Predictive Data Transformation Suggestions in Grafterizer Using Machine Learning

Saliha Sajid[1,2], Bjørn Marius von Zernichow[2], Ahmet Soylu[2], and Dumitru Roman[1,2]

[1] University of Oslo, Norway
salihasa@student.matnat.uio.no, dumitrur@uio.no
[2] SINTEF AS, Norway
{*firstname.lastname*}@sintef.no

Abstract. Data preprocessing is a crucial step in data analysis. A substantial amount of time is spent on data transformation tasks such as data formatting, modification, extraction, and enrichment, typically making it more convenient for users to work with systems that can recommend most relevant transformations for a given dataset. In this paper, we propose an approach for generating relevant data transformation suggestions for tabular data preprocessing using machine learning (specifically, the Random Forest algorithm). The approach is implemented for Grafterizer, a Web-based framework for tabular data cleaning and transformation, and evaluated through a usability study.

Keywords: Data preprocessing · Data transformation · Transformation suggestions.

1 Introduction

With the increasing amount of data being generated every day, organizations rely on reliable data quality to ensure that data analysis is done accurately and without any predispositions. Ideally it is preferred to perform analysis on clean data which is free of irrelevant values, but in real life, that kind of data is seldom available [11]. For data analysis, data scientists need to preprocess data to ensure that the data is in the right format and conforms to a certain set of rules. Through a series of interviews with data professionals, it was revealed that a majority of their time is spent on time-consuming data transformation tasks in the data preprocessing phase [8].

Data transformation, as part of preprocessing phase, plays a critical role in ensuring data quality before analysis [6]. Data transformation is a domain specific problem that focuses on the statistical properties, semantics, and structure of data and typically domain experts have the knowledge required to apply the right transformations on data. Several commercial tools and frameworks exist for data preprocessing, offering a large number of data cleaning and transformation actions. In addition to commercial tools, most common frameworks and languages for data analysis such as Pandas[3], scikit-learn[4] and R[5] also come with several useful methods for data preprocessing tasks. However, it can be challenging for data scientists to choose from a large number of transformations and interactively view the changes made to the dataset. This time- and cost-consuming process could be made more efficient by automatically recommending users suitable data cleaning and transformation actions in an interactive graphical user interface (GUI).

In this paper, we propose an approach for the generation of relevant data transformation suggestions for tabular data preprocessing. Our approach is based on providing user interactions as input to a recommender system built using machine learning (ML) techniques (more specifically, the Random Forest algorithm). Random Forest [1], an ensemble learning method for classification, constructs a number of decision trees and provides output by aggregating the predictions of the ensemble. The ability of the Random Forest to formulate rules and predict output by going through the characteristics of training data motivates the work in this paper to investigate its benefits for the problem at hand. Our approach offers a GUI providing users with the most relevant data transformation suggestions and enabling users to transform data by choosing one of the suggested transformations. Our proposed approach was implemented for Grafterizer [16] – a tabular data transformation and Linked Data generation tool, developed as part of the DataGraft platform [15,14], and evaluated through a usability study.

The rest of the paper is structured as follows. Section 2 provides background knowledge on data transformation, while Section 3 presents the related work. Section 4 describes our solution approach and Section 5 reports on its evaluation findings. Finally, Section 6 concludes the paper.

2 Tabular Data Transformations

Data transformation is a process of changing the format or structure of the data. It may be done for tasks such as extracting meaningful knowledge, enrichment, or fixing incorrect data to prepare it for analysis.

We categorize tabular transformations as *table-based*, *format-based*, and *string-based*. The set of transformations chosen for the purpose of this paper represents a subset of a potentially large number of transformations that can be applied to a tabular dataset. This subset of transformations was selected based on various relevant sources (*e.g.*, [12,16] and Trifacta[6]), and consists of basic and intuitive transformations that may not require any help from an expert. These transformations can be made in a short number of steps, making it possible for the user to see the result when performed on a data object (*i.e.*, row, cell, and column).

[3] https://pandas.pydata.org

[4] https://scikit-learn.org

[5] https://www.r-project.org

[6] https://www.trifacta.com

Table 1. Table-based transformations.

Scope	**Name**	**Description**
Row	Convert row to header	Convert the selected row to header
Row	Drop row	Drop the selected row
Row	Keep row	Keep the selected row
Column	Delete column	Delete the selected column
Column	Keep column	Keep the selected column
Column	Rename column	Rename the selected column

Table-based transformations transform a dataset on the structural level. These transformations do not individually impact the data in rows and columns, but instead they change the formation of the dataset. They include feature extraction methods such as removing excess rows and columns, which may be unnecessary for analysis. Table 1 lists the selected table-based transformations.

Format-based transformations change the format of each individual cell in the selected data object without having much impact on the structure of the entire dataset. For instance, "count data by group" adds a column to the dataset containing the number of corresponding values existing in the source column. Table 2 for lists the selected format-based transformations.

Table 2. Format-based transformations.

Scope	**Name**	**Description**
Column	Normalise	Normalise numeric values
Column	Count data by group	Generate a new column with summed count of unique values of selected column
Column	Format date	Convert date to the given format
Column	To uppercase	Convert data to uppercase
Column	To lowercase	Convert data to lowercase
Cell	Round to nearest	Round the selected number to nearest integer

String-based transformations are applied to strings or text in the selected data object. This group of transformations can be used on a cell, column, and a row. It includes methods, such as filtering and modifications, applied to the individual strings in the dataset. Table 3 lists the selected string-based transformations.

3 Related Work

There is a significant amount of research done in the area of data transformation, including several commercial tools and libraries that can be used both programmatically and visually through GUIs. In what follows, we discuss the most notable ones.

Table 3. String-based transformations.

Scope	**Name**	**Description**
Row	Set row to *null*	Set the whole selected row to *null*
Column	Set column to *null*	Set the whole selected column to *null*
Column	Fill	Fill *nulls* in the column with the given input
Column	Split column	Split column using custom separator
Cell	Set cell to *null*	Set the value of selected cell to *null*
Cell	Extract	Extract values matching the selected text into a new column
Cell	Replace	Replace values matching the selected text with the given input in the corresponding column
Cell	Remove special characters	Remove special characters from the corresponding column

Tableau[7] is an interactive data analysis and visualisation platform that allows users to view data in understandable format and helps in generating customized dashboards. In addition to identifying problems in the data, Tableau analyses the given data to recommend transformations that may be of interest to the user. In also provides visual data profiling and a graphical list of steps taken to transform data in the form of a flowchart. Talend Data Preparation[8] is a tool that comes with a user-friendly interface to transform data before the analysis. It provides capabilities to filter, modify and enrich data by providing transformations intelligently. HoloClean[9] [13], a statistical inference engine to impute, clean, and enrich data is a weakly supervised ML system which makes use of data integrity constraints, quantitative statistics, value correlations, and external reference data to build a probabilistic model for data cleaning tasks. HoloClean identifies incorrect data values and conflicting tuples in a dataset but suffers from the lack of a convenient user interface.

Trifacta Wrangler is a powerful tool which comes with many data manipulation functionalities including restructuring, cleaning, and enrichment of data. A predictive model computes a ranked set of suggestions in the form of suggestion cards based on user's selection and historical data in an attempt to interpret the data transformation intent [7]. Trifacta also allows users to modify these suggestions to identify which ones suit best, in addition to providing the user the ability to modify a particular suggestion. Though Trifacta provides transformations through predictive interactions, our work bases itself on creating a diverse set of transformations provided to the user based on historical data. OpenRefine[10], an open source tool, enables users to apply basic and advanced transformations on datasets, including normalization of numerical data and fil-

[7] https://www.tableau.com

[8] https://www.talend.com/products/data-preparation

[9] http://www.holoclean.io

[10] http://openrefine.org

tering of text. NADEEF [5] is a generalized data cleaning system which relies on rules to clean data, allowing users to specify data quality rules including functional dependencies for the given dataset. These rules are then used to find and repair violations in the given data. KATARA [4] is a data cleaning system which uses a knowledge base and human help. It can be used to clean various datasets by providing a table as input, and a knowledge base to interpret table semantics. It identifies incorrect data and the possible repairs for it, and uses the help of humans to disambiguate the table semantics and to annotate the data.

Despite substantial research in data preprocessing and use of ML, the above mentioned tools come with certain limitations. Though some of the tools provide graphical user interfaces, it is not always clear whether these tools make use of ML for providing data cleaning and transformation suggestions. HoloClean, on the other hand, uses ML to identify and correct anomalies in the given dataset but does not come with an interactive user interface to visualize the transformations performed on the dataset. OpenRefine, NADEEF, and KATARA rely on human input to clean data for analysis. Though Trifacta comes with a user-friendly interface providing data transformation suggestions based on user interactions, it is unclear which algorithm it uses to generate data transformation suggestions.

4 Approach for Data Transformation Suggestions

Our approach to develop a system for generating data transformation suggestions falls within the broader area of recommender systems [9,3]. We base our approach on feeding the features of the selected data object as input to a ML algorithm, in our case a Random Forest classifier [1]. The classifier then generates the most appropriate transformation by creating a set of decision trees and choosing the output that has most votes [10]. We use the features of the given dataset (*i.e.*, metadata) and the data object selected by the user as input to the Random Forest.

The former group of features contains the attributes of the dataset, including the number of variables and total observations. The latter group contains the properties of the data object selected by the user at a certain instant. That would include the selected row, column, or cell of the dataset and properties of the data it contains. Table 4 and Table 5 show a detailed description of dataset features required and the type of data these could be extracted from.

4.1 Architecture and Process

The proposed architecture is depicted in Figure 1. The application takes tabular data in CSV format as input and allows users to select a row, column or a cell. Relevant transformation suggestions are then generated based on user interaction and the properties of dataset. The different steps involved in the generation of tabular data transformation suggestions include:

1. **Input dataset:** As the first step for data preprocessing, the user is prompted to input a tabular dataset in CSV format into the system. The imported dataset is made visible to the user on the GUI in the form of a table.

Table 4. Features of a given dataset.

Feature	**Scope**
Number of attributes	Entire dataset
Number of observations	Entire dataset
Percentage of missing values	Entire dataset
Percentage of categorical attributes	Entire dataset
Percentage of numeric attributes	Entire dataset
Percentage of boolean attributes	Entire dataset
Percentage of date attributes	Entire dataset
Percentage of *null* attributes	Entire dataset

Table 5. Features of selected data item.

Feature	**Description**	**Scope**
Data type	Number, string, boolean, and date	Column, cell
Data object selected	Row, column, and cell	Entire dataset
Number of numeric values	Number of numeric values in the selected data (zero or more)	Row
Number of boolean values	Number of values with *boolean* data type in the selected data (zero or more)	Row
Number of string values	Number of values with *string* data type in the selected data (zero or more)	Row
Number of date values	Number of values with *date* data type in the selected data (zero or more)	Row
Number of categories	Number of unique categories in the selected data (one or more)	Column
Number of *nulls*	Number of *null* values in the selected data (zero or more)	Row, column
Special characters	Check if any special characters exist in the selected data	Cell

2. **User selection:** The user can then select either a row, column or cell to transform the underlying data.
3. **Features of selected data and metadata of the dataset:** As the user selects data to be transformed, the features of the entire dataset along with the features of data object selected are fetched and saved as test data to be sent to the recommender system.
4. **Training data:** Training data consists of several entries comprised of dataset features and the applied transformations. These entries are part of historical data used to train the ML model to predict the transformations for test data.
5. **Input data:** Both test and training data are sent as input to the Random Forest.
6. **Random Forest:** The Random Forest trains on historical data to learn the mappings between the features of the selected data and applied transforma-

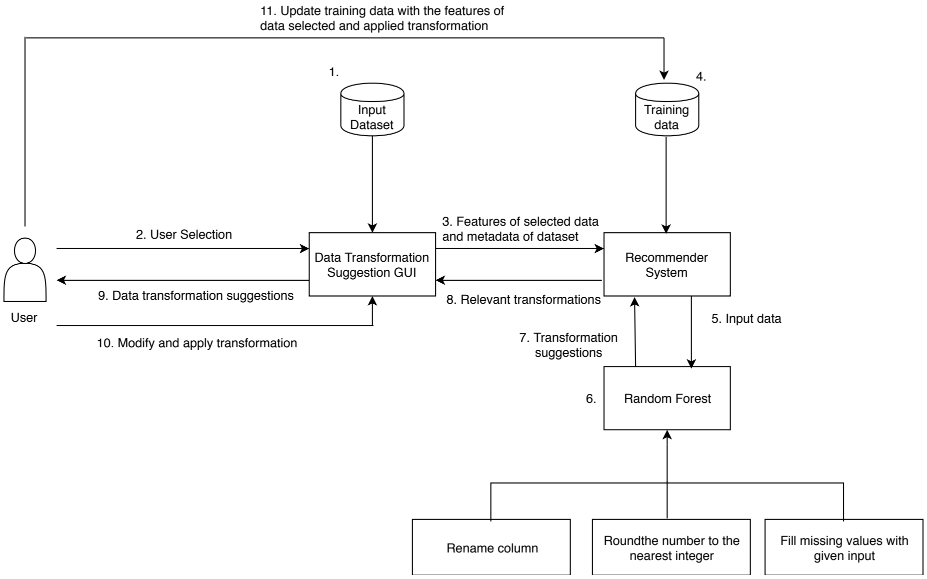

11. Update training data with the features of data selected and applied transformation
1.
Input Dataset
4.
Training data
2. User Selection
Data Transformation Suggestion GUI
3. Features of selected data and metadata of dataset
Recommender System
9. Data transformation suggestions
8. Relevant transformations
User
10. Modify and apply transformation
5. Input data
7. Transformation suggestions
6.
Random Forest
Rename column
Roundthe number to the nearest integer
Fill missing values with given input

Fig. 1. Architecture of prototype for tabular data transformation suggestions.

tions. To include diversity in the transformations generated, we use the same test data to build predictive models with three different training sets (see Figure 2).

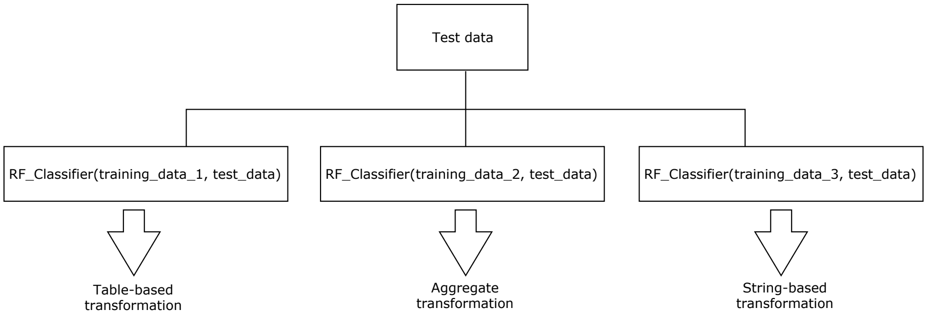

Test data
RF_Classifier(training_data_1, test_data)
RF_Classifier(training_data_2, test_data)
RF_Classifier(training_data_3, test_data)
Table-based transformation
Aggregate transformation
String-based transformation

Fig. 2. Generation of multiple data transformation suggestions.

7. **Transformation suggestions:** At most three transformations are generated as a result of applying the Random Forest algorithm.
8. **Relevant transformations:** The resulting transformations are then sent to the user interface.
9. **Data transformation suggestions:** The data transformation suggestions are shown to user.

10. **Modify and apply transformation:** The user selects one of the transformations and applies it to the dataset. The dataset is then updated with the applied changes.
11. **Update training data:** After the user selected and applied a transformation to the dataset, the features of selected data object that led to the generation of that particular transformation along with the target variable are added to the respective training dataset (see Figure 3).

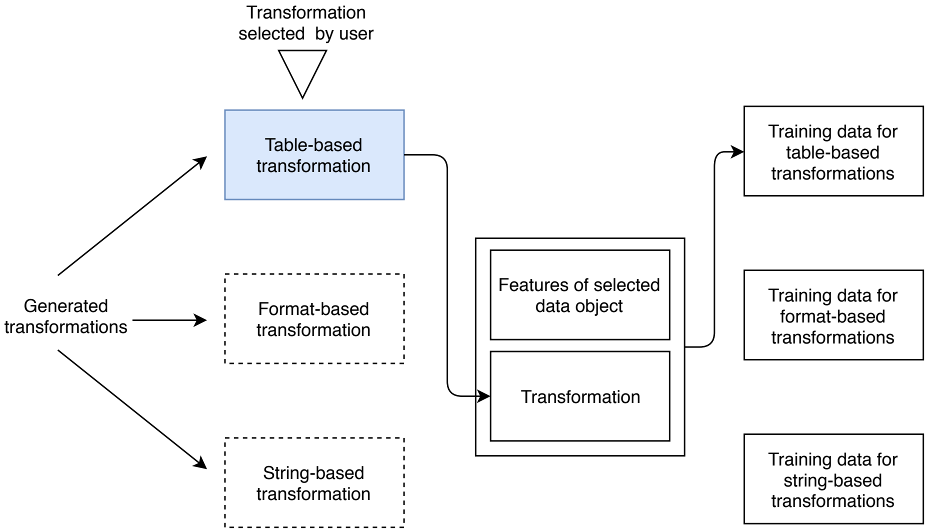

Transformation selected by user
Table-based transformation
Generated transformations
Format-based transformation
String-based transformation
Features of selected data object
Transformation
Training data for table-based transformations
Training data for format-based transformations
Training data for string-based transformations

Fig. 3. Updating training data.

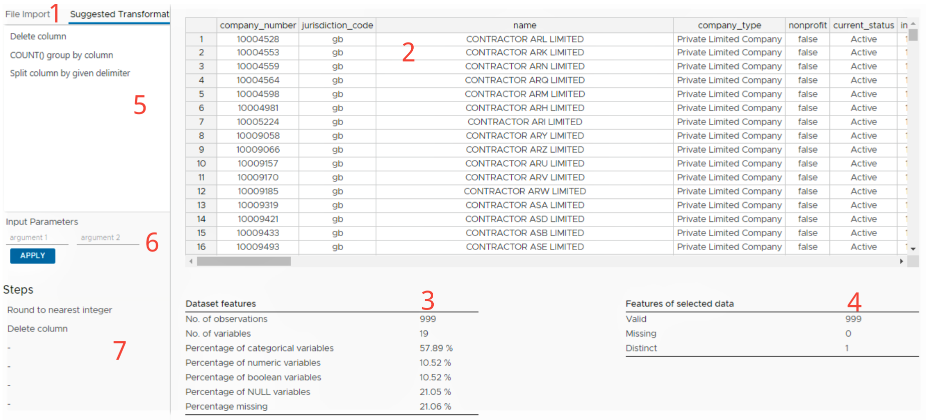
4.2 Implementation

A prototype was implemented by reusing a user-friendly interface for tabular data visualization, taken from two of the authors' previous work [17]. Angular 2[11], an open source Web application framework based on TypeScript[12] was used as development framework for implementing the prototype for data transformation suggestions. For generation of transformation suggestions, we used `random-forest-classifier`[13], which is an open source Javascript library under MIT license.

A screenshot of application prototype is shown in Figure 4. It includes (1) a component for file import in tabular format, (2) tabular representation of data, (3) characteristics of the dataset, (4) features of corresponding column of selected

[11] https://angular.io
[12] https://www.typescriptlang.org
[13] https://www.npmjs.com/package/random-forest-classifier

data object, (5) a set of generated transformations, (6) fields for user input, and (7) the steps taken to transform data.



Delete column

COUNT() group by column

Split column by given delimiter

Input Parameters

	company_number	jurisdiction_code	name	company_type	nonprofit	current_status	in
1	10004528	gb	CONTRACTOR ARL LIMITED	Private Limited Company	false	Active	1
2	10004553	gb	CONTRACTOR ARK LIMITED	Private Limited Company	false	Active	1
3	10004559	gb	CONTRACTOR ARN LIMITED	Private Limited Company	false	Active	1
4	10004564	gb	CONTRACTOR ARQ LIMITED	Private Limited Company	false	Active	1
5	10004598	gb	CONTRACTOR ARM LIMITED	Private Limited Company	false	Active	1
6	10004981	gb	CONTRACTOR ARH LIMITED	Private Limited Company	false	Active	1
7	10005224	gb	CONTRACTOR ARI LIMITED	Private Limited Company	false	Active	1
8	10009058	gb	CONTRACTOR ARY LIMITED	Private Limited Company	false	Active	1
9	10009066	gb	CONTRACTOR ARZ LIMITED	Private Limited Company	false	Active	1
10	10009157	gb	CONTRACTOR ARU LIMITED	Private Limited Company	false	Active	1
11	10009170	gb	CONTRACTOR ARV LIMITED	Private Limited Company	false	Active	1
12	10009185	gb	CONTRACTOR ARW LIMITED	Private Limited Company	false	Active	1
13	10009319	gb	CONTRACTOR ASA LIMITED	Private Limited Company	false	Active	1
14	10009421	gb	CONTRACTOR ASD LIMITED	Private Limited Company	false	Active	1
15	10009433	gb	CONTRACTOR ASB LIMITED	Private Limited Company	false	Active	1
16	10009493	gb	CONTRACTOR ASE LIMITED	Private Limited Company	false	Active	1

Steps

Round to nearest integer

Delete column

-

-

-

-

Dataset features	
No. of observations	999
No. of variables	19
Percentage of categorical variables	57.89 %
Percentage of numeric variables	10.52 %
Percentage of boolean variables	10.52 %
Percentage of NULL variables	21.05 %
Percentage missing	21.06 %

Features of selected data	
Valid	999
Missing	0
Distinct	1

Fig. 4. Screenshot of the application prototype.

The dataset characteristics contain basic statistical features of data to help the user view the dimensions and the different data types included in the dataset. These statistical features include the number of observations, number of variables, percentage of categorical variables, percentage of numeric variables, percentage of boolean variables, percentage of null variables, and percentage missing variables.

The features of corresponding column of selected data object include the number of values that are not null, undefined or NaN[14], the number of null and undefined values, and the number of distinct values.

At the initialization of the application, we use training data generated beforehand to generate transformations. For inducing diversity, training data is divided into three categories. These three categories do not necessarily include data transformations for all three data objects, *i.e.*, row, column and cell. The ML classifier therefore returns null for the corresponding category if the user selection has not been mapped to an appropriate transformation. To ensure that no irrelevant transformations are suggested to the user, we implemented two checks, as follows. (1) Initially, due to lack of sufficient training data, the Random Forest classifier may not generate the appropriate transformations. A check for irrelevant transformations ensures that no incorrect transformations with respect to the data object selected and its data type are suggested to the user. Also (2) if there are no transformations returned, we generate transformations randomly based on the data type and the type of data object selected. This check is implemented to ensure that at least one recommendation is provided.

[14] `https://en.wikipedia.org/wiki/NaN(Not-A-Number)`

Once the transformation has been applied, the system needs feedback for the generation of future transformations. The recommendation system collects feedback implicitly by interpreting the preferences of the user from the applied transformation and adds an entry containing features of the dataset, features of the selected data object and the applied transformation to the training data. Adding more examples in the training dataset increases the performance of the prediction algorithm.

5 Evaluation

We ran a usability study with 15 target users testing the prototype. All selected users conduct data transformation tasks in their daily work. After the testing, users answered a set of questions from the System Usability Scale (SUS) [2].

The data used as the use case for this evaluation was a sample from OpenCorporates [15], which is an open database of companies and company-related data. The dataset is in CSV format and consists of 999 rows and 19 columns. This dataset includes textual, numerical and boolean data in addition to dates. There are several transformations that can be applied to tabular data for preprocessing. For the purpose of this evaluation, however, only a few have been selected and implemented as shown in Table 6.

For the evaluation, each participant was provided with a list of transformations implemented in the prototype, type of data they can be applied to, and the information about external input, if any. The participants were then required to choose the dataset containing information about companies for preprocessing. Each participant was allowed to perform necessary preprocessing of the dataset by selecting rows, columns and cells, and applying transformations on those data objects. In addition to the predefined test data, personal preferences were automatically inferred for each user of the prototype by analyzing the transformations applied.

Each participant of the usability testing ranked each of the 10 SUS questions with a scale from 1 to 5. An SUS score was calculated individually for each participant's responses. For calculating the SUS score, for each of the odd numbered questions, 1 is subtracted from the response, and for each of the even numbered questions, 5 is subtracted from the response.

The values calculated were added up and the sum was multiplied by 2.5 to obtain the overall value of SUS in a range of 0 to 100. Even though the score is on the scale of 0 - 100, it is not percentage. We obtained an average score of 72 which lies above 50th percentile.

Based on feedback from the users who did hands-on testing of prototype, we could draw the following conclusions: (1) the users found the application easy to use, (2) data transformations are often repeated on similar data objects, (3) the users prefer a wide variety of data transformations to choose from, (4) editable transformations should be implemented, which can provide flexibility

[15] `https://opencorporates.com`

Table 6. Selected data transformations.

Function name	**Data Type**	**Scope**
deleteColumn	Any	Column
keepColumn	Any	Column
renameColumn	Any	Column
convertRowToHeader	N/A	Row
deleteRow	N/A	Row
setToNULL	Any	Cell
normalize	Number	Column
countGroupByColumn	Number, String, Boolean	Column
upperCase	String	Column
lowerCase	String	Column
roundToNearest	Number	Cell
setRowToNULL	N/A	Row
setColumnToNULL	Any	Column
fillMissingValues	Any	Column
extract	String	Cell
replace	Number, String, Boolean, NULL	Cell
splitColumn	String	Cell
removeSpecialCharacters	String	Cell

while transforming data, and (5) more intelligent detection of data types such as zip codes and URLs ease data transformation process.

6 Conclusions

In this work, we developed an approach for the use of ML to recommend data transformation suggestions based on user interactions, and analyzed the usefulness of the approach for users of data cleaning and transformation tools. The proposed approach based on the Random Forest algorithm was implemented and tested on a company dataset. By tracking the user interactions performed by the selection of data objects in the tabular dataset, the application prototype recommends relevant data transformations. The implementation was evaluated with a usability study and found efficient by the users. Future work includes detection and support for non-primitive data types (*e.g.*, geographical), a multi-label classification model to generate multiple possible target variables, and a friendlier user interface that would help users transform data with ease.

Acknowledgements. The work in this paper was partly funded by the EC H2020 projects euBusinessGraph (Grant nr. 732003), EW-Shopp (Grant nr. 732590), and TheyBuyForYou (Grant nr. 780247).

References

1. Breiman, L.: Random forests. Machine learning **45**(1), 5–32 (2001)
2. Brooke, J.: SUS-A quick and dirty usability scale. Usability evaluation in industry **189**(194), 4–7 (1996)
3. van Capelleveen, G., Amrit, C., Yazan, D.M., Zijm, H.: The recommender canvas: a model for developing and documenting recommender system design. Expert Systems with Applications **129**, 97–117 (2019)
4. Chu, X., Morcos, J., Ilyas, I.F., Ouzzani, M., Papotti, P., et al.: Katara: A data cleaning system powered by knowledge bases and crowdsourcing. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2015). pp. 1247–1261 (2015)
5. Dallachiesa, M., Ebaid, A., Eldawy, A., Elmagarmid, A., et al.: NADEEF: a commodity data cleaning system. In: Proceedings of the ACM SIGMOD International Conference on Management of Datav (SIGMOD 2013). pp. 541–552 (2013)
6. Famili, A., Shen, W.M., Weber, R., et al.: Data preprocessing and intelligent data analysis. Intelligent Data Analysis **1**(1-4), 3–23 (1997)
7. Heer, J., Hellerstein, J.M., Kandel, S.: Predictive Interaction for Data Transformation. In: Proceedings of the 7th Biennial Conference on Innovative Data Systems Research (CIDR 2015) (2015)
8. Kandel, S., Paepcke, A., Hellerstein, J.M., Heer, J.: Enterprise data analysis and visualization: An interview study. IEEE Transactions on Visualization and Computer Graphics **18**(12), 2917–2926 (2012)
9. Melville, P., Sindhwani, V.: Recommender systems. Encyclopedia of Machine Learning and Data Mining pp. 1056–1066 (2017)
10. Oshiro, T.M., Perez, P.S., Baranauskas, J.A.: How many trees in a random forest? In: Proceedings of the 8th International Conference International Workshop on Machine Learning and Data Mining in Pattern Recognition (MLDM 2012). LNCS, vol. 7376, pp. 154–168. Springer (2012)
11. Rahm, E., Do, H.H.: Data cleaning: Problems and current approaches. IEEE Data Engineering Bulletin **23**(4), 3–13 (2000)
12. Raman, V., Hellerstein, J.M.: Potters wheel: an interactive framework for data cleaning and transformation. In: Proceedings of the 27th International Conference on Very Large Data Bases (VLDB 2001). pp. 381–390 (2001)
13. Rekatsinas, T., Chu, X., Ilyas, I.F., Ré, C.: Holoclean: Holistic data repairs with probabilistic inference. Proceedings of the VLDB Endowment **10**(11), 1190–1201 (2017)
14. Roman, D., Dimitrov, M., Nikolov, N., Putlier, A., Sukhobok, D., Elvesæter, B., Berre, A., Ye, X., Simov, A., Petkov, Y.: Datagraft: Simplifying open data publishing. In: European Semantic Web Conference. pp. 101–106. Springer (2016)
15. Roman, D., Nikolov, N., Putlier, A., Sukhobok, D., Elvesæter, B., Berre, A., Ye, X., Dimitrov, M., Simov, A., Zarev, M., et al.: Datagraft: One-stop-shop for open data management. Semantic Web **9**(4), 393–411 (2018)
16. Sukhobok, D., Nikolov, N., Pultier, A., Ye, X., et al.: Tabular data cleaning and linked data generation with Grafterizer. In: Proceedings of the ESWC 2016 Satellite Events. LNCS, vol. 9989, pp. 134–139. Springer (2016)
17. von Zernichow, B.M., Roman, D.: Usability of visual data profiling in data cleaning and transformation. In: Panetto, H., Debruyne, C., Gaaloul, W., Papazoglou, M., Paschke, A., Ardagna, C.A., Meersman, R. (eds.) On the Move to Meaningful Internet Systems. OTM 2017 Conferences. pp. 480–496. Springer International Publishing, Cham (2017)