MLProvCodeGen - User Evaluation May 2022

Results

Survey 861741

Number of records in this query:	12
Total records in survey:	12
Percentage of total:	100.00%

Summary for Q00

Our first exercise is to open MLProvCodeGen and inspect its user interface. The application should be available from the link you opened at the start. The JupyterLab interface should look like this: Please proceed by pressing the 'MLProvCodeGen' button located in the 'other' section to open the extension. There are 3 main parts to this opening page: The header The provenance file input The input fields to generate code with Today, we will be mostly concerned with the 3rd part, since our main focus in this evaluation is to test user experience with generating machine learning code from these input fields. For our first exercise, please select 'Multiclass Classification' as your desired ML exercise and press the submit button. Note: You can press the 'Reset' button at the top at any point in this process if you feel like you made a mistake. After pressing the submit button, are you able to see new input elements? Note: In case you encounter any error messages or can't open access MLProvCodeGen, please try reloading the page or waiting a few minutes. There are several errors that might occur, however most of them can be solved by reloading and/or waiting.

Answer	Count	Percentage
Yes (Y)	12	100.00%
No (N)	0	0.00%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G01Q13

We now hope to evaluate MLProvCodeGen's user interface and the explanations given to users. Before we start, please tell us what field you research or work in (Computer Science, Bioinformatics, Engineering, ...). Note: If you don't want to answer this question, you can leave the input field blank as this question is not mandatory.

Answer	Count	Percentage
Answer	8	66.67%
No answer	4	33.33%
Not displayed	0	0.00%

ID	Response
7	Computer Science
9	research data management
17	Computer Science
21	Data science
22	Computer Science
26	Remote Sensing/Geoinformatics
27	Computer Science
28	Geographic information science

Summary for G02Q02(SQ001)[I have [...] theoretical knowledge of machine learning (from lectures etc.).]

Please rate your experience with machine learning experiments, data provenance, and reproducibility.

Answer	Count	Percentage
Poor (AO01)	1	8.33%
Fair (AO02)	6	50.00%
Average (AO03)	2	16.67%
Good (AO05)	2	16.67%
Excellent (AO04)	1	8.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q02(SQ002)[I have [...] knowledge of common terms and definitions used in ML.]

Please rate your experience with machine learning experiments, data provenance, and reproducibility.

Answer	Count	Percentage
Poor (AO01)	2	16.67%
Fair (AO02)	5	41.67%
Average (AO03)	2	16.67%
Good (AO05)	0	0.00%
Excellent (AO04)	3	25.00%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q02(SQ003)[I have [...] experience implementing a ML exercise before.]

Please rate your experience with machine learning experiments, data provenance, and reproducibility.

Answer	Count	Percentage
Poor (AO01)	7	58.33%
Fair (AO02)	0	0.00%
Average (AO03)	2	16.67%
Good (AO05)	1	8.33%
Excellent (AO04)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q02(SQ004)[I have [...] knowledge of reproducibility in scientific contexts.]

Please rate your experience with machine learning experiments, data provenance, and reproducibility.

Answer	Count	Percentage
Poor (AO01)	2	16.67%
Fair (AO02)	2	16.67%
Average (AO03)	3	25.00%
Good (AO05)	2	16.67%
Excellent (AO04)	3	25.00%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q02(SQ005)[I have [...] knowledge of scientific data management and FAIR data (Findable, Accessible, Interoperable, Reusable data).]

Please rate your experience with machine learning experiments, data provenance, and reproducibility.

Answer	Count	Percentage
Poor (AO01)	1	8.33%
Fair (AO02)	0	0.00%
Average (AO03)	6	50.00%
Good (AO05)	4	33.33%
Excellent (AO04)	1	8.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q02(SQ006)[I have [...] knowledge of provenance data.]

Please rate your experience with machine learning experiments, data provenance, and reproducibility.

Answer	Count	Percentage
Poor (AO01)	4	33.33%
Fair (AO02)	1	8.33%
Average (AO03)	4	33.33%
Good (AO05)	2	16.67%
Excellent (AO04)	1	8.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q06(SQ001)[Please rate your experience using Jupyter Notebooks.]

Please rate your experience using Jupyter Notebooks.

Answer	Count	Percentage
No Experience (AO01)	1	8.33%
Beginner (AO02)	6	50.00%
Advanced (AO03)	3	25.00%
Proficient (AO04)	0	0.00%
Expert (AO05)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q03(SQ001)[I expect this exercise to be [...].]

At a glance, please rate how difficult you think this exercise will be based on your first impressions of the user interface and input fields.

Answer	Count	Percentage
Difficult (AO01)	1	8.33%
Slightly Challenging (AO02)	1	8.33%
Appropriately Challenging (AO03)	6	50.00%
Fairly Easy (AO04)	2	16.67%
Easy (AO05)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q04(SQ001)[I now have a general sense of what the terms used in MLProvCodeGen mean.]

Please take a moment to familiarize yourself with the input elements. Remember that you can hover over input elements to see explanations of what specific terms mean. The input elements are ordered according to a machine learning pipeline. A ML pipeline is the general workflow that all machine learning experiments have to follow. From top to bottom, the first three sections contain details for the data used in the experiment. In data ingestion, you can select a specific dataset. This option selects the raw data, before any preprocessing. Data preparation is the pipeline step that takes the raw data from data ingestion and performs a number of operation on that data to better fit it to the machine learning task at hand and the model that we plan on using. These preprocessing steps can include resizing image files, converting coloured images to greyscale or black and white, or rescaling the tabular data that we use for the iris data set to fit specific intervals. Data segregation splits our data into training and testing datasets, which are first used to train the model and then used to evaluate its performance. The model parameters section is concerned with the model itself. For multiclass classification, we train a linear neural network (NN) with 3 layers. The number of input and output neurons depend on our input dataset, i.e. how many feature dimensions the dataset has and how many classes the dataset is split into. The number of neurons in the middle layer can be adjusted. There are also options for the NN's activation function, which defines how neurons compute their outputs given their inputs, the models optimizer, which is the function that defines how the NN's hyperparameters are adjusted to improve accuracy, the optimizer's learning rate, which specifies the magnitude of each hyperparameter adjustment, and the loss function, which evaluates the model's performance based on a loss metric. In the training step, we can adjust the number of epochs the model is trained for. For each epoch, the whole training dataset is iterated through once. Varying numbers of epochs can change the models final performance. If the number of epochs is too low, then the NN possibly didn't have enough time to optimize its hyperparameters. Given the explanations in MLProvCodeGen and those in this survey, please rate your personal understanding of the terms used.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO05)	1	8.33%
Agree (AO02)	7	58.33%
Strongly Agree (AO03)	4	33.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q04(SQ002)[I understand why the input elements are ordered in the way they are.]

Please take a moment to familiarize yourself with the input elements. Remember that you can hover over input elements to see explanations of what specific terms mean. The input elements are ordered according to a machine learning pipeline. A ML pipeline is the general workflow that all machine learning experiments have to follow. From top to bottom, the first three sections contain details for the data used in the experiment. In data ingestion, you can select a specific dataset. This option selects the raw data, before any preprocessing. Data preparation is the pipeline step that takes the raw data from data ingestion and performs a number of operation on that data to better fit it to the machine learning task at hand and the model that we plan on using. These preprocessing steps can include resizing image files, converting coloured images to greyscale or black and white, or rescaling the tabular data that we use for the iris data set to fit specific intervals. Data segregation splits our data into training and testing datasets, which are first used to train the model and then used to evaluate its performance. The model parameters section is concerned with the model itself. For multiclass classification, we train a linear neural network (NN) with 3 layers. The number of input and output neurons depend on our input dataset, i.e. how many feature dimensions the dataset has and how many classes the dataset is split into. The number of neurons in the middle layer can be adjusted. There are also options for the NN's activation function, which defines how neurons compute their outputs given their inputs, the models optimizer, which is the function that defines how the NN's hyperparameters are adjusted to improve accuracy, the optimizer's learning rate, which specifies the magnitude of each hyperparameter adjustment, and the loss function, which evaluates the model's performance based on a loss metric. In the training step, we can adjust the number of epochs the model is trained for. For each epoch, the whole training dataset is iterated through once. Varying numbers of epochs can change the models final performance. If the number of epochs is too low, then the NN possibly didn't have enough time to optimize its hyperparameters. Given the explanations in MLProvCodeGen and those in this survey, please rate your personal understanding of the terms used.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO05)	1	8.33%
Agree (AO02)	6	50.00%
Strongly Agree (AO03)	5	41.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q05(SQ001)[I understand the purpose of each code cell.]

We now ask you to generate a ML script to train and evaluate a model to classify three different species of iris flowers. Please use the user interface to input the following values: Dataset: Iris Number of Epochs: 10 Note: Please keep all other input elements at their default values. Now, please submit your values and open the generated notebook through the buttons at the bottom of the page. Note: You can use the reset button at the top of the page at any point to reset the application. The generated notebook is now open in a new browser tab, please switch to that tab. The first prompt asks you select a kernel, please press the 'Select' button. As your first action we ask you to use the 'Restart Kernel and Run All Cells...' button at the top toolbar as seen in this picture: Please also confirm that you want to restart the kernel by pressing the 'Restart' button. The kernel will now be used to execute the whole notebook, in the meantime we can get familiar with the structure of the notebook: Jupyter Notebooks are a useful tool for coding because they can be used to split a script into many code cells that can be executed seperately. The notebooks generated by MLProvCodeGen use cells to give logical structure to scripts and to better support human-readability. In general, the cells of this notebook are ordered along the structure of a machine learning pipeline, exactly like the input elements of our application. However, there are some peripheral requirements that have to be met before data ingestion. These are addressed in the first three cells: Install required Python packages so we can import them to this notebook. Import the packages so we can use them in our code. Setup provenance data capture and capture provenance data of the experiment in general, packages used and their versions, and hardware information (Note: Hardware information is captured from the virtual machine, not your local PC). Each cell is devided into two parts, the code itself and the provenance data capture of the processes of that cell. These parts are seperated through the lines of code that capture start-, end-, and execution time. After the evaluation cells, the last part of the notebook is concerned with generating a graph of captured provenance data and writing the data to a .json file. At this point we invite you to inspect the notebook in as much detail as you want to up to the 'Generate Provenance Data' cell. Note: We will go into more detail on the provenance graph on the next page. We now ask you to rate your understanding of the notebook.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	0	0.00%
Neutral (AO03)	2	16.67%
Agree (AO04)	8	66.67%
Strongly Agree (AO05)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q05(SQ002)[I understand the purpose of the code within each cell.]

We now ask you to generate a ML script to train and evaluate a model to classify three different species of iris flowers. Please use the user interface to input the following values: Dataset: Iris Number of Epochs: 10 Note: Please keep all other input elements at their default values. Now, please submit your values and open the generated notebook through the buttons at the bottom of the page. Note: You can use the reset button at the top of the page at any point to reset the application. The generated notebook is now open in a new browser tab, please switch to that tab. The first prompt asks you select a kernel, please press the 'Select' button. As your first action we ask you to use the 'Restart Kernel and Run All Cells...' button at the top toolbar as seen in this picture: Please also confirm that you want to restart the kernel by pressing the 'Restart' button. The kernel will now be used to execute the whole notebook, in the meantime we can get familiar with the structure of the notebook: Jupyter Notebooks are a useful tool for coding because they can be used to split a script into many code cells that can be executed seperately. The notebooks generated by MLProvCodeGen use cells to give logical structure to scripts and to better support human-readability. In general, the cells of this notebook are ordered along the structure of a machine learning pipeline, exactly like the input elements of our application. However, there are some peripheral requirements that have to be met before data ingestion. These are addressed in the first three cells: Install required Python packages so we can import them to this notebook. Import the packages so we can use them in our code. Setup provenance data capture and capture provenance data of the experiment in general, packages used and their versions, and hardware information (Note: Hardware information is captured from the virtual machine, not your local PC). Each cell is devided into two parts, the code itself and the provenance data capture of the processes of that cell. These parts are seperated through the lines of code that capture start-, end-, and execution time. After the evaluation cells, the last part of the notebook is concerned with generating a graph of captured provenance data and writing the data to a .json file. At this point we invite you to inspect the notebook in as much detail as you want to up to the 'Generate Provenance Data' cell. Note: We will go into more detail on the provenance graph on the next page. We now ask you to rate your understanding of the notebook.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	2	16.67%
Neutral (AO03)	2	16.67%
Agree (AO04)	7	58.33%
Strongly Agree (AO05)	1	8.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q05(SQ003)[The visualizations of the data help my understanding of the notebook.]

We now ask you to generate a ML script to train and evaluate a model to classify three different species of iris flowers. Please use the user interface to input the following values: Dataset: Iris Number of Epochs: 10 Note: Please keep all other input elements at their default values. Now, please submit your values and open the generated notebook through the buttons at the bottom of the page. Note: You can use the reset button at the top of the page at any point to reset the application. The generated notebook is now open in a new browser tab, please switch to that tab. The first prompt asks you select a kernel, please press the 'Select' button. As your first action we ask you to use the 'Restart Kernel and Run All Cells...' button at the top toolbar as seen in this picture: Please also confirm that you want to restart the kernel by pressing the 'Restart' button. The kernel will now be used to execute the whole notebook, in the meantime we can get familiar with the structure of the notebook: Jupyter Notebooks are a useful tool for coding because they can be used to split a script into many code cells that can be executed seperately. The notebooks generated by MLProvCodeGen use cells to give logical structure to scripts and to better support human-readability. In general, the cells of this notebook are ordered along the structure of a machine learning pipeline, exactly like the input elements of our application. However, there are some peripheral requirements that have to be met before data ingestion. These are addressed in the first three cells: Install required Python packages so we can import them to this notebook. Import the packages so we can use them in our code. Setup provenance data capture and capture provenance data of the experiment in general, packages used and their versions, and hardware information (Note: Hardware information is captured from the virtual machine, not your local PC). Each cell is devided into two parts, the code itself and the provenance data capture of the processes of that cell. These parts are seperated through the lines of code that capture start-, end-, and execution time. After the evaluation cells, the last part of the notebook is concerned with generating a graph of captured provenance data and writing the data to a .json file. At this point we invite you to inspect the notebook in as much detail as you want to up to the 'Generate Provenance Data' cell. Note: We will go into more detail on the provenance graph on the next page. We now ask you to rate your understanding of the notebook.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	0	0.00%
Neutral (AO03)	2	16.67%
Agree (AO04)	5	41.67%
Strongly Agree (AO05)	5	41.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q05(SQ004)[The lines printed at the end of the 'Model' cell help my understanding of the neural network.]

We now ask you to generate a ML script to train and evaluate a model to classify three different species of iris flowers. Please use the user interface to input the following values: Dataset: Iris Number of Epochs: 10 Note: Please keep all other input elements at their default values. Now, please submit your values and open the generated notebook through the buttons at the bottom of the page. Note: You can use the reset button at the top of the page at any point to reset the application. The generated notebook is now open in a new browser tab, please switch to that tab. The first prompt asks you select a kernel, please press the 'Select' button. As your first action we ask you to use the 'Restart Kernel and Run All Cells...' button at the top toolbar as seen in this picture: Please also confirm that you want to restart the kernel by pressing the 'Restart' button. The kernel will now be used to execute the whole notebook, in the meantime we can get familiar with the structure of the notebook: Jupyter Notebooks are a useful tool for coding because they can be used to split a script into many code cells that can be executed seperately. The notebooks generated by MLProvCodeGen use cells to give logical structure to scripts and to better support human-readability. In general, the cells of this notebook are ordered along the structure of a machine learning pipeline, exactly like the input elements of our application. However, there are some peripheral requirements that have to be met before data ingestion. These are addressed in the first three cells: Install required Python packages so we can import them to this notebook. Import the packages so we can use them in our code. Setup provenance data capture and capture provenance data of the experiment in general, packages used and their versions, and hardware information (Note: Hardware information is captured from the virtual machine, not your local PC). Each cell is devided into two parts, the code itself and the provenance data capture of the processes of that cell. These parts are seperated through the lines of code that capture start-, end-, and execution time. After the evaluation cells, the last part of the notebook is concerned with generating a graph of captured provenance data and writing the data to a .json file. At this point we invite you to inspect the notebook in as much detail as you want to up to the 'Generate Provenance Data' cell. Note: We will go into more detail on the provenance graph on the next page. We now ask you to rate your understanding of the notebook.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	1	8.33%
Neutral (AO03)	3	25.00%
Agree (AO04)	6	50.00%
Strongly Agree (AO05)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q05(SQ005)[The training vizualization helps my understanding of the training process.]

We now ask you to generate a ML script to train and evaluate a model to classify three different species of iris flowers. Please use the user interface to input the following values: Dataset: Iris Number of Epochs: 10 Note: Please keep all other input elements at their default values. Now, please submit your values and open the generated notebook through the buttons at the bottom of the page. Note: You can use the reset button at the top of the page at any point to reset the application. The generated notebook is now open in a new browser tab, please switch to that tab. The first prompt asks you select a kernel, please press the 'Select' button. As your first action we ask you to use the 'Restart Kernel and Run All Cells...' button at the top toolbar as seen in this picture: Please also confirm that you want to restart the kernel by pressing the 'Restart' button. The kernel will now be used to execute the whole notebook, in the meantime we can get familiar with the structure of the notebook: Jupyter Notebooks are a useful tool for coding because they can be used to split a script into many code cells that can be executed seperately. The notebooks generated by MLProvCodeGen use cells to give logical structure to scripts and to better support human-readability. In general, the cells of this notebook are ordered along the structure of a machine learning pipeline, exactly like the input elements of our application. However, there are some peripheral requirements that have to be met before data ingestion. These are addressed in the first three cells: Install required Python packages so we can import them to this notebook. Import the packages so we can use them in our code. Setup provenance data capture and capture provenance data of the experiment in general, packages used and their versions, and hardware information (Note: Hardware information is captured from the virtual machine, not your local PC). Each cell is devided into two parts, the code itself and the provenance data capture of the processes of that cell. These parts are seperated through the lines of code that capture start-, end-, and execution time. After the evaluation cells, the last part of the notebook is concerned with generating a graph of captured provenance data and writing the data to a .json file. At this point we invite you to inspect the notebook in as much detail as you want to up to the 'Generate Provenance Data' cell. Note: We will go into more detail on the provenance graph on the next page. We now ask you to rate your understanding of the notebook.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	1	8.33%
Neutral (AO03)	3	25.00%
Agree (AO04)	5	41.67%
Strongly Agree (AO05)	3	25.00%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G02Q05(SQ006)[The evaluation graphs and metrics help my understanding of the end result.]

We now ask you to generate a ML script to train and evaluate a model to classify three different species of iris flowers. Please use the user interface to input the following values: Dataset: Iris Number of Epochs: 10 Note: Please keep all other input elements at their default values. Now, please submit your values and open the generated notebook through the buttons at the bottom of the page. Note: You can use the reset button at the top of the page at any point to reset the application. The generated notebook is now open in a new browser tab, please switch to that tab. The first prompt asks you select a kernel, please press the 'Select' button. As your first action we ask you to use the 'Restart Kernel and Run All Cells...' button at the top toolbar as seen in this picture: Please also confirm that you want to restart the kernel by pressing the 'Restart' button. The kernel will now be used to execute the whole notebook, in the meantime we can get familiar with the structure of the notebook: Jupyter Notebooks are a useful tool for coding because they can be used to split a script into many code cells that can be executed seperately. The notebooks generated by MLProvCodeGen use cells to give logical structure to scripts and to better support human-readability. In general, the cells of this notebook are ordered along the structure of a machine learning pipeline, exactly like the input elements of our application. However, there are some peripheral requirements that have to be met before data ingestion. These are addressed in the first three cells: Install required Python packages so we can import them to this notebook. Import the packages so we can use them in our code. Setup provenance data capture and capture provenance data of the experiment in general, packages used and their versions, and hardware information (Note: Hardware information is captured from the virtual machine, not your local PC). Each cell is devided into two parts, the code itself and the provenance data capture of the processes of that cell. These parts are seperated through the lines of code that capture start-, end-, and execution time. After the evaluation cells, the last part of the notebook is concerned with generating a graph of captured provenance data and writing the data to a .json file. At this point we invite you to inspect the notebook in as much detail as you want to up to the 'Generate Provenance Data' cell. Note: We will go into more detail on the provenance graph on the next page. We now ask you to rate your understanding of the notebook.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	0	0.00%
Neutral (AO03)	3	25.00%
Agree (AO04)	5	41.67%
Strongly Agree (AO05)	4	33.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G03Q07

We now ask you to inspect the generated provenance graph.

Note: As the graph is a large image, please enhance it by either double clicking on it in your current window, or using the URLs printed above to open the image in a new browser tab.

The provenance graph compiles all data points captured from executing this notebook. It has three parts:

Bottom Left to Bottom Right: The machine learning pipeline is represented from start to finish. Each pipeline step is depicted as a cell, a call to execute said cell, and the data that was generated by executing it.
Top Right: Information about the experiment, hardware, and packages is represented seperately from the pipeline steps.
Middle: The ex:notebook entity is the center part of this graph. It represents the notebook itself, that it was generated by MLProvCodeGen, who it was authored by, and outgoing lines to each notebook cell.

Please inspect the generated evaluation data at the bottom right of the provenance graph and input the 'accuracy' into the number input field below.

Note: Please input the number in the format of 0.XXX

Calculation	Result
Count	12
Sum	674.3203333134
Standard deviation	183.86
Average	56.19
Minimum	0.0000000000
1st quartile (Q1)	0.72175
2nd quartile (Median)	0.8495
3rd quartile (Q3)	0.982583334325
Maximum	666.0000000000

Null values are ignored in calculations
Q1 and Q3 calculated using minitab method

Summary for G03Q14

For the second user task, we want to evaluate how easy or hard it is to find information in the provenance graph without giving users any directions. Execution of the function 'set_experiment_info()' generates 'Experiment Info Data' that includes a datapoint called 'creation date'. To the best of your ability, please input the function/activity that informs 'set_experiment_info()' of the creation date. If you could not find the information, please indicate that by inputting 'N/A'.

Answer	Count	Percentage
Answer	12	100.00%
No answer	0	0.00%
Not displayed	0	0.00%

ID	Response
7	ex:Experiment Info Date
9	ex:creation_date
10	Experiment Info Data
12	N/A
14	ex:date.today()
17	N/A
18	N/A
21	a_setexperimentinfo = d1.activity('ex:set_experiment_info()') a_setdate = d1.activity('ex:date.today()') d1.wasStartedBy(a_setexperimentinfo,e_experimentinfo, time=datetime.datetime.now()) d1.wasInformedBy(a_setexperimentinfo, a_setdate) d1.hadMember(e_notebook, e_experimentinfo) e_experimentinfo_data = d1.entity('ex:Experiment Info Data',(('ex:title', 'Multiclass Classification'), ('ex:creation_date', str(date.today())), ('ex:task_type', 'MulticlassClassification'),)) d1.wasGeneratedBy(e_experimentinfo_data, a_setexperimentinfo)
22	Data Ingestion Data
26	ex:date.today()
27	N/A
28	ex:date.today()

Summary for G03Q08(SQ001)[It was easy to find the 'Accuracy' in the provenance graph.]

In this part, we ask you to rate the clarity and level of information of the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	3	25.00%
Agree (AO03)	5	41.67%
Stongly Agree (AO04)	4	33.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G03Q08(SQ002)[It was easy to complete the second user task.]

In this part, we ask you to rate the clarity and level of information of the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	8	66.67%
Agree (AO03)	3	25.00%
Stongly Agree (AO04)	1	8.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G03Q08(SQ003)[The data shown in the graph allows me to see the most important parts of the ML experiment.]

In this part, we ask you to rate the clarity and level of information of the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	4	33.33%
Agree (AO03)	8	66.67%
Stongly Agree (AO04)	0	0.00%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G03Q08(SQ004)[There is too much clutter in the graph/ It's hard to get an overview.]

In this part, we ask you to rate the clarity and level of information of the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	2	16.67%
Agree (AO03)	6	50.00%
Stongly Agree (AO04)	4	33.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G03Q08(SQ005)[The relational information shown in the graph (labeled lines that connect datapoints) allows me to understand how datapoints influence each other.]

In this part, we ask you to rate the clarity and level of information of the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	2	16.67%
Agree (AO03)	9	75.00%
Stongly Agree (AO04)	1	8.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G04Q09(SQ001)[Overall, I had a good experience using MLProvCodeGen.]

Finally, we ask you to answer some general questions about your experience with MLProvCodeGen, the generated notebook, and the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	0	0.00%
Agree (AO03)	11	91.67%
Strongly Agree (AO04)	1	8.33%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G04Q09(SQ002)[This program has issues that negatively impact my user experience (If you agree, then please state the problem in the free-text box below).]

Finally, we ask you to answer some general questions about your experience with MLProvCodeGen, the generated notebook, and the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	3	25.00%
Disagree (AO02)	3	25.00%
Agree (AO03)	6	50.00%
Strongly Agree (AO04)	0	0.00%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G04Q09(SQ003)[Given the explanations in this survey and the program itself, I can use MLProvCodeGen.]

Finally, we ask you to answer some general questions about your experience with MLProvCodeGen, the generated notebook, and the provenance graph.

Answer	Count	Percentage
Strongly Disagree (AO01)	0	0.00%
Disagree (AO02)	1	8.33%
Agree (AO03)	9	75.00%
Strongly Agree (AO04)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G04Q10(SQ001)[I would recommend MLProvCodeGen to a beginner.]

If you knew someone who works with or is interested in machine learning, implementing machine learning problems, or data provenance and reproducibility; how likely would you be to recommend MLProvCodeGen to them considering their experience in the field?

Answer	Count	Percentage
Would not recommend - 0 (AO01)	0	0.00%
1 (AO02)	0	0.00%
2 (AO03)	2	16.67%
3 (AO04)	1	8.33%
4 (AO05)	1	8.33%
5 (AO06)	2	16.67%
6 (AO07)	1	8.33%
7 (AO08)	1	8.33%
8 (AO09)	1	8.33%
9 (AO10)	1	8.33%
Would absolutely recommend - 10 (AO11)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G04Q10(SQ002)[I would recommend MLProvCodeGen to a domain expert.]

If you knew someone who works with or is interested in machine learning, implementing machine learning problems, or data provenance and reproducibility; how likely would you be to recommend MLProvCodeGen to them considering their experience in the field?

Answer	Count	Percentage
Would not recommend - 0 (AO01)	0	0.00%
1 (AO02)	0	0.00%
2 (AO03)	2	16.67%
3 (AO04)	0	0.00%
4 (AO05)	0	0.00%
5 (AO06)	1	8.33%
6 (AO07)	4	33.33%
7 (AO08)	1	8.33%
8 (AO09)	0	0.00%
9 (AO10)	4	33.33%
Would absolutely recommend - 10 (AO11)	0	0.00%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G04Q12(SQ001)[Completing the survey was [...].]

Please rate how difficult completing the survey actually was.

Answer	Count	Percentage
Difficult (AO01)	0	0.00%
Slightly Challenging (AO02)	2	16.67%
Appropriately Challenging (AO03)	3	25.00%
Fairly Easy (AO04)	5	41.67%
Easy (AO05)	2	16.67%
No answer	0	0.00%
Not displayed	0	0.00%

Summary for G04Q11

Finally, if you have any other comments about MLProvCodeGen, please put them here. We appreciate any kind of feedback.

Answer	Count	Percentage
Answer	5	41.67%
No answer	7	58.33%
Not displayed	0	0.00%

ID	Response
9	I saw no lines or graphics in the jupiter notebook below 2 of the mentioned cells. (I put neutral there) Double clicking the larger graphics led to an bigger picture but the scroll bars were poorly usable - really frustrating. By using copy+paste the link in another tab, led to an better usage.
22	When a datapoint meets too many arrow lines, it will be better if set them in different colors or give them orders.
26	Very nice and informative survey!
27	The imageClassification task code seems like do not work, maybe that's due to my wrong operation. Generally, it is good.
28	- too much clutter, especially too many purely technical details in provenance graph; perhaps different "views" would be good, e.g. a data analyst vs a software engineering view -> less important information could be greyed out or hidden - MAE and MSE not really appropriate for a classification task - generated code needs additional comments to make it human-readable - many applications require different settings, for example different performance measures or data splitting strategies -> users could start with the code / notebook generated by MLProvCodeGen, but I think it not straightforward to understand how the provenance information in the code chunks needs to be adapted to match the modified code. For example, it is easy to change the optimizer (Adam) in the code but to forget to update the character string that stores this information.