

COSMIC - ISO 19761

Sizing AI Software Requirements Case Study

Examples with three machine learning algorithms

COSMIC Version 5.0
November 2026

Foreword.

This Case Study is based on three examples of sizing machine learning algorithms of the ETS Master-Thesis [1] of Shaghayegh Vedadi. It reports on the research carried out to explore a measurement of functional size of existing software applications in the specific software development context of real-time embedded and of AI software. This research project is to assist engineers by providing the size information on the software requirements (both before and after software delivery) and to use this size to build realistic estimation models for their software projects and improve their planning. These functional size requirements can be documented at different levels of granularity when estimating future projects.

Editors:

Shaghayegh Vedadi
Alain Abran, Ecole de technologie supérieure (Canada),
Arlan Lesterhuis (The Netherlands).

Other members of the COSMIC Measurement Practices Committee:

Jean-Marc Desharnais, Ecole de technologie supérieure – University of Quebec (Canada),
Peter Fagg, Pentad (UK),
Andrés Gutierrez, Leda MC (Colombia),
Dylan Ren, Measures Technology LLC (China),
Bruce Reynolds, Tecolote Research (USA),
Hassan Soubra, ECE Lyon - Ecole d'Ingénieurs - Engineering School (France),
Sylvie Trudel, Université du Québec à Montréal - UQAM (Canada),
Francisco Valdés Souto, Spingere (Mexico),
Frank Vogelezang, Adviescollege ICT-toetsing (The Netherlands).

Copyright 2026. All Rights Reserved. The Common Software Measurement International Consortium (COSMIC). Permission to copy all or part of this material is granted provided that the copies are not made or distributed for commercial advantage and that the title of the publication, its version number, and its date are cited and notice is given that copying is by permission of the Common Software Measurement International Consortium (COSMIC). To copy otherwise requires specific permission.

Table of Contents

1 INTRODUCTION AND AUDIENCE. 4

2 MEASUREMENT STRATEGY. 4

2.1 Context for the Examples. 4

2.2 Measurement purpose. 5

2.3 Measurement scope. 5

2.4 Functional users of the AI software. 5

2.5 Measurement process 5

3 EXAMPLE 1: K-MEANS CLUSTERING. 6

3.1 Software Functional Requirements. 6

3.2 Measurement. 7

4 EXAMPLE 2: LINEAR REGRESSION WITH A MACHINE LEARNING ALGORITHM. 10

4.1 Software Functional Requirements. 10

4.2 Measurement 11

5 EXAMPLE 3: AI DASHBOARD 13

5.1 Software Functional Requirements. 13

5.2 Measurement 15

6 REFERENCES AND DOCUMENTATION. 18

1 INTRODUCTION AND AUDIENCE.

COSMIC Function Points [1] is an ISO standard for measuring the size of the functional user requirements of a piece of software – see www.cosmic-sizing.org.

This document presents the sizing of three sets of requirements for programming an AI application using a machine learning algorithm:

Example 1: with a K-means clustering algorithm

Example 2: linear regression with a machine learning algorithm

Example 3: AI dashboard

In the three Examples the 'functional processes' are not explicitly described. This Case Study shows that such AI software requirements can be properly sized although their functional processes aren't explicitly written. The Examples are therefore valid COSMIC Case studies and therefore useful for both industry and academia.

Audience of this document is:

A) data scientists of the AI community.

B) programmers when they have to code pieces of software to implement the software requirements specified by the data scientist.

C) AI managers for AI project planning and monitoring.

2 MEASUREMENT STRATEGY.

2.1 Context for the Examples.

See Figure 1:

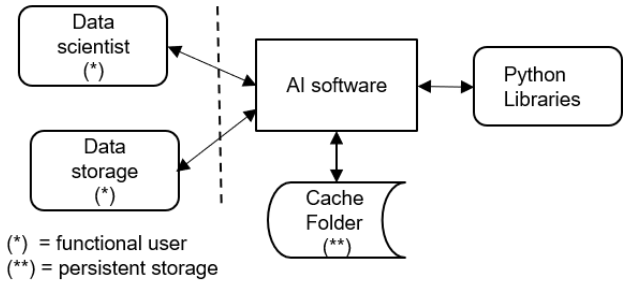

Figure 1 - Context diagram of the AI software to be sized.

1. The software development environment is the Visual Studio with Python Libraries.
2. This document segregates the functionality specific to the examples from the reused software (e.g. the machine learning algorithm) and from the Visual Studio code platform.
   - This segregation enables the data scientist to measure the software size data (and related development effort data) in a standardized fashion to develop estimation models for planning purposes and for on-going monitoring of the software tasks within an AI development environment.
3. The AI software to be sized is new and must be developed – see Figure 1.
4. There are no changes to any other pieces of software.

5. The data scientist who elicits and prepares the functional requirements is knowledgeable of the Python environment and how requirements are specified and documented in this AI environment.
6. Data cleaning and index changing procedures are not specific to this context and not described in this document: therefore, their measurement is not included.

2.2 Measurement purpose.

The purpose of the measurement is to enable AI data scientists, programmers and their managers to collect software size historical data for estimating future software development work for AI applications.

2.3 Measurement scope.

The scope of measurement is limited to the functionality to be coded (i.e. excludes the sizing of reused code).

2.4 Functional users of the AI software.

The functional users of the AI software are – see Figure 1:

- the data scientist,
- the data storage (where the data is maintained).

2.5 Measurement process

For these examples, the measurement is done using:

a) The functional requirements as they are written.

b) The four COSMIC definitions of the four types of data movements: **Entry**, **Exit**, **Read** and **Write**.

c) There are no requirements for error/confirmation messages.

Note 1. In the following Examples only the Python library names have to be specified, together with the input. The chosen algorithm produces the output. In consequence there are changes but no functional processes.

Note 2. The COSMIC measurement method includes a number of additional definitions and rules – see Reference [1].

3 EXAMPLE 1: K-MEANS CLUSTERING.

3.1 Software Functional Requirements.

This AI clustering application identifies patterns and data trends for calculating a vehicle's work performance and for predicting events pertaining to the maintenance of the vehicles:

- Detecting different events with respect to vehicle maintenance can help to predict a problem and fix it before it occurs.
- Event detection aimed at identifying immobility periods of vehicles can help to identify the reasons why the vehicle productivity has increased or decreased over a period of time.

Functional reuse of the K-Means algorithm.

a) The K-Means is an unsupervised machine learning algorithm (e.g. it groups similar data points together and detects underlying patterns of data points, rather than assigning data points to given groups).
b) The K-Means algorithm identifies the centroid(s) that represent a mean immobility engine speed in vehicles. The obtained values can be used as an input value for monitoring and optimizing purposes.

The requirements are those prepared by a data scientist for coding a data points clustering software application using a K-means clustering algorithm. This data points clustering application consists of – see Figure 1:

- A Data Input/Output software component (Data I/O component) to be specified and coded.
- K-means Algorithm to be reused from a Python library.

Software Requirement 1 – SR1: Receive Python library names.

The Python library names specified by the data scientist are:

- **Numpy**: for numerical operations
- **Pandas**: for data manipulation and analysis
- **Matplotlib**: to design the graphs and charts
- **Sklearn.cluster**: for the K-means clustering algorithm

Requirements to be coded: input Library name.

Software Requirement 2 – SR2: Specify file location/name and column titles for uploading raw data.

Requirements to be coded: input

a) raw data file location and file name,
b) column titles of the data frame: date/time, vehicle ID, engine speed, wheel speed and dump positioning, state, idle.

Software Requirement 3 – SR3: Enter selection parameters for data points removal.

Requirement to be coded: input selection parameters to remove unneeded data points from the data frame:

- engine speed = 0,
- wheel speed ≠ 0.

(Continued)

Software Requirement 4 – SR4: Enter quantile values to tag the data points.

Requirement to be coded: input:

- quantile values = 0.5 (median), 0.8 and 0.9

Software Requirement 5 – SR5: Enter the constraint to obtain the vehicle's immobility.

Requirement to be coded: input constraint value.

Software Requirement 6 – SR6: Model fitting: Enter desired number of clusters/centroids.

Requirement to be coded:

- input number of clusters K,
- feed K to the K-Means algorithm,
- receive the K centroid values from the K-Means algorithm,
- print the K centroid values.

Software Requirement 7 – SR7: Enter data points and centroids for the scatter graph.

Plot the cluster data points, add the centroid values and add these to the graph in another colour, so as to visually observe the idling engine speeds.

Requirement to be coded: input the data points and the K centroid values.

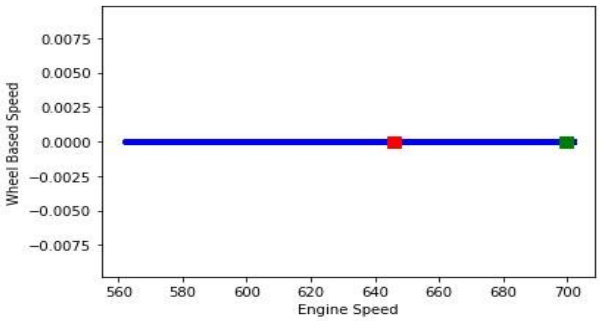

Figure 2 - Example of identified clusters

(Data points with wheel speed = 0 and both centroids on the X-axis).

3.2 Measurement.

In the Software Requirements SR1 to SR7 the following data movements can be identified.

Size of SR 1: Receive Python library names.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Library name	Name of the library to be imported into the code environment specified by the data scientist	1
Size of SR 1			**1 CFP**

(Continued)

Size of SR 2: Specify file location/name and column titles.

Data Movement	Data Group	Explanation	Size in CFP
Entry	File location, file name	The data scientist specifies the file location and file name of the raw data to be uploaded	1
Entry	Column title	The column titles of a data frame to format the raw data.	1
Size of SR 2			**2 CFP**

Size of SR 3: Enter selection parameters for data points removal.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Selection parameters	The selected data points are unneeded and removed from the data frame.	1
Size of SR 3			**1 CFP**

Size of SR 4: Enter quantile values to tag the data points.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Quantile value	The data scientist enters the desired quantile values to tag the data points so as to remove the vehicle's actual working data points.	1
Size of SR 4			**1 CFP**

Size of SR 5: Enter the constraint to obtain the vehicle's immobility.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Constraint value	Obtain the vehicle's immobility time using the tagging values in the state column and put the results in the idle column. Remove data points with immobility less than the constraint.	1
Size of SR 5			**1 CFP**

(Continued)

Size of SR 6: Model fitting: Enter desired number of clusters/centroids.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Number of clusters	The data scientist enters the desired number of clusters K into the K-Means algorithm so as to determine the K centroids	1
Exit	Number of clusters	Number of clusters to the K-Means algorithm	1
Entry	Centroid value	Receipt of centroid values from the K-Means algorithm	1
Exit	Centroid value	Print centroid values	1
Size of SR 6			**4 CFP**

Size of SR 7: Enter data points and centroids for the scatter graph.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Data point	Cluster data points to be displayed	1
Entry	Centroid	Centroids to be displayed	1
Size of SR 7			**2 CFP**

The functional size within the scope of the AI software is the sum of the sizes of the Software Requirements SR1 to SR7, that is:

1 + 2 + 1 + 1 + 1 + 4 + 2 CFP = **12 CFP**

4 EXAMPLE 2: LINEAR REGRESSION WITH A MACHINE LEARNING ALGORITHM.

4.1 Software Functional Requirements.

This section describes the FUR of the AI software to be measured from the data scientist viewpoint, i.e. the data scientist's requirements to develop the linear regression algorithm.

Software Requirement 1 – SR1: Receive Python library names.

The Python library names specified by the data scientist are:

- **Numpy**: for numerical operations
- **Pandas**: for data manipulation and analysis
- **Scipy**: for using the mathematics and scientific computing (including the identification of outliers)
- **Sklearn.liner_model**: for linear regression algorithm.

Requirements to be coded: input Library names

Software Requirement 2 – SR2: Upload the raw data and create a data frame.

Requirements to be coded: input

a) raw data file location and file name,

b) column titles of the data frame: date/time, air filter restriction, total engine hours and engine speed.

Software Requirement 3 – SR3: Create one empty column in the existing data frame for further calculation.

Calculate the duration of each cycle the air filter is changed by the maintenance and put the numbers in the 'hours' column.

Requirement to be coded: column title ('Hours'), calculate cycle duration and fill the Hours column with these.

Software Requirement 4 – SR4: Process the raw data.

Requirement to be coded: Removed unneeded data from the data frame:

- Total engine hours = 0 (the engine is in offline mode), air filter < 0 (the air filter does not work) and engine speed more than quantile (0.25) or average, depends on data scientist judgment (e.g. engine works with power, so the air filter works),
- Outliers, using the z-score method

The start and end of each cycle and duration of each cycle into the Hours column.

Software Requirement 5 – SR5: Define the regression variables, inputs and training sets.

Requirement to be coded:

- the independent variable (x,hours)
- the dependent variable (y,air filter restriction indicator)
- the training set.

(Continued)

Software Requirement 6 – SR6: Apply the linear regression function.

Requirement to be coded: Enter the linear regression function, then fit the model to the training set.

Software Requirement 7 – SR7: Obtain the calculated equation.

Derive the estimated remaining hours at prediction date (e.g., if the maximum admissible indicator value pair is (x2,y2) and at prediction date there is (x1,y1) then the remaining hours are x2 – x1).

Requirement to be coded: subtract y and the hours at prediction date.

4.2 Measurement

In the Software Requirements SR1 to SR7 the following data movements can be identified.

Size of SR 1: Receive Python library names.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Library name	Name of the library to be imported into the code platform, specified by the data scientist	1
Size of SR 1			**1 CFP**

Size of SR 2: Upload the raw data and create a data frame.

Data Movement	Data Group	Explanation	Size in CFP
Entry	File location, file name	The data scientist specifies the file location and file name of the raw data to be uploaded	1
Entry	Column title	The column titles of a data frame to format the raw data.	1
Size of SR 2			**2 CFP**

Size of SR 3: Create one empty column in the existing data frame for further calculation.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Column name	To add empty column	1
Write	Hours per cycle	Store cycle durations	1
Size of SR 3			**2 CFP**

(Continued)

Size of SR 4: Process the raw data.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Parameters	Unneeded values	1
Entry	z-score function	Removes noise	1
Entry	find_peak function	Find start and end of each cycle	1
Entry	Mathematical formula	Calculates duration of each cycle in hours	1
Size of SR 4			**4 CFP**

Size of SR 5: Define the regression variables, inputs and training sets.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Inputs	Enter Inputs	1
Entry	Equation	Enter Equation	1
Entry	Variables	Enter Variables	1
Entry	Training sets	Enter Training sets	1
Size of SR 5			**4 CFP**

Size of SR 6: Apply the linear regression function.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Linear function	Enter linear function	1
Exit	Fit command	Enter fit command	1
Size of SR 6			**2 CFP**

Size of SR 7: Obtain the calculated equation.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Request	Request of the calculated results	1
Size of SR 7			**1 CFP**

The functional size within the scope of the AI software is the sum of the sizes of the Software Requirements SR1 to SR7, that is:

1 + 2 + 2 + 4 + 4 + 2 + 1 CFP = **16 CFP**

5 EXAMPLE 3: AI DASHBOARD

5.1 Software Functional Requirements.

The AI dashboard is a user-friendly tool for someone without programming knowledge who wants to see a display of data, such as senior managers and stakeholders. The user interface allows one to select, choose, upload, and execute data without touching the script. The staff using the same domain in a company can connect to it by the URL on their web browser.

The AI dashboard supports data loading, statistical summaries, adding equations, applying filters and visualize data.

This section describes the FUR of the AI software to be measured from the data scientist viewpoint, i.e. the data scientist's requirements to develop the linear regression algorithm.

Software Requirement 1 – SR1: Receive Python library names.

The Python library names specified by the data scientist are:

- **Numpy**: for numerical operations
- **Pandas**: for data manipulation and analysis
- **Streamlit**: a framework for ML projects
- **mysql.connector**: to build database applications
- **seaborn**: for demonstrating the correlation between values
- **plotly express**: for data visualization
- **plotly subplot**: for merging plotly objects into a single view

Requirements to be coded: input Library names

Software Requirement 2 – SR2: Create checkbox for the required steps.

The checkbox acts as trigger for the required functionalities and allows the software to apply the functions.

Requirements to be coded: enter checkbox command.

Software Requirement 3 – SR3: Create a widget list for uploading data.

Create a list of options and based on the chosen option the next step appears to the user.

Requirement to be coded: enter a list of options

Software Requirement 4 – SR4: Create a widget list of equipment.

The data scientist defines and enters a list of input equipment

Requirement to be coded: create a list of available equipment for user selection.

Software Requirement 5 – SR5: Connect Python to the SQL database.

Link Python to the SQL database to be able to query the data directly from the SQL database.

Requirement to be coded: enter the connection function and the server credentials

Software Requirement 6 – SR6: Create date picker sidebar widgets.

The date picker widget enables to choose a start and end date for uploading data.

Requirement to be coded: define and enter the sidebar function

Software Requirement 7 – SR7: Create a path for uploading data from a local computer.

Define and create a path (file location).

Requirement to be coded: enter a path

Software Requirement 8 – SR8: Create Excel/csv file picker widgets to drag the data file.

Define and enter the file picker widget. The file picker allows to drag the file from the computer into a box dragger of the web browser.

Requirement to be coded: enter the file picker widget.

Software Requirement 9 – SR9: Upload data and create data frame.

Add cache function for performance, it avoids the software to upload data every time for the operation and browsing. When the data is uploaded and columns are chosen, the data frame is created. A checkbox is created to allow to create the data frame.

Requirement to be coded: install the cache function, upload the data, define the columns and create the data frame.

Software Requirement 10 – SR10: Exert statistics summary.

The “describe” function provides the statistics summary of the created data frame. A checkbox is created to allow to present the summary.

Requirement to be coded: enter the “describe” function.

Software Requirement 11 – SR11: Create box for equations.

A textbox is created, and the calculation is applied to the data frame using the “eval” command.

Requirement to be coded: define the textbox function and enter the “eval” command

Software Requirement 12 – SR12: Create box for filter selection.

A textbox is created, and the calculation is applied to the data frame using “eval” command. The coding commands for the equations and filter selection are different, so need to be created separately.

Requirement to be coded: define the textbox function and enter the “eval” command.

Software Requirement 13 – SR13: Create seaborn heatmap plot.

The seaborn plot can be created using the seaborn function in the seaborn library.

Requirement to be coded: enter the “seaborn.heatmap” function

Software Requirement 14 – SR14: Create scatter plot.

The scatter plot can be created using the px.scatter function from the plotly_express library. The X and Y variables need to be defined by the selectbox function.

Requirement to be coded: Enter the “px.scatter” function and enter corresponding inputs (x and y) using the selectbox function

Software Requirement 15 – SR15: Create multi-line chart.

The multi-line chart can be created using the plotly_express library and subplot function. The X and Y variables need to be defined by the selectbox and multiselect functions.

Requirement to be coded: enter the “make_subplots” function, define corresponding inputs X using the selectbox function, define corresponding inputs Y using the multiselect function and enter the “go.line” function to plot the y variables in the order of choice.

5.2 Measurement

In the Software Requirements SR1 to SR7 the following data movements can be identified.

Size of SR 1: Receive Python library names.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Library name	Name of the library to be imported into the code platform	1
Size of SR 1			**1 CFP**

Size of SR 2: Create checkbox for the required steps.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Checkbox command	Enter checkbox command	1
Size of SR 2			**1 CFP**

Size of SR 3: Create a widget list for uploading data.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Options	Define and enter a list of options	1
Size of SR 3			**1 CFP**

Size of SR 4: Create a widget list of equipment.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Equipment	Enter a list of input equipment	1
Size of SR 4			**1 CFP**

Size of SR 5: Connect Python to the SQL database.

Data Movement	Data Group	Explanation	Size in CFP
Entry	SQL connection	Enter the connection function	1
Entry	Server credentials	Enter server credentials	1
Size of SR 5			**2 CFP**

(Continued)

Size of SR 6: Create date picker sidebar widgets.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Sidebar function	Define and enter the sidebar function	1
Size of SR 6			**1 CFP**

Size of SR 7: Create a path for uploading data from a local computer.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Path	Define and enter a path	1
Size of SR 7			**1 CFP**

Size of SR 8: Create Excel/csv file picker widgets to drag the data file.

Data Movement	Data Group	Explanation	Size in CFP
Entry	File picker widget	Enter the file picker widget	1
Size of SR 8			**1 CFP**

Size of SR 9: Upload data and create data frame.

Data Movement	Data Group	Explanation	Size in CFP
Entry	“@st.cache” function	Enter cache function	1
Entry	“pd.read” function	Upload data using “pd.read” function	1
Entry	“multiselect” function	Choose the columns	1
Entry	“pd.dataframe” function	Enter the pd.dataframe to create data frame	1
Size of SR 9			**4 CFP**

Size of SR 10: Exert statistics summary.

Data Movement	Data Group	Explanation	Size in CFP
Entry	enters “describe” function	Enter the “describe” function	1
Size of SR 10			**1 CFP**

Size of SR 11: Exert Create box for equations.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Textbox function	Define and enter the textbox function	1
Entry	“eval” command	Enter the “eval” command	1
Size of SR 11			**2 CFP**

(Continued)

Size of SR 12: Create box for filter selection.

Data Movement	Data Group	Explanation	Size in CFP
Entry	Textbox function	Define and enter the textbox function	1
Entry	“eval” command	Enter the “eval” command	1
Size of SR 12			**2 CFP**

Size of SR 13: Create seaborn heatmap plot.

Data Movement	Data Group	Explanation	Size in CFP
Entry	“seaborn.heatmap” function	Enter the “seaborn.heatmap” function	1
Size of SR 13			**1 CFP**

Size of SR 14: Create scatter plot.

Data Mvnt	Data Group	Explanation	Size in CFP
Entry	“px.scatter” function	Enter the “px.scatter” function	1
Entry	selectbox function	Enter inputs using selectbox function	1
Size of SR 14			**2 CFP**

Size of SR 15: Create multi-line chart.

Data Mvnt	Data Group	Explanation	Size in CFP
Entry	“make_subplots” function	Enter “make_subplots” function	1
Entry	selectbox function	Enter corresponding inputs X	1
Entry	multiselect function	Enter corresponding inputs Y	1
Entry	receives “go.line” function	Enter the “go.line” function to plot the y variables in the order of choice.	1
Size of SR 15			**4 CFP**

The functional size within the scope of the AI software is the sum of the sizes of the Software Requirements SR1 to SR15, that is:

1 + 1 + 1 +1 + 2 + 1 + 1 + 1 + 4 + 1 + 2 + 2 + 1 + 2+ 4 CFP = **25 CFP**

6 REFERENCES AND DOCUMENTATION.

All COSMIC publications are available for free download from the Knowledge Base of www.cosmic-sizing.org.

[1] Microsoft Word - VEDS03598906_FinalSubmission (1).docx

[2] Measurement Manual v5.0, Parts 1, 2 and 3.

[3] What is a COSMIC Function Point?

[4] Neural networks and deep learning, M. Nielsen, Determination Press, 2015 (available at neuralnetworksanddeeplearning.com)

[5] Sizing software in a Machine Learning context: A COSMIC Case study.

[6] Abran, A., Software Project Estimation, John Wiley & Sons, 2015.