Nemoa

Release 0.5.581

Patrick Michl

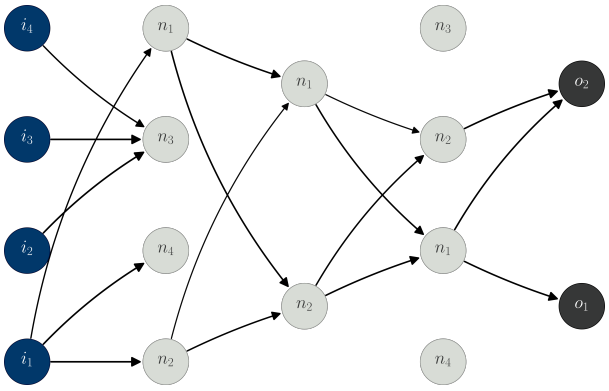
Mar 26, 2019

Contents

CHAPTER 1

Introduction

In many domains of enterprise and scientific data analysis the lack of structural knowledge encourages the use of machine intelligence methods, which surpass the structural flexibility of classical statistical methods. Since this approach, however, generally is paid by high mathematical uncertainty, it is indispensable to properly select and adapt the method to it's respective application. This requires a close cooperation between enterprise analytics and data science.



Nemoa is a data analysis framework for collaborative data science and enterprise application. The key goal of the project is to provide a long-term data analysis framework, which seemingly integrates into existing enterprise data environments and thereby supports collaborative data science. To achieve this goal nemoa orchestrates established frameworks like TensorFlow® and SQLAlchemy and dynamically extends their capabilities by community driven algorithms for probabilistic graphical modeling [PGM], machine learning [ML] and structured data-analysis [SDA].

1.1 Components

Nemoa is open source and based on the Python programming language. It provides:

- A transparent DW architecture for the seamless integration of existing SQL databases, flat data from laboratory measurement devices or data generators.
- A versatile and fast data modeling and data analysis framework.

CHAPTER 2

Install

Nemoa requires Python 3.7 or later. If you do not already have a Python environment configured on your computer, please see the instructions for installing the full scientific Python stack.

Note: If you are using the Windows platform and want to install optional packages (e.g., *scipy*), then it may be useful to install a Python distribution such as: Anaconda, Enthought Canopy, Python(x,y), WinPython, or Pyzo. If you already use one of these Python distributions, please refer to their online documentation.

Below it is assumed, that you have the default Python environment configured on your computer and you intend to install Nemoa inside of it. If you want to create and work with Python virtual environments, please follow instructions on venv and virtual environments.

2.1 Install the latest distributed package

You can install the latest distributed package of Nemoa by using *pip*:

```
$ pip install nemoa
```

2.2 Install the development branch

The installation requires that you have Git installed on your system. Under this prerequisite the first step is to clone the GitHub repository of Nemoa:

```
$ git clone https://github.com/frootlab/nemoa.git
```

Thereupon the development branch can locally be installed by using *pip*:

```
$ cd nemoa
$ pip install -e .
```

The `pip install` command allows you to follow the development branch as it changes by creating links in the right places and installing the command line scripts to the appropriate locations.

2.3 Update the development branch

Once you have cloned the Nemoa GitHub repository onto a local directory, you can update it anytime by running a `git pull` in this directory:

```
$ git pull
```

2.4 Testing the development branch

Nemoa uses the Python builtin package unittest for testing. Since the tests are not included in the distributed package you are required to install the Nemoa development branch. Thereupon you have to switch to the repository directory and run:

```
$ python3 tests
```

2.5 Required packages

Note: Some required packages (e.g., *numpy*) may require compiling C or C++ code. If you have difficulty installing these packages with *pip*, it is highly recommended to review the instructions for installing the full scientific Python stack.

By using the `pip install` the required packages should by installed automatically. These packages include:

```
- `numpy <https://www.numpy.org/>`_ (>= 1.15.0)
- `NetworkX <https://networkx.github.io/>`_ (>= 2.1)
- `Matplotlib <https://matplotlib.org/>`_ (>= 2.2.2)
- `AppDirs <https://github.com/ActiveState/appdirs>`_ (>= 1.1.0)
```

CHAPTER 3

nemoa

3.1 nemoa package

3.1.1 Subpackages

nemoa.base package

Submodules

nemoa.base.array module

nemoa.base.nbase module

Module contents

nemoa.core package

Subpackages

nemoa.core.ui package

Submodules

nemoa.core.ui.shell module

Module contents

Submodules

3.1.2 Submodules

nemoa.typing module

3.1.3 Module contents

CHAPTER 4

Glossary

4.1 Math Glossary

4.1.1 Norms and Metrices

1-Norm The *1-norm* provides a length measure for the interpretation of vector spaces as orthogonal lattices. For a vector space of dimension *n*, the norm is given by:[1]

$$\|\vec{x}\|_1 := \sum_{i=1}^{n} |x_i|$$

The 1-norm specifies the *p-norm* for the case $p = 1$ and induces the *Manhattan distance* to the underlying vector space. When applied to a random sample, the Manhattan distance is also known as the *sum of absolute differences*.

References

Chebyshev Distance The *Chebyshev Distance* generates the *Chebyshev Metric*. For a vector space of dimension *n*, the Chebyshev distance is given by:[2]

$$d_\infty(\vec{x}, \vec{y}) := \max_i (|y_i - x_i|)$$

The Chebyshev distance is induced by the *Maximum norm* and specifies the *Minkowski distance* for the transition $p \to \infty$.

Euclidean Distance The *Euclidean Distance* corresponds to the natural geometric interpretation of a vector space. For an underlying vector space of dimension *n*, the Euclidean distance is given by:[3]

$$d_2(\vec{x}, \vec{y}) := \left(\sum_{i=1}^{n} |y_i - x_i|^2 \right)^{1/2}$$

[1] https://en.wikipedia.org/wiki/Taxicab_geometry
[2] https://en.wikipedia.org/wiki/Chebyshev_distance
[3] https://en.wikipedia.org/wiki/Euclidean_distance

The Euclidean distance is induced by the *Euclidean norm* and specifies the *Minkowski distance* for the case $p = 2$. With respect to regression analysis the Euclidean distance endows the components of a random sample with a discrepancy measure, between observed an estimated realizations. This discrepancy measure is commonly referred as *Residual sum of squares* (RSS) and provides the foundation for the method of least squares.[4]

Euclidean Norm The *Euclidean norm* provides the fundamental length measure for natural geometric interpretations of vector spaces. For a vector space of dimension *n*, the Euclidean norm is given by:[5]

$$\|\vec{x}\|_2 := \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}$$

The Euclidean norm equals the *p-norm* for $p = 2$ and induces a the *Euclidean distance* to it's domain.

Frobenius Distance The *Frobenius Distance* is a distance measure for matrices. For a vector space of dimension $n \times m$, the Frobenius distance is defined by:[6]

$$d_F(A,\, B) := \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |b_{ij} - a_{ij}|^2 \right)^{1/2}$$

The Frobenius distance is induced by the *Frobenius norm* and specifies the *pq-distance* for the case $p = q = 1$.

Frobenius Norm The *Frobenius norm* is a matrix norm, which is derived by the consecutive evaluation of the *Euclidean norm* for the rows and columns of a matrix. For an underlying vector space of dimension $n \times m$, the Frobenius norm is given by:[7]

$$\|A\|_F := \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{1/2}$$

The Frobenius norm specifies the *pq-norm* for the case $p = q = 2$.

Hölder Mean The *Hölder means* generalize the *Arithmetic mean* and the *Geometric mean*, in the same way as the *p-norm* generalizes the *Euclidean norm* and the *1-norm*. For a positive real number *p* and a vector space of dimension *n*, the Hölder mean for absolute values is given by:[8]

$$M_p(\vec{x}) := \left(\frac{1}{n} \sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

By it's definition it follows, that for $p \geq 1$ the Hölder means for absolute values are linear related to the p-norms:

$$M_p(\vec{x}) = \left(\frac{1}{n} \right)^{1/p} \|\vec{x}\|_p$$

As a consequence for $p \geq 1$ the Hölder means of absolute values are norms and thus induce distances to their underlying domains. These are occasionally referred as *power mean difference*.

The Hölder means and their respective distances, have important applications in regression analysis. When applied to the components of a random sample, the Hölder means of absolute values are known as the absolute sample moments and their induces metrices provide normalized measures of statistical dispersion.

[4] https://en.wikipedia.org/wiki/Least_squares
[5] https://en.wikipedia.org/wiki/Euclidean_norm
[6] https://en.wikipedia.org/wiki/Frobenius_norm
[7] https://en.wikipedia.org/wiki/Frobenius_norm
[8] https://en.wikipedia.org/wiki/Power_mean

Manhattan Distance The *Manhattan Distance* corresponds to the interpretation of vector spaces as orthogonal lattices. For a vector space of dimension *n*, the Manhattan distance is given by:[9]

$$d_1(\vec{x},\, \vec{y}) := \sum_{i=1}^{n} |y_i - x_i|$$

The Manhattan distance is induced by the *1-norm* and specifies the *Minkowski distance* for $p = 1$. When applied to a fixed set of outcomes of a random variable, the Minkowski distance is a measure of *discrepancy measure* and referred as *Sum of Absolute Differences*.

Maximum Norm The *Maximum norm* provides a length measure for vector spaces. For a vector space of dimension *n*, the Maximum norm is given by:[10]

$$\|\vec{x}\|_\infty := \max_i \left(|x_i|\right)$$

The Maximum norm specifies the *p-norm* for the case $p \to \infty$ and induces the *Chebyshev distance* to it's domain.

Mean Absolute The *Mean Absolute* provides a normalized length measure for the interpretation of vector spaces as orthogonal lattices. For a vector space of dimension *n*, it is given by:

$$M_1(\vec{x}) := \frac{1}{n} \sum_{i=1}^{n} |x_i|$$

The Mean Absolute specifies the *Hölder mean* of absolute values for the case $p = 1$ and is linear dependent to the *1-norm*:

$$M_1(\vec{x}) = \frac{\|\vec{x}\|_1}{n}$$

Due to this linear relationship the Mean Absolute is a valid vector space norm and thus induces a distance to it's underlying domain, which is referred as *mean absolute difference*.

Mean Absolute Difference The *Mean Absolute Difference* (MD) is a normalized distance measure for the interpretation of vector spaces as orthogonal lattices. For a vector space of dimension *n*, this distance is given by:

$$\mathrm{MD}_1(\vec{x},\, \vec{y}) := \frac{1}{n} \sum_{i=1}^{n} |y_i - x_i|$$

The mean absolute difference is induced by the *mean absolute* and specifies the *power mean difference* for the case $p = 1$. Furthermore the mean absolute difference is linear dependent to the *Manhattan distance*:

$$\mathrm{MD}_1(\vec{x},\, \vec{y}) = \frac{d_1(\vec{x},\, \vec{y})}{n}$$

The term 'mean absolute difference' is frequently associated with it's application to sampled values[11]. In regression analysis it provides a consistent and unbiased estimator for the *mean absolute error* of a predictor.

Minkowski Distance The class of *Minkowski Distances* provides different geometric interpretations of vector spaces. For a real number $p \geq 1$ and a vector space of dimension *n*, the Minkowski distance is given by:[12]

$$d_p(\vec{x},\, \vec{y}) := \left(\sum_{i=1}^{n} |y_i - x_i|^p \right)^{1/p}$$

The class of Minkowski distances is induced by the *p-norm* and comprises the *Euclidean distance* the *Manhattan distance* and the *Chebyshev distance*

[9] https://en.wikipedia.org/wiki/Taxicab_geometry
[10] https://en.wikipedia.org/wiki/Maximum_norm
[11] https://en.wikipedia.org/wiki/Mean_absolute_difference
[12] https://en.wikipedia.org/wiki/Minkowski_distance

p-Norm The *p-norms* provide length measures for different geometric interpretations of vector spaces. For a real number $p \geq 1$ and a vector space of dimension *n*, the p-norm is given by:[13]

$$\|\vec{x}\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

For $0 \leq p < 1$ an evaluation according to the p-norm does not satisfy the triangle inequality and yields a quasi-norm.

The p-norms generalize the *1-Norm*, the *Euclidean Norm* and the *Maximum Norm*. The class of distances, induced by the p-norms are referred as *Minkowski distance*.

pq-Distance The *pq-Distances* are matrix distances, which are derived by an elementwise application of the *p-norm* to the rows of two matrices, followed by an elementwise application of another p-norm to the columns. For real numbers $p,\ q \geq 1$ and an underlying vector space of dimension $n \times m$, the pq-distance is given by:[14]

$$d_{p,q}(A,\ B) := \left(\sum_{j=1}^{m} \left(\sum_{i=1}^{n} |a_{ij} - b_{ij}|^p \right)^{q/p} \right)^{1/q}$$

For the case $p = q = 2$, the pq-distance is also referred as *Frobenius distance*.

pq-Norm The *pq-Norms* are matrix norms, which are derived by an elementwise application of the *p-norm* to the rows of a matrix, followed by an elementwise application of another p-norm to the columns. For real numbers $p,\ q \geq 1$ and an underlying vector space of dimension $n \times m$, the pq-norm is given by:[15]

$$\|A\|_{p,q} := \left(\sum_{j=1}^{m} \left(\sum_{i=1}^{n} |a_{ij}|^p \right)^{q/p} \right)^{1/q}$$

For the case $p = q = 2$, the pq-norm is also referred as *Frobenius norm*.

Power Mean Difference The *Power Mean Differences* are normalized distance measures for different geometric interpretations of vector spaces. For a real number $p \geq 1$ and a vector space of dimension *n*, the power mean difference is given by:

$$\mathrm{MD}_p(\vec{x},\ \vec{y}) := \left(\frac{1}{n} \sum_{i=1}^{n} |y_i - x_i|^p \right)^{1/p}$$

The power mean differences are induced by the *Hölder mean* for absolute values and linear related to the *Minkowski distance*:

$$\mathrm{MD}_p(\vec{x},\ \vec{y}) = \left(\frac{1}{n} \right)^{1/p} d_p(\vec{x},\ \vec{y})$$

When applied to the components of a random sample, the Power-Mean differences are normalized measures of statistical dispersion.

Quadratic Mean The *Quadratic Mean* is a normalized length measure for the geometric interpretation of vector spaces. For a vector space of dimension *n*, it is given by:[16]

$$M_2(\vec{x}) := \left(\frac{1}{n} \sum_{i=1}^{n} |x_i|^2 \right)^{1/2}$$

[13] https://en.wikipedia.org/wiki/P_norm
[14] https://en.wikipedia.org/wiki/Matrix_norm#L2,1_and_Lp,q_norms
[15] https://en.wikipedia.org/wiki/Matrix_norm#L2,1_and_Lp,q_norms
[16] https://en.wikipedia.org/wiki/Quadratic_mean

The quadratic mean specifies the *Hölder mean* for $p = 2$ and is linear dependent to the *Euclidean norm*:

$$M_2(\vec{x}) = \frac{\|\vec{x}\|_2}{\sqrt{n}}$$

Due to this linear relationship the quadratic mean is a valid vector space norm and thus induces a distance to it's underlying domain, which occasionally is referred as the *quadratic mean difference*. When applied to the components of a random sample, the quadratic mean norm is a sample statistic, which is referred as *Root-Mean-Square error*.

Quadratic Mean Difference The *Quadratic Mean Difference* is a normalized distance measure for the natural geometric interpretation of vector spaces. For a vector space of dimension *n*, the distance is given by:

$$\mathrm{MD}_2(\vec{x},\, \vec{y}) := \left(\frac{1}{n} \sum_{i=1}^{n} |y_i - x_i| \right)^{1/2}$$

The quadratic mean difference is induced by the *quadratic mean* and specifies the *power mean difference* for $p = 2$. Furthermore the quadratic mean difference is linear dependent to the *Euclidean distance*:

$$\mathrm{MD}_2(\vec{x},\, \vec{y}) = \frac{d_2(\vec{x},\, \vec{y})}{\sqrt{n}}$$

When applied to individual components of a random sample, the quadratic mean difference is a measure of statistical dispersion and referred as *Root-Mean-Square Error*.

4.1.2 Statistics

Association Measure *Association measures* refer to a wide variety of coefficients, that measure the statistical strength of relationships between the variables of interest. These measures can be directed / undirected, signed / unsigned and normalized or unnormalized. Examples for association measures are the Pearson correlation coefficient, Mutual information or Statistical Interactions.

Discrepancy Measure *Discrepancy measures* are binary functions in spaces of random variables, that induce a semi-metric to the underlying space.[17] In regression analysis discrepancies are used to assess the accuracy of a predictor, by quantifying the expected deviation between observed and predicted realizations. By minimizing a discrepancy with respect to parameters, it serves as an objective function for parameter and model selection.

Mean Absolute Error The *Mean Absolute Error* (MAE) is a *discrepancy measure*, that assesses the accuracy of a predictor. For an observable random variable Y and a corresponding predictor $\hat{Y}$ the MAE is given by:

$$\mathrm{MAE} := \mathrm{E}\left[|Y - \hat{Y}| \right]$$

The MAE has a consistent and unbiased estimator, given by the *mean absolute difference* of observations and predictions. For *n* observations $\mathbf{y}$ with corresponding predictions $\hat{\mathbf{y}}$ the MAE is estimated by:

$$\mathrm{MD}_1(\mathbf{y},\, \hat{\mathbf{y}}) \xrightarrow{n \to \infty} \mathrm{MAE}$$

Due to this transition, the MAE adopts all required properties from the mean absolute difference, to induce a valid metric to the space of random variables.

Mean Squared Error The *Mean Squared Error* (MSE) is a *discrepancy measure*, that assesses the accuracy of a predictor. For an observable random variable Y and a corresponding predictor $\hat{Y}$ the MSE is given by:

$$\mathrm{MSE} := \mathrm{E}\left[(Y - \hat{Y})^2 \right]$$

[17] https://en.wikipedia.org/wiki/discrepancy_function

The MSE has a consistent and unbiased estimator, given by the squared *quadratic mean difference* of observations and predictions. For *n* observations $\mathbf{y}$ with corresponding predictions $\hat{\mathbf{y}}$ the MSE is estimated by:

$$\mathrm{MD}_2(\mathbf{y}, \hat{\mathbf{y}})^2 \xrightarrow{n \to \infty} \mathrm{MSE}$$

In difference to the *Root-Mean-Square Error*, the MSE does not satisfy the triangle inequality and therefore does not define a valid distance measure. Since the MSE, however, is positive definite and subhomogeneous, it induces a semi-metric to the underlying space of random variables.

Residual Sum of Squares The *Residual Sum of Squares* (RSS) is a *discrepancy measure*, that assesses the accuracy of a predictor with respect to a fixed (finite) set of observations. For an observable random variable Y with *n* fixed observations $\mathbf{y}$ and a predictor $\hat{Y}$ with corresponding predictions $\hat{\mathbf{y}}$ the RSS is given by:

$$\mathrm{RSS}(\mathbf{y}, \hat{\mathbf{y}}) := \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

The RSS equals the squared *Euclidean distance*, which does not satisfy the triangle inequality and therefore does not define a valid distance measure. Since the RSS, however, is positive definite and subhomogeneous, it induces a semi-metric to the underlying space of random variables.

Root-Mean-Square Error The *Root-Mean-Square Error* (RMSE) is a *discrepancy measure*, that assesses the accuracy of a predictor. For an observable random variable Y and a corresponding predictor $\hat{Y}$ the RMSE is given by:

$$\mathrm{RMSE} := \mathrm{E}\left[(Y - \hat{Y})^2\right]^{1/2}$$

The RMSE has a consistent and unbiased estimator, given by the *quadratic mean difference* of observations and predictions. For *n* observations $\mathbf{y}$ with corresponding predictions $\hat{\mathbf{y}}$ the RMSE is estimated by:

$$\mathrm{MD}_2(\mathbf{y}, \hat{\mathbf{y}}) \xrightarrow{n \to \infty} \mathrm{RMSE}$$

Due to this transition, the RMSE adopts all required properties from the quadratic mean difference, to induce a valid metric to the space of random variables.

Sum of Absolute Differences The *Sum of Absolute Differences* (SAD) is a *discrepancy measure*, that assesses the accuracy of a predictor with respect to a fixed (finite) set of observations. For an observable random variable Y with *n* fixed observations $\mathbf{y}$ and a predictor $\hat{Y}$ with corresponding predictions $\hat{\mathbf{y}}$ the RSS is given by:

$$\mathrm{SAD}(\mathbf{y}, \hat{\mathbf{y}}) := \sum_{i=1}^{n} |y_i - \hat{y}_i|$$

The SAD equals the *Manhattan distance* and therefore is also a valid distance measure within the underlying space of random variables. The SAD is effectively the simplest possible distance, that takes into account every observation of a fixed finite set. This makes SAD an extremely fast distance measure.

4.2 API Glossary

4.2.1 Basic Parameters and Formats

Field Identifier *Field Identifiers* uniquely identify fields within objects. Since objects may comprise fields, given by it's attributes, items or / and other accessing mechanisms, the identification mechanism of the fields depends on the domain type of the object:

objects For the domain type `object` fields are identified by attribute names, and therefore required to be valid identifiers as specified in **PEP 3131**. Additionally the functions in the module `nemoa.base.operator` accept the usage of dots, for the identification of arbitrary sub-objects within object hierarchies.

mappings For the domain type `dict` (or any subclass of the `Mapping class`) fields are identified by the keys of the mappings and therefore required to be hashable. Additionally the functions in the module `nemoa.base.operator` require, that the field identifiers are not given by tuples.

sequences For the domain type `tuple` (or any subclass of the `Sequence class`) the fields are identifiers are not the names of fields, but their position within the sequence (starting with 0) and therefore required to be non-negative integers.

Domain Like *Domain Like* parameters are used to specify the *type* and the *frame* of the domain and the target (codomain) of an operator. Thereby the format, which is used for the specification depends an the respective type:

Domain types that use named *field identifiers* (like `object` or `dict`) do not require the specification of a frame. In this case the parameter format is given by a single term *<type>*, where *<type>* may ether be a supported `type` of the respective parameter or None, for the default behavior of the respective function.

Domain types that use positional field identifiers (like `tuple` or `list`) require the specification of a frame, to map variable names to their positions within the frame. In this case the parameter format is given by a tuple *(<type>, <frame>)*, where the term *<frame>* is required to be a tuple of valid field identifiers in the domain type.

Variable Definition *Variable Definition* parameters are used to specify the mapping of fields to variables. For a variable that represents a named field (e.g. group keys in grouped sequences, etc.), it is sufficient to only provide the *identifier* of the field. If the underlying domain type, however, does not support named fields or the variable name shall be different to the field identifier, then the variable name has to be included in the definition by *(<field>, <name>)*, where *<name>* is required to be a valid identifier.

For a variable that only depends on a single named field, but is not identical identical to the field, the mapping from the field to the variable has to be specified by an operator (e.g. an *aggregation function*), which has to be included within the definition by *(<field>, <operator>)*. If the underlying domain type, does not support named fields or the variable name is required to be different to the field identifier, then the definition has to be given by the format *(<field>, <operator>, <name>)*.

For a variable that depends on multiple fields the definition has to be given in the format *(<fields>, <operator>, <name>)*, where *<fields>* is a tuple of valid field identifiers in the domain type and *<operator>* an operator, which accepts the specified fields as arguments.

4.2.2 Parameters and Formats used in Data Warehousing

Row Like *Row like* data comprises different data formats, which are used to represent table records. This includes tuples, mappings and instances of the `Record class`. The `Table class` accepts these data types for appending rows by `insert()` and for retrieving rows by `select()`.

Cursor Mode The *cursor mode* defines the *scrolling type* and the *operation mode* of a cursor. Internally the respective parameters of the `Cursor class` are identified by binary flags. The public interface uses a string representation, given by the space separated names of the scrolling type and the the operation mode. Supported scrolling types are:

forward-only The default scrolling type of cursors is called a forward-only cursor and can move only forward through the result set. A forward-only cursor does not support scrolling but only fetching rows from the start to the end of the result set.

scrollable A scrollable cursor is commonly used in screen-based interactive applications, like spreadsheets, in which users are allowed to scroll back and forth through the result set. However, applications should use scrollable cursors only when forward-only cursors will not do the job, as scrollable cursors are generally more expensive, than forward-only cursors.

random Random cursors move randomly through the result set. In difference to a randomly sorted cursor, the rows are not unique and the number of fetched rows is not limited to the size of the result set. If the method `fetch()` is called with a zero value for size, a CursorModeError is raised.

Supported operation modes are:

dynamic A **dynamic cursor** is built on-the-fly and therefore comprises any changes made to the rows in the result set during it's traversal, including new appended rows and the order of it's traversal. This behavior is regardless of whether the changes occur from inside the cursor or by other users from outside the cursor. Dynamic cursors are thread-safe but do not support counting filtered rows or sorting rows.

indexed Indexed cursors (aka Keyset-driven cursors) are built on-the-fly with respect to an initial copy of the table index and therefore comprise changes made to the rows in the result set during it's traversal, but not new appended rows nor changes within their order. Keyset driven cursors are thread-safe but do not support sorting rows or counting filtered rows.

static Static cursors are buffered and built during it's creation time and therefore always display the result set as it was when the cursor was first opened. Static cursors are not thread-safe but support counting the rows with respect to a given filter and sorting the rows.

Aggregation Function *Aggregation Functions* are `callable` objects, that transform sequences of objects of a given domain into a single value. Examples include `len()`, `sum()`, `min()` or `max()`, but depending on the domain, many out-of-the-box aggregators are shipped with the standard library package `statistics` or with third party packages like `numpy`.

CHAPTER 5

License

Nemoa is distributed with the GNU General Public License v3 [GPLV3].

```
GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies of this license
document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and
other kinds of works.

The licenses for most software and other practical works are designed to
take away your freedom to share and change the works.  By contrast, the GNU
General Public License is intended to guarantee your freedom to share and
change all versions of a program--to make sure it remains free software for
all its users.  We, the Free Software Foundation, use the GNU General Public
License for most of our software; it applies also to any other work released
this way by its authors.  You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price.  Our
General Public Licenses are designed to make sure that you have the freedom
to distribute copies of free software (and charge for them if you wish),
that you receive source code or can get it if you want it, that you can
change the software or use pieces of it in new free programs, and that you
know you can do these things.

To protect your rights, we need to prevent others from denying you these
rights or asking you to surrender the rights.  Therefore, you have certain
responsibilities if you distribute copies of the software, or if you modify
it: responsibilities to respect the freedom of others.
```

(continues on next page)

(continued from previous page)

```
For example, if you distribute copies of such a program, whether gratis or
for a fee, you must pass on to the recipients the same freedoms that you
received.  You must make sure that they, too, receive or can get the source
code.  And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1)
assert copyright on the software, and (2) offer you this License giving you
legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that
there is no warranty for this free software.  For both users' and authors'
sake, the GPL requires that modified versions be marked as changed, so that
their problems will not be attributed erroneously to authors of previous
versions.

Some devices are designed to deny users access to install or run modified
versions of the software inside them, although the manufacturer can do so.
This is fundamentally incompatible with the aim of protecting users' freedom
to change the software.  The systematic pattern of such abuse occurs in the
area of products for individuals to use, which is precisely where it is most
unacceptable.  Therefore, we have designed this version of the GPL to
prohibit the practice for those products.  If such problems arise
substantially in other domains, we stand ready to extend this provision to
those domains in future versions of the GPL, as needed to protect the
freedom of users.

Finally, every program is threatened constantly by software patents. States
should not allow patents to restrict development and use of software on
general-purpose computers, but in those that do, we wish to avoid the
special danger that patents applied to a free program could make it
effectively proprietary.  To prevent this, the GPL assures that patents
cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification
follow.

TERMS AND CONDITIONS

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of
works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License.
Each licensee is addressed as "you".  "Licensees" and "recipients" may be
individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a
fashion requiring copyright permission, other than the making of an exact
copy.  The resulting work is called a "modified version" of the earlier work
or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the
Program.
```

(continues on next page)

(continued from previous page)

```
To "propagate" a work means to do anything with it that, without permission,
would make you directly or secondarily liable for infringement under
applicable copyright law, except executing it on a computer or modifying a
private copy.  Propagation includes copying, distribution (with or without
modification), making available to the public, and in some countries other
activities as well.

To "convey" a work means any kind of propagation that enables other parties
to make or receive copies.  Mere interaction with a user through a computer
network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the
extent that it includes a convenient and prominently visible feature that
(1) displays an appropriate copyright notice, and (2) tells the user that
there is no warranty for the work (except to the extent that warranties are
provided), that licensees may convey the work under this License, and how to
view a copy of this License.  If the interface presents a list of user
commands or options, such as a menu, a prominent item in the list meets this
criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making
modifications to it.  "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official
standard defined by a recognized standards body, or, in the case of
interfaces specified for a particular programming language, one that is
widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than
the work as a whole, that (a) is included in the normal form of packaging a
Major Component, but which is not part of that Major Component, and (b)
serves only to enable use of the work with that Major Component, or to
implement a Standard Interface for which an implementation is available to
the public in source code form.  A "Major Component", in this context, means
a major essential component (kernel, window system, and so on) of the
specific operating system (if any) on which the executable work runs, or a
compiler used to produce the work, or an object code interpreter used to run
it.

The "Corresponding Source" for a work in object code form means all the
source code needed to generate, install, and (for an executable work) run
the object code and to modify the work, including scripts to control those
activities.  However, it does not include the work's System Libraries, or
general-purpose tools or generally available free programs which are used
unmodified in performing those activities but which are not part of the
work.  For example, Corresponding Source includes interface definition files
associated with source files for the work, and the source code for shared
libraries and dynamically linked subprograms that the work is specifically
designed to require, such as by intimate data communication or control flow
between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate
automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.
```

(continues on next page)

(continued from previous page)

```
2. Basic Permissions.

All rights granted under this License are granted for the term of copyright
on the Program, and are irrevocable provided the stated conditions are met.
This License explicitly affirms your unlimited permission to run the
unmodified Program.  The output from running a covered work is covered by
this License only if the output, given its content, constitutes a covered
work.  This License acknowledges your rights of fair use or other
equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey,
without conditions so long as your license otherwise remains in force.  You
may convey covered works to others for the sole purpose of having them make
modifications exclusively for you, or provide you with facilities for
running those works, provided that you comply with the terms of this License
in conveying all material for which you do not control copyright.  Those
thus making or running the covered works for you must do so exclusively on
your behalf, under your direction and control, on terms that prohibit them
from making any copies of your copyrighted material outside their
relationship with you.

Conveying under any other circumstances is permitted solely under the
conditions stated below.  Sublicensing is not allowed; section 10 makes it
unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure
under any applicable law fulfilling obligations under article 11 of the WIPO
copyright treaty adopted on 20 December 1996, or similar laws prohibiting or
restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid
circumvention of technological measures to the extent such circumvention is
effected by exercising rights under this License with respect to the covered
work, and you disclaim any intention to limit operation or modification of
the work as a means of enforcing, against the work's users, your or third
parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive
it, in any medium, provided that you conspicuously and appropriately publish
on each copy an appropriate copyright notice; keep intact all notices
stating that this License and any non-permissive terms added in accord with
section 7 apply to the code; keep intact all notices of the absence of any
warranty; and give all recipients a copy of this License along with the
Program.

You may charge any price or no price for each copy that you convey, and you
may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce
it from the Program, in the form of source code under the terms of section
```

(continues on next page)

(continued from previous page)

```
4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and
giving a relevant date.

b) The work must carry prominent notices stating that it is released under
this License and any conditions added under section 7.  This requirement
modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to
anyone who comes into possession of a copy.  This License will therefore
apply, along with any applicable section 7 additional terms, to the whole of
the work, and all its parts, regardless of how they are packaged.  This
License gives no permission to license the work in any other way, but it
does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display
Appropriate Legal Notices; however, if the Program has interactive
interfaces that do not display Appropriate Legal Notices, your work need not
make them do so.

A compilation of a covered work with other separate and independent works,
which are not by their nature extensions of the covered work, and which are
not combined with it such as to form a larger program, in or on a volume of
a storage or distribution medium, is called an "aggregate" if the
compilation and its resulting copyright are not used to limit the access or
legal rights of the compilation's users beyond what the individual works
permit.  Inclusion of a covered work in an aggregate does not cause this
License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of
sections 4 and 5, provided that you also convey the machine-readable
Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including
a physical distribution medium), accompanied by the Corresponding Source
fixed on a durable physical medium customarily used for software
interchange.

b) Convey the object code in, or embodied in, a physical product (including
a physical distribution medium), accompanied by a written offer, valid for
at least three years and valid for as long as you offer spare parts or
customer support for that product model, to give anyone who possesses the
object code either (1) a copy of the Corresponding Source for all the
software in the product that is covered by this License, on a durable
physical medium customarily used for software interchange, for a price no
more than your reasonable cost of physically performing this conveying of
source, or (2) access to copy the Corresponding Source from a network server
at no charge.

c) Convey individual copies of the object code with a copy of the written
offer to provide the Corresponding Source.  This alternative is allowed only
occasionally and noncommercially, and only if you received the object code
with such an offer, in accord with subsection 6b.
```

(continues on next page)

(continued from previous page)

```
d) Convey the object code by offering access from a designated place (gratis
or for a charge), and offer equivalent access to the Corresponding Source in
the same way through the same place at no further charge.  You need not
require recipients to copy the Corresponding Source along with the object
code.  If the place to copy the object code is a network server, the
Corresponding Source may be on a different server (operated by you or a
third party) that supports equivalent copying facilities, provided you
maintain clear directions next to the object code saying where to find the
Corresponding Source.  Regardless of what server hosts the Corresponding
Source, you remain obligated to ensure that it is available for as long as
needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you
inform other peers where the object code and Corresponding Source of the
work are being offered to the general public at no charge under subsection
6d.

A separable portion of the object code, whose source code is excluded from
the Corresponding Source as a System Library, need not be included in
conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any
tangible personal property which is normally used for personal, family, or
household purposes, or (2) anything designed or sold for incorporation into
a dwelling.  In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage.  For a particular
product received by a particular user, "normally used" refers to a typical
or common use of that class of product, regardless of the status of the
particular user or of the way in which the particular user actually uses, or
expects or is expected to use, the product.  A product is a consumer product
regardless of whether the product has substantial commercial, industrial or
non-consumer uses, unless such uses represent the only significant mode of
use of the product.

"Installation Information" for a User Product means any methods, procedures,
authorization keys, or other information required to install and execute
modified versions of a covered work in that User Product from a modified
version of its Corresponding Source.  The information must suffice to ensure
that the continued functioning of the modified object code is in no case
prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as part of
a transaction in which the right of possession and use of the User Product
is transferred to the recipient in perpetuity or for a fixed term
(regardless of how the transaction is characterized), the Corresponding
Source conveyed under this section must be accompanied by the Installation
Information.  But this requirement does not apply if neither you nor any
third party retains the ability to install modified object code on the User
Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates for
a work that has been modified or installed by the recipient, or for the User
Product in which it has been modified or installed.  Access to a network may
be denied when the modification itself materially and adversely affects the
operation of the network or violates the rules and protocols for
```

(continues on next page)

(continued from previous page)

```
communication across the network.

Corresponding Source conveyed, and Installation Information provided, in
accord with this section must be in a format that is publicly documented
(and with an implementation available to the public in source code form),
and must require no special password or key for unpacking, reading or
copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License
by making exceptions from one or more of its conditions. Additional
permissions that are applicable to the entire Program shall be treated as
though they were included in this License, to the extent that they are valid
under applicable law.  If additional permissions apply only to part of the
Program, that part may be used separately under those permissions, but the
entire Program remains governed by this License without regard to the
additional permissions.

When you convey a copy of a covered work, you may at your option remove any
additional permissions from that copy, or from any part of it.  (Additional
permissions may be written to require their own removal in certain cases
when you modify the work.)  You may place additional permissions on
material, added by you to a covered work, for which you have or can give
appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to
a covered work, you may (if authorized by the copyright holders of that
material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of
sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author
attributions in that material or in the Appropriate Legal Notices displayed
by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or
requiring that modified versions of such material be marked in reasonable
ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors
of the material; or

e) Declining to grant rights under trademark law for use of some trade
names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by
anyone who conveys the material (or modified versions of it) with
contractual assumptions of liability to the recipient, for any liability
that these contractual assumptions directly impose on those licensors and
authors.

All other non-permissive additional terms are considered "further
restrictions" within the meaning of section 10.  If the Program as you
received it, or any part of it, contains a notice stating that it is
governed by this License along with a term that is a further restriction,
```

(continues on next page)

(continued from previous page)

```
you may remove that term.  If a license document contains a further
restriction but permits relicensing or conveying under this License, you may
add to a covered work material governed by the terms of that license
document, provided that the further restriction does not survive such
relicensing or conveying.

If you add terms to a covered work in accord with this section, you must
place, in the relevant source files, a statement of the additional terms
that apply to those files, or a notice indicating where to find the
applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of
a separately written license, or stated as exceptions; the above
requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided
under this License.  Any attempt otherwise to propagate or modify it is
void, and will automatically terminate your rights under this License
(including any patent licenses granted under the third paragraph of section
11).

However, if you cease all violation of this License, then your license from
a particular copyright holder is reinstated (a) provisionally, unless and
until the copyright holder explicitly and finally terminates your license,
and (b) permanently, if the copyright holder fails to notify you of the
violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated
permanently if the copyright holder notifies you of the violation by some
reasonable means, this is the first time you have received notice of
violation of this License (for any work) from that copyright holder, and you
cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the
licenses of parties who have received copies or rights from you under this
License.  If your rights have been terminated and not permanently
reinstated, you do not qualify to receive new licenses for the same material
under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a
copy of the Program.  Ancillary propagation of a covered work occurring
solely as a consequence of using peer-to-peer transmission to receive a copy
likewise does not require acceptance.  However, nothing other than this
License grants you permission to propagate or modify any covered work.
These actions infringe copyright if you do not accept this License.
Therefore, by modifying or propagating a covered work, you indicate your
acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a
license from the original licensors, to run, modify and propagate that work,
subject to this License.  You are not responsible for enforcing compliance
```

(continues on next page)

(continued from previous page)

```
by third parties with this License.

An "entity transaction" is a transaction transferring control of an
organization, or substantially all assets of one, or subdividing an
organization, or merging organizations.  If propagation of a covered work
results from an entity transaction, each party to that transaction who
receives a copy of the work also receives whatever licenses to the work the
party's predecessor in interest had or could give under the previous
paragraph, plus a right to possession of the Corresponding Source of the
work from the predecessor in interest, if the predecessor has it or can get
it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights
granted or affirmed under this License.  For example, you may not impose a
license fee, royalty, or other charge for exercise of rights granted under
this License, and you may not initiate litigation (including a cross-claim
or counterclaim in a lawsuit) alleging that any patent claim is infringed by
making, using, selling, offering for sale, or importing the Program or any
portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License
of the Program or a work on which the Program is based.  The work thus
licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or
controlled by the contributor, whether already acquired or hereafter
acquired, that would be infringed by some manner, permitted by this License,
of making, using, or selling its contributor version, but do not include
claims that would be infringed only as a consequence of further modification
of the contributor version.  For purposes of this definition, "control"
includes the right to grant patent sublicenses in a manner consistent with
the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent
license under the contributor's essential patent claims, to make, use, sell,
offer for sale, import and otherwise run, modify and propagate the contents
of its contributor version.

In the following three paragraphs, a "patent license" is any express
agreement or commitment, however denominated, not to enforce a patent (such
as an express permission to practice a patent or covenant not to sue for
patent infringement).  To "grant" such a patent license to a party means to
make such an agreement or commitment not to enforce a patent against the
party.

If you convey a covered work, knowingly relying on a patent license, and the
Corresponding Source of the work is not available for anyone to copy, free
of charge and under the terms of this License, through a publicly available
network server or other readily accessible means, then you must either (1)
cause the Corresponding Source to be so available, or (2) arrange to deprive
yourself of the benefit of the patent license for this particular work, or
(3) arrange, in a manner consistent with the requirements of this License,
to extend the patent license to downstream recipients.  "Knowingly relying"
means you have actual knowledge that, but for the patent license, your
conveying the covered work in a country, or your recipient's use of the
```

(continues on next page)

(continued from previous page)

```
covered work in a country, would infringe one or more identifiable patents
in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement,
you convey, or propagate by procuring conveyance of, a covered work, and
grant a patent license to some of the parties receiving the covered work
authorizing them to use, propagate, modify or convey a specific copy of the
covered work, then the patent license you grant is automatically extended to
all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope
of its coverage, prohibits the exercise of, or is conditioned on the
non-exercise of one or more of the rights that are specifically granted
under this License.  You may not convey a covered work if you are a party to
an arrangement with a third party that is in the business of distributing
software, under which you make payment to the third party based on the
extent of your activity of conveying the work, and under which the third
party grants, to any of the parties who would receive the covered work from
you, a discriminatory patent license (a) in connection with copies of the
covered work conveyed by you (or copies made from those copies), or (b)
primarily for and in connection with specific products or compilations that
contain the covered work, unless you entered into that arrangement, or that
patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any
implied license or other defenses to infringement that may otherwise be
available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or
otherwise) that contradict the conditions of this License, they do not
excuse you from the conditions of this License.  If you cannot convey a
covered work so as to satisfy simultaneously your obligations under this
License and any other pertinent obligations, then as a consequence you may
not convey it at all.  For example, if you agree to terms that obligate you
to collect a royalty for further conveying from those to whom you convey the
Program, the only way you could satisfy both those terms and this License
would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to
link or combine any covered work with a work licensed under version 3 of the
GNU Affero General Public License into a single combined work, and to convey
the resulting work.  The terms of this License will continue to apply to the
part which is the covered work, but the special requirements of the GNU
Affero General Public License, section 13, concerning interaction through a
network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the
GNU General Public License from time to time.  Such new versions will be
similar in spirit to the present version, but may differ in detail to
address new problems or concerns.
```

(continues on next page)

(continued from previous page)

```
Each version is given a distinguishing version number.  If the Program
specifies that a certain numbered version of the GNU General Public License
"or any later version" applies to it, you have the option of following the
terms and conditions either of that numbered version or of any later version
published by the Free Software Foundation.  If the Program does not specify
a version number of the GNU General Public License, you may choose any
version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of
the GNU General Public License can be used, that proxy's public statement of
acceptance of a version permanently authorizes you to choose that version
for the Program.

Later license versions may give you additional or different permissions.
However, no additional obligations are imposed on any author or copyright
holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE
LAW.  EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR
OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND,
EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.  THE
ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU.
SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY
SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL
ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE
PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY
GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE
OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA
OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD
PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS),
EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF
SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above
cannot be given local legal effect according to their terms, reviewing
courts shall apply local law that most closely approximates an absolute
waiver of all civil liability in connection with the Program, unless a
warranty or assumption of liability accompanies a copy of the Program in
return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible
use to the public, the best way to achieve this is to make it free software
which everyone can redistribute and change under these terms.
```

(continues on next page)

(continued from previous page)

```
To do so, attach the following notices to the program.  It is safest to
attach them to the start of each source file to most effectively state the
exclusion of warranty; and each file should have at least the "copyright"
line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software: you can redistribute it and/or modify it
under the terms of the GNU General Public License as published by the Free
Software Foundation, either version 3 of the License, or (at your option)
any later version.

This program is distributed in the hope that it will be useful, but WITHOUT
ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE.  See the GNU General Public License for
more details.

You should have received a copy of the GNU General Public License along with
this program.  If not, see <https://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like
this when it starts in an interactive mode:

<program>  Copyright (C) <year>  <name of author> This program comes with
ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software,
and you are welcome to redistribute it under certain conditions; type `show
c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License.  Of course, your program's commands
might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary. For
more information on this, and how to apply and follow the GNU GPL, see
<https://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program
into proprietary programs.  If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library.  If this is what you want to do, use the GNU Lesser General Public
License instead of this License.  But first, please read
<https://www.gnu.org/licenses/why-not-lgpl.html>.
```

CHAPTER 6

Bibliography

6.1 Publications

6.2 Links

Bibliography

[ELLIOT1993] "A better Activation Function for Artificial Neural Networks", David L. Elliott (1993)

[LECUN1998] "Efficient BackProp", Y. LeCun, L. Bottou, G. B. Orr, K. Müller (1998)

[HINTON1986] "Learning representations by back-propagating errors", Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986)

[SDA] https://en.wikipedia.org/wiki/Structured_data_analysis_(statistics)

[DCAM] http://dublincore.org/documents/2007/06/04/abstract-model/

[DCMI-TYPE] http://dublincore.org/documents/dcmi-type-vocabulary/

[DCMI-TERMS] http://dublincore.org/documents/dcmi-terms/

[DCMI-TYPE] http://dublincore.org/documents/dcmi-type-vocabulary/

[GPLV3] https://www.gnu.org/licenses/gpl.txt

[IEEE754] https://ieeexplore.ieee.org/document/4610935/

[ML] https://en.wikipedia.org/wiki/Machine_learning

[MIME] http://www.iana.org/assignments/media-types/

[PGM] https://en.wikipedia.org/wiki/Graphical_model

[TGN] http://www.getty.edu/research/tools/vocabulary/tgn/

[UAX31] http://unicode.org/reports/tr31/

[W3CDTF] http://www.w3.org/TR/NOTE-datetime

Index

Symbols

A

C

D

E

F

H

M

P

Q

R

S

V