Smallset Timelines: A Visual Representation of Data Preprocessing Decisions

Lydia R. Lucchesi
Australian National University & CSIRO's Data61
Lydia.Lucchesi@anu.edu.au

Petra M. Kuhnert
CSIRO's Data61 & Australian National University
Petra.Kuhnert@data61.csiro.au

Jenny L. Davis
Australian National University
jennifer.davis@anu.edu.au

Lexing Xie
Australian National University & CSIRO's Data61
lexing.xie@anu.edu.au

ABSTRACT

Data preprocessing is a crucial stage in the data analysis pipeline, with both technical and social aspects to consider. Yet, the attention it receives is often lacking in research practice and dissemination. We present the Smallset Timeline, a visualisation to help reflect on and communicate data preprocessing decisions. A "Smallset" is a small selection of rows from the original dataset containing instances of dataset alterations. The Timeline is comprised of Smallset snapshots representing different points in the preprocessing stage and captions to describe the alterations visualised at each point. Edits, additions, and deletions to the dataset are highlighted with colour. We develop the R software package, `smallsets`, that can create Smallset Timelines from R and Python data preprocessing scripts. Constructing the figure asks practitioners to reflect on and revise decisions as necessary, while sharing it aims to make the process accessible to a diverse range of audiences. We present two case studies to illustrate use of the Smallset Timeline for visualising preprocessing decisions. Case studies include software defect data and income survey benchmark data, in which preprocessing affects levels of data loss and group fairness in prediction tasks, respectively. We envision Smallset Timelines as a go-to data provenance tool, enabling better documentation and communication of preprocessing tasks at large.

CCS CONCEPTS

• **Human-centered computing** → **Visualization toolkits**.

KEYWORDS

data preprocessing, visualization, communication, open-source software, reflexivity

1 INTRODUCTION

Prior to an estimation task, data practitioners are often faced with difficult decisions about how to make their dataset functional *for* the estimation task. For example, one may need to decide how to deal with missing values to build a random forest classifier. These data preprocessing decisions are important as they not only make the intended analysis possible but can also influence its outcome. Their influence on estimation outcomes has been demonstrated, quantitatively, in the fields of fair machine learning [17], natural language processing [11], and psychology [46], to name a few. Yet, in general it is less common to encounter meaningful detail about the preprocessing stage in discussions about research outputs, than it is to learn about how the data were collected and modelled [32]. Preprocessing decisions often remain tucked away in code—either inaccessible or difficult to parse, limiting our ability to interpret and replicate results.

Communicating and documenting data preprocessing is one aspect of data provenance, a broader concept referring to all aspects of dataset production. An influx of interest in data provenance in the machine learning community has led to work exploring how we might better record and utilise information about a dataset's creation [12, 19, 26, 27, 32, 38, 43]. Preprocessing is mentioned in the provenance literature, but because there are many aspects of provenance, it receives limited attention. Meanwhile, the field of information visualisation has produced tools to study data provenance and its effects. Some support visualisation of the entire data pipeline [51], data lineage [8], or data flow [54]. Others are interactive [4, 5, 36] or animated [29, 41]. To the best of our knowledge, none of the existing tools focus on visualising the *decisions* made during preprocessing in a way that is static and compact. We choose to focus on this.

We present the Smallset Timeline (or Timeline), a visualisation of data practitioners' preprocessing decisions (Section 4). A Smallset is a small collection of rows from the dataset containing examples of data alterations. Rows are selected by random sampling or one of the proposed optimisation algorithms (Section 5). The Timeline is comprised of Smallset snapshots representing different points in the preprocessing steps and captions to describe the alterations visualised at each point. Edits, additions, and deletions to the data are highlighted with colour. It is a static, compact visualisation designed to be useful for both Timeline creators and readers (Table 1). A Timeline creator is one who makes a Smallset Timeline to reflect on and communicate their decisions. A Timeline reader is one who views it to understand, evaluate, and/or replicate the preprocessing

steps. We present the smallsets R package (Section 6), which is used to produce all Timelines in this work, including those in the case studies (Section 7). The first case study uses Smallset Timelines to document decisions related to varying amounts of data loss in software defect data from the NASA Metrics Data Program. The second case study explores American Community Survey benchmark datasets from the folktables tool [13] and the subtle downstream effects of combining different filtering and threshold decisions.

The main contributions of this work are:

- The Smallset Timeline, a static, compact visualisation to communicate data preprocessing decisions.
- The open-source package smallsets[1], for producing Smallset Timelines for R and Python preprocessing scripts.
- Two case studies, in which Smallset Timelines document preprocessing decisions that affect comparability of results from different studies as well as dataset imbalance and group fairness in machine learning tasks.

2 RELATED WORK

We review several areas of related research that motivated and inspired the creation of the Smallset Timeline. These areas include 1) studying the effects of preprocessing decisions on outcomes from data analytics tasks, 2) documenting data provenance, and 3) visualising data provenance information.

There is rarely a clear-cut preprocessing route for practitioners to follow. Instead, practitioners must make decisions about how to prepare data for analyses. Research about **preprocessing effects** investigates if study outcomes are sensitive to these decisions. For example, Friedler et al. [17] uncover dependence between performance of fairness-enhancing algorithms and preprocessing choices. Blocker and Meng [3] introduce the concept of multiphase inference for preprocessing to obtain better estimators. Steegen et al. [46] propose multiverse analyses, in which a dataset is prepared a number of reasonable ways for estimation. They demonstrate with a psychology case study that estimation outcomes can be sensitive to differences in data preprocessing. Similarly, Denny and Spirling [11] show that, in the preparation of political texts for unsupervised learning tasks, "under relatively small perturbations of preprocessing decisions—none of which were *a priori* unreasonable—very different substantive interpretations would emerge" [p. 187]. Conversely, in an experiment by Shirk et al. [45], three participants remove artefacts from electroencephalogram (EEG) data, and despite varied approaches, the results remain stable. However, generally speaking, these works do provide compelling quantitative evidence that preprocessing *can* shape the trajectory of an analysis. From this literature, we can conclude that careful review and strong communication of preprocessing decisions are important.

The **documentation of data provenance** is a growing area of interest for the machine learning and natural language processing communities. The goal is to record important information about a dataset and support informed use of data and models. Datasheets for datasets [19], model cards [34], FactSheets [1], data statements [2], and the Dataset Nutrition Label [25] are proposed templates and frameworks for recording information about a dataset, including changes made to it. For example, Question 33 in the datasheets for

datasets template asks: "Was any preprocessing/cleaning/labeling of the data done (for example, discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)?" [19, p. 90]. These documentation techniques aim to be comprehensive in their coverage of provenance information. Recorded preprocessing details are just one part of the documentation. With the Smallset Timeline, we hope to contribute to this research area with a technique focused exclusively on the topic of data preprocessing. Furthermore, we explore the pairing of text *and* visuals to describe data alterations.

Next, we discuss **visualisations for data provenance** that include information about preprocessing or enable end users to study data transformations. Wang et al. [51] use data comics [55] to describe an analytical process, with some panels dedicated to data transformations. Cui and Widom [8] propose a data lineage tracing algorithm and exploration tool. DQProv Explorer [4] and VisTrails [5] are multi-view interactive visualisation systems providing insight into the transformations undergone by a dataset. TACO [36] is another interactive system offering several visual summaries for data table comparisons across time. Khan et al. [29] develop "data tweening," which involves animating the transformations occurring between two database queries. A "datamation" [41] animates plotted data points to showcase restructuring tasks, while Yang et al. [54] propose *fair*-DAGs for identifying bias in preprocessing pipelines. These tools convey provenance information using sketches, interactivity, animation, and directed acyclic graphs (DAGs). We focus solely on preprocessing and propose a static timeline of steps. The design is intended to be simple and practical. We discuss the Smallset Timeline in detail in Section 4 after outlining the role of this visualisation in Section 3.

3 THE ROLE OF SMALLSET TIMELINES

In this section, we first clarify the meaning of *data preprocessing* used in this work. We then define the roles of the Smallset Timeline for different users and goals. *Data preprocessing* is a commonly used term in the research and practice of data science, but the term carries a diverse set of meanings that vary with context and audience. Following are three example views of preprocessing, ranging from specific to general. For text data, Denny and Spirling [11] view preprocessing as the set of "decisions about how words are to be converted into numbers" [p. 168]. Focused on the role of preprocessing in data mining, García et al. [18] define it in terms of two broad task categories, including *data preparation* and *data reduction*, and the sub-tasks they encompass, e.g., *data cleaning* or *feature selection*. One notable challenge of this approach is developing a classification scheme that is comprehensive. In a general overview of data preprocessing, Famili et al. [16] simply define it as " all the actions taken before the actual data analysis process starts" [p. 5]. We build on this last conceptualisation and adhere to a minimalist conception of data preprocessing, focusing on its boundaries with other stages in the data pipeline.

Figure 1 pictures a three-stage pipeline, consisting of 1) data collection, 2) data preprocessing, and 3) estimation and modelling. We consider data collection complete when the information of interest exists in a location separate from the source and data preprocessing complete when the dataset can be used to produce the intended

---

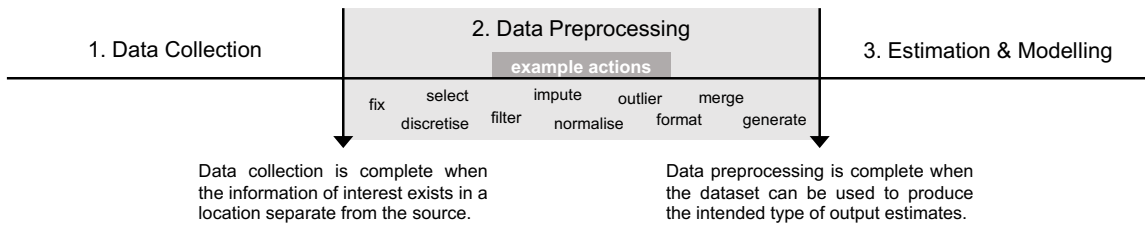[1]https://github.com/lydialucchesi/smallsets

Figure 1: Operational definition for data preprocessing. Boundary definitions distinguish preprocessing from neighbouring stages in the data pipeline. In practice, some iteration between stages may be necessary.

Table 1: Design goals for Smallset Timelines - including users, utilities, and the corresponding format variations.

	Reflect	Replicate	Comprehend & Evaluate
User: action	Data practitioner: create	Data practitioner: read	Target audience: read
Outcome	Asks practitioners to recount their decision-making process in the Timeline captions, encouraging reflection for the data preprocessing stage.	Provides visual examples and written descriptions of preprocessing steps performed in a programming language. Documents information in a stable format that can be saved.	Provides an accessible preprocessing narrative with enough information for decisions to be understood and assessed. Highlights the role humans play in data production.
Presentation	Reflection occurs while writing captions for the Timeline.	Smallset has more rows. Timeline has more snapshots. Captions are detailed.	Smallset has fewer rows. Timeline has fewer snapshots. Captions are succinct.
Example	Figures 6 to 8	Figure 7	Figures 6 and 8

type of output estimates. These boundaries delimit the beginning and end of preprocessing. Actions altering the dataset within these boundaries are considered preprocessing. Figure 1 includes some example actions, and more examples can be found in Famili et al. [16], Kasica et al. [28], and Luengo et al. [31], to name a few. This conceptualisation of preprocessing is robust to variability in the location of data operations across analyses. For instance, inference can generate a new feature or produce the final result. Here, if the inference task changes the dataset to facilitate the analysis, it is preprocessing. Although our definition implies a linear and pre-defined analytic strategy, we acknowledge and account for exploratory or iterative processes through the **resume marker** feature introduced in Section 4.2. However, we omit a full discussion of this feature due to space constraints.

We develop the Smallset Timeline to capture the nature of preprocessing actions altering the dataset. We create the visualisation and tool to serve several functions, outlined in Table 1. One function is to support reflection on preprocessing decisions by the person who made the decisions, or the Timeline creator. There is a growing call to incorporate reflection—especially reflexivity [15, 33, 49]—into data science work to acknowledge the context and subjectivities involved in it. Asking practitioners to recount their decision-making process in the Timeline captions aims to promote reflection about the preprocessing stage. The second function is to support replication of the steps by other researchers. Reproducibility is considered a cornerstone of science [37], and being able to replicate preprocessing is an essential component of reproducing data-based results. The third function is to support comprehension among a Timeline

creator's target audience. Given the importance of preprocessing decisions, as established in Section 2, getting preprocessing decisions out of code and into an accessible and practical format is crucial for making these decisions legible and thus open to evaluation. Next, we describe the Smallset Timeline design and how it affords these socio-technical functions.

4 SMALLSET TIMELINE DESIGN

The design of any tool enacts priorities through its "affordances," or how technical features interplay with human users to produce socially meaningful effects [10]. In this section, we describe the design of the main visual artefact, the Smallset Timeline, noting how specific design choices relate to intended use-functions, including reflection, replication, and comprehension/evaluation (Table 1).

A Smallset Timeline has three basic components: a *Smallset* consisting of a small subset of data to illustrate preprocessing decisions, *snapshots* that each visualise one or more preprocessing decisions, and *captions* that describe changes made to the data (Section 4.1). The timeline also has four enrichment design features: *printed data*, *missing data tints*, *ghost data*, and *resume markers* (Section 4.2). We also generate alternative narratives (*alt text*) for Smallset Timelines for those with visual impairments (Section 4.3). Throughout this section, we use a synthetic dataset and preprocessing scenario to illustrate various design components and their functions. The synthetic dataset consists of 100 rows and 8 features. The main preprocessing steps are 1) filtering rows, 2) dealing with missing data, and 3) generating a new feature. More information about the synthetic dataset and preprocessing scenario can be found in Appendix A.

4.1 Key components

A **"Smallset"** is a small collection of observations featuring examples of data alterations occurring in the dataset of interest. In this section, we assume these observations are given in order to focus on the visual elements of the design component. Smallset selection criteria and algorithms are discussed in Section 5. The design goal is to create a small object that can demonstrate preprocessing steps at a manageable scale for comprehension and figure production. A Smallset contains approximately 5-15 observations to keep the visualisation compact. The current version of the Smallset Timeline tool works for tabular data only, meaning the Smallset also has this table format. Each observation is a row. Each attribute is a column, and there is no nested data structure (e.g., lists or other key-value structures) in a cell.

Small empty tables have long been used in the programming community to explain coding commands for data manipulation. For example, the cheat sheet for the R dplyr package [42] uses little (empty) tables and colour to visually explain to data scientists what happens to the data object when a dplyr command is applied to it. With the Smallset, we employ the same technique. It allows Timeline creators to demonstrate to Timeline readers what happens to a dataset as a result of their preprocessing decisions. A Smallset is not limited to showing one operation at a time but can instead show multiple programming steps at once (e.g., Figure 2). Providing real examples of dataset changes, in a convenient viewing format, is one way to make preprocessing transparent to Timeline readers.

Snapshots are pictures of the Smallset table at a particular moment in the data preprocessing steps. Snapshots break the process into digestible pieces and are plotted sequentially in a timeline to mirror the sequence of programming instructions used to implement a data preprocessing strategy. The first snapshot shows the data prior to any preprocessing, while the last presents it fully preprocessed. Snapshots in-between represent intermediary points, selected by the Timeline creator (by simply inserting structured comments, see Section 6). Snapshots can be arranged in a single row or across multiple rows.

The system uses a set of colours to highlight data changes in a snapshot. The colours represent general changes undergone by a dataset: 1) it gets bigger, 2) it gets smaller, or 3) it stays the same size, but the contents change. In short, it is a colour scheme distinguishing between data additions, deletions, edits, and unchanged data. We limit the number of colours to four to minimise consultation with the colour legend while reading a Timeline. Timeline creators can choose a four-colour palette consistent with the visual style of their document, and colourblind-friendly palettes are available in the smallsets package. The colour for a data change not appearing in the Timeline is dropped from the legend (e.g., Figure 6). We leave experimenting with the number and type of labelled changes, as well as the option to assign colours to specific operations, as future work.

Timeline creators are expected to exercise discretion in snapshot-taking based on their goals and presentation format. In Figure 2, Timeline creator Alice chooses to take snapshots showing exactly one operation at a time. As noted in Table 1, this type of approach emphasises the effects of each operation and helps prepare documentation for replicating the data preprocessing tasks. Alternatively, another Timeline creator, Bob, groups related operations together as a composite preprocessing step. This type of approach aims to convey the conceptual outline rather than the details of preprocessing. It is suited to mediums in which space and reader attention span are limited, such as a research article, white paper, or blog post. It should be noted that if a data point has been altered more than once since the last snapshot, the cell colour will reflect the most recent change, i.e., one operation becomes hidden behind another. However, we choose to prioritise simplicity and minimise visual clutter.

Captions accompany snapshots to provide information about the alterations visualised in the Smallset. Timeline creators are responsible for providing the captions (by populating a caption template, see Section 6), which should supply Timeline readers with information that enhances their understanding of the process. This text is generally located beneath snapshots but could be placed to the side, if a Timeline is arranged vertically.

At the most basic level, a caption says what was done in the preprocessing step. The colour categories for data changes are broad, so a caption allows the exact nature of the change to be stated. From there, the caption can be upgraded to also explain why it was done. Timeline creators can use the caption space to defend and discuss their preprocessing decisions. Explaining why is especially important if a decision deviates from a preprocessing norm in one's field. In some instances, it may be necessary to also specify how it was done. This part can be essential for Timeline readers trying to replicate the preprocessing steps.

The caption style will depend on the purpose of the Timeline. To caption appropriately for general comprehension (Table 1 column 3), jargon is avoided, and the text is pared back to the most relevant parts to prevent information overload. Caption 1 in Figure 6 provides an example of a simple caption for general comprehension: *Remove columns that have the same value for every row because they do not provide any information for modelling.* For the purpose of replicating data preprocessing tasks (Table 1 column 2), captions may be detailed, include jargon, and reference preprocessing code. Those reproducing the steps likely have some familiarity with the topic, such that the amount and type of information are not overwhelming. The captions in Figure 7 are an example of captioning to enable replication. For example, the step 4 caption lists the integrity rules used to check for implausible values.

4.2 Enrichment features

The Smallset Timeline is designed with four enrichment features. Their use is at the discretion of Timeline creators and should depend on data privacy as well as audience and goal (Table 1). A visual overview of the features is in Figure 3.

Printed data (Figure 3 column A) can be included in Smallset tables for a glimpse of the data and real examples of how it changes between steps. If reading a Smallset Timeline to help reproduce a dataset, printed values provide a chance to compare values between the original dataset and the reproduced version. Even if preprocessing code appears to run successfully, having the printed values might verify that it still does what the author intended. If data

Figure 2: Diagram showing discretion in snapshot point selection. Alice generates snapshots for each step. Bob combines preprocessing steps in snapshots. Snapshots are based on synthetic data (Appendix A).
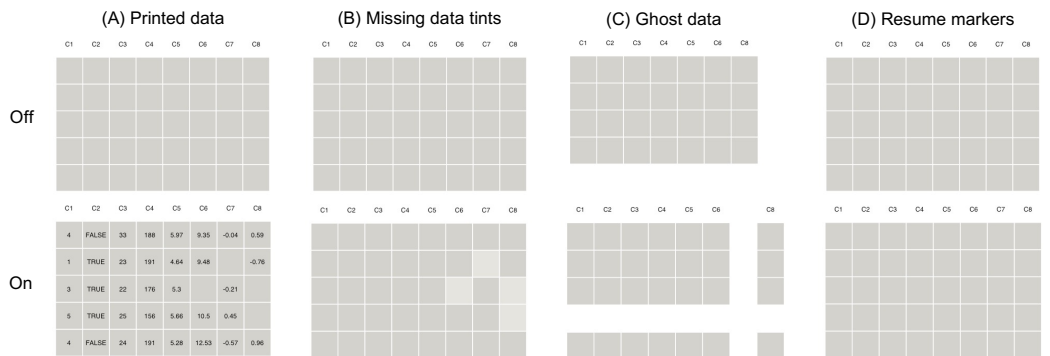
Figure 3: Overview of Smallset enrichment features. See Section 4.2 for descriptions. Snapshots are based on synthetic data (Appendix A).

are not publicly available, the Timeline can be configured without printed data. Note that omitting the data does not necessarily guarantee data privacy. We hope to address Smallset data privacy in future work for applications with sensitive data.

The "missing value shadow" [47], or "shadow matrix" [50], is a visual technique that contrasts light and dark to reveal missing values in a table of data. Utilising the concept, the Smallset Timeline tool offers the option to indicate missing values in a Smallset with **missing data tints** (Figure 3 column B). A subtle tint makes the issue noticeable without diverting attention from other visual elements. It also has metaphorical value, as if part of the colour is missing. If values are imputed, the table cells are not filled with tints in the following snapshots.

When deletion is demonstrated in the Smallset, the Smallset table naturally shrinks. When this happens in a Timeline, it can become difficult to track data points across the Timeline as they shift in space relative to each other. The **ghost data** enrichment feature (Figure 3 column C) provides the option to plot blank ("ghost") rows and columns where data have been deleted. The position of each

table cell is then maintained throughout the Timeline, and data can be readily traced across it (e.g., Figure 6, Figure 7, and Figure 8).

Resume markers (Figure 3 column D) are the fourth enrichment feature. A resume marker is a vertical line placed between two snapshots to denote that preprocessing stopped to begin the estimation task and then subsequently restarted for additional data alterations to be made. This feature is designed to be of use when discussing iterations in more exploratory analyses or unexpected issues that necessitate modifying the initial estimation plan. In the latter case, this enrichment feature is not meant to condone any type of data dredging but allow Timeline creators to be transparent about unforeseen roadblocks, resulting in multiple attempts at estimation. An example Smallset Timeline with a resume marker can be found in Appendix A (Figure 12).

4.3 Alt text

In the present work, we argue that visualisation is a good way to make preprocessing information accessible. However, visualisations are not accessible to people with visual impairments unless there is *alt text*. Therefore, we develop an *alt text* template (available in

Appendix B) for generating text descriptions of Smallset Timelines. The software tool, discussed in Section 6, automatically populates the template and saves the output in a text file. When populated, it details the Timeline title, snapshot count, colour legend, and individual snapshots. This output can be manually modified for clarity and readability and included alongside figures. An example of the automated *alt text* as well as the manual edits a practitioner could make prior to dissemination are included in Appendix A.5.

5 SMALLSET SELECTION

Section 4 assumes a small set of rows from a tabular dataset is given. This section discusses three strategies to automatically select these rows from the original dataset. There are two main criteria for selecting a Smallset. *Preprocessing coverage* tries to ensure that at least one row in the Smallset is affected by each preprocessing step, so that all snapshots in the timeline have a visible change. *Visual variety* aims to select a set of rows that are affected by the set of preprocessing steps differently, so as to represent a range of changes from preprocessing. Although manual selection is an option, it is subject to cherry-picking, which may result in a misleading visualisation. The tool offers three methods for automated selection to bolster integrity of the visualisation presented.

Random sampling, the first automated selection method, may achieve *preprocessing coverage* and *visual variety* if preprocessing operations are widespread throughout a dataset. However, when the number of Smallset rows is low, and when some preprocessing operations affect only a small number of rows in the original data, neither *preprocessing coverage* nor *visual variety* are guaranteed, which makes other automated selection methods desirable.

5.1 Two optimisation problems

Automatic selection algorithms require two additional data representations generated from the preprocessing steps: the *coverage indicator matrix*, $\mathbf{C}$, and the *visual appearance matrix*, $\mathbf{A}$. Denote the original dataset $\mathbf{X}$ as an $N \times M$ matrix, with x_{ij} being the data value in the i-th row and j-th column. Data matrix $\mathbf{X}$ goes through $h = 1, \ldots, H$ preprocessing steps, $f_1, \ldots, f_H$, resulting in a processed data matrix after each step $\hat{\mathbf{X}}_h = f_h \ldots f_1(\mathbf{X})$. The binary coverage matrix $\mathbf{C} \in \{0, 1\}^{N \times H}$ is sized by the N data points and H preprocessing steps. Each element c_{ih} is 1 *iff* the ith row is altered by preprocessing step f_h, 0 otherwise. The appearance matrix $\mathbf{A} \in \mathcal{R}^{N' \times M'}$ is the size of the original data matrix plus any rows/ columns added. Its elements $a_{ij} \in \{'U', 'E', 'A', 'D'\}$ (corresponding to *unchanged, edited, added, deleted*, respectively) encode the last change that a data cell undergoes from the original data matrix $\mathbf{X}$ to the final data matrix $\hat{\mathbf{X}}_H$. Example *coverage indicator* and *visual appearance* matrices for the synthetic dataset are available in Appendix A.6.

We use these data structures to set-up two optimisation problems (Problem 1 and Problem 2 shown in Table 2) for selecting a Smallset of size K. Problem 1 accounts for *preprocessing coverage* only. The output of this selection problem is an indicator vector $\mathbf{z} \in \{0, 1\}^N$, with z_i being 1 if row i is selected, 0 otherwise. The first constraint ensures that exactly K rows are selected out of the original N rows. In the second constraint, the left hand side computes the number of rows that preprocessing step h affects, and we require this to

be greater than 0 for each step. This is an integer linear program solved using the Gurobi [23] optimisation software. In other words, the *coverage* problem tries to satisfy the two constraints without any additional objective (hence the max 1 term in Table 2). We tried maximising the number of changes shown, but that led to solutions that favour rows with many changes, that may all be similar to each other – which motivates the *visual variety* criterion and Problem 2.

Problem 2 additionally accounts for *visual variety*. This requires a pre-calculated $N \times N$ distance matrix Q containing the hamming distance between the appearance vector of any two rows. That is, $q_{il} = \sum_j d(a_{ij}, a_{lj})$, with distance function $d(\cdot, \cdot)$ being 0 if the two values are the same, 1 otherwise. The objective function $\mathbf{z}^\top Q \mathbf{z}$, therefore, computes the total pair-wise hamming distance among the selected rows. The two constraints remain the same as Problem 1. This is an integer quadratic problem, also solved with Gurobi [23].

Table 2: Two optimisation problems for Smallset selection.

Problem 1 - Coverage	Problem 2 - Coverage + Variety
$\max_{\mathbf{z}}\ 1$	$\max_{\mathbf{z}}\ \mathbf{z}^\top Q \mathbf{z}$
s.t. $\displaystyle\sum_{i=1}^{N} z_i = K$	s.t. $\displaystyle\sum_{i=1}^{N} z_i = K$
$\displaystyle\sum_{i=1}^{N} z_i c_{ih} > 0,\ \forall h = 1, \ldots, H$	$\displaystyle\sum_{i=1}^{N} z_i c_{ih} > 0,\ \forall h = 1, \ldots, H$

Figure 4 illustrates the three different approaches for selecting $K = 5$ rows: random selection, selection that prioritises coverage, and selection that prioritises coverage and variety. We can see that random sampling misses a row affected by Step 1. While the solution from Problem 1 satisfies the constraints of covering Steps 1, 2, and 3, the first four selected rows underwent the same preprocessing steps. The solution from Problem 2, thanks to the *visual variety* criterion, selects three rows affected by Step 2 and 3 differently. Another desirable by-product of *visual variety* is having one row with minimal changes included (row 32).

5.2 Discussion

5.2.1 Novelty. The two proposed optimisation algorithms are similar in spirit to known combinatorial problems on sets [7], but the particular formulation incorporating preprocessing workflow and visual appearance criteria is new. Our Smallset selection algorithms are distinct from known subset selection approaches, as our objective functions are not submodular [52]. Our solution relies on auxiliary data generated from the preprocessing steps and does not need to cluster the input [9], noting clustering would require preprocessing having been completed.

5.2.2 Comment on running time. Despite being combinatorial optimisation problems, we obtained solutions for Problems 1 and 2 for the synthetic dataset in a few seconds. Problem 2 generates visually more desirable outputs, at the cost of needing to precompute and optimise with a distance matrix Q that is quadratic in the number of rows.

5.2.3 Potential problem variants. One may wonder whether this methodology could be used to select a subset of columns, e.g., 41 columns in the MDP CM1 dataset (Section 7.1.1) is clearly too
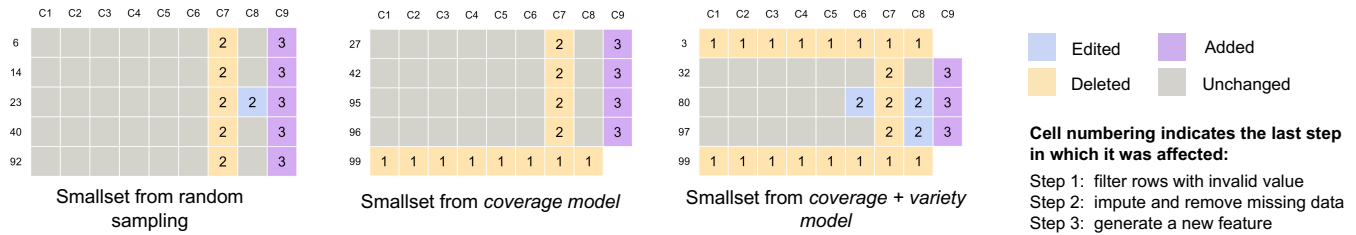
Figure 4: Smallsets selected by random sampling (left), coverage model (middle), and coverage+variety model (right) on the synthetic dataset (Appendix A). Shown here is one snapshot with accumulated changes (indicated by cell color) across the three processing steps (indicated by the numbering of the cell). Row numbers refer to those in the original dataset.

many for the visualisation. Indeed, one can envision formulating variants of the objective and constraints from the coverage and appearance matrices for each column. We leave this as future work. One may also wonder whether other objectives are needed, such as *representativeness* – for more selected rows to reflect changes that often occur, and vice versa. This is possible, e.g., by minimising the difference between the fraction of changes in sampled rows and those for the whole dataset in Problem 1. However, we note that computing reprensentativeness on a small subset is prone to statistical noise and exclude it from our primary criteria.

5.2.4 Do Smallset Timelines have to be small? We recommend that Smallsets be 5-15 rows and Timelines be 2-10 snapshots, due to the cognitive limits of the Timeline reader and the constraints of having readable visualisations within limited page or screen space. Multiple snapshots with a larger Smallset can be used to accommodate longer chains of operations that affect a large dataset in diverse ways. In these scenarios, the choice is left to Timeline creators to trade-off between a large visualisation with many details or a small one with fewer details.

6 THE SMALLSETS R PACKAGE

All Smallset Timelines presented in this work are made with the `smallsets` tool. Miceli et al. [33] provide qualitative evidence that suggests producing documentation often feels like a "burden" to data practitioners. We develop a tool that aims to alleviate the burden and in turn encourage production of data preprocessing documentation. To integrate with existing or new preprocessing workflows, the software requires two inputs from Timeline creators: adding structured comments to an R or Python preprocessing script and populating an R Markdown caption template generated by `smallsets`. Figure 5 contains a procedural overview of these inputs along with the `smallsets` processing tasks.

Producing a Smallset Timeline begins with a Timeline creator adding a series of `smallsets` comments to an R or Python pre-processing script (Figure 5 Step 1). Incorporating docstrings and comments to generate documentation is a common technique (e.g., [20, 53]). It is used here to assist in generating visual documentation of data preprocessing. The added comments provide instructions for `smallsets`, advising it where to take snapshots of the data or insert a resume marker (Section 4.2). Each comment consists of one of four actions – *start smallset*, *resume*, *end smallset*, or *snap* – and the variable storing the data frame (e.g., *# snap mydata*). In Step 2, the software prepares the Smallset, takes snapshots based on Step

1 input, analyses the snapshots for data changes, and generates a customised R Markdown caption template. Step 3 requires Timeline creators to populate this template with captions for the snapshots. The caption input is used by `smallsets` as it assembles the Timeline and produces the *alt text* (Section 4.3) in Step 4. Timeline creators can specify their preferences regarding Smallset properties – e.g., selection method (Section 5) and size – and Timeline design – e.g., colours, font, and enrichment features (Section 4.2) – in Steps 2 and 4, respectively.

We chose to do the initial implementation of Smallset Timelines in R because it is a popular programming language for preprocessing datasets and offers strong graphics capabilities. We have enabled the software to also accept scripts in Python, another popular preprocessing choice. Future software development work can include increasing the capacity of `smallsets` to manage more complex preprocessing workflows, involving multiple scripts and the merging and joining of datasets.

7 CASE STUDIES

We present two case studies to illustrate the use of Smallset Timelines. The first is on software defect detection data from the NASA Metrics Data Program (MDP). Despite being widely used to develop defect classification models, a lack of consistency in data preprocessing and documentation has jeopardised the utility of research outputs [21, 22, 40, 44]. The second case study examines benchmark datasets containing American Community Survey (ACS) data. We quantify differences in fairness metrics due to different preprocessing decisions, whereas recent work [13] has focused on differences in fairness metrics across fairness interventions and income thresholds.

7.1 NASA MDP data

In the early 2000s, the MDP released 13 datasets for software defect detection research. Like many real-world datasets, the data require preprocessing. There are missing, erroneous, extraneous, and duplicate data to address. We chose these data as a case study because of existing literature [21, 22, 40, 44] focused on assessing MDP preprocessing practices. For example, Gray et al. [22] note the issue of duplicate data occurring in the testing and training set (i.e., the model is not tested on unseen data). Their concern is that "the impression given from the literature is that many defect prediction researchers using this data have not been aware of this issue" [22, p. 557]. Shepperd et al. [44] highlight that studies

1. Comment preprocessing script

```
1   import data
2
3   # start smallset mydata
4   proc_operation_1(mydata)
5   proc_operation_2(mydata)
6   # snap mydata
7   proc_operation_3(mydata)
8   # snap mydata
9   proc_operation_4(mydata)
10  proc_operation_5(mydata)
11  # end smallset mydata
12
13  model data
```

2. Run `prepare_smallset`

- **prepares the Smallset**
- **takes snapshots**
- **finds snapshot changes**
- **generates a caption template**

3. Complete caption template

```
1   Timeline title:
2
3   Timeline subtitle:
4
5   ### Starting smallset
6
7   Caption:
8
9   ### proc_operation_3(mydata)
10
11  Caption:
12
13  ### proc_operation_4(mydata)
14
15  Caption:
16
17  ### Ending smallset
18
19  Caption:
```

4. Run `create_timeline`

- **builds a Smallset Timeline**
- **produces *alt text***

Figure 5: Steps for creating a Smallset Timeline using the `smallsets` R software. Steps 1 and 3 requires Timeline creator input of structured comments and captions, respectively. In Steps 2 and 4, `smallsets` takes data snapshots and builds the Timeline. Example `smallsets` comments and the R `Markdown` caption for the synthetic data are in Appendix A.7.

Step 1: Remove columns that have the same value for every row because they do not provide any information for modelling.

Step 2: Replace missing *DECISION_DENSITY* values with zero. Based on other MDP datasets without missing *DECISION_DENSITY* values, one can deduce that they likely occurred due to a division by zero error and can be replaced with zeros.

Step 3: Remove rows that are duplicates of other rows to assure models are tested on unseen data only. Also remove rows that are inconsistent, meaning all column values are the same except for the class label (one is classified as defective and the other is not).

The CM1 dataset is now ready for use in modelling. The full dataset and R preprocessing code are available on GitHub.

Data has not changed. Tint is missing data. | Data has been edited. Tint is missing data. | Data will be deleted. Tint is missing data.

Figure 6: Smallset Timeline for MDP CM1 dataset preprocessed according to [21]. See Section 7.1.1 for discussion. Smallset selected using Problem 1 algorithm (see Section 5).

deal with the data issues differently and are "not in the habit of providing complete information regarding preprocessing of data" [p. 1213]. The literature presents a clear example of insufficient documentation for preprocessing decisions, i.e., an example of the problem that Smallset Timelines are designed to address. In the rest of this section, we use the MDP CM1 dataset [48]. It has 505 rows and 41 columns. For the Smallset Timelines in Section 7.1.1 and Section 7.1.2, we choose to display 10 and 15 columns, respectively.

7.1.1 Smallset Timelines for comprehension.
Figure 6 contains a Smallset Timeline for the preprocessing strategy recommended by Gray et al. [21]. The Timeline uses a Smallset with six rows and consists of four snapshots. It discloses how the common MDP data issues have been dealt with. For instance, snapshot 2 discusses missing data. In some work, they are simply dropped [44]. However, this Timeline creator argues that, based on study of other MDP datasets, the missing values can be attributed to a division by zero error and retained by imputing zeros [21]. Snapshot 3 addresses the issue of duplicate data, noting that it is removed and why this is necessary. Figure 6 uses about the same amount of space as the

other figures in this work (which are not Smallset Timelines, e.g., Figure 1 or Figure 5) and remains legible.

7.1.2 Smallset Timelines for replication.
In this section, we restructure the Smallset Timeline presented in Figure 6 to better support replication efforts (Figure 7, Table 1). Shepperd et al. [44] suggest that practitioners using MDP data "report any preprocessing in sufficient detail to enable meaningful replication" [p. 1208]. For replication, the Timeline needs to be comprehensive and specific. We increase the Smallset size to include additional data examples and take more snapshots to separate the process into its component parts. As a result, the Timeline is larger and may be located in an appendix or with the preprocessing code. The captions contain specific information, including the total number of rows an operation affects and the rules for checking data integrity.

Data integrity checks are an important MDP preprocessing step that remove implausible values. As noted in caption 4 of Figure 7, the checks do not actually affect any rows in the CM1 dataset. The step was left out of Figure 6 for brevity, but it is included here for clarity. If replicating the preprocessing strategy on another dataset, it would be necessary for accuracy and consistency to conduct the data

Step 1: All constant attributes are removed from the dataset. These attributes do not offer any useful information when building the classifier. There are three constant attributes in the CM1 dataset: *GLOBAL_DATA_COMPLEXITY*, *GLOBAL_DATA_DENSITY*, and *PATHOLOGICAL_COMPLEXITY*.

Step 2: Remove repeated attributes. There are none in CM1.

Step 3: The *DECISION_DENSITY* attribute is the only attribute that contains missing values in CM1. This attribute is equal to the *CONDITION_COUNT* divided by the *DECISION_COUNT*. Missing values only occur in *DECISION_DENSITY* when both of these other attributes equal zero. In other MDP datasets, where both equal zero, so does *DECISION_DENSITY*. Thus, the missing values are replaced with zero. (161 rows affected)

Step 4: All instances which fail one or more of the data integrity checks are removed from the dataset. The integrity checks identify instances that cannot realistically happen. The following integrity checks are used: *NUM_OPERANDS + NUM_OPERATORS = HALSTEAD_LENGTH*; *CYCLOMATIC_COMPLEXITY <= NUM_OPERATORS + 1*; *CALL_PAIRS <= NUM_OPERATORS*. (0 rows affected)

Step 5: Remove repeated and inconsistent cases. Duplicates are dropped (49 rows affected). Inconsistent cases refer to cases in which all values but the class label are equal, and these are dropped as well (2 rows affected).

The CM1 dataset is preprocessed and ready to be used in the analysis. The full version of CM1 and the R preprocessing code are available on GitHub.

Data has not changed. Tint is missing data.
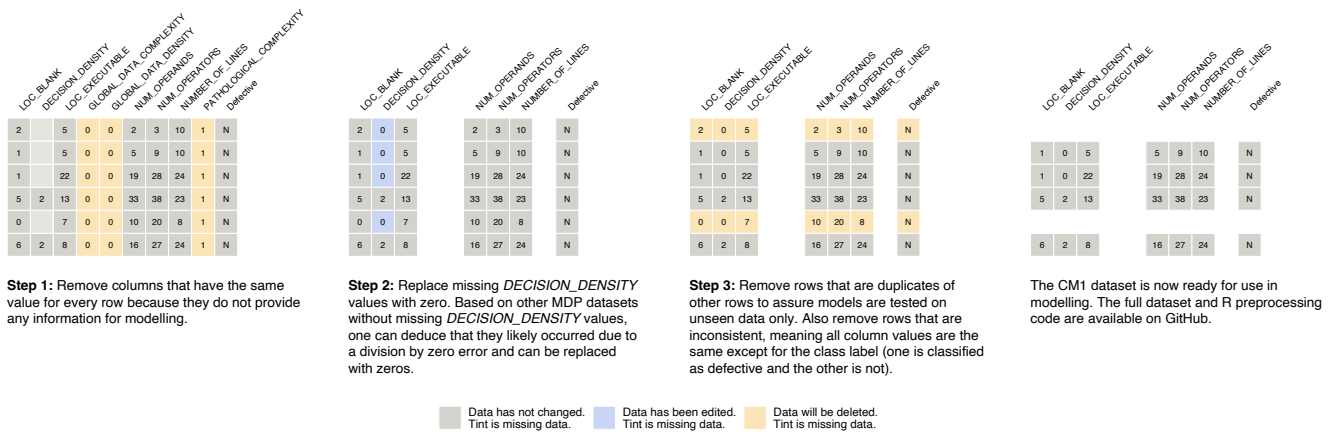Data has been edited. Tint is missing data.
Data will be deleted. Tint is missing data.

Figure 7: Smallset Timeline for MDP CM1 dataset, for replication. See Section 7.1.2 for discussion. Smallset selected using Problem 1 algorithm (see Section 5).

checks. It is worth noting that the preprocessing strategy proposed in Shepperd et al. [44] suggests running 18 different integrity checks, while Petrić et al. [40] suggest 20 different integrity checks. The additional checks do result in the loss of observations in CM1. In other words, indicating that "the data checks were run" is not enough information. Replication will require greater specification.

7.2 The folktables data

The UCI Adult dataset [30] consists of 1994 census income data, and the associated estimation task is to predict if an individual earns more than 50,000 dollars per year. It has been used in hundreds of research papers related to machine learning fairness. A recent paper by Ding et al. [13] challenges the machine learning community's continued reliance on the dataset, given its age and defects. For example, the 50,000 dollar threshold leads to imbalance by race and gender in the dataset as it represents the "88th quantile in the Black population, and the 89th quantile among women" [13, p. 2]. In turn, they develop a tool, folktables, to generate recent benchmark datasets from the American Community Survey (ACS) and define

new prediction tasks. It allows adjustable income thresholds and data filtering criteria. We explore effects of these preprocessing decisions on 2015 ACS income data from California (CA), Connecticut (CT), and Utah (UT), retrieved with folktables.

We explore four different preprocessing scenarios, starting with the default setting used by Ding et al. [13] – referred to here as *default-50K*. In this setting, an income threshold of $50K is applied to generate positive and negative labels, after filtering the dataset to retain an individual's record when they 1) are older than 16 years of age, 2) have a survey weight of at least one, 3) earn more than 100 dollars, and 4) report at least one hour of usual weekly work. The next setting, called *default-median*, uses the same set of default data filters but sets the income threshold to the sample median after filtering ($36K, $45K, and $31.1K for CA, CT, and UT, respectively) to generate more balanced prediction tasks. The remaining settings aim to be inclusive of all the target population, on the grounds that individuals who did not work and/or reported income losses are still valid instances for prediction, by dropping the last two filters. We refer to this filtering approach as "validity." The setting

We first filter to individuals older than 16 years of age (*AGEP*) and with survey weights (*PWGTP*) of at least one. It is common to also filter these data by reported income (*PINCP* > 100) and working hours (*WKHP* >= 1), but we choose to omit these filters.

We replace missing values for "weekly hours worked" (*WKHP*) with zero and for class of worker (*COW*) and occupation (*OCCP*) with -1, i.e., create categories for "no class" and "no occupation." The median income of the dataset (after filtering), $22.5K, is used as the threshold to generate the class label (*INCOME*).

■ Data has not changed. ■ Data has been edited. ■ Data has been added. ■ Data will be deleted.

Figure 8: Smallset Timeline of ACS California data preprocessed with the *validity-median* setting. Smallset selected with random sampling. See Section 7.2 for discussion and Appendix C.1 for the Python preprocessing script behind this Timeline.
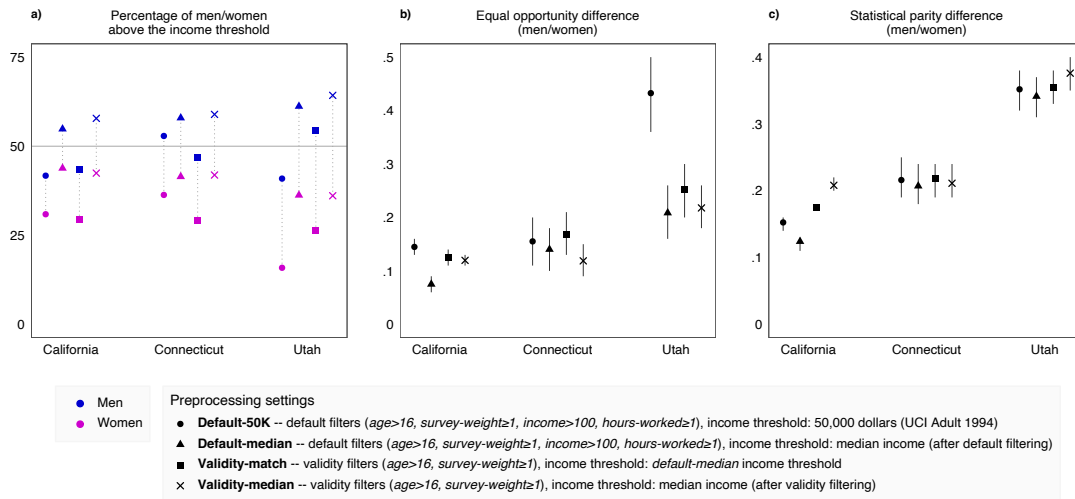
Figure 9: The effect of four different preprocessing settings on data and prediction. (a) Dataset imbalance by sex. (b) and (c) Group fairness measures in predictions, error bars refer to 95% Newcombe intervals. See Section 7.2 for discussions.

validity-match uses the same threshold as *default-median*, such that the thresholds "match" (e.g., $36K for CA) despite different filters. Lastly, the *validity-median* setting uses its own sample median after validity filtering ($22.5K, $30.2K, and $23.5K for CA, CT, and UT, respectively). Figure 8 is an example Smallset Timeline depicting the *validity-median* steps applied to the California dataset. Additional dataset statistics are available in Appendix C.2.

Figure 9 presents results across preprocessing settings on class imbalance in gender[2] and fairness levels in classification results. Figure 9(a) compares the percentage of men and women above the income threshold. It shows that certain income thresholds achieve greater balance than others, e.g., for California, *default-median* is more balanced among groups (aiming for equal splits overall)

than *default-50K*. However, using the same threshold alone will not guarantee balance or consistency across studies that use different preprocessing filters. For example, comparing *default-median* and *validity-match* for California, which have matching thresholds ($36K) but different filters, we see a substantial change in the percentage of women above the income threshold (43.8% and 29.4% for *default-median* and *validity-match*, respectively). Thus, it is necessary to communicate both filtering decisions and the threshold selection.

With `folktables`, we define prediction tasks that correspond with *default-50K*, *default-median*, *validity-match*, and *validity-median*. All tasks predict if an individual's income is over the threshold. For each task and state, we train and test a logistic regression model (with `scikit-learn` [39] default settings) on 80% and 20% of the

_____
[2]In the original dataset, the attribute corresponds with a male/female encoding and does not include nonbinary gender options.

dataset, respectively. For men and women in the dataset, we compute differences of equality of opportunity (EO) [24] and statistical parity (SP) [14] from the test set predictions and 95% Newcombe intervals [35] for the differences. Figure 9(b) shows a significant difference in the EO between *default-50K* and the other three settings for Utah, but variations among different settings in Connecticut are much smaller. Figure 9(c) shows that, across the four settings, the SP values are significantly different for California but not for Connecticut or Utah.

8 CONCLUSION

We present the Smallset Timeline, a visualisation of data preprocessing decisions. It is designed to support reflection by Timeline creators and replication, comprehension, and evaluation by Timeline readers. Its static, compact nature makes it a practical figure to include in research outputs. We develop the `smallsets` tool, an R software package for producing Smallset Timelines from R and Python scripts. Timeline creators only need to add a few structured comments to the preprocessing script and supply captions. We also present case studies on software defect data and income survey benchmark data, highlighting the importance of communicating decisions that affect the dataset and prediction outcomes. We include several Smallset Timelines to illustrate use of the visualisation and software tool.

Future work involves new features, visual design, and software development. Examples include: to incorporate data statistics in diagrams alongside the Smallset snapshots, to support *comparison* between different preprocessing decisions, and to design succinct visualisations for complex workflows such as dataset joins and a richer set of data formats. We are also interested in techniques for assuring data privacy in a Smallset, visual representation for specific preprocessing tasks, and new applications. Lastly, it will be great to potentially incorporate Smallset Timelines within other data science provenance tools. For instance, a Smallset Timeline could be included as part of Question 33 in datasheets [19], in the *Evaluation data* section of model cards [34], and the dataset composition section of Dataset Nutrition Labels [25].

ACKNOWLEDGMENTS

REFERENCES

[1] Matthew Arnold, Rachel KE Bellamy, Michael Hind, Stephanie Houde, Sameep Mehta, Aleksandra Mojsilović, Ravi Nair, K Natesan Ramamurthy, Alexandra Olteanu, David Piorkowski, et al. 2019. FactSheets: Increasing trust in AI services through supplier's declarations of conformity. *IBM Journal of Research and Development* 63, 4/5 (2019), 6–1.

[2] Emily M Bender and Batya Friedman. 2018. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics* 6 (2018), 587–604.

[3] Alexander W Blocker and Xiao-Li Meng. 2013. The potential and perils of preprocessing: Building new foundations. *Bernoulli* 19, 4 (2013), 1176–1211.

[4] Christian Bors, Theresia Gschwandtner, and Silvia Miksch. 2019. Capturing and visualizing provenance from data wrangling. *IEEE computer graphics and applications* 39, 6 (2019), 61–75.

[5] Steven P Callahan, Juliana Freire, Emanuele Santos, Carlos E Scheidegger, Cláudio T Silva, and Huy T Vo. 2006. VisTrails: visualization meets data management. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. 745–747.

[6] Scott Chamberlain and Kyle Voytovich. 2020. *charlatan: Make Fake Data*. https://CRAN.R-project.org/package=charlatan R package version 0.4.0.

[7] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.

[8] Yingwei Cui and Jennifer Widom. 2000. Practical lineage tracing in data warehouses. In *Proceedings of 16th International Conference on Data Engineering (Cat. No. 00CB37073)*. IEEE, 367–378.

[9] Michal Daszykowski, Beata Walczak, and DL Massart. 2002. Representative subset selection. *Analytica chimica acta* 468, 1 (2002), 91–103.

[10] Jenny L Davis. 2020. *How artifacts afford: The power and politics of everyday things*. MIT Press.

[11] Matthew J Denny and Arthur Spirling. 2018. Text preprocessing for unsupervised learning: Why it matters, when it misleads, and what to do about it. *Political Analysis* 26, 2 (2018), 168–189.

[12] Catherine D'Ignazio and Lauren F Klein. 2020. *Data feminism*. Mit Press.

[13] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. 2021. Retiring Adult: New Datasets for Fair Machine Learning. *Advances in Neural Information Processing Systems* 34 (2021).

[14] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*. 214–226.

[15] Madeleine Clare Elish and Danah Boyd. 2018. Situating methods in the magic of Big Data and AI. *Communication monographs* 85, 1 (2018), 57–80.

[16] A Famili, Wei-Min Shen, Richard Weber, and Evangelos Simoudis. 1997. Data preprocessing and intelligent data analysis. *Intelligent data analysis* 1, 1 (1997), 3–23.

[17] Sorelle A Friedler, Carlos Scheidegger, Suresh Venkatasubramanian, Sonam Choudhary, Evan P Hamilton, and Derek Roth. 2019. A comparative study of fairness-enhancing interventions in machine learning. In *Proceedings of the conference on fairness, accountability, and transparency*. 329–338.

[18] Salvador García, Julián Luengo, and Francisco Herrera. 2015. *Data preprocessing in data mining*. Vol. 72. Springer.

[19] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. 2021. Datasheets for datasets. *Commun. ACM* 64, 12 (2021), 86–92.

[20] David Goodger. 2001. *PEP 257 – Docstring Conventions*. Retrieved December 28, 2021 from https://www.python.org/dev/peps/pep-0257/

[21] David Gray, David Bowes, Neil Davey, Yi Sun, and Bruce Christianson. 2011. The misuse of the NASA metrics data program data sets for automated software defect prediction. In *15th Annual Conference on Evaluation & Assessment in Software Engineering (EASE 2011)*. IET, 96–103.

[22] David Gray, David Bowes, Neil Davey, Yi Sun, and Bruce Christianson. 2012. Reflections on the NASA MDP data sets. *IET software* 6, 6 (2012), 549–558.

[23] Gurobi Optimization, LLC. 2022. Gurobi Optimizer Reference Manual. https://www.gurobi.com

[24] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems* 29 (2016).

[25] Sarah Holland, Ahmed Hosny, Sarah Newman, Joshua Joseph, and Kasia Chmielinski. 2018. The dataset nutrition label: A framework to drive higher data quality standards. *arXiv preprint arXiv:1805.03677* (2018).

[26] Ben Hutchinson, Andrew Smart, Alex Hanna, Emily Denton, Christina Greer, Oddur Kjartansson, Parker Barnes, and Margaret Mitchell. 2021. Towards accountability for machine learning datasets: Practices from software engineering and infrastructure. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. 560–575.

[27] Eun Seo Jo and Timnit Gebru. 2020. Lessons from archives: Strategies for collecting sociocultural data in machine learning. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 306–316.

[28] Stephen Kasica, Charles Berret, and Tamara Munzner. 2020. Table Scraps: An Actionable Framework for Multi-Table Data Wrangling From An Artifact Study of Computational Journalism. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 957–966.

[29] Meraj Khan, Larry Xu, Arnab Nandi, and Joseph M Hellerstein. 2017. Data tweening: incremental visualization of data transforms. *Proceedings of the VLDB Endowment* 10, 6 (2017), 661–672.

[30] R. Kohavi and B. Becker. 1996. UCI Adult data set. https://archive.ics.uci.edu/ml/datasets/adult

[31] Julián Luengo, Diego García-Gil, Sergio Ramírez-Gallego, Salvador García, and Francisco Herrera. 2020. *Big data preprocessing: enabling smart data*. Springer Nature.

[32] Xiao-Li Meng. 2021. Enhancing (publications on) data quality: Deeper data minding and fuller data confession. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* (2021).

[33] Milagros Miceli, Tianling Yang, Laurens Naudts, Martin Schuessler, Diana Serbanescu, and Alex Hanna. 2021. Documenting Computer Vision Datasets: An Invitation to Reflexive Data Practices. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. 161–172.

[34] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proceedings of the conference on fairness, accountability, and transparency*. 220–229.

[35] Robert G Newcombe. 1998. Two-sided confidence intervals for the single proportion: comparison of seven methods. *Statistics in medicine* 17, 8 (1998), 857–872.

[36] Christina Niederer, Holger Stitz, Reem Hourieh, Florian Grassinger, Wolfgang Aigner, and Marc Streit. 2017. TACO: visualizing changes in tables over time. *IEEE transactions on visualization and computer graphics* 24, 1 (2017), 677–686.

[37] Brian A Nosek, Tom E Hardwicke, Hannah Moshontz, Aurélien Allard, Katherine S Corker, Anna Dreber Almenberg, Fiona Fidler, Joseph Hilgard, Melissa Kline, Michèle B Nuijten, et al. 2021. Replicability, robustness, and reproducibility in psychological science. (2021).

[38] Samir Passi and Steven Jackson. 2017. Data vision: Learning to see through algorithmic abstraction. In *Proceedings of the 2017 ACM conference on computer supported cooperative work and social computing*. 2436–2447.

[39] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[40] Jean Petrić, David Bowes, Tracy Hall, Bruce Christianson, and Nathan Baddoo. 2016. The jinx on the NASA software defect data sets. In *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering*. 1–5.

[41] Xiaoying Pu, Sean Kross, Jake M Hofman, and Daniel G Goldstein. 2021. Datamations: Animated Explanations of Data Analysis Pipelines. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.

[42] RStudio. 2021. *Data transformation with dplyr::cheat sheet*. Retrieved December 29, 2021 from https://github.com/rstudio/cheatsheets/blob/main/data-transformation.pdf

[43] Morgan Klaus Scheuerman, Alex Hanna, and Emily Denton. 2021. Do datasets have politics? disciplinary values in computer vision dataset development. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW2 (2021), 1–37.

[44] Martin Shepperd, Qinbao Song, Zhongbin Sun, and Carolyn Mair. 2013. Data quality: Some comments on the nasa software defect datasets. *IEEE Transactions on Software Engineering* 39, 9 (2013), 1208–1215.

[45] Steven D Shirk, Donald G McLaren, Jessica S Bloomfield, Alex Powers, Alec Duffy, Meghan B Mitchell, Ali Ezzati, Brandon A Ally, and Alireza Atri. 2017. Inter-rater reliability of preprocessing EEG data: Impact of subjective artifact removal on associative memory task ERP results. *Frontiers in neuroscience* 11 (2017), 322.

[46] Sara Steegen, Francis Tuerlinckx, Andrew Gelman, and Wolf Vanpaemel. 2016. Increasing transparency through a multiverse analysis. *Perspectives on Psychological Science* 11, 5 (2016), 702–712.

[47] Deborah F Swayne and Andreas Buja. 1998. Missing data in interactive high-dimensional data visualization. *Computational Statistics* 13, 1 (1998), 15–26.

[48] Chakkrit Tantithamthavorn. 2016. NASADefectDataset. https://github.com/klainfo/NASADefectDataset.

[49] Anissa Tanweer, Emily Kalah Gade, PM Krafft, and Sarah K Dreier. 2021. Why the Data Revolution Needs Qualitative Thinking. *Harvard Data Science Review* (2021).

[50] Nicholas Tierney, Di Cook, Miles McBain, and Colin Fay. 2021. *naniar: Data Structures, Summaries, and Visualisations for Missing Data*. https://CRAN.R-project.org/package=naniar R package version 0.6.1.

[51] Zezhong Wang, Jacob Ritchie, Jingtao Zhou, Fanny Chevalier, and Benjamin Bach. 2021. Data Comics for Reporting Controlled User Studies in Human-Computer Interaction. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 967–977. https://doi.org/10.1109/TVCG.2020.3030433

[52] Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2015. Submodularity in data subset selection and active learning. In *International Conference on Machine Learning*. PMLR, 1954–1963.

[53] Hadley Wickham, Peter Danenberg, Gábor Csárdi, and Manuel Eugster. 2020. *roxygen2: In-Line Documentation for R*. https://CRAN.R-project.org/package=roxygen2 R package version 7.1.1.

[54] Ke Yang, Biao Huang, Julia Stoyanovich, and Sebastian Schelter. 2020. Fairness-Aware Instrumentation of Preprocessing~ Pipelines for Machine Learning. In *Workshop on Human-In-the-Loop Data Analytics (HILDA'20)*.

[55] Zhenpeng Zhao, Rachael Marr, and Niklas Elmqvist. 2015. Data comics: Sequential art for data-driven storytelling. *tech. report* (2015).

A SYNTHETIC DATASET

For illustrative purposes, we generate a synthetic dataset and preprocessing scenario. This section describes its basic profile (Appendix A.1), the three preprocessing steps it undergoes (Appendix A.2), a Smallset Timeline for these steps (Appendix A.3), another Timeline with a resume marker (Appendix A.4), *alt text* for the Smallset Timeline in Appendix A.3 (Appendix A.5), the data matrices for the Smallset selection algorithms (Appendix A.6), and the R preprocessing script with structured comments and populated caption template passed to the `smallsets` software (Appendix A.7).

A.1 Dataset

The dataset is synthesised with the `charlatan` software package [6] in R. The initial dataset consists of 100 rows and 8 features. The features are described in Table 3, and the first ten rows of the dataset are printed in Figure 10.

Name	Type	Missing Values
C1	Categorical	No
C2	Binary	No
C3	Discrete	No
C4	Discrete	No
C5	Continuous	No
C6	Continuous	Yes (14%)
C7	Continuous	Yes (44%)
C8	Continuous	Yes (19%)

Table 3: Features descriptions for the synthetic dataset.

```
C1    C2 C3  C4   C5    C6     C7    C8
 2  TRUE 33 199 5.44    NA   0.99    NA
 3  TRUE 31 161 4.84  6.74   1.24  1.22
 4 FALSE 33 188 5.97  9.35  -0.04  0.59
 2 FALSE 24 129 4.33  7.80    NA    NA
 2  TRUE 32 115 6.64 10.64    NA    NA
 3  TRUE 22 101 3.93  7.18  -1.12 -0.63
 3 FALSE 38 131 4.99  8.14  -0.84 -2.12
 2  TRUE 30 183 3.13 10.73    NA   0.25
 4  TRUE 34 159 5.02 11.12    NA   0.13
 3  TRUE 36 157 4.75 11.88    NA  -0.98
```

Figure 10: First ten rows of the synthetic dataset.

A.2 Preprocessing

The data preprocessing for the synthetic dataset consists of three main steps.

(1) Filter rows
   - Remove rows where C2 is FALSE
(2) Deal with missing data
   - Replace missing values in C6 and C8 with mean values by C1 category
   - Drop C7
(3) Generate a new feature
   - Create C9 by summing C3 and C4

A.3   Smallset Timeline

Figure 11 is a Smallset Timeline for the synthetic data example.

C1	C2	C3	C4	C5	C6	C7	C8
4	FALSE	33	188	5.97	9.35	-0.04	0.59
1	TRUE	23	191	4.64	9.48		-0.76
3	TRUE	22	176	5.3		-0.21	
5	TRUE	25	156	5.66	10.5	0.45	
4	FALSE	24	191	5.28	12.53	-0.57	0.96

Remove rows where C2 is FALSE.

C1	C2	C3	C4	C5	C6	C7	C8
1	TRUE	23	191	4.64	9.48		-0.76
3	TRUE	22	176	5.3	10.6	-0.21	-0.41
5	TRUE	25	156	5.66	10.5	0.45	-0.55

Replace missing values in C6 and C8 with category (C1) means. Drop C7 because it has too many missing values.

C1	C2	C3	C4	C5	C6	C8	C9
1	TRUE	23	191	4.64	9.48	-0.76	214
3	TRUE	22	176	5.3	10.6	-0.41	198
5	TRUE	25	156	5.66	10.5	-0.55	181

Create a new column, C9, by summing C3 and C4.

Data has not changed. Tint is missing data.
Data has been edited. Tint is missing data.
Data has been added. Tint is missing data.
Data will be deleted. Tint is missing data.

Figure 11: Smallset Timeline for the synthetic dataset and the preprocessing scenario. Smallset selected using Problem 2 algorithm.

A.4   Resume markers

Figure 12 includes an additional step (generation of another feature) in the Timeline to illustrate use of the **resume markers** enrichment feature.

C1	C2	C3	C4	C5	C6	C7	C8
4	FALSE	33	188	5.97	9.35	-0.04	0.59
1	TRUE	23	191	4.64	9.48		-0.76
3	TRUE	22	176	5.3		-0.21	
5	TRUE	25	156	5.66	10.5	0.45	
4	FALSE	24	191	5.28	12.53	-0.57	0.96

Remove rows where C2 is FALSE.

C1	C2	C3	C4	C5	C6	C7	C8
1	TRUE	23	191	4.64	9.48		-0.76
3	TRUE	22	176	5.3	10.6	-0.21	-0.41
5	TRUE	25	156	5.66	10.5	0.45	-0.55

Replace missing values in C6 and C8 with category (C1) means. Drop C7 because it has too many missing values.

C1	C2	C3	C4	C5	C6	C8	C9
1	TRUE	23	191	4.64	9.48	-0.76	214
3	TRUE	22	176	5.3	10.6	-0.41	198
5	TRUE	25	156	5.66	10.5	-0.55	181

Create a new column, C9, by summing C3 and C4.

C1	C2	C3	C4	C5	C6	C8	C9	C10
1	TRUE	23	191	4.64	9.48	-0.76	214	High
3	TRUE	22	176	5.3	10.6	-0.41	198	Med
5	TRUE	25	156	5.66	10.5	-0.55	181	Med

Create a new categorical variable, C10, based on C9 terciles.

Data has not changed. Tint is missing data.
Data has been edited. Tint is missing data.
Data has been added. Tint is missing data.
Data will be deleted. Tint is missing data.

Figure 12: Smallset Timeline with a resume marker.

A.5 *Alt text*

```
automated alt text output from
smallset software
```

→

```
manually edited for
dissemination
```

● ● ●   📄 **figureAltText.txt**

```
The Smallset Timeline contains 3 Smallset
snapshots. Data edits are represented with the
colour lightsteelblue2. Data additions are
represented with the colour plum. Data
deletions are represented with the colour
moccasin.
 Snapshot 1 is 5 rows by 8 columns. The
columns, in order from left to right, are: C1,
C2, C3, C4, C5, C6, C7, and C8. Two rows are
moccasin. The caption is quote "Remove rows
where C2 is FALSE."
 Snapshot 2 is 3 rows by 8 columns. Column C7
is moccasin. Three table cells are
lightsteelblue2. The caption is quote "Replace
missing values in C6 and C8 with category (C1)
means. Drop C7 because it has too many missing
values."
 Snapshot 3 is 3 rows by 8 columns. Column C9
is plum. The caption is quote "Create a new
column, C9, by summing C3 and C4."
```

The Smallset Timeline contains 3 Smallset snapshots. Data edits are represented with the colour blue. Data additions are represented with the colour purple, and data deletions are represented with the colour yellow.

The first Timeline snapshot is 5 rows by 8 columns. The column names, in order from left to right, are: C1, C2, C3, C4, C5, C6, C7, and C8. Two rows are yellow. The caption is quote "Remove rows where C2 is FALSE."

Snapshot 2 is 3 rows by 8 columns. Column C7 is yellow, and three table cells are blue. The caption is quote "Replace missing values in C6 and C8 with category (C1) means. Drop C7 because it has too many missing values."

Snapshot 3 is 3 rows by 8 columns. Column C9 is purple. The caption is quote "Create a new column, C9, by summing C3 and C4."

Figure 13: Example of automated *alt text* generated by the `smallsets` **software (left) and a manually edited version of it prepared for dissemination (right). The *alt text* is a description of the Smallset Timeline in Figure 11.**

A.6 Data representations for Smallset selection

	*Step*1	*Step*2	*Step*3
1	0	1	1
2	0	1	1
3	1	0	0
4	1	0	0
5	0	1	1
⋮	⋮	⋮	⋮
98	0	1	1
99	1	0	0
100	0	1	1

	C1	*C2*	*C3*	*C4*	*C5*	*C6*	*C7*	*C8*	*C9*
1	U	U	U	U	U	E	D	E	A
2	U	U	U	U	U	U	D	U	A
3	D	D	D	D	D	D	D	D	D
4	D	D	D	D	D	D	D	D	D
5	U	U	U	U	U	U	D	E	A
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
98	U	U	U	U	U	U	D	U	A
99	D	D	D	D	D	D	D	D	D
100	U	U	U	U	U	U	D	U	A

Figure 14: *Coverage indicator matrix* (left) and *visual appearance matrix* (right) for the synthetic data example. The letters in the *visual appearance matrix* represent the change last affecting a cell (*U*: unchanged, *E*: edit, *A*: addition, *D*: deletion).

A.7 `smallsets` software input for synthetic data

```
1   # start smallset df
2   df <- df[df$C2 == TRUE, ]
3
4   mC6 <- tapply(df$C6, df$C1, function(x) mean(x, na.rm = TRUE))
5   C6sub <- as.factor(df$C1)
6   levels(C6sub) <- mC6
7   df$C6[is.na(df$C6)] <- round(as.numeric(levels(C6sub))[C6sub][is.na(df$C6)], 2)
8
9   mC8 <- tapply(df$C8, df$C1, function(x) mean(x, na.rm = TRUE))
10  C8sub <- as.factor(df$C1)
11  levels(C8sub) <- mC8
12  # snap df
13  df$C8[is.na(df$C8)] <- round(as.numeric(levels(C8sub))[C8sub][is.na(df$C8)], 2)
14
15  df$C7 <- NULL
16
17  df$C9 <- df$C3 + df$C4
18  # end smallset df
```

Figure 15: Structured `smallsets` comments in the R preprocessing script for the synthetic dataset and Smallset Timeline in Figure 11.

```
---
title: "Captions for the Smallset Timeline"
author:
date:
output: html_document
---

Timeline title:

Timeline subtitle:

### Starting smallset

Caption: Remove rows where C2 is FALSE.

### `df$C8[is.na(df$C8)] <- round(as.numeric(levels(C8sub))[C8sub][is.na(df$C8)], 2)`

Caption: Replace missing values in C6 and C8 with category (C1) means. Drop C7 because
it has too many missing values.

### Ending smallset

Caption: Create a new column, C9, by summing C3 and C4.
```

Figure 16: Completed R Markdown caption template for the synthetic dataset and Smallset Timeline in Figure 11.

B ALT TEXT TEMPLATE

This Smallset timeline is titled _____ and subtitled _____.
        title   subtitle

The Smallset timeline contains _____ Smallset snapshots.
                number of snapshots

Data edits are represented with the colour _____.
                        colour name

Data additions are represented with the colour _____.
                            colour name

Data deletions are represented with the colour _____.
                            colour name

Snapshot 1 is _____ rows by _____ columns. The columns, in
        number of rows   number of columns

order from left to right, are: _____. *[Describe dataset changes*
                column names

here]. The caption is quote "_____."
                caption

Snapshot 2 is _____ rows by _____ columns. *[Describe dataset*
        number of rows   number of columns

changes here]. The caption is quote "_____."
                            caption

[Repeat for remaining snapshots].

Figure 17: *Alt text* template for Smallset Timelines.

C FOLKTABLES DATA

C.1 Preprocessing script

```python
# start smallset ca_data
ca_data = ca_data[ca_data["AGEP"] > 16]
ca_data = ca_data[ca_data["PWGTP"] >= 1]
ca_data["WKHP"] = ca_data["WKHP"].fillna(0)
ca_data["COW"] = ca_data["COW"].fillna(-1)
ca_data["OCCP"] = ca_data["OCCP"].fillna(-1)
ca_data["INCOME"] = (ca_data["PINCP"] > 22500).astype(int)
# end smallset ca_data
```

Figure 18: Python preprocessing script for the Smallset Timeline in Figure 8. No intermediary snapshot points are specified with a "# snap ca_data" comment, resulting in a two-snapshot Timeline. This script does not mirror the exact folktables preprocessing workflow but does execute the same preprocessing operations and demonstrates the capacity of smallsets to accept Python scripts. Future work includes increasing the capacity of smallsets to handle different workflow styles, such as the one used in folktables, which includes calling/called functions.

C.2 Dataset information

Table 4: Sample sizes and male/female counts for state datasets, before and after data filtering.

	Before filtering			After default filtering			After validity filtering		
	Total (n)	Males	Females	Total (n)	Males	Females	Total (n)	Males	Females
California	374,943	184,637	190,306	187,475	99,518	87,957	299,619	146,131	153,488
Connecticut	35,787	17,270	18,517	19,398	9,926	9,472	29,232	13,868	15,364
Utah	29,290	14,614	14,676	14,868	8,174	6,694	21,235	10,439	10,796