A System Framework for Personalized and Transparent Data-Driven Decisions

Sarah Oppold(✉) and Melanie Herschel

University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany
{Sarah.Oppold,Melanie.Herschel}@ipvs.uni-stuttgart.de

Abstract. Decision support systems that rely on data analytics are used in numerous applications. Their advantages are indisputable, however, they also present risks, possibly having severe impact on people's lives. Consequently, the need to support ethical or responsible behavior of such systems has recently emerged, putting an emphasis on ensuring fairness, transparency, accountability, etc. This paper presents a novel system framework that offers *transparent* and *personalized* services tailored to user profiles to serve their best interest. Our framework personalizes the choice of model for individuals or groups of users based on metadata about data sets and machine learning models. Querying and processing these metadata ensures transparency by supporting various kinds of queries by different stakeholders. We discuss our framework in detail, show why existing solutions are inadequate, and highlight research questions that need to be tackled in the future. Based on a prototypical implementation, we showcase that even a baseline implementation of our framework supports the desired transparency and personalization.

Keywords: Data analytics · Metadata · Model ensembles

1 Introduction

Data-driven *decision support systems (DSS)* are used in numerous applications today. They leverage algorithms and models of varying complexity that provide humans with predictions helpful in making decisions. While the advantages of such DSS are indisputable, they also present risks, when (inadvertently) misused or misconfigured, possibly causing severe consequences on people's lives. This has become a subject of public interest, e.g., when media divulged news on biased services. Examples include ad services that displayed ads implying a criminal record more frequently when searching for black-sounding names [24] or a system used to filter job applications that desk-rejected applicants solely because of their sex or racial origin [18]. Our vision is to provide information management solutions that facilitate the identification and avoidance of such prejudiced DSS, thereby enabling responsible DSS behavior. This paper presents a first DSS architecture with built-in capabilities that pave the way towards offering fair, transparent, and accountable services.

S. Dustdar et al. (Eds.): CAiSE 2020, LNCS 12127, pp. 153–168, 2020.
https://doi.org/10.1007/978-3-030-49435-3_10

Recent research, e.g., on bias in machine learning, shows that there are pitfalls everywhere in the development process of a DSS that can distort the generated predictions [2]. We believe that, despite the efforts put into developing fair [5,10], transparent [4,22], and accountable [15,23] machine learning models, we will still rely on imperfect models in many applications, including DSS. This situation reminds us of the field of medicine, where the use of drugs for the treatment of diseases poses a similar problem. For drugs, society has accepted that no perfect, side-effect free, and success-guaranteeing drug may exist. Still, they are used after they were responsibly tested and improved before going to market and transparently communicate risks and side-effects in their notice.

Analogously, we advocate that a significant step towards our vision of responsible systems would already be to devise systems that allow us to provably get as responsible as we can practically get. Two key features on this path are *personalization* and *transparency*. Here, personalization is a prerequisite for fair, best-effort, and context-aware solutions, as it allows to adapt the choice of model to a user (group) profile with some pre-defined quality criteria. This is orthogonal to research on improving models in isolation, as it takes user profiles and different optimization goals into account. Transparency, to be provided in a holistic way for the DSS, allows to improve accountability and understandability. This notion of transparency goes beyond explainable AI [22], which mostly focuses on explaining how a decision was made by a model. This ignores for instance the process of gathering training data or information on why a specific model was applied, which are integrated in our holistic notion of transparency for DSS.

This paper presents a system framework that, to the best of our knowledge, is the first holistic approach to personalized and transparent DSS. The framework optimizes the choice of model for subgroups of the population or individual persons, relying on metadata collected about data, models, and their creation processes. Optimization goals are defined based on both "classical" quality criteria such as accuracy and criteria relating to responsibility, e.g., fairness. To achieve transparency, the framework captures information to explain both choices made and results at different scales to different stakeholders in a user-friendly way. Overall, this paper contributes:

- A system framework for data-driven decision support, centered around the non-functional requirements of transparency and personalization, taking quality and usability into account. Achieving these requirements relies on several types of metadata, user profiles, and quality specifications to be met by models (Sect. 2).
- A detailed discussion on how to implement the framework components, where we describe which existing research we can build on and identify open issues that require new methods (Sect. 3).
- Using a first prototypical implementation of our framework, named LuPe [20], we showcase that even a baseline implementation of our framework supports the desired non-functional requirements of transparency and personalization on sample use cases (Sect. 4).

2 System Framework Overview

Our framework for personalized and transparent DSS considers different stakeholders (Sect. 2.1). Section 2.2 introduces and illustrates the framework's non-functional requirements. Section 2.3 highlights the framework components.

2.1 Stakeholders

A *service provider* is the legal entity that provides a specific service that incorporates a DSS. It typically employs or commissions *developers* who are responsible for the implementation and maintenance of the DSS. For the purpose of the discussion, we focus on developer tasks relating to defining models used for decision support. A *user* is an individual person that uses the service of the service provider. Finally, a *regulator* is an entity that aims at verifying the compliance of the service offered by the service provider with respect to a set of regulations. These can be internal regulations subject to audits as well as legal regulations.

2.2 Non-functional Requirements

Our framework considers the non-functional requirements of quality, personalization, transparency, and usability. We discuss and illustrate these using the example of a company *BestJob* that offers an online job market. *BestJob* applies a supervised learning model that recommends the most suitable job offers for each job seeking user, given their user profile.

Quality. We consider various quality metrics, including quality metrics to quantify model performance (e.g., accuracy) and metrics relating to fairness (e.g., bias). Given a set of quality metrics and model candidates for a specific task, the system shall determine an optimal model with respect to the quality metrics or determine a model guaranteeing certain quality bounds.

As a simple example, consider *BestJob* that aims at achieving high quality results in terms of recall while the system also has to comply with different regulations. Indeed, *BestJob* has to obey laws stating job recommendation systems must not make discriminating recommendations based on, e.g., gender. Among all available models for the task of job recommendation, suppose that one model M_1 obtains highest recall but is known to be gender-biased due to the training data under-representing women. Another model M_2 presents less gender-bias (trained on other data) but has slightly less (but still high) accuracy. Then, the latter model would be selected given the stated global optimization goals.

Personalization. For different individual users or user groups, the quality of a given model may vary. Thus, the system shall adapt its choice of used model depending on a user (group) profile and quality requirements. For example, reconsidering the models M_1 and M_2 previously mentioned, our framework allows to apply M_1 to male users while M_2 is applied for female users.

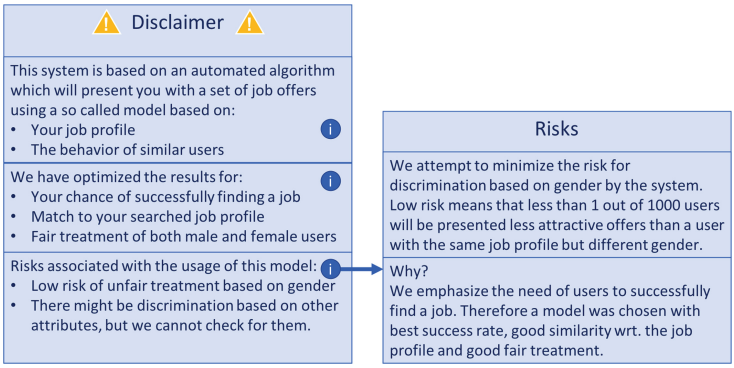


Fig. 1. Sample disclaimer making model behavior and risks transparent

Transparency. The system shall offer capabilities that easily allow different stakeholders to see what actions are performed based on which quality criteria and user profiles. In particular, the system shall be able to account for the definition and use of a model.

For instance, querying and properly processing metadata collected by our framework allows *BestJob* to communicate information that makes risks associated with their recommendations transparent. An example of a disclaimer presented to users is given in Fig. 1. It provides an overview of the data used for the recommendations, the optimization criteria the recommendations were based on, and the risks the user may be exposed to due to uncertainties and probabilities inherent in the underlying machine learning models.

Usability. The system shall offer tools and interfaces that allow different stakeholders to easily use the system.

Continuing the example based on Fig. 1, the initial interface provides a concise overview and allows a user to look into the details. The figure shows details for the risk assessment, focusing on gender discrimination. These details explain to users what risk of gender discrimination they are exposed to despite the effort to avoid it using an easy to understand "1 out of 1000" metaphor.

Remarks. While the above list is not an exhaustive list of desirable non-functional requirements for responsible DSS in general, we believe that the selected requirements are a reasonable first step in exploring solutions for responsible DSS. They are defined quite broadly to cover different aspects (including, for instance, bias as one quality dimension or diverse usability objectives) and form the foundation for further "derivable" non-functional requirements (e.g., personalization towards fairness, transparency for accountability).

Despite this limited set of non-functional requirements, many interesting research challenges lie ahead, especially given that the requirements are not independent from one another. For instance, (i) disclosing all information about the development process maximizes transparency, possibly at the cost of usability;

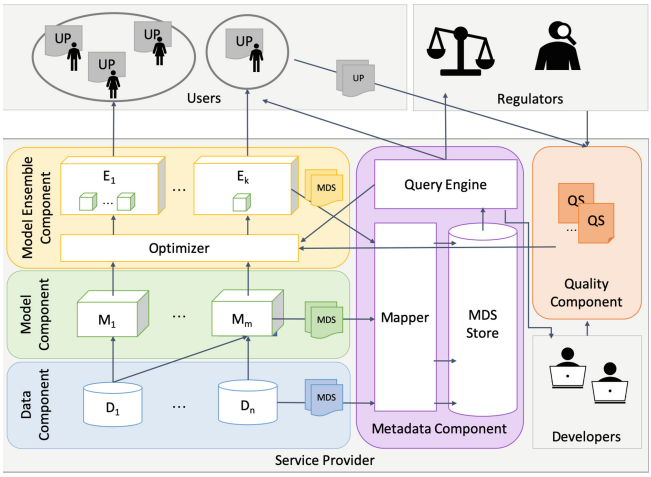


Fig. 2. System framework overview. (Color figure online)

(ii) global quality optimization may counter (local) personalization; (iii) multi-dimensional quality specification may reduce usability by making the system more difficult to develop. Furthermore, personalization coupled with transparency may improve the overall quality, e.g., fairness, as unfair practices of a service provider would be more easily revealed.

2.3 Framework Component Overview and Interplay

We now give an overview of the framework's components (colored boxes in Fig. 2) and their interplay.

The bottom layer (blue) is the *Data Component*. It manages all data sets $D_1 \dots D_n$ relevant for models $M_1 \dots M_m$. These models are managed by the *Model Component* (green). Each M_i managed here is assumed to handle a common task T, providing decision support within some service offered by a service provider. We understand decision support tasks very broadly here, covering machine-learning based decision making, recommendations, etc. Apart from the fact that they share T, the models may differ in several ways, e.g., in the data they are trained on or in the machine learning technique used (ranging from simple supervised learning to deep learning). Therefore, both the Data and the Model Component generate so-called *Metadata Sheets (MDS)* where they store generated or collected metadata. These MDS contain metadata on quality criteria, provenance, descriptions, as well as on legal and ethical aspects. To ensure unified processing across the various system components, the MDS are stored in the *Metadata Component* (purple). The MDS received from different components are mapped to a unified format by the *Mapper* before the data is stored in the *MDS Store*. The metadata stored is then queried and processed to answer

queries issued by different stakeholders or other components. In particular, metadata processing enables transparency. Personalization is achieved by the *Model Ensemble Component* (yellow), which selects the optimal model with respect to the task T for individuals or groups of users who are characterized by the same *user profile (UP)*. The *Optimizer* creates multiple ensembles $E_1, \ldots, E_k$ which are intended for the different users or groups of users. The ensembles are created by selecting or combining the models available in the *Model Component*. Optimization goals or constraints for the different settings are specified in *Quality Specifications (QS)*. These represent the desiderata of different stakeholders on the performance quality of the service, and are managed by the *Quality Component* (orange).

To further describe the interplay of components and their relationships to non-functional requirements, let us discuss the components' purpose during model development and model application in a productive system, respectively.

When developers devise appropriate models for a given task, both the data component and the model component, together with their MDS, are relevant. As model appropriateness is guided by the quality component, it is equally relevant during the development stage. Overall, during model development, transparency offered via usable interfaces supports developers in testing and refining their models to best match predefined quality criteria. Personalization is only indirectly considered, as developers have to ensure that available data, models, and quality criteria are rich and diverse enough to support personalization.

An example of framework operation during the model application stage has been discussed in Sect. 2.2. Clearly, at this stage, transparency is ensured by the metadata component and targets primarily users and regulators, by offering them appropriate access and interfaces to the metadata. But also, developers have an interest in accessing metadata collected during model application, e.g., to subsequently tune their models. The optimizer, user profiles, and the quality component are the main components involved in personalization for users.

3 Component Details

After the overview of our framework focusing on non-functional properties and how different components relate to these, we now discuss details on the framework's individual components. Due to space constraints, we generalize the discussion of Metadata Sheets spread over several components (see Fig. 2). We also omit details on the Metadata Component because, while there certainly are interesting research questions to be solved for this component, it most readily can use existing technology for data integration of semi-structured data and possibly evolving query workloads (e.g., data lakes or RDF triple stores).

3.1 Metadata Sheets

Metadata Sheets (MDS) are collections of one or more metadata documents describing an object that is part of the DSS, i.e., a data set, model, or model

ensemble. MDS are ideally automatically and incrementally populated alongside the development, limiting the captured metadata to pertinent information for transparency and personalization.

The scope of information to be covered by MDS in order to support personalization and transparency is quite large. Basically, every decision during development is important and should be recorded, as it may serve as evidence of responsible development or can be used by developers to improve their product. While it is fairly easy to describe the desired information to be included in MDS, it is more challenging to precisely define, model, and collect these metadata.

To define and model the desired metadata, we can leverage existing work on responsible data set metadata [9,12] and machine learning model metadata [6, 19,28]. Take for example datasheets for data sets [9], which is a question-answer based solution. It divides questions about data sets in different categories, i.e., creation motivation, composition, collection process, preprocessing, distribution, maintenance, and legal and ethical considerations. Important information, e.g., on data set quality is not included. Moreover, the free text fields leave the amount of information, level of granularity, and level of formality open to the developers. Overall, none of the existing metadata models for data sets and models provides information on all topics we consider important. More importantly, no systematic analysis of what metadata is actually needed for transparency or personalization exists, calling for research on identifying relevant metadata for our targeted transparency and personalization queries.

While it is unlikely that all metadata can be captured automatically, we aim at maximizing the automatic collection and population of the MDS to ease the burden otherwise put on developers. We can build on provenance techniques [11] that yield metadata that keep track of (data) processing. Furthermore, data profiling techniques [1], quality metrics [3], and non-discrimination metrics [5, 10] evaluated over data sets and machine learning models allow us to capture their properties and quality. However, most research in these areas results in a new, separate tool that typically does not seamlessly integrate in a software development environment. Research is necessary to instrument software that developers are already using to capture the necessary information. This includes for instance using code markup to automatically generate metadata (analogous to, e.g., JavaDoc, requiring developers to do some, but familiar work) or "piggybacking" code on the familiar software that automatically collects metadata behind the scenes (e.g., code that automatically creates a data dependency graph from a program). Additional metadata can be automatically extracted from documents that have already been created and used for other purposes, such as requirement specifications. While natural language processing (NLP) can extract some metadata form such sources that often provide information in text form, NLP typically does not target the type of metadata required for transparency and personalization. This calls for research on information extraction techniques tailored to the metadata model used for MDS.

3.2 Optimizer

Intuitively, the goal of the Optimizer is to select the best model for each individual user or group of users, such that the choice of model is personalized rather than using a single model for all users that may not be able to manage complex decision boarders in the data stemming from different behavioral user profiles. What best means is encoded by a quality specification associated to user profiles, further discussed when presenting the Quality Component in Sect. 3.3. To achieve this personalization, we employ model ensembles [21]. Using model ensembles, the risk of making bad decisions is potentially reduced by reducing the risk of wrong results by a single poor performing model.

More formally, given the set of models for a task T of the Model Component and associated MDS, as well as a collection of multi-dimensional optimization goals that define different model application contexts, we need model ensemble algorithms that output a minimal set of model ensembles $\{E_1, \ldots, E_k\}$ suited for the optimization goals.

A first line of relevant research towards addressing this problem are approaches for fair model ensembles. Dwork et al. [8] train a linear classifier model for each user group which they then combine using a joint loss function which can also enforce fairness constraints. Calders et al. [5] train one Bayes model for each value of the sensitive value and, on top, an overall classifier which chooses the decision of one of the separate Bayes models. These model ensemble solutions demonstrate that improving fairness is possible using model ensembles. But we need more flexible solutions that call for novel algorithms for context-aware model ensembles and the support for multi-dimensional, possibly conflicting optimization goals, e.g., accuracy vs fairness. That is, we aim at creating a model ensemble based on the model quality with respect to the feature space matching a user profile. Existing algorithms in a second line of research, in particular for dynamic ensemble generation [7,14,25–27] can neither incorporate the context information nor go beyond one optimization criterion (typically accuracy).

We plan to extend the dynamic ensemble generation algorithms in order to enable personalized decision support system. First, we have to incorporate new metrics such as bias metrics, and allow for optimization of different quality goals, e.g., using skyline algorithms for pareto optimal solutions. In Sect. 4.1 we present a baseline algorithm for dynamic ensemble selection.

Of course, the performance gained by using model ensembles heavily relies on the set of input models. Within the Model Component, no model has to be of good quality for all quality metrics and all possible data objects. It suffices to achieve good quality results with respect to some quality metrics for some of the data space. In this setting, developers face the challenging problem to learn a diverse set of models that is able to cover the optimization for all quality metrics and the whole data space.

While there is research on machine learning and diversity, most of it focuses on either creating models to produce diverse results [17] or are learned to

determine appropriate diversity, or distance metrics [16]. To the best of our knowledge, no research on diversity of models has been conducted so far.

The set of multi-dimensional optimization goals stems from different application requirements on multiple dimensions by different stakeholders. When combined, these may present some goals that simply cannot be met (their score being maximized) simultaneously. It is then crucial that the set of models available in the Model Component holds at least one model that is "good enough" in at least one dimension, to achieve a partial optimization goal. However, the set of models should not be too specific or large. Thus, in devising algorithms that determine a finite set of models, future research needs to find the "sweet spot" between minimizing size of the finite set of models vs maximizing the coverage of models with respect to optimization goals, while ensuring diversity of models.

3.3 Quality Component

The Quality Component encapsulates quality specifications (QS) that specify the optimization goals mentioned above. These can relate to user preferences and profiles, be imposed by regulators, or translate the optimization goals developers aim at when devising the decision support system.

The QS are separated from the actual models as they should not be part of the application itself. The developers should not be the ones to single-handedly decide which responsibility criterion is more important than others or which fairness metric to use. That is, our framework aims for a separation of concerns regarding the development of the DSS and the decision on which optimization goals the system should strive for (given the different stakeholders with different interests in mind). After careful consideration, we opt to place the Quality Component in the service provider space (see Fig. 2) because this is where it is used and no user wants to store multiple of these files for multiple systems on their different computers. However, the service provider should allow transparent access to QS (to stakeholders with legitimate interest) and should provide means to add or update QS.

For users, we want to use user-specific information to obtain a personalized model ensemble. This requires users to provide some personal information through their user profile. We are aware of the discussion on including protected attributes in decision making. In this work, we follow the path to explore how to improve the decision making process in terms of diversity or fairness by leveraging protected attributes that allow to detect and eliminate discrimination. Nevertheless, this should comply with legal and ethical rules. When not stored at the level of a single person, i.e., when using user profiles as a generic description of a group of persons, our framework could still achieve its goal without requiring protected attributes from individuals.

Overall, QS define how a DSS should behave, in general along multiple dimensions (accuracy, runtime, bias, etc.). This resembles policies that are formal guidelines for behavior of information systems. They are for example used in the context of autonomic computing [13]. In particular Goal Policies, where a predefined desired state of the system is reached automatically if respective conditions

are met by the current state, may be used to implement QS. However, we are not aware of any techniques that deal with multiple QS with possibly conflicting desired behaviors. We have already raised this issue previously when discussing the components taking such possibly conflicting QS as input.

Given a collection of possibly conflicting QS that take multiple dimensions into account, we need an efficient algorithm to determine the minimal set of QS that is representative of the full collection. Otherwise, the other components relying on the QS would have to deal with an "unclean" input possibly yielding higher runtime or lower output quality. Clearly, the notions of minimal and representative first have to be formalized. Based on these foundations, research on algorithms to find this core set of QS or, alternatively, to enumerate possible alternative solutions can be explored.

4 Framework Implementation and Validation

Above, we have discussed how to possibly implement the different framework components, leveraging existing research and identifying open research questions. Clearly, our system framework is in its exploratory stage and it may take years to investigate all open research questions fully, especially as there is room for a variety of approaches. In order to demonstrate the framework's pertinence to filling the gap in providing personalized and transparent DSS, this section presents a first prototypical implementation, called LuPe [20]. We discuss in Sect. 4.1 how we developed the critical parts of the system framework to produce a proof of concept of our system framework. Further details on LuPe can be found in a demonstration description [20]. Section 4.2 then uses LuPe in a use case based validation of achieved personalization and transparency, respectively.

4.1 The LuPe System Prototype

So far, LuPe focuses on binary classification tasks. For illustration and also in the following evaluation, we refer to a classification problem that arises in a credit allocation scenario, where a bank employee uses LuPe to help her decide whether a customer should receive a loan or not. For our implementation, we rely on open source tools, i.e., Apache Spark ML library[1] or Angular IO[2]. We also use the publicly available German Credit Data[3] data set.

While deciding on the metadata to model within the MDS processed by LuPe, we simply settled for metadata that could be captured easily and that trivially combine existing methods. That is, we manually derived data to form MDS templates and included information on general characteristics, provenance information, variable descriptions for data sets, and model specific details for machine learning models (similar to data set nutrition labels [12]). We enriched

[1] https://spark.apache.org/docs/latest/api/python/pyspark.ml.html.
[2] https://angular.io.
[3] https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data).

the templates with automatically, collected metadata e.g., using simple data profiling methods provided by Apache Spark and bias metrics that we implemented ourselves. All MDS can be viewed online[4].

In order to create a diverse set of models in LuPe, we chose to optimize models for accuracy because this option was available out-of-the-box in Apache Spark ML. We introduced model diversity by using used different types of Apache Spark ML learners and different subsets of the training data set to learn multiple binary classification models.

In LuPe, the quality specifications define for a given state (i.e., of user group feature characteristics) the desired state in form of a quality metric that has to be optimized by the system. More formally, we define a quality specification QS as a 3-tuple $QS = (og, c, s)$, with a set og of optimization goals (e.g, minimize false positives), a set of constraints c that the model should satisfy, and an optional corresponding action if the constraint cannot be met, e.g.,

$$\begin{aligned} &\text{apply if } \frac{p(label = positive|gender = female)}{p(label = positive|gender = male)} \geq 0.8 \\ &\text{discard } otherwise \end{aligned}$$

An example of a quality specification in JSON format is given below. The optimization goal of $q1$ is "maximize accuracy metric" and the expression in the user group determines that $q1$ applies to women. Apart from optimizing the system for all users, this Quality Profile allows to specifically aim at maximizing the accuracy for women.

```
{‘‘id’’: ‘‘q1’’,
‘‘description’’: ‘‘Maximize accuracy for women’’,
‘‘optimizationGoal’’: {
    ‘‘qualityMetric’’: ‘‘accuracy’’,
    ‘‘direction’’: ‘‘max’’},
‘‘userGroup’’: ‘‘sexAndStatusclassVec_A92 = 1.0’’}
```

To implement an Optimizer for the Model Ensemble Component of LuPe, we extend dynamic ensemble generation as existing algorithms [7,14,25–27] cannot take multi-dimensional optimization goals into account.

Algorithm 1 shows pseudocode for a baseline Optimizer implementation for dynamic model selection. More formally, given a quality specification $QS = (og, c, s)$, a set of models $\mathcal{M}$, and respective metadata MDS_i^M for model $M_i \in \mathcal{M}$, find the model $M \in \mathcal{M}$ which fulfils the set of constraints c and optimizes the optimization goal og. Given a quality specification QS, we first compare the available models to the constraints in order to eliminate all invalid models, i.e., all models whose quality profile does not qualify them to be used. Next, we apply Dynamic Classifier Selection by Local Accuracy [26] on the valid set of models. To explain our model decision in detail, we use the ranked list of models with

[4] LuPe website: https://www.ipvs.uni-stuttgart.de/departments/de/research/projects/fat_dss/.

Algorithm 1: Dynamic model selection implementation on quality specifications.

Input: $QS = (og, c, s)$, $\mathcal{M} = \{M_1, ..., M_n\}$
Output: $\mathcal{M}_{sorted}$: list of models ranked by their quality
$\mathcal{M}_{sorted} \leftarrow \emptyset$ *//new List of models*
$\mathcal{M}_{valid} \leftarrow \emptyset$ *//new List of models*
forall the $M \in \mathcal{M}$ **do**
  forall the $c \in QS.c$ **do**
    if $M.getMDS.qualityprofile\ satisifes\ c \wedge c.event \neq$ *'discard'* **then**
      $\mathcal{M}_{valid} \leftarrow \mathcal{M}_{valid} \cup M$
$\mathcal{M}_{sorted} \leftarrow DCSLA(og, \mathcal{M}_{valid})$
$MDS \leftarrow generateMetadata(\mathcal{M}_{sorted}, QS)$
$MDSStore.add(MDS)$
return $\mathcal{M}_{sorted}$

respect to the accuracy for *og* in the feature space next to *s*. As we described earlier, we also store a MDS^E for each model ensemble, which is why we collect the necessary information and store it in the MDS store.

The Metadata Component is, at its essence, a data integration system, for which a large body of research and practical solutions exist. However, since we opted to create all MDS in a similar fashion and we only created one MDS for each model or data set in our prototype, we do not need to integrate the MDS to illustrate the power of our system framework idea. We simply store the JSON files locally without making use of bigger storage system solutions.

In general, implementations of our framework need sophisticated querying capabilities to satisfy the different needs of the various stakeholders. This can be achieved using visualizations, query languages, chatbots or a combination of all approaches. To keep it simple in our first framework implementation, we opt to rely on simple functions which can be called to access the stored MDS data, which obviously restricts the flexibility of the query engine. LuPe provides access to lists of models and data sets, as well as all MDS details for other components. In accordance with the implementation description of the Model Ensemble Component, the optimizer queries and uses this information to identify personalized model ensembles. We also provide human stakeholders with access to the MDS Store through visualizations. Screenshots of these visualizations, in particular informed consent forms provided for users and component overview and component details provided for developers and regulators are shown in Fig. 3.

Given the above description of the LuPe framework implementation, we validate that it achieves personalization and transparency through two use cases.

4.2 Use Case Based Validation of Personalization and Transparency

Personalization. Our first study focuses on personalized decision support, with the intention of reducing gender bias. In order to assess whether the quality has indeed improved when personalizing for the user group gender, we perform a quantitative evaluation. For evaluation purposes, the data set needs to be split into two data sets $D1$ and $D2$, the former being used to train and evaluate the

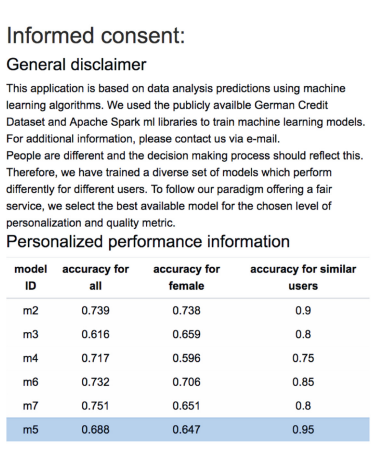
Table 1. Accuracy of individual models during training (a) and model ensemble (b)

Model	Accuracy (All)	Accuracy (sex='Female')	Accuracy (sex='Male')
$m1$	0.8283	0.9255	0.7796
$m2$	0.8310	0.9247	0.7841
$m3$	0.7633	0.9059	0.6918
$m4$	0.7731	0.9116	0.7036
$m5$	0.7771	0.9056	0.7127
$m6$	0.8402	0.9101	0.8052
$m7$	0.8397	0.9140	0.8025
$m8$	0.8141	0.8910	0.7756
$m9$	0.7977	0.8959	0.7484
$m10$	0.7650	0.8756	0.7096

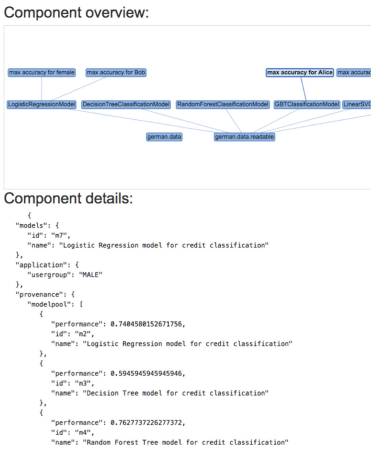
(a) Accuracy of different models for all users and for gender user groups

Number of users	9792
Accuracy	
Without personalization	0.8509
With personalization	0.8531

(b) Accuracy of model ensemble



(a) Alice's informed consent sheet



(b) Backend overview

Fig. 3. Screenshots of the credit classification scenario LuPe frontend [20].

models and the latter to evaluate the model ensembles. Since the previously mentioned German Credit data set is too small to be further split, we apply LuPe for a second scenario. In this scenario, we classify users with respect to their wage using the publicly available Adult[5] data set. The respective MDS for this second scenario are available at the LuPe website as well. Accuracy results for ten individual models $m1$ through $m10$ during the training phase are shown in Table 1(a), while Table 1(b) reports results of the model ensembles.

Using the overall best model $m6$ during training for all users in the test data set achieves an accuracy of 0.8509. When opting for the best model for different user groups, i.e., $m1$ for female and $m6$ for male users, we achieve a slightly higher accuracy of 0.8531. While LuPe performs slightly better on this very low personalization level that merely distinguishes users according to gender, this is

[5] https://archive.ics.uci.edu/ml/datasets/adult.

not yet satisfactory. Having used LuPe, we can study the metadata sheets. We uncover that the data shows class imbalance with about two thirds of the data in the bad outcome class. We assume as a result that several models present high false negative rates and that the models' overall quality might not be sufficient. This insight is gained by leveraging the transparency feature of LuPe, further discussed below. Clearly, the results we obtain show that further research is needed on diversifying the set of trained models as discussed in Sect. 3.2. Also, for better results, we have explored more fine grained personalization levels, i.e., based on profiles similar to a user profile (see Fig. 3(a)). While accuracy results are promising, the algorithm does not scale to the larger Adult data set. This emphasizes the importance of future research on more fine grained model quality profiles and more efficient personalization algorithms.

Transparency. For regulators and developers, LuPe offers a glimpse behind the scenes which is depicted in Fig. 3(b). This visualization provides these stakeholders with an overview of the ingredients forming the system, including all data sets, models, and model ensembles. Additionally, in this simplistic visualization we provide all metadata information for each ingredient on demand. Using the metadata stored by LuPe we are already able to give an overview on possibly complex systems and to answer various audit questions such as *"Has the data been normalized before model training?"* or *"What data was the model trained on?"*. But clearly, a lot of additional metadata is necessary to support the wide range of possible queries. While this emphasizes the need to model and collect additional metadata for various types of queries, these queries need to be efficiently executed and the results presented to users in an understandable way.

For a user, LuPe offers a visual interface that displays an informed consent sheet (see Fig. 3(a)). It textually points out the intention of using personalization and the impact that different personalization levels may have on the accuracy experienced by the user. Determining if, how, and to what extent such information (and its presentation) affect users needs to be systematically studied, e.g., through user studies.

5 Conclusion and Outlook

This paper presented a system framework for personalized and transparent decision support systems. Exploring such systems is essential in addressing the challenges that arise with the increasing demand to devise fair, accountable, and transparent data-driven decision support systems.

After a system framework overview, we discussed individual components in detail, highlighting the current state of the art and discussing why existing solutions are not adequate. We then presented a first implementation of our framework, incorporating first baseline solutions that address the more general research challenges that need to be further explored. Based on two use cases, we validated that our solutions to enable personalized and transparent decision support systems are promising, and are worth investigating in the future.

Exploring and optimizing the technical feasibility and aspects of supporting personalization and transparency is only the first step in quickening the adoption of responsible decision support systems in practice. further research is necessary in many different directions. To name just a few, these include aspects of security and privacy, reduction of ramp-up costs of setting up and maintaining such systems to continue to allow start-ups to be profitable, or resolving the tension between transparency and intellectual property, as proprietary models are at the heart of many modern business models.

References

1. Abedjan, Z., Golab, L., Naumann, F.: Profiling relational data: a survey. VLDB J. **24**(4), 557–581 (2015). https://doi.org/10.1007/s00778-015-0389-y
2. Barocas, S., Selbst, A.D.: Big data's disparate impact. Calif. L. Rev. **104**, 671 (2016)
3. Batini, C., Scannapieco, M.: Data Quality: Concepts Methodologies and Techniques. Data-Centric Systems and Applications. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-33173-5
4. Burrell, J.: How the machine 'thinks': understanding opacity in machine learning algorithms. Big Data Soc. **3**(1), 1–12 (2016)
5. Calders, T., Verwer, S.: Three naive Bayes approaches for discrimination-free classification. Data Min. Knowl. Discov. **21**(2), 277–292 (2010)
6. Data Mining Group: Portable format for analytics (PFA). http://dmg.org/pfa/index.html. Accessed 11 Oct 2019
7. Dos Santos, E.M., Sabourin, R., Maupin, P.: A dynamic overproduce-and-choose strategy for the selection of classifier ensembles. Pattern Recogn. **41**(10), 2993–3009 (2008)
8. Dwork, C., Immorlica, N., Kalai, A.T., Leiserson, M.: Decoupled classifiers for group-fair and efficient machine learning. In: FAT* (2018)
9. Gebru, T., et al.: Datasheets for datasets. In: FAT ML (2018)
10. Hajian, S., Domingo-Ferrer, J.: Direct and indirect discrimination prevention methods. In: Custers, B., Calders, T., Schermer, B., Zarsky, T. (eds.) Discrimination and Privacy in the Information Society: Data Mining and Profiling in Large Databases, vol. 3. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-30487-3_13
11. Herschel, M., Diestelkämper, R., Ben Lahmar, H.: A survey on provenance: What for? What form? What from? VLDB J. **26**(6), 881–906 (2017). https://doi.org/10.1007/s00778-017-0486-1
12. Holland, S., Hosny, A., Newman, S., Joseph, J., Chmielinski, K.: The dataset nutrition label: a framework to drive higher data quality standards. Draft (2018)
13. Kephart, J.O., Walsh, W.E.: An artificial intelligence perspective on autonomic computing policies. In: POLICY (2004)
14. Ko, A.H., Sabourin, R., de Souza Britto Jr., A.: From dynamic classifier selection to dynamic ensemble selection. Pattern Recogn. **41**(5), 1718–1731 (2008)
15. Kroll, J.A., Barocas, S., Felten, E.W., Reidenberg, J.R., Robinson, D.G., Yu, H.: Accountable algorithms. Univ. PA Law Rev. **165**, 633–705 (2016)
16. Kulis, B.: Metric learning: a survey. Found. Trends® Mach. Learn. **5**(4), 287–364 (2013)
17. Li, J., Galley, M., Brockett, C., Gao, J., Dolan, B.: A diversity-promoting objective function for neural conversation models. In: NAACL-HLT (2016)

18. Lowry, S., Macpherson, G.: A blot on the profession. Br. Med. J. (Clin. Res. Ed.) **296**(6623), 657–658 (1988)
19. Mitchell, M., et al.: Model cards for model reporting. In: FAT* (2019)
20. Oppold, S., Herschel, M.: LuPe: a system for personalized and transparent data-driven decisions. In: CIKM (2019)
21. Polikar, R.: Ensemble based systems in decision making. IEEE Circ. Syst. Mag. **6**(3), 21–45 (2006)
22. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should I trust you?": explaining the predictions of any classifier. In: KDD (2016)
23. Singh, J., Cobbe, J., Norval, C.: Decision provenance: harnessing data flow for accountable systems. IEEE Access **7**, 6562–6574 (2019)
24. Sweeney, L.: Discrimination in online ad delivery. Queue **11**(3), 1–19 (2013)
25. Woloszynski, T., Kurzynski, M.: A probabilistic model of classifier competence for dynamic ensemble selection. Pattern Recogn. **44**, 2656–2668 (2011)
26. Woods, K., Kegelmeyer, W.P., Bowyer, K.: Combination of multiple classifiers using local accuracy estimates. TPAMI **19**(4), 405–410 (1997)
27. Woźniak, M., Graña, M., Corchado, E.: A survey of multiple classifier systems as hybrid systems. Inf. Fusion **16**, 3–17 (2014)
28. Yang, K., Stoyanovich, J., Asudeh, A., Howe, B., Jagadish, H.V., Miklau, G.: A nutritional label for rankings. In: SIGMOD (2018)