The Landscape of R Packages for Automated Exploratory Data Analysis

by Mateusz Staniak and Przemysław Biecek

Abstract

The increasing availability of large but noisy data sets with a large number of heterogeneous variables leads to the increasing interest in the automation of common tasks for data analysis. The most time-consuming part of this process is the Exploratory Data Analysis, crucial for better domain understanding, data cleaning, data validation, and feature engineering.

There is a growing number of libraries that attempt to automate some of the typical Exploratory Data Analysis tasks to make the search for new insights easier and faster. In this paper, we present a systematic review of existing tools for Automated Exploratory Data Analysis (autoEDA). We explore the features of fifteen popular R packages to identify the parts of analysis that can be effectively automated with the current tools and to point out new directions for further autoEDA development.

Introduction

With the advent of tools for automated model training (autoML), building predictive models is becoming easier, more accessible and faster than ever. Tools for R such as mlrMBO (Bischl et al., 2017), parsnip (Kuhn and Vaughan, 2019); tools for python such as TPOT (Olson et al., 2016), auto-sklearn (Feurer et al., 2015), autoKeras (Jin et al., 2018) or tools for other languages such as H2O Driverless AI (H2O.ai, 2019; Cook, 2016) and autoWeka (Kotthoff et al., 2017) supports fully- or semi-automated feature engineering and selection, model tuning and training of predictive models.

Yet, model building is always preceded by a phase of understanding the problem, understanding of a domain and exploration of a data set. Usually, in the process of the data analysis much more time is spent on data preparation and exploration than on model tuning. This is why the current bottleneck in data analysis is in the exploratory data analysis (EDA) phase. Recently, a number of tools were developed to automate or speed up the part of summarizing data and discovering patterns. Since the process of building predictive models automatically is referred to as autoML, we will dub the automation of data exploration autoEDA. The surge in interest in autoEDA tools is evident in the Figure 1. Table 1 describes the popularity of autoEDA tools measured as the number of downloads from CRAN and usage statistics from Github.

There is an abundance of R libraries that provide functions for both graphical and descriptive data exploration. Here, we restrict our attention to packages that aim to automatize or significantly speed up the process of exploratory data analysis for tabular data. Such tools usually work with full data frames, which are processed in an automatic or semi-automatic manner, for example by guessing data types, and return summary tables, groups of plots or full reports. Currently, there is no CRAN Task View dedicated to packages for automated Exploratory Data Analysis and neither was there any repository that would catalogue them. Here, we make a first attempt to comprehensively describe R tools for autoEDA. We chose two types of packages. The first group explicitly aims to automate EDA, as stated in the description of the package. These includes packages for fast, easy, interactive or automated data exploration. The second group contains packages that create data summaries. These packages were included, as long as they address at least two analysis goals listed in Table 2. We do not describe in detail packages that are either restricted to one area of application (for example RBioPlot (Zhang and Storey, 2016) package dedicated to biomolecular data or intvy (Caro and Biecek, 2017) package focused on international large-scale assessments), designed for one specific task (for example creating tables), or in an early development phase. Some of the more task-specific packages are briefly discussed in Section 2.2.16. Some packages, such as radiant (Nijs, 2019) cover the full analysis pipeline and, as such, are too general for our purposes, even though they include an EDA module.

This paper has two main goals. First is to characterize existing R packages for automated Exploratory Data Analysis and compare their ranges of capabilities. To our best knowledge, this is first such a review. Previously, a smaller comparison of seven packages was done in Putatunda et al. (2019). Second is to identify areas, where automated data exploration could be improved. In particular, we are interested in gauging the potential of AI-assisted EDA tools.

The first goal is addressed in Sections 2.2 R packages for automated EDA and 2.3 Feature comparison where we first briefly describe each package and the compare, how are different EDA tasks are tackled.
Automated EDA tools aim to make the Data understanding phase as fast and as easy as possible. This part of a project can be further divided into smaller tasks. These include a description of a dataset, data exploration, and data quality verification. All these tasks can be achieved both by providing descriptive statistics and numerical summaries and by visual means. AutoEDA packages provide functions to deal with these challenges. Some of them are also concerned with simple variable transformations and data cleaning. Both these tasks belong in the Data preparation phase, which precedes and supports the model building phase. Let us notice that business understanding is affected by data understanding, which makes this part of the analysis especially important.

Goals of autoEDA tools are summarised in Table 2. The Phase and Tasks columns are taken from the CRISP-DM standard, while Type and Examples columns provide examples based on current functionalities of autoEDA packages.

Each task should be summarised in a report, which makes reporting another relevant problem of autoEDA. Uni- and bivariate data exploration is a part of the analysis that is most thoroughly covered by the existing autoEDA tools. The form of univariate summaries depends on the variable type. For numerical variables, most packages provide descriptive statistics such as centrality and dispersion measures. For categorical data, unique levels and associated counts are reported. Bivariate relationships descriptions display either dependency between one variable of interest and all other variables, which includes contingency tables, scatter plots, survival curves, plots of distribution by

Figure 1: Trends in number of downloads of autoEDA packages available on CRAN since the first release. Data was gathered on 12.07.2019 with the help of the cranlogs package (Csardi, 2015).
Table 1: Popularity of R packages for autoEDA among users and package developers. First two columns summarise CRAN statistics, last five columns summarise package development at GitHub. When a repository owned by the author is not available, the data were collected from a CRAN mirror repository. Data was gathered on 12.07.2019.

<table>
<thead>
<tr>
<th>package</th>
<th>CRAN</th>
<th>GitHub</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>downl.</td>
<td>debut</td>
</tr>
<tr>
<td>arsenal</td>
<td>39234</td>
<td>2016-12-30</td>
</tr>
<tr>
<td>autoEDA</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DataExplorer</td>
<td>82624</td>
<td>2016-03-01</td>
</tr>
<tr>
<td>dataMaid</td>
<td>23972</td>
<td>2017-01-02</td>
</tr>
<tr>
<td>dlookr</td>
<td>13268</td>
<td>2018-04-27</td>
</tr>
<tr>
<td>ExPanDaR</td>
<td>5713</td>
<td>2018-05-11</td>
</tr>
<tr>
<td>explore</td>
<td>808</td>
<td>2019-05-16</td>
</tr>
<tr>
<td>exploreR</td>
<td>8112</td>
<td>2016-02-10</td>
</tr>
<tr>
<td>funModeling</td>
<td>54232</td>
<td>2016-02-07</td>
</tr>
<tr>
<td>inspectdf</td>
<td>3252</td>
<td>2019-04-24</td>
</tr>
<tr>
<td>RtutoR</td>
<td>10502</td>
<td>2016-03-12</td>
</tr>
<tr>
<td>SmartEDA</td>
<td>5150</td>
<td>2018-04-06</td>
</tr>
<tr>
<td>summarytools</td>
<td>84737</td>
<td>2014-08-11</td>
</tr>
<tr>
<td>visdat</td>
<td>68978</td>
<td>2017-07-11</td>
</tr>
<tr>
<td>xray</td>
<td>8300</td>
<td>2017-11-22</td>
</tr>
</tbody>
</table>

values of a variable (histograms, bar plots, box plots), or between all pairs of variables (correlation matrices and plots), or chosen pairs of variables.

### R packages for automated EDA

In this section, fifteen R libraries are shortly summarised. One of them is only available on GitHub (autoEDA), all others are available at CRAN. For each library, we include example outputs. The exact versions of packages that were used to create them can be found in the reference section. All examples are based on a subset of typical_data
data set from visdat package. Whenever possible, archivist (Biecek and Kosinski, 2017) hooks are provided for easy access to the presented objects. When a function call only gives side-effects, a link is provided to the full result (PDF/PNG files). Tables were prepared with the xtable package (Dahl et al., 2018).

### The arsenal package

The arsenal package (Heinzen et al., 2019) is a set of four tools for data exploration:

1. table of descriptive statistics and p-values of associated statistical tests, grouped by levels of a target variable (the so-called Table 1). Such a table can also be created for paired observation, for example longitudinal data (tableby and paired functions),
2. comparison of two data frames that can detect shared variables (compare function),
3. frequency tables for categorical variables (freqlist function),
4. fitting and summarizing simple statistical models (linear regression, Cox model, etc) in tables of estimates, confidence intervals and p-values (modelsum function).

Results of each function can be saved to a short report using the write2 function. An example can be found in Figure 2.

A separate vignette is available for each of the functions. arsenal is the most statistically-oriented package among reviewed libraries. It borrows heavily from SAS-style procedures used by the authors at the Mayo Clinic.

### The autoEDA package

autoEDA package (Horn, 2018a) is a GitHub-based tool for univariate and bivariate visualizations and
### Table 2: Early phases of data mining project according to CRISP-DM standard, their specific goals and examples of how they are aided by autoEDA tools. (Wirth, 2000)

<table>
<thead>
<tr>
<th>Phase</th>
<th>Task</th>
<th>Type</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data understanding</td>
<td>Data description</td>
<td>dimensions</td>
<td>variables number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>variables meta-data</td>
<td>variable type size in RAM</td>
</tr>
<tr>
<td></td>
<td>Data validity</td>
<td>invalid values</td>
<td>typos</td>
</tr>
<tr>
<td></td>
<td></td>
<td>missing values</td>
<td>NA count</td>
</tr>
<tr>
<td></td>
<td></td>
<td>atypical values</td>
<td>outliers</td>
</tr>
<tr>
<td></td>
<td>Data exploration</td>
<td>univariate</td>
<td>histogram scatter plot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bivariate</td>
<td>Parallel coord. plot</td>
</tr>
<tr>
<td></td>
<td></td>
<td>multivariate</td>
<td></td>
</tr>
<tr>
<td>Data preparation</td>
<td>Data cleaning</td>
<td>Imputation</td>
<td>Impute mean</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Outlier treatment</td>
<td>Impute median</td>
</tr>
<tr>
<td></td>
<td>Derived attributes</td>
<td>Dimension reduction</td>
<td>PCA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Continuous</td>
<td>Box-Cox transform</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Categorical</td>
<td>Binning</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Merge rare factors</td>
</tr>
</tbody>
</table>

**Figure 2:** An example output from the `arsenal::tableby` function saved using `arsenal::write2` (arsenal v 2.0). Smokes and Race variables are compared by the levels of Died variable.

summarizes. The `dataOverview` function returns a data frame that describes each feature by its type, number of missing values, outliers and typical descriptive statistics. Values proposed for imputation are also included. Two outlier detection methods are available: Tukey and percentile-based. A PDF report can be created using the `autoEDA` function. It consists of the plots of distributions of predictors grouped by outcome variable or distribution of outcome by predictors.

The package can be found on Xander Horn’s GitHub page: [https://github.com/XanderHorn/autoEDA](https://github.com/XanderHorn/autoEDA). It does not include a vignette, but a short introduction article was published to LinkedIn ([Horn, 2018b](https://linkedin.com/posts)) and similar examples can be found in the readme of the project. Plots from a report generated by `autoEDA` are displayed in Figure 3.

### The DataExplorer package

**DataExplorer** ([Cui, 2019](https://dataexplorer.org)) is a recent package that helps automatize EDA and simple data transformations. It provides functions for:

1. whole dataset summary: dimensions, types of variables, missing values, etc (introduce and

---

The R Journal Vol. XX/YY, AAAA 20ZZ

ISSN 2073-4859
Figure 3: Sample pages from the report generated by the autoEDA::autoEDA function (autoEDA v. 1.0) displaying bivariate relationships between the target and explanatory variable.

The create_report function generates a report. By default, it consists of all the above points except for data transformations and it can be further customized. An introductory vignette Introduction to DataExplorer that showcases all the functionalities is included in the package. It is noticeable that the package almost entirely relies on visual techniques. Plots taken from an example report are presented in Figure 4.

The dataMaid package

The dataMaid (Petersen and Ekstrom, 2018) package has two central functions: the check function, which performs checks of data consistency and validity, and summarize, which summarizes each column. Another function, makeDataReport, automatically creates a report in PDF, DOCX or HTML format. The goal is to detect missing and unusual - outlying or incorrectly encoded - values. The report contains whole dataset summary: variables and their types, number of missing values, and univariate summaries in the form of descriptive statistics, histograms/bar plots and an indication of possible problems.

User-defined checks and summaries can be also included in the analysis. The vignette Extending dataMaid explains how to define them. It is also possible to customize the report. In particular, variables for which no problems were detected can be omitted. An example report\textsuperscript{8} can be found in Figure 5.

The dlookr package

The dlookr (Ryu, 2019) package provides tools for 3 types of analysis: data diagnosis including correctness, missing values, outlier detection; exploratory data analysis; and variable transformations: imputation, dichotomization, and transformation of continuous features. It can also automatically generate a PDF report for all these analyses.

For data diagnosis, types of variables are reported along with counts of missing values and unique values. Variables with a low proportion of unique values are described separately. All the typical descriptive statistics are provided for each variable. Outliers are detected and distributions of variables before and after outlier removal are plotted. Both missing values and outliers can be treated using impute_na and impute_outlier functions.

In the EDA report, descriptive statistics are presented along with normality tests, histograms of variables and their transformations that reduce skewness: logarithm and root square. Correlation plots are shown for numerical variables. If the target variable is specified, plots that show the relationship between the target and each predictor are also included.

A transformation report compares descriptive statistics and plots for each variable before and after imputation, skewness-removing transformation and binning. If the right transformation is found among the candidate transformations, it can be applied to the feature through one of the binning\textsuperscript{9}, binning\_by, or transform functions.

Every operation or summary presented in the reports can also be performed manually. A dedicated vignette explains each of the main functionalities (Data quality diagnosis, Data Transformation, Exploratory Data Analysis vignettes). An example\textsuperscript{9} taken from one of the reports can be found in Figure 6.

The ExPanDaR package

Notably, while the ExPanDaR package (Gassen, 2018) was designed for panel data exploration, it can also be used for standard EDA after adding an artificial constant time index. In this case, the package offers interactive shiny application for exploration. Several types of analysis are covered:

1. missing values and outlier treatment,

---

\textsuperscript{8}Find the full report at https://github.com/mstaniak/autoEDA-resources/blob/master/autoEDA-paper/plots/dataMaid/dataMaid_report.pdf

\textsuperscript{9}Access the full report at https://github.com/mstaniak/autoEDA-resources/blob/master/autoEDA-paper/plots/dlookr/dlookr_eda.pdf
Part 1

Data report overview

The dataset examined has the following dimensions:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of observations</td>
<td>500</td>
</tr>
</tbody>
</table>

Checks performed:

The following variable checks were performed, depending on the data type of each variable:

<table>
<thead>
<tr>
<th>Checks performed</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity missing values</td>
<td>x</td>
</tr>
<tr>
<td>Identity perfect values</td>
<td>x</td>
</tr>
<tr>
<td>Identity non-missing</td>
<td></td>
</tr>
<tr>
<td>Identity missing non-missing</td>
<td>x</td>
</tr>
<tr>
<td>Identity non-missing non-missing</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5: Two pages from a data validity report generated using the `dataMaid::makeDataReport` function (`dataMaid` v. 1.2). Atypical values are listed under the variable summary.

Chapter 1

Introduction

The EDA Report provides exploratory data analysis information on objects to its `data.frame` and `data.table`.

1.1 Information of Dataset

The dataset that generated the EDA Report is an `data.frame` object. It consists of observations and 9 variables.

1.2 Information of Variables

<table>
<thead>
<tr>
<th>Variables</th>
<th>Type</th>
<th>Missing</th>
<th>Min</th>
<th>Quartile 1</th>
<th>Median</th>
<th>Quartile 3</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td>factor</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Age</td>
<td>numeric</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Height</td>
<td>numeric</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>IQ</td>
<td>numeric</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Smoke</td>
<td>numeric</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Figure 6: Two pages from a report generated by the `dlookr::eda_report` function (`dlookr` v. 0.3.8). First, the dataset is summarised, then each variable is described. Optionally, plots of bivariate relationships can be added.
2. univariate summaries (descriptive statistics) and plots (histograms/bar plots),
3. bivariate analysis via correlation matrices and plots. Interestingly, scatter plots can be enriched by associating size and color of points with variables,
4. multivariate regression analysis.

For each functionality of the application, there is a corresponding standalone function.

Three vignettes describe how the library can be used for data exploration (Using the functions of the ExPanDaR package), how to customize it (Customize ExPanD) and how to analyze panel data (Using ExPanD for Panel Data Exploration) Example instances of ExPanDaR shiny applications are available online. Links and other examples can be found in the GitHub repository of the package: https://github.com/joachim-gassen/ExPanDaR. An example of a scatter plot created by the package can be found in Figure 7.

**Figure 7**: Scatter plot with of Income and IQ variables with Died variable denoted by the color. Created with the prepare_scatter_plot function (ExPanDaR v0.4.0).

The functionalities of the explore package (Krasser, 2019) can be accessed in three ways: through an interactive shiny (Chang et al., 2019) application, through an automatically generated HTML report or via standalone functions. In addition to data exploration, relationships with a binary target can be explored. The package includes functions for

1. full dataset summaries - dimensions, data types, missing values and summary statistics (describe function),
2. uni- and bivariate visualizations, including density plots, bar plots and boxplots (a family of explore functions, in particular explore_all function that creates plots for all variables),
3. simple modeling based on decision trees (explain_tree function) or logistic regression (explain_logreg function).

All result can be saved to HTML via the report function. Dataset and variable summaries can also be save to an MD file using the data_dict_md function. The explore vignette includes a thorough description of the package. An example decision tree can be found in Figure 8.

---

10 Access the R object with the archivist::aread("mstaniak/autoEDA-resources/autoEDA-paper/9c5d")
11 Find examples at https://github.com/mstaniak/autoEDA-resources/tree/master/autoEDA-paper/plots/explore
12 Access the R object with the archivist::aread("mstaniak/autoEDA-resources/autoEDA-paper/dc47").
Figure 8: A decision tree fitted using the explain_tree function (explore v. 0.4.3). The tree can also be based on multiple explanatory variables.

Figure 9: Univariate regression plot created using the exploreR::massregplot (exploreR v. 0.1).

The exploreR package

The exploreR package (Coates, 2016) takes a unique approach to data exploration compared to other packages. The analysis is based on linear regression. There are three functionalities:

1. fitting univariate regression model for each independent variable and summarizing the results in a table that consists of estimated parameters, p-values, and $R^2$ values (masslm function),
2. plotting target variable against each independent variable along with the fitted least squares line (massregplot function),
3. feature standardization by scaling to the interval $[0, 1]$ or subtracting mean and dividing by standard deviation.

Regression plots can be saved to a PDF file. A vignette called The How and Why of Simple Tools explains all the functions and provides examples. One of the regression plots\footnote{A PDF file with all the plots can be found at \url{https://github.com/mstaniak/autoEDA-resources/blob/master/autoEDA-paper/plots/exploreR.pdf}} is presented in Figure 9.
Figure 10: An example output from the funModeling::cross_plot function (funModeling v. 1.7). Such a plot is drawn for every variable in the dataset or for a specified subset of variables. Continuous features are discretized.

The funModeling package

The package funModeling (Casas, 2019) is a rich set of tools for EDA connected to the book Casas (2018). These tools include

1. dataset summary (df_status function),
2. plots and descriptive statistics for categorical and numerical variables (plot_num, profiling_num and freq functions),
3. classical and information theory-based correlation analysis for target variable vs other variables - (correlation_table function for numerical predictors, var_rank_info function for all predictors),
4. plots of distribution of target variables vs predictors (bar plots, box plots and histograms via cross_plot and plotar functions),
5. quantitative analysis for binary target variables (categ_analysis function),
6. different methods of binning continuous features (discretize_df, convert_df_to_categoric and discretize_rgr functions),
7. variable normalization by transforming to the [0,1] interval (range01 function),
8. outlier treatment (prep_outliers, tukey_outlier and hampel_outlier functions),
9. gain and lift curves (gain_lift function).

It is the only library that encompasses visualizations related to predictive models and non-standard correlation analysis. The range of tools covered by funModeling is very wide. The package includes an exhaustive introduction vignette called funModeling quick-start. One of the bivariate visualizations offered by the package can be found in Figure 10.

The inspectdf package

The inspectdf package (Rushworth, 2019) provides several tools for basic data exploration with a consistent interface. Each of the inspect_* functions returns a data frame with summaries (and additional attributes). The results can be then plotted using the show_plot function. The function are related to three aspects of EDA:

1. whole dataset can be summarised by numbers of missing values, number of variables of each type and memory used by each variable (inspect_na, inspect_types and inspect_mem functions),

Find all the plots at https://github.com/mstaniak/autoEDA-resources/tree/master/autoEDA-paper/plots/funmodeling

---

14Find all the plots at https://github.com/mstaniak/autoEDA-resources/tree/master/autoEDA-paper/plots/funmodeling
Figure 11: A comparison of correlations between numerical variables in two data frames. Plot created using the `inspectdf` package v. 0.0.3.

2. univariate analysis is done via summary statistics and histograms for numerical variables (`inspect_num` function), bar plots for categorical variables (`inspect_cat` function). Additionally, factors dominated by a single level can be found with the `inspect_imb` function,

3. bivariate relationships are described by Pearson correlation coefficient for numerical variables (`inspect_cor` function).

Notably, each function can take two data frames as parameters and return their comparison. An example of a correlation analysis plot comparing two data frames can be found in Figure 11. While the library does not include a vignette, extensive documentation with examples is provided on the GitHub webpage of the project: https://github.com/alastairrushworth/inspectdf.

The RtutoR package

The `RtutoR` package (Nair, 2018a) is a tool for automated reporting. There are three options for creating a report that contains univariate and bivariate data summaries:

1. plots can be created interactively in a shiny app (`launch_plotter` function),
2. the whole report can be generated from a shiny app that allows the user to tweak the report (`gen_exploratory_report_app` function),
3. the report can be created by a direct call to the `generate_exploratory_analysis_ppt` function.

The report is saved in the PPTX format. Notably, this package can identify the top k relevant variables based on a chosen criterion, for example, information gain, and display plots only for these variables. An example report can be found in the GitHub repository of the package. The package was introduced in an R-Bloggers blog post (Nair, 2018b).

The SmartEDA package

The `SmartEDA` package (Ubrangala et al., 2018), is focused entirely on data exploration through graphics and descriptive statistics. It does not provide any functions which modify existing variables. The range of tools it includes is wide:

1. dataset summary (`ExpData` function),

Find the report at https://github.com/anup50695/RtutoR/blob/master/titanic_exp_report_2.pptx
Exploratory Data analysis (EDA)

Analyzing the data sets to summarize their main characteristics of variables, often with visualizations, without using a statistical model.

1. Overview of the data
   Understanding the dimensions of the dataset, variable names, overall missing summary and data type of each variable.

- Overview of the data
- Structure of the data

Figure 12: Sample pages from a report generated by the SmartEDA::ExpReport function (SmartEDA v. 0.3), including dataset overview and bivariate dependency for categorical variables.

2. descriptive statistics that may include correlation with target variable and density or bar plots (ExpNumStat, ExpNumViz, ExpCatStat and ExpCatViz functions). All visualizations may include the target variable;
3. QQ plots (ExpOutQQ function),
4. contingency tables (ExpCTable function),
5. information value and Weight of the Evidence coding (ExpWoETable, ExpInfoValue functions),
6. parallel coordinate plot for multivariate visualization (ExpParcoord function).

Plotting functions return grids of ggplot2 object. The results can be written to a HTML report (ExpReport function). There are also additional functionalities dedicated to data.table objects from data.table package (Dowle and Srinivasan, 2019). An introductory vignette called Explore data using SmartEDA (Intro) is attached to the library. Another vignette Custom summary statistics describe customizing output tables. The package is also described in the Putatunda et al. (2019) paper. Examples can be found in Figure 12.

The summarytools package

The summarytools package (Comtois, 2019) builds summary tables for whole datasets, individual variables, or pairs of variables. In addition, the output can be formatted to be included in knitr (Xie, 2015) or plain documents, HTML files and shiny apps (Chang et al., 2019). The are four main functionalities:

1. whole dataset summary including variable types and a limited number of descriptive statistics, counts of unique values and missing values and univariate plots within the output table (dfSummary function),
2. descriptive statistics, including skewness and kurtosis, for numerical variables, possibly grouped by levels of a factor (descr, stby functions),
3. counts and proportions for levels of categorical features (freq function),
4. contingency tables for pairs of categorical variables (ctable function).

All results can be saved and displayed in different formats. The package includes a vignette titled Introduction to summarytools. An example of univariate summaries can be found in Figure 3.

17 Access the R object with archivist::aread("mstaniak/autoEDA-resources/autoEDA-paper/9e12").
<table>
<thead>
<tr>
<th></th>
<th>Height(cm)</th>
<th>IQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>175.09</td>
<td>100.23</td>
</tr>
<tr>
<td>Std.Dev.</td>
<td>9.83</td>
<td>10.03</td>
</tr>
<tr>
<td>Min</td>
<td>146.30</td>
<td>68.00</td>
</tr>
<tr>
<td>Q1</td>
<td>168.20</td>
<td>93.00</td>
</tr>
<tr>
<td>Median</td>
<td>175.30</td>
<td>100.00</td>
</tr>
<tr>
<td>Q3</td>
<td>182.05</td>
<td>107.00</td>
</tr>
<tr>
<td>Max</td>
<td>207.20</td>
<td>137.00</td>
</tr>
<tr>
<td>MAD</td>
<td>10.38</td>
<td>10.38</td>
</tr>
<tr>
<td>IQR</td>
<td>13.83</td>
<td>14.00</td>
</tr>
<tr>
<td>CV</td>
<td>0.06</td>
<td>0.10</td>
</tr>
<tr>
<td>Skewness</td>
<td>-0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>SE.Skewness</td>
<td>0.08</td>
<td>0.08</td>
</tr>
<tr>
<td>Kurtosis</td>
<td>-0.30</td>
<td>-0.04</td>
</tr>
<tr>
<td>N.Valid</td>
<td>1000.00</td>
<td>898.00</td>
</tr>
<tr>
<td>% Valid</td>
<td>100.00</td>
<td>89.80</td>
</tr>
</tbody>
</table>

Table 3: An example table of descriptive statistics generated by the `summarytools::descr` function (`summarytools` v. 0.9.2).

**The visdat package**

The package **visdat** (Tierney, 2017) is maintained by rOpenSci. It consists of six functions that help visualize:

1. variables types and missing data (**vis_dat** function),
2. types of each value in each column (**vis_guess** function),
3. clusters of missing values (**vis_miss** function),
4. differences between the two datasets (**vis_compare** function),
5. where given conditions are satisfied in the data (**vis_expect** function),
6. correlation matrix for the numerical variables (**vis_cor** function).

Each of these functions returns a single **ggplot2** (Wickham, 2016) plot that shows a rectangular representation of the dataset where the expected information is denoted by colors. An example of this visualization\(^\text{18}\) can be seen in Figure 13.

The package includes a vignette *Using visdat* that provides examples for all package options. Interestingly, it is the only package that use solely visual means of exploring the data.

**The xray package**

The **xray** (Seibelt, 2017) package has three functions for the analysis of data prior to statistical modeling:

1. detecting anomalies: missing data, zero values, blank strings, and infinite numbers (**anomalies** function),
2. drawing and printing univariate distributions of each variable through histograms, bar plots and quantile tables (**distributions** function),
3. drawing plots of variables over time for a specified time variable (**timebased** function).

Examples are presented in the readme file in the GitHub repository of the project (https://github.com/sicarul/xray), but no vignette is attached to it. Plots\(^\text{19}\) generated by the package are presented in Figure 14.

**Other packages**

As mentioned before, there are numerous R packages that aim to make data exploration faster or the outputs more polished.

For table summaries of data that often include statistical tests, there are a few packages worth mentioning. The package **tableone** (Yoshida and Bohn., 2018) provides a `CreateTableOne` function to...\(^\text{18}\)Access the plot object with `archivist::aread("mstaniak/autoEDA-resources/autoEDA-paper/3cfd")`

\(^{19}\)Access the associated table with `archivist::aread("mstaniak/autoEDA-resources/autoEDA-paper/a3a3")`
Figure 13: Example output of the visdat::vis_guess function (visdat v. 0.5.3), which displays types of each value in the data frame and the missing values. We can see that the Age variable consists of integer values, even though it is coded as a character.

Figure 14: Example output from the xray::distributions function (xray v. 0.2). Such plots are created for each variable in the dataset along with a table of descriptive statistics.
make publication-ready tables referred to as Table 1 - traditional name of tables that describe patients' characteristics, usually stratified and including p-values from significance tests. The `describe` function from `describer` package (Hendricks, 2015) prints a summary of a `data.frame` or a vector which includes data types, counts and descriptive statistics. Similarly, the `skimr` (Quinn et al., 2019) package summarises data frames, vectors and matrices. It can also handled grouped data frames. The summary consist of data dimensions, missing and complete value counts, typical descriptive statistics and simple histograms. A function of the same name from `prettyR` (Lemon and Grosjean, 2018) returns descriptive statistics for each column in a `data.frame`. This package is focused on improving the aesthetics of R statistical outputs. Similarly, the package `Hmisc` (Harrell Jr et al., 2019) includes a `describe` function that displays typical descriptive statistics and number of unique and missing values for each column. The `plot` method called on the result of the `describe` function returns a dot plot for each categorical and a spike histogram for each continuous column. The scope of this package is bigger than just Exploratory Data Analysis, as it includes many tools related to regression models.

There are also many packages related to data visualization. Two of them are particularly worth mentioning. The `ggfortify` package (Tang et al., 2016) serves as a uniform interface to plots of different statistical objects, including PCA results that can be used for data exploration and time series plots. The `autoplotly` library (Tang, 2018) was built on top of `ggfortify` to provide automatically generated, interactive visualizations of many statistical models. While these two packages are focused on statistical modeling, they can be helpful in exploratory analysis and exemplify the potential of quick and interactive visualization in R.

Two more packages are relevant to our interest. `gpairs` (Emerson and Green, 2014) and `GGally` (Schloerke et al., 2018) packages implement the generalized pairs plot (Emerson et al., 2013). This type of plot extends well known scatter plot matrices, that visualize bivariate relationships for many variables, by handling both numerical and categorical variables. It is helpful in data exploration and shares similarities to `walls of histograms` that can be found in automated EDA libraries.

**Feature comparison**

In this section, we compare how different packages address autoEDA tasks as described in Section 2.1.1. A quick overview of the functionalities of different packages can be found in Table 4.

**Data description**

Almost all packages contain functions for summarizing datasets. Tools that support data validity analysis are less common.

**Whole dataset summaries**

Most packages that provide a whole dataset summary take a similar approach and present names and types of variables, number of missing values and sometimes unique values or other statistics. This is true for `summarytools` (dfSummary function), `autoEDA` (dataOverview function), `dataMaid` (makeDataReport result), `funModeling` (df_status function), `explore` (describe function), `ExPanDaR` (prepare_descriptive_table function), and `DataExplorer` (introduce function). These outputs are sometimes mixed with univariate summaries. That is the case for one of the most popular summary-type functions: the dfSummary functions from the `summarytools` package. An example is given in Figure 15.

In the `dlookr` package, summaries for numerical variables and categorical variables are only presented separately in the report (describe function).

The `visdat` package introduces the most original summaries of full dataset. The drawback of this approach is that it is not well suited for high dimensional data. But for a smaller number of variables, it gives a good overview of the dataset.

**Data validity**

Some packages can perform automated checks for the data, including at least outlier detection. The `dataMaid` package’s main purpose is to find inconsistencies and errors in the data. It finds possible outliers, missing values, low-frequency and possibly miscoded factor levels. All this information can be summarised in a quality report. The `dlookr` package covers similar functionality. There are two main differences: the report does not describe possibly miscoded factors, but outlier analysis is supplemented with plots showing variable distribution before and after removing the outliers. In all
cases, the analysis is rather simple, for example in zero-inflated variables non-zero values are treated as outliers (\texttt{dlookr}). The \texttt{ExPanDaR} package handles outliers by providing function that calculate winsorized or trimmed mean. Other packages only provide information about the number of missing values/outliers and identify columns that consist of a single value.

Data exploration

While multivariate analysis is rarely supported, there are many tools for descriptive and graphical exploration of uni- and bivariate patterns in the data.

Univariate statistics

All the tools that support univariate analysis take a similar approach to univariate analysis. For categorical variables, counts are reported and bar plots are presented, while histogram or boxplots and typical descriptive statistics (including quantiles, sometimes skewness) are used for continuous variables.

In \texttt{dataMaid} and \texttt{dlookr} packages, these plots are presented variable-by-variable in the report. In other packages (\texttt{DataExplorer}, \texttt{funModeling}, \texttt{SmartEDA}, \texttt{inspectdf}) groups of plots of the same type are shown together - as a wall of histograms or bar plots. Similarly, the \texttt{explore} package present all the plots at once. The \texttt{ExPanDaR} package allows user to choose variables to display in a shiny applications. Notably, \texttt{dlookr} reports skewness of variables and in case a skewed variable is found, it shows the distribution after some candidate transformations to reduce the skewness have been applied. This library also reports normality. The \texttt{SmartEDA} package also reports skewness and displays QQ plots against normal distribution, but it does not provide any means of reducing skewness.

Bivariate statistics

The \texttt{funModeling} and \texttt{SmartEDA} packages only support calculating correlations between variables and a specified target. \texttt{DataExplorer} and \texttt{visdat} packages can plot correlation matrices. They differ in categorical variables treatment. Some packages require only numerical features (\texttt{visdat}). Interestingly, in \texttt{DataExplorer}20, low-cardinality categorical features are converted to 0-1 variables and plotted alongside numerical variables, as seen in Figure 16.

The \texttt{arsenal} package only presents variable summaries by levels of a chosen categorical variable. The report from the \texttt{autoEDA} package consists of a limited number of bar plots/boxplots with target variable as one of the dimensions. Similarly, in \texttt{DataExplorer}, \texttt{dlookr}, \texttt{funModeling} and \texttt{SmartEDA}, scatter plots and box plots or histograms with a specified target variable on one of the axis can be plotted. Additionally, \texttt{funModeling} and \texttt{dlookr} draw histograms/densities of continuous features by the target. In shiny applications provided by \texttt{ExPanDaR} and \texttt{explore} packages, the user can choose target variables and explanatory variables to display bivariate plots. Interestingly, scatter plots provided by the \texttt{ExPanDaR} package can be extended to display multivariate dependencies by mapping variables to size and color of the points. The \texttt{funModeling} package also has unique

---

\textbf{Figure 15:} An example of whole data frame description that includes univariate summary and simple graphics. Created with the \texttt{dfSummary} function (\texttt{summarytools} package v. 0.9.3).
### Data cleaning and data transformation

The `dataMaid` package assumes that every decision regarding the data should be made by the analyst and does not provide any tools for data manipulation after diagnosis. Most of the packages only provide exploration tools. Exceptions are `dlookr`, `funModeling`, `DataExplorer` and `exploreR`. `DataExplorer` provides tools for normalization, imputation by a constant, merging levels of factors, creating dummy variables and transforming columns.

The `dlookr` package can create a report that presents different possible transformations of features. Missing values can be imputed by mean/median/mode and distributions of variables before and after the procedure can be compared. The same is done for imputation of outliers. Logarithmic and root square transforms are proposed for skewed variables. Different methods of binning continuous variables are also presented, including Weight of the Evidence.

The `funModeling` package can perform discretization of a variable using an equal frequency criterion or gain ratio maximization. It can also scale variables to the interval \([0, 1]\). Outliers can be treated using the Tukey or Hampel method.

### Reporting

`DataExplorer`, `dlookr`, `dataMaid`, `SmartEDA`, `explore` and `RtutoR` have an option of generating a report and saving it to a file. Such a report usually consists of all or most possible outputs of the package. The plots and summaries are organized by the exploration task (for example univariate, then bivariate analysis) and either simply variable-by-variable (`dataMaid`, `dlookr`) or grouped by variable type (`DataExplorer`, `SmartEDA`). The `autoEDA` package generates a minimal report with bivariate plots. Packages `arsenal`, `funModeling`, `xray`, `summarytools` and `exploreR` have an option of saving outputs - plots or tables - to files.

### Discussion

Automated EDA can be either directed towards a general understanding of a particular dataset or be more model-oriented, serving as a foundation for good modeling. While presented packages include some tools related to simple variable transformations, they are more focused on data understanding.

---

**Figure 16:** Correlation plot as returned by the `DataExplorer::plot_correlation` function.
For this task, they have many advantages. In this section, we summarize the strong points of existing tools and point out some possible improvements and new directions for autoEDA.

Strengths of autoEDA packages

1. The packages dlookr, dataMaid, DataExplorer, SmartEDA are capable of creating good quality reports.
2. DataExplorer has very good visualizations for PCA.
3. DataExplorer handles categorical variables on correlation plots by creating dummy features, which is a unique idea compared to other packages.
4. The visdat package, while probably not the best choice for high dimensional data, features interesting take on initial whole dataset exploration.
5. The dlookr package is capable of selecting skewed variables and proposing transformations. Some of the other packages display binned continuous variables, which can also help in seeing visualizing dependencies.
6. dataMaid is a good tool for finding problems in the data. Thanks to the structure of check and summarize functions results, discovered issues can be treated effectively.
7. For datasets with a moderate number of features, DataExplorer, funModeling, dlookr and SmartEDA give a reasonable insight into variables distributions and simple relationships.
8. SmartEDA package provides a method of visualizing multivariate relationships - parallel coordinate plot.

We can see that tasks related to data quality and whole dataset summary are well by the existing libraries. Getting the big picture of the data and finding possible data quality problems is easy, especially with the dataMaid package. For classical applications, for example, statistical analyses in medicine, the current tools provide very good tables, such as the ones from tableone or arsenal packages, and uni-/bivariate plots. The inspectdf and summarytools packages can also provide quick insights into a dataset. Univariate analysis can be performed either variable-after-variable (dlookr, dataMaid), where we can see the statistical properties of each variable, or as groups of plots based on variable type (DataExplorer, funModeling). Both ways can be useful for a reasonable number of predictors. While multivariate tools are scarce, the available tools, PCA in DataExplorer and PCP in SmartEDA, are very well done. Notably, the ExPanDaR package provides very high flexibility thanks to the possibility of interactively choosing variables to display, adding new variables on-the-fly and customizing plots in the shiny application.

Future directions and possible improvements

The field of autoEDA is growing. New packages are being developed rapidly - there are recent additions from April and May. Features are added to existing packages and bugs are corrected, as new issues are suggested by users on GitHub. At this moment, we can identify the following problems and challenges.

All the presented tools can fail in situations with imperfect data. In particular, they are usually not robust to issues like zero-variance/constant variables. Such problems are expected to be solved in the nearest future, as suggested for example by issues in the GitHub repo of the DataExplorer package. In general, error messages can be uninformative. Moreover, in some situations, they lack flexibility. For example, in DataExplorer arguments can be passed to cor function, but not to corrplot function.

In case of walls of histograms (or bar plots), no selection is being done and no specific order is chosen to promote most interesting distributions. The same is true for automatically created reports. This problem is only addressed by the RtutoR package, which allows to select top k relevant variables. Moreover, for high-dimensional data or high-cardinality factors, the plots often become unreadable or impractical. Partial solutions to this problems are applied, for example DataExplorer removes too large factors from the panels. More generally, many GitHub issues for the described packages are related to customizing and improving plots and output tables. It is a challenging task due to the diversity of possible input data and a major concern for developers of autoEDA packages.

Typical EDA tasks are limited to exploring bivariate relationships. Searching for higher dimensional dependencies would be interesting, for example by adding color and size dimensions to the plots, which was already done in the ExPanDaR package. For wall of plots type of display, such an addition would result in a large number of new plots. Thus, it would require a proper method of
finding the most relevant visualization. Interactivity partially helps address this issue. PCA, parallel coordinate plots and model summaries are supported, but each by a separate package. It is evident that there is a shortage of multivariate tools. Univariate regression models can be plotted by the `exploreR` package. The `explore` package plots decision trees for binary target variables. In other cases, exploration based on simple statistical models (such as scatter plot smoothing) is not an option. Using regression models and feature transformations to identify and measure relevant relationships could improve bivariate or multivariate analyses supported by automated EDA.

Regarding variable transformation, only one of the packages addresses the issue of skewed variables. Proposing transformations of continuous features other than binning would be helpful and could improve visualizations, for example, scatter plots with skewed variables. Missing data imputation more advanced than imputing a constant is delegated to other packages, although, it is known that imputation by a constant is usually not the best method of missing values treatment. Some of the above issues limit the packages’ usefulness in iterative work. Though, the comparisons of transform and original features and the possibility of applying discovered transformations to data in `dlookr` package are steps in the right direction.

Support for time-varying variables and non-classical (not IID) problems such as survival analysis is limited or non-existent. For survival analysis, the automation level is low, but there are two notable tools for summarizing dependencies. First is the recognized package survminer (Kassambara and Kosinski, 2018), which helps visualize survival curves, while also displaying survival tables and other information. The other tool is the cri7 package (Młynarczyk and Biecek, 2017), which includes summarizeCR function that returns several tables and plots for competing risks analysis. More tools for fast visualization of at least bivariate relationships in such problems would be a big help for analysts. Cluster analysis is sometimes regarded a part of the EDA process, but it is not available in any of the packages.

The tools available in R have similar range to other languages’ libraries, for example from Python. Python packages such as Dora (Epstein, 2017) or lens (Zabalza and Engineers, 2018) also cover feature-by-feature descriptive statistics and plots, bivariate visualizations of the relationships between predictors and target variable, contingency tables, basic data transformations, and imputation. Tools for visual data exploration supports also tools for visual model exploration like DALEX (Biecek, 2018) or iml (Molnar et al., 2018). In both cases visual summaries help to quickly grasp key relations between variables or between input features and model predictions.

Since EDA is both closely connected to feature engineering and based on visual insights, automated EDA can draw from existing tools for automated feature extraction like SAFE ML (Gosiewska et al., 2019) or TPOT (Olson et al., 2016) and visualization recommendations. When it comes to aiding visual exploration of a dataset, standalone software carries possibilities beyond what we can expect from R packages or analogous libraries in other languages. A recent notable example is DIVE (Hu et al., 2018). It is an example of a growing number of tools for visual data exploration that aim to distinguish between relevant and irrelevant visualization and help the analyst find the most interesting plots. DIVE is one of the mixed-initiative visualization systems, meaning it uses both statistical properties of the dataset and user interactions to find the relevant plots. Building recommendation systems into autoEDA tools can help address the issue of dealing with high-dimensional data and multivariate dependencies by letting the ML-based system deal with the complexity of a large number of candidate visualizations. AI-assisted data exploration can be even faster and more efficient.

As autoEDA tools are still maturing, the efforts in the field are somewhat fragmented. Many packages try to achieve similar goals, but they can be quite inconsistent. It is especially visible in the multiplicity of names for the summary-type function to describe a whole data frame. As the libraries develop, new standards and conventions should be proposed.

### Acknowledgement

This work was financially supported by the NCN Opus grant 2016/21/B/ST6/02176.

### Bibliography


P. Biecek and M. Kosinski. archivist: An R package for managing, recording and restoring data analysis


Mateusz Staniak
Faculty of Mathematics and Information Science
Warsaw University of Technology
Poland
m.staniak@mini.pw.edu.pl

Przemyslaw Biecek
Faculty of Mathematics, Informatics and Mechanics
University of Warsaw
Poland
ORCiD: 0000-0001-8423-1823
przemyslaw.biecek@gmail.com
<table>
<thead>
<tr>
<th>Task type</th>
<th>Task</th>
<th>a</th>
<th>aE</th>
<th>DE</th>
<th>dM</th>
<th>d</th>
<th>EPD</th>
<th>e</th>
<th>eR</th>
<th>fM</th>
<th>i</th>
<th>R</th>
<th>SE</th>
<th>s</th>
<th>v</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset</td>
<td>Variable types</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Dimensions</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Other info</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compare datasets</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Validity</td>
<td>Missing values</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Redundant col.</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Outliers</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Atypical values</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Level encoding</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Univar.</td>
<td>Descriptive stat.</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Histograms</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Other dist. plots</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bar plots</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>QQ plots</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bivar.</td>
<td>Descriptive stat.</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Correlation matrix</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>1 vs each corr.</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Time-dependency</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bar plots by target</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Num. plots by target</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Scatter plots</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Contingency tables</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other stats. (factor)</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Multivar.</td>
<td>PCA</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stat. models</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Transform.</td>
<td>Imputation</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Scaling</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Skewness</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Outlier treatment</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Binning</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Merging levels</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reporting</td>
<td>Reports</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>Saving outputs</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 4: Overview of functionalities of all described packages. Package names were shortened to make the table as compact as possible. a denotes arsenal, aE - autoEDA, DE - DataExplorer, dM - dataMaid, d - dlookr, EPD - ExPanDaR, e - explore, eR - exploreR, fM - funModeling, i - inspectdf, R - RtutoR, SE - SmartEDA, s - summarytools, v - vis dat, x denotes xray. Num. plots by target refers to either histogram, density, violin or box plot.