

|  |
| --- |
| D1.4 Open-source library for data mining techniques |
|  |
| D 1.4 | 06.09.2023 |

***Logo, company name

Description automatically generated***

Deliverable No. 1.4 Open-source library for data minig techniques

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project Acronym** | **Grant Agreement #** | **Project Title** | **Deliverable Reference #** | **Deliverable Title** |
| I4Driving | 101076165 | Integrated 4D Driver Modelling under Uncertainty | 1.4 | Open-source library for data mining techniques |

**AUTHORS**

|  |  |
| --- | --- |
| Giulia Vannucci | Institute for Cognitive Sciences and Technologies (Cnr-ISTC) |
| Andrea Saltelli | Institute for Cognitive Sciences and Technologies (Cnr-ISTC) |
| Roberta Siciliano | University of Naples Federico II |
| Antonio D’Ambrosio | University of Naples Federico II |
| Michele Staiano | University of Naples Federico II |

**INTERNAL REVIEWERS**

|  |  |
| --- | --- |
| Vincenzo Punzo | University of Naples Federico II |

**DISSEMINATION LEVEL**

|  |  |  |
| --- | --- | --- |
| X | P | PUBLIC |
|  | C | CONFIDENTIAL |

This project has received funding from the European Union’s Horizon Europe programme, under grant agreement No 101076165.

**Disclaimer**

Funded by the European Union. Views and opinions expressed in this publication are, however, those of the author(s) only and do not necessarily reflect those of the European Union or CINEA. Neither the European Union nor the granting authority can be held responsible for them.

|  |  |
| --- | --- |
|  | Funded by the Horizon 2020 programme of the European Union  Grant Agreement No 101076165 |

Version History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Revision | Date | Authors | Organisaton | Description |
| Original | August 30 | Giulia Vannucci, Andrea Saltelli | CNR | Level 0 draft |
| 0.1 | September 4 | Roberta Siciliano, Antonio D’Ambrosio, Michele Staiano | UniNA | Integration |
| 0.2 | September 6 | Giulia Vannucci | CNR | Revision |
|  |  |  |  |  |

# Executive Summary

The major scientific challenge of the **i4Driving Project** is a human driver model that captures the relevant behavioral mechanisms for safety assessment. For the Driving Behavioral Analysis, Data Mining and Machine Learning must be considered. In this deliverable, we suggest and describe a list of available open-sources Language Programs and Libraries that are useful to implement data mining techniques. The present deliverable is integrated by a GitHub repository that contains proper linkages to the resources described here. The GitHub will be a living entity, and will register worked cases from i4Driving as these are developed in the course of the project.

Contents

[Executive Summary 3](#_Toc144885640)

[1. Introduction 5](#_Toc144885641)

[2. Open-source Software 5](#_Toc144885642)

[2.1 Python Software 5](#_Toc144885643)

[2.1.1 Install Python 5](#_Toc144885644)

[2.1.2 Text editors and integrated development environments (IDEs) available 6](#_Toc144885645)

[2.1.3 Install packages 6](#_Toc144885646)

[2.2 R Software 7](#_Toc144885647)

[2.2.1 Install R 7](#_Toc144885648)

[2.2.2 Text editors and IDEs available 7](#_Toc144885649)

[2.2.3 Install packages 7](#_Toc144885650)

[2.3 Interfacing Python and R 8](#_Toc144885651)

[3. Libraries for data mining techniques 8](#_Toc144885652)

[3.1 Data manipulation and visualization 9](#_Toc144885653)

[3.1.1 Python libraries for data manipulation and visualization 9](#_Toc144885654)

[3.1.2 R libraries for data manipulation and visualization 9](#_Toc144885655)

[3.2 Unsupervised learning 9](#_Toc144885656)

[3.2.1 Python libraries for unsupervised learning 9](#_Toc144885657)

[3.2.2 R libraries for unsupervised learning 10](#_Toc144885658)

[3.3 Supervised learning 11](#_Toc144885659)

[3.3.1 Python libraries for supervised learning 11](#_Toc144885660)

[3.3.2 R libraries for supervised learning 12](#_Toc144885661)

[4. Means for interoperability with other simulation software 12](#_Toc144885662)

[5. Conclusions 13](#_Toc144885663)

[References 13](#_Toc144885664)

# Introduction

This report describes a selection of Open-source library for data mining techniques, which are also linked on GitHub repository “i4Driving-OpenSourceLibaryDataMining” (<https://github.com/Giuliana87/i4Driving-OpenSourceLibaryDataMining>), keeping in mind that research in the area of data mining techniques and thus machine learning is constantly evolving. We believe, however, that the software, the libraries and other resources that we suggest in this report are a good selection out of the vast amount of available material. Moreover, the repository will contain the codes developed within the i4Driving project, and will thus grow with time as new analyses are carried out.

In Section 2, we briefly describe two of the most widely used open-source software, Python and R programming languages within their capability to interface each other; in Section 3 we describe some of the most useful libraries for data mining techniques; in Section 4, the interoperability with traffic simulation software is briefly described; in Section 5, the conclusions are drawn.

# Open-source Software

Data Mining is the study of collecting, processing and analysing large volumes of data in order to discover meaningful patterns, trends, correlations, or insights. It implies analysing data patterns in large batches of data using one or more software. The choice of the software to be used is strongly linked to several issues: better efficiency in organising data, better computational performance, technical choices linked to the machine where the software is to run (i.e. the operating system or the physical memory of the available computer), the need to work online or offline, the actual availability of the algorithms to be used on that software. In this section, we briefly describe how to install two of the most popular open-source software available, along with the most widely text editor and IDEs used, and how to install libraries from the relative software.

For the needs of scientific computation in the framework of i4Driving the main approach to developing and deploying computational tools, especially oriented to the task of statistical data analysis, relies on scripting languages. Python and R are the most common choices aimed to this end. Additionally the two languages can be interfaced with one another, and their environments can be made to effectively exchange data structures. This can be achieved by means of several Integrated Development Environments (IDE) packages. Thus, Python and R are presented in this deliverable as synergistic instead of antagonistic tools.

## Python Software

Python (<https://www.python.org>) is a high-level, general-purpose programming language. It is a a multi-paradigm programming language, where object-oriented programming and structured programming are fully supported.

### Install Python

The latest Python source distribution is always available from python.org at <https://www.python.org/downloads/>. The latest development sources can be obtained at <https://github.com/python/cpython/>. The standard documentation for the current stable version of Python is available at <https://docs.python.org/3/>.

### Text editors and integrated development environments (IDEs) available

Using an Integrated Development Environment (IDE) or text editor is not an absolute requirement, but it is often recommended. Integrated Development Environments (IDEs) are coding tools that make writing, debugging, and testing the code easier. Many provide helpful features like code completion, syntax highlighting, debugging tools, variable explorers, visualization tools, and many other features. Here, some of the most widely used IDEs and Text editor for Python.

* **Jupyter Notebook / JupyterLab (**[**https://jupyter.org**](https://jupyter.org)**):** Jupyter Notebook and its updated version JupyterLab are interactive environments (both released under a [modified BSD license](https://opensource.org/licenses/BSD-3-Clause)) widely used for data science and scientific computing not only in Python, since Project Jupyter's name is a reference to the three core programming languages supported by Jupyter, which are Julia, Python and R. They allow you to combine code, visualizations, and narrative text in a single document.
* **Spyder (**[**https://www.spyder-ide.org**](https://www.spyder-ide.org)**):** Spyder is an IDE (released under the MIT License) specifically designed for scientific computing with Python. It offers features like an integrated IPython console, variable explorer, and support for data visualization libraries. It integrates with a number of prominent packages in the scientific Python stack.
* **Atom (**<https://github.com/atom/atom>**):** Atom is a customizable and free text editor (released under the MIT License) developed by GitHub with support for plug-ins written in JavaScript, and embedded Git Control. It can be extended using packages to support Python development.

It is worth to mention two more very common IDEs, even if they are not fully open source and free projects, since it is possible to exploit some FLOSS versions of them:

* **PyCharm (**[**https://www.jetbrains.com/pycharm/**](https://www.jetbrains.com/pycharm/)**)**: PyCharm is a powerful commercial IDE used for programming in Python. It offers code analysis, debugging, integrated testing, and support for web development frameworks, scientific libraries, and more. An academic free license is available to qualified users. Moreover, the Community edition is Free and built on open-source code released under Apache License 2.0.
* **Visual Studio Code (VS Code,** [**https://code.visualstudio.com**](https://code.visualstudio.com)**):** VS Code is a lightweight and highly customizable code editor by Microsoft. It has a strong Python extension ecosystem, providing features like debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add functionality. Microsoft’s VS Code source code is open source (MIT-licensed), but the product available for download (Visual Studio Code) is licensed under [a not-FLOSS license](file:///C:\Users\mike\Downloads\a%20not-FLOSS%20license) and contains telemetry/tracking. VSCodium (<https://vscodium.com/>) is a community-driven, freely-licensed binary distribution of Microsoft’s editor VS Code (released under the MIT License) and obtaining its binaries guarantees that telemetry is disabled.

### Install packages

The installation of packages in Python can be done by opening the Command Prompt or terminal and use the *pip install* command followed by the package name. The detailed instruction can be founded at <https://packaging.python.org/installing/>

There are also package and environment management systems alternative to pip (or its last version pip3) mainly based on conda (released under BSD 3-Clause License) or derived from it. The most common platform for scientific computation is Anaconda (<https://www.anaconda.com/download>) and it is highly recommended to resort on it for smoothly setting up a Data Science workstation.

## R Software

R (<https://www.r-project.org>) is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. It provides a wide variety of statistical techniques (i.e. linear and nonlinear modelling, statistical tests, time-series analysis, classification, clustering, and machine learning algorithms) and graphical techniques, and is highly extensible. R has gained immense popularity in the fields of statistics, data analysis, and data visualization due to its rich set of built-in functions and packages tailored for these tasks. R provides an Open Source route to participation in that activity.

### Install R

The latest R source distribution is available from CRAN Mirrors at <https://cran.r-project.org/mirrors.html>, where one has to choose the closest location and download the desired version of R.

### Text editors and IDEs available

As mentioned for Python, the usage of a text editor or an IDE for R is suitable. Some of the most widely used IDEs and Text editor for R are listed below.

* **RStudio (**[**https://posit.co/download/rstudio-desktop/**](https://posit.co/download/rstudio-desktop/)**):** RStudio is a widely-used and feature-rich IDE (released under the AGPL v3 license) designed specifically for R programming. It provides an intuitive interface for writing R code, managing packages, and creating visualizations. It is worth mentioning that by free registration at posit.cloud it is possible to use at no cost R by a cloud instance of RStudio with a limited set of resources – that can be enhanced on subscription at need. Posit (formerly RStudio) also offers a commercial desktop license for RStudio, called RStudio Desktop Pro, and a commercial license for Posit Workbench (previously RStudio Workbench/RStudio Server Pro).
* **Jupyter Notebook / JupyterLab:** as mentioned for Python, Jupyter Notebook and JupyterLab (both released under a [modified BSD license](https://opensource.org/licenses/BSD-3-Clause)) support R kernels, enabling you to create interactive documents with R code and visualizations.

As in the case of Python, some IDEs exist that are well suited and can be used for free even if not released as open-source software:

* **Visual Studio Code (VS Code):** as mentioned for Python, VS Code, as well as VSCodium, has extensions that allow you to work with R. It provides features like syntax highlighting, debugging, and integrated terminal support for R.
* **Tinn-R (**[**https://tinn-r.org/en/**](https://tinn-r.org/en/)**):** Tinn-R is an editor/word processor ASCII/UNICODE generic for the Windows operating system, very well integrated into the R, with R specific features including syntax highlighting, code completion, and integration with R's help system. The Tinn-R program remains free for use in the education sector at any level.

### Install packages

In R, the installation of packages is straightforward and can be done using the *install.packages()* function. More detailed instruction can be found at <https://cran.r-project.org/doc/manuals/r-patched/R-admin.html#Installing-packages>.

## Interfacing Python and R

There are two options to exploit the best from both languages: calling R code from a Python script or invoke Python code from within the R environment.

When Python is preferred as the main language, e.g. since its good capabilities for data manipulation and integration with existing code for side tasks, it is useful to perform data mining analyses by running R code from within Python processes (via the rpy2 Python library, GNU General Public License Version 2, <https://rpy2.github.io/doc/latest/html/introduction.html> ). This offers the following advantages:

1. Access to a vast number of statistical algorithms and packages: the R community has developed a vast number of packages for statistical analysis and data science. By using rpy2, Python users can tap into this extensive library and leverage the well-tested algorithms and methodologies implemented in R.

2. Seamless integration with Python: rpy2 provides a seamless interface between Python and R, allowing users to easily move between the two languages. This enables ‘Pythonistas’ to take advantage of R's statistical capabilities while still being able to utilize the wide range of Python libraries for tasks such as data manipulation, visualization, and web development.

3. Compatibility with existing R code and workflows: For users who already have code or workflows written in R, rpy2 allows them to reuse and integrate their R code into Python processes. This eliminates the need to rewrite or replicate existing R code, saving time and effort.

4. Flexibility and versatility: By combining the strengths of both Python and R, users can leverage the best features of each language for their analysis or data science tasks. Python provides a powerful and expressive programming language, while R offers a rich ecosystem of statistical functions and packages. This combination of languages gives users the flexibility to choose the best tool for each part of their workflow.

5. Support and community: The R community has a long history and a large user base, which means that there is a wealth of resources and support available by R coders and users for many data mining task. By using rpy2, Python users can tap into this community and benefit from the knowledge and expertise of R users.

Overall, rpy2 provides a powerful and convenient way for Python users to harness the capabilities of R's statistical packages and algorithms, while still enjoying the flexibility and functionality of the Python ecosystem.

On the other hand, if R is the customary environment for the analysis, the R package reticulate offers the possibility of seamlessly integrate Python code chunks in an R notebook or markdown file managed by a suitable IDE (like RStudio).

# Libraries for data mining techniques

There are several important open-source libraries for data mining techniques for Python and R. First, we briefly describe the methods most used for unsupervised and supervised learning, and then we listed some of the most widely used libraries. For a comprehensive description of the statistical models and machine learning methods mentioned in this section, the reader is referred to the reading of Anderson et al. (1958), Agresti (2015), McCullagh, P. (2019), Hastie, Tibshirani et al. (2021, 2023), Saltelli et al. (2004, 2008).

## Data manipulation and visualization

As a crucial step of every type of analysis is an efficient manipulation of data and a good visualization of the results, we recall some useful libraries for these steps.

### Python libraries for data manipulation and visualization

For data manipulation, Pandas, NumPy and scipy are some very important libraries for data manipulation and data analysis. They provide data structures and functions needed to effectively manipulate and analyse structured data. For data visualization, ggplot (ggpy) is the counterpart of R library ggplot2, and it provides a high-level interface for creating complex visualizations.

### R libraries for data manipulation and visualization

Packages dplyr and data.table are excellent for data manipulation, the latter is strongly efficient in handling large datasets. For data visualization, ggplot2 it is the most widely used package for plotting and visualizing data. It provides helpful commands to create complex plots from data in a data frame and a more programmatic interface for specifying what variables to plot, how they are displayed, and general visual properties

## Unsupervised learning

Algorithms of unsupervised learning have the objective to identify patterns and structures in data and the relationship among the features without a target variable to predict or classify. Common tasks in unsupervised learning include Association rules mining, clustering, where data points are grouped based on similarities, and dimensionality reduction, where the algorithm reduces the complexity of the data while preserving important features. Among dimensionality reduction methods, the most widely used methods are the Principal Component Analysis, Correspondence Analysis and Multiple Correspondence Analysis, Non-linear factorial methods, Multidimensional Scaling Methods.

### Python libraries for unsupervised learning

* scikit-learn: this is one of the most popular libraries for machine learning in Python. Among clustering algorithms, if offers from sklearn.cluster: KMeans, AgglomerativeClustering for hierarchical clustering, and DBSCAN for density-based clustering. Among dimensionality reduction techniques, it offers PCA from sklearn.decomposition for Principal Component Analysis (PCA), t-SNE from sklearn.manifold, t-Distributed Stochastic Neighbor Embedding that is useful for visualizing high-dimensional data by reducing it to a lower-dimensional space. If the work involves multi-label classification tasks, Scikit-multilearn provides options like MLkNN, which combines multi-label k-nearest neighbors and dimensionality reduction.
* Apache Spark: it is a fast and general-purpose cluster computing system that also includes libraries for machine learning (MLlib) and graph processing (GraphX). It's well-suited for big data processing, Exploratory Data Analysis (EDA), feature extraction and Machine Learning models.
* PyCaret: it is a low-code machine learning library that automates machine learning workflows. It is an end-to-end machine learning and model management tool that exponentially speeds up the experiment cycle and makes you more productive. Compared with the other open-source machine learning libraries, PyCaret is an alternate low-code library that can be used to replace hundreds of lines of code with a few lines only, which makes experiments exponentially fast and efficient. This repository works as a wrapper around several machine learning libraries and frameworks, such as scikit-learn for unsupervised and supervised learning, XGBoost, LightGBM and CatBoost for boosting algorithms, Optuna for an automatic hyperparameter optimization software framework, and others. While it's a high-level machine learning library, PyCaret also provides support for various clustering algorithms, making it easier to experiment with different techniques.
* Yellowbrick: it is a suite of visual analysis and diagnostic tools designed to facilitate machine learning with scikit-learn. In particular, it focuses on visualization tools to understand the effect of dimensionality reduction and tools to help visualize, i.e., clustering results.
* Prince: this library specializes in factor analysis and dimensionality reduction, offering implementations of various techniques, including PCA and correspondence analysis. It provides efficient implementations, using a scikit-learn API.

### R libraries for unsupervised learning

* arules: it is focused on association rule mining, a technique that identifies interesting relationships or associations between items in a dataset. This package provides tools needed to create and manipulate input data sets for the mining algorithms and for analysing the resulting item sets and generating association rules
* cluster: it provides various clustering algorithms, including hierarchical clustering, K-Means clustering, and more advanced techniques like PAM (Partitioning Around Medoids) and CLARA (Clustering Large Applications).
* dbscan: it provides an implementation of the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm, which is effective for identifying clusters of varying shapes and sizes.
* mclust: it is focused on model-based clustering, and it offers tools for fitting Gaussian Mixture Models to the data, allowing the identification of clusters with different shapes and orientations.
* clustermesh: it offers tools for hierarchical clustering using dendrograms, including the possibility of creating clustered heatmaps and visualizing the results.
* dendextend: it is a versatile package for extending the functionality of dendrograms, providing additional options for manipulating and visualizing hierarchical clustering results.
* NbClust: it is designed to determine the number of clusters in a dataset, and it offers a comprehensive suite of indices for cluster validation
* factoextra: it is a helpful tool for extracting and visualizing information from various multivariate analysis methods. This package provides functions to visualize the results of clustering algorithms, including hierarchical clustering and K-Means, and includes plotting cluster assignments, dendrograms, and more. It offers tools to visualize the outcomes of dimensionality reduction techniques like PCA (Principal Component Analysis) and t-SNE (t-Distributed Stochastic Neighbor Embedding), to visualize the factor analysis results, including factor loadings and factor correlations, and it provides heatmap functions that can display clustering results and PCA outcomes.
* FactoMineR: this comprehensive package offers a range of multivariate analysis techniques, including Principal Component Analysis (PCA), Multiple Correspondence Analysis (MCA), and Factor Analysis of Mixed Data (FAMD).

## Supervised learning

In supervised learning, the algorithm is provided with input data (the set of explanatory variables) and corresponding target outputs (the set of response variables), allowing it to learn how to map inputs to desired outputs. When the target is numerical, we deal with a regression problem, instead when the target is categorical with a set of response classes, we deal with a classification problem. The goals of the models of supervised learning are to identify the best model capable of understanding how the response variable is related to the explanatory variables and to make accurate predictions. Among the supervised learning models, we recall linear regression, logistic regression and discriminant analysis. With high dimensional data, we recall LASSO, RIDGE, ELASTIC-NET Regression, or dimension reduction methods such as Principal Component Regression or Partial Least Squares Regression. Among nonparametric models we recall Regression Splines, Smoothing Splines, and Local Regression. Among algorithms based on decision trees the most popular are Classification and Regression Trees, Boosting, Random Forests and Bayesian Additive Regression Trees (BART). Finally, we recall Deep Learning with Neural Networks. Additionally, we suggest some library for Sensitivity Analysis.

### Python libraries for supervised learning

* statsmodels: it is a library for statistical modeling and hypothesis testing, which can be useful for supervised learning when interpretability and inference are important. It provides a complement to scipy for statistical computations including descriptive statistics and estimation and inference for statistical models.
* scikit-learn: as already mentioned above, this is one of the most popular libraries for machine learning in Python. It covers a wide range of supervised learning algorithms, including linear and logistic regression, penalized regression, PCA, decision trees, random forests, support vector machines, and more. It also provides various tools for model fitting, data pre-processing, model selection, model evaluation, and many other utilities.
* XGBoost: Short for "Extreme Gradient Boosting" it is a powerful machine learning algorithm for boosting models known for its speed and performance.
* LightGBM: it is another gradient-boosting framework that is designed to be efficient and scalable. It's particularly useful for large datasets and is known for its speed and accuracy.
* CatBoost: a gradient boosting library that is capable of handling categorical features effectively without extensive preprocessing.
* TensorFlow: it is an open-source deep learning library that provides a flexible framework for building and training various neural network models. It's particularly well-suited for tasks involving neural networks and deep learning.
* PyTorch: it is another popular deep learning library. It's known for its dynamic computation graph and is widely used for research in machine learning and artificial intelligence.
* SAlib: it is a library containing implementations of commonly used sensitivity analysis methods, including Sobol, Morris, and FAST methods, useful in systems modeling to calculate the effects of model inputs or exogenous factors on outputs of interest.

### R libraries for supervised learning

* stats package is one of the core packages in R and comes pre-installed with R itself. It provides a wide range of statistical functions, including basic statistical tests, linear and nonlinear modelling, regression, and much more.
* kknn: it implements k-nearest neighbours (k-NN) algorithms, which is particularly useful for classification tasks based on similarity metrics.
* glmnet: it is particularly popular for fitting Lasso and Ridge regression models, and it offers efficient implementations of these regularization techniques.
* rpart and tree: these two packages are the most widely used for fitting classification and regression trees.
* party: it implements recursive partitioning methods, including the C4.5 algorithm and conditional inference trees, which can be used for decision tree-based models.
* caret: it is a versatile and powerful tool for training, tuning, and evaluating machine learning models. It provides a unified interface for working with a wide range of algorithms, preprocessing techniques, and model evaluation methods. It supports parallel processing, which can significantly speed up the training and tuning process.
* randomForest: it implements the random forest algorithm, an ensemble learning method that's effective for classification and regression tasks. It's known for its robustness and ability to handle high-dimensional data.
* ranger: it is a fast and efficient implementation of random forest algorithm.
* xgboost: it offers an implementation of the extreme gradient boosting algorithm, which is highly efficient and performs well on a variety of machine learning tasks.
* gbm: The gbm (Generalized Boosted Regression Models) package provides an implementation of gradient boosting, a machine learning technique for regression and classification tasks.
* nnet: it is used for training and evaluating neural networks. While more specialized deep learning frameworks are available (i.e. TensorFlow, Keras of Python), nnet offers a simpler approach for basic neural network tasks.
* e1071: it provides a collection of functions for support vector machines (SVMs), as well as other machine learning algorithms for classification and regression.
* sensobol: it is a package that provides several functions to conduct variance-based uncertainty and sensitivity analysis, from the estimation of sensitivity indices to the visual representation of the results.

# Means for interoperability with other simulation software

The software for traffic simulation used in the course of the i4Driving project could be easily interfaced to the code deployed for data mining through the open-source solutions so far described.

From the very simple interfacing of data mining via an interoperable file format (e.g. as straightforward as using csv files for data exchange) to the tighter integration made possible by exploiting statistical procedures’ call from within the simulation software.

Interfacing to Python software is really a straightforward task (see for a wide set of languages the official documentation: https://wiki.python.org/moin/IntegratingPythonWithOtherLanguages).

About R code, it can directly be invoked from within Python processes by running an R script as a subprocess (https://docs.python.org/3/library/subprocess.html#module-subprocess) and collecting its output or by exploiting suitable libraries, as already discussed (see rpy2 in section 2.3).

For software written in Java, Renjin (https://www.renjin.org/) is an open-source library (released under GNU GENERAL PUBLIC LICENSE Ver. 2) that easily adds to Java projects the capability to spawn an R process allowing zero-overhead data sharing between R and Java.

# Conclusions

This deliverable describes Open-source software and library for data mining techniques. The two most widely used software in the field of data science, statistics, and analytics, are Python and R, and for this reason we suggest and describe them in this deliverable. Each language has its strengths and weaknesses, and their potential depends on various factors, including the specific use case and your preferences. However, the two software are appreciated for their versatility, their capability to do machine learning and deep learning task, their capability of interoperability with each other and with other languages, their community and support, thanks to which one can find extensive documentation, tutorials, and support online, and their support for creating reproducible research, which is crucial in scientific context. Within both software, there is an extensive choice of libraries and frameworks available for various tasks, and in this report we concentrate our attention to the most used library for data mining, bearing in mind that there is a huge choice of packages to perform the same task and that new packages are developed and made available practically every day, so it is crucial for the user to keep up to date.

# References

Anderson, T. W., Anderson, T. W., Anderson, T. W., & Anderson, T. W. (1958). An introduction to multivariate statistical analysis (Vol. 2, pp. 1468-1482). New York: Wiley.

Agresti, A. (2015). Foundations of linear and generalized linear models. John Wiley & Sons.

McCullagh, P. (2019). Generalized linear models. Routledge.

Hastie, T., Tibshirani, R., & Friedman, J. (2009) The Elements of Statistical Learning. Second Ed., Springer. New York,

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An introduction to statistical learning. Second Ed., Springer, New York, NY.

James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). An Introduction to Statistical Learning - with Applications in Python. Springer Cham.

Saltelli, A., et al. (2004) *Sensitivity analysis in practice: a guide to assessing scientific models*. Vol. 1. New York: Wiley.

Saltelli, A., et al. (2008) *Global sensitivity analysis: the primer*. John Wiley & Sons.