(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
03 March 2022 (03.03.2022)

WIPO | PCT

(10) International Publication Number
WO 2022/046759 A1

(51) International Patent Classification:
G06N 5/04 (2006.01) *G06N 20/00* (2019.01)

(21) International Application Number:
PCT/US2021/047334

(22) International Filing Date:
24 August 2021 (24.08.2021)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
17/002,717 25 August 2020 (25.08.2020) US

(71) Applicant: **ALTERYX, INC.** [US/US]; 3345 Michelson Drive, Suite 400, Irvine, CA 92612 (US).

(72) Inventors: **BLANCHARD, Dylan**; ALTERYX, INC., 3345 Michelson Drive, Suite 400, Irvine, CA 92612 (US). **HEINL, Tyler**; ALTERYX, INC., 3345 Michelson Drive, Suite 400, Irvine, CA 92612 (US). **HOCHMUTH, Roland Manfred**; ALTERYX, INC., 3345 Michelson Drive, Suite 400, Irvine, CA 92612 (US).

(74) Agent: **HOFFMAN, Brian M.** et al.; Fenwick & West LLP, Silicon Valley Center, 801 California Street, Mountain View, CA 94041 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report (Art. 21(3))*

(54) Title: HYBRID MACHINE LEARNING

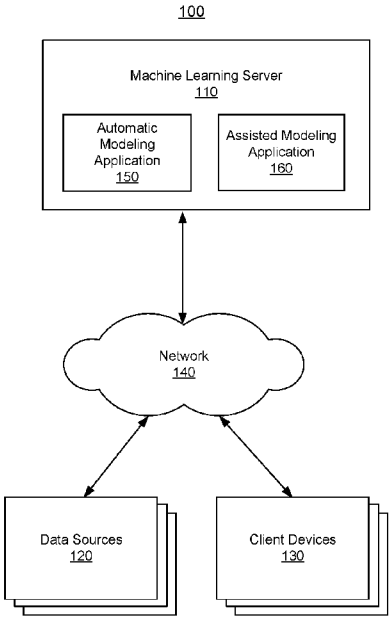


FIG. 1

(57) Abstract: A model is trained through a hybrid machine learning process. In the hybrid machine landing process, an automatic machine learning process is performed on a dataset to generate a model for making a prediction. The automatic machine learning process uses a pipeline to train the model and makes decisions in the steps of the pipeline. After the model is trained through the automatic machine learning process, a representation of the pipeline is generated and presented to a user in a user interface. The user interface allows the user to modify at least some decision made in the automatic machine learning process. One or more modifications are received from the user through the user interface and are used to refine the trained model. The refined model is deployed to make the prediction based on new data.

WO 2022/046759 A1

HYBRID MACHINE LEARNING

FIELD OF ART

[0001] The described embodiments pertain in general to processing data streams, and in particular to using a hybrid machine learning technique to train models for making predictions based on the data streams.

DESCRIPTION OF THE RELATED ART

[0002] Automatic machine learning tools automate the process of applying machine learning to real-world problems. Currently, automatic machine learning tools cover the complete pipeline from receiving the raw dataset to creating the deployable machine learning model. These tools advantageously produce simple solutions and allow for fast and efficient creation of those solutions. However, automatic machine learning tools often produce solutions that are sub-optimal because they do not incorporate domain knowledge pertaining to the dataset. The tools provide little ability for data analysts who have understanding of the dataset to control or otherwise modify the machine learning process. Consequently, the models generated by currently available automatic machine learning tools are not as good as they could be at making predictions based on the data.

SUMMARY

[0003] The above and other issues are addressed by a method, computer-implemented data analytics system, and computer-readable memory for processing data blocks in a data analytics system. An embodiment of the method includes receiving a dataset. The method further includes performing an automatic machine learning process on the received dataset to generate a model for making a prediction based on new data. The automatic machine learning process comprise making a sequence of decisions based on a machine learning pipeline. The method further includes after the trained model is generated, generating a representation of the machine learning pipeline. The method further includes providing the representation of the machine learning pipeline for display in a user interface. The user interface allows a user to modify at least some of the decisions made in the automatic machine learning process. The method further includes receiving one or more modifications from the user through the user interface.

The method also includes refining the model based on the one or more modifications from the user. The refined model is to be used to make the prediction based on new data.

[0004] An embodiment of the computer-implemented data analytics system includes a computer processor for executing computer program instructions. The system also includes a non-transitory computer-readable memory storing computer program instructions executable by the computer processor to perform operations. The operations include receiving a dataset. The operations further include performing an automatic machine learning process on the received dataset to generate a model for making a prediction based on new data. The automatic machine learning process comprise making a sequence of decisions based on a machine learning pipeline. The operations further include after the trained model is generated, generating a representation of the machine learning pipeline. The operations further include providing the representation of the machine learning pipeline for display in a user interface. The user interface allows a user to modify at least some of the decisions made in the automatic machine learning process. The operations further include receiving one or more modifications from the user through the user interface. The operations also include refining the model based on the one or more modifications from the user. The refined model is to be used to make the prediction based on new data.

[0005] An embodiment of the non-transitory computer-readable memory stores executable computer program instructions. The instructions are executable to perform operations. The operations include receiving a dataset. The operations further include performing an automatic machine learning process on the received dataset to generate a model for making a prediction based on new data. The automatic machine learning process comprise making a sequence of decisions based on a machine learning pipeline. The operations further include after the trained model is generated, generating a representation of the machine learning pipeline. The operations further include providing the representation of the machine learning pipeline for display in a user interface. The user interface allows a user to modify at least some of the decisions made in the automatic machine learning process. The operations further include receiving one or more modifications from the user through the user interface. The operations also include refining the model based on the one or more modifications from the user. The refined model is to be used to make the prediction based on new data.

BRIEF DESCRIPTION OF DRAWINGS

[0006] FIG. 1 is a block diagram illustrating a machine learning environment including a machine learning server according to one embodiment.

[0007] FIG. 2 is a block diagram illustrating an automatic modeling application according to one embodiment.

[0008] FIG. 3 is a block diagram illustrating an assisted modeling application according to one embodiment.

[0009] FIG. 4 is an example user interface for hybrid machine learning according to one embodiment.

[0010] FIG. 5 is a flow chart illustrating a hybrid machine learning process according to one embodiment.

[0011] FIG. 6 is a high-level block diagram illustrating a functional view of a typical computer system for use as the machine learning server of FIG. 1 according to an embodiment.

[0012] The figures depict various embodiments for purposes of illustration only. One skilled in the art will readily recognize from the following discussion that alternative embodiments of the structures and methods illustrated herein may be employed without departing from the principles of the embodiments described herein. Like reference numbers and designations in the various drawings indicate like elements.

DETAILED DESCRIPTION

[0013] FIG. 1 is a block diagram illustrating a machine learning environment 100 including a machine learning server 110 according to one embodiment. The environment 100 further includes multiple data sources 120 and client devices 130 connected to the machine learning server 110 via a network 140. Although the illustrated environment 100 contains only one machine learning server 110 coupled to multiple data sources 120 and client devices 130, embodiments can have multiple machine learning servers, a singular data source, and a singular client device, or other variations thereof.

[0014] The machine learning server 110 is a computer-based system utilized for constructing machine learning models and deploying the models to make predictions based on data. Example predictions include whether or not a customer will make a transaction within a time period, whether or not a transaction is fraudulent, whether or not a user will perform a computer-based interaction, etc. The data are collected, gathered, or otherwise accessed from one or more of the multiple data sources 120 or one or more of the multiple client devices 130 via the network 140. The machine learning server 110 can implement scalable software tools and hardware resources employed in accessing, preparing, blending, and analyzing data from a wide variety of data sources 120 or client devices 130.

[0015] In some embodiments, the machine learning server 110 is a computer system that

implements hybrid machine learning. The machine learning server 110 includes an automatic modeling application 150 and an assisted modeling application 160. The automatic modeling application 150 performs an automatic machine learning process on a dataset to train a model. In contrast, the assisted modeling application 160 performs an assisted machine learning process using both a dataset and user input to train a model. The user input specifies how the assisted modeling application 160 processes the dataset to train the model. For example, the user input may select a data type, data imputation method, feature, or algorithm, tune a hyperparameter, provide other guidance to the machine learning process, or some combination thereof.

[0016] The two applications may run separately or together to train models. When the two applications run together, they train a model through a hybrid machine learning process in which the assisted modeling application 160 is used to constrain or otherwise refine the model generated by the automatic modeling application 150.

[0017] In embodiments where the hybrid machine learning process starts with the automatic machine learning process, the automatic modeling application 150 follows a machine learning pipeline (also referred to as "pipeline") to perform the automatic machine learning process and generate a model. The pipeline is a workflow of the automatic machine learning process and specifies a sequence of steps to train the model. In one example, the steps in the pipeline include data preparation, feature engineering, model training, model validation, and model deployment. A step may include sub-steps. For instance, the step of data preparation may include data type setting, data encoding, and data imputation, the step of feature engineering may include feature selection and feature ranking, and the step of model training may include hyperparameters tuning and algorithm selection. In some embodiments, the pipeline includes steps in a different order and/or more, fewer, or different steps than those described herein. The pipeline can either be created by a user or by the automatic modeling application 150.

[0018] The automatic modeling application 150 makes a sequence of decisions as it performs the steps of the pipeline. The automatic modeling application 150 may make one or more decisions in each step. In some embodiments, the automatic modeling application 150 iteratively processes some steps of the pipeline to optimize one or more decisions made in these steps. The automatic modeling application 150 may address the steps of the pipeline out of order. For instance, the automatic modeling application 150 may select features before it sets data types for the features. The automatic modeling application 150 may revisit a step after it addresses a subsequent step to optimize the initial decision made in the step. For

instance, the automatic modeling application 150 may first select a data type for a variable, then extract a feature from the variable, and then go back to change the data type for the variable.

[0019] After the trained model is generated, the hybrid machine learning process moves to the assisted machine learning process. The assisted modeling application 160 generates a representation of the machine learning pipeline used by the automatic modeling application 150, e.g., by extracting optimizations and decisions made by the automatic modeling application 150 from the automatic machine learning process. The representation of the machine learning pipeline may include representations of some or all steps in the pipeline. A representation of a step includes the decision made by the automatic modeling application 150 during the automatic machine learning process as well as alternative options for the step. The decision and alternative options are referred to together as "options" hereinafter. The representation of the step may include other information, such as a recommendation score for each option that indicates a level of recommendation for the option, a description of the option that can help the user to make a modification, and so on. In some embodiments, the recommendation score is determined by the automatic modeling application 150 in the automatic machine learning process, and the assisted modeling application 160 derives the recommendation score from the automatic machine learning process. In some embodiments, the assisted modeling application 160 determines the recommendation score based on the automatic machine learning process.

[0020] The assisted modeling application 160 provides the representation of the pipeline for display to the user, e.g., in a GUI. The user may review the steps in the pipeline and/or modify the decisions made in the automatic machine learning process. The assisted modeling application 160 receives modifications from the user and can refine the trained model based on the modifications. The refined trained model is then deployed to make predictions based on new data.

[0021] In some embodiments, the hybrid machine learning process starts with the assisted machine learning process. The assisted modeling application 160 receives user input and creates a set of pipelines based on the user input. For instance, the assisted modeling application 160 may present recommendations (e.g., recommendations of data type, transformer, feature, algorithm, and/or hyperparameter) in a user interface and the user provides the user input based on the recommendations through the user interface. The assisted modeling application 160 sends the set of pipelines to the automatic modeling application 150 and the automatic modeling application 150 performs the automatic machine

learning process based on the set of pipelines. Compared with a pure automatic machine learning process, the hybrid machine learning process consumes less time and/or computing resources as the automatic modeling application 150 can limit its search/optimization to the set of pipelines. Also, by using the user input to create the set of pipelines, it takes advantages of the user's domain knowledge that the automatic modeling application 150 usually does not have.

[0022] The data sources 120 provide electronic data to the machine learning server 110. A data source 120 may be a storage device such as a hard disk drive (HDD) or solid-state drive (SSD), a computer managing and providing access to multiple storage devices, a storage area network (SAN), a database, or a cloud storage system. A data source 120 may also be a computer system that can retrieve data from another source. The data sources 120 may be remote from the machine learning server 110 and provide the data via the network 140. In addition, some or all data sources 120 may be directly coupled to the data analytics system and provide the data without passing the data through the network 140.

[0023] The data provided by the data sources 120 may be organized into data records (e.g., rows). Each data record includes one or more values. For example, a data record provided by a data source 120 may include a series of comma-separated values. The data describe information of relevance to an enterprise using the data analytics system 110. For example, data from a data source 120 can describe computer-based interactions (e.g., click tracking data) with content accessible on websites and/or with applications. As another example, data from a data source 120 can describe customer transactions online and/or in stores. The enterprise can be in one or more of various industries, such as manufacturing, sales, financing, and banking.

[0024] The client devices 130 are one or more computing devices capable of receiving user input as well as transmitting and/or receiving data via the network 140. In one embodiment, a client device 130 is a conventional computer system, such as a desktop or a laptop computer. Alternatively, a client device 130 may be a device having computer functionality, such as a personal digital assistant (PDA), a mobile telephone, a smartphone, or another suitable device. The client devices 130 are configured to communicate with one or more data sources 120 and the machine learning server 110 via the network 140. In one embodiment, a client device 130 executes an application allowing a user of the client device 130 to interact with the machine learning server 110. For example, a client device 130 executes an application to enable interaction between the client device 130 and the machine learning server 110 via the network 140, e.g., through by running a GUI supported by the machine learning server 110. The client

device 130 includes or is otherwise associated with a display device that displays the GUI. The client device 130 is also associated with input devices, e.g., keyboard, mouse, etc., that allow the user to interact with the GUI, such as provide inputs to the GUI. In another embodiment, a client device 130 interacts with the machine learning server 110 through an application programming interface (API) running on a native operating system of the client device 130, such as IOS® or ANDROID™. The client device 130 may interact with one or more data sources 120 to transmit data to a data source 120 or obtain data from a data source 120.

[0025] The network 140 represents the communication pathways between the machine learning server 110 and data sources 120. In one embodiment, the network 140 is the Internet and uses standard communications technologies and/or protocols. Thus, the network 140 can include links using technologies such as Ethernet, 802.11, worldwide interoperability for microwave access (WiMAX), 3G, Long Term Evolution (LTE), digital subscriber line (DSL), asynchronous transfer mode (ATM), InfiniBand, PCI Express Advanced Switching, etc. Similarly, the networking protocols used on the network 140 can include multiprotocol label switching (MPLS), the transmission control protocol/Internet protocol (TCP/IP), the User Datagram Protocol (UDP), the hypertext transport protocol (HTTP), the simple mail transfer protocol (SMTP), the file transfer protocol (FTP), etc.

[0026] The data exchanged over the network 140 can be represented using technologies and/or formats including the hypertext markup language (HTML), the extensible markup language (XML), etc. In addition, all or some of links can be encrypted using conventional encryption technologies such as secure sockets layer (SSL), transport layer security (TLS), virtual private networks (VPNs), Internet Protocol security (IPsec), etc. In another embodiment, the entities can use custom and/or dedicated data communications technologies instead of, or in addition to, the ones described above.

[0027] FIG. 2 is a block diagram illustrating an automatic modeling application 200 according to one embodiment. The automatic modeling application 200 is an embodiment of the automatic modeling application 150 in FIG. 1. The automatic modeling application 200 uses a pipeline to perform an automatic machine learning process on a dataset to train a model. The trained model is to be used for predicting a target variable that may be defined by a user. The automatic modeling application 200 includes a data preparing module 210, a feature engineering module 220, a model training module 230, a model validating module 240, and a database 250. Those of skill in the art will recognize that other embodiments can have different and/or other components than the ones described here, and that the functionalities can

be distributed among the components in a different manner. The components of the automatic modeling application 200 operate together to perform the automatic machine learning process by making a sequence of decisions in the steps of the pipeline.

[0028] The data preparing module 210 processes data in the dataset to prepare a training dataset for training the model. The data preparing module 210 determines data types for variables associated within the dataset. A variable associated with the dataset can be a variable in the dataset or be transformed from one or more variables in the dataset. In some embodiments, the variable associated with the dataset is a predicting variable, i.e., a feature. In some embodiments, the data preparing module 210 selects the data type of a variable from a pool of optional data types, such as whether the variable is numerical data type, categorical data type, time series data type, zip code data type, or textual data type. In some embodiments, the data preparing module 210 determines the one or more data types for a variable through a rule-based analysis. The determination is based on rules maintained by the data preparing module 210. In one example, the rules specify which data type for a variable based on data values of the variable, e.g., variables including numerical values have the numerical data type and variable including textual values have the textual data type. If the data preparing module 210 determines that a variable is numeric, it may further determine whether the variable is integer or float under the rule. If it is integer, the data preparing module 210 may then determine the number of unique integer values of the variable. If the data preparing module 210 determines that the number of unique integer values of the variable is below a threshold, the rule indicates that the data type of the variable is categorical. The rule may also suggest that the datatype of the variable could be numerical. In another example, the rules specify which data type for a variable based on a description of the variable, e.g., a variable named “user ID” in the dataset has the ID data type and a variable named “date of birth” has the time series data type.

[0029] The data preparing module 210 may determine a ranking score for each optional data type of a variable. The ranking score of a data type indicates a probability of the optional data type being the true data type of the variable. In some embodiments, the rules maintained by the data preparing module 210 may indicate probabilities of different data types for a variable. In the example discussed above, if the data preparing module 210 determines that the number of unique integer values of the variable is below a threshold, the rule can indicate that the categorical data type has the higher probability than the numerical data type. The data preparing module 210 may also determine whether the integer values are zip codes and determine a probability for zip code based on the rule. In another example, the rule

indicates that for a variable named "zip code" in the dataset, the probabilities for categorical data type and numerical data type are higher than the probability for time series data type and textual data type. In yet another example, the rule indicates that for a variable including textual values, the probabilities for categorical data type and textual data type are higher than the probability for time series data type and numerical data type.

[0030] In some embodiments, the data preparing module 210 may determine the ranking score of each optional data type by evaluating performance of a model trained using the data type. For instance, the data preparing module 210 trains a model using an optional data types and determines the ranking scores of the optional data type based on the performance of the model. The data preparing module 210 may compare performances of models trained using different data types (other decisions in the machine learning processes of the models can be the same) and determine ranking scores of the data types based on the comparison. For instance, the data preparing module 210 determines that the ranking score of a data type is higher than another data type if the model trained using the data type shows better performance than the model trained using the other data type. The ranking scores/probabilities can be used to constrain further search done by the automatic modeling application 200 in the automatic machine learning process, such as search for features, algorithms, hyperparameters, etc.

[0031] The data preparing module 210 selects one of the four data types as the data type of the variable based on the ranking scores. For instance, the data preparing module 210 selects the data type that has the highest probability.

[0032] The data preparing module 210 may encode data in the dataset based on the determined data types. Values of some data types, e.g., categorical values, are encoded into different representations that are better suited to model training. In some embodiments, the data preparing module 210 encodes the values of categorical variables, time series variables, and/or textual variables to convert the values to binary values. The data preparing module 210 may encode the date using various methods, such as translating text to numerical values, label encoding, one hot encoding, custom binary encoding, backward difference encoding, polynomial encoding, and so on.

[0033] The data preparing module 210 detects missing values in the dataset and performs data imputation to supply the values. In some embodiments, the data preparing module 210 determines new values based on the present values in the dataset to replace the missing values. For instance, for each column that has missing values, the data preparing module 210 replaces the missing values in the column with the mean or median of the present values in the column, with the most frequent values in the column, or with values from new samples not in the

dataset. The data preparing module 210 may use other imputation methods, such as k-Nearest Neighbor (kNN) imputation, hot deck imputation, cold deck imputation, regression imputation, Stochastic regression imputation, extrapolation and interpolation, single imputation, multiple imputation, Multivariate Imputation by Chained Equation (MICE), imputation using Deep Neural Networks, and so on. The data preparing module 210 may identify multiple imputation methods and rank the identified imputation methods based on the data type of the corresponding variable. In one example, for a numeric variable, the data preparing module 210 determines if there are outliers in the dataset, and if there are, it ranks the imputation method of median higher than the imputation method of mean.

[0034] The feature engineering module 220 extracts features from the dataset. The feature engineering module 220 may extract a variable in the dataset as a feature and/or use a transformer to convert a variable in the dataset to a feature. When the transformer is applied to the values of the variable, it generates values of the feature. In some embodiments, the feature engineering module 220 selects the transformers from a pool of model training module 230 transformers based on one or more factors, such as variables in the dataset, the target variable, business problem related to the target variable, and so on.

[0035] The feature engineering module 220 ranks the features and determines a ranking score for each feature. The ranking score of a feature indicates how important the feature is for predicting the target variable, in other words, how good the feature is as a predictor. In some embodiments, the feature engineering module 220 constructs a random forest based on the features and the dataset. The feature engineering module 220 determines a ranking score of a feature based on each decision tree in the random forest and obtains an average of the individual ranking scores as the ranking score of the feature. The feature engineering module 220 may use GINI impurity as part of each decision tree to measure how much a feature contributes to the whole predictive model. The ranking score of a feature determined by using the random forest indicates how important the feature is relative to the other features and are referred to as "relative ranking score." In one example, the ranking module 330 determines that the relative ranking scores of the highest ranked selected feature is 1. The ranking module 330 then determines a ratio of the ranking score of each of the rest of the features to the ranking score of the highest ranked feature as the relative ranking scores of the corresponding selected feature.

[0036] The feature engineering module 220 may determine absolute ranking score for each selected feature, e.g., based on Goodman-Kruskal Tau (GKT) measure. GKT measure is a measure of association that is local or absolute and indicates how well a feature predicts a

target. The feature engineering module 220 may select a subset of the group of features based on their relative ranking scores and/or absolute ranking scores as features to train the model.

[0037] The model training module 230 trains the model based on the features and their ranking scores as determined by the feature engineering module 220. In some embodiments, the model training module 230 selects an algorithm (also referred to as an "estimator") from a pool of candidate algorithms. Examples of a candidate algorithms include, e.g., decision tree, logistic regression, random forest, XGBoost, linear support vector machine (linear SVM), AdaBoost, neural networks, naïve Bayes, memory-based learning, random forests, bagged trees, boosted trees, boosted stumps, and so on. The model training module 230 trains the model by using the selected algorithm. In some embodiments, the model training module 230 may constrain the number of candidate algorithms in the pool based on available information, e.g., time limit for training the model, computational resource limitations (e.g., processor limitations, memory usage limitations, etc.), the predictive problem to be solved, characteristics of the dataset, selected features, and so on. The model training module 230 may test each candidate algorithm and select the best one. The model training module 230 can define a testing harness associated with a performance measure (e.g., classification accuracy) to assess performance of the candidate algorithms. For example, the model training module 230 applies a model trained with a candidate algorithm to a validation dataset (a different dataset from the dataset used to train the model) to quantify the accuracy of the trained model. Common metrics applied in accuracy measurement include: Precision = TP / (TP + FP) and Recall = TP / (TP + FN), where precision is how many outcomes the model correctly predicted (TP or true positives) out of the total it predicted (TP + FP or false positives), and recall is how many outcomes the model correctly predicted (TP) out of the total number that actually occurred (TP + FN or false negatives). The F score (F-score = 2 * PR / (P + R)) unifies precision and recall into a single measure.

[0038] The outcome of testing the candidate algorithms against the testing harness estimates how the candidate algorithms perform on the predictive problem against the performance measures. The model training module 230 selects the candidate algorithm that has the best performance. In some embodiments, the model training module 230 further optimizes the selected candidate algorithm. The model training module 230 may analyze pros and cons of the candidate algorithms based on the testing outcome or other information of the candidate algorithms.

[0039] The model training module 230 may also determine hyperparameters required by the selected algorithm. A hyperparameter is a parameter whose value is used to control the

machine learning process. The hyperparameters are used in the training process and may influence the speed or quality of the training process. Examples of hyperparameters include learning rate, mini-batch size, C and sigma hyperparameters for support vector machines, size of a neural network, and the topology of a neural network. The model training module 230 can determine a hyperparameter by grid searching, random searching, or other methods. In some embodiments, the model training module 230 obtain the hyperparameters by deriving one or more hyperparameters from historical training processes, using default values specified for automatic machine learning, or other techniques. The database 240 stores data associated with the automatic modeling application 200, such as data received, used, and generated by the automatic modeling application 200. For instance, the database 240 stores the dataset, pipeline, decisions made in the steps of the pipeline, training data set, features, transformers, algorithms, hyperparameters, trained model, and so on.

[0040] FIG. 3 is a block diagram illustrating an assisted modeling application 300 according to one embodiment. The assisted modeling application 300 is an embodiment of the assisted modeling application 160 in FIG. 1. The assisted modeling application 300 refines a model trained by the automatic modeling application 200 based on user input. The assisted modeling application 300 includes a pipeline representation module 310, a user interface module 320, a model refining module 330, and a database 340. Those of skill in the art will recognize that other embodiments can have different and/or other components than the ones described here, and that the functionalities can be distributed among the components in a different manner.

[0041] The pipeline representation module 310 generates a representation of a machine learning pipeline. In some embodiments, the pipeline representation module 310 obtains the machine learning pipeline from an automatic modeling application, such as the automatic modeling application 150. For instance, the pipeline representation module 310 retrieves the machine learning pipeline from an automatic machine learning process performed by the automatic modeling application. In some embodiments, the pipeline representation module 310 retrieves data stored in the database 240 of the automatic modeling application 200, such as data used and generated in the automatic machine learning process. The pipeline representation module 310 analyzes the data and maps the data into the steps of the pipeline and options in each step. The pipeline representation module 310 generates the representation of the pipeline based on the analysis and mapping. In some embodiments, the pipeline representation module 310 queries the automatic modeling application 200 for more than one pipelines. For instance, the pipeline representation module 310 queries for a

predetermined number of pipelines identified by the automatic modeling application 200. In response to the query, the automatic modeling application 200 sends the predetermined number of best pipelines in its search to the pipeline representation module 310. The pipeline representation module 310 can select one of the pipelines received from the automatic modeling application 200 based on performance measurement of models trained using the pipelines, such as model accuracy. The pipeline representation module 310 then translates the selected pipeline and steps/components of the pipeline to generate the representation of the pipeline.

[0042] The pipeline representation module 310 generates a representation for one or more steps in the pipeline. The representation of a step (also referred to as "step representation") includes options for the step. One of the options is the decision made in the step during the automatic machine learning process. The representation may also include a ranking score or recommendation indicator for each option. The ranking score or recommendation indicator indicates the level of recommendation for the option. For some steps, the representation includes a description of each option. The description includes, e.g., explanation of the function of the option, assessment (e.g., pros and cons) of the option, evaluation of the option, description of reasons for choosing and/or not choosing the option, and so on. The ranking scores and/or descriptions can be either determined by the automatic model application 200 or by the pipeline representation module 310, e.g., by using the techniques described above in conjunction with FIG. 2. In some embodiments, the ranking score and/or description of an option is determined based on a performance measure (e.g., prediction accuracy) of a model trained using the option. For instance, the recommendation score of an option is higher when the performance of the model trained using the option is determined to be better.

[0043] In one example, the pipeline representation module 310 generates a data type list as a representation of the data type setting step. The data type list is associated with one or more variables, e.g., features used in the automatic machine learning process. It includes a plurality of optional data types for each feature. The data type list also includes a ranking score for each optional data type. The ranking score indicates a probability that the optional data type is the true data type of the feature.

[0044] In another example, the pipeline representation module 310 generates a data imputation list as a representation of the data imputation step. The data imputation list includes one or more features having missing values that were imputed in the automatic machine learning process. For each feature, the data imputation list includes a number of optional imputation methods, which includes the imputation method used in the automatic

machine learning process to impute the value of the feature. The data imputation list may include a percentage for each feature indicating the percentage of rows with missing values in the total rows of the feature. The pipeline representation module 310 may calculate replacing values based on each optional imputation method and include replacing values in the representation of the data imputation step.

[0045] In yet another example, the pipeline representation module 310 generates a feature list as a representation of the feature engineering step. The feature list includes some or all of the features used in the automatic machine learning process and may also include features that were not used in the automatic machine learning process. The feature list may include a ranking score of each feature that indicates how important the feature is for the prediction, a description of each feature, an evaluation of the feature, or some combination thereof. The feature list may also include transformers used for extracting features.

[0046] In yet another example, the pipeline representation module 310 generates an algorithm list as a representation of the algorithm selecting step. The algorithm list includes optional algorithms used for training models. One of the optional algorithms is the algorithm selected and used in the automatic machine learning process and the pipeline representation module 310 may label it as "recommended" algorithm in the algorithm list. For each optional algorithm, the algorithm list may include a description showing pros and cons of the optional algorithm. The pros and cons can be specific to the model to be refined. The pipeline representation module 310 may also determine a ranking score for each optional algorithm, e.g., based on the testing outcome of the performance of the optional algorithm in the automatic machine learning process. In one example, the pipeline representation module 310 determines the ranking score based on the measured accuracy of a model trained using the optional algorithm.

[0047] The user interface module 320 supports a GUI and provides the representation of the pipeline for presentation through the GUI. The GUI allows the user to view the representation of the pipeline, review the decisions made in the automatic machine learning process based on the pipeline, and make modifications to at least some of the decisions. In one example, the GUI allows the user to change the data type of a variable used in the automatic machine learning process to a different data type. In another example, the GUI allows the user to edit features, such as change transformers, remove a feature that was selected in the automatic machine learning process, add a feature not selected in the automatic machine learning process, or change the ranking of a feature that was determined in the automatic machine learning process. For instance, the user can use his/her domain

knowledge to remove a feature that results in target leakage but was selected in the automatic machine learning process due to the lack of the domain knowledge of the automatic modeling application 200. The GUI may also allow the user to change algorithm and/or hyperparameter.

[0048] In some embodiments, the GUI includes a plurality of control elements with which the user can interact to select and modify a step of the pipeline. Examples of the control elements include checkboxes, buttons, tabs, icons, dropdown lists, list boxes, radio buttons, toggles, text fields, date field. For instance, the GUI includes a tab for each step and the user can click the tab to access the decision and alternative options for the step. The options in each step may be presented in a dropdown list or checkboxes that allow the user to select one or more options at a time. The decision for the corresponding step may be distinguished from the alternative options, e.g., the decision may be marked as "recommended" in the GUI or may be highlighted. The GUI allows a user to either keep the decision (e.g., by doing nothing or by clicking an icon representing the decision) or choose an alternative option (e.g., by clicking an icon representing the alternative option).

[0049] In some embodiments, the GUI allows the user to navigate the steps in the pipeline out of order. For instance, the user may review the selecting feature step before the user reviews the setting data type step. The GUI may allow the user to revisit a step and/or make multiple modifications to a step.

[0050] The GUI may allow other types of user input. For instance, the GUI allows the user to modify the target variable, edit the dataset, select automation level for refining the model, and so on.

[0051] The model refining module 330 refines the trained model based on modifications received from the user through the GUI. In some embodiments, the model refining module 330 sends the modifications to the automatic modeling application 200 and the automatic modeling application 200 tunes part or all of the automatic machine learning process with the modifications. The automatic modeling application 200 may update its decisions in the pipeline based on the modifications and trains a new model based on the updated decisions. For instance, the automatic modeling application 200 uses the user's modification to constrain its search for candidate algorithms or hyperparameters. In one example, the GUI receives the user's specification of one or more hyperparameters to be used to refine the model, the model refining module 330 sends the one or more hyperparameters to the automatic modeling application 200 and the automatic modeling application 200 uses the one or more hyperparameters to search for new pipeline and algorithm. The automatic modeling

application 200 then uses the new pipeline and algorithm to re-train the model.

[0052] In some embodiments, the model refining module 330 re-trains the model based on the user's modifications, e.g., by using the machine learning techniques described above in conjunction with FIG. 2. The model refining module 330 may perform part or all of the pipeline to refine the trained model, which involves in making similar decisions as the automatic machine learning process or making new decisions. For instance, after receiving a user's modification of a data type of a feature from numerical to categorical, the model refining module 330 makes a new decision by selecting a data encoding method to convert the categorical values of the feature to numerical features. The model refining module 330 may also validate the refined model and deploy the refined model for making predictions based on new data after the validation.

[0053] The database 340 stores data associated with the assisted modeling application 300, such as data received, used, and generated by the assisted modeling application 300. For instance, the database 240 stores the representation of the pipeline, data from the automatic modeling application 200, data for the GUI, user input, data associated with refining the model, and so on.

[0054] FIG. 4 is an example user interface 400 for hybrid machine learning according to one embodiment. The user interface 400 is an embodiment of the GUI generated by the assisted modeling application 300 as described above. The user interface 400 presents a representation of a pipeline. It includes representations of four steps in the pipeline and four tabs 410A-D (collectively referred to as "tabs 410"). The user interface 400 allows a user to click one of the four tabs 410 to choose and navigate the corresponding step. As illustrated in FIG. 4, step 1 is chosen by the user. Step 1 is about setting data type. In response to receiving the user's click of the tab 410A for step 1, the user interface presents a data type list 420 to the user. The data type list 420 includes a list of features 430, optional data types 440 for the features, and details 450. The user interface has a tab 435 for each feature so that it allows the user to choose a feature by clicking the tab. The user interface also has a triangle button 445 and a drop list 447 for the optional data types of each feature. The default data type presented in the drop list 447 is the data type used in the automatic machine learning process and is marked as "recommended." In response to the user clicking the triangle button 445, the user interface presents other optional data types in the drop list. As illustrated in FIG. 3, the user chooses the zip code feature and the recommendation data type is numeric. After the user clicks the triangle button 445, the user interface presents three other data types to the user and allows the user to select one of the three to replace the recommended data type.

In FIG. 4, the recommended data type for the zip code feature was mistakenly identified in the automatic machine learning process, but the user has a chance to review and correct the mistake. The user can change the data type of the zip code feature to zip code by choosing zip code in the drop list 447.

[0055] The user interface also presents further details 450 of the data types. The details include probabilities, each of which corresponds to an optional data type and indicates how likely the data type of the feature is the corresponding optional data type.

[0056] Even though not shown in FIG. 4, the user interface 400 presents different information as the user switches to a different step or make a different input to the user interface. For instance, after receiving the user's click of tab 401C, the user interface presents optional features and/or transformers to the user. As another example, after receiving the user's selection of an imputation method, the user interface presents replacing values calculated by using the imputation method to the user and the replacing values change as the user selects a different imputation method. Also, the user interface 400 may present representations of other pipeline steps.

[0057] FIG. 5 is a flow chart illustrating a hybrid machine learning model process 500 according to one embodiment. In some embodiments, the method is performed by the machine learning server 110, although some or all of the operations in the method may be performed by other entities in other embodiments. In some embodiments, the operations in the flow chart are performed in different orders and include different and/or additional steps.

[0058] The machine learning server 110 receives 510 a dataset. In some embodiments, the machine learning server 110 receives the dataset from a data source (e.g., a data source 120) that is associated with an enterprise. The enterprise can be in one or more of various industries, such as manufacturing, sales, financing, and banking. In some embodiments, the machine learning server 110 receives the dataset from a client device, such as a client device 130.

[0059] The machine learning server 110 performs 520 an automatic machine learning process on the received dataset to train a model for making a prediction based on new data. The automatic machine learning process includes making a sequence of decisions based on a pipeline. In some embodiments, the pipeline includes a sequence of steps, such as data type setting, data encoding, data imputation, feature selection, feature ranking, algorithm selection, hyperparameters tuning, model training, and model validation. In some other embodiments, the pipeline may include fewer, more, or different steps. Each of the sequence of decisions is made in a step in the pipeline.

[0060] After the trained model is generated, the machine learning server 110 generates 530 a representation of the pipeline. The machine learning server 110 may identify some or all of the steps in the pipeline and generates a representation of each identified step. The representation of a step includes a plurality of options for the step that include the decision made in the step during the automatic machine learning process. The representation of the step may also include a ranking score for each of the plurality of options that indicates a level of recommendation for the option or other information of each option, such as assessment, description, and so on.

[0061] In some embodiments, the machine learning server 110 generates a data type list that includes a feature used in the automatic machine learning process to train the model, a plurality of optional data types for the features, and a ranking score for each optional data type that indicates a probability that the data type being a true data type of the feature. The plurality of optional data types includes a first data type selected for the feature in the automatic machine learning process to train the model. The machine learning server 110 may generate a feature list that includes a plurality of features and a description of each feature indicating how important the user is for the prediction. The plurality of features include features used in the automatic machine learning process to train the model. The machine learning server 110 may generate an algorithm list that includes a plurality of algorithm and description of each algorithm that indicates a reason for choosing or not choosing the algorithm for training the model. The algorithms including an algorithm used in the automatic machine learning process to train the model.

[0062] The machine learning server 110 provides 540 the representation of the machine learning pipeline for display in a user interface. The user interface allows a user to modify at least some of the decisions made in the automatic machine learning process. For instance, the user interface allows the user to select a new data type for a feature, select a new data imputing method to impute missing values of a feature, edit (e.g., adding, removing, or changing ranking) feature selection made in the automatic machine learning process, select a different algorithm, tune hyperparameters, and so on.

[0063] The machine learning server 110 receives 550 one or more modifications from the user through the user interface. The machine learning server 110 refines 560 the model based on the one or more modifications from the user. The refined model is to be used to make the prediction based on new data. For instance, the machine learning server 110 receives a selection of a different data type for a feature and may encode values in the feature based on the different data type.

[0064] FIG. 6 is a high-level block diagram illustrating a functional view of a typical computer system 600 for use as the machine learning server 110 of FIG. 1 according to an embodiment.

[0065] The illustrated computer system includes at least one processor 602 coupled to a chipset 604. The processor 602 can include multiple processor cores on the same die. The chipset 604 includes a memory controller hub 620 and an input/output (I/O) controller hub 622. A memory 606 and a graphics adapter 612 are coupled to the memory controller hub 620 and a display 618 is coupled to the graphics adapter 612. A storage device 608, keyboard 610, pointing device 614, and network adapter 616 may be coupled to the I/O controller hub 622. In some other embodiments, the computer system 600 may have additional, fewer, or different components and the components may be coupled differently. For example, embodiments of the computer system 600 may lack displays and/or keyboards. In addition, the computer system 600 may be instantiated as a rack-mounted blade server or as a cloud server instance in some embodiments.

[0066] The memory 606 holds instructions and data used by the processor 602. In some embodiments, the memory 606 is a random-access memory. The storage device 608 is a non-transitory computer-readable storage medium. The storage device 608 can be a HDD, SSD, or other types of non-transitory computer-readable storage medium. Data processed and analyzed by the machine learning server 110 can be stored in the memory 606 and/or the storage device 608.

[0067] The pointing device 614 may be a mouse, track ball, or other type of pointing device, and is used in combination with the keyboard 610 to input data into the computer system 600. The graphics adapter 612 displays images and other information on the display 618. In some embodiments, the display 618 includes a touch screen capability for receiving user input and selections. The network adapter 616 couples the computer system 600 to the network 160.

[0068] The computer system 600 is adapted to execute computer modules for providing the functionality described herein. As used herein, the term “module” refers to computer program instruction and other logic for providing a specified functionality. A module can be implemented in hardware, firmware, and/or software. A module can include one or more processes, and/or be provided by only part of a process. A module is typically stored on the storage device 608, loaded into the memory 606, and executed by the processor 602.

[0069] The particular naming of the components, capitalization of terms, the attributes, data structures, or any other programming or structural aspect is not mandatory or significant,

and the mechanisms that implement the embodiments described may have different names, formats, or protocols. Further, the systems may be implemented via a combination of hardware and software, as described, or entirely in hardware elements. Also, the particular division of functionality between the various system components described herein is merely exemplary, and not mandatory; functions performed by a single system component may instead be performed by multiple components, and functions performed by multiple components may instead performed by a single component.

[0070] Some portions of above description present features in terms of algorithms and symbolic representations of operations on information. These algorithmic descriptions and representations are the means used by those skilled in the data processing arts to most effectively convey the substance of their work to others skilled in the art. These operations, while described functionally or logically, are understood to be implemented by computer programs. Furthermore, it has also proven convenient at times, to refer to these arrangements of operations as modules or by functional names, without loss of generality.

[0071] Unless specifically stated otherwise as apparent from the above discussion, it is appreciated that throughout the description, discussions utilizing terms such as “processing” or “computing” or “calculating” or “determining” or “displaying” or the like, refer to the action and processes of a computer system, or similar electronic computing device, that manipulates and transforms data represented as physical (electronic) quantities within the computer system memories or registers or other such information storage, transmission or display devices.

[0072] Certain embodiments described herein include process steps and instructions described in the form of an algorithm. It should be noted that the process steps and instructions of the embodiments could be embodied in software, firmware or hardware, and when embodied in software, could be downloaded to reside on and be operated from different platforms used by real time network operating systems.

[0073] Finally, it should be noted that the language used in the specification has been principally selected for readability and instructional purposes, and may not have been selected to delineate or circumscribe the inventive subject matter. Accordingly, the disclosure of the embodiments is intended to be illustrative, but not limiting.

We claim:

1. A computer implemented method, comprising:

receiving a dataset;

performing an automatic machine learning process on the received dataset to train a model for making a prediction based on new data, the automatic machine learning process comprising making a sequence of decisions based on a pipeline;

after the model is trained, generating a representation of the pipeline;

providing the representation of the pipeline for display in a user interface, the user interface allowing a user to modify at least some of the decisions made in the automatic machine learning process;

receiving one or more modifications from the user through the user interface; and

refining the model based on the one or more modifications from the user, the refined model to be used to make the prediction based on new data.

2. The method of claim 1, wherein each decision is made in a step of a sequence of steps in the pipeline during the automatic machine learning process and generating the representation of the pipeline comprises:

generating a representation for each of one or more steps of the sequence of steps, the representation of a step comprising a plurality of options for the step, and the plurality of options including the decision made in the step during the automatic machine learning process.

3. The method of claim 2, wherein the representation of the step further comprises a ranking score for each of the plurality of options, the ranking score of an option indicating a level of recommendation for the option.

4. The method of claim 1, wherein generating the representation of the pipeline comprises:

generating a data type list comprising a feature used in the automatic machine learning process to train the model, a plurality of optional data types for the features, and a ranking score for each optional data type, the plurality of optional data types including a first data type selected for the feature in the automatic machine learning process to train the model, the ranking score of an optional data type indicating a probability that the data type being a true data type of the feature.

5. The method of claim 4, wherein receiving one or more modifications from the user through the user interface comprises:

receiving a selection of a second data type for the feature from the plurality of optional data types from the user through the user interface, the second data type being a different data type from the first data type; and

encoding values in the feature based on the second data type.

6. The method of claim 1, wherein generating the representation of the pipeline comprises:

generating a feature list, the feature list comprising a plurality of features and a description of each feature indicating how important the feature is for the prediction, the plurality of features including features used in the automatic machine learning process to train the model.

7. The method of claim 1, wherein generating the representation of the pipeline comprises:

generating an algorithm list, the algorithm list comprising a plurality of algorithms and description of each algorithm that indicates a reason for choosing or not choosing the algorithm for training the model, the plurality of algorithm including an algorithm used in the automatic machine learning process to train the model.

8. A non-transitory computer-readable memory storing executable computer program instructions for processing data blocks in a data analytics system, the instructions executable to perform operations comprising:

receiving a dataset;

performing an automatic machine learning process on the received dataset to train a model for making a prediction based on new data, the automatic machine learning process comprising making a sequence of decisions based on a pipeline;

after the model is trained, generating a representation of the pipeline;

providing the representation of the pipeline for display in a user interface, the user interface allowing a user to modify at least some of the decisions made in the automatic machine learning process;

receiving one or more modifications from the user through the user interface; and

refining the model based on the one or more modifications from the user, the refined model to be used to make the prediction based on new data.

9. The non-transitory computer-readable memory of claim 8, wherein each decision is made in a step of a sequence of steps in the pipeline during the automatic machine learning process and generating the representation of the pipeline comprises:

generating a representation for each of one or more steps of the sequence of steps, the representation of a step comprising a plurality of options for the step, and the plurality of options including the decision made in the step during the automatic machine learning process.

10. The non-transitory computer-readable memory of claim 9, wherein the representation of the step further comprises a ranking score for each of the plurality of options, the ranking score of an option indicating a level of recommendation for the option.

11. The non-transitory computer-readable memory of claim 8, wherein generating the representation of the pipeline comprises:

generating a data type list comprising a feature used in the automatic machine learning process to train the model, a plurality of optional data types for the features, and a ranking score for each optional data type, the plurality of optional data types including a first data type selected for the feature in the automatic machine learning process to train the model, the ranking score of an optional data type indicating a probability that the data type being a true data type of the feature.

12. The non-transitory computer-readable memory of claim 11, wherein receiving one or more modifications from the user through the user interface comprises:

receiving a selection of a second data type for the feature from the plurality of optional data types from the user through the user interface, the second data type being a different data type from the first data type; and

encoding values in the feature based on the second data type.

13. The non-transitory computer-readable memory of claim 8, wherein generating the representation of the pipeline comprises:

generating a feature list, the feature list comprising a plurality of features and a description of each feature indicating how important the user is for the prediction, the plurality of features including features used in the automatic machine learning process to train the model.

14. The non-transitory computer-readable memory of claim 8, wherein generating the representation of the pipeline comprises:

generating an algorithm list, the algorithm list comprising a plurality of algorithms and description of each algorithm that indicates a reason for choosing or not choosing the algorithm for training the model, the plurality of algorithm including an algorithm used in the automatic machine learning process to train the model.

15. A system, comprising:

a computer processor for executing computer program instructions; and

a non-transitory computer-readable memory storing computer program instructions executable by the computer processor to perform operations comprising:

receiving a dataset;

performing an automatic machine learning process on the received dataset to train a model for making a prediction based on new data, the automatic machine learning process comprising making a sequence of decisions based on a pipeline;

after the model is trained, generating a representation of the pipeline;

providing the representation of the pipeline for display in a user interface, the user interface allowing a user to modify at least some of the decisions made in the automatic machine learning process;

receiving one or more modifications from the user through the user interface; and

refining the model based on the one or more modifications from the user, the refined model to be used to make the prediction based on new data.

16. The system of claim 15, wherein each decision is made in a step of a sequence of steps in the pipeline during the automatic machine learning process and generating the representation of the pipeline comprises:

generating a representation for each of one or more steps of the sequence of steps, the representation of a step comprising a plurality of options for the step, and the plurality of options including the decision made in the step during the automatic machine learning process.

17. The system of claim 16, wherein the representation of the step further comprises a ranking score for each of the plurality of options, the ranking score of an option indicating a level of recommendation for the option.

18. The system of claim 15, wherein generating the representation of the pipeline comprises:

generating a data type list comprising a feature used in the automatic machine learning process to train the model, a plurality of optional data types for the features, and a ranking score for each optional data type, the plurality of optional data types including a first data type selected for the feature in the automatic machine learning process to train the model, the ranking score of an optional data type indicating a probability that the data type being a true data type of the feature.

19. The system of claim 15, wherein generating the representation of the pipeline comprises:

generating a feature list, the feature list comprising a plurality of features and a description of each feature indicating how important the user is for the prediction, the plurality of features including features used in the automatic machine learning process to train the model.

20. The system of claim 15, wherein generating the representation of the pipeline comprises:

generating an algorithm list, the algorithm list comprising a plurality of algorithms and description of each algorithm that indicates a reason for choosing or not choosing the algorithm for training the model, the plurality of algorithm including an algorithm used in the automatic machine learning process to train the model.

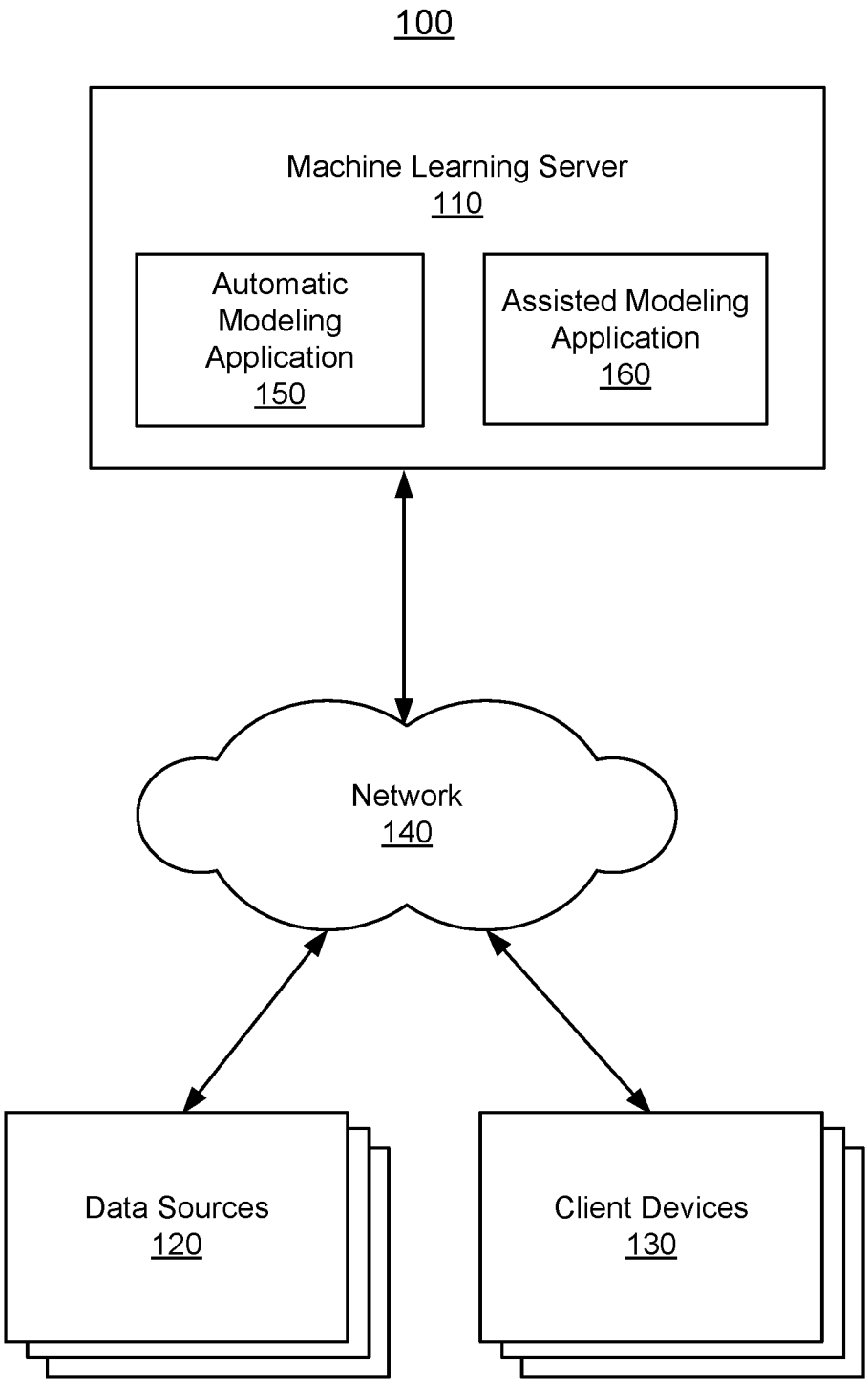

100
Machine Learning Server
110
Automatic Modeling Application
150
Assisted Modeling Application
160
Network
140
Data Sources
120
Client Devices
130

FIG. 1

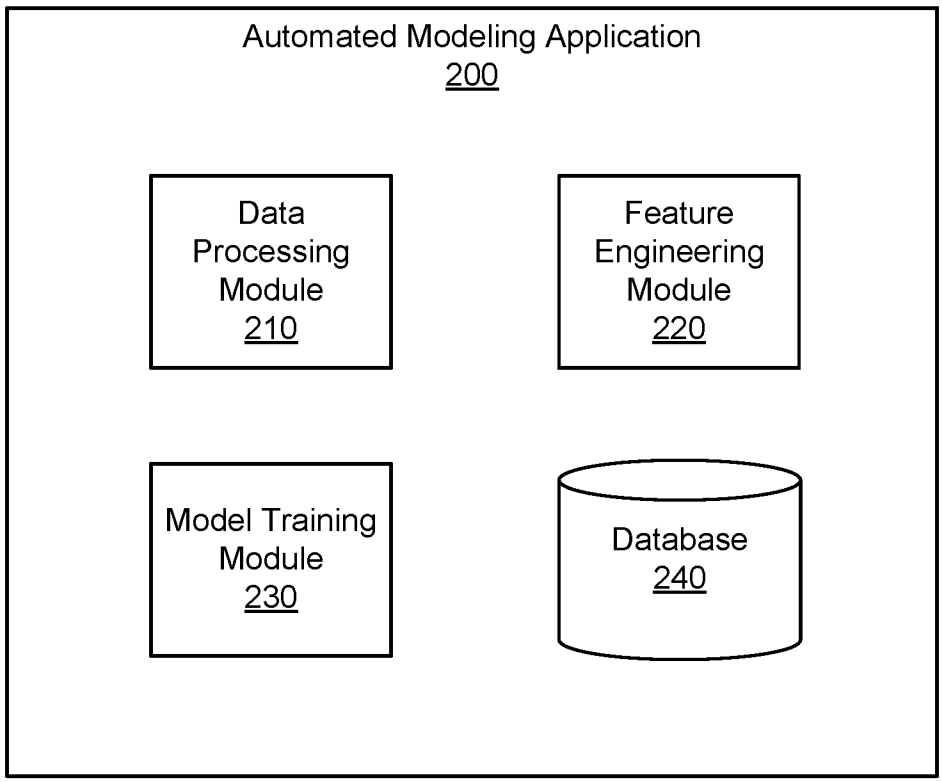

Automated Modeling Application
200
Data Processing Module
210
Feature Engineering Module
220
Model Training Module
230
Database
240

FIG. 2

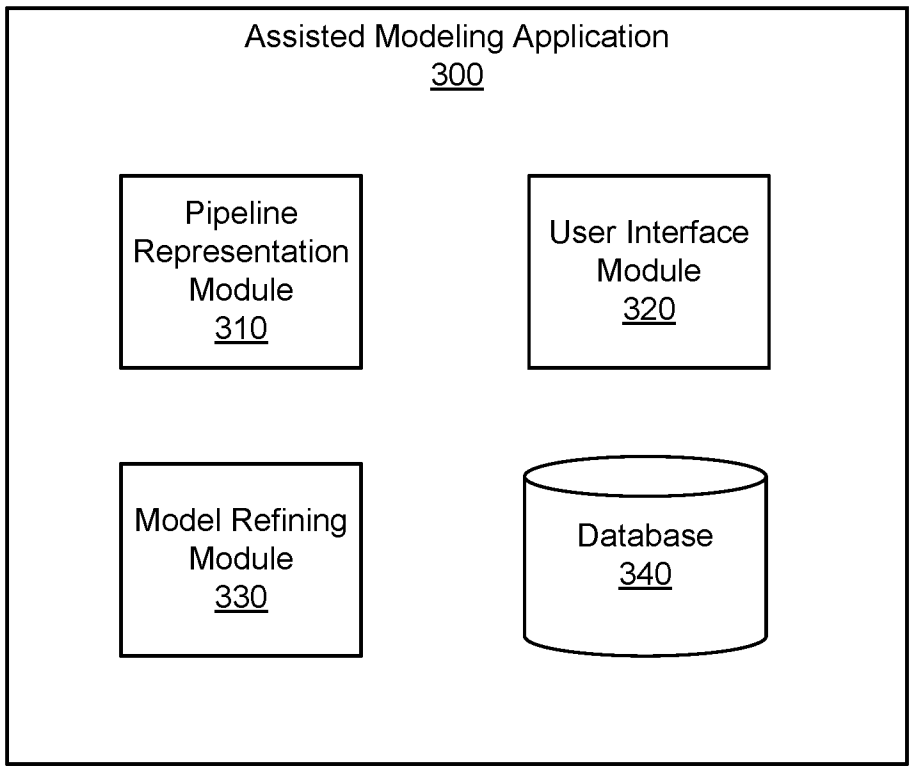

Assisted Modeling Application
300
Pipeline Representation Module
310
User Interface Module
320
Model Refining Module
330
Database
340

FIG. 3

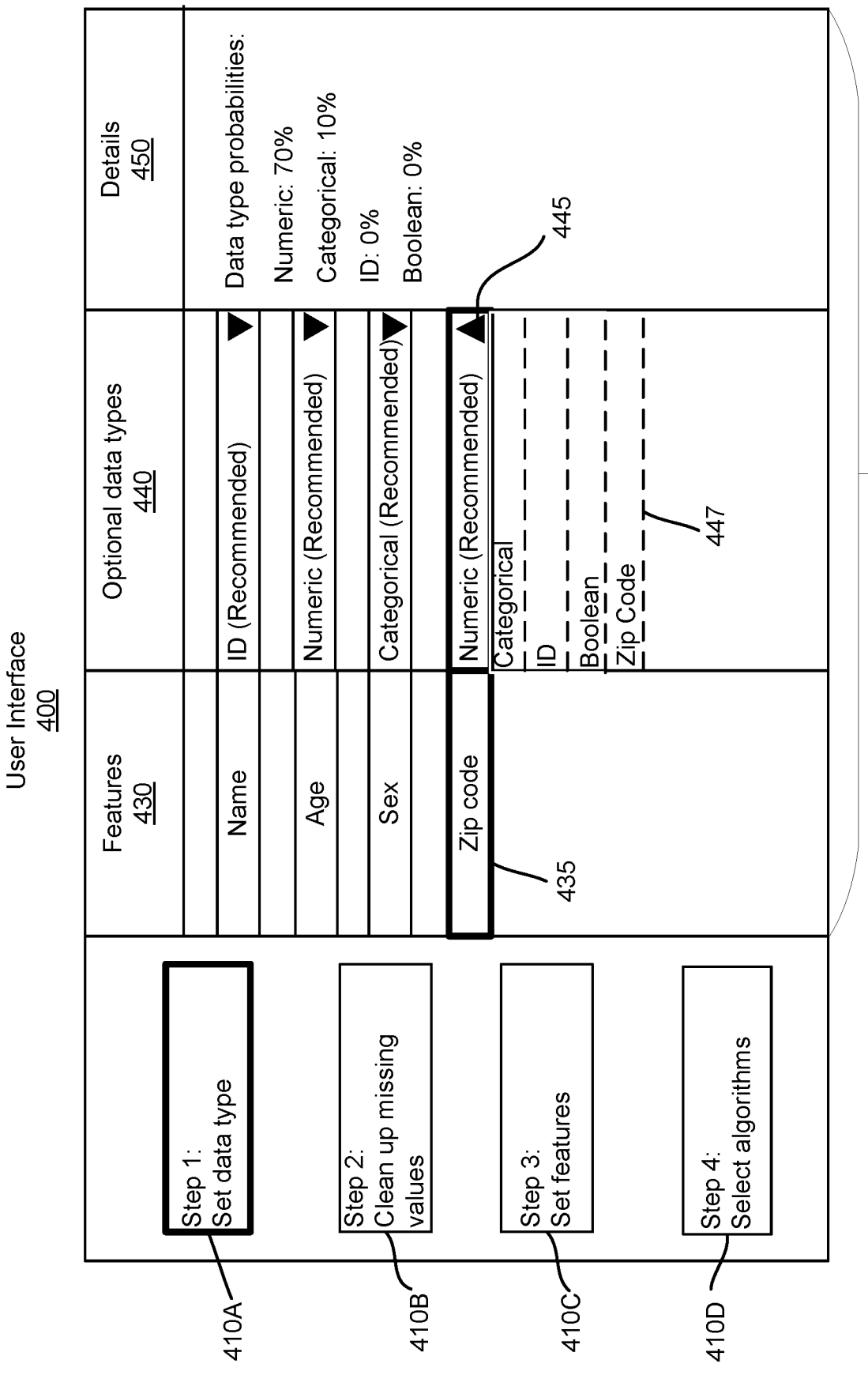

User Interface
400
Features
430
Optional data types
440
Details
450
Name
ID (Recommended)
Data type probabilities:
Numeric: 70%
Categorical: 10%
ID: 0%
Boolean: 0%
Age
Numeric (Recommended)
Sex
Categorical (Recommended)
Zip code
Numeric (Recommended)
Categorical
ID
Boolean
Zip Code
435
445
447
Data type list
420
Step 1:
Set data type
Step 2:
Clean up missing values
Step 3:
Set features
Step 4:
Select algorithms
410A
410B
410C
410D

FIG. 4

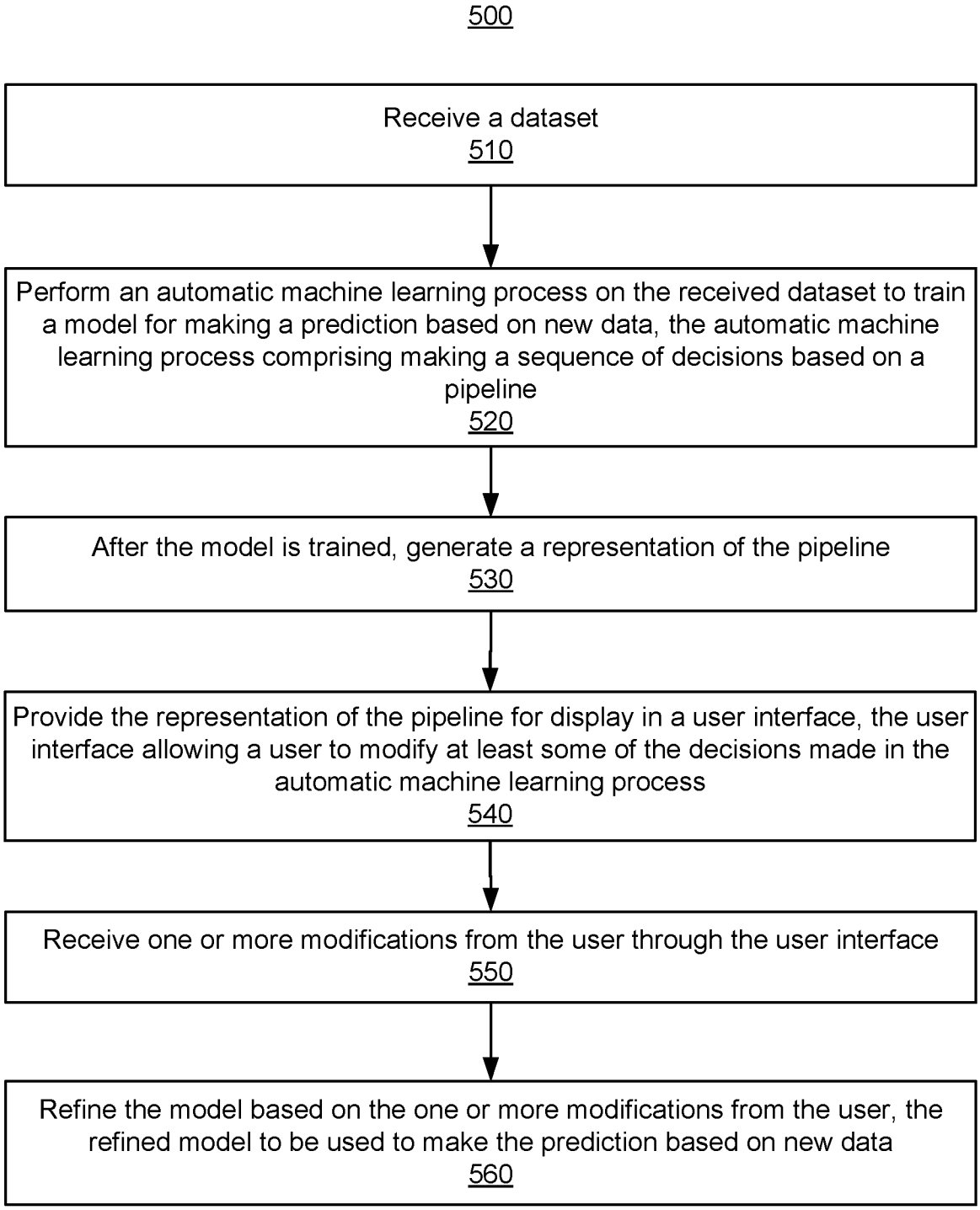

500
Receive a dataset
510
Perform an automatic machine learning process on the received dataset to train a model for making a prediction based on new data, the automatic machine learning process comprising making a sequence of decisions based on a pipeline
520
After the model is trained, generate a representation of the pipeline
530
Provide the representation of the pipeline for display in a user interface, the user interface allowing a user to modify at least some of the decisions made in the automatic machine learning process
540
Receive one or more modifications from the user through the user interface
550
Refine the model based on the one or more modifications from the user, the refined model to be used to make the prediction based on new data
560

FIG. 5

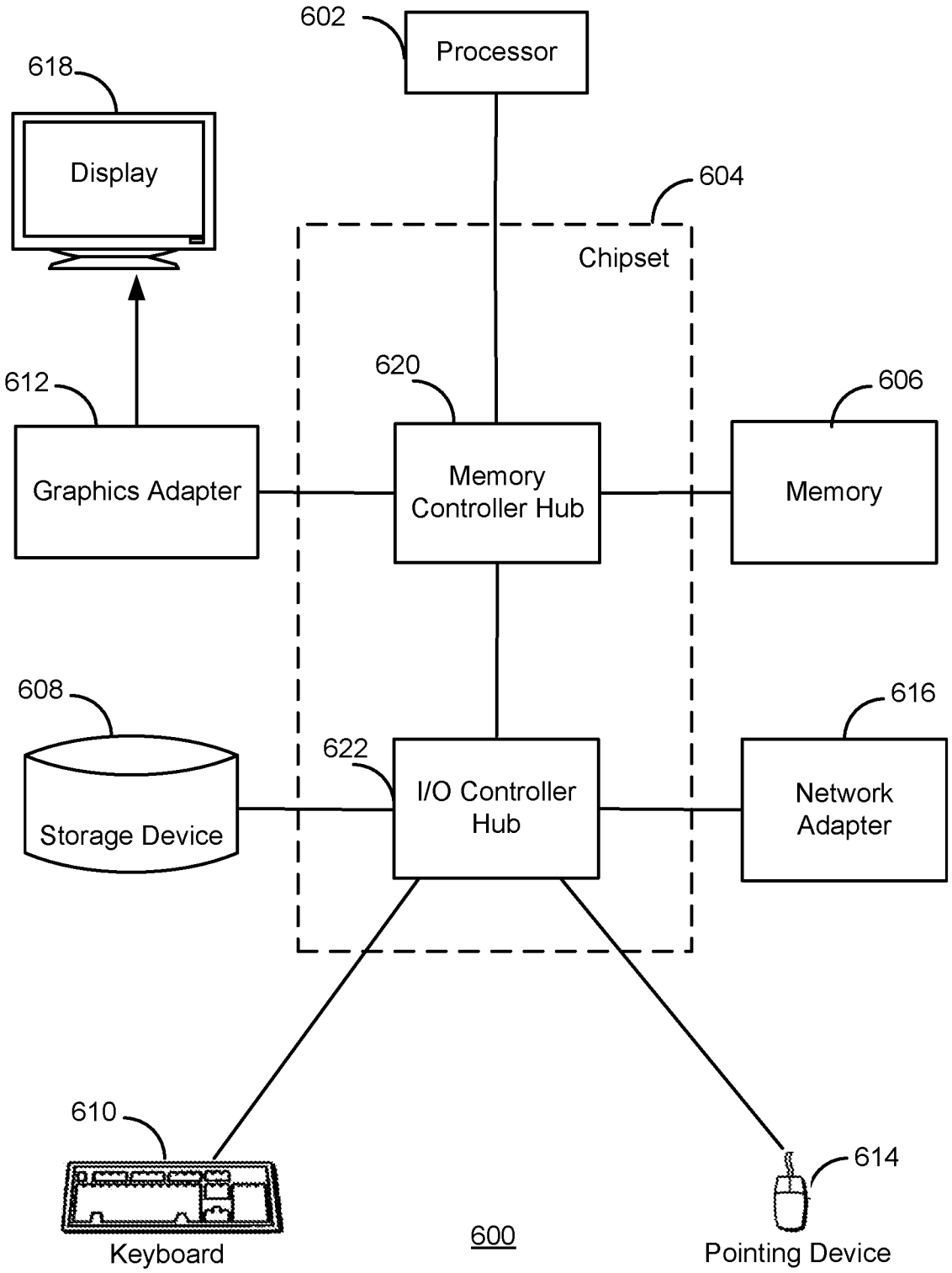

602
Processor
618
Display
604
Chipset
620
612
606
Graphics Adapter
Memory Controller Hub
Memory
608
616
622
Storage Device
I/O Controller Hub
Network Adapter
610
614
600
Keyboard
Pointing Device

FIG. 6

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 21/47334

A. CLASSIFICATION OF SUBJECT MATTER
IPC - G06N 5/04, G06N 20/00 (2021.01)
CPC - G06N 20/00, G09B 5/00, G06N 5/04

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
See Search History document

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
See Search History document

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2015/0379429 A1 (AMAZON TECHNOLOGIES, INC), 31 December 2015 (31.12.2015), entire document, especially Abstract; para [0115], [0154], [0157], [0159], [0204]	1-20
Y	US 2019/0384461 A1 (SAP SE), 19 December 2019 (19.12.2019), entire document, especially Abstract; para [0019], [0029]-[0030], [0038]-[0041]	1-20
Y	US 2014/0280191 A1 (Cronin et al.), 18 September 2014 (18.09.2014), entire document, especially Abstract; para [0162]-[0166], [0481]-[0486]	3-5, 10-12, 17-18
Y	US 2003/0126102 A1 (Borthwick), 03 July 2003 (03.07.2003), entire document, especially Abstract; para [0041]-[0045], [0097], [0152]-[0155]	6, 13, 19
Y	US 2017/0308800 A1 (Smokescreen Intelligence, LLC), 26 October 2017 (26.10.2017), entire document, especially Abstract; para [0027]-[0030], [0044], [0049]-[0052], [0071]	7, 14, 20

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
20 October 2021	NOV 23 2021
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-8300	Authorized officer Kari Rodriquez Telephone No. PCT Helpdesk: 571-272-4300