

CLAIMS

What is claimed is:

1. A computer-implemented data processing method comprising:
 - receiving a text-based project name input;
 - receiving a first graphical input specifying a data source;
 - accessing a data storage device associated with the data source and storing, in computer main memory, first input data from the data source, the first input data comprising one or more columns each comprising one or more of numerical values, alphanumeric characters, or text strings;
 - receiving a graphical command to invoke data exploration logic;
 - receiving a second graphical input selecting one or more columns of the first input data upon which to perform a data transformation operation;
 - receiving a third graphical input specifying a type of data transformation operation to perform;
 - applying, to a subset of the first input data contained in the one or more columns, the specified type of data transformation operation, and generating first output data from the data transformation operation;
 - generating and transmitting rendering instructions for a graphical user interface to another computing device, the graphical user interface comprising one or more graphs corresponding to the first output data;
 - storing, in non-volatile memory associated with the project name input, a first exploration session comprising the first input data, the type of data transformation operation, and the first output data;
 - receiving a third graphical input specifying a plurality of exploration sessions stored in non-volatile memory in association with the project name input, including the first exploration session;
 - generating and storing in non-volatile memory associated with the project name input, based on the plurality of exploration sessions specified in the third graphical input, a named pipeline comprising an execution graph, the execution graph programmatically expressing an order of execution of the plurality of selected exploration sessions, the execution graph specifying a pipelined relationship between the plurality of selected exploration sessions in

-40-

which output data from at least one selected exploration session comprises input data for at least one other exploration session;

initiating, graphically, the named pipeline to generate pipeline output data; and
causing to display, in a graphical user interface, one or more pipeline graphs
corresponding to the pipeline output data.

2. The method of Claim 1, further comprising:

receiving a fourth graphical input selecting one or more execution types for the named pipeline, and associating the one or more selected execution types with the named pipeline in non-volatile memory associated with the project name input; and

scheduling execution of the named pipeline for at least one of a current time or a time specified by the one or more selected execution types.

3. The method of Claim 1, further comprising receiving a fifth graphical input selecting at least one of a success handler or a failure handler for the named pipeline and associating each selected success handler or failure handler with the named pipeline in non-volatile memory associated with the project name input.

4. The method of Claim 1, the specified type of data transformation operation comprising a machine learning classification function comprising a random forest classifier;
at least one of the selected exploration sessions being programmed to execute dividing model input data into one or more training segments and one or more test segments for executing the machine learning classification function using random sampling, rolling validation, or walk forward validation; and
at least one of the one or more pipeline graphs comprising a validation curve, a learning curve, a calibration curve, or a precision and recall curve.

5. The method of Claim 4, further comprising:

after receiving the graphical command to invoke data exploration logic and before scheduling execution of the named pipeline, determining that data leakage will occur; and

-41-

in response thereto generating, in the graphical user interface, at least one of a warning that data leakage will occur or a suggestion to execute a particular methodology to prevent data leakage.

6. The method of Claim 4, further comprising, responsive to receiving the graphical command to invoke data exploration logic, executing feature selection logic to graphically designate, in the graphical user interface, at least one column of the one or more columns as likely to produce useful output.

7. The method of Claim 1, further comprising receiving the third graphical input specifying a type of data transformation operation to perform by receiving graphical input to form one or more chain transformations.

8. The method of Claim 3, further comprising receiving the fifth graphical input as selecting both a success handler and a failure handler, the selected success handler being programmed to execute one of pushing success data to a database, generating and sending an email notification of success, or transmitting a success chatroom message to a chatroom, the selected failure handler being programmed to execute one of retrying execution of the named pipeline, blocking execution of the execution graph, generating and sending an email notification of failure, or transmitting a failure chatroom message to the chatroom.

9. The method of Claim 4, further comprising:
programmatically transmitting executable instructions defining the named pipeline via a network to a plurality of server instances that have been instantiated in a cloud computing center;

programmatically instructing a first server instance of the plurality of server instances to execute the machine learning classification function; and

programmatically instructing one or more additional server instances of the plurality of server instances to execute one or more additional classification functions different from the machine learning classification function.

-42-

10. A computer-implemented data processing method comprising:
 - using a server computer executing an application layer and a persistent data storage layer, importing, through the application layer, and storing in non-volatile memory, business logic that is programmed to one or more business data transformation operations;
 - storing, in at least one of main memory of the server computer or, through the persistent data storage layer, a back-end database, one or more Tiles respectively corresponding to one or more data sources, each Tile being associated with a plurality of metadata including at least permissions metadata, usage metadata, and version metadata.
 - storing, in main memory, one or more Edges respectively corresponding to one or more data transformation operations, each Edge being associated with a plurality of metadata including at least permissions metadata, usage metadata, and version metadata;
 - receiving graphical input that graphically arranges, in a graphical user interface, visual representations in the graphical user interface of the one or more Tiles and the one or more Edges into a Mosaic, the Mosaic forming a directed acyclic graph, and storing the Mosaic in non-volatile memory in association with a project name; and
 - programmatically traversing the one or more Tiles and the one or more Edges comprising the Mosaic, including executing the one or more data transformation operations, to produce a final data set, and storing the final data set in non-volatile memory associated with the project name;
 - at least one of the Edges that is stored in main memory being an Edge Instance of an imported business data transformation operation.
11. The method of Claim 10, further comprising importing, through the application layer, the business logic by executing:
 - serializing the business logic and any in-memory dependencies to text;
 - programmatically calling one or more network calls to send the serialized business logic into the application layer; and
 - storing the business logic and any associated metadata in a repository configured as a functioning programming library with install requirements and versioning.

-43-

12. The method of Claim 11, the Edge Instance being associated with edge type metadata partly defined by a number of inputs accepted and a number of outputs produced by the Edge Instance.

13. The method of Claim 10, at least one Edge of the Mosaic comprising a machine learning classification function; and

the method further comprising traversing the one or more tiles and the one or more Edges comprising the Mosaic to produce the final data set by dividing model input data into one or more training segments and one or more test segments for executing the machine learning classification function using random sampling, rolling validation, or walk forward validation.

14. The method of Claim 13, further comprising causing to display, in the graphical user interface, at least one of a validation curve, a learning curve, a calibration curve, or a precision and recall curve related to execution of the machine learning classification function, wherein the machine learning classification function comprises a random forest classifier.

15. The method of Claim 10, further comprising generating Python program code by:
selecting one of a user-defined Edge comprising a user-defined data transformation operation or a user-defined Edge Queue comprising a plurality of user-defined data transformation operations;

extracting user-defined logic, in the form of Python code, from the selected user-defined Edge or user-defined Edge Queue; and

packaging and storing, in memory, the user-defined logic extracted in the form of Python code in a portable and installable Python library.

16. A computer system, comprising:

one or more processors;

a memory comprising a set of instructions which when executed causes the one or more processors to execute a method, the method comprising:

-44-

using a server computer executing an application layer and a persistent data storage layer, importing, through the application layer, and storing in non-volatile memory, business logic that is programmed to one or more business data transformation operations;

storing, in at least one of main memory of the server computer or, through the persistent data storage layer, a back-end database, one or more Tiles respectively corresponding to one or more data sources, each Tile being associated with a plurality of metadata including at least permissions metadata, usage metadata, and version metadata.

storing, in main memory, one or more Edges respectively corresponding to one or more data transformation operations, each Edge being associated with a plurality of metadata including at least permissions metadata, usage metadata, and version metadata;

receiving graphical input that graphically arranges, in a graphical user interface, visual representations in the graphical user interface of the one or more Tiles and the one or more Edges into a Mosaic, the Mosaic forming a directed acyclic graph, and storing the Mosaic in non-volatile memory in association with a project name; and

programmatically traversing the one or more Tiles and the one or more Edges comprising the Mosaic, including executing the one or more data transformation operations, to produce a final data set, and storing the final data set in non-volatile memory associated with the project name;

at least one of the Edges that is stored in main memory being an Edge Instance of an imported business data transformation operation.

17. The computer system of Claim 16, the method further comprising importing, through the application layer, the business logic by executing:

serializing the business logic and any in-memory dependencies to text;

programmatically calling one or more network calls to send the serialized business logic into the application layer; and

storing the business logic and any associated metadata in a repository configured as a functioning programming library with install requirements and versioning.

18. The computer system of Claim 16, at least one Edge of the Mosaic comprising a machine learning classification function; and

-45-

the method further comprising traversing the one or more tiles and the one or more Edges comprising the Mosaic to produce the final data set by dividing model input data into one or more training segments and one or more test segments for executing the machine learning classification function using random sampling, rolling validation, or walk forward validation.

19. The computer system of Claim 18, the method further comprising causing to display, in the graphical user interface, at least one of a validation curve, a learning curve, a calibration curve, or a precision and recall curve related to execution of the machine learning classification function, wherein the machine learning classification function comprises a random forest classifier.

20. The computer system of Claim 16, the method further comprising generating program code in Python by:

- selecting one of a user-defined Edge comprising a user-defined data transformation operation or a user-defined Edge Queue comprising a plurality of user-defined data transformation operations;

- extracting user-defined logic, in the form of Python code, from the selected user-defined Edge or user-defined Edge Queue; and

- packaging and storing, in memory, the user-defined logic extracted in the form of Python code in a portable and installable Python library.