

## 1 Introduction

The widespread adoption of automated decision-making in high-stakes systems has brought about an increased attention to the underlying algorithms and to their effects across sensitive groups. Typically, sensitive groups are subpopulations determined by social and demographic factors, such as race and sex. The unfair treatment of such demographic groups is ruled out by anti-discrimination laws and studied by a growing community of algorithmic fairness researchers. Important works in this space have addressed problems that may arise in the judicial system, in healthcare, in job search, and in computer vision, just to name a few domains that may be impacted. A common trait of these works is a careful definition and measurement of *group fairness*, typically viewed in terms of differences in quantities of interest, such as the acceptance rate, recall, or accuracy, across the salient subpopulations. According to popular definitions of fairness, large such differences correspond to low fairness on the part of the algorithms.

Unfortunately, sensitive demographic data, such as the race and sex of users, is often hard to obtain, for various reasons. There are several barriers to demographic data procurement which make measurement of fairness non-trivial even for the company that is developing and deploying a model. Legislation plays a major role in this, forbidding the collection of sensitive attributes in some domains. Even in the absence of explicit prohibition, privacy-by-design standards and a data minimization ethos push companies in the direction of avoiding the collection of sensitive attributes from their customers. Similarly, the prospect of negative media coverage is a clear concern, so companies often err on the side of caution and inaction. For these reasons, in a recent survey of industry practitioners, a majority of respondents stated that the availability of tools supporting fairness auditing without access to individual-level demographics would be very useful. In other words, the problem of *measuring algorithmic fairness under unawareness of sensitive attributes* is pressing, and requires ad-hoc solutions.

In the algorithmic fairness literature, much work has been done to propose techniques directly aimed at improving the fairness of a model (Donini et al., 2018; Hashimoto et al., 2018; He et al., 2020; Zafar et al., 2017). Comparably little attention, though, has been devoted to the problem of reliably measuring fairness. This represents an important and rather overlooked preliminary step to enforcing fairness and making algorithms more equitable across groups. More recent works have studied non-ideal conditions, such as noisy or missing group labels (Awasthi et al., 2020) and non-iid samples (Singh et al., 2021), showing that naïve fairness-enhancing algorithms may actually make a model *less* fair (Mehrotra and Celis, 2021).

In this work, we tackle the problem of measuring algorithmic fairness under unawareness of sensitive attributes, by using techniques from *quantification* (Esuli et al., 2023). Estimating, rather than the class labels of individual data points, the class prevalence values for sets (usually referred to as "samples") of such data points, is precisely the goal of practitioners looking to measure fairness under unawareness of sensitive attributes. When auditing an algorithm for group fairness, the aim is not the development of a model that is accurate for individual predictions (i.e., classification), which may be misused to infer people's demographics, such as a user's race, and may thus lead to the inappropriate and non-consensual utilization of this information. Rather, the central interest of fairness audits is the reliable estimation of group-level quantities (i.e., quantification), such as the prevalence of women among the instances to which a certain class has been assigned by the model.

We consider several methods that have been proposed in the quantification literature and assess their suitability for estimating the fairness of a classifier under unawareness of sensitive attributes. More precisely, we adapt quantification approaches to measure a classifier's *demographic disparity* (Barocas et al., 2019), defined as the difference in acceptance rate across relevant subpopulations. Overall, we make the following contributions:

- **Five experimental protocols for five major challenges**. Drawing from the algorithmic fairness literature, we identify five important factors for the problem of estimating fairness under unawareness of sensitive attributes. These factors are based on challenges encountered in real-world applications, including the non-stationarity of processes generating the data, and the variable cardinality of the available samples. For each factor, we define and formalize a precise experimental protocol, through which we compare the performance of quantifiers (i.e., group-level prevalence estimators) generated by six different quantification methods (Sections 4.3–4.7).
- **Adaptation and ablation study**. We demonstrate a simple procedure to adapt and integrate quantification approaches into a wider machine learning pipeline with minimal orchestration effort. We prove the importance of each component through an ablation study (Section 4.8).
- **Quantifying without classifying.** We consider the problem of potential model misuse to maliciously infer demographic characteristics at an individual level, which represents a concern for methods based on proxy attributes. Proxy methods are estimators of sensitive attributes which exploit the correlation between available attributes (e.g., ZIP code) and the sensitive attributes (e.g., race) in order to infer the values of the latter. Through a set of experiments, we demonstrate two methods that yield precise estimates of demographic disparity but poor classification performance, thus decoupling the objectives of group-level prevalence estimation and individual-level class label prediction (Section 4.9).

It is worth noting some intrinsic limitations of fairness measures and proxy methods which are also applicable to this work. In essence, proxy methods exploit co-occurrence of membership in a group and display of a given trait, potentially learning, encoding and reinforcing stereotypical associations. Even when labels for sensitive attributes are available, they are not all equivalent. Self-reported labels are preferable to avoid external assignment (i.e., inference of sensitive attributes), which may be harmful. More in general, approaches that define sensitive attributes as rigid and fixed categories are limited since they impose a taxonomy onto people, erasing the needs and experiences of those who do not fit the envisioned categories. While acknowledging these limitations, we hope our work will help highlight, investigate and mitigate unfavourable outcomes for disadvantaged groups brought about by automated decision-making systems.

The outline of this work is the following. Section 2 presents the notation employed throughout this manuscript. Section 3 shows how these approaches can be adapted and integrated to measure demographic disparity. Section 4 discusses our experiments; we omit the actual results for reasons of space, and report them in the extended version of this paper (Fabris et al., 2023). Section 5 contains concluding remarks, discussing limitations and avenues for future work.

## 2 Notation

In this paper, we use the following notation. By $\mathbf{x}$ we indicate a data item drawn from a domain $\mathcal{X}$, encoding a set of non-sensitive attributes (i.e., features) taken by classifiers and quantifiers as an input. We use $\mathcal{S}$ to denote the domain of a sensitive attribute, binarily encoded to $\mathcal{S} = \{0, 1\}$ for ease of exposition, and by $s$ a value that $\mathcal{S}$ may take. By $y$ we indicate a class taking values on a binary domain $\mathcal{Y} = \{\ominus, \oplus\}$, representing the target of a prediction task.[1]

Symbol $\sigma$ denotes a *sample*, i.e., a non-empty set of data points drawn from $\mathcal{X}$. By $p_\sigma(s)$ we indicate the true prevalence of attribute $s$ in sample $\sigma$, while by $\hat{p}^q_\sigma(s)$ we indicate the estimate of this prevalence obtained by means of quantifier $q$, which we define as a function $q : 2^{\mathcal{X}} \rightarrow [0, 1]$. Since $0 \leq p_\sigma(s) \leq 1$ and $0 \leq \hat{p}^q_\sigma(s) \leq 1$ for all $s \in \mathcal{S}$, and since $\sum_{s \in \mathcal{S}} p_\sigma(s) = \sum_{s \in \mathcal{S}} \hat{p}^q_\sigma(s) = 1$, the $p_\sigma(s)$'s and the $\hat{p}^q_\sigma(s)$'s form two probability distributions across $\mathcal{S}$.

We also introduce random variables $X, S, Y, \hat{Y}$ which denote, respectively, data points from $\mathcal{X}$, their sensitive attributes, true labels, and predicted labels. By $\Pr(V = v)$ we indicate, as usual, the probability that random variable $V$ takes value $v$, which we shorten as $\Pr(v)$ when $V$ is clear from context. By $h : \mathcal{X} \rightarrow \mathcal{Y}$ we indicate a binary classifier that assigns classes in $\mathcal{Y}$ to data points; by $k : \mathcal{X} \rightarrow \mathcal{S}$ we instead indicate a binary classifier that assigns sensitive attributes $\mathcal{S}$ to data points (e.g., that predicts if a certain data item $\mathbf{x}$ is "female"). It is worth re-emphasizing that both $h$ and $k$ only use non-sensitive attributes from $\mathcal{X}$ as input variables. For ease of use, we will interchangeably write $h(\mathbf{x}) = y$ or $h_y(\mathbf{x}) = 1$, and $k(\mathbf{x}) = s$ or $k_s(\mathbf{x}) = 1$.

We consider three separate datasets, following the workflow of a realistic machine learning pipeline.

- A *training set* $\mathcal{D}_1$ for $h$, $\mathcal{D}_1 = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}\}$, typically of large cardinality. Given the inherent difficulties in demographic data procurement, we expect this dataset to contain no explicit information on the sensitive attributes $\mathcal{S}$.

[1] In this paper we assume the existence of a single binary sensitive attribute $\mathcal{S}$; however, there is no loss of generality in this, since everything we say can straightforwardly be extended to the case in which multiple sensitive attributes are present at the same time. Moreover, we focus on the case in which the classifier that we want to audit is a binary one, but the definitions and techniques we employ can be straightforwardly extended to a multiclass setting.

- A small *auxiliary set* $\mathcal{D}_2 = \{(\mathbf{x}_i, s_i) \mid \mathbf{x}_i \in \mathcal{X}, s_i \in \mathcal{S}\}$, employed to learn quantifiers for the sensitive attribute. This dataset may originate from a targeted effort, such as interviews, surveys sent to customers asking for voluntary disclosure of sensitive attributes, or other optional means to share demographic information. Alternatively it could derive from data acquisitions carried out for other purposes. Both $\mathcal{D}_1$ and $\mathcal{D}_2$ are in the development domain of our machine learning pipeline.
- A *deployment set* $\mathcal{D}_3 = \{\mathbf{x}_i \mid \mathbf{x}_i \in \mathcal{X}\}$ which emulates the production domain for classifier $h$, whose demographic parity we aim to measure. Alternatively, acting proactively rather than reactively, $\mathcal{D}_3$ could also be a held-out test set available at a company for pre-deployment audits. From the perspective of the estimation task at hand, i.e. estimating the demographic disparity of $h$, $\mathcal{D}_2$ represents the quantifiers' training set, while $\mathcal{D}_3$ is their test set.

## 3 Using quantification to measure fairness under unawareness of sensitive attributes

We adapt the above quantification approaches for estimating a classifier's fairness. We define classifier fairness in terms of *demographic parity* (also called *statistical parity* (Dwork et al., 2012) or *independence* (Barocas et al., 2019)), and, in particular, of a flavour of demographic parity based on the distribution of sensitive attribute $\mathcal{S}$ conditional on the prediction of the classifer, as proposed in (Wachter et al., 2020). We call our estimand the *demographic disparity* of classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$ for attribute value $s$, and define it as

$$\varDelta(s) = \Pr(S = s|\hat{Y} = \ominus) - \Pr(S = s|\hat{Y} = \oplus) \tag{1}$$

or, more concisely,

$$\varDelta(s) = \Pr(s|\ominus) - \Pr(s|\oplus) \tag{2}$$

It is worth reemphasizing that the sensitive attribute $\mathcal{S}$ does *not* belong to the set of attributes $\mathcal{X}$ which generate the feature space on which classifier $h$ operates (in other words, when training $h$ we are *unaware* of $\mathcal{S}$). Demographic disparity measures whether the prevalence of the sensitive attribute in the group assigned to the positive class is the same as in the group assigned to the negative class; a value $\varDelta(s) = 0$ indicates maximum fairness, while values of $\varDelta(s) = -1$ or $\varDelta(s) = +1$ indicate minimum fairness, with the sign of $\varDelta(s)$ indicating whether, for $S = s$, the classifier is biased towards the $\oplus$ class or the $\ominus$ class, respectively.

*Example 1.* Assume that $\mathcal{S}$ stands for "sex", $s$ for "female", and that the classifier is in charge of recommending loan applicants for acceptance, classifying them as "grant" ($\oplus$) or "deny" ($\ominus$). For simplicity, let us assume the outcome of the classifier to directly translate into a decision without human supervision. The bank might want to check that the fraction of females out of the total number of loan recipients is approximately the same as the fraction of females out of the total number of applicants who are denied the loan. In other words, the bank might want $\Delta(s)$ to be close to 0. Of course, if the bank is aware of the sex of each applicant, this constraint is very easy to check and, potentially, enforce. If the bank is unaware of applicants' sex, as we assume here, the problem is not trivial, and this is where our techniques come in.

In estimating the demographic disparity of $h$, our focus is on the deployment set where $h$ is supporting the decision-making process. To highlight this fact, we rewrite Equation 2 by making the dependence of $\Delta(s)$ on $\mathcal{D}_3$ explicit, i.e.,

$$\Delta(s) = p_{\mathcal{D}_3^{\ominus}}(s) - p_{\mathcal{D}_3^{\oplus}}(s) \tag{3}$$

where we define

$$\begin{aligned} \mathcal{D}_3^{\oplus} =& \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \oplus\} \\ \mathcal{D}_3^{\ominus} =& \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \ominus\} \end{aligned} \tag{4}$$

and where we make explicit the fact that, if a value $s$ that attribute $\mathcal{S}$ can take is viewed as a class, the probabilities $\Pr(s|\ominus)$ and $\Pr(s|\oplus)$ of Equation 2 may be seen as the prevalence values of class $s$ in the two samples $\mathcal{D}_3^{\oplus}$ and $\mathcal{D}_3^{\ominus}$. In other words, measuring demographic disparity is reduced to estimating the prevalence values of class $s$ in the two samples $\mathcal{D}_3^{\oplus}$ and $\mathcal{D}_3^{\ominus}$, i.e., *it can be framed as a task of quantification.*

This approach can be easily integrated into existing machine learning pipelines with little orchestration effort. Below, we summarize the workflow we envision:

1. A classifier $h : \mathcal{X} \rightarrow \mathcal{Y}$ is trained (under unawareness of sensitive attribute $\mathcal{S}$) on $\mathcal{D}_1$ and ready for production. The assumption that, at this stage, we are unaware of sensitive attribute $\mathcal{S}$ is due to the inherent difficulties in demographic data procurement already mentioned in Section 1.
2. Classifier $h$ naturally imposes a partition of the auxiliary set $\mathcal{D}_2$ into $\mathcal{D}_2^{\ominus} = \{(\mathbf{x}_i, s_i) \in \mathcal{D}_2 \mid h(\mathbf{x}) = \ominus\}$ and $\mathcal{D}_2^{\oplus} = \{(\mathbf{x}_i, s_i) \in \mathcal{D}_2 \mid h(\mathbf{x}) = \oplus\}$. These two disjoint datasets act as the training sets for the two quantifiers $q_{\ominus}$ and $q_{\oplus}$. Quantifier $q_{\ominus}$ (or its dual $q_{\oplus}$) is trained on $\mathcal{D}_2^{\ominus}$ (resp., $\mathcal{D}_2^{\oplus}$) to estimate the prevalence of data points where $S = s$ among the data points labelled with $\ominus$ (resp., $\oplus$).
3. Classifier $h$ also imposes a partition of the deployment set $\mathcal{D}_3$ into $\mathcal{D}_3^{\ominus} = \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \ominus\}$ and $\mathcal{D}_3^{\oplus} = \{\mathbf{x} \in \mathcal{D}_3 \mid h(\mathbf{x}) = \oplus\}$. Quantifiers $q_{\ominus}$ and $q_{\oplus}$ trained in Step 2 are applied to these datasets to obtain an estimate of the prevalence of $s$ in $\mathcal{D}_3^{\ominus}$ and $\mathcal{D}_3^{\oplus}$. The demographic disparity of $h$, defined in Equation 1, can thus be estimated as

$$\hat{\Delta}(s) = \hat{p}^{q_{\ominus}}_{\mathcal{D}_3^{\ominus}}(s) - \hat{p}^{q_{\oplus}}_{\mathcal{D}_3^{\oplus}}(s) \tag{5}$$

   where, as we recall from Section 2, $\hat{p}^{q}_{\sigma}(s)$ denotes the prevalence of attribute value $s$ in set $\sigma$ as estimated via quantification method $q$.

This quantification-based way of tackling demographic disparity is suited for a non-invasive auditing procedure, since it allows unawareness of the sensitive

Table 1: Summary of experimental protocols.

| Protocol name | Variable | Section |
|---|---|---|
| `sample-prev-`$\mathcal{D}_1$ | joint distribution of $(S, Y)$ in $\mathcal{D}_1$, via sampling | § 4.3 |
| `flip-prev-`$\mathcal{D}_1$ | joint distribution of $(S, Y)$ in $\mathcal{D}_1$, via label flipping | § 4.4 |
| `sample-size-`$\mathcal{D}_2$ | size of $\mathcal{D}_2$, via sampling | § 4.5 |
| `sample-prev-`$\mathcal{D}_2$ | joint distribution of $(S, \hat{Y})$ in $\mathcal{D}_2$, via sampling | § 4.6 |
| `sample-prev-`$\mathcal{D}_3$ | joint distribution of $(S, \hat{Y})$ in $\mathcal{D}_3$, via sampling | § 4.7 |

attribute $\mathcal{S}$ in the set $\mathcal{D}_1$ used for training the classifier $h$ to be audited and in the set $\mathcal{D}_3$ on which this classifier is going to be deployed; it only requires the availability of an auxiliary data set $\mathcal{D}_2$ where attribute $\mathcal{S}$ is present. Dataset $\mathcal{D}_2$ may originate from a targeted effort, such as interviews, surveys sent to customers asking for voluntary disclosure of sensitive attributes, or other optional means to share demographic information. Alternatively it could derive from data acquisitions carried out for other purposes.

Additionally, we note that this approach is extremely suitable to situations in which the prevalence of attribute value $s$ in $\mathcal{D}_2$ is possibly very different from the prevalence of $s$ in the test set $\mathcal{D}_3$ (a situation that certainly characterizes many operational environments) since the best quantification approaches are robust by construction to distribution drift, as we will show in the next section.

# 4 Experiments

## 4.1 General setup

In this section we describe an evaluation of different estimators of demographic disparity. We propose five experimental protocols (Sections 4.3–4.7) summarized in Table 1. Each protocol focuses on a single factor of import for the estimation problem, varying the size and mutual shift of the training, auxiliary, and deployment set. Protocol names are in the form `action-characteristic-dataset`, as they act on datasets ($\mathcal{D}_1$, $\mathcal{D}_2$ or $\mathcal{D}_3$) modifying their characteristics (size or class prevalence) through one of two actions (sampling or label flipping). We investigate the effect of each factor on the performance of six estimators of demographic disparity, keeping the remaining factors constant.

Under each experimental protocol, the size or the prevalence of a given dataset is carefully varied based on the protocol definition. For every protocol, we perform an extensive empirical evaluation as follows:

- We compare the performance of each estimation technique on three datasets (Adult, COMPAS, and Credit Card Default). The datasets and respective preprocessing are described in detail in Section 4.2.
- We split a given dataset into $\mathcal{D}_A, \mathcal{D}_B, \mathcal{D}_C$, three stratified subsets of identical size and same distribution over $(S, Y)$. Five such random splits are

performed. To test each estimator under the same conditions, these splits are the same for every method.

- For each split, we permute the role of the stratified subsets $\mathcal{D}_A, \mathcal{D}_B, \mathcal{D}_C$, so that each subset alternatively serves as the training ($\mathcal{D}_1$), auxiliary ($\mathcal{D}_2$), or deployment set ($\mathcal{D}_3$). All (six) such permutations are tested.
- Whenever an experimental protocol requires sampling from a subset, for instance when artificially altering a class prevalence value, we perform 10 different samplings. To perform extensive experiments at a reasonable computational cost, every time an experimental protocol requires changing a dataset $\mathcal{D}$ into a shifted version $\breve{\mathcal{D}}$, we also reduce its cardinality to $|\breve{\mathcal{D}}| = 500$. Further details and implications of this choice on each experimental protocol are provided in the context of the protocol's setup.
- Different learning approaches can be used to train the sensitive attribute classifier $k$ underlying each quantification method. We test Logistic Regression (LR) and Support Vector Machines (SVM).[2] Sections 4.3–4.7 report results of quantification algorithms wrapped around a LR classifier. Analogous results obtain for SVMs are reported in (Fabris et al., 2023).
- The classifier $h$, whose demographic disparity we aim to estimate, is LR trained with balanced class weights (i.e., loss weights inversely proportional to class frequencies).
- To measure the effect of a given factor on the performance of different quantifiers, we report the signed estimation error, derived from Equations 3 and 5 as follows:

$$\begin{aligned} e &= \hat{\varDelta}(s) - \varDelta(s) \\ &= \left[\hat{p}^{q_\ominus}_{\mathcal{D}_3^\ominus}(s) - \hat{p}^{q_\oplus}_{\mathcal{D}_3^\oplus}(s)\right] - \left[p_{\mathcal{D}_3^\ominus}(s) - p_{\mathcal{D}_3^\oplus}(s)\right] \end{aligned} \tag{6}$$

  We summarize the experiments by reporting the Mean Absolute Error (MAE) and Mean Squared Error (MSE), where the mean of errors $e_i$ is computed over multiple experiments $E$.

Overall, our experiments consist of over 700,000 separate estimates of demographic disparity.[3] The actual results of our experiments are omitted from this paper for reasons of space; for these results we refer the reader to the extended version of this paper (Fabris et al., 2023).

The remainder of this section is organized as follows. Section 4.2 presents the chosen datasets and the applied preprocessing. Sections 4.3–4.7 motivate and detail the experimental protocols, reporting the performance of different demographic disparity estimators. Section 4.8 describes an ablation study, aimed at investigating the benefits of training and maintaining multiple class-specific

[2] Some among the quantification methods we test in this study require the classifier to output posterior probabilities (as is the case for LR). If a classifier natively outputs classification scores that are not probabilities (as is the case for SVM), the former can be converted into the latter via "probability calibration".

[3] Code available at `https://github.com/alessandro-fabris/ql4facct`.

Table 2: Dataset statistics after preprocessing.

| Dataset | #data items | #non-sensitive features | sensitive attribute | $S = 1$ | $Pr(S = 1)$ | target variable | $Y = \oplus$ | $Pr(Y = 1)$ |
|---|---|---|---|---|---|---|---|---|
| Adult | 45,222 | 84 | sex | Male | 0.675 | income | $> 50$K | 0.248 |
| COMPAS | 5,278 | 6 | race | Caucasian | 0.398 | recidivist | no | 0.498 |
| CreditCard | 30,000 | 81 | sex | Male | 0.396 | default | no | 0.779 |

quantifiers rather than a single one. Finally, Section 4.9 shows that good estimators of demographic disparity are not necessarily good at classifying the sensitive attribute at an individual level, so that reliable fairness auditing may be decoupled from this undesirable misuse of the same models.

### 4.2 Datasets

We perform our experiments on three datasets. We choose Adult and COMPAS, two standard datasets in the algorithmic fairness community, and Credit Card Default (hereafter: CreditCard), which serves as a representative use case for a bank performing a fairness audit of a prediction tool used internally. A summary of these datasets and related statistics is reported in Table 2. See the extended version of this paper (Fabris et al., 2023) for more details on these datasets.

### 4.3 Protocol `sample-prev-`$\mathcal{D}_1$

In the first experimental protocol, we evaluate the impact of shifts in the training set $\mathcal{D}_1$, by drawing different subsets $\breve{\mathcal{D}}_1$ as we vary $\Pr(Y = S)$.[4] More specifically, we vary $\Pr(Y = S)$ between 0 and 1 with a step of 0.1. In other words, we sample at random from $\mathcal{D}_1$ a proportion $p$ of instances $(\mathbf{x}_i, s_i, y_i)$ such that $Y = S$ and a proportion $(1 - p)$ such that $Y \neq S$, with $p \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$. It is worth noting that we defined $\mathcal{D}_1$, in Section 2, as a training set involving $(\mathcal{X}, \mathcal{Y})$. Here we exploit our knowledge of $\mathcal{S}$ to control the dataset shift between training and test conditions, emulating a biased data collection procedure. Once a training set has been selected, the classifier $h$ is learnt exclusively from non-sensitive attributes $\mathcal{X}$, completely disregarding the sensitive attribute $\mathcal{S}$. We choose a limited cardinality $|\breve{\mathcal{D}}_1| = 500$, which lets us perform multiple repetitions at reasonable computational costs, as outlined in Section 4.1. While this may impact the quality of the classifier $h$, this aspect is not the central focus of the present work.

This experimental protocol aligns with biased data collection procedures, sometimes referred to as *censored data*. Indeed, it is common for the ground truth variable to represent a mere proxy for the actual quantity of interest, with non-trivial sampling effects between the two. For instance, the validity of

[4] Although $Y$ and $S$ take values from different domains, by $Y = S$ we mean $(Y = \oplus \wedge S = 1) \vee (Y = \ominus \wedge S = 0)$, i.e. a situation where positive outcomes are associated with group $S = 1$ and negative outcomes with group $S = 0$.

arrest data as a proxy for offence has been brought into question. Indeed, in this domain, different sources of sampling bias may be in action, such as uneven allocation of police resources across jurisdictions and neighbourhoods and lower levels of cooperation in populations who feel oppressed by law enforcement.

By varying $\Pr(Y = S)$ we are imposing a spurious correlation between $Y$ and $S$, which may be picked up by the classifier $h$. In extreme situations, such as when $\Pr(Y = S) \simeq 1$, a classifier $h$ may end up confounding the concepts behind $S$ and $Y$. In turn, we expect this to unevenly impact the acceptance rates for the two demographic groups, effectively changing the demographic disparity of $h$, i.e., our estimand $\Delta(s)$.

### 4.4 Protocol `flip-prev-`$\mathcal{D}_1$

Certain biases in the training set resulting from domain-specific practices, such as the use of arrest as a substitute for offence, may be modelled as either a selection bias or a label bias distorting the ground truth variable $Y$. With this experimental protocol, we impose the latter bias by actively flipping some ground truth labels $Y$ in $\mathcal{D}_1$ based on their sensitive attribute. Similarly to `sample-prev-`$\mathcal{D}_1$, this protocol achieves a given association between the target $Y$ and sensitive variable $S$ in the training set $\mathcal{D}_1$. However, instead of sampling, it does so by flipping the $Y$ label of some data points. More specifically, we impose $\Pr(Y = \ominus|S = 0) = \Pr(Y = \oplus|S = 1) = p$ and let $p$ take values across eleven evenly spaced values between 0 and 1. For every value of $p$, we firstly sample a random subset $\check{\mathcal{D}}_1$ of the training set with cardinality 500. Next, we actively flip some $Y$ labels in both demographic groups, until both $\Pr(Y = \ominus|S = 0)$ and $\Pr(Y = \oplus|S = 1)$ reach a desired value of $p \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$. Finally, we train a classifier $h$ on the attributes $\mathcal{X}$ and modified ground truth $Y$ of $\check{\mathcal{D}}_1$.

This experimental protocol is compatible with settings where the training data captures a distorted ground truth due to systematic biases and group-dependent annotation accuracy. As an example, the quality of medical diagnoses can depend on race, sex and socio-economical status. Moreover, health care expenditures have been used as a proxy to train an algorithm deployed nationwide in the US to estimate patients' health care needs, resulting in systematic underestimation of the needs of black patients. In the hiring domain, employer response rates to resumes have been found to vary with the perceived ethnic origin of an applicant's name. Finally, the gender gap in mathematical performance, while negligible in elementary school, has been found to increase with age, possibly due to gender stereotypes arising in this domain from an early age and to the prescriptive nature of these stereotypes. These are all examples where the "ground truth" associated with a dataset is distorted to the disadvantage of a sensitive demographic group.

Similarly to Section 4.3, we expect this experimental protocol to bring about sizeable variations in the demographic disparity of classifier $h$ due to the strong correlation we are imposing between $S$ and $Y$ via label flipping.

### 4.5 Protocol `sample-size-`$\mathcal{D}_2$

A further factor of interest for the estimation problem is the size of the auxiliary set $\mathcal{D}_2$, whose influence is studied in this experimental protocol. Our goal is to understand how low we can go in the small data regime, before degrading the performance of different estimation techniques. We consider subsets $\breve{\mathcal{D}}_2$ of the auxiliary set, sampling instances uniformly without replacement from it. We let cardinality $|\breve{\mathcal{D}}_2|$ take five values that are evenly spaced on a log scale, between a minimum sample size $|\breve{\mathcal{D}}_2|$=1,000 and a maximum size $|\breve{\mathcal{D}}_2| = |\mathcal{D}_2|$. In other words, we let the cardinality of the auxiliary set take five different values between 1,000 and $|\mathcal{D}_2|$ in a geometric progression. As described in Section 4.1, for each cardinality of the auxiliary set we wish to test, we perform ten samplings over five splits and six permutations, for a total of 300 repetitions per approach per dataset.

This protocol is justified by the well-documented difficulties in demographic data procurement for industry practitioners, which vary depending on domain, company, and other factors of disparate nature. Furthermore, the collection of sensitive attributes in the US is highly industry-dependent, ranging from mandatory to forbidden, depending on the fragmented regulation applicable in each domain. Finally, high quality auxiliary sets may be obtained through optional surveys, for which response rates are highly dependent on trust, and can be improved by making the intended use for the data clearer.

For these reasons, the cardinality of the auxiliary set $\mathcal{D}_2$ is an interesting variable in the context of fairness audits. The estimation methods we consider have peculiar data requirements, with diverse purposes (e.g., estimation of true positive rates – *tpr*) and approaches. For this reason, interesting patterns should emerge from this protocol. We expect key trends for the estimation error to vary monotonically with cardinality $|\breve{\mathcal{D}}_2|$, which is why we let it vary according to a geometric progression.

### 4.6 Protocol `sample-prev-`$\mathcal{D}_2$

The auxiliary set $\mathcal{D}_2$ can also display significant dataset shifts with respect to the the sets $\mathcal{D}_1$ and $\mathcal{D}_3$ available during training or at deployment. With this experimental protocol, we assess the estimation error under shifts which affect either $\mathcal{D}_2^\ominus$ or $\mathcal{D}_2^\oplus$, i.e., the subsets of $\mathcal{D}_2$ labelled positively or negatively by classifier $h$. We consider two experimental sub-protocols, describing variations in the prevalence of sensitive variable $S$ in either subset. More specifically, we let $\Pr(s|\ominus)$ (or its dual $\Pr(s|\oplus)$) take 9 evenly spaced values between 0.1 and 0.9. We avoid extreme values of 0 and 1 which would make either demographic group $S = 0$ or $S = 1$ absent from the training set of one quantifier. To exemplify, in sub-protocol `sample-prev-`$\mathcal{D}_2^\ominus$ we let the prevalence $\Pr(s|\ominus)$ in $\breve{\mathcal{D}}_2^\ominus$ take values in $\{0.1, 0.2\ldots, 0.8, 0.9\}$, while the remaining subset $\breve{\mathcal{D}}_2^\oplus$ remains at is natural prevalence $\Pr(s|\oplus)$.[5] For each repetition, we set $|\breve{\mathcal{D}}_2^\ominus| = |\breve{\mathcal{D}}_2^\oplus| = 500$. This makes

[5] The natural prevalence is matched allowing for small fluctuations due to subsampling.

for a challenging quantification setting and allows for fast training of multiple quantifiers across many repetitions.

This protocol captures issues of representativeness in demographic data, e.g., due to non-uniform response rates across subpopulations. Given the importance of trust for the provision of one's sensitive attributes, in some domains this practice is considered akin to a *data donation*. Individuals from groups that historically had worse quality or lower acceptance rates for a service can be hesitant to disclose their membership to said group, fearing it may be used against them as grounds for rejection or discrimination. This may be especially true for individuals who perceive to be at high risk of rejection, bringing about complex selection biases, jointly dependent on $S$ and $Y$, or $S$ and $\hat{Y}$ if individuals have some knowledge of the classification procedure. For example, health care providers are advised to collect information about patients' race to monitor the quality of services across subpopulations. In a field study, 28% of patients reported discomfort about disclosure of their own race to a clerk, with black patients significantly less comfortable than white patients on average.

This is the first protocol we describe where quantifiers are trained on subsets $\breve{\mathcal{D}}_2^{\ominus}$, $\breve{\mathcal{D}}_2^{\oplus}$ that have a different prevalence for the sensitive variable $S$ with respect to their counterparts $\mathcal{D}_3^{\ominus}$, $\mathcal{D}_3^{\oplus}$ in the deployment set. More specifically, with this protocol, we vary the joint distribution of $(S, \hat{Y})$ to directly influence the demographic disparity of the classifier $h$ on the auxiliary set $\mathcal{D}_2$, and move it away from the value $\varDelta(s)$ of the same measure computed on the deployment set $\mathcal{D}_3$. This is a fundamental evaluation protocol as it makes our estimand different across $\mathcal{D}_2$ (or, more precisely, its modified version $\breve{\mathcal{D}}_2$) and $\mathcal{D}_3$, which is typically expected in practice. If this were not the case, a practitioner could simply resort to an explicit computation of demographic disparity on the auxiliary set $\mathcal{D}_2$ and deem it representative of any deployment condition. Given this reasoning, we borrow this protocol from the quantification literature to cause sizeable variations in the demographic disparity of $h$ across $\mathcal{D}_2$ and $\mathcal{D}_3$, which act as the training and test set to different quantifiers. We expect these variations to bring about clear trends in the estimation error of demographic parity for the approaches considered in this work.

### 4.7 Protocol `sample-prev-`$\mathcal{D}_3$

This is essentially the counterpart of protocol `sample-prev-`$\mathcal{D}_2$ (Section 4.6), focusing on shifts in the test set $\mathcal{D}_3$. Similarly, we consider two sub-protocols that model changes in the prevalence of a sensitive variable $S$ in the test subset of either positively or negatively predicted instances, called $\mathcal{D}_3^{\ominus}$ and $\mathcal{D}_3^{\oplus}$. More in detail, we let $\Pr(s|\ominus)$ (or its dual $\Pr(s|\oplus)$) in $\breve{\mathcal{D}}_3$ take eleven evenly spaced values between 0 and 1. For example, under sub-protocol `sample-prev-`$\mathcal{D}_3^{\ominus}$, we vary the prevalence of sensitive attribute $S$ in $\breve{\mathcal{D}}_3^{\ominus}$, so that $\Pr(s|\ominus) \in \{0.0, 0.1 \ldots, 0.9, 1.0\}$, while keeping the prevalence in $\breve{\mathcal{D}}_3^{\oplus}$ fixed. Contrary to protocol `sample-prev-`$\mathcal{D}_2$, here we also allow for extreme prevalence values of 0 and 1 for the sensitive attribute $S$, as this does not invalidate the quantifiers' training. For both sub-protocols, in each repetition we sample subsets of the test set $\mathcal{D}_3$ such that $|\breve{\mathcal{D}}_3^{\ominus}| = |\breve{\mathcal{D}}_3^{\oplus}| = 500$.

This protocol accounts for the inevitable evolution of phenomena, especially those related to human behaviour. Indeed, it is common in real-world scenarios for data generation processes to be non-stationary and change across training and test, due e.g., to seasonality or any sort of unmodelled novelty and difference in populations. Given most work on algorithmic fairness focuses on decisions or predictions about people, and given the unavoidable role of change in human lives, values, and behaviour, the above considerations about non-stationarity seem particularly relevant in this context. For instance, data available from one population is often repurposed to train algorithms that will be deployed on a different population, requiring ad-hoc fair learning approaches and evoking the *portability trap* of fair machine learning. Moreover, agents may be responsive to novel technology in their social context and adapt their behaviour accordingly, causing *ripple effects* and *feedback loops*. Furthermore, as a concrete (although spurious) example of a shift in a popular fairness dataset, the repeated offense rate for black and white defendants in the COMPAS datasetincreases sharply between 2013 and 2014. As a final example, personalized pricing constitutes an increasingly possible practice with non-trivial fairness concerns and inevitable shifts due to changing habits and environments.

In the quantification literature, this is the most common evaluation protocol. Similarly to `sample-prev-`$\mathcal{D}_2$, it imposes shifts in the estimand between the training and testing conditions of a quantifier, represented by the auxiliary set $\mathcal{D}_2$ and the deployment set $\mathcal{D}_3$, respectively. Through this protocol, we expect to find similar patterns to those highlighted in Section 4.6, with the roles of the auxiliary set $\mathcal{D}_2$ and test set $\mathcal{D}_3$ now switched. Under this protocol, $\mathcal{D}_3$ has a smaller cardinality and variable prevalence (and is referred to as $\breve{\mathcal{D}}_3$ for this reason), while $\mathcal{D}_2$ is left to its original cardinality and prevalence of sensitive attribute $S$.

### 4.8 Ablation study

In Sections 4.3–4.7 we tested six approaches to estimate demographic disparity. For each approach, we exploited multiple quantifiers for the sensitive attribute $\mathcal{S}$, namely one for each class in the codomain of classifier $h$. In the binary setting adopted in this work, where $\mathcal{Y} = \{\ominus, \oplus\}$, we trained two quantifiers. One quantifier was trained on the set of positively-classified instances of the auxiliary set $\mathcal{D}_2^{\oplus} = \{(\mathbf{x}_i, s_i) \in \mathcal{D}_2 \mid h(\mathbf{x}) = \oplus\}$ and deployed to quantify the prevalence of sensitive instances (such that $S = s$) within the deployment subset $\mathcal{D}_3^{\oplus}$. The remaining quantifier was trained on $\mathcal{D}_2^{\ominus}$ and deployed on $\mathcal{D}_3^{\ominus}$.

Training and maintaining multiple quantifiers is more expensive and cumbersome than having a single one. Firstly, quantifiers that depend on the classification outcome $\hat{y} = h(\mathbf{x})$ require re-training every time $h$ is modified, e.g., due to a model update being rolled out. Secondly, the cost of maintenance is multiplied by the number of classes $|\mathcal{Y}|$ that are possible for the outcome variable. To ensure these downsides are compensated by performance improvements, we perform an ablation study evaluating the performance of different estimators of demographic disparity supported by a single quantifier.

In this section, we concentrate on three estimation approaches, namely CC, SLD and PACC. CC is chosen as the naïve baseline adopted by practitioners unaware of ad-hoc approaches for prevalence estimation. SLD and PACC are among the best performers in Sections 4.3–4.7, displaying low bias or variance across all protocols. We compare their performance under two settings. In the first setting, adopted thus far, two separate quantifiers $q_{\ominus}$ and $q_{\oplus}$ are trained on $\mathcal{D}_2^{\ominus}$, $\mathcal{D}_2^{\oplus}$ and deployed on $\mathcal{D}_3^{\ominus}$, $\mathcal{D}_3^{\oplus}$, respectively. In the second setting, we train a single quantifier $q$ on $\mathcal{D}_2$ and deploy it separately on $\mathcal{D}_3^{\ominus}$ and $\mathcal{D}_3^{\oplus}$ to estimate $\hat{\Delta}(s)$ via Equation 5, specialized so that $q_{\ominus}$ and $q_{\oplus}$ are the same quantifier.

### 4.9 Quantifying without classifying

The motivating use case for this work are internal audits of group fairness, to characterize a model and its potential to harm sensitive categories of users. We envision this as an important first step to empower practitioners in arguing for resources and, more broadly, advocate for deeper understanding and careful evaluation of models. Unfortunately, developing a tool to infer demographic information, even if motivated by careful intentions and good faith, leaves open the possibility for misuse, especially at an individual level. Once a predictive tool, also capable of instance-level classification, is available, it will be tempting for some actors to exploit it precisely for this purpose.

For example, the *Bayesian Improved Surname Geocoding* (BISG) method is intended to estimate population-level disparities in health care. However, it was also used to identify individuals potentially eligible for settlements related to discriminatory practices by auto lending companies. Automatic inference of sensitive attributes of individuals is problematic for several reasons. Such procedure exploits the co-occurrence of membership in a group and display of a given trait, running the risk of learning, encoding and reinforcing stereotypical associations. While also true of group-level estimates, this practice is particularly troublesome at an individual level, where it is likely to cause harms for people who do not fit the norm, resulting, for instance, in misgendering and the associated negative effects. Even when “accurate”, the mere act of externally assigning sensitive labels can be problematic. For example, gender assignment may be forceful and lead to psychological harm for individuals.

We here aim to demonstrate that it is possible to decouple the objective of (group-level) quantification of sensitive attributes from that of (individual-level) classification. For protocols in previous sections, we compute the accuracy and $F_1$ score (defined below) of the sensitive attribute classifier $k$ underlying the tested quantifiers, comparing it against their estimation error for class prevalence of the sensitive attribute $S$ (Equation 6).

## 5 Discussion and conclusion

Measuring the differential impact of models on groups of individuals is important to understand their effects in the real world and their tendency to encode and reinforce divisions and privilege across sensitive attributes. Unfortunately, in practice, demographic attributes are often unavailable. In this work we have taken the perspective of responsible practitioners, interested in internal fairness audits of production models. We have tackled the problem of measuring group fairness under unawareness of sensitive attributes by applying approaches from the quantification learning literature that are specifically designed for group-level estimation rather than individual-level classification; this is convenient, since practitioners who try to measure fairness under unawareness are precisely interested in group-level estimates.

We have studied the problem of estimating a classifier's demographic disparity at deployment under unawareness of sensitive attributes, with access to a disjoint auxiliary set of data for which demographic information is available. Drawing from the algorithmic fairness literature, we have identified five factors of import for this problem, associating each of them with a formal evaluation protocol. These factors mirror challenges in real-world applications, including dataset shift and variable cardinality for auxiliary datasets comprising demographic information. We have tested five quantification methods under every protocol, comparing them against the naïve Classify-and-Count (CC) method, which represents the default approach for practitioners unaware of quantification. Each quantification approach was shown to outperform CC under most combinations of 5 protocols, 3 datasets, and 2 underlying learners. Moreover, we have shown a simple approach to integrate quantification methods into existing machine learning pipelines with little orchestration effort, and demonstrated the importance of each component through an ablation study.

Finally, we have considered the problem of model misuse to infer demographic characteristics at an individual level, which represents a concern when developing models to measure group fairness via proxy attributes. Through a dedicated set of experiments, we have shown that it is possible to obtain precise estimates of demographic disparity from methods that have poor classification performance. This is a positive result for decoupling these two objectives, which should help deter from the exploitation of these models for individual-level inference.

## Acknowledgments

The work by Alessandro Fabris was supported by MIUR (Italian Minister for Education) under the "Departments of Excellence" initiative (Law 232/2016). The work by Andrea Esuli, Alejandro Moreo, and Fabrizio Sebastiani has been supported by the SoBigData++ project, funded by the European Commission (Grant 871042) under the H2020 Programme INFRAIA-2019-1, and by the AI4Media project, funded by the European Commission (Grant 951911) under the H2020 Programme ICT-48-2020. The authors' opinions do not necessarily reflect those of the European Commission.

# Bibliography

Awasthi P, Kleindessner M, Morgenstern J (2020) Equalized odds postprocessing under imperfect group information. In: Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS 2020), Virtual Event, pp 1770–1780

Barocas S, Hardt M, Narayanan A (2019) Fairness and machine learning. fairmlbook.org, URL http://www.fairmlbook.org

Donini M, Oneto L, Ben-David S, Shawe-Taylor JS, Pontil M (2018) Empirical risk minimization under fairness constraints. In: Proceedings of the 32nd Annual Conference on Neural Information Processing Systems (NeurIPS 2018), Montreal, CA, pp 2791–2801

Dwork C, Hardt M, Pitassi T, Reingold O, Zemel R (2012) Fairness through awareness. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS 2012), Cambridge, US, pp 214–226, DOI 10.1145/2090236.2090255

Esuli A, Fabris A, Moreo A, Sebastiani F (2023) Learning to quantify. Springer Nature, Cham, CH

Fabris A, Esuli A, Moreo A, Sebastiani F (2023) Measuring fairness under unawareness of sensitive attributes: A quantification-based approach. Journal of Artificial Intelligence Research 76:1117–1180, DOI 10.1613/jair.1.14033

Hashimoto T, Srivastava M, Namkoong H, Liang P (2018) Fairness without demographics in repeated loss minimization. In: Proceedings of the 35th International Conference on Machine Learning (ICML 2018), Stockholm, SE, pp 1929–1938

He Y, Burghardt K, Lerman K (2020) A geometric solution to fair representations. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (AIES 2020), New York, US, pp 279–285, DOI 10.1145/3375627.3375864

Mehrotra A, Celis LE (2021) Mitigating bias in set selection with noisy protected attributes. In: Proceedings of the 4th ACM Conference on Fairness, Accountability, and Transparency (FAccT 2021), Toronto, CA, pp 237–248, DOI 10.1145/3442188.3445887

Singh H, Singh R, Mhasawade V, Chunara R (2021) Fairness violations and mitigation under covariate shift. In: Proceedings of the 4th ACM Conference on Fairness, Accountability, and Transparency (FAccT 2021), Toronto, CA, pp 3–13, DOI 10.1145/3442188.3445865

Wachter S, Mittelstadt B, Russell C (2020) Why fairness cannot be automated: Bridging the gap between EU non-discrimination law and AI. Computer Law and Security Review 41, DOI 10.1016/j.clsr.2021.105567, article 105567

Zafar MB, Valera I, Rogriguez MG, Gummadi KP (2017) Fairness constraints: Mechanisms for fair classification. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS 2017), Fort Lauderdale, US, pp 962–970

# Multi-Label Quantification (Extended Abstract)

Alejandro Moreo[1], Manuel Francisco[2], Fabrizio Sebastiani[1]

[1]Istituto di Scienza e Tecnologie dell'Informazione
Consiglio Nazionale delle Ricerche
56124 Pisa, Italy
E-mail: {alejandro.moreo,fabrizio.sebastiani}@isti.cnr.it

[2]Department of Computer Science and Artificial Intelligence
University of Granada
18071 Granada, Spain
E-mail: francisco@decsai.ugr.es

**Abstract.** While many quantification methods have been proposed in the past for binary problems and, to a lesser extent, single-label multiclass problems, the multi-label setting (i.e., the scenario in which the classes of interest are not mutually exclusive) remains by and large unexplored. A straightforward solution to the multi-label quantification problem could simply consist of recasting the problem as a set of independent binary quantification problems. Such a solution is simple but naïve, since the independence assumption upon which it rests is, in most cases, not satisfied. In these cases, knowing the relative frequency of one class could be of help in determining the prevalence of other related classes. We propose the first truly multi-label quantification methods, i.e., methods for inferring estimators of class prevalence values that strive to leverage the stochastic dependencies among the classes of interest in order to predict their relative frequencies more accurately. We show empirical evidence that natively multi-label solutions outperform the naïve approaches by a large margin.

## 1 Introduction

One important setting which remains to a large extent unexplored in the quantification literature is *multi-label quantification* (MLQ), the scenario in which every datapoint may belong to zero, one, or several classes at the same time; in this paper we set out to analyze MLQ systematically.

We start by noting that, since quantification systems are expected to be robust to prior probability shift, we need to test them against datasets exhibiting substantial amounts of shift. Our first contribution is the first experimental protocol specifically designed for multi-label quantification, a protocol that guarantees that the data MLQ systems are tested against do comply with the above desideratum.

We carry on by noting that a trivial solution for MLQ could simply consist of training one independent binary quantifier for each of the classes in the codeframe. However, such a solution is arguably a “naïve” one, as it assumes the classes to be independent of each other, and thus does not attempt to leverage the *class-class correlations*, i.e., the stochastic dependencies that may exist among different classes. We show empirical evidence that multi-label quantifiers built according to this naïve intuition yield suboptimal performance, and that this happens independently of the method used for training the binary quantifiers.

We then move on to studying different possible strategies for tackling MLQ, and subdivide these strategies in four groups, based on their way of addressing (if at all) the multi-label nature of the problem. While the first two groups can be instantiated by using already available techniques, the other two cannot, since this would require “aggregation” techniques (see Section 4) that leverage the stochastic relations between classes, and no such method has been proposed before. We indeed propose two such methods, called RQ and LPQ. Extensive experiments that we have carried out using 15 publicly available datasets show that, when working in combination with a classifier that itself leverages the above-mentioned stochastic relations, LPQ and (especially) RQ outperform all other MLQ techniques. The code to reproduce all our experiments is available at `https://github.com/manuel-francisco/quapy-ml/`. An extended version of this paper, available at `https://dl.acm.org/doi/10.1145/3606264` and forthcoming as (Moreo et al., 2024), reports all the experimental results, that we here omit for reasons of space.

## 2 Notation and Definitions

In this paper we use the following notation. By $\mathbf{x}$ we indicate a datapoint drawn from a domain $\mathcal{X}$ of datapoints, while by $y$ we indicate a class drawn from a finite, predefined set of classes (also known as a *codeframe*) $\mathcal{Y} = \{y_1, ..., y_n\}$, with $n$ the number of classes of interest. Symbol $\sigma$ denotes a *sample*, i.e., a non-empty set of (labelled or unlabelled) datapoints drawn from $\mathcal{X}$. By $p_\sigma(y)$ we indicate the *true* prevalence of class $y$ in sample $\sigma$, by $\hat{p}_\sigma(y)$ we indicate an *estimate* of this prevalence, and by $\hat{p}^q_\sigma(y)$ we indicate the estimate of this prevalence obtained by means of quantification method $q$. We will denote by $\mathbf{p} = (p_1, \ldots, p_n)$ a real-valued vector. When $\mathbf{p}$ is a vector of class prevalence values, then $p_i$ is short for $p_\sigma(y_i)$.

We first formalize the SLQ problem (Section 2.1) and then propose a definition of the MLQ problem (Section 2.2).

### 2.1 Single-Label Codeframes

In single-label problems, each datapoint $\mathbf{x}$ belongs to one and only one class in $\mathcal{Y}$. We denote a datapoint with its true class label as a pair $(\mathbf{x}, y)$, indicating that $y \in \mathcal{Y}$ is the true label of $\mathbf{x} \in \mathcal{X}$. We represent a set of $k$ datapoints as $\{(\mathbf{x}^{(i)}, y^{(i)})_{i=1}^{k} : \mathbf{x}^{(i)} \in \mathcal{X}, y^{(i)} \in \mathcal{Y}\}$. By $L$ we denote a collection of $\underline{\text{l}}$abelled datapoints, that we typically use as a training set, while by $U$ we denote a collection of $\underline{\text{u}}$nlabelled datapoints, that we typically use for testing purposes.

We define a *single-label hard classifier* as a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ i.e., a predictor of the class attributed to a datapoint. We will instead take a *single-label soft classifier* to be a function $s : \mathcal{X} \rightarrow \Delta^{n-1}$ with $\Delta^{n-1}$ the unit ($n$-1)-simplex (aka *probability simplex* or *standard simplex*) defined as $\Delta^{n-1} = \{(p_1, \ldots, p_n) \mid p_i \in [0,1], \sum_{i=1}^{n} p_i = 1\}$ i.e., as the domain of all vectors representing probability distributions over $\mathcal{Y}$. We define a *single-label quantifier* as a function $q : 2^{\mathcal{X}} \rightarrow \Delta^{n-1}$ i.e., a function mapping samples drawn from $\mathcal{X}$ into probability distributions over $\mathcal{Y}$.

Note that, despite the fact that the codomains of soft classifiers and quantifiers are the same, in the former case the $i$-th component of $s(\mathbf{x})$ denotes the posterior probability $\Pr(y_i|\mathbf{x})$, i.e., the probability that $\mathbf{x}$ belongs to class $y_i$ as estimated by $s$, while in the latter case it denotes the class prevalence value $p_\sigma(y_i)$ as estimated by $q$.

By $d(\mathbf{p}, \hat{\mathbf{p}})$ we denote an evaluation measure for SLQ; these measures are typically *divergences*, i.e., functions that measure the amount of discrepancy between two probability distributions. Everything we say for single-label problems applies to the binary case as well, since the latter is the special case of the former in which $n = 2$, with one class typically acting as the "positive class", and the other as the "negative class".

### 2.2 Multi-Label Codeframes

In multi-label problems each datapoint $\mathbf{x}$ can belong to zero, one, or more than one class in $\mathcal{Y}$; as a result, the sum $\sum_{i=1}^{n} p_i$ may be different from 1. We denote a datapoint with its true labels as a pair $(\mathbf{x}, Y)$, in which $Y \subseteq \mathcal{Y}$ is the set of true labels assigned to $\mathbf{x} \in \mathcal{X}$. A multi-label collection with $k$ datapoints is represented as $\{(\mathbf{x}^{(i)}, Y^{(i)})_{i=1}^{k} : \mathbf{x}^{(i)} \in \mathcal{X}, Y^{(i)} \subseteq \mathcal{Y}\}$. We define a *multi-label hard classifier* as a function $h : \mathcal{X} \rightarrow 2^{\mathcal{Y}}$ i.e., as a classifier that can assign zero, one, or more than one label to each datapoint, while we define a *multi-label soft classifier* as a function $s : \mathcal{X} \rightarrow [0,1]^n$ Note that, unlike in the single-label case, the codomain of function $s$ is not a probability simplex, since the sum $\sum_{i=1}^{n} p_i$ may be different from 1, but the set of all real-valued vectors $(p_1, \ldots, p_n)$ such that $p_i \in [0,1]$.

We define a *multi-label quantifier* as a function $q : 2^{\mathcal{X}} \rightarrow [0,1]^n$ i.e., a function mapping samples from $\mathcal{X}$ into vectors of $n$ class prevalence values, where, differently from the single-label multiclass case, the class prevalence values in a vector do not need to sum up to 1.

## 3 An Evaluation Protocol for Testing Multi-Label Quantifiers

For the evaluation of quantifiers, researchers often use the same datasets that are elsewhere used for testing classifiers. On one hand this looks natural, because both classification and quantification deal with datapoints that belong to classes in a given codeframe. On the other hand this looks problematic, since classification deals with estimating class labels for individual datapoints while quantification deals with estimating class prevalence values for *samples* (sets) of such datapoints. Simply estimating the accuracy of a quantifier on the entire test set of a dataset used for classification purposes (hereafter: a "classification dataset") would not be enough, since this would be a single prediction only, which would be akin to testing a classifier on a single datapoint only. As a result, it is customary to generate a dataset to be used for quantification purposes (a "quantification dataset") from a classification dataset by extracting from the test set of the latter a number of samples than will form the test set of the quantification dataset. Exactly how these samples are extracted is specified by an *evaluation protocol.* Different evaluation protocols for the binary case (Esuli and Sebastiani, 2015; Forman, 2005), for the single-label multiclass case (Esuli et al., 2022), and for the ordinal case (Bunse et al., 2022), have been proposed in the quantification literature.

For the binary case, the most widely adopted protocol is the so-called *artificial prevalence protocol* (APP) (Forman, 2005). The APP consists of extracting, from a set of test datapoints, many samples at controlled prevalence values. The APP takes four parameters as input: the unlabelled collection $U$, the sample size $k$, the number of samples $m$ to draw for each predefined vector of prevalence values, and a grid of prevalence values $\mathbf{g}$ (e.g., $\mathbf{g} = (0.0, 0.1, \ldots, 0.9, 1.0)$). We then generate all the vectors $\mathbf{p} = (p(\oplus), p(\ominus))$ of $n = 2$ prevalence values consisting of combinations of values from the grid $\mathbf{g}$ that represent valid distributions (i.e., such that the elements in $\mathbf{p}$ sum up to 1). For each such prevalence vector, we then draw $m$ different samples of $k$ elements each, which become the elements of our test set. The APP thus confronts the quantifier with samples characterized by class prevalence values very different from the ones seen during training, and can thus test the robustness of the quantifiers to the presence of prior probability shift. This protocol is, by far, the most popular one in the quantification literature (see, e.g., (Card and Smith, 2018; Esuli et al., 2018; Fernandes Vaz et al., 2019; Forman, 2005; Maletzke et al., 2019; Moreira dos Reis et al., 2018; Moreo and Sebastiani, 2022; Pérez-Gállego et al., 2019, 2017; Schumacher et al., 2021)).

For the single-label multiclass case (which is the closest to our concerns) the APP needs to take a slightly different form, since the number of vectors $\mathbf{p} = (p(y_1), ..., p(y_n))$ representing valid distributions for arbitrary $n$ is combinatorially high, for any reasonable grid of class prevalence values. As a solution, one can generate a number of random points on the probability simplex, without constraining the individual class prevalence values to lie on a predetermined grid; when this number is high enough, it probabilistically covers the entire spectrum of valid combinations.

However, even this form of the APP is not directly applicable to the multi-label scenario, because in this latter the class prevalence values in a valid vector do not necessarily sum up to 1. One could attempt to simply treat the multi-label problem as a set of independent binary problems, and then apply the

APP independently to each of the classes. Unfortunately, such a solution is impractical, for at least three reasons:

- The first reason is that the number of samples thus generated is exponential in $n$, since there are $m|\mathbf{g}|^n$ such combinations. Note that $n$ (the number of classes in the codeframe) cannot be set at will since it is fixed, and thus, in order to keep the number of combinations tractable in cases in which $n$ is large (in our experiments we use datasets with up to $n = 983$ classes), one would be compelled to set $m = 1$ and choose a very coarse grid $\mathbf{g}$ of values (this would anyway prove insufficient when dealing with large codeframes).
- The second and perhaps most problematic reason is that, in any case, many of the combinations are not even realisable. That is, there may be prevalence vectors for which no sample could be drawn at all. To see why, assume that, among others, we have classes $y_1$, $y_2$, $y_3$ in our codeframe, and assume that in our test set $U$, every time a datapoint is labelled with $y_1$ it is also labelled with either $y_2$ or $y_3$ but not both. This means that all samples $\sigma$ for which prevalence values $p_\sigma(y_1) \neq (p_\sigma(y_2) + p_\sigma(y_3))$ are requested, cannot be generated.
- Yet another reason why applying the APP would be, in any case, undesirable, is that the classes in most multi-label datasets typically follow a power-law distribution, i.e., there are few very popular classes and a long tail of many rare, or extremely rare, classes. The APP will sometimes impose high prevalence values for classes that in reality are very rare, which means that the sampling must be carried out *with replacement*; this would generate samples consisting of many replicas of the same few datapoints, which is clearly undesirable.

For all these reasons we have designed a brand new protocol for MLQ, that we call ML-APP, since it is an adaptation of the APP to multi-label datasets. The protocol amounts to performing multiple rounds of the APP, each targeting a specific class, but with the range of prevalence values explored for each class being limited by the amount of available positive examples. This allows all samples to be drawn *without* replacement. In each round, a class $y_i$ is actively sampled at controlled prevalence values while the prevalence values for the remaining classes are not predetermined.

The ML-APP covers the entire spectrum of class prevalence values, by drawing without replacement, for every single class. This means that, for large enough classes, there will be samples for which the prevalence of the class exhibits a large prior probability shift with respect to the training prevalence, while for rare classes the amount of shift will be limited by the availability of positive examples. Note that, when actively sampling a class $y_i$, any other class $y_j$ will co-occur with it with a probability that depends on the correlation between $y_i$ and $y_j$. For cases in which the class $y_i$ being sampled is completely independent of the class $y_j$, the samples generated will display a class prevalence for $y_j$ that is approximately similar to the prevalence of $y_j$ in $U$. In other words, samples generated via the ML-APP have a desirable property, i.e., they preserve the stochastic correlations between the classes while also exhibiting widely varying degrees of prior probability shift. Finally, note that the total number of samples that can be generated via the ML-APP can vary from dataset to dataset (even if they have the same number of classes), and depends on the actual number of positive instances for each class that are contained in the dataset. In any case, the maximum number of samples that can be generated via the ML-APP is bounded by $mn|\mathbf{g}|$.

## 4 Performing Multi-Label Quantification

In this section we present the multi-label quantification methods that we will experimentally compare in Section 5. Throughout this paper we will focus on *aggregative* quantification methods, i.e., methods that require all unlabelled datapoints to be classified (by a hard or a soft classifier, depending on the method) as an intermediate step, and that aggregate the individual (hard or soft) predictions in some way to generate the class prevalence estimates. The reason why we focus on aggregative methods is that they are by far the most popular quantification methods in the literature, and that this focus allows us an easier exposition. We will later show how the most interesting intuitions for performing MLQ that we discuss in this paper also apply to the non-aggregative case.

### 4.1 Multi-Label Quantification

In this paper we will describe and compare many different (aggregative) MLQ methods. In order to better assess their relative merits, we subdivide them into four different groups, depending on whether the correlations between different classes are exploited in the classification phase (i.e., by the classifier which provides input to an aggregative quantifier), or in the aggregation phase (i.e., in the phase in which the individual predictions are aggregated), or in both phases, or in neither of the two phases.

The first and simplest such group is that of MLQ methods that treat each class as completely independent, and thus solve $n$ independent binary quantification problems. We call such an approach BC+BA ("binary classification followed by binary aggregation"), since in both the classification phase and the aggregation phase we treat the multi-label task as $n$ independent binary tasks; we thus disregard, in both phases, the correlations among classes when predicting their class prevalence values. This is similar to the binary relevance (BR) problem transformation for classification, and consists of transforming the multi-label dataset $L$ into a set of binary datasets $L_1, \ldots, L_n$ in which $L_i = \{(\mathbf{x}, \mathbf{1}[y_i \in Y]) : (\mathbf{x}, Y) \in L\}$ is labelled according to $\mathcal{Y}_i = \{\mathbf{0}, \mathbf{1}\}$, since the datapoints are relabelled using the indicator function $\mathbf{1}[z]$ that returns $\mathbf{1}$ (the minority class) if $z$ is true or $\mathbf{0}$ (the majority class) otherwise. BC+BA methods then train one quantifier $q_i$ for each training set $L_i$. At inference time, the prevalence vector for a given sample $\sigma$ is computed as $\mathbf{p}_\sigma^{\text{BC+BA}} = (p_\sigma^{q_1}(\mathbf{1}), p_\sigma^{q_2}(\mathbf{1}), \ldots, p_\sigma^{q_n}(\mathbf{1}))$. Although this is technically a multi-label quantification method, BC+BA is actually the trivial solution that we expect any truly multi-label quantifier to beat.

A second, less trivial group is that of MLQ methods based on the use of binary aggregative quantifiers that receive input from (truly) multi-label classifiers. Methods in this group consist of $n$ independent binary aggregative quantifiers that rely on the (hard or soft) predictions returned by a classifier natively designed to tackle the multi-label problem. Each binary quantifier takes into account only the predictions for its associated class, disregarding the predictions for the other classes. This represents a straightforward solution to the MLQ problem, as it simply combines already existing technologies (binary aggregative quantifiers built via off-the-shelf methods and (truly) multi-label classifiers built via off-the-shelf methods). In such a setting, the classification stage is influenced by the class-class correlations, but the quantification methods in charge of producing the class prevalence estimates for each class do not pay attention to any such correlation, and are disconnected from each other. Since methods in this group will consist of a (truly) multi-label classification phase followed by a binary quantification phase, we will refer to this group of methods as MLC+BA.

We next propose a third group of MLQ systems, i.e., ones consisting of natively multi-label quantification methods that receive ad input the outputs of $n$ independent binary classifiers. Methods like these represent a non-trivial novel solution for the field of quantification, because no natively multi-label quantification method has been proposed so far in the literature; in Section 4.1.1 we propose some such methods. In order to clearly evaluate the merits of such a multi-label aggregation phase, as the underlying classifiers we use independent binary classifiers only. For this reason, we will call this group of methods BC+MLA.

The methods in the fourth and last group that we consider consist of combinations of a (truly) multi-label classification method and a (truly) multi-label quantification method among our newly proposed ones; this allows to exploit the class dependencies both at the classification stage and at the aggregation stage. We call this group of methods MLC+MLA.

Figure 1 illustrates in diagrammatic form the four types of multi-label quantification methods we study in this paper. In order to generate members of these four classes, we already have off-the-shelf components for implementing the binary classification, multi-label classification, and binary aggregation phases, but we have no known method from the literature to implement multi-label aggregation; Sections 4.1.1 and 4.1.2 are devoted to proposing two novel methods of this type.

#### 4.1.1 Exploiting Class-Class Correlations at the Aggregation Stage by means of Regression

Let us assume we have a multi-label quantifier $q$ of type BC+BA or MLC+BA. Our idea is to detect how quantifier $q$ fails in capturing the correlations between classes, and to correct $q$ accordingly. This is somehow similar to the type of correction implemented in ACC (with respect to CC) and PACC (with respect to PCC). However, we will formalize this intuition as a general regression problem, thus not necessarily assuming this correction to be linear (as ACC and PACC instead do).

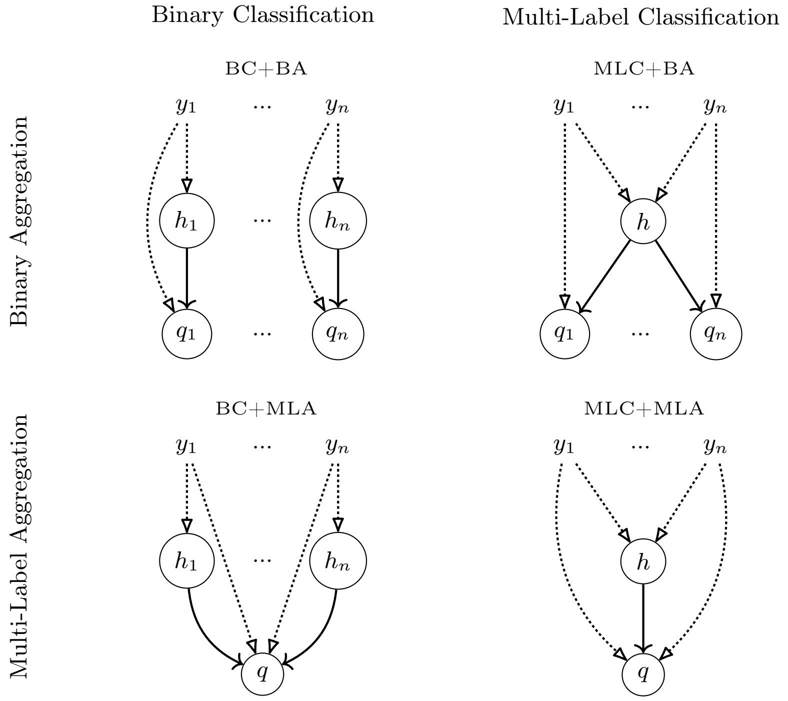

**Fig. 1.** The four groups of multi-label quantification methods. Dotted lines connecting class labels with a model (classifier or quantifier) indicate that the model learns from (or has access to) the class labels of the training datapoints. Solid lines connecting classifiers with quantifiers indicate a transfer of outputs from the classifier to the quantifier. With a slight deviation from our notation, here $h$ denotes any classifier, hard or soft.

Roughly speaking, the idea that underlies our method is that of learning a regression function $r : \mathbb{R}^n \rightarrow \mathbb{R}^n$ that takes as input the vector of prevalence values as estimated by $q$, and returns a vector of *corrected* prevalence values. More concretely, we split our training set $L$ into two parts, $L_Q$ (that we use for training our quantifier $q$) and $L_R$ (that we use for training a regressor $r$ i.e., a function $r : \mathbb{R}^n \rightarrow \mathbb{R}^n$).[1] We then use the ML-APP protocol described in Section 3 to extract, from set $L_R$, a new training set $\mathcal{R} = \{\sigma_i \sim \text{ML-APP}(L_R, k, m, \mathbf{g})\}$ of $l$ samples, where $k$ (sample size), $m$ (number of samples to draw for each prevalence value on the grid), and $\mathbf{g}$ (grid of prevalence values) are the parameters of the ML-APP protocol.

Having done this, we first train our quantifier $q$ on $L_Q$. Note that, since $q$ is a multi-label quantifier, it is a function that, given a sample $\sigma$, returns a vector $\hat{\mathbf{p}}^q_\sigma$ of $n$ class prevalence values, not necessarily summing up to 1. We then apply $q$ to all the samples in our newly created dataset $\mathcal{R}$. As a result, for each sample $\sigma_i \in \mathcal{R}$, we haveobtain a pair $(\hat{\mathbf{p}}^q_{\sigma_i}, \mathbf{p}_{\sigma_i})$, where $\hat{\mathbf{p}}^q_{\sigma_i}$ is the vector of the $n$ prevalence values estimated by $q$, and $\mathbf{p}_{\sigma_i}$ is the vector of the $n$ true prevalence values. We use this set of $l$ pairs as the training set for training a multi-output regressor $r : \mathbb{R}^n \rightarrow \mathbb{R}^n$ that takes as input a vector of $n$ "uncorrected" prevalence values (i.e., values generated without exploiting the class-class correlations) and returns a vector of $n$ "corrected" prevalence values (i.e., values generated by exploiting the class-class correlations); for training the regressor we can use any off-the-shelf multi-output regression algorithm. Note that the regressor indeed captures the correlations between classes, since it receives as input, for each sample, the class prevalence estimates for all the $n$ classes.[2]

At inference time, given an (unlabelled) sample $\sigma$, we first obtain a preliminary estimate of the class prevalence values $\hat{\mathbf{p}}^q_{\sigma}$ by means of $q$, and then apply the correction learned by the regressor $r$, thus computing $\hat{\mathbf{p}}^r_{\sigma} = r(\hat{\mathbf{p}}^q_{\sigma})$. We then normalize, by means of clipping,[3] every prevalence value in $\hat{\mathbf{p}}^r_{\sigma}$ so that it falls in the $[0, 1]$ interval, and return the estimate.

As noted above, the regressor exploits the class-class correlations during the aggregation phase. This means that, according to the subdivision of MLQ methods illustrated in Table 1, the addition of a regression layer on top of an existing quantifier $q$ has the effect of transforming a BC+BA method into a BC+MLA method, or of transforming a MLC+BA method into a MLC+MLA method.

**4.1.2 Exploiting Class-Class Correlations at the Aggregation Stage by means of Label Powersets** We investigate an alternative way of modelling class-class correlations at the quantification level, this time by gaining inspiration from label powersets (LPs – see (Spolaôr et al., 2013)) and the heuristics for making their application tractable.

LP is a problem transformation technique devised for transforming any multi-label classification problem into a single-label one by replacing the original codeframe with another one that encodes subsets of this codeframe into "synthetic" classes. This problem transformation is directly applicable to the case of quantification as well. Of course, the combinatorial explosion of the number of synthetic classes has to be controlled somehow but, fortunately enough, the same heuristics investigated for MLC can come to the rescue.

Our method (which we here call LPQ, for "label powerset -based quantification") consists of generating, by means of any existing clustering algorithm, a set $\mathcal{C}$ of (non-overlapping) clusters consisting of few classes each, before applying the LP strategy, so that the number of possible synthetic classes remains under reasonable bounds. For example, if our codeframe has $n = 100$ classes, extracting 25 clusters of 4 classes each results in the maximum possible number of synthetic classes being $25 \cdot 2^4 = 400$, which is much smaller than the original $2^{100}$. We perform this clustering by treating classes in $\mathcal{Y}$ as instances and training datapoints as features, so that a class is represented by a binary vector of datapoints, where 1 indicates that the datapoint belongs to the class and 0 that it does not. The clustering algorithm is thus expected to put classes displaying similar assignment patterns (i.e., classes that tend to label the same documents) in the same cluster.

Once we have performed the clustering, given the subset of classes $\mathcal{Y}_c \subseteq \mathcal{Y}$ contained in each cluster $c \in \mathcal{C}$, we need to convert the multi-label assignments into single-label assignments. This amounts to defining a mapping $2^{\mathcal{Y}_c} \rightarrow \mathcal{Y}'_c$, so that, e.g., the set of classes $\{y_1, y_5, y_6\} \subseteq \mathcal{Y}_c$ corresponds to a new synthetic class $y_{1:5:6} \in \mathcal{Y}'_c$. Once (single) labels have been assigned, we can train a single-label quantifier. This process is independently carried out for each cluster. codeframe take the single-label codeframe $\mathcal{Y}'_c$ determined from the $2^{\mathcal{Y}} \rightarrow \mathcal{Y}'$ multi-label-to-single-label mapping (a mapping that, e.g., would attribute to the set of classes $\{y_1, y_5, y_6\} \subseteq \mathcal{Y}_c$ the synthetic class $y_{1:5:6} \in \mathcal{Y}'_c$) and train a single-label quantifier on it; this needs to be repeated for each cluster. At inference time, in order to provide class prevalence estimates for the classes in $\mathcal{Y}_c$ from the predictions made for the classes in $\mathcal{Y}'_c$ by the above-mentioned quantifier, we have to "reverse" the multi-label-to-single-label mappingassignment. This process is straightforward since the mapping is bijective. By doing so, we can reconstruct the estimated prevalence value for class $y_i \in \mathcal{Y}_c$ as , so that the estimated prevalence value of $y_i \in \mathcal{Y}_c$ is the sum of the estimated prevalence values of all labels $y' \in \mathcal{Y}'_c$ that involve $y_i$.; performing this for each cluster $c \in \mathcal{C}$ returns prevalence estimates for all classes $y_i \in \mathcal{Y}$. This process is repeated for each cluster $c \in \mathcal{C}$ in order to obtain prevalence estimates for all classes $y_i \in \mathcal{Y}$.

More formally, let us define a matrix $\mathbf{A}$ that records the label assignment in cluster $c$, so that $a_{ij} = 1$ if the set of classes represented by the synthetic class $y'_i \in \mathcal{Y}'_c$ contains class $y_j \in \mathcal{Y}_c$, and $a_{ij} = 0$ if this is not the case. Note that $\mathbf{A}$ has as many rows as there are classes in $\mathcal{Y}'_c$ and as many columns as there are classes in $\mathcal{Y}_c$. Once our single-label quantifier $q$ produces an output $\hat{\mathbf{p}}^q_\sigma$, we only need to compute the product $(\hat{\mathbf{p}}^q_\sigma)^\top \mathbf{A}$ to obtain the vector of prevalence estimates for the classes in $\mathcal{Y}_c$. Performing all this for each cluster $c \in \mathcal{C}$ returns prevalence estimates for all classes $y_i \in \mathcal{Y}$.

In principle, the disadvantage of this method is that it cannot learn the correlations between classes that belong to different clusters. However, the method is based on the intuition that classes that are indeed correlated tend to end up in the same cluster, and that the inability to model correlations between classes that belong to different clusters will be more than compensated by the reduction in the number of combinations that one needs to take into account.

## 5 Experiments

In this section we turn to describing the experiments we have carried out in order to evaluate the performance of the different methods for MLQ that we have presented in the previous sections. In Section 5.1 we discuss the evaluation measure we adopt, while in Section 5.2 we describe the datasets on which we perform our experiments. The results, omitted here for reasons of space, can be found in the extended version of this paper at `https://dl.acm.org/doi/10.1145/3606264`.

### 5.1 Evaluation Measures

Any evaluation measure for binary quantification can be easily turned into an evaluation measure for multi-label quantification, since evaluating a multi-label quantifier can be done by evaluating how well the prevalence value $p(y_i)$ of each class $y_i \in |\mathcal{Y}|$ is approximated by the prediction $\hat{p}(y_i)$. As a result, it is natural to take a standard measure $d(\mathbf{p}, \hat{\mathbf{p}})$ for the evaluation of binary quantification, and turn it into a measure

$$\mathrm{D}(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{n}\sum_{i=1}^{n} d((p_i, (1-p_i)), (\hat{p}_i, (1-\hat{p}_i))) \tag{1}$$

for the evaluation of multi-label quantification. (This is exactly what we do in multi-label *classification*, in which we take $F_1$, a standard measure for the evaluation of binary classification, and turn it into macroaveraged $F_1$, which is the standard measure for the evaluation of multi-label classification.)

The study of evaluation measures for binary (and single-label multiclass) quantification performed in (Sebastiani, 2020) concludes that the most satisfactory such measures are *absolute error* and *relative absolute error*; these are the two measures that we are going to use in this paper. In the binary case, absolute error is defined as

$$\begin{aligned}\mathrm{ae}(\mathbf{p}, \hat{\mathbf{p}}) &= \frac{|p_1 - \hat{p}_1| + |p_2 - \hat{p}_2|}{2} \\ &= \frac{|p_1 - \hat{p}_1| + |(1-p_1) - (1-\hat{p}_1)|}{2} \\ &= |p_1 - \hat{p}_1|\end{aligned} \tag{2}$$

which yields the multi-label version

$$\mathrm{AE}(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{n}\sum_{i=1}^{n} |p_i - \hat{p}_i| \tag{3}$$

In the binary case, relative absolute error is instead defined as

$$\begin{aligned}\mathrm{rae}(\mathbf{p}, \hat{\mathbf{p}}) &= \frac{1}{2}\left(\frac{|p_1 - \hat{p}_1|}{p_1} + \frac{|p_2 - \hat{p}_2|}{p_2}\right) \\ &= \frac{1}{2}\left(\frac{|p_1 - \hat{p}_1|}{p_1} + \frac{|(1-p_1) - (1-\hat{p}_1)|}{(1-p_1)}\right)\end{aligned} \tag{4}$$

which yields the multi-label version

$$\mathrm{RAE}(\mathbf{p}, \hat{\mathbf{p}}) = \frac{1}{2n}\sum_{i=1}^{n}\left(\frac{|p_i - \hat{p}_i|}{p_i} + \frac{|(1-p_i) - (1-\hat{p}_i)|}{(1-p_i)}\right) \tag{5}$$

Since RAE is undefined when $p_i = 0$ or $p_i = 1$, we smooth the probability distributions $\mathbf{p}$ and $\hat{\mathbf{p}}$ via additive smoothing; in the binary case, this maps a distribution $\mathbf{p} = (p_i, (1-p_i))$ into

$$\mathrm{s}(\mathbf{p}) = \left(\frac{\epsilon + p_i}{2\epsilon + 1}, \frac{\epsilon + (1-p_i)}{2\epsilon + 1}\right) \tag{6}$$

with $\epsilon$ the smoothing factor; following (Forman, 2008), we set $\epsilon = (2|\sigma|)^{-1}$.

In the experiments we describe in Section 5, the trends we observe and the conclusions we draw for AE hold for RAE as well. In Section 5 we will thus report our results in terms of AE only, deferring the results in terms of RAE to (Moreo et al., 2024).

### 5.2 Datasets

For our experiments we use 15 popular MLC datasets, including 3 datasets specific to text classification (Reuters-21578,[4] Ohsumed (Hersh et al., 1994), and RCV1-v2[5]), plus all the datasets linked from the SCIKIT-MULTILEARN package (Szymanski and Kajdanowicz, 2017) with the exception of the RCV1-v2 subsets (we omit them since we already include the much larger collection from which they were extracted). We refer to the original sources for detailed descriptions of these datasets.[6]

For the three textual datasets, we apply lowercasing, stop word removal, and punctuation removal, as implemented in SCIKIT-LEARN,[7] and mask numbers with a special token. We retain all terms appearing at least 5 times in the training set, and convert the resulting set of words into (sparse) tfidf-weighted vectors using SCIKIT-LEARN's default vectorizer.[8]

For all datasets, we remove very rare classes (i.e., those with fewer than 5 training examples) from consideration, since they pose a problem when it comes to generating validation (i.e., held-out data) sets. Indeed, since we optimize the hyperparameters for all the methods we use (as explained below), we need validation sets, and it is sometimes impossible to have positive examples for these classes in both the training and validation sets (let us remember that pure stratification in multi-label datasets is not always achievable, as argued in (Sechidis et al., 2011; Szymański and Kajdanowicz, 2017)). Note that all this only concerns the training set, and has nothing to do with the test set, which can include (and indeed includes, for most datasets) extremely rare classes, since removing classes that are rare in the test set would lead to an unrealistic experimentation. Note also that removing classes that are rare in the training set is "fair", i.e., equally affects all methods that we experimentally compare, since all of them involve hyperparameter optimization. Finally, note that, whenever a method requires generating *additional* (and maybe nested) validation sets, it is inevitably exposed to the problems mentioned above, and can thus be at a disadvantage with respect to other methods that do not require additional validation data. (Moreo et al., 2024) gives a complete description of the datasets we use (after deleting rare classes), along with some useful statistics proposed in (Read, 2010; Zhang and Zhou, 2014), and shows the distribution of prevalence values for each dataset. Note that, in most datasets, this distribution obeys a power law.

We set the parameters of the ML-APP for generating test samples (see Section 3) as follows. We fix the sample size to $k = 100$ in all cases. We set the grid of prevalence values to $\mathbf{g} = \{0.00, 0.01, \ldots, 0.99, 1.00\}$ in all cases but for dataset `Delicious`, since in this latter the number of combinations thus generated would be intractable, given that this is dataset with no fewer than 983 classes; for

Delicious we use the coarser-grained grid $\mathbf{g} = \{0.00, 0.05, \ldots, 0.95, 1.00\}$. We set $m$ (the number of samples to be drawn for each prevalence value) independently for each dataset, to the smallest number that yields more than 10,000 test samples ($m$ ranges from 1 in Delicious to 40 in Birds).

We break down the results into three groups, each corresponding to a different amount of shift. The rationale behind this choice is to allow for a more meaningful analysis of the quantifiers' performance, since the APP (and, by extension, the ML-APP) has often been the subject of criticism for generating samples exhibiting degrees of shift that are judged unrealistic and unlikely to occur in real cases (Esuli and Sebastiani, 2015; Hassan et al., 2021). We instead believe that general-purpose quantification methods should be tested in widely varying situations, from low-shift to high-shift ones, and we thus prefer to test all such scenarios, but split the corresponding results into groups characterized by of more or less homogeneous amounts of shift.

More specifically, for each test sample generated via the ML-APP, we compute its prior probability shift with respect to the training set in terms of AE between the vectors of training and test class prevalence values. We then bring together all the resulting shift values and split the range of such values in three equally-sized intervals (that we dub *low shift*, *mid shift*, and *high shift*). The accuracy values we report are thus not averages across the same number of experiments, since the ML-APP often tends to generate more samples in the low-shift region than samples in the mid-shift region and (above all) in the high-shift region. The number of samples, as well as the distribution of shift values, depends on each dataset.

The results of our experiments are omitted here for reasons of space, and can be found in the extended version of this paper at https://dl.acm.org/doi/10.1145/3606264. The results clearly show that there is an ordering BC+BA $\prec$ MLC+BA $\prec$ BC+MLA $\prec$ MLC+MLA, in which $\prec$ means "performs worse than", which holds, independently of the base quantifier of choice, in almost all cases. The same experiments also indicate that *there is a substantial improvement in performance that derives from simply replacing the binary classifiers with one multi-label classifier* (moving from BC+BA to MLC+BA or from BC+MLA to MLC+MLA), i.e., from bringing to bear the class-class correlations at the classification stage, and that *there is an equally substantial improvement when binary aggregation is replaced by multi-label aggregation* (switching from BC+BA to BC+MLA or from MLC+BA to MLC+MLA), i.e., when the class-class correlations are exploited at the aggregation stage. What also emerges from these results is that, consistently with the above observations, *the best-performing group of methods is* MLC+MLA, i.e., methods that explicitly take class dependencies into account *both* at the classification stage and at the aggregation stage. Methods that learn from the stochastic correlations among the classes perform way better than methods that do not, even in the low-shift regime. Overall, the best-performing method on average is MLC+MLA when equipped with PCC as the base quantifier.

## 6 Conclusions

In this paper we have investigated MLQ, a quantification task which had remained, since the origins of quantification research, essentially unexplored.

The first contribution of this paper is ML-APP, the first protocol for the evaluation of MLQ systems that is able to confront these systems with samples that exhibit from low to high levels of prior probability shift while at the same time preserving the stochastic correlations between the classes.

As a second contribution, we have also described and experimentally compared a number of MLQ methods. For ease of exposition, we have particularly focused on multi-label quantifiers that work by aggregating predictions for individual datapoints issued by a classifier ("aggregative" multi-label quantifiers), and have subdivided them into four groups, based on whether the correlations between classes are brought to bear in the classification stage (MLC+BA), in the quantification stage (BC+MLA), in both stages (MLC+MLA), or in neither of the two stages (BC+BA). Some of these methods (specifically: those in the BC+BA and MLC+BA groups) are trivial combinations of available classification and quantification methods, while others (specifically: those in the BC+MLA and MLC+MLA groups) are non-obvious, and proposed here for the first time. The thorough experimentation (reported in (Moreo et al., 2024)) that we have carried out on a large number of datasets has clearly shown that there is a substantial improvement in performance that derives from simply replacing binary classifiers with truly multi-label classifiers (i.e., from switching from BC to MLC), and that there is an equally substantial improvement when binary aggregation is replaced by truly multi-label aggregation (i.e., when switching from BA to MLA). Consistently with these two intuitions, MLC+MLA methods unequivocally prove the best of the lot; of the two MLC+MLA methods we have proposed, RQ proves clearly superior to LPQ. In the light of this superiority of MLA with respect to BA, it is also interesting that both RQ and LPQ can be straightforwardly used in association to non-aggregative quantifiers too.

## Acknowledgments

The work of A. Moreo and F. Sebastiani has been supported by the SoBigData++ project, funded by the European Commission (Grant 871042) under the H2020 Programme INFRAIA-2019-1, by the AI4Media project, funded by the European Commission (Grant 951911) under the H2020 Programme ICT-48-2020, and by the SoBigData.it and FAIR projects funded by the Italian Ministry of University and Research under the NextGenerationEU program; the authors' opinions do not necessarily reflect those of the funding agencies. The work of M. Francisco has been supported by the FPI 2017 predoctoral programme, from the Spanish Ministry of Economy and Competitiveness (MINECO), grant BES-2017-081202.

# Bibliography

Bunse, M., Moreo, A., Sebastiani, F., and Senz, M. (2022). Ordinal quantification through regularization. In *Proceedings of the 33rd European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML / PKDD 2022)*, Grenoble, FR. Forthcoming.

Card, D. and Smith, N. A. (2018). The importance of calibration for estimating proportions from annotations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2018)*, pages 1636–1646, New Orleans, US.

Esuli, A., Moreo, A., and Sebastiani, F. (2018). A recurrent neural network for sentiment quantification. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM 2018)*, pages 1775–1778, Torino, IT.

Esuli, A., Moreo, A., and Sebastiani, F. (2022). LeQua@CLEF2022: Learning to Quantify. In *Proceedings of the 44th European Conference on Information Retrieval (ECIR 2022)*, pages 374–381, Stavanger, NO.

Esuli, A. and Sebastiani, F. (2015). Optimizing text quantifiers for multivariate loss functions. *ACM Transactions on Knowledge Discovery and Data*, 9(4):Article 27.

Fernandes Vaz, A., Izbicki, R., and Bassi Stern, R. (2019). Quantification under prior probability shift: The ratio estimator and its extensions. *Journal of Machine Learning Research*, 20:79:1–79:33.

Forman, G. (2005). Counting positives accurately despite inaccurate classification. In *Proceedings of the 16th European Conference on Machine Learning (ECML 2005)*, pages 564–575, Porto, PT.

Forman, G. (2008). Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery*, 17(2):164–206.

Hassan, W., Maletzke, A. G., and Batista, G. (2021). Pitfalls in quantification assessment. In *Proceedings of the CIKM 2021 Workshop on Learning to Quantify*, Virtual Event.

Hersh, W., Buckley, C., Leone, T., and Hickman, D. (1994). OHSUMED: An interactive retrieval evaluation and new large text collection for research. In *Proceedings of the 17th ACM International Conference on Research and Development in Information Retrieval (SIGIR 1994)*, pages 192–201, Dublin, IE.

Maletzke, A., Moreira dos Reis, D., Cherman, E., and Batista, G. (2019). DyS: A framework for mixture models in quantification. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI 2019)*, pages 4552–4560, Honolulu, US.

Moreira dos Reis, D., Maletzke, A. G., Silva, D. F., and Batista, G. E. (2018). Classifying and counting with recurrent contexts. In *Proceedings of the 24th ACM International Conference on Knowledge Discovery and Data Mining (KDD 2018)*, pages 1983–1992, London, UK.

Moreo, A., Francisco, M., and Sebastiani, F. (2024). Multi-label quantification. *ACM Transactions on Knowledge Discovery and Data*, 18(1):Article 4.

Moreo, A. and Sebastiani, F. (2022). Tweet sentiment quantification: An experimental re-evaluation. *PLOS ONE*, 17(9):1–23.

Pérez-Gállego, P., Castaño, A., Quevedo, J. R., and del Coz, J. J. (2019). Dynamic ensemble selection for quantification tasks. *Information Fusion*, 45:1–15.

Pérez-Gállego, P., Quevedo, J. R., and del Coz, J. J. (2017). Using ensembles for problems with characterizable changes in data distribution: A case study on quantification. *Information Fusion*, 34:87–100.

Read, J. (2010). *Scalable multi-label classification.* PhD thesis, University of Waikato, Hamilton, NZ.

Schumacher, T., Strohmaier, M., and Lemmerich, F. (2021). A comparative evaluation of quantification methods. arXiv:2103.03223.

Sebastiani, F. (2020). Evaluation measures for quantification: An axiomatic approach. *Information Retrieval Journal*, 23(3):255–288.

Sechidis, K., Tsoumakas, G., and Vlahavas, I. (2011). On the stratification of multi-label data. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML/PKDD 2011)*, pages 145–158, Athens, GR.

Spolaôr, N., Cherman, E. A., Monard, M. C., and Lee, H. D. (2013). A comparison of multi-label feature selection methods using the problem transformation approach. *Electronic Notes in Theoretical Computer Science*, 292:135–151.

Szymanski, P. and Kajdanowicz, T. (2017). A scikit-based Python environment for performing multi-label classification. arXiv:1702.01460 [cs.LG].

Szymański, P. and Kajdanowicz, T. (2017). A network perspective on stratification of multi-label data. In *Proceedings of the 1st International Workshop on Learning with Imbalanced Domains: Theory and Applications (LIDTA 2017)*, pages 22–35, Skopje, MK.

Zhang, M.-L. and Zhou, Z.-H. (2014). A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837.

# Invariance assumptions for class distribution estimation

Dirk Tasche[0000−0002−2750−2970]

Independent Scholar, dirk.tasche@gmx.net

First version: May 25, 2023
This version: August 27, 2023

**Abstract.** We study the problem of class distribution estimation under dataset shift. On the training dataset, both features and class labels are observed while on the test dataset only the features can be observed. The task then is the estimation of the distribution of the class labels, i.e. the estimation of the class prior probabilities, in the test dataset. Assumptions of invariance between the training joint distribution of features and labels and the test distribution can considerably facilitate this task. We discuss the assumptions of covariate shift, factorizable joint shift, and sparse joint shift and their implications for class distribution estimation.

**Keywords:** Class prior estimation · quantification · prevalence estimation · dataset shift · distribution shift · covariate shift · factorizable joint shift · sparse joint shift.

## 1 Introduction

We consider class distribution estimation against the backdrop of dataset shift (also called distribution shift) between training and test dataset. On the training dataset, both features and class labels are observed while on the test dataset only the features can be observed. In this context, important tasks of interest are the prediction of the labels (classification) and the estimation of the label distribution (class distribution estimation) in the test dataset. In the literature, class distribution estimation is also referred to as class prior estimation, class prevalence estimation, quantification, and with a number of other terms.

Referring to Forman (2005), Esuli et al. (2023, Preface) made the following case for class distribution estimation as a research topic of its own: “In a number of applications involving classification, the final goal is not determining which class (or classes) individual unlabelled instances belong to, but estimating the prevalence (or ‘relative frequency’, or ‘prior probability’) of each class in the unlabelled data.”

Class distribution estimation for the target (test) dataset when its distribution is allowed to differ from the distribution of the training (source) dataset, in general, is an ill-posed problem, because joint target (test) distributions of features and labels whose marginal feature distributions perfectly match the observed target feature distribution cannot be distinguished. Constraints are needed on the range of joint target distributions taken into account for the estimation exercise in order to make the problem well-posed. The consideration of causality is a popular approach for specifying such constraints. Typically, this approach leads to making a decision either for prior probability shift (label shift) or for covariate shift as the model for the joint target distribution (Fawcett and Flach, 2005).

Other approaches to the problem include

– Assumptions on the evolution of parts of the joint distribution of labels and features between training and test times (e.g. Zhang et al., 2013; Krempl et al., 2019).
– Implicit assumptions, for instance by the choice of the distance function for measuring the difference of the source and the target feature distributions (e.g. Hofer, 2015; Kirchmeyer et al., 2021).

In this paper, we revisit three approaches to class distribution estimation and, more generally, to modelling dataset shift under invariance assumptions between the joint source and target distributions: Covariate shift (Shimodaira, 2000), factorizable joint shift (FJS, He et al., 2021), and sparse joint shift (SJS, Chen et al., 2022).

The contribution of this paper to the literature is twofold. On the one hand, two new approaches to class distribution estimation under covariate shift are presented. These approaches may prove useful for cross-checking estimates obtained by application of the popular ‘probabilistic classify and count’ approach. On the other hand, some results on FJS and SJS which were presented in Tasche (2022b) and Tasche (2023) in uncommon notation are revisited in a notation more familiar to the machine learning community.

Class distribution estimation under prior probability shift has been receiving a lot of attention by the research community for at least the last sixty years, beginning with Gart and Buck (1966) if not earlier. For this reason, in this paper we do not dive into any detail of prior probability shift. Regarding this topic, we refer to the recent overviews by González et al. (2017) and Esuli et al. (2023) of the literature on class distribution estimation under prior probability shift and the references therein.

This paper is organised as follows:

– Section 2 ‘Notation and general assumptions’ sets the scene in technical terms for the remainder of the paper.
– Section 3 ‘Types of dataset shift with invariance assumptions’ provides the formal definitions of the four most important types of distribution shift considered in more or less detail in the following: Prior probability shift, covariate shift, factorizable joint shift (FJS), and sparse joint shift (SJS).
– Section 4 ‘Covariate shift’ looks at class distribution estimation under covariate shift, based on previous work by Card and Smith (2018) and Tasche (2022a). Eq. (9b) and Proposition 1 are new results.
– Section 5 ‘Factorizable joint shift (FJS)’ revisits the notion of distribution shift proposed by He et al. (2021). FJS is found to be unsuitable for class

distribution estimation due to lack of identifiability unless additional constraints are applied.

- Section 6 'Sparse joint shift (SJS)' summarises findings of Chen et al. (2022) and Tasche (2023). Proposition 3 on the 'conditional confusion matrix approach' presents a new interpretation of a result of Tasche (2023). SJS is shown to be a generalisation of prior probability shift and found to be a suitable assumption for designing class distribution estimators.
- The paper concludes with a brief assessment of the findings in Section 7.

## 2 Notation and general assumptions

We adopt notation and assumptions similar to the setting used in Scott (2019):

There are a feature space $\mathcal{X}$ (not necessarily with $\mathcal{X} \subset \mathbb{R}^d$ for any fixed $d$) and a label space $\mathcal{Y} = \{1, \ldots, \ell\}$ for some integer $\ell \geq 2$. This is the common machine learning setting for multinomial classification and class distribution estimation.

As in Scott (2019, Section 1.2), "… there are two distributions, $P$ and $Q$, referred to as the *source and target distributions*. We consider the semi-supervised setting where the learner observes $(X_1, Y_1), \ldots, (X_m, Y_m) \sim P$ and $X_{m+1}, \ldots, X_{m+n} \sim Q_X$ …".

$P$, $Q$ are probability distributions on $\mathcal{X} \times \mathcal{Y}$. $P$ is also called *training distribution*, $Q$ *test distribution*. $X$ is a generic random variable which shows the features of an object (or instance), $Y$ is a generic random variable showing the class label of an object. $Q_X$ stands for the marginal distribution of the features under the target distribution.

We suppose for the purpose of this paper that the sample sizes $m$ of the training sample and $n$ of the test sample are sufficiently large if not infinite such that $P$ and $Q_X$ can be perfectly inferred and assumed to be known.

Class distribution estimation then may be phrased as the problem of how to find the marginal distribution $Q_Y$ of the labels (i.e. the class distribution) under the target distribution, i.e. the prior probabilities $Q[Y = 1]$, …, $Q[Y = \ell]$.

**Densities.** In the following, we assume that the joint target distribution $Q$ of features and labels $(X, Y)$ is absolutely continuous (see Klenke, 2013, Definition 7.30) with respect to the joint source distribution $P$ of $(X, Y)$. We also suppose that $p = p(x, y)$ is a joint density of $(X, Y)$ under $P$ and $q = q(x, y)$ is a joint density of $(X, Y)$ under $Q$, with respect to some third measure. Absolute continuity of $Q$ with respect to $P$ is implied in particular if the support of $Q$ is a subset of the support of $P$, i.e. if it holds that

$$q(x, y) > 0 \quad \Rightarrow \quad p(x, y) > 0. \tag{1}$$

For the sake of simplying the notation, for the remainder of the paper we assume that (1) is true.

Under the assumption that (1) holds, define the general *importance weight* function $w(x, y)$ for $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ by

$$w(x, y) = \begin{cases} \frac{q(x,y)}{p(x,y)}, & \text{for } p(x, y) > 0, \\ 0, & \text{for } p(x, y) = 0. \end{cases} \tag{2a}$$

Function $w$ reflects the change caused by transitioning from source $P$ to target $Q$. It can also be interpreted as the density of $Q$ with respect to $P$ on $\mathcal{X} \times \mathcal{Y}$.

Besides the full densities $p$ and $q$ also the marginal densities $p_X$, $q_X$ of the feature variable $X$ are of interest:

$$p_X(x) = \sum_{y=1}^{\ell} p(x, y), \quad q_X(x) = \sum_{y=1}^{\ell} q(x, y).$$

The feature densities $p_X$, $q_X$ give rise to the *feature importance weight* function $w_X(x)$ for $x \in \mathcal{X}$ which is defined by

$$w_X(x) = \begin{cases} \frac{q_X(x)}{p_X(x)}, & \text{for } p_X(x) > 0, \\ 0, & \text{for } p_X(x) = 0. \end{cases} \tag{2b}$$

**Posterior probabilities.** We denote the *posterior probability* (conditional probability) of class $y \in \mathcal{Y}$ given the feature variable $x$ under the source distribution $P$ by $P[Y = y \,|\, X = x]$. This is a single number. $P[Y = y \,|\, X]$ stands for the random variable created by sampling $x$ from the feature distribution $P_X$ and evaluating $P[Y = y \,|\, X = x]$ at $x$.
$Q[Y = y \,|\, X = x]$ and $Q[Y = y \,|\, X]$ respectively denote the corresponding posterior probabilities under the target distribution $Q$.

Recall also the definition of the *class-conditional feature distributions* $P_{Y=y}$ and $Q_{Y=y}$ under the source distribution $P$ and target distribution $Q$ respectively by

$$\begin{aligned} P_{Y=y}[X \in M] &= P[X \in M \,|\, Y = y] = \frac{P[X \in M, Y = y]}{P[Y = y]}, \\ Q_{Y=y}[X \in M] &= Q[X \in M \,|\, Y = y] = \frac{Q[X \in M, Y = y]}{P[Y = y]}, \end{aligned} \tag{3}$$

for $M \subset \mathcal{X}$.

**Further notation.** In the following, we denote by $\mathbf{C} = (C_1, \ldots, C_\ell)$ hard *multinomial classifiers* in the sense that

$$\begin{aligned} & C_i \subset \mathcal{X} \text{ for all } i = 1, \ldots, \ell, \\ & C_1, \ldots, C_\ell \text{ is a disjoint decomposition of } \mathcal{X}, \text{ and} \\ & Y = y \text{ is predicted when } X \in C_y \text{ is observed.} \end{aligned} \tag{4}$$

The *indicator function* $\mathbf{1}_S$ of a set $S$ is defined as $\mathbf{1}_S(s) = 1$ for $s \in S$ and $\mathbf{1}_S(s) = 0$ for $s \notin S$.

## 3 Types of dataset shift with invariance assumptions

This section formally introduces the types of dataset shift to be discussed in the remainder of the paper.

The dataset shift type denoted here by prior probability shift is also called label shift, target shift, global drift, or named in other ways in the literature. Under this type of shift, the class-conditional feature distributions are invariant between source and target distribution. Its definition is given here mainly as a point of reference.

**Definition 1 (Prior Probability Shift).** *For each $y \in \mathcal{Y}$, the class-conditional feature distributions $P_{Y=y}[X \in M]$ and $Q_{Y=y}[X \in M]$ for measurable $M \subset \mathcal{X}$ as defined by* (3) *are equal, i.e. it holds that*

$$P_{Y=y}[X \in M] = Q_{Y=y}[X \in M], \quad \textit{for } y \in \mathcal{Y},\ M \subset \mathcal{X}.$$

The notion of covariate shift was introduced by Shimodaira (2000). It is based on the possibly most popular invariance assumption for the relationship between source distribution and target distribution: The posterior class probabilities (sometimes called the 'concept') remain unchanged. We quote mutandis mutatis the definition of covariate shift from Kpotufe and Martinet (2021).

**Definition 2 (Covariate Shift).** *For each $y \in \mathcal{Y}$, there exists a measurable function $\eta_y : \mathcal{X} \to [0, 1]$, called* posterior class probability*, such that*

$$P[Y = y \,|\, X = x] = \eta_y(x) = Q[Y = y \,|\, X = x], \tag{5}$$

*almost surely for all $x$ under $P_X$ and under $Q_X$.*

Class distribution estimation in the presence of covariate shift is discussed below in Section 4.

Against the backdrop that, under the assumptions of this paper, it is impossible to distinguish prior probability shift and covariate shift solely on the basis of data, the following notion of factorizable joint shift (FJS) as proposed by He et al. (2021) is very appealing at first glance. For it includes both prior probability shift and covariate shift as special cases and, thus, may be interpreted as interpolating between these two poles of dataset shift.

**Definition 3 (Factorizable joint shift (FJS)).** *There exist non-negative functions $u$ on $\mathcal{X}$ and $v$ on $\mathcal{Y}$ such that for the importance weight function $w$ as defined in* (2a)*, it holds that*

$$w(x, y) = u(x)\, v(y), \tag{6a}$$

*almost surely for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$ under $P$.*

Observe that the functions $u$ and $v$ of Definition 3 are not uniquely determined because for any $c > 0$ the functions $u_c = c\, u$ and $v_c = v/c$ also satisfy (6a):

$$w(x, y) = u_c(x)\, v_c(y). \tag{6b}$$

No invariance property between the source and target distributions is obvious from Definition 3. Such a property, nonetheless, is implied by Theorem 1 below in Section 5 which is devoted to a discussion of FJS.

Chen et al. (2022) proposed “a new distribution shift model, Sparse Joint Shift (SJS), which considers the joint shift of both labels and a few features. This unifies and generalizes existing shift models including label shift and sparse covariate shift[1], where only marginal feature or label distribution shifts are considered.”

**Definition 4 (Sparse Joint Shift (SJS)).** *Let $T : \mathcal{X} \to \mathcal{T}$ be a measurable transformation of the feature values $x$. The source distribution $P$ and the target distribution $Q$ are related through $T$-SJS if it holds for all $y \in \mathcal{Y}$ and $M \subset \mathcal{X}$ that*

$$P_{Y=y}[X \in M \,|\, T(X) = t] = Q_{Y=y}[X \in M \,|\, T(X) = t] \tag{7}$$

*for all $t \in \mathcal{T}$ almost surely under $P_{T(X)}$ and $Q_{T(X)}$.*

Under SJS, the doubly conditioned (by class and by a transformation of the features) feature distributions are invariant between source distribution and target distribution. Note that $T(X)$ in general creates a ‘sparse’ or ‘thinned out’ version of the features. Chen et al. (2022, Section 3.1) called this type of shift ‘sparse’ because “the sparsity is necessary for the shift to be identifiable”.
Choosing $T$ in Definition 4 as $T(x) = c$ for all $x \in \mathcal{X}$, where $c$ is some fixed value, shows that prior probability shift in the sense of Definition 1 is a special case of SJS. In certain limited circumstances, covariate shift implies SJS and vice versa, as is discussed below in Section 6. In general, however, covariate shift is not a special case of SJS.
If $P$ and $Q$ are related through an ‘exponential tilt model’ as defined in Section 3 of Maity et al. (2023) then $P$ and $Q$ are also related through SJS.

## 4 Covariate shift

This section gives a brief overview of class distribution estimation under covariate shift. The topic appears to not have received much attention in the literature, with the exceptions of Card and Smith (2018) and Tasche (2022a).

**Class prior estimators.** If $\mathbf{C} = (C_1, \ldots, C_\ell)$ is a multinomial classifier as defined by (4), *classify & count* (Forman, 2005) might be the most obvious class prior estimator $\widetilde{Q}_n[Y = y]$, $y = 1, \ldots, \ell$, under any type of dataset shift:

$$\widetilde{Q}_n[Y = y] = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}_{C_y}(x_i),$$

where $x_1, \ldots, x_n$ is a test sample of feature values, assumed to have been generated with the target feature distribution $Q_X$. If $x_1, \ldots, x_n$ is an i.i.d. sample from $Q_X$, it follows that $\widetilde{Q}_n[Y = y] \to Q[X \in C_y]$ for $n \to \infty$. However, given that $Q_X$ may be any distribution on $\mathcal{X}$, under covariate shift there is no reason

[1] See Definition 6 below for a definition of sparse covariate shift.

why $Q[X \in C_y]$ should equal $Q[Y = y]$ unless $\mathbf{C}$ is a perfect classifier under the target distribution $Q$ – which is an unrealistic assumption.

As noted by Card and Smith (2018), valid estimates $\widehat{Q}_n[Y = y]$ of the target prior probabilities $Q[Y = y]$, $y = 1, \ldots, \ell$, under covariate shift can be obtained by taking recourse to the law of total probability. The law of total probability implies

$$Q[Y = y] = E_Q\big[P[Y = y \,|\, X]\big] = \int_{\mathcal{X}} P[Y = y \,|\, X = x]\, Q_X(dx). \tag{8a}$$

This gives the estimator

$$\widehat{Q}_n[Y = y] = \frac{1}{n} \sum_{i=1}^{n} \widehat{P}[Y = y \,|\, X = x_i], \tag{8b}$$

where $x_1, \ldots, x_n$ is a test sample of feature values, as described above, and $\widehat{P}[Y = y \,|\, X = x]$ denotes an estimate of the posterior probability $P[Y = y \,|\, X = x]$ under the source distribution $P$, evaluated at the feature value $x$. Estimator (8b) was called *probabilistic classify and count (PCC)* by Card and Smith (2018) and *probability estimation & average (P&A)* by Bella et al. (2010).

With the feature importance weight function $w_X$ defined by (2b), under covariate shift it holds true that

$$Q[Y = y] = E_P[w_X(X)\, \mathbf{1}_{\{y\}}(Y)], \quad y \in \mathcal{Y}. \tag{9a}$$

Hence, once the importance weight function $w_X$ has been estimated from a sample of features generated under $P$ and another sample of features generated under $Q$, the class prior probabilities $Q[Y = y]$ can be estimated by means of the estimator

$$\bar{Q}_m[Y = y] = \frac{1}{m} \sum_{i=1}^{m} w_X(x_i)\, \mathbf{1}_{\{y\}}(y_i), \tag{9b}$$

where $(x_1, y_1), \ldots, (x_m, y_m)$ is an i.i.d. sample of $(X, Y)$ under the source distribution $P$. A variety of methods is available for estimating $w_X$, see e.g. Sugiyama et al. (2012) or Bickel et al. (2009). Card and Smith (2018) might have deployed estimator (9b), calling it *reweighting* estimator. They did not, however, provide an explicit formula for it. A potential application of (9b) would be to make use of it for cross-checking primary estimates of the target prior probabilities resulting from an application of (8b).

**Dimension reduction.** $X$ may be a high dimensional random vector such that precisely estimating $x \mapsto P[Y = y \,|\, X = x]$ is difficult, and also the computation of the high-dimensional integral on the right-hand side of (8a) is a hard task. Hence, is it possible to reduce the dimension of $X$ by applying a transformation $T$ such that $T(X)$ has a lower dimension than $X$ but some version of (8a), e.g. like (10a), still holds true:

$$Q[Y = y] \overset{?}{=} E_Q\big[P[Y = y \,|\, T(X)]\big] = \int_{\mathcal{T}} P[Y = y \,|\, T(X) = t]\, Q_{T(X)}(dt), \tag{10a}$$

supposing that the transformation $T$ takes its values in $\mathcal{T}$.

Tasche (2022a, Theorem 1) showed that

$$P[Y = y \,|\, T(X)] = Q[Y = y \,|\, T(X)] \tag{10b}$$

is true under covariate shift with the same transformation $T(X)$ for all target distributions $Q$ which are absolutely continuous with respect to the fixed source distribution $P$ if and only if

$$P[Y = y \,|\, T(X) = T(x)] = P[Y = y \,|\, X = x], \tag{11}$$

almost surely for all $x$ under $P_X$. (11) means that $T(X)$ is *sufficient* for $X$ with respect to $Y = y$ (see Tasche, 2021, Section 3). In general, requesting sufficiency for $T(X)$ excludes simple approaches to dimension reduction for $X$. Hence, most of the time there is no guarantee that (10b) and consequently also (10a) are applicable.

Although (10b) is not true in general without an assumption of sufficiency, thanks to the generalised Bayes' theorem (Klebaner, 2005, Theorem 10.8) covariate shift can still be shown to imply the following variation of (5) for a fixed target distribution $Q$:

**Proposition 1.** *Suppose that $Q$ is absolutely continuous with respect to $P$ and $Q$ and $P$ are related through covariate shift in the sense of Definition 2. Then it follows for any measurable transformation $T : \mathcal{X} \to \mathcal{T}$ and all $y \in \mathcal{Y}$ that*

$$Q[Y = y \,|\, T(X) = t] = \frac{E_P\big[w_X(X)\, \mathbf{1}_{\{y\}}(Y) \,|\, T(X) = t\big]}{E_P[w_X(X) \,|\, T(X) = t]},$$

*for all $t \in \mathcal{T}$ almost surely under $P_{T(X)}$, where $w_X$ is defined as in (2b).*

As a consequence of Proposition 1, (10b) holds true for fixed $Q$ if and only if

$$E_P\big[w_X(X)\, \mathbf{1}_{\{y\}}(Y) \,|\, T(X)\big] = E_P[w_X(X) \,|\, T(X)]\, P[Y = y \,|\, T(X)], \tag{12}$$

i.e. if $w_X(X)$ and $\{Y = y\}$ are independent conditional on $T(X)$ under $P$. Such conditional independence, in particular, follows if $T(X)$ is sufficient for $X$ with respect to $\{Y = y\}$. Accordingly, in principle it is possible to check by means of verification of (12) whether or not (10a) can be applied. This involves the estimation of $w_X$ which, at first glance, might not be much easier or even harder than estimating $P[Y = y \,|\, X]$.

See, however, Stojanov et al. (2019, Section 3) for a method to identify a transformation $T$ such that $T(X)$ is approximately sufficient for $X$ with respect to all $\{Y = y\}$, $y \in \mathcal{Y}$. By (12), then (10b) holds for the target distribution $Q$ in question such that (10a) is applicable.

## 5 Factorizable joint shift (FJS)

He et al. (2021) characterised FJS by claiming that "the biases coming from the data and the label are statistically independent", without specifying any detail of the claim in technical terms. Tasche (2022b) suggested that FJS might be interpreted as a structural property similar to the 'separation of variables' which plays an important role for finding closed-form solutions to differential equations.

As noted by He et al. (2021), covariate shift is a special case of FJS because of

$$w(x, y) = w_X(x) \tag{13a}$$

for $w_X$ defined by (2b), and prior probability shift is a special case of FJS because of

$$w(x, y) = \frac{Q[Y = y]}{P[Y = y]}. \tag{13b}$$

**Characterising FJS.** He et al. (2021) also noted that FJS is not fully identifiable in the unsupervised setting of this paper, i.e. if no labels are observed in the target dataset. In the remainder of this section, we summarise the analysis of FJS performed by Tasche (2022b) and clarify the additional assumptions needed to achieve identifiability for FJS.

The following theorem implies, among other things, an invariance property between source distribution $P$ and target distribution $Q$ thanks to FJS (see Eq. (15) below).

**Theorem 1.** *Suppose that the source distribution $P$ and the target distribution $Q$ are related by FJS in the sense of Definition 3. Denote by $w_X$ the feature importance weight function defined by (2b) and let $q_i = Q[Y = i]$ and $p_i = P[Y = i]$, $i = 1, \ldots, \ell$.*
*Then, up to a constant factor $c$ as in (6b), it follows that*

$$v(y) = \sum_{i=1}^{\ell-1} \varrho_i \,\frac{q_i}{p_i}\, \mathbf{1}_{\{i\}}(y) + \frac{q_\ell}{p_\ell}\, \mathbf{1}_{\{\ell\}}(y) \quad \textit{and} \tag{14a}$$

$$u(x) = \frac{w_X(x)}{\sum_{i=1}^{\ell-1} \varrho_i \,\frac{q_i}{p_i}\, P[Y = i \,|\, X = x] + \frac{q_\ell}{p_\ell}\, P[Y = \ell \,|\, X = x]}, \tag{14b}$$

*where the constants $\varrho_1, \ldots, \varrho_{\ell-1}$ are positive and finite and satisfy the following equation system (with $j = 1, \ldots, \ell - 1$):*

$$p_j = \varrho_j \, E_P\left[\frac{w_X(X)\, P[Y = j \,|\, X]}{\sum_{i=1}^{\ell-1} \varrho_i \,\frac{q_i}{p_i}\, P[Y = i \,|\, X] + \frac{q_\ell}{p_\ell}\, P[Y = \ell \,|\, X]}\right]. \tag{14c}$$

*Conversely, suppose that for the source distribution $P$ a function $w_X : \mathcal{X} \to [0, \infty)$ with $E_P\big[w_X(X)\big] = 1$ and $(q_i)_{i=1,\ldots,\ell} \in (0, 1)^\ell$ with $\sum_{i=1}^{\ell} q_i = 1$ are given. Assume also that $\varrho_1 > 0, \ldots, \varrho_{\ell-1} > 0$ are solutions of the equation system (14c) and $u$ and $v$ are defined by (14b) and (14a), respectively. Then $w(x, y) = u(x)\, v(y)$ has the property that $w(x, y)\, p(x, y)$ is the density of a probability measure $Q$ on $\mathcal{X} \times \mathcal{Y}$ such that $w_X(x)\, p_X(x)$ is the marginal density of the feature variable $X$ under $Q$ and $Q[Y = i] = q_i$ holds for $i = 1, \ldots, \ell$.*

See Tasche (2022b, Theorem 2) for a proof of Theorem 1. The theorem characterises FJS through equations (14b), (14a) and (14c) but does not provide any information regarding the existence or uniqueness of solutions to (14c). A result on existence and uniqueness of the solutions to (14c) was proven for the binary case $\ell = 2$ by Tasche (2022b, Proposition 2).

It can be shown (Tasche, 2022b, Corollary 4) that Theorem 1 implies the following version of the correction formula for class posterior probabilities of Saerens et al. (2001, Eq. (2.4)) and Elkan (2001, Theorem 2) under FJS.

**Corollary 1.** *Suppose that the source distribution $P$ and the target distribution $Q$ are related through FJS in the sense of Definition 3. Then the target posterior probabilities $Q[Y = j \,|\, X = x]$, $j = 1, \ldots, \ell$, can be represented almost surely for all $x$ under $Q_X$ as functions of the source posterior probabilities $P[Y = j \,|\, X = x]$, $j = 1, \ldots, \ell$, in the following way:*

$$Q[Y = j \,|\, X = x] = \frac{\varrho_j \, \frac{Q[Y=j]}{P[Y=j]} P[Y = j \,|\, X = x]}{\sum_{i=1}^{\ell-1} \varrho_i \, \frac{Q[Y=i]}{P[Y=i]} P[Y = i \,|\, X = x] + \frac{Q[Y=\ell]}{P[Y=\ell]} P[Y = \ell \,|\, X = x]},$$
$$j = 1, \ldots, \ell - 1,$$
$$Q[Y = \ell \,|\, X = x] = \frac{\frac{Q[Y=\ell]}{P[Y=\ell]} P[Y = \ell \,|\, X = x]}{\sum_{i=1}^{\ell-1} \varrho_i \, \frac{Q[Y=i]}{P[Y=i]} P[Y = i \,|\, X = x] + \frac{Q[Y=\ell]}{P[Y=\ell]} P[Y = \ell \,|\, X = x]},$$

*where the positive constants $\varrho_1, \ldots, \varrho_{\ell-1}$ satisfy the equation system (14c).*

Corollary 1 in turn implies that under FJS the following invariance property holds true:

$$\frac{Q[Y = j \,|\, X]}{Q[Y = \ell \,|\, X]} \frac{Q[Y = \ell]}{Q[Y = j]} = \varrho_j \frac{P[Y = j \,|\, X]}{P[Y = \ell \,|\, X]} \frac{P[Y = \ell]}{P[Y = j]}, \quad j = 1, \ldots, \ell - 1, \qquad (15)$$

where the constants $\varrho_j$ satisfy the equation system (14c). Eq. (15) may be interpreted as stating that under factorizable joint shift the ratios of the class-conditional feature densities are invariant between source and target distributions up to a constant factor (see Tasche, 2022b, Remark 1).

**Class distribution estimation under FJS.** Theorem 1 suggests two obvious ways to learn the characteristics of factorizable joint shift:

a) If the target prior class probabilities $Q[Y = i] = q_i$ are known (for instance from external sources) solve (14c) for the constants $\varrho_i$.
b) If the target prior class probabilities $Q[Y = i] = q_i$ are unknown (as would be the case for the problem of class distribution estimation), fix values for the constants $\varrho_i$ and solve (14c) for the $q_i$. Letting $\varrho_i = 1$ for all $i$ is a natural choice that converts (14c) into the system of maximum likelihood equations for the $q_i$ under the prior probability shift assumption.

See Section 4.2.4 of Tasche (2013) for an example of approach a) from the area of credit risk. Whenever for a given marginal target feature distribution $Q_X$ there is more than one set of potential target class prior probabilities $q_y$, $y = 1, \ldots, \ell$, such that (14c) can be solved for the $\varrho_i$, then a case of unidentifiability of the joint target distribution $Q$ under FJS is incurred. This always holds for the binary case $\ell = 2$ because for any given combination of joint source distribution $P$, target feature distribution $Q_X$ and target prior probability $q_1 = Q[Y = 1]$, a constant $\varrho_1$ can be found such that $P$ and $Q$ are related through FJS (Tasche, 2022b, Proposition 2).

Regarding the interpretation of (14c) in approach b) as maximum likelihood equations, see Du Plessis and Sugiyama (2014). This interpretation, in particular, implies that an EM (expectation maximisation) algorithm can be deployed for solving the equation system (Saerens et al., 2001) in the case $1 = \varrho_1 = \ldots = \varrho_{\ell-1}$.

## 6 Sparse joint shift (SJS)

Definition 4 of SJS slightly generalises Definition 1 of Chen et al. (2022) as can be seen by choosing $T$ as extractor of a subset of the components of the feature vector. The equivalence of this special case of Definition 4 and the definition of Chen et al. (2022) then follows from Proposition 3.8 of Tasche (2023).

Observe that by the generalised Bayes' theorem (Klebaner, 2005, Theorem 10.8), (7) can equivalently be stated as

$$\frac{P[X \in M, Y = y \,|\, T(X) = t]}{P[Y = y \,|\, T(X) = t]} = \frac{Q[X \in M, Y = y \,|\, T(X) = t]}{Q[Y = y \,|\, T(X) = t]}. \qquad (16)$$

The following properties of SJS were first noted by Tasche (2023).

**Proposition 2 (Properties of SJS).** *Suppose that the source distribution $P$ and the target distribution $Q$ are related through $T$-SJS in the sense of Definition 4. Then the following two statements hold true:*

*(i) If $T' : \mathcal{X} \to \mathcal{T}'$ and $S : \mathcal{T}' \to \mathcal{T}$ are measurable transformations such that for all $x \in \mathcal{X}$ it holds that $T(x) = (S \circ T')(x) = S\big(T'(x)\big)$, then $P$ and $Q$ are also related through $T'$-SJS.*

*(ii) For all $i \in \mathcal{Y}$, it holds that*

$$Q[Y = i \,|\, X = x] = \frac{\frac{Q[Y=i \,|\, T(X)=T(x)]}{P[Y=i \,|\, T(X)=T(x)]}\, P[Y = i \,|\, X = x]}{\sum_{j=1}^{\ell} \frac{Q[Y=j \,|\, T(X)=T(x)]}{P[Y=j \,|\, T(X)=T(x)]}\, P[Y = j \,|\, X = x]},$$

*for all $x \in \mathcal{X}$ almost surely under $Q_X$.*

See Tasche (2023, Corollary 4.3) for a proof of Proposition 2 (i) and Tasche (2023, Proposition 4.5) for a proof of Proposition 2 (ii). By Proposition 2 (i), prior probability shift implies $T$-SJS for any transformation $T : \mathcal{X} \to \mathcal{T}$. Proposition 2 (ii) is another generalisation of the posterior correction formula of Saerens et al. (2001, Eq. (2.4)) and Elkan (2001, Theorem 2), this time under the assumption of SJS.

The next result rephrases the identifiability result of (Chen et al., 2022, Theorem 1) in terms of conditional expectations instead of joint densities.

**Theorem 2 (Identifiability under SJS).** *Suppose that there are distributions $P$, $Q$ and $Q'$ on $\mathcal{X} \times \mathcal{Y}$ as well as transformations $T : \mathcal{X} \to \mathcal{T}$ and $T' : \mathcal{X} \to \mathcal{T}'$ such that $P$ and $Q$ are related through $T$-SJS and $P$ and $Q'$ are related through $T'$-SJS. For given measurable functions $f_i : \mathcal{X} \to [0, \infty)$, $i = 1, \ldots, \ell$, define the random matrix $R(X) = \big(R_{ij}(X)\big)_{i,j\in\{1,\ldots,\ell\}}$ by*

$$R_{ij}(X) = \frac{E_P\big[f_i(X)\,\mathbf{1}_{\{j\}}(Y)\,|\,(T(X), T'(X))\big]}{P\big[Y = j\,|\,(T(X), T'(X))\big]}.$$

*If $Q_X = Q'_X$ and $P\big[\operatorname{rank}\big(R(X)\big) = \ell\big] = 1$ is true, then it follows that $Q[Y = y,\, X \in M] = Q'[Y = y,\, X \in M]$ for all $y \in \mathcal{Y}$ and measurable $M \subset \mathcal{X}$.*

See Tasche (2023, Theorem 4.7) for a proof of Theorem 2. The rank condition of Theorem 2 is likely to be satisfied for instance if $f_i(X) = \mathbf{1}_{C_i}(X)$ for some reasonably accurate classifier $\mathbf{C} = (C_1, \ldots, C_\ell)$ as in (4). Hence identifiability of SJS ought to be given most of the time.

**SJS and covariate shift.** As seen above, prior probability shift is not only a special case of SJS but also implies $T$-SJS for any transformation $T$ of the features. In contrast, examples by Chen et al. (2022) and Tasche (2023) show that covariate shift and SJS are unrelated properties in the sense that they do not imply one another but do not exclude each other either.

For a full understanding of the relationship of covariate shift and SJS, we introduce two further types of dataset shift. The first of these was proposed by Tasche (2023, Definition 4.11).

**Definition 5 (Conditional distribution invariance (CDI)).** *Let $T : \mathcal{X} \to \mathcal{T}$ be a measurable transformation of the feature variable $X$. The source distribution $P$ and the target distribution $Q$ are related through* $T$-CDI *if it holds for all $M \subset \mathcal{X}$ that*

$$P[X \in M\,|\,T(X) = t] = Q[X \in M\,|\,T(X) = t] \tag{17}$$

*for all $t \in \mathcal{T}$ almost surely under $P_{T(X)}$ and $Q_{T(X)}$.*

The property CDI is interesting because in principle its presence can be evidenced by comparing statistics estimated from the feature observations in the training and test datasets. No label observations are needed. Moreover, in the presence of CDI, there is basically no difference between covariate shift and SJS, as we will see below.

The following additional type of dataset shift was introduced by Chen et al. (2022, Definition 3).

**Definition 6 (Sparse Covariate Shift (SCS)).** *Let $T : \mathcal{X} \to \mathcal{T}$ be a measurable transformation of the feature variable $X$. The source distribution $P$ and the target distribution $Q$ are related through* $T$-SCS *if it holds for all $y \in \mathcal{Y}$ and $M \subset \mathcal{X}$ that*

$$P[X \in M, Y = y\,|\,T(X) = t] = Q[X \in M, Y = y\,|\,T(X) = t] \tag{18}$$

*for all $t \in \mathcal{T}$ almost surely under $P_{T(X)}$ and $Q_{T(X)}$.*

The following theorem describes the interplay of SJS and covariate shift in the presence of CDI.

**Theorem 3.** *Let $T : \mathcal{X} \to \mathcal{T}$ be a measurable transformation of the feature variable $X$. Suppose that a source distribution $P$ and a target distribution $Q$ on $\mathcal{X} \times \mathcal{Y}$ are given. Then the following three statements hold true:*

*(i) If $P$ and $Q$ are related through both $T$-CDI in the sense of Definition 5 and covariate shift in the sense of Definition 2, then $P$ and $Q$ are also related through $T$-SCS in the sense of Definition 6.*

*(ii) If $P$ and $Q$ are related through $T$-SCS, they are also related through both $T$-SJS and $T$-CDI.*

*(iii) For given measurable functions $f_i : \mathcal{X} \to [0, \infty)$, $i = 1, \ldots, \ell$, define the random matrix $R(X) = \big(R_{ij}(X)\big)_{i,j\in\{1,\ldots,\ell\}}$ by*

$$R_{ij}(X) = \frac{E_p\big[f_i(X)\,\mathbf{1}_{\{j\}}(Y)\,|\,T(X)\big]}{P\big[Y = j\,|\,T(X)\big]}.$$

*Suppose that $P\big[\mathrm{rank}\big(R(X)\big) = \ell\big] = 1$ holds true. Then, if $P$ and $Q$ are related through both $T$-SJS and $T$-CDI, they are also related through covariate shift.*

For the derivation of Theorem 3, see Theorem 4.16 and Remark 4.18 of Tasche (2023). Somewhat oversimplifying, we might summarise Theorem 3 with the following 'equation': $SCS = covariate\ shift \cap CDI = SJS \cap CDI.$

**Class distribution estimation under SJS.** Chen et al. (2022) proposed two methods for estimating SJS: SEES-c for the case of continuous features and SEES-d for the case of discrete features (SEES = "shift estimation and explanation under SJS"). In this paper, we briefly describe only an important special case of SEES-d (Tasche, 2023, Eq. (C.6)) because the results presented by Chen et al. (2022) appear to suggest that SEES-d is more efficient than SEES-c. By sufficiently fine discretisation of the feature space, SEES-d can also be applied to continuous or mixed continuous and discrete feature settings.

**Proposition 3 (Conditional confusion matrix approach).** *Let $T : \mathcal{X} \to \mathcal{T}$ be a measurable and discrete transformation of the feature variable $X$, i.e. with range $\mathcal{T} = \{t_1, \ldots, t_N\}$. Suppose that the source distribution $P$ and a target distribution $Q$ are related through $T$-SJS in the sense of Definition 4 and that $\mathbf{C} = (C_1, \ldots, C_\ell)$ is a classifier as in (4). Then for each $t \in \mathcal{T}$, the target posterior probabilities $q_{y,t} = Q[Y = y\,|\,T(X) = t]$, $y \in \mathcal{Y}$, satisfy the linear equation system (with $j = 1, \ldots, \ell$)*

$$\sum_{y=1}^{\ell} q_{y,t}\, P[X \in C_j\,|\,Y = y, T(X) = t] = Q[X \in C_j\,|\,T(X) = t]. \tag{19a}$$

Once the $q_{y,t}$, $y \in \mathcal{Y}, t \in \mathcal{T}$, have been determined, by the law of total probability the target class prior probabilities $Q[Y = y]$ can be calculated via

$$Q[Y = y] = \sum_{i=1}^{N} q_{y,t_i}\, Q[T(X) = t_i]. \tag{19b}$$

Therefore, Proposition 3 provides a solution to the class distribution estimation problem under an assumption of SJS, thereby generalising the confusion matrix approach as described by Saerens et al. (2001, Section 2.3.1). In particular, Proposition 3 could be deployed to check assumptions of prior probability shift. By Proposition 2 (i), prior probability shift implies $T$-SJS for any transformation $T$. Hence, in principle, results under prior probability shift by any suitable method of class distribution estimation must coincide with the results obtained by combining (19a) and (19b), for any choice of $T$ taking discrete values.

In practice, develop the classifier on the full training dataset. Then stratify both training dataset and test dataset by $T$ applied to the feature (or covariate) variable $X$. After that, treat each of the resulting sub-samples with the confusion matrix approach as in Saerens et al. (2001, Section 2.3.1) to estimate for each $t \in \mathcal{T}$ the posterior probabilities $Q[Y = y \,|\, T(X) = t] = q_{y,t}$, $y \in \mathcal{Y}$. Combine the posterior probabilities by means of (19b) to obtain estimates of the target prior probabilities $Q[Y = y]$, $y \in \mathcal{Y}$.

Examples for possible choices of the transformation $T$ of Proposition 3 might be found in medical applications: It is plausible that the sensitivity and specificity of a test for an infection change between training and test datasets but that they are preserved within the strata when there is stratification by age group and gender. This would mean that the dataset shift can be described by $T$-sparse joint shift with $T$ being the transformation that provides the age group and the gender of an instance (patient).

## 7 Conclusions

This paper provides analyses of invariance assumptions for distribution (dataset) shift, with focus on their suitability for designing class distribution estimators. Covariate shift, factorizable joint shift, and sparse joint shift are studied in some detail. Both the 'covariate' and the 'sparse joint' types of shift are found fit for designing class distribution estimators. In contrast, factorizable joint shift is found unsuitable due to lack of identifiability unless additional constraints are applied.

Sparse joint shift (SJS) is particularly appealing for the fact that it generalises prior probability shift (label shift) and, therefore, has the potential to provide meaningful estimates even in contexts where an assumption of prior probability shift is found untenable. An open research problem is how to identify feature transformations that entail SJS if they cannot be identified by theoretical considerations. Chen et al. (2022, Section 4.1) suggested two brute-force approaches but these approaches have issues which might make their application questionable (Tasche, 2023, Section 5).

**Acknowledgement** The author would like to thank three anonymous reviewers for their useful comments and suggestions.

## References

A. Bella, C. Ferri, J. Hernandez-Orallo, and M.J. Ramírez-Quintana. Quantification via probability estimators. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*, pages 737–742. IEEE, 2010.

S. Bickel, M. Brückner, and T. Scheffer. Discriminative Learning Under Covariate Shift. *The Journal of Machine Learning Research*, 10:2137–2155, 2009.

D. Card and N.A. Smith. The Importance of Calibration for Estimating Proportions from Annotations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1636–1646, 2018. https://doi.org/10.18653/v1/N18-1148.

L. Chen, M. Zaharia, and J.Y. Zou. Estimating and Explaining Model Performance When Both Covariates and Labels Shift. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems, NeurIPS 2022*, volume 35, pages 11467–11479. Curran Associates, Inc., 2022.

M.C. Du Plessis and M. Sugiyama. Semi-supervised learning of class balance under class-prior change by distribution matching. *Neural Networks*, 50:110–119, 2014.

C. Elkan. The foundations of cost-sensitive learning. In B. Nebel, editor, *Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001*, pages 973–978. Morgan Kaufmann, 2001.

A. Esuli, A. Fabris, A. Moreo, and F. Sebastiani. *Learning to Quantify*. Springer Cham, 2023. https://doi.org/https://doi.org/10.1007/978-3-031-20467-8.

T. Fawcett and P.A. Flach. A response to Webb and Ting's On the Application of ROC Analysis to Predict Classification Performance under Varying Class Distributions. *Machine Learning*, 58(1):33–38, 2005.

G. Forman. Counting Positives Accurately Despite Inaccurate Classification. In *European Conference on Machine Learning (ECML 2005)*, pages 564–575. Springer, 2005.

J.J. Gart and A.A. Buck. Comparison of a screening test and a reference test in epidemiologic studies. II. A probabilistic model for the comparison of diagnostic tests. *American Journal of Epidemiology*, 83(3):593–602, 1966.

P. González, A. Castaño, N.V. Chawla, and J.J. Del Coz. A Review on Quantification Learning. *ACM Comput. Surv.*, 50(5):74:1–74:40, 2017.

H. He, Y. Yang, and H. Wang. Domain Adaptation with Factorizable Joint Shift. Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning, 2021.

V. Hofer. Adapting a classification rule to local and global shift when only unlabelled data are available. *European Journal of Operational Research*, 243(1):177–189, 2015.

M. Kirchmeyer, A. Rakotomamonjy, E. de Bezenac, and P. Gallinari. Mapping conditional distributions for domain adaptation under generalized target shift, 2021. URL https://arxiv.org/abs/2110.15057. Presented at ICLR 2022.

F.C. Klebaner. *Introduction to Stochastic Calculus with Applications*. Imperial College Press, second edition, 2005.

A. Klenke. *Probability Theory: A Comprehensive Course*. Springer Science & Business Media, 2013.

S. Kpotufe and G. Martinet. Marginal singularity and the benefits of labels in covariate-shift. *The Annals of Statistics*, 49(6):3299–3323, 2021. https://doi.org/10.1214/21-AOS2084.

G. Krempl, D. Lang, and V. Hofer. Temporal density extrapolation using a dynamic basis approach. *Data mining and knowledge discovery*, 33:1323–1356, 2019.

S. Maity, M. Yurochkin, M. Banerjee, and Y. Sun. Understanding new tasks through the lens of training data via exponential tilting. In *The Eleventh International Conference on Learning Representations (ICLR 2023)*, 2023. URL https://openreview.net/forum?id=DBMttEEoLbw.

M. Saerens, P. Latinne, and C. Decaestecker. Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. *Neural Computation*, 14(1):21–41, 2001.

C. Scott. A Generalized Neyman-Pearson Criterion for Optimal Domain Adaptation. In *Proceedings of Machine Learning Research, 30th International Conference on Algorithmic Learning Theory*, volume 98, pages 1–24, 2019.

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

P. Stojanov, M. Gong, J. Carbonell, and K. Zhang. Low-Dimensional Density Ratio Estimation for Covariate Shift Correction. In K. Chaudhuri and M. Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 3449–3458. PMLR, 2019.

M. Sugiyama, T. Suzuki, and T. Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

D. Tasche. The art of probability-of-default curve calibration. *Journal of Credit Risk*, 9(4):63–103, 2013. https://doi.org/10.21314/JCR.2013.169.

D. Tasche. Calibrating sufficiently. *Statistics*, 55(6):1356–1386, 2021. https://doi.org/10.1080/02331888.2021.2016767.

D. Tasche. Class Prior Estimation under Covariate Shift: No Problem? Working paper, presented at ECML/PKDD 2022 workshop Learning to Quantify: Methods and Applications (LQ 2022), 2022a.

D. Tasche. Factorizable Joint Shift in Multinomial Classification. *Machine Learning and Knowledge Extraction*, 4(3):779–802, 2022b. https://doi.org/10.3390/make4030038.

D. Tasche. Sparse joint shift in multinomial classification. *arXiv preprint arXiv:2303.16971*, 2023.

K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang. Domain Adaptation Under Target and Conditional Shift. In *Proceedings of the 30th International Conference on International Conference on Machine Learning – Volume 28*, ICML'13, pages III–819–III–827. JMLR.org, 2013.