

Comparison of petition paragraphs and cited paragraphs of the Mitzenmacher Declaration.

Petition at 7	Mitzenmacher Decl. ¶ 38
In discussing the “predictive compiler module 210,” the ’574 patent discloses algorithm steps to achieve recited function(s), such as generating machine learning (claim 1) and generating multiple predictive programs (claim 21). STRATEGY-1001, 19:13-21 (discussing module 210), 20:7-15 (summarizing algorithm). For instance, the algorithm steps include (i) receiving training data (<i>id.</i> , 20:24-38), (ii) generating learned functions from the training data (<i>id.</i> , 21:17-35), and (iii) forming a machine learning ensemble from the learned functions (<i>id.</i> , 23:28-38), where the output is executable program code (<i>id.</i> , 19:22-33, 6:55-59). STRATEGY-1003, ¶¶38-39.	In discussing the “predictive compiler module 210,” the ’574 patent discloses algorithm steps to achieve functions such as generating machine learning (claim 1) and generating multiple predictive programs (claim 21). STRATEGY- 1001, 19:13-21 (discussing module 210), 20:7-15 (summarizing algorithm). The algorithm steps include (i) receiving training data (<i>id.</i> , 20:24-38), (ii) generating learned functions from the training data (<i>id.</i> , 21:17-35), and (iii) forming a machine learning ensemble from the learned functions (<i>id.</i> , 23:28-38), where the output is executable program code (<i>id.</i> , 19:22-33, 6:55-63).

Petition at 8	Mitzenmacher Decl. ¶ 40
In discussing the “input module 202,” the ’574 patent discloses a user interface that receives user input through user manipulation of “a slider graphical user interface element,” “a graphical table,” or a “displayed object.” STRATEGY-1001, 15:60-16:42; STRATEGY-1003, ¶40.	In discussing the “input module 202,” the ’574 patent discloses a user interface that receives user input of values through user manipulation of “a slider graphical user interface element,” “a graphical table,” or a “displayed object.” STRATEGY-1001, 15:60-16:42.

Petition at 8	Mitzenmacher Decl. ¶ 41
In discussing the “pre-compute module 208,” the ’574 patent discloses algorithm steps to predetermine various machine learning results. STRATEGY-1001, 16:51-57. The algorithm steps include (i) computing multiple results over a range before the results are requested by a user (<i>id.</i> , 18:12-27), and (ii) caching the results	In discussing the “pre-compute module 208,” the ’574 patent discloses algorithm steps to predetermine various machine learning results. STRATEGY- 1001, 16:51-57. The algorithm steps include (i) computing multiple results over a range before the results are requested by a user (<i>id.</i> , 18:12-27), and (ii) caching the results

indexed by the inputs (<i>id.</i> , 18:46-61). The results are stored in a “results data structure” which may include “a lookup table, a tree, a linked list, a heap, a stack, or another indexable data structure.” <i>Id.</i> , 18:51-53; STRATEGY-1003, ¶41.	indexed by the inputs (<i>id.</i> , 18:46-61). The results are stored in a “results data structure” which may include “a lookup table, a tree, a linked list, a heap, a stack, or another indexable data structure.” <i>Id.</i> , 18:51-53.
--	--

Petition at 8	Mitzenmacher Decl. ¶ 42
In discussing the “display module 204,” the ’574 patent discloses a user interface that displays information about precomputed results, including machine learning results (claim 1) or machine learning parameters (claim 21). STRATEGY-1001, 17:14-36. Similar to the input module 202, the display module can display information using a “graphical table,” a “slider graphical user interface element,” or a “displayed object.” <i>Id.</i> ; STRATEGY-1003, ¶42.	In discussing the “display module 204,” the ’574 patent discloses a user interface that displays information about precomputed results, including machine learning results (claim 1) or machine learning parameters (claim 21). STRATEGY-1001, 17:14-36. Similar to the input module 202, the display module can display information using a “graphical table,” a “slider graphical user interface element,” or a “displayed object.” <i>Id.</i>

Petition at 9	Mitzenmacher Decl. ¶ 43
In discussing the “update module 206,” the ’574 patent discloses algorithm steps to update displayed information in response to user input. STRATEGY-1001, 17:37-61. The algorithm steps include (i) detecting user input (<i>id.</i> , 17:45-55), (ii) looking up information that corresponds to the user input in a data structure (<i>id.</i> , 17:56-61), and (iii) updating information displayed by the display module (<i>id.</i> , 17:37-44). STRATEGY-1003, ¶43.	In discussing the “update module 206,” the ’574 patent discloses algorithm steps to update displayed information in response to user input. STRATEGY-1001, 17:37-61. The algorithm steps include (i) detecting user input (<i>id.</i> , 17:45-55), (ii) looking up information that corresponds to the user input in a data structure (<i>id.</i> , 17:56-61), and (iii) updating information displayed by the display module (<i>id.</i> , 17:37-44).

Petition at 9	Mitzenmacher Decl. ¶ 44
In discussing the “collaboration module 216,” the ’574 patent discloses algorithm steps to indicate an impact that different	In discussing the “collaboration module 216,” the ’574 patent discloses algorithm steps to indicate an impact that different

factors or values have on machine learning results. STRATEGY-1001, 19:45-20:1. The algorithm steps include (i) accessing data relating to different models or different user inputs (<i>id.</i> , 19:1-10), and (ii) determining an impact (e.g., difference or change) on results from the accessed data (<i>id.</i> , 19:45-20:1). STRATEGY-1003, ¶44.	factors or values have on machine learning results. STRATEGY-1001, 19:45-20:1. The algorithm steps include (i) accessing data relating to different models or different user inputs (<i>id.</i> , 19:1-10), and (ii) determining an impact (e.g., difference or change) in results from the accessed data (<i>id.</i> , 19:45-20:1).
--	--

Petition at 9	Mitzenmacher Decl. ¶ 45
In discussing the “recommendation module 214,” the ’574 patent discloses an algorithm step to “select a suggested or optimal set of machine learning results” such as “a set of default or recommended machine learning results or the like.” STRATEGY-1001, 19:39-44; STRATEGY-1003, ¶45.	In discussing the “recommendation module 214,” the ’574 patent discloses an algorithm step to “select a suggested or optimal set of machine learning results” such as “a set of default or recommended machine learning results or the like.” STRATEGY-1001, 19:39-44.

Petition at 10	Mitzenmacher Decl. ¶ 28
A person of ordinary skill in the art (“POSITA”) relating to the subject matter of the ’574 patent as of the Critical Date would have had (1) a bachelor’s degree in computer science, data science, or a related field, and (2) two years of industry experience in machine learning applications and software. STRATEGY-1003, ¶28.	In my opinion, a person of ordinary skill in the art (“POSITA”) relating to the subject matter of the ’574 patent as of May 29, 2013, would have had (1) a bachelor’s degree in computer science, data science, or a related field, and (2) two years of industry experience in machine learning applications and software. Additional education could substitute for industry experience, and vice versa. <i>Id.</i> Further, all positions in this declaration are from the vantage point of a POSITA as of May 29, 2013.

Petition at 10	Mitzenmacher Decl. ¶ 68
The ’574 patent is directed to “apparatuses, systems, methods, and computer program products for machine learning results.” STRATEGY-1001,	The ’574 patent is directed to “apparatuses, systems, methods, and computer program products for machine learning results.” STRATEGY-1001,

Abstract. The '574 patent describes a “results module 102” that allows users to “request machine learning ensembles, to make analysis requests, and to receive machine learning results....” <i>Id.</i>, 6:48-63. The results module 102 “generates machine learning ensembles or other machine learning program code” for making predictions. <i>Id.</i>, 10:47-57. With the generated program code, the results module 102 pre-computes and stores various permutations of machine learning results, so results can be presented “with little or no delay.” <i>Id.</i>, 7:41-53.	Abstract. The '574 patent describes a “results module 102” that allows users to “request machine learning ensembles, to make analysis requests, and to receive machine learning results. . . .” <i>Id.</i>, 6:48-63. The results module 102 “generates machine learning ensembles or other machine learning program code” for making predictions. <i>Id.</i>, 10:47-57. With the generated program code, the results module 102 pre-computes and stores various permutations of machine learning results, so that results can be presented “with little or no delay in response to receiving input from a user.” <i>Id.</i>, 7:41-53.
---	--

Petition at 10–11	Mitzenmacher Decl. ¶ 69
The application that issued as the '574 patent was filed with 26 claims, including three independent claims. STRATEGY-1002, 533-572. The examiner issued only one office action, which rejected all claims as obvious.	The application that issued as the '574 patent was filed with 26 claims, including three independent claims. STRATEGY-1002, 533-572. The examiner issued only one office action, which rejected all claims as obvious under 35 U.S.C. § 103 as being unpatentable over U.S. Pre-Grant Pub. No. 2005/0090911 to Ingargiola et al. and combinations including U.S. Patent No. 8,260,117 to Xu et al. and U.S. Pre-Grant Pub. No. 2009/0222742 to Pelton et al. <i>Id.</i>, 148-173.

Petition at 11	Mitzenmacher Decl. ¶ 70
The applicant responded to the office action by cancelling one claim and adding new features to each of the independent claims, including (i) adding the “predictive compiler module” and “pre-compute module” to claim 1, (ii) adding “generating a predictive program” and “dynamically changing a display of one or more input permutations” to the claim that issued as	The applicant responded to the office action by cancelling one claim and adding several new features to each of the independent claims, including (i) adding the “predictive compiler module” and “pre-compute module” to claim 1, (ii) adding “generating a predictive program” and “dynamically changing a display of one or more input permutations” to the claim

claim 16, and (iii) adding the “predictive compiler module” and “collaboration module” to the claim that issued as claim 21.” STRATEGY-1002, 71-90. After this amendment, the examiner issued a notice of allowance that stated simply “[s]ee applicants’ arguments” from the response as reasons for allowance. <i>Id.</i> , 16-21.	that issued as claim 16, and (iii) adding the “predictive compiler module” and “collaboration module” to the claim that issued as claim 21.” STRATEGY-1002, 71-90. After this amendment, the examiner issued a notice of allowance that stated simply “[s]ee applicants’ arguments” from the response as reasons for allowance. <i>Id.</i> , 16-21.
--	---

Petition at 11	Mitzenmacher Decl. ¶ 71
The file history contains no indication that the examiner ever considered Johnson, Lin, Purcell, or Mihaylov discussed in Grounds 1A-1D below.	The file history contains no indication that the examiner ever considered Johnson, Lin, Purcell, or Mihaylov discussed in Grounds 1A-1D below.

Petition at 11–12	Mitzenmacher Decl. ¶ 72
Johnson describes an “information and decisioning control system that provides business forecasts.” STRATEGY-1004, ¶[0029]. Johnson’s system provides a “digital cockpit” user interface that displays forecasts or predictions “based on the course of the business thus far” and based on hypothetical “what-if” situations defined “by inputting a set of assumptions.” <i>Id.</i> , ¶¶[0030]-[0031], [0052]. The system generates forecasts with models 136 that can use machine learning, e.g., “regressive analysis techniques, time series analyses, artificial intelligence analyses, extrapolation and logic analyses, etc.” <i>Id.</i> , ¶¶[0047], [0070] (models can use a “function fitted to empirical data”); STRATEGY-1003, ¶72. Below, FIG. 3 shows an example of Johnson’s “digital cockpit” interface.	Johnson describes an “information and decisioning control system that provides business forecasts.” STRATEGY-1004, ¶[0029]. Johnson’s system describes a “digital cockpit” user interface that allows a user to view forecasts or predictions “based on the course of the business thus far in time” or based on hypothetical “what-if” situations defined “by inputting a set of assumptions.” <i>Id.</i> , ¶¶[0030]-[0031], [0052]. According to Johnson, the system generates forecasts with models 136 that can use “regressive analysis techniques, time series analyses, artificial intelligence analyses, extrapolation and logic analyses, etc.,” which a POSITA would have understood are examples of machine learning. <i>Id.</i> , ¶[0047]. A POSITA would also have recognized that Johnson’s models 136 represent machine learning by being based on a “function fitted to empirical data,” which means that a function is

	learned from the empirical data (e.g., training data). <i>Id.</i>, ¶[0070]. Even though Johnson does not use the term “machine learning,” the models and techniques that Johnson describes were understood by a POSITA to be referring to machine learning. STRATEGY-1027, 1-3, 9-11 (describing obtaining a “fitted function” through regression and supervised learning as machine learning techniques, where one can “collect [] training data and the machine learning program fits a function to this data to learn Y as a function of X.”); STRATEGY-1028, 19 (describing machine learning to include an “algorithm generates a function that maps inputs to desired outputs”).
--	--

Petition at 12–13	Mitzenmacher Decl. ¶ 73
A key aspect of Johnson’s system is “generating business analysis results in advance of a request for the results.” STRATEGY-1004, Title. The system “includes pre-loading logic 230” and “a database 234 for storing the results pre-calculated by the pre-loading logic 230.” <i>Id.</i>, ¶¶[0068]–[0071]. Johnson recognizes that “processing cases using the models 136 may be time-intensive,” and “a delay may be present when a user requests a particular analysis to be performed in real-time fashion.” <i>Id.</i>, ¶[0068]; STRATEGY-1003, ¶73. “To reduce this delay, the pre-loading logic 230 performs analysis in advance of a user’s request.” <i>Id.</i>	A key aspect of Johnson’s system is “generating business analysis results in advance of a request for the results.”² STRATEGY-1004, Title. The system includes “includes pre-loading logic 230” and “a database 234 for storing the results pre-calculated by the pre-loading logic 230.” <i>Id.</i>, ¶¶[0068]–[0071]. Johnson observes that “processing cases using the models 136 may be time-intensive,” and so “a delay may be present when a user requests a particular analysis to be performed in real-time fashion.” <i>Id.</i>, ¶[0068]. “To reduce this delay, the pre-loading logic 230 performs analysis in advance of a user’s request.” <i>Id.</i>

Petition at 13	Mitzenmacher Decl. ¶ 74
Once “the results pre-calculated by the pre-loading logic 230” are stored in the database 234, “the digital cockpit 104	According to Johnson, once “the results pre-calculated by the pre-loading logic 230” are stored in the database 234, “the

can retrieve results from this database when the user requests these results, instead of calculating these results at the time of request,” which “reduces the time delay” for presenting results and “supports the overarching aim of the digital cockpit 104, which is to provide timely and accurate results to the cockpit user 138....” <i>Id.</i>, ¶¶[0071], [0152] (system “provides the results to the user substantially instantaneously, as opposed to imposing a delay”). Johnson’s FIG. 2, below, illustrates the preloading logic 230 and database 234 to store the pre-calculated results. <i>Id.</i>, FIG. 2, ¶¶[0068], [0071]; see also <i>id.</i>, FIG. 10 (showing database 234, pre-calculation 1012, and result retrieval 1020); STRATEGY-1003, ¶74.	digital cockpit 104 can retrieve results from this database when the user requests these results, instead of calculating these results at the time of request. This retrieval reduces the time delay associated with the presentation of output results, and supports the overarching aim of the digital cockpit 104, which is to provide timely and accurate results to the cockpit user 138 when the cockpit user 138 needs such results.” <i>Id.</i>, ¶¶[0071], [0152] (pre-calculation and retrieval “provides the results to the user substantially instantaneously, as opposed to imposing a delay”), [0153] (“pre-loading logic 230 within analysis logic 222 determines calculations that should be performed in advance, and then proceeds to perform these calculations in an off-line manner.... Once results are computed, the pre-loading logic 230 stores the results in the pre-loaded results database 234.”
--	--

Petition at 14–15	Mitzenmacher Decl. ¶ 83
Lin describes machine learning techniques that can be used to create the types of predictive models that generate forecasts in Johnson’s system. STRATEGY-1005, Abstract, 3:4-59; STRATEGY-1004, ¶¶[0047], [0070]; STRATEGY-1003, ¶83. Lin explains that a “machine learning algorithm can train a predictive model by systematically analyzing the applied training dataset and generating an inferred function that ‘fits’ the training data.” STRATEGY-1005, 3:41-47. Lin’s training techniques create an “inferred function” and this provides the kind of “function fitted to empirical data” used in Johnson’s system. STRATEGY-1005, 3:41-47; STRATEGY-1004, ¶[0070]; STRATEGY-1003, ¶83.	Lin describes techniques for creating and hosting predictive models. STRATEGY-1005, Title, Abstract. Lin describes machine learning techniques that can be used to create the types of predictive models that generate forecasts in Johnson’s system. STRATEGY-1005, Abstract, 3:4-59; STRATEGY-1004, ¶¶[0047], [0070]. Lin explains that a “machine learning algorithm can train a predictive model by systematically analyzing the applied training dataset and generating an inferred function that ‘fits’ the training data. The trained predictive model can be representative of (or defined by) the training dataset and operable to map a feature vector to a predictive outcome according to the inferred

	function.” <i>Id.</i> , 3:41-47. Lin’s training to create an “inferred function” from training data allows creation of the same type of “function fitted to empirical data” used in Johnson’s system. STRATEGY-1005, 3:41-47; STRATEGY-1004, ¶[0070].
--	---

Petition at 15	Mitzenmacher Decl. ¶ 84
Lin’s system includes functionality to implement predictive models in an executable format. STRATEGY-1005, 3:60-4:5; STRATEGY-1003, ¶84. Lin’s system can “implement [a] machine learning algorithm to generate a predictive model that is representative of [a] training dataset,” and then “output the predictive model in a suitable computer readable and executable format.” STRATEGY-1005, 3:65-4:5. Lin’s FIG. 2 (below) shows creation of a “model implementation 214 [that] is one or more computer programs that execute on one or more servers.” <i>Id.</i> , 9:15-38.	Lin’s system includes functionality to implement or represent predictive models in an executable format. STRATEGY-1005, 3:60-4:5. For example, Lin’s system includes software that can “implement [a] machine learning algorithm to generate a predictive model that is representative of [a] training dataset,” and then “output the predictive model in a suitable computer readable and executable format.” <i>Id.</i> , 3:65-4:5. This functionality is illustrated in FIG. 2 of Lin, where one or more servers of a data warehouse 108 create a “model implementation 214 [that] is one or more computer programs that execute on one or more servers.” <i>Id.</i> , 9:15-38.

Petition at 16	Mitzenmacher Decl. ¶ 85
Lin’s system also enables training and creation of predictive models generally. STRATEGY-1005, 2:24-34, 10:44-56; STRATEGY-1003, ¶85. This includes the ability for users to “actively generate new predictive models” (STRATEGY-1005, 2:24-34) and “upload training datasets from web application data files (or from other sources of data) and use the uploaded datasets to train a predictive model” (<i>id.</i> , 6:14-25). Using its FIG. 4 (below), Lin provides an example where “models are trained by a training system 416 which receives requests from the prediction API	Lin’s FIG. 2 shows a model implementation 214 created from an imported model, and Lin’s system also enables training and creation of predictive models more generally. Lin’s system enables users to “actively generate new predictive models” (<i>id.</i> , 2:24-34) and “upload training datasets from web application data files (or from other sources of data) and use the uploaded datasets to train a predictive model” (<i>id.</i> , 6:14-25). Lin describes an example where “models are trained by a training system 416 which receives requests from the prediction API 408 to

408 to initiate training and check the status of training” of predictive models 316 based on training data 320. <i>Id.</i> , 10:44-56.	initiate training and check the status of training” of predictive models 316 based on training data 320. <i>Id.</i> , 10:44-56.
--	---

Petition at 17	Mitzenmacher Decl. ¶ 86
Lin’s system also combines predictive models into an ensemble to improve performance. STRATEGY-1005, 4:16-35; STRATEGY-1003, ¶86. Lin explains that “a number of individual predictive models can be combined to form an ensemble” and that the ensemble can “achieve a more accurate prediction than can be provided by the individual models themselves.” <i>Id.</i> , 4:16-20; see also <i>id.</i> , 5:8-22.	Lin’s system also combines predictive models into an ensemble to improve performance. STRATEGY-1005, 4:16-35. Lin explains that “a number of individual predictive models can be combined to form an ensemble. For example, output from the individual predictive models can be combined or integrated to achieve a more accurate prediction than can be described by the individual models themselves.” <i>Id.</i> , 4:16-20; see also <i>id.</i> , 5:8-22.

Petition at 18	Mitzenmacher Decl. ¶ 92
A POSITA would have found it obvious and would have been motivated to apply Lin’s teachings to Johnson’s forecasting system to augment functionality using predictive models using machine learning algorithms based on Lin. STRATEGY-1003, ¶92. Aware of Lin, a POSITA implementing Johnson’s techniques would have found it obvious to train models using then-known system components and machine learning algorithms, based on Lin, allowing the combined system to not only apply Johnson’s models 136 but also create and train models 136 for Johnson’s system to use. <i>Id.</i>	A POSITA would have found it obvious and would have been motivated to apply Lin’s teachings to Johnson’s forecasting system to add functionality to create predictive models using machine learning algorithms based on Lin. A POSITA implementing Johnson’s techniques would have found it obvious to include system components and machine learning algorithms to train models, based on Lin, so the combined system can create and train models 136 for Johnson’s system to use in making its forecasts. The training techniques taught by Lin were known prior art elements, and in the combination with Johnson they are used in the same way and for the same purpose that Lin describes. A POSITA would have been able to use conventional method to apply Lin’s model training techniques to achieve the natural result of adding model training capability to

	Johnson’s system. STRATEGY-1005, 3:41-47, 2:24-34, 6:14-25, 10:44-56. Moreover, a POSITA would have recognized that using ensembles was obvious as the application of known techniques to Johnson’s system that was ready for improvement with the increased accuracy and reliability Lin’s ensemble techniques provide.
--	--

Petition at 18–19	Mitzenmacher Decl. ¶ 93
Lin teaches “generating an inferred function that ‘fits’ the training data” using a “machine learning algorithm,” which allows a predictive model to map an input to a predicted output. STRATEGY-1005, 3:41-47. Johnson’s models 136 similarly use machine learning and use transfer functions to map inputs to predicted outputs, and a POSITA would have recognized that Lin’s machine learning training would enable Johnson’s system to create new models and not simply apply them. STRATEGY-1004, ¶[0047]; STRATEGY-1003, ¶¶75-82, 90-92. A POSITA also would have recognized that Lin’s use of machine learning to fit a dataset is applicable to Johnson’s system from Johnson’s use of similar technique to make predictions based on datasets (e.g., “regressive analysis techniques,” “artificial intelligence analyses,” “extrapolation”). STRATEGY-1005, 3:39-59; STRATEGY-1004, ¶¶[0047], [0070]; STRATEGY-1003, ¶¶93-94. Thus, the configuration was obvious as the application of Lin’s known techniques to Johnson’s system that was ready for improvement. STRATEGY-1003, ¶¶88, 92.	Lin teaches “generating an inferred function that ‘fits’ the training data” using a “machine learning algorithm” so that the trained predictive model is “operable to map [an input] feature vector to a predictive outcome according to the inferred function.” STRATEGY-1005, 3:41-47. Johnson’s models 136 similarly use transfer functions to map inputs to predicted outputs, and so a POSITA would have recognized that Lin’s use of machine learning algorithms to train the functions of predictive models would enhance the capability of Johnson’s system. STRATEGY-1004, ¶¶[0047], [0070]. A POSITA also would have recognized that Lin’s use of machine learning to fit a dataset are applicable to Johnson’s system because Johnson also already demonstrates using models and techniques that generate predictions based on characteristics of previous datasets (e.g., “regressive analysis techniques,” “artificial intelligence analyses,” “extrapolation,” functions “fitted to empirical data”). STRATEGY-1005, 3:39-59; STRATEGY-1004, ¶¶[0047], [0070].

Petition at 21	Mitzenmacher Decl. ¶ 102
It would further have been obvious for individual computer programs to each incorporate multiple functions, which enables computer programs to implement models that include “plural transfer functions” and generate output based on multiple models based on Johnson and to benefit from the improved accuracy of ensembles, which Lin describes. STRATEGY-1004, ¶¶ [0047], [0080]; STRATEGY-1005, 4:16-20; STRATEGY-1003, ¶102. In addition, including multiple functions in a computer program was an obvious design choice, since a POSITA would have only had two options: (1) a separate program for each function, or (2) a program that includes multiple functions. STRATEGY-1003, ¶102.	It would further have been obvious for individual computer programs to each incorporate multiple functions, so that the computer programs can implement models that include “plural transfer functions” and to generate outputs based on multiple models and to benefit from the improved accuracy of ensembles. STRATEGY-1004, ¶¶[0047], [0080] (describing output generated “using one or more models 136”); STRATEGY-1005, 4:16-20 (describing improved accuracy of ensembles). In addition, including multiple functions in a computer program was an obvious design choice. When implementing predictive models or learned functions as computer programs, a POSITA would have had only had two options, either to make each predictive model or learned function a separate program, or to combine multiple models or functions together in a program. As discussed above, with an ensemble of predictive models, multiple models or learned functions are intended to operate together. As a result, a POSITA would have found it convenient and obvious to include the multiple inferred functions or models of the ensemble in a single program. That way, the components of the ensemble that are needed to generate a forecast would be in the same program, making deployment of the program and generation of a prediction easier and more reliable. STRATEGY-1012, ¶¶[0005] noting how compiling models into programs was known to improve execution and distribution), [0033].

Petition at 22	Mitzenmacher Decl. ¶ 103
Johnson explains that “the cockpit control module 132 can be implemented as any kind of computer device.” STRATEGY-1004, ¶[0065]. Johnson’s FIG. 2 example illustrates the cockpit control module 132 as a server, but Johnson’s FIG. 1 example and claims further demonstrate that Johnson’s techniques are not limited to a client/server architecture. STRATEGY-1004, FIGS. 1-2, ¶[0065], claims 22-23 (showing that a server implementation recited in dependent claim 23 is not required for independent claim 22). A POSITA would have found it obvious and would have been motivated to enable the cockpit control module 132 to receive user input and display output, and for the cockpit control module 132 to correspondingly receive input and display output as represented in Johnson’s FIG. 1. STRATEGY-1003, ¶¶103-104.	Johnson explains that “the cockpit control module 132 can be implemented as any kind of computer device.” STRATEGY-1004, ¶[0065]. Johnson’s FIG. 2 example illustrates the cockpit control module 132 as a server, but Johnson’s FIG. 1 example and claims further demonstrate that Johnson’s techniques are not limited to a client/server architecture. STRATEGY-1004, FIGS. 1-2, ¶[0065], claims 22-23 (showing that a server implementation recited in dependent claim 23 is not required for independent claim 22). A POSITA would have found it obvious and would have been motivated to enable the cockpit control module 132 to receive user input and display output, so the cockpit control module 132 can receive input and display output as represented in Johnson’s FIG. 1. For the POSITA to achieve this result, it would have been obvious for the cockpit control module 132 to include input and output devices to present the digital cockpit interface 134 and receive corresponding user input. STRATEGY-1004, ¶[0065].

Petition at 22–23	Mitzenmacher Decl. ¶ 105
A POSITA would have viewed an implementation of Johnson’s system in a manner that applied Lin’s teachings (e.g., that trains models, combines them into ensembles, and implements the ensembles as computer programs) to be the predictable result of applying Lin’s known techniques to Johnson’s system that was ready for improvement, and a POSITA would have expected success in	The application of Lin’s techniques I have described above provides an implementation of Johnson’s system that that trains models, combines them into ensembles, and implements them as computer programs. Using Lin’s techniques in this way is within the ability of a POSITA, and a POSITA would have expected that Lin’s techniques would be successfully applied in Johnson’s system. For example, the changes to Johnson’s system represent applying Lin’s known

enhancing Johnson’s system in this way. STRATEGY-1003, ¶105.	techniques to achieve the same result Lin achieves, e.g., generating executable computer programs that implement machine learning ensembles, where Johnson’s system is ready to be improved by these techniques.
--	--

Petition at 25	Mitzenmacher Decl. ¶ 158
To the extent the preamble is limiting, the Johnson-Lin combination provides this element. STRATEGY-1003, ¶158. Johnson describes generating forecasts using models 136 that use machine learning techniques such as “regressive analysis techniques,” “artificial intelligence analyses,” and “extrapolation,” as well as a “function fitted to empirical data ” and thus learned by the system. STRATEGY-1004, ¶¶[0047], [0147], [0070]. Lin describes using a “ machine learning algorithm to train a predictive model ” and “ generating an inferred function that ‘fits’ the training data. ” STRATEGY-1005, 3:39-44.	To the extent the preamble is limiting, the Johnson-Lin combination describes this feature. Johnson describes generating forecasts using models 136 that can use “regressive analysis techniques,” “artificial intelligence analyses,” “extrapolation,” and other machine learning techniques. STRATEGY-1004, ¶¶[0047], [0147]. Johnson teaches that the models can represent a function that is “a mathematical equation or other function fitted to empirical data, ” and thus is learned by the system. STRATEGY-1004, ¶[0070]. Lin describes that “a training dataset can be applied to a selected machine learning algorithm to train a predictive model. In more detail, the machine learning algorithm can train a predictive model by systematically analyzing the applied training dataset and generating an inferred function that ‘fits’ the training data. ” STRATEGY-1005, 3:39-44.

Petition at 25	Mitzenmacher Decl. ¶ 159
In the Johnson-Lin combination, Johnson’s cockpit control module 132 is configured to train Johnson’s models 136 with machine learning algorithms, based on Lin. STRATEGY-1003, ¶159. The cockpit control module 132 generates machine learning results from the trained models and	In the Johnson-Lin combination, Johnson’s cockpit control module 132 is configured to train Johnson’s models 136 with machine learning algorithms based on Lin. The cockpit control module 132 generates results from the models that are the result of machine learning and then displays the

provides the results (e.g., from database 234), and so provides an apparatus for machine learning results. STRATEGY-1004, ¶[0071]; STRATEGY-1003, ¶159.	results (e.g., from preloaded results 234), and so provides an apparatus for machine learning results. STRATEGY-1004, ¶[0071].
---	--

Petition at 25–26	Mitzenmacher Decl. ¶ 160
The Johnson-Lin combination provides a predictive compiler module configured to generate machine learning comprising program code for a plurality of learned functions. STRATEGY-1003, ¶160. As discussed above, a POSITA would have found it obvious to configure the cockpit control module 132 of Johnson’s forecasting system to include a module that trains accurate predictive models with machine learning algorithms and implements them as computer programs and so acts as a compiler. STRATEGY-1003, ¶¶160, 39.	The Johnson-Lin combination describes a predictive compiler module configured to generate machine learning including program code for a plurality of learned functions. As discussed above, a POSITA would have found it obvious to configure the cockpit control module 132 of Johnson’s forecasting system to include a module that trains accurate predictive models with machine learning algorithms and implements them as computer programs, so the configured system can create and host accurate models as Lin teaches.

Petition at 26	Mitzenmacher Decl. ¶ 161
Lin teaches using a “machine learning algorithm” to “train a predictive model by systematically analyzing the applied training dataset and generating an inferred function that ‘fits’ the training data.” STRATEGY-1005, 3:41-47. This use of a machine learning algorithm to learn an “inferred function fitted to a training dataset” demonstrates generating machine learning as described in the ’574 patent. <i>Id.</i> , 3:39-62; STRATEGY-1001, 6:48-52; STRATEGY-1003, ¶161.	For example, Lin teaches using a “machine learning algorithm” to “train a predictive model by systematically analyzing the applied training dataset and generating an inferred function that ‘fits’ the training data.” STRATEGY-1005, 3:41-47. The training of predictive models with a machine learning algorithm to learn an “inferred function fitted to a training dataset” demonstrates generating machine learning as described in the ’574 patent. <i>Id.</i> , 3:39-62; STRATEGY-1001, 6:48-52.

Petition at 26	Mitzenmacher Decl. ¶ 162
More specifically, Lin teaches the functionality of a predictive compiler through “computer program(s) [that] can include a number of instructions executable by a computer processor to perform one or more of the following operations: receive a training dataset stored in a suitable electronic data format; implement the machine learning algorithm to generate a predictive model that is representative of the training dataset; and output the predictive model in a suitable computer readable and executable format.” STRATEGY-1005, 3:65 to 4:5, 9:28-32; STRATEGY-1003, ¶¶162-163.	In more detail, Lin teaches the functionality of a predictive compiler through “computer program(s) [that] can include a number of instructions executable by a computer processor to perform one or more of the following operations: receive a training dataset stored in a suitable electronic data format; implement the machine learning algorithm to generate a predictive model that is representative of the training dataset; and output the predictive model in a suitable computer readable and executable format.” STRATEGY-1005, 3:65 to 4:5. Although Lin does not explicitly label this functionality as a compiler, a POSITA would have understood the functionality to act as a compiler, because translating modules or algorithms (e.g., predictive models in this case) into machine language that a computer can run is the well-known function and purpose of a compiler. STRATEGY-1021, 103; STRATEGY-1022, 93; STRATEGY-1023, 91.

Petition at 27	Mitzenmacher Decl. ¶ 164
Johnson describes using forecasting models 136, including models 136 having a “function fitted to empirical data,” but does not specifically identify how the models 136 are trained. STRATEGY-1004, ¶¶[0047], [0070]; STRATEGY-1003, ¶164. In view of Lin’s teachings, a POSITA would have found it obvious to enhance Johnson’s system with a module that creates and trains models with Lin’s techniques. STRATEGY-1005, 3:41-47; STRATEGY-1003, ¶164.	Johnson uses forecasting models 136, including models 136 having a “function fitted to empirical data,” but does not describe in detail how the models 136 are trained. STRATEGY-1004, ¶¶[0047], [0070]. In view of Lin’s teachings, a POSITA would have found it obvious to enhance Johnson’s system with a module that can create and train models with Lin’s technique of applying a machine learning algorithm to generate functions that fit training data. STRATEGY-1005, 3:41-47.

Petition at 27	Mitzenmacher Decl. ¶ 165
Johnson states that the “overarching aim” of its system is “to provide timely and accurate results” to the user. STRATEGY-1004, ¶[0071]. Lin encourages models that are “combined to form an ensemble” that “achieve[s] a more accurate prediction than can be provided by the individual models themselves.” STRATEGY-1005, 4:16-35; STRATEGY-1003, ¶165. Based on Lin, it would have been obvious for model training in the combined system to generate multiple functions and to combine them in ensembles, to improve accuracy and thus better achieve the “overarching aim” Johnson describes. STRATEGY-1005, 4:16-20; STRATEGY-1004, ¶[0071]; STRATEGY-1003, ¶165. This was also obvious from Johnson’s native support models having multiple functions. STRATEGY-1004, ¶[0047]; STRATEGY-1003, ¶165.	Johnson states that the “overarching aim” of its system is “to provide timely and accurate results” to the user. STRATEGY-1004, ¶[0071]. Lin teaches that it is advantageous for models to be “combined to form an ensemble,” because an ensemble can “achieve a more accurate prediction than can be provided by the individual models themselves.” STRATEGY-1005, 4:16-35. As noted above, Lin’s “ensemble include[s] a plurality of individual predictive models.” <i>Id.</i>, 5:9-10. Based on Lin, it would have been obvious for model training in Johnson’s system to generate multiple functions and to combine them in ensembles, to improve accuracy and thus better achieve the “overarching aim” of Johnson’s system. STRATEGY-1005, 4:16-20; STRATEGY-1004, ¶[0071]. Johnson’s system also natively supports models with multiple functions, explaining that models 136 can each use “plural transfer functions.” STRATEGY-1004, ¶[0047].

Petition at 27–28	Mitzenmacher Decl. ¶ 166
Johnson’s cockpit control module 132 “can be implemented as any kind of computer device,” including as “computer server coupled to a network.” STRATEGY-1004, ¶[0065]. Based on Lin, a POSITA would have found it obvious to configure Johnson’s cockpit control module 132 to include the functionality to implement models as computer programs, making trained model implementations executable by the cockpit control module 132, which itself is described as a “computer device.” STRATEGY-1004, ¶[0065]; STRATEGY-1005, 3:65-4:5, 9:28-	Johnson’s cockpit control module 132 “can be implemented as any kind of computer device,” including potentially as “computer server coupled to a network.” STRATEGY-1004, ¶[0065]. Lin teaches implementing predictive models in a “computer readable and executable format,” such as “one or more computer programs.” STRATEGY-1005, 3:65-4:5, 9:28-32. Lin also teaches functionality that can convert a model into a “model implementation 214” that “is one or more computer programs.” <i>Id.</i>, 9:15-38, 3:65-4:5. Based on Lin, a POSITA would have

32; STRATEGY-1003, ¶166. Implementing a model in executable format as “one or more computer programs” provides program code and the function of a compiler as described by the ’574 patent. STRATEGY-1001, 10:47-57, 19:13-33; STRATEGY-1003, ¶¶166, 39.	found it obvious to configure Johnson’s cockpit control module 132 to include the functionality to implement models as computer programs, so that trained model implementations are executable by the cockpit control module 132, which is described as a “computer device.” STRATEGY-1004, ¶[0065]. Implementing a model in executable format as “one or more computer programs” describes program code as described by the ’574 patent. STRATEGY-1001, 10:47-57, 19:13-33.
---	---

Petition at 29	Mitzenmacher Decl. ¶ 167
The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the predictive compiler module, e.g., algorithm steps of (i) receiving training data (STRATEGY-1005, 10:44-56, 9:64-10:5, FIGS. 3-4), (ii) generating learned functions from the training data (<i>id.</i>, 3:39-62), and (iii) forming a machine learning ensemble from the learned functions (<i>id.</i>, 4:16-20, 5:8-22), where the output is executable program code (<i>id.</i>, 3:65-4:5, 9:28-32). STRATEGY-1003, ¶167. The Johnson-Lin system performs the claimed function in substantially the same way described by the ’574 patent to achieve substantially the same result, and thus satisfies the function-way-result test for equivalence. STRATEGY-1003, ¶168.	The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the predictive compiler module. For example, the predictive compiler module shown above in the system of the Johnson-Lin combination performs the algorithm steps of (i) receiving training data (STRATEGY-1005, 1:17 to 2:34, 3:65-4:5, 10:44-56, 9:64-10:5, FIGS. 3-4), (ii) generating learned functions from the training data (<i>id.</i>, 3:39-62, 6:14 to 7:27), and (iii) forming a machine learning ensemble from the learned functions (<i>id.</i>, 4:16-35, 5:8-22, 16:5-14 (claim 8), 17:10-14 (claim14)). Like the predictive compiler module of the ’574 patent, the output of these steps in the Johnson-Lin combination is executable program code that implements the trained machine learning models. <i>Id.</i>, 3:65-4:5, 9:28-32.

Petition at 33	Mitzenmacher Decl. ¶ 176
The interactive display 266 receives user input values for the “one or more X variables,” and the user enters those input values by interacting with knobs, sliders, or other controls on a graphical user interface. STRATEGY-1004, ¶¶[0070], [0082]-[0083], [0118], [0105] (“the cockpit user 138 can experiment with different permutations of these actionable X variables by independently adjusting the settings on these five input knobs”), [0112]-[0117] (describing a user adjusting input values for machine learning parameters), [0149], [0172], [0179], [0184]; STRATEGY-1003, ¶176.	The interactive display 266 receives user input values for the “one or more X variables,” and the user enters those input values by interacting with knobs, sliders, or other controls on a graphical user interface. <i>Id.</i> , ¶¶[0070], [0082]-[0083], [0118]. Furthermore, Johnson teaches that “the cockpit user 138 can experiment with different permutations of these actionable X variables by independently adjusting the settings on these five input knobs.” <i>Id.</i> , ¶¶[0105]; [0112]-[0117] (describing a user adjusting input values for machine learning parameters), [0149], [0172], [0179], [0184].

Petition at 33	Mitzenmacher Decl. ¶ 177
Johnson’s FIG. 3 shows an example of these controls in a “window 316 that provides various control mechanisms,” with knobs or dials, slider bars, data entry boxes, and other graphical input mechanisms. STRATEGY-1004, ¶¶[0082]-[0083], [0066], [0102], FIG. 3; STRATEGY-1003, ¶177.	Johnson’s FIG. 3 shows an example of these controls, with the cockpit interface 134 having a “window 316 that provides various control mechanisms,” with knobs or dials, slider bars, data entry boxes, and other graphical input mechanisms. <i>Id.</i> , ¶¶[0082], [0102], FIG. 3. The “input mechanisms (318, 320, 322, 324) provided in the window [316] can be used to input various what-if assumptions” that set the parameter values for requesting a forecast. <i>Id.</i> , ¶[0083], [0066].

Petition at 34	Mitzenmacher Decl. ¶ 178
The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the input module, e.g., a user interface that receives user input (STRATEGY-1004, FIGS. 3, 13, ¶¶[0070], [0082]-[0083], [0118],	The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the input module, e.g., a user interface that receives user input (STRATEGY-1004, FIGS. 3, 13, ¶¶[0070], [0081]-[0083], [0105], [0118])

[0105]) through user manipulation of a slider (<i>id.</i> , ¶¶[0082], [0102], FIG. 3) or another displayed object (<i>id.</i> , ¶¶[0075], [0082], [0105] (e.g., knobs, dials, switches, etc.)). ⁷ STRATEGY-1003, ¶¶178-179.	through user manipulation of a slider (<i>id.</i> , FIG. 3, ¶¶[0082], [0102]) or another displayed object (<i>id.</i> , ¶¶[0075] (e.g., knobs, switches, etc.), [0082] (e.g., knobs, dials, “graphical input devices 324 (such as data entry boxes, radio buttons, etc.)”), [0105] (“The use of five knobs is merely illustrative. In other implementations, other kinds of input mechanisms can be used besides knobs.”)).
--	---

Petition at 35	Mitzenmacher Decl. ¶ 180
The Johnson-Lin combination provides a pre-compute module with “ pre-loading logic 230 [that] performs analysis in advance of a user’s request. ” STRATEGY-1004, ¶[0068], FIG. 2, FIG. 10; see <i>id.</i> , Title, Abstract, Claim 22, ¶¶[0153], [0164], [0165]-[0173] (discussing pre-calculation processes performed); STRATEGY-1003, ¶¶180-181.	The Johnson-Lin combination describes a pre-compute module configured to predetermine, using the generated machine learning, permutations of the machine learning results at one or more increments between a minimum value and a maximum value for the one or more machine learning parameters, prior to the input module receiving the user input.

Petition at 35	Mitzenmacher Decl. ¶ 181
The results calculated by the pre-loading logic 230 are machine learning results predetermined using generated machine learning, e.g., predictions or forecasts from the programs that implement models trained using a machine learning algorithm based on Lin. STRATEGY-1004, ¶¶[0066] (analysis logic 222 generates predictions and forecasts as results), [0068] (preloading logic 230 generates the results in advance), [0070]-[0071], [0098], [0153]; STRATEGY-1005, 3:41-47; STRATEGY-1003, ¶181. An illustration based on Johnson’s FIG. 2 (below) shows the pre-loading logic 230 as a pre-compute module.	In the Johnson-Lin combination, as Johnson describes, a “ pre-loading logic 230 performs analysis in advance of a user’s request ” and thus acts as a precompute module. STRATEGY-1004, ¶[0068], FIG. 2, FIG. 10; see <i>id.</i> , Title, Abstract, Claim 22, ¶¶[0153], [0164], [0165]-[0173] (discussing pre-calculation processes performed). The results calculated by the pre-loading logic 230 are machine learning results, e.g., predictions or forecasts from the models trained using a machine learning algorithm, which Lin describes. STRATEGY-1004, ¶¶[0066] (analysis logic 222 generates predictions and forecasts as results), [0068] (preloading logic 230 generates the results

	in advance), [0070]-[0071], [0098], [0153]; STRATEGY-1005, 3:41-47.
--	---

Petition at 37–38	Mitzenmacher Decl. ¶ 182
Using the pre-loading logic 230, “pre-calculation can be performed for specified permutations of input assumptions (e.g., specific combinations of input X variables),” and consequently different input permutations create different permutations of results. STRATEGY-1004, ¶[0153]. Johnson also shows this in FIG. 9 (below), where pre-calculated results for different combinations of input values form a “response surface 900” representing various scenarios. STRATEGY-1004, ¶¶[0155], [0159], FIG. 9; STRATEGY-1003, ¶182. The graph shows calculated results along the vertical “Y” axis, resulting from (1) variation in the value of input for an “actionable X variable” (a machine learning parameter) along an “X” axis and (2) “different slices of time” (another machine learning parameter) along a third axis. STRATEGY-1004, ¶[0155], FIG. 9; STRATEGY-1003, ¶182.	Using the pre-loading logic 230, “pre-calculation can be performed for specified permutations of input assumptions (e.g., specific combinations of input X variables),” and consequently different input permutations create different permutations of results. STRATEGY-1004, ¶[0153]. Johnson, in FIG. 9, shows how the pre-calculated results for different combinations of input values allow a predicted “response surface 900” to be characterized. STRATEGY-1004, ¶¶[0155], [0159], FIG. 9. The graph represents calculated results along the vertical “Y” axis, resulting from variation in the value of input for (1) an “actionable X variable” (a machine learning parameter) along an “X” axis and (2) “different slices of time” (another machine learning parameter) along a third axis. STRATEGY-1004, ¶[0155], FIG. 9.

Petition at 39	Mitzenmacher Decl. ¶ 183
As shown in Johnson’s FIG. 9 above, the pre-loading logic 230 pre-calculates results at various increments over a range of input values, e.g., between a minimum value and maximum value for the “X” input variable and for the time variable. STRATEGY-1004, ¶¶[0151], [0159]; STRATEGY-1003, ¶183. Pre-loading results involves calculation by “sequencing through a great number of permutations of actionable X variables” that serve as input variables (e.g., machine learning parameters). STRATEGY-1004, ¶¶[0151], [0153] (computing results for “specified permutations of input assumptions (e.g., specific combinations of input X variables)”). Johnson explains that the preloading logic 230 can select points to pre-calculate “in an orderly way,” such as “at regular intervals,” which involves and renders obvious varying the values of input variables (machine learning parameters) at one or more increments. STRATEGY-1004, ¶[0159]. STRATEGY-1003, ¶183.	As shown in Johnson’s FIG. 9 above, the pre-loading logic 230 pre-calculates results at various increments over a range of input values, e.g., between a minimum value and maximum value for the “X” input variable and for the time variable. STRATEGY-1004, ¶¶[0151], [0159]. Pre-loading results involves calculation by “sequencing through a great number of permutations of actionable X variables” that serve as input variables (e.g., machine learning parameters). <i>Id.</i>, ¶¶[0151], [0153] (computing results for “specified permutations of input assumptions (e.g., specific combinations of input X variables)”). Johnson explains that “the preloading logic 230 can probe the response surface 900 in an orderly way, for instance, by selecting sample points for investigation at regular intervals within the response surface 900.” STRATEGY-1004, ¶[0159]. Obtaining output results at “regular intervals” involves and renders obvious varying the values of input variables at regular intervals, which shows that results are obtained “at one or more increments” for the machine learning parameters, as recited in claim 1.

Petition at 39	Mitzenmacher Decl. ¶ 184
The pre-loading logic 230 performs calculations for a defined “scope” or range of values for input variable, including using existing “empirical business information” to determine “ranges” for the input variables. STRATEGY-1004, ¶[0160]-[0161] (ranges determined by “an automated routine” or “human business analysts”); STRATEGY-1003, ¶184.	The pre-loading logic 230 performs calculations for a defined “scope” or range of values for input variables, including using existing “empirical business information” to describe “guidance on the kinds of ranges of input variables that are typically used in exploring the behavior of the particular business sector.” STRATEGY-1004, ¶[0160]. The determination of which calculations to perform, and thus which range of input values to use, can be made by “an automated routine” or “human business analysts.” <i>Id.</i>, ¶[0160]. Similarly, Johnson describes that the scope of pre-calculation, and thus the range of input values, can be based on a “pre-analysis on the response surface” to select regions for pre-calculation that include a “desired output result.” <i>Id.</i>, ¶[0161].

Petition at 40	Mitzenmacher Decl. ¶ 185
The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the pre-compute module, e.g., algorithm steps of (i) computing multiple results over a range before the results are requested by a user (STRATEGY-1004, ¶¶[0151]-[0161]), and (ii) caching the results indexed by the inputs (<i>id.</i>, ¶¶[0066], [0071], [0124], [0167], [0170], FIGS. 2, 10) in a results data structure (<i>id.</i>, ¶¶[0071], [0153], [0166], FIGS. 2, 10).⁸ STRATEGY-1003, ¶¶185-186.	The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the pre-compute module, e.g., algorithm steps of (i) computing multiple results over a range before the results are requested by a user (STRATEGY-1004, ¶¶[0151]-[0161]), and (ii) caching the results indexed by the inputs (<i>id.</i>, ¶¶[0066], [0071], [0124], [0167], [0170], FIGS. 2, 10) in a results data structure (<i>id.</i>, ¶¶ [0063], [0066], [0071], [0153], [0166], FIGS. 2, 10).

Petition at 40–41	Mitzenmacher Decl. ¶¶ 187–188
The Johnson-Lin combination provides a display module that includes Johnson’s “display presentation logic 236,” which manages display of output results of the analyses performed by the analysis logic 222, which includes the pre-loading logic 230 (e.g., precompute module). STRATEGY-1004, ¶¶[0072], [0068]; STRATEGY-1003, ¶¶187-188. The display presentation logic 236 generates various views of pre-calculated results displayed on the cockpit interface 134. STRATEGY-1004, ¶¶[0072] (noting “Section E” showing functions performed by the display presentation logic 236), [0174]-[0207] (“Section E” discussing display functionality), [0083], [0086], FIG. 3 (showing results in elements 308, 310), FIGS. 12-18 (showing various displays of results). Johnson’s FIG. 2 (modified version below) shows the display module and its interactions to display results.	The Johnson-Lin combination describes a display module configured to display, from the pre-compute module, a first predetermined permutation of the one or more predicted machine learning results for the one or more identified machine learning parameters. In the Johnson-Lin combination, “display presentation logic 236” acts as a display module, and the display presentation logic 236 performs various tasks associated with displaying the output results of the analyses performed by the analysis logic 222,” which includes the pre-loading logic 230 (e.g., precompute module). STRATEGY-1004, ¶¶[0072], [0068]. The display presentation logic 236 generates various views of previously determined results shown in the digital cockpit 104. <i>Id.</i>, ¶¶[0072] (noting “Section E” showing functions performed by the display presentation logic 236), [0174]-[0207] (“Section E” discussing display functionality), [0083], [0086]; FIG. 3 (showing results in elements 308, 310), Figs. 12-18 (showing various displays of results).

Petition at 41–42	Mitzenmacher Decl. ¶ 189
When a user provides input, the results displayed in response are pre-calculated results retrieved from a database 234 of pre-loaded results retrieved by the pre-loading logic 230. STRATEGY-1004, ¶[0071]; <i>see also id.</i> ¶¶[0153], [0164], [0166]; STRATEGY-1003, ¶189. Thus, the results are “from” the pre-loading logic 230 (e.g., precompute module) for being generated by and retrieved by the pre-loading logic 230. STRATEGY-1004, ¶¶[0153], [0164], [0166], [0170]; STRATEGY-1003, ¶189.	In the digital cockpit 104, the displayed results were previously calculated by the pre-loading logic 230, and thus are from the precompute module. STRATEGY-1004, ¶[0071]; <i>see also id.</i> ¶¶[0153], [0164], [0166]. The results generated by the pre-loading logic 230 are stored in a database 234 of pre-loaded results, so the “digital cockpit 104 can retrieve results from this database when the user requests these results, instead of calculating these results at the time of request.” <i>Id.</i> The displayed results are also “from” the pre-loading logic 230 because the pre-loading logic 230 retrieves and describes stored results for display. <i>Id.</i> , ¶¶[0153], [0170]

Petition at 42	Mitzenmacher Decl. ¶ 190
As an example, the cockpit interface 134 shows predicted results, arranged by the display presentation logic 236 to display results from the preloading logic 230 in windows 308, 310 of Johnson’s FIG. 3. STRATEGY-1004, [0080], FIG. 3, ¶¶[0068], [0153]; <i>see also id.</i> , FIGS. 12-18 (showing additional display examples); STRATEGY-1003, ¶190.	As an example, the digital cockpit 104 shows predicted results, arranged by the display presentation logic 236 and presenting previously-determined results from the preloading logic 230, in windows 308, 310 of Johnson’s FIG. 3. STRATEGY-1004, FIG. 3, ¶¶[0071], [0080], [0068], [0153], [0166]–[0167]; <i>see also id.</i> , Figs. 12-18 (showing additional display examples).

Petition at 43–44	Mitzenmacher Decl. ¶ 191
The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the display module, e.g., a user interface (STRATEGY-1004, FIGS. 3, 12-18, ¶¶[0081]–[0083]) that displays information about precomputed results (<i>id.</i> , ¶¶[0071], [0153], [0166]), such as machine learning parameters and machine learning results (<i>id.</i> , FIGS. 3, 12-	The Johnson-Lin combination provides and renders obvious the same structure that the ’574 patent teaches for the display module, e.g., a user interface (STRATEGY-1004, FIGS. 3, 12-18, ¶¶[0081]–[0083]) that displays information about precomputed results (<i>id.</i> , ¶¶[0071], [0153], [0166]), such as machine learning parameters and machine learning results (<i>id.</i> , FIGS. 3, 12-

18, ¶¶[0081]-[0083], [0178]-[0185], [0202]-[0203]), and includes a graphical table (<i>id.</i> , ¶[0083] (“spread sheet format”), a slider (<i>id.</i> , ¶¶[0075] (“slide bars”), [0082]), or another displayed object (<i>id.</i> , FIG. 3, ¶¶[0082], [0101], [0105]). ⁹ STRATEGY-1003, ¶¶191-192.	18, ¶¶[0081]-[0083], [0178]-[0185], [0202]-[0203]), and includes a graphical table (<i>id.</i> , ¶[0083] (“spread sheet format”), a slider (<i>id.</i> , ¶¶[0075] (“slide bars”), [0082]), or another displayed object (<i>id.</i> , FIG. 3, ¶¶[0082], [0101], [0105]).
---	--

Petition at 47–48	Mitzenmacher Decl. ¶ 199
With respect to updating the display, Johnson states broadly that the presentation logic 236 handles “tasks associated with displaying the output results.” STRATEGY-1004, ¶¶[0072] (noting logic 236 handles functionality of Section E), [0174]-[0207] (Section E, discussing user interaction and display functionality); STRATEGY-1003, ¶199. When the interactive display 266 receives additional user input (e.g., additional values for parameters), the digital cockpit interface can “dynamically display changes to the response surface 312 while the cockpit user 138 varies one or more input mechanisms (318, 320, 322, 324),” with “real-time coupling” between user inputs and displayed outputs. STRATEGY-1004, ¶¶[0102], [0118], [0179]. As Johnson describes, “the cockpit user 138 adjusts the input devices (318, 320, 322, 324) to select a particular permutation of actionable X variables. The digital cockpit 104 responds by simulating how the business 102 would react to this combination of input actionable X variables....” STRATEGY-1004, ¶[0101]; see <i>also id.</i>, ¶¶[0089], [0105], [0117]-[0118], [0167]; STRATEGY-1003, ¶199.	Regarding updating of the display, Johnson describes that “the cockpit user 138 adjusts the input devices (318, 320, 322, 324) to select a particular permutation of actionable X variables. Johnson’s digital cockpit 104 responds by simulating how the business 102 would react to this combination of input actionable X variables as if these actionable X variables were actually implemented within the business 102.” STRATEGY-1004, ¶[0101]; see <i>also id.</i>, ¶¶[0105], [0118], [0167]. Moreover, “the digital cockpit 104 can be configured to dynamically display changes to the response surface 312 while the cockpit user 138 varies one or more input mechanisms (318, 320, 322, 324). The real-time coupling between actuations made in the control window 316 and changes presented to the response surface 312 allows the cockpit user 138 to gain a better understanding of the characteristics of the response surface 312.” STRATEGY-1004, ¶¶[0102], [0118], [0179]

Petition at 48–49	Mitzenmacher Decl. ¶ 200
When the user enters additional user input to change the simulation scenario, “if the digital cockpit 104 determines that the results have been pre-calculated, then the digital cockpit 104 retrieves and supplies those results to the cockpit user 138 (in step 1020).” STRATEGY-1004, ¶[0170], FIG. 10, ¶¶[0167]-[0168] (discussing identification of requested results). Thus, the update to the displayed results presents a second permutation of results from the preloading module 230 (e.g., precompute module). <i>Id.</i>, ¶[0170] (noting that retrieval can be done by the preloading module 230), FIG. 10; STRATEGY-1003, ¶200.	When the user enters additional values to change the scenario for simulation, “if the digital cockpit 104 determines that the results have been pre-calculated, then the digital cockpit 104 retrieves and supplies those results to the cockpit user 138 (in step 1020).” <i>Id.</i>, ¶[0170], FIG. 10, ¶¶[0167]-[0168] (discussing identification of requested results). Thus, the update to the displayed results can present a second permutation of results from the preloading module 230 (e.g., precompute module). <i>Id.</i>, ¶[0170] (noting that retrieval can be done by the preloading module 230), FIG. 10.

Petition at 49–50	Mitzenmacher Decl. ¶ 201
Thus, a user adjustment to controls in the control window 316 of the interactive display 266 (e.g., input module) identifies additional values for the actionable X variables (e.g., machine learning parameters). STRATEGY-1004, ¶¶[0101], [0105], [0117]-[0118]; STRATEGY-1003, ¶201. In response to the changed values for the X variables, the system will “dynamically display” a changed set of results (e.g., second permutation) for the new user-specified scenario. <i>Id.</i>, ¶¶[0102], [0179]. The updated results that the display presentation logic 236 causes to be displayed are from the preloading logic 230 (e.g., precompute module), retrieved from pre-calculated results cached in the database 234. <i>Id.</i>, FIG. 2 (element 234), FIG. 10 (element 1020), ¶¶[0071], [0153], [0166]-[0167].	Thus, a user adjustment to the knobs or sliders in the control window 316 (e.g., input module) identifies additional values for the actionable X variables (e.g., machine learning parameters). STRATEGY-1004, ¶¶[0101], [0105], [0117]-[0118]. In response to the changed values for the X variables, the system will “dynamically display” a changed set of results (e.g., second permutation) for the new scenario the user indicated. <i>Id.</i>, ¶¶[0102], [0179]. The updated results are from the preloading logic 230 (precompute module), retrieved from pre-calculated results cached in the database 234. <i>Id.</i>, FIG. 2 (database 234), FIG. 10 (element 1020), ¶¶[0071], [0153], [0166]-[0167].

Petition at 50	Mitzenmacher Decl. ¶ 202
The Johnson-Lin combination provides and renders obvious the same structure that the '574 patent teaches for the update module, e.g., algorithm steps of (i) detecting user input (STRATEGY-1004, ¶¶[0167]-[0168], [0101]-[0102]), (ii) looking up information that corresponds to the user input in a data structure (<i>id.</i>, FIG. 10, ¶¶[0153], [0167]-[0168], [0170]), and (iii) updating information displayed by the display module (<i>id.</i>, ¶¶[0102], [0085], [0179]). STRATEGY-1003, ¶¶202-203.	The Johnson-Lin combination provides and renders obvious the same structure that the '574 patent teaches for the update module, e.g., algorithm steps of (i) detecting user input (STRATEGY-1004, ¶¶[0167]-[0168], [0101]-[0102]), (ii) looking up information that corresponds to the user input in a data structure (<i>id.</i>, FIG. 10, ¶¶[0152]-[0153], [0167]-[0168], [0170]), and (iii) updating information displayed by the display module (<i>id.</i>, ¶¶[0102], [0085], [0179], [0197], [0203]).

Petition at 53	Mitzenmacher Decl. ¶ 204
In the Johnson-Lin combination, the cockpit control module 132 includes the predictive compiler module, the input module, the pre-compute module, the display module, and the update module. STRATEGY-1003, ¶204. The cockpit control module 132 includes one or more of logic hardware and a non-transitory computer readable storage medium storing program code executable by a processor. STRATEGY-1004, ¶¶[0065]-[0066]; STRATEGY-1005, 12:38-14:25; STRATEGY-1003, ¶204. Two examples of obvious arrangements providing these modules are shown below.	In the Johnson-Lin combination, the cockpit control module 132 includes the predictive compiler module, the input module, the pre-compute module, the display module, and the update module. The cockpit control module 132 has one or more of logic hardware and a non-transitory computer readable storage medium storing program code executable by a processor. STRATEGY-1004, ¶¶[0065]-[0066]; STRATEGY-1005, 12:38-14:25

Petition at 53–54	Mitzenmacher Decl. ¶ 205
As Johnson describes, the cockpit control module 132 “can be implemented as any kind of computer device, including one or more processors 210, various memory media (such as RAM, ROM, disc storage, etc.),” and “other computer architecture features that are known in the art.” STRATEGY-1004, ¶[0065]. The memory media stores “application logic 218” that can include “different modules of program instructions.” <i>Id.</i>, ¶[0066], Claim 40 (“computer-readable medium including instructions for carrying out the control module logic”). Johnson and Lin thus provide and render obvious implementation of the modules as logic hardware and/or stored program code. STRATEGY-1005, 12:3 to 14:44; STRATEGY-1003, ¶205.	As Johnson describes, the cockpit control module 132 “can be implemented as any kind of computer device, including one or more processors 210, various memory media (such as RAM, ROM, disc storage, etc.),” and “other computer architecture features that are known in the art.” STRATEGY-1004, ¶[0065]. The memory media stores “application logic 218” that can include “different modules of program instructions.” <i>Id.</i>, ¶[0066], Claim 40 (“computer-readable medium including instructions for carrying out the control module logic”). Implementing the modules in computer hardware and/or a computer-readable medium storing program code was also obvious in view of Lin’s disclosure of computer-implemented features. STRATEGY-1005, 12:3 to 14:44. Accordingly, the Johnson-Lin combination provides and renders obvious implementation of the various modules as logic hardware and/or a non-transitory computer readable storage medium storing program code executable by a processor.

Petition at 54	Mitzenmacher Decl. ¶ 206
In the Johnson-Lin combination, the preloading logic 230 (e.g., precompute module) is configured to cache the predetermined permutations of the machine learning results. STRATEGY-1004, FIG. 2 (database 234, labeled “Cache of Pre-stored Analytical Results”), FIG. 10 (database 234), ¶¶[0071], [0094], [0153], [0164], [0166]–[0167]; STRATEGY-1003, ¶206. The results are stored for the update module, for later use in response to a user providing an input scenario that calls for	In the Johnson-Lin combination, the preloading logic 230 (e.g., precompute module) is configured to cache the predetermined permutations of the machine learning results. STRATEGY-1004, FIG. 2 (database 234, labeled “Cache of Pre-stored Analytical Results”), FIG. 10 (database 234), ¶¶[0071], [0094], [0153], [0164], [0166]–[0167]. The results are stored for the update module, so they can be later used and described when a user describes an input scenario that calls for

those results. EX1004, ¶¶[0170] (display is updated from stored results), [0166]-[0167], Abstract; STRATEGY-1003, ¶206.	those results. STRATEGY-1004, ¶¶[0170] (display is updated from stored results), [0166]-[0167], Abstract.
---	---

Petition at 55–56	Mitzenmacher Decl. ¶ 207
The preloading logic 230 stores results in a results data structure indexed by the one or more machine learning parameters. STRATEGY-1004, ¶[0066]; STRATEGY-1003, ¶207. In the Johnson-Lin combination, the record storage 220, which includes database 234 storing cached results, is for “storing different groups of records using appropriate data structures.” STRATEGY-1004, ¶[0066]. Storing results involves “codifying the output results for storage to enable the output results to be retrieved at a later point in time.” <i>Id.</i>, ¶[0124]. The structure of the database 234 links results to a corresponding set of input assumptions for X variables (e.g., machine learning parameters), and accordingly a set of conditions defining these input assumptions “are used as a key to search the database 234” for results from “a prior analysis session” or for “retrieving pre-calculated results.” STRATEGY-1004, ¶¶[0167], [0070] (noting input assumptions refer to values for X variables); STRATEGY-1003, ¶207.	The preloading logic 230 stores results in a results data structure indexed by the one or more machine learning parameters. In the Johnson-Lin combination, the record storage 220, which includes database 234 storing cached results, is for “storing different groups of records using appropriate data structures.” <i>Id.</i>, ¶[0066]. Storing results involves “codifying the output results for storage to enable the output results to be retrieved at a later point in time.” <i>Id.</i>, ¶[0124]. The structure of the database 234 links results to a corresponding set of input assumptions for X variables (e.g., machine learning parameters), so that a set of conditions defining these input assumptions “are used as a key to search the database 234” for results from “a prior analysis session” or for “retrieving pre-calculated results.” <i>Id.</i>, ¶¶[0167], [0070] (noting input assumptions refer to values for X variables). The fact that the values for the input assumptions “are used as a key” indicates that the results are indexed by those input values to facilitate lookup by the input values.

Petition at 56	Mitzenmacher Decl. ¶ 208
Storing results indexed to the machine learning parameters was also obvious based on Johnson’s use of indexing as a general storage technique, noting that the architecture can use “multi-dimensional data structures” in which “dimensions serve as indices for retrieving information from a multi-dimensional array of information.” STRATEGY-1004, ¶[0063]; STRATEGY-1003, ¶208	Storing results indexed to the machine learning parameters was also obvious based on Johnson’s use of indexing as a general storage technique. <i>Id.</i>, ¶[0063]. Johnson notes that the architecture can use “multi-dimensional data structures” in which “dimensions serve as indices for retrieving information from a multi-dimensional array of information.” <i>Id.</i> Although Johnson does not refer to the database 234 specifically when mentioning these multi-dimensional data structures, a POSITA would have recognized that these common database techniques discussed for the architecture 200 are broadly applicable and are obvious to apply to the database 234.

Petition at 56	Mitzenmacher Decl. ¶ 209
<i>Supra</i> element [1.5], Claim 2. In response to user input, the additional input values “are used as a key to search the database 234” to determine the corresponding second permutation of results. STRATEGY-1004, ¶[0167], FIG. 10; STRATEGY-1003, ¶209. This is performed by the update module, which includes the preloading logic 230 that looks up results from the database 234. STRATEGY-1004, ¶[0170]; STRATEGY-1003, ¶209.	<i>Supra</i> Section XI.A.[1.5], Claim 2. In response to user input, the additional input values “are used as a key to search the database 234” to determine the corresponding second permutation of results. STRATEGY-1004, ¶[0167], FIG. 10. Johnson’s update module performs this task, which includes the preloading logic 230 that can look up results from the database 234. STRATEGY-1004, ¶[0170].

Petition at 57	Mitzenmacher Decl. ¶ 213
In the Johnson-Lin combination, the machine learning parameters include Johnson’s “X variables,” which affect results and are adjusted by user input. STRATEGY-1004, ¶¶[0047], [0070], [0100]-[0102], [0105], [0113]-[0117]; STRATEGY-1003, ¶¶213-219. These X variables provide and render obvious: • an attribute of a feature, since values and ranges of input variables are attributes according to the ’574 patent (STRATEGY-1004, ¶¶[0047], [0070], [0100]-[0102] (discussing user input setting X variable values), [0061], [0160] (discussing range of input variables); STRATEGY-1001, 12:37-57, 14:28-51, 8:5-13);	The Johnson-Lin combination provides and renders obvious machine learning parameters that comprise one or more of an attribute of a feature, a target value for a goal, an action relating to a goal, a confidence metric, and a target of an action. In particular, the machine learning parameters include Johnson’s “X variables.” The values of these X variables affect the results that are calculated and displayed, and the values of the X variables and are adjusted by user input. STRATEGY-1004, ¶¶[0047], [0070], [0100]-[0102], [0105], [0113]-[0117]. These X variables provide and render obvious the machine learning parameters including an attribute of a feature, since values and ranges of input variables are attributes according to the ’574 patent (STRATEGY-1004, ¶¶[0047], [0070], [0100]-[0102] (discussing user input setting X variable values), [0061], [0160] (discussing range of input variables); STRATEGY-1001, 12:37-57, 14:28-51, 8:5-13).

Petition at 57–58	Mitzenmacher Decl. ¶ 214
• an action relating to a goal, where the variable is “actionable” and “corresponds to an aspect of the business 102 that the business 102 can deliberately manipulate” to achieve a goal (STRATEGY-1004, ¶¶[0100], [0105], [0113]-[0117], [0067] (discussing user’s “predefined goal”); and	The “X variables” described by Johnson also provide and render obvious the machine learning parameters including an action relating to a goal, where the variable is “actionable” because it “corresponds to an aspect of the business 102 that the business 102 can deliberately manipulate” to achieve a goal (STRATEGY-1004, ¶¶[0100], [0105], [0113]-[0117], [0067] (discussing user’s “predefined goal”).

Petition at 58	Mitzenmacher Decl. ¶ 215
• a target of an action, since X variables can represent the “target resources” that will receive a command or adjustment. STRATEGY-1004, ¶¶[0056], [0090], [0089] (noting “desired target” of adjustments). STRATEGY-1003, ¶¶217-219.	The “X variables” described by Johnson also provide and render obvious the machine learning parameters including a target of an action, since X variables can represent the “target resources” that will receive a command or adjustment. STRATEGY-1004, ¶¶[0056], [0090], [0089] (noting “desired target” of adjustments).

Petition at 58	Mitzenmacher Decl. ¶ 216
It was obvious for the machine learning parameters to include a target value for a goal, such as a “predefined goal” or “desired target results” based on Johnson. STRATEGY-1004, ¶¶[0067], [0092], [0117], [0161] (human analyst designating desirable outcomes); STRATEGY-1003, ¶220. It was also obvious to set a confidence metric, since Johnson explains that “desired target results” include achieving “suitable confidence” in the results. STRATEGY-1004, ¶¶[0105]-[0106], [0117], [0161]; STRATEGY-1003, ¶221.	Based on Johnson, it also would have been obvious to a POSITA for the machine learning parameters to include a target value for a goal, such as a “predefined goal” or “desired target results” as Johnson describes. STRATEGY-1004, ¶¶[0067], [0092], [0117], [0161] (human analyst designating desirable outcomes). It also would have been obvious for the machine learning parameter to include a confidence metric, since Johnson explains that “desired target results” include achieving “suitable confidence” in the results. STRATEGY-1004, ¶¶[0105]-[0106], [0117], [0161].

Petition at 59–60	Mitzenmacher Decl. ¶ 224
The input module receives adjustment to a slider graphical user interface element corresponding to the one or more machine learning parameters, as shown in FIG. 3 below. STRATEGY-1004, FIG. 3, ¶¶[0075] (“cockpit interface 134 can include various graphical knobs, slide bars, switches, etc. for receiving the user's commands”), [0082] (control window 316 includes “a collection of graphical input slider bars 320”), [0083], [0167], [0179] (control	The input module receives adjustment to a slider graphical user interface element corresponding to the one or more machine learning parameters. Johnson discloses that a “cockpit interface 134 can include various graphical knobs, slide bars, switches, etc. for receiving the user's commands.” STRATEGY-1004, ¶[0075]; see also <i>id.</i>, ¶[0082] (control window 316 includes “a collection of graphical input slider bars 320”), FIG. 3, ¶¶[0083], [0167],

window 316 controls vary values for “X variables” (machine learning parameters)); STRATEGY-1003, ¶224.	[0179] (control window 316 controls vary values for “X variables” (machine learning parameters)).
---	--

Petition at 60–61	Mitzenmacher Decl. ¶ 225
In the Johnson-Lin combination, the interactive display 266 (e.g., input module) is configured to receive user input as a user manipulation of a displayed object, such as a knob, slider, switch, data entry box, etc. STRATEGY-1004, ¶¶[0082], [0101], [0105], FIG. 3 (showing objects in window 316), FIGS. 12, 13, 18; STRATEGY-1003, ¶225.	In the Johnson-Lin combination, the interactive display 266 (e.g., input module) configured to receive user input as a user manipulation of a displayed object, such as a knob, slider, switch, data entry box, etc. STRATEGY-1004, ¶¶[0082], [0101], [0105], FIG. 3 (showing objects in window 316), Figs. 12, 13, 18.

Petition at 69–70	Mitzenmacher Decl. ¶ 227
In the Johnson-Lin-Purcell combination, user input to the knob panel 1312 adjusts input values for X variables (input to machine learning) to produce a predicted Y-variable output value (output of machine learning) as a result based on the input values. STRATEGY-1004, ¶¶[0101], [0105], [0167]-[0170]; STRATEGY-1003, ¶227; see Sections III.B.2 (above) and Element [16.5] (below)	In the Johnson-Lin-Purcell combination, user input to the knob panel 1312 can adjust input values for X variables (input to machine learning) to produce a predicted Y-variable output value (output of machine learning) as a result based on the input values. STRATEGY-1004, ¶¶[0101], [0105], [0167]-[0170].

Petition at 70	Mitzenmacher Decl. ¶ 228
In addition, user input to the vertical slider adjusts an output “goal value” for the Y variable (output of machine learning) to produce corresponding X-variable input values (input to machine learning) as a result based on the goal value, which shows the function inputs that would achieve the user-specified goal value. STRATEGY-1006, 5:29-55, 37:1-21; STRATEGY-1003, ¶228.	In addition, user input to the vertical slider can adjust an output “goal value” for the Y variable (output of machine learning) to produce corresponding X-variable input values (input to machine learning) as a result based on the goal value, which shows the function inputs that would achieve the user-specified goal value. STRATEGY-1006, 5:29-55, 37:1-21. See

	Sections X.G (above) and XI.B.[16.5] (below).
--	---

Petition at 71	Mitzenmacher Decl. ¶ 229
In the Johnson-Lin-Purcell combination, the display presentation logic 236 provides the cockpit interface 134 with the interactive display 266 (e.g., display module) that can present output results in “ spread sheet format, ” and is configured to display results as entries in a graphical table. STRATEGY-1004, ¶[0083]; STRATEGY-1003, ¶229. Based on Purcell, the spreadsheet interface also receives user input, allowing users can “see and test changes” to input values that define “what-if” scenarios and see the corresponding result values. STRATEGY-1006, 14:47-57, 2:1-21, 3:5-30; Section III.B.2; STRATEGY-1003, ¶229.	In the Johnson-Lin-Purcell combination, the display presentation logic 236 describes the cockpit interface 134 with the interactive display (e.g., display module) that can present output results in “ spread sheet format, ” and thus is configured to display results as entries in a graphical table. STRATEGY-1004, ¶[0083]. The spreadsheet interface also receives user input, allowing users can “see and test changes” to input values that define “what-if” scenarios and see the corresponding result values. STRATEGY-1006, 14:47-57, 2:1-21, 3:5-30; see Section X.E.2.

Petition at 71	Mitzenmacher Decl. ¶ 230
To the extent the preamble is limiting, the Johnson-Lin-Purcell combination provides this element by the method performed by the cockpit control device 134. <i>Supra</i> element [1pre].	To the extent the preamble is limiting, the Johnson-Lin-Purcell combination describes this feature by the method performed by the cockpit control device 134. <i>Supra</i> Section XI.A.[1pre].

Petition at 71	Mitzenmacher Decl. ¶ 232
<i>Supra</i> Element [1.3]. In the Johnson-Lin-Purcell combination, the preloading logic 230 inputs permutations of X variable values (machine learning parameters) incrementally into the predictive program that implements a trained machine learning ensemble. STRATEGY-1004, ¶¶[0151] (“sequencing through a great	<i>Supra</i> Section XI.A.[1.3]. In the Johnson-Lin-Purcell combination, the preloading logic 230 inputs permutations of X variable values (machine learning parameters) incrementally into the predictive program that implements a trained machine learning ensemble. As discussed above for Element [1.3], Johnson describes

number of permutations of actionable X variables”), [0153] (computing results for “specified permutations of input assumptions (e.g., specific combinations of input X variables)”), [0159] (pre-calculating in an “orderly way” for sample points “at regular intervals within the response surface 900”); STRATEGY-1003, ¶¶232-237. The permutations of inputs are between minimum values and maximum values for the X variables and time. STRATEGY-1004, ¶¶[0160]-[0161]; STRATEGY-1003, ¶¶235-237.	calculating results by “sequencing through a great number of permutations of actionable X variables” that serve as machine learning parameters (STRATEGY-1004, ¶[0151]), which can involve computing results for various “permutations of input assumptions (e.g., specific combinations of input X variables)” (<i>id.</i>, ¶[0153]). Johnson also explains that “the preloading logic 230 can probe the response surface 900 in an orderly way, for instance, by selecting sample points for investigation at regular intervals within the response surface 900.” STRATEGY-1004, ¶[0159].
---	--

Petition at 73	Mitzenmacher Decl. ¶ 240
<i>Supra</i> Section III.B.2. In the Johnson-Lin-Purcell combination, user input to the vertical scroll bar adjusts a displayed goal value for the Y variable (machine learning results) from the predictive program. STRATEGY-1006, 5:29-45, 38:1-21; STRATEGY-1003, ¶240. In response, the system dynamically changes the displayed value of the X₁ variable (machine learning parameters), by retrieving and displaying the X₁ value from the database 234 (results data structure) that stores pre-computed analysis results. STRATEGY-1004, ¶¶[0153], [0170], [0068]; STRATEGY-1006, 5:29-45, 38:1-21; STRATEGY-1003, ¶240. By changing the displayed value for the X₁ variable, the displayed input permutation is different from the permutation before adjustment to the goal value. STRATEGY-1003, ¶240.	<i>Supra</i> Section X.E. In the Johnson-Lin-Purcell combination, user input to the vertical scroll bar adjusts a displayed goal value for the Y variable (machine learning results) from the predictive program. STRATEGY-1006, 5:29-45, 38:1-21. In response, the system dynamically changes the displayed value of the X₁ variable (machine learning parameters), by retrieving and displaying the X₁ value that from the database 234 (results data structure) that stores pre-computed analysis results. STRATEGY-1004, ¶¶[0153], [0170], [0068]; STRATEGY-1006, 5:29-45, 38:1-21. By changing the displayed value for the X₁ variable, the displayed input permutation is different from the permutation before adjustment to the goal value.

Petition at 86	Mitzenmacher Decl. ¶ 243
To the extent the preamble is limiting, the Johnson-Lin-Mihaylov combination provides a system for machine learning results, e.g., generating forecasts using functions trained using machine learning algorithms. <i>Supra</i> Section III.C.2; <i>supra</i> element [1pre]; STRATEGY-1003, ¶243.	To the extent the preamble is limiting, the Johnson-Lin-Mihaylov combination describes a system for machine learning results, e.g., generating forecasts using functions trained using machine learning algorithms. <i>Supra</i> Section X.G; <i>supra</i> Section X1.A.[1pre].

Petition at 87	Mitzenmacher Decl. ¶ 244
The Johnson-Lin-Mihaylov combination provides a predictive compiler module configured to generate a plurality of predictive programs. <i>Supra</i> element [1.1a]; STRATEGY-1003, ¶244. Different forecasting models, implemented as predictive programs, are provided for different metrics to be forecasted. <i>Supra</i> Section III.C.2; STRATEGY-1004, ¶¶[0100] (each forecasted Y variable has corresponding functions defined), [0130] (listing various forecasted items); STRATEGY-1007, ¶¶[0002], [0024]; STRATEGY-1003, ¶244. In addition, as Mihaylov teaches, models are changed and recalibrated so there are multiple models, and thus multiple predictive programs, for each metric. STRATEGY-1007, ¶¶[0043]-[0044]; STRATEGY-1004, ¶[0200] (where tracking results of “a particular version of a model,” indicates multiple versions exist); STRATEGY-1003, ¶244.	The Johnson-Lin-Mihaylov combination describes a predictive compiler module configured to generate a plurality of predictive programs. <i>Supra</i> [1.1a]. Different forecasting models, implemented as predictive programs, are described for different metrics to be forecasted. <i>Supra</i> Section X.G; STRATEGY-1004, ¶¶[0100] (each forecasted Y variable can have corresponding functions defined), [0130] (listing various forecasted items); STRATEGY-1007, FIGS. 10-12, ¶¶[0002], [0024], [0050]-[0051]. In addition, as Mihaylov teaches that models are changed and recalibrated so there are multiple models, and thus predictive programs, for each metric. STRATEGY-1007, ¶¶[0043]-[0044]; STRATEGY-1004, ¶[0200] (where tracking results of “a particular version of a model,” indicates multiple versions exist).

Petition at 87	Mitzenmacher Decl. ¶ 245
Each predictive program includes a plurality of learned functions. <i>Supra</i> element [1.1a]; STRATEGY-1004, ¶[0047] (“plural transfer functions”); STRATEGY-1005, 4:16-35, 5:8-22 (discussing ensemble models); STRATEGY-1003, ¶245.	Each predictive program includes a plurality of learned functions, for example, each predictive program can be an ensemble of predictive models. <i>Supra</i> [1.1a]; STRATEGY-1004, ¶[0047] (“plural transfer functions”); STRATEGY-1005, 4:16-35, 5:8-22 (discussing ensemble models).

Petition at 88	Mitzenmacher Decl. ¶ 247
<i>Supra</i> elements [1.3], [16.2]. The Johnson-Lin-Mihaylov combination provides preloading logic 230 (precompute module), which determines permutations of forecasted results for each of the predictive programs, including for programs that forecast different metrics and for different versions of programs that forecast the same metric. STRATEGY-1004, ¶¶[0068], [0071], [0151]-[0152], [0167] (noting “database 234 can store different versions of the output results”); STRATEGY-1003, ¶¶246-247.	The Johnson-Lin-Mihaylov combination describes preloading logic 230 (precompute module), which determines permutations of forecasted results for each of the predictive programs, including for programs that forecast different metrics and for different versions of programs that forecast the same metric. STRATEGY-1004, ¶¶[0068], [0071], [0151]-[0152], [0167] (noting “database 234 can store different versions of the output results”); Section X.G. As discussed above, it was obvious to pre-compute and store error information as part of the precalculated results, so that error information, like other results, can be presented to a user “substantially instantaneously, as opposed to imposing a delay.” STRATEGY-1004, ¶[0152]; Section X.G.1. In the Johnson-Lin-Mihaylov combination, the system is configured to provide error information for multiple versions of models, and to provide comparisons of the accuracy of different models. Accordingly, it was obvious to precompute and store the error information (e.g., holdout values indicating error attributable to the various models, for the various forecasts made) for these

	different predictive programs (e.g., for current and previous predictive programs, for each of the various metrics forecasted).
--	---

Petition at 89	Mitzenmacher Decl. ¶ 248
Results are determined for predefined increments in a range for the input variables (machine learning parameters). STRATEGY-1004, ¶[0159] (precomputation “in an orderly way,” with “sample points for investigation at regular intervals”), ¶[0160] (“ranges of input variables” defined for precomputation); STRATEGY-1003, ¶¶248-249.	Johnson describes and renders obvious that results are determined for predefined increments in a range for the input variables (machine learning parameters). STRATEGY-1004, ¶¶[0159], [0160] (“ranges of input variables” defined for precomputation); see <i>supra</i> Sections XI.A.[1.3] and XI.B.[16.2]. As discussed above, a POSITA would have understood that a range specifies a minimum value and a maximum value, and would have found it obvious to perform precomputation in a bounded range for efficiency. In addition, Johnson states that precomputation “in an orderly way,” with “sample points for investigation at regular intervals.” STRATEGY-1004, ¶[0159]. A POSITA would have understood this use of “regular intervals” to indicate or at least render obvious consistent increments or equally sized intervals.

Petition at 90	Mitzenmacher Decl. ¶ 250
<i>Supra</i> element [1.2]. The Johnson-Lin-Mihaylov combination provides an input module, including an interactive interface 266 presented locally or over a network 216 through a communication interface 212, that is configured to receive user input that identifies values for input variables (machine learning parameters) for predictive programs. STRATEGY-1004, FIGS. 2, 3 (control window 316), 12	<i>Supra</i> [1.2]. The Johnson-Lin-Mihaylov combination describes an input module, including an interactive display 266 presented locally or over a network 216 through a communication interface 212, that is configured to receive user input that identifies values for input variables (machine learning parameters) for predictive programs. STRATEGY-1004, Figs. 2, 3 (control window 316), 12 (knob panel 1210), 13 (knob panel 1312), ¶¶[0052], [0070], [0077], [0101]-[0102],

(knob panel 1210), 13 (knob panel 1312), ¶¶[0052], [0070], [0077], [0101]-[0102], [0105]; STRATEGY-1003, ¶250.	[0105]. Johnson shows that it was obvious for user input to represent actual conditions or hypothetical scenarios for a forecast, noting that “the what-if field 146 allows the cockpit user 138 to enter information into the cockpit interface 134 regarding hypothetical or actual conditions within the business 102.” STRATEGY-1004, [0052].
--	---

Petition at 91	Mitzenmacher Decl. ¶ 252
In the Johnson-Lin-Mihaylov combination, there are predictive programs for each of multiple different metrics. STRATEGY-1007, ¶¶[0002], [0024]; STRAT-EGY-1003, ¶¶251-252. There are also multiple versions of predictive programs for each metric, e.g., one version for the current or most-recent model and a prior version for the previous model. STRATEGY-1007, ¶¶[0043]-[0045]; STRAT-EGY-1003, ¶252. The current versions of the predictive programs represent the “first subset” of the predictive programs. STRATEGY-1003, ¶252.	As discussed above, in the Johnson-Lin-Mihaylov combination, there are predictive programs for each of multiple different metrics as Mihaylov describes. STRATEGY-1007, FIGS. 10-12 (showing Metrics 1-17), ¶¶[0002], [0024], [0051]-[0052]. For each metric there are multiple predictive programs, including (1) a current version implementing the current or most-recent predictive model for the metric and (2) a previous version implementing the previous predictive model for the metric. STRATEGY-1007, ¶¶[0043]-[0045]. The various predictive programs for the various metrics can be subdivided into subsets in different ways. One way is for the “first subset” to include the current predictive programs for the metrics, and to exclude the prior predictive programs for those metrics. This is the example that I will use for discussing claim 21 and 23 below. (Of course, other groupings of the predictive models also show the features of the claims, such as by instead considering the “first subset” to be the set of previous predictive programs for the metrics and excluding the current predictive programs.)

Petition at 92	Mitzenmacher Decl. ¶ 253
In the error analysis interface of the Johnson-Lin-Mihaylov combination, selecting a metric to analyze selects a corresponding predictive program. STRATEGY-1007, FIG. 11, ¶¶[0050]-[0051]; STRATEGY-1003, ¶253.	In the error analysis interface of the Johnson-Lin-Mihaylov combination, selecting a metric to analyze chooses a corresponding predictive program. STRATEGY-1007, FIG. 11, ¶¶[0050]-[0051]. When a metric is selected by a user, as shown in Mihaylov's FIG. 11 (where "Metric 1" has been selected), a corresponding predictive program (e.g., a predictive model for the selected metric, implemented as a computer program) is part of a "first subset" of the predictive programs. <i>Id.</i> In other words, there is a "first subset" that includes at least the current predictive program for the selected metric, and can include one or more of the current predictive programs for other metrics. (In this case, there is also a different program that is excluded from this first subset, the program that implements the previous version of the predictive model for the same selected metric.) As discussed further below with respect to claim 23, the "first subset" can include multiple programs, such as one or more of the current programs for various metrics, while programs for previous versions of these models are excluded from the first subset.

Petition at 92	Mitzenmacher Decl. ¶ 254
The current or most recent predictive program for the selected metric is included in the "first sub-set" of the predictive programs. <i>Id.</i> The user-specified input values are "for" the first subset, because those input values define the forecasting scenario (e.g., inputs or input assumptions) under which the selected	The user-identified input values discussed for element [21.2] above are "for" the first subset of predictive programs, because those input values define the forecasting scenario (e.g., the set of inputs or input assumptions used to make a forecast) for which the predictive programs are analyzed in the error analysis interface. STRATEGY-1004, [0030], [0052], [0067];

program is analyzed in the error analysis interface. STRATEGY-1003, ¶254.	STRATEGY-1007, [0004], [0040]-[0041]. In other words, the error information shown in the interface will represent error for the forecast generated based on the user-identified input values. When the user has selected a metric, the error information indicates error for the current predictive program (which is included in the “first subset”) for the scenario defined by the user-identified input values. If the user selects a different metric, the error information would indicate error for the same scenario for the newly-selected metric’s current predictive program (which is also part of the “first subset”).
---	---

Petition at 93	Mitzenmacher Decl. ¶ 255
The Johnson-Lin-Mihaylov combination provides a collaboration module, including error analysis logic included based on Mihaylov, that is configured to attribute error to different sources, including determining an amount of error attributable to changing to the current predictive program from the previous predictive program. STRATEGY-1007, ¶¶[0043]-[0044]; STRATEGY-1003, ¶255. The previous predictive program represents a “different one” of the predictive programs that is outside the “first subset.” STRATEGY-1003, ¶255.	The Johnson-Lin-Mihaylov combination describes a collaboration module, including error analysis logic included based on Mihaylov, that is configured to attribute error to different sources, including determining an amount of error attributable to changing to the current predictive program from the previous predictive program. STRATEGY-1007, ¶¶[0043]-[0044]. The previous predictive program for a metric represents a “different one” of the predictive programs that is outside the “first subset” of predictive programs, which is the group of current predictive programs.

Petition at 93	Mitzenmacher Decl. ¶ 258
As Mihaylov teaches, error analysis includes determining holdout values for the current model that are compared with holdout values for the previous model. STRATEGY-1007, FIG. 7, ¶¶[0043]-[0044], [0034] (“holdout values” measure the extent that deviation from actual performance is “a consequence of flaws in the model itself”); STRATEGY-1003, ¶¶256-258. Comparing the holdout values determines the impact (e.g., difference in accuracy) on machine learning results of the current predictive program from a different predictive program, e.g., the program for the previous model version. STRATEGY-1007, FIG. 7 (“model change impact”), FIG. 11; STRATEGY-1003, ¶¶256-258.	As Mihaylov describes, error analysis includes determining holdout values for the current model that are compared with holdout values for the previous model. <i>Id.</i>, FIG. 7, ¶¶[0043]-[0044], [0034] (“holdout values” measure the extent that deviation from actual performance is “a consequence of flaws in the model itself”). Comparing the holdout values determines the impact (e.g., difference in accuracy) on machine learning results of the current predictive program from a different predictive program, e.g., the program for the previous model version. STRATEGY-1007, FIG. 7 (“model change impact”), FIG. 11.

Petition at 93–94	Mitzenmacher Decl. ¶ 259
The error analysis logic performs the same algorithm that the collaboration module described in the ’574 patent, including accessing data relating to different models (e.g., holdout values for different models), and determining an impact on results (e.g., determining model change impact). STRATEGY-1003, ¶259. The current version of the model represents the “first subset of the predictive programs, and the prior version of the model represents the “different one of the predictive programs.” <i>Id.</i>	The error analysis logic performs the same algorithm as the collaboration module described in the ’574 patent, including accessing data relating to different models (e.g., holdout values for different models), and determining an impact on results (e.g., determining model change impact). The current version of the model represents the “first subset of the predictive programs, and the prior version of the model represents the “different one of the predictive programs.”

Petition at 94	Mitzenmacher Decl. ¶ 260
In the Johnson-Lin-Mihaylov combination, the user specifies the parameters for the “what-if” scenario to be assessed, and so the holdout values for programs implementing both the current model and the prior model—and thus the model change impact—are “based on the one or more identified values for the one or more of the machine learning parameters” specified by user input. STRATEGY-1003, ¶260. Even when actual outcomes (e.g., historical data) provide some inputs for generating the holdout values, the user’s input defining the “what-if” scenario nevertheless specifies “one or more” of the parameters that affect model output. <i>Id.</i>	In the Johnson-Lin-Mihaylov combination, the user determines the parameters for the “what-if” scenario to be assessed, and so the holdout values for both the current model and the prior model—and thus the model change impact—are “based on the one or more identified values for the one or more of the machine learning parameters” specified by user input. Even when actual outcomes (e.g., historical data) describe some inputs for generating the holdout values, the user’s input defining the “what-if” scenario nevertheless identifies “one or more” of the parameters that affect model output. For example, Johnson describes that the user can enter “actual conditions within the business” as inputs that are used to define the scenario for forecasting, or the scenario can reflect user-specified hypothetical conditions. STRATEGY-1004, ¶[0052].

Petition at 94–95	Mitzenmacher Decl. ¶ 261
In the Johnson-Lin-Mihaylov combination, the system pre-calculates and caches machine learning results, including error information such as the holdout values that are generated using the predictive programs. STRATEGY-1003, ¶261. Johnson describes that its predictions include forecasted values and associated “error information indicating the degrees of confidence” for the forecasts. STRATEGY-1004, ¶[0066]; see also <i>id.</i>, ¶¶[0072] (result display includes displaying confidence), [0080], [0106] (example of prediction including confidence measures), [0133] (error measures used by a user or algorithm). It was obvious to pre-calculate and store error information in the database	In the Johnson-Lin-Mihaylov combination, the system pre-calculates and caches machine learning results, including error information such as the holdout values that are generated using the predictive programs. Johnson describes that its predictions include forecasted values and associated “error information indicating the degrees of confidence” for the forecasts. STRATEGY-1004, ¶[0066]; see also <i>id.</i>, ¶¶[0072] (result display includes displaying confidence), [0080], [0106] (example of prediction including confidence measures), [0133] (error measures used by a user or algorithm). It was obvious to pre-calculate and store error information in the database 234, so

234, so the system can avoid delay and present results with associated error information rapidly, based on Johnson's teachings. <i>Id.</i>, ¶[0152]; STRATEGY-1003, ¶261. Accordingly, the holdout values used to determine the model change impact are predetermined permutations of machine learning results, retrieved from the database 234 of cached results. STRATEGY-1003, ¶261.	the system can avoid delay and present results with associated error information "substantially instantaneously," as Johnson teaches. <i>Id.</i>, ¶[0152]. Thus, the holdout values used to determine the model change impact are predetermined permutations of machine learning results, retrieved from the database 234 of cached results.
--	---

Petition at 96	Mitzenmacher Decl. ¶ 262
The Johnson-Lin-Mihaylov combination provides and renders obvious the same structure that the '574 patent teaches for the collaboration module, e.g., algorithm steps of (i) accessing data relating to different models or different user inputs (STRATEGY-1004, ¶¶[0071], [0153], [0166] (accessing precomputed data); STRATEGY-1007, FIGS. 6-7, 10-12, ¶¶[0040]-[0044], [0050]-[0051] (accessing data for different models or input assumptions)), and (ii) determining an impact on results from the accessed data (STRATEGY-1007, FIGS. 7, 10-12, ¶¶[0043]-[0044], [0050]-[0051]).¹¹ STRATEGY-1003, ¶¶262-263.	The Johnson-Lin-Mihaylov combination provides and renders obvious the same structure that the '574 patent teaches for the collaboration module, e.g., algorithm steps of (i) accessing data relating to different models or different user inputs (STRATEGY-1004, ¶¶[0071], [0153], [0166]-[0168] (accessing precomputed data); STRATEGY-1007, FIGS. 6-7, 10-12, ¶¶[0026]-[0027], [0040]-[0044], [0050]-[0051] (accessing data for different models or input assumptions)), and (ii) determining an impact on results from the accessed data (STRATEGY-1007, FIGS. 7, 10-12, ¶¶[0034]-[0035], [0043]-[0044], [0050]-[0051]).

Petition at 96-97	Mitzenmacher Decl. ¶ 264
The Johnson-Lin-Mihaylov combination provides a display module (see [1.4] above) configured to display the impact on the machine learning results to the one or more users. STRATEGY-1003, ¶264. The display module includes the display presentation logic 236 that provides the cockpit interface 134, configured to additionally show error information including the impact of model change, based on Mihaylov. STRATEGY-1004,	The Johnson-Lin-Mihaylov combination describes a display module configured to display the impact on the machine learning results to the one or more users. The display module includes the display presentation logic 236 and the user interface of the cockpit interface 234, configured to additionally show the impact of model change based on Mihaylov. STRATEGY-1004, ¶¶[0072], [0077]; STRATEGY-1007, FIG. 11, ¶¶[0051], [0044]-

¶¶[0072], [0077]; STRATEGY-1007, FIG. 11, ¶¶[0051], [0044]-[0046]; STRATEGY-1003, ¶264. . . . The user interface can show model change impact in a displayed object, such as a graph, and in a graphical table, as shown in Mihaylov's FIG. 11 (below). STRATEGY-1007, FIG. 11; STRATEGY-1003, ¶264.	[0046]. The user interface can show model change impact in a displayed object, such as a graph, and in a graphical table, as shown in Mihaylov's FIG. 11. STRATEGY-1007, FIG. 11.
--	--

Petition at 99–100	Mitzenmacher Decl. ¶ 266
The Johnson-Lin-Mihaylov combination provides an update module configured to dynamically update the display of information in response to users changing values for machine learning parameters. <i>Supra</i> element [1.5]; STRATEGY-1004, ¶¶[0072], [0101], [0105], [0167]-[0170]; STRATEGY-1003, ¶266. The update module includes the display presentation logic 236 and the pre-loading logic 230, which perform the same algorithm described in the '574 patent. The steps include (i) detecting user input that changes parameter values (e.g., changes to the “X” variables or a “what-if” scenario) (STRATEGY-1004, ¶¶[0182]-[0184], [0102]), (ii) retrieving cached information corresponding to the input (STRATEGY-1004, ¶¶[0007], [0153], [0167]-[0170, FIG. 10), and (iii) displaying the updated information on the cockpit interface 134 (STRATEGY-1004, ¶¶[0007], [0153], [0167]-[0170], FIG. 10). STRATEGY-1003, ¶266.	The Johnson-Lin-Mihaylov combination describes an update module configured to dynamically update the display of information in response to users changing values for machine learning parameters. <i>Supra</i> [1.5]; STRATEGY-1004, ¶¶[0072], [0101], [0105], [0167]-[0170]. The update module includes the display presentation logic 236 and the pre-loading logic 230, which perform the same algorithm described in the '574 patent. Johnson's process includes (i) detecting user input that changes parameter values (e.g., changes to the “X” variables or a “what-if” scenario) (STRATEGY-1004, ¶¶[0182]-[0184], [0102]), (ii) retrieving cached information corresponding to the input (STRATEGY-1004, ¶¶[0007], [0153], [0167]-[0170], FIG. 10), and (iii) displaying the updated information on the cockpit interface 134 (STRATEGY-1004, ¶¶[0007], [0153], [0167]-[0170], FIG. 10).

Petition at 100	Mitzenmacher Decl. ¶ 267
The system is configured to “dynamically display changes” to the displayed results “while the cockpit user 138 varies one or more input mechanisms (318, 320, 322, 324).” STRATEGY-1004, ¶[0152]; <i>see also id.</i>, ¶¶[0102], [0118], [0179]; STRATEGY-1003, ¶267. In the Johnson-Lin-Mihaylov combination, which pre-calculates and displays error information such as model change impact, the “real-time coupling between actuations made in the control window 316 and changes presented” in the interface 134 extends to error information, such as the model change impact, in addition to the forecasted values. STRATEGY-1003, ¶267. As users vary parameters with the input mechanisms, the impact on results provided as model change impact is also updated. STRATEGY-1003, ¶267.	The system is configured to “dynamically display changes” to the displayed results “while the cockpit user 138 varies one or more input mechanisms (318, 320, 322, 324).” STRATEGY-1004, ¶[0152]; <i>see also id.</i>, ¶¶[0102], [0118], [0179]. In the Johnson-Lin-Mihaylov combination, which pre-calculates and displays error information such as model change impact, the “real-time coupling between actuations made in the control window 316 and changes presented” in the interface 134 extends to error information, such as the model change impact, in addition to the forecasted values. Thus, as users vary parameters with the input mechanisms, the impact on results described as model change impact is also updated.

Petition at 100–101	Mitzenmacher Decl. ¶ 268
The Johnson-Lin-Mihaylov combination supports multiple users, with users 138 and 250 at respective workstations 246 and 248. STRATEGY-1004, FIG. 2 (showing), ¶¶[0076]–[0078]. Multiple users can also use the same workstation or same display screen, e.g., sitting physically at the same desk. STRATEGY-1003, ¶268.	The Johnson-Lin-Mihaylov combination supports multiple users, with users 138 and 250 at respective workstations 246 and 248. STRATEGY-1004, FIG. 2 (showing), ¶¶[0076]–[0078]. Multiple users can also use the same workstation or same display screen, e.g., sitting physically at the same desk. A POSITA would have understood that it was common for multiple users to view and provide input at the same computer or workstation. STRATEGY-1030, ¶[0003] (“For a computer user to share [a] document with other users, traditionally the two users would view the same screen while being at the same physical location;” “such sharing is convenient when both users are in the same vicinity”); STRATEGY-1031, 79

	(“Users may wish to collaborate on the same workstation screen”). When multiple users are collaborating at the same computer, it would be obvious and expected for multiple users to each provide some of the input parameter values that define the scenario to be forecast, especially as scenarios are incrementally updated and changed during the collaboration.
--	--

Petition at 101	Mitzenmacher Decl. ¶ 269
The Johnson-Lin-Mihaylov combination also allows predictions based on multiple values for input parameters, such as multiple “X” variables as well as selecting a product, time period, and other parameters. STRATEGY-1004, ¶¶[0082], [0105]; STRATEGY-1007, ¶¶[0050]-[0051]; STRATEGY-1003, ¶269.	The Johnson-Lin-Mihaylov combination also allows predictions and forecasts to be based on multiple values for input parameters, such as values for multiple “X” variables to define a scenario to forecast (STRATEGY-1004, FIG. 13 (showing a panel of five knobs to separately adjust five different X variables), ¶¶[0101]-[0105], [0121], [0167], [0184], [0202]) as well as selecting a product, time period, and other parameters (STRATEGY-1007, FIGS. 10-12, ¶¶[0050]-[0051]). See STRATEGY-1004, ¶[0121] (system enables “selecting a set of input assumptions, such as a particular combination of actionable X variables associated with a set of input knobs provided on the cockpit interface 134”).

Petition at 101	Mitzenmacher Decl. ¶ 270
Any of the multiple users can set multiple values for the machine learning parameters. STRATEGY-1003, ¶270. These values define the “what-if” scenario to be forecasted, and are used for multiple different programs in the first subset of predictive programs. <i>Id.</i> For example, the first subset can be the set of all current	Any of the multiple users can set multiple values for the machine learning parameters. These values define the “what-if” scenario to be forecasted, e.g., the input assumptions or input case to analyze, and can be used for multiple different programs in the first subset of predictive programs. STRATEGY-1004,

programs for the various metrics. STRATEGY-1007, FIG. 11-17 (showing metrics 1-17), ¶[0050] (selecting a metric “allow[s] a user to view and/or analyze a different model”); STRATEGY-1003, ¶270.	¶¶[0069]-[0070], [0083], [0167]. For example, the first subset can be the set of all current programs for the different metrics that can be predicted. STRATEGY-1007, FIG. 11-17 (showing metrics 1-17), ¶[0050] (selecting a metric “allow[s] a user to view and/or analyze a different model”).
---	---

Petition at 101	Mitzenmacher Decl. ¶ 271
The scenario specified with user input would be used for any of the different programs for the different metrics. STRATEGY-1003, ¶¶270-271. In other words, after defining a scenario with user input, the current programs for any of the metrics would use those values for the parameters. STRATEGY-1003, ¶¶270-271.	When users specify the input scenario, e.g., the values for multiple X variables that Johnson describes, the user-specified input scenario would be used for any of the different programs corresponding to the different metrics, which are respectively based on current and previous models as Mihaylov describes (STRATEGY-1007, ¶¶[0043]-[0045]). After defining a scenario with user input, any of the current programs would use those values for the parameters that specify those parameters, until the users change the parameter values to define a new scenario to be forecasted. Based on Mihaylov, it was obvious to enable users to select different metrics to forecast, with each metric having different corresponding programs. STRATEGY-1007, FIGS. 10-12 (showing metric drop-down menu 1003), ¶¶[0050]-[0051] (“metric drop-down menu 1003, which may allow a user to view and/or analyze a different model and data associated therewith”). Accordingly, the same input parameter values that define a forecasting scenario are used and maintained “for” different predictive programs in the first subset (e.g., the set of current predictive programs across the different metrics). In other words, it is obvious that when users select different metrics, or change from one metric to another, the same multiple input values

	are being used to provide results for different predictive programs.
--	--

Petition at 102	Mitzenmacher Decl. ¶ 272
In the Johnson-Lin-Mihaylov combination, the display module displays the impact on the machine learning results on a shared display, e.g., a single window or view shows the model change impact. STRATEGY-1003, ¶272. Two users sharing a workstation would both see the display. STRATEGY-1003, ¶272.	In the Johnson-Lin-Mihaylov combination, the display module displays the impact on the machine learning results on a shared display, e.g., a single window or view shows the model change impact. A POSITA would have recognized that it was common for multiple users to view the same display or to share a workstation. STRATEGY-1030, ¶[0003] “For a computer user to share [a] document with other users, traditionally the two users would view the same screen while being at the same physical location,” and “such sharing is convenient when both users are in the same vicinity”); STRATEGY-1031, 79 (“Users may wish to collaborate on the same workstation screen.”). Two users sharing a workstation would both see the display, e.g., by being in the same office or conference room and viewing the same screen, and would have a shared display. <i>Id.</i> When users use a shared display, the impact on machine learning results (such as the model change impact indicating error or accuracy differences for models) is shown on the shared display.

Petition at 102	Mitzenmacher Decl. ¶ 273
The display module is also configured to display the machine learning results for the different predictive programs on different displays for the multiple users. STRATEGY-1004, FIG. 2, ¶[0076]; STRATEGY-1003, ¶273. For example, users 138 and 250 at different workstations 246 and 248 can each select a different metric to view, and they will be shown corresponding results	In addition to supporting multiple users viewing a shared display, the display module in the Johnson-Lin-Mihaylov combination is also configured to display the machine learning results for different predictive programs on different displays for the multiple users. For example, Johnson describes multiple users 138 and 250 at different workstations 246 and 248.

for different programs (e.g., the programs that implement the current model for the respectively selected metric). <i>Id.</i>	STRATEGY-1004, FIG. 2, ¶[0076]. A POSITA would have understood that different users with different workstations, which Johnson describes, can provide different inputs and select different metrics to view. <i>Id.</i> , FIG. 2, ¶[0076]. When different users select different metrics, they will be shown corresponding results for different programs, e.g., the programs that implement the current model for the respectively-selected metric. See STRATEGY-1007, Figs. 10-12 (showing metric drop-down menu 1003), ¶¶[0050]-[0051] (“metric drop-down menu 1003, which may allow a user to view and/or analyze a different model and data associated therewith”). Accordingly, it was obvious that two users at different workstations, able to separately select the metrics and thus programs to be used for forecasting, would have different results displayed to them as a result of their selections.
---	--

Petition at 102	Mitzenmacher Decl. ¶ 274
The Johnson-Lin-Mihaylov combination provides an input module configured to receive user input from separate client devices, e.g., workstations 246 and 248, over a data network, e.g., network 216. STRATEGY-1004, FIG. 2, ¶¶[0076]-[0078], [0065]; STRATEGY-1003, ¶274. The input module obtains user input through the communication interface 212. <i>Id.</i>	The Johnson-Lin-Mihaylov combination describes an input module configured to receive user input from separate client devices, e.g., workstations 246 and 248, over a data network, e.g., network 216. STRATEGY-1004, FIG. 2, ¶¶[0076]-[0078], [0065]. The input module obtains user input through the communication interface 212. <i>Id.</i> It is obvious that users at separate client devices, as Johnson describes, provide input to their respective devices, and that to support these multiple devices the input module of the cockpit control module 132 would be able to receive input from both devices, e.g., an interactive display 266 is provided to each device, and the communication interface 212 and

	display presentation logic 236 handle input from multiple devices. See <i>id.</i>
--	---

Petition at 103	Mitzenmacher Decl. ¶ 275
In the Johnson-Lin-Mihaylov combination, the input module also provides a shared interface for the multiple users on a single device. STRATEGY-1003, ¶275. Multiple users sitting at a single workstation or display would share the interface 134. <i>Id.</i>	In the Johnson-Lin-Mihaylov combination, the input module also describes a shared interface for the multiple users on a single device. Multiple users sitting at a single workstation or display would share the interface 134, including the interactive display 266. As discussed above, a POSITA would have understood that it was obvious for multiple users to share the interface of a single device. See, e.g., STRATEGY-1030, ¶[0003] (“For a computer user to share [a] document with other users, traditionally the two users would view the same screen while being at the same physical location;” “such sharing is convenient when both users are in the same vicinity”); STRATEGY-1031, 79 (“Users may wish to collaborate on the same workstation screen”).

Petition at 106–107	Mitzenmacher Decl. ¶ 276
In the Johnson-Lin-Purcell-Mihaylov combination, the display module is configured to display one or more attributes of a data set used by the machine learning to determine the results. STRATEGY-1003, ¶¶276-280. Mihaylov states that models can receive inputs of “statistics, metrics, sub-metrics, and/or other data” (e.g., attributes) “from one or more account information tables and/or databases” (e.g., data sets). STRATEGY-1007, ¶[0026] (noting house price index as one example). Johnson’s X variables also represent attributes of a data set that are displayed. STRATEGY-1004 FIGS. 3, 12-18, ¶¶[0069]-[0070], [0075], [0100]-[0102],	In the Johnson-Lin-Purcell-Mihaylov combination, the display module is configured to display one or more attributes of a data set used by the machine learning to determine the one or more machine learning results. Mihaylov states that models can receive inputs of “statistics, metrics, sub-metrics, and/or other data” (e.g., attributes) “that are loaded from one or more account information tables and/or databases” (e.g., data sets). STRATEGY-1007, ¶[0026]. As an example, a model may “rel[y] on the current house price index,” which represents one attribute of the data set. <i>Id.</i> Purcell shows that it was obvious to

[0182]-[0184]; STRATEGY-1003, ¶¶277-278. Purcell shows that it was obvious to display the input values to modeling functions, including displaying values, labels, and other information. STRATEGY-1006, FIG. 47; STRATEGY-1003, ¶276.	display the input values to modeling functions, including displaying values, labels, and other information in spreadsheet format. STRATEGY-1006, FIG. 47.
---	--

Petition at 107	Mitzenmacher Decl. ¶ 279–280
In the Johnson-Lin-Purcell-Mihaylov combination, the display module is also configured to display impact metrics for displayed attributes. STRATEGY-1003, ¶279. For example, Mihaylov teaches determining an error attribution measure for each of various different attributes or inputs, “so as to attribute various components of the assumption error to particular input variables.” STRATEGY-1007, ¶[0040]; STRATEGY-1003, ¶¶279-280. It was obvious to display error information, e.g., an impact metric, for each input (e.g., attribute of the data set) to the predictive program. <i>Id.</i> It was also obvious to determine and display error measures (e.g., impact metrics) for attributes “so as to facilitate the analysis of and/or the preparation of reports regarding one or more modeling functions and their corresponding forecast error analyses.” STRATEGY-1007, ¶[0048]; STRATEGY-1003, ¶¶279-280.	In the Johnson-Lin-Purcell-Mihaylov combination, the display module is also configured to display one or more impact metrics for each displayed attribute. For example, Mihaylov teaches determining an error attribution measure for each of various different attributes or inputs that a model relies on, “so as to attribute various components of the assumption error to particular input variables.” STRATEGY-1007, ¶[0040]. Mihaylov’s gives an example of apportioning “particular error amounts” to errors for “unemployment” and “house price index (HPI),” both of which are specified as “economic inputs” to the forecasting process. <i>Id.</i>, ¶¶[0040]-[0041]. This demonstrates Mihaylov’s technique of determining and displaying an impact metric for each input value (e.g., attribute of the data set), where the impact metrics represent an amount of error resulting from the corresponding input or metric. <i>Id.</i> Based on Mihaylov’s technique, it was obvious to also determine and display similar error information (e.g., impact metrics) for the different X variables that Johnson describes as inputs to predictive models. Thus, it was obvious to determine error measures for particular attributes, which indicate the impact or influence of those attributes on forecasting, and display those measures to the user “so as to facilitate the analysis of and/or the preparation of reports regarding one or

	more modeling functions and their corresponding forecast error analyses.” <i>Id.</i> , ¶[0048].
--	--

Petition at 107–108	Mitzenmacher Decl. ¶ 281
The Johnson-Lin-Purcell-Mihaylov combination provides a recommendation module configured to select a suggested set of machine learning results. STRATEGY-1003, ¶281. Johnson describes an “automated optimization routine 428” that “can automatically sequence through a number of case assumptions and then select one or more case assumptions that best accomplish a predefined objective (such as maximizing profitability, minimizing risk, etc.).” STRATEGY-1004, ¶[0092]; STRATEGY-1003, ¶281. This generates a “recommendation” provided so the user can select an action to take for the business. <i>Id.</i> , ¶¶[0092], [0093] (noting that the routine 428 can be “automatically triggered” without manual user action); STRATEGY-1003, ¶281.	The Johnson-Lin-Purcell-Mihaylov combination describes a recommendation module configured to select a suggested set of machine learning results. Johnson describes an “automated optimization routine 428” that “can automatically sequence through a number of case assumptions and then select one or more case assumptions that best accomplish a predefined objective (such as maximizing profitability, minimizing risk, etc.). The cockpit user 138 can use the recommendation generated by the automated optimization routine 428” to select an action to take for the business. STRATEGY-1004, ¶¶[0092], [0093] (noting that the routine 428 can be “automatically triggered” without manual user action).

Petition at 108	Mitzenmacher Decl. ¶ 282
In the Johnson-Lin-Purcell-Mihaylov combination, the display module is configured to display the suggested set of results to the user prior to the input module receiving the user input. STRATEGY-1003, ¶282. As Johnson explains, the generation of recommended results (e.g., optimized results) can be “automatically triggered” based on “predefined events” or “collected data,” showing selection and display before the user provides user input. STRATEGY-1004, ¶[0093]; STRATEGY-1003, ¶282. A POSITA would understand that the recommended	In the Johnson-Lin-Purcell-Mihaylov combination, the display module is configured to display the suggested set of machine learning results to the user prior to the input module receiving the user input. As Johnson explains, the generation of recommended results (e.g., optimized results) can be “automatically triggered” based on “predefined events” or “collected data,” and thus can be selected and displayed before the user provides user input. STRATEGY-1004, ¶[0093]. A POSITA would understand that the recommended results would be displayed

results would be displayed on the cockpit interface 134, where that the “cockpit user 138 can use the recommendation generated by the automated optimization routine 428 to select an appropriate do-what command.” STRATEGY-1004, ¶[0092].	on the cockpit interface 134 so that the “cockpit user 138 can use the recommendation generated by the automated optimization routine 428 to select an appropriate do-what command.” <i>Id.</i> , ¶[0092].
---	--

Petition at 109	Mitzenmacher Decl. ¶ 283
The Johnson-Lin-Purcell-Mihaylov combination provides and renders obvious the same structure that the '574 patent teaches for the recommendation module, e.g., an algorithm step to select a suggested or optimal set of machine learning results that is recommended (STRATEGY-1004, ¶¶[0067], [0092]-[0097]). STRATEGY-1003, ¶283-284.	The Johnson-Lin-Purcell-Mihaylov combination provides and renders obvious the same structure that the '574 patent teaches for the recommendation module, e.g., an algorithm step to select a suggested or optimal set of machine learning results that is recommended (STRATEGY-1004, ¶¶[0067], [0092]-[0097]).

Petition at 109	Mitzenmacher Decl. ¶ 285
The Johnson-Lin-Purcell-Mihaylov combination includes a collaboration module as discussed in Section III.C.2 and Element [21.4]. STRATEGY-1003, ¶¶285-288. The collaboration module determines an impact on the machine learning results based on different user input received from a different user. <i>Id.</i> This includes the error impact corresponding to the “Overlays & Initiatives” trend line in Mihaylov’s FIG. 8, which represents error due to a “forecast adjustment created by management,” which is user input from a different user, e.g., a management user. STRATEGY-1007, FIG. 8, ¶[0047]; STRATEGY-1003, ¶¶285, 288.	The Johnson-Lin-Purcell-Mihaylov includes a collaboration module as discussed in Section X.G and XI.C.[21.4]. The collaboration module is configured to determine an impact on the one or more machine learning results, and that impact can be determined based on different user input (e.g., an additional or separate user input from what is described in claim 1) that is received from a different user. For example, Mihaylov describes determining and displaying the impact, such as error measurements, resulting from any of various different sources of error, including model changes, assumptions, management decisions, and more. STRATEGY-1007, FIGS. 5-12, ¶¶[0004], [0042]-[0052].

Petition at 109	Mitzenmacher Decl. ¶ 286
Similarly, it was obvious to determine impact for user input from a different user located at the same workstation or a different workstation. STRATEGY-1003, ¶¶286-287.	It is obvious for the determined impact on results to be based on input from a different user in several ways. For example, a POSITA knew that it was common for multiple users to collaborate at a single workstation or screen. STRATEGY-1030, ¶[0003] (“traditionally ... two users would view the same screen while being at the same physical location ... such sharing is convenient when both users are in the same vicinity”); STRATEGY-1031, 79 (“Users may wish to collaborate on the same workstation screen”). In this common situation, where a second user provides additional user input that alters or supplements the input scenario or other assumptions, it was obvious for the system to determine and display the corresponding change or impact to the results. This is consistent with Johnson’s stated goal to show updates in response to changes in input. STRATEGY-1004, ¶¶[0085], [0102], [0197].

Petition at 110	Mitzenmacher Decl. ¶ 290
The Johnson-Lin-Purcell-Mihaylov combination allows input from multiple users 138 and 250, demonstrating that the input module is configured to receive adjustments to “what-if” scenarios or forecasts (“the different user input”) from a different user. STRATEGY-1004, FIG. 2, ¶[0076]; STRATEGY-1003, ¶289. The “different user input” discussed for Claim 14 can include a “forecast adjustment” that causes a change from the scenario or assumptions. STRATEGY-1007, FIG. 8, ¶[0047]; STRATEGY-1003, ¶289. The “forecast adjustment” represents a “different value” (e.g., a change from the	In the Johnson-Lin-Purcell-Mihaylov combination, the “different user input” can represent adjustments to “what-if” scenarios or forecasts (“the different user input”), as discussed above for claim 14. STRATEGY-1004, ¶[0101]. For example, when two users are together at the same workstation or computer screen, it is obvious that one user can supply a value for one machine learning parameter (e.g., an input to the predictive program), and the other user can provide the value for a different machine learning parameter.

value that would otherwise have been used) for a “different machine learning parameter” (e.g., a parameter other than the ones set for generating the standard unadjusted forecast). <i>Id.</i> It was obvious for the different user input to be a change to the predictive program inputs (e.g., Johnson’s X variables) or other machine learning parameters. STRATEGY-1003, ¶¶290-292.	
---	--

Petition at 110–111	Mitzenmacher Decl. ¶ 293
In the Johnson-Lin-Purcell-Mihaylov combination, the display module is configured to display multiple sets of one or more machine learning results to multiple users, including with different views at each of multiple workstations 246 and 248. STRATEGY-1004, FIG. 2, ¶[0076]; STRATEGY-1003, ¶293. The update module is configured to dynamically update the displayed multiple sets of one or more machine learning results, e.g., with each display being dynamically adjusted for the corresponding user’s inputs. STRATEGY-1004, ¶¶[0101], [0105], [0118], [0167]; STRATEGY-1003, ¶293.	The Johnson-Lin-Purcell-Mihaylov combination includes an update module as discussed in Section XI.A.[1.5] above. In the Johnson-Lin-Purcell-Mihaylov combination, the display module is configured to display multiple sets of one or more machine learning results to multiple users, including with different views at each of multiple workstations 246 and 248. STRATEGY-1004, FIG. 2. In addition, as Johnson shows in FIG. 3, a single user interface can show multiple windows and thus provide multiple sets of results to the multiple users even on a single user interface. <i>Id.</i> , FIG. 3, ¶¶[0079]-[0086] (discussing multiple windows, including multiple windows of results “on a single display presentation”). The update module is configured to dynamically update the displayed multiple sets of one or more machine learning results, e.g., with each display being dynamically adjusted for the corresponding user’s inputs. <i>Id.</i> , ¶¶[0101], [0105], [0118], [0167]. It would have been obvious to apply Johnson’s technique to provide dynamic updates to each of multiple users that interact with the system through user interfaces (e.g., the interactive display 266 of the cockpit interface 134).

Petition at 111	Mitzenmacher Decl. ¶ 294
The Johnson-Lin-Purcell-Mihaylov combination determines an impact on machine learning results from a different predictive program, e.g., a model change impact. Section III.C.2, Element [21.4]; STRATEGY-1003, ¶294. The model change impact is determined from the results data structure because the data for determining the impact is taken from the database 234 of pre-calculated results. <i>Id.</i>	The Johnson-Lin-Purcell-Mihaylov combination determines an impact on machine learning results from a different predictive program, e.g., a model change impact. Sections X.G, XI.C.[21.4]. The model change impact is determined from the results data structure because the data for determining the impact is taken from the database 234 of pre-calculated results. <i>Id.</i>

Petition at 111	Mitzenmacher Decl. ¶ 295
In the Johnson-Lin-Purcell-Mihaylov combination, the user is able to specify a goal value, which adjusts the displayed value for machine learning results. Section III.B.2, Element [16.5]. By changing the goal value (“displayed value for machine learning results”), the user changes the set of results and the corresponding error information that are applicable. STRATEGY-1003, ¶295. A user changing the goal value causes the system to determine the impact on results from a different predictive program, e.g., the program for the prior model version, for the particular goal value result adjusted by the user. <i>Id.</i>	In the Johnson-Lin-Purcell-Mihaylov combination, the user is able to identify a goal value, and thus adjust the displayed value for machine learning results. See Sections X.E.1, XI.B.[16.5]. By changing the goal value (“displayed value for machine learning results”), the user changes the set of results and corresponding error information that are applicable. Thus, changing the goal value causes determination of impact of the model change impact for the particular goal value adjusted by the user.

Petition at 111–112	Mitzenmacher Decl. ¶ 296
The Johnson-Lin-Purcell-Mihaylov combination displays the impact on machine learning results from the different predictive program, determined as discussed for claim 19. See Element [21.5], Claim 19; STRATEGY-1007, FIG. 10-12 (showing model change impact), ¶¶ [0051]-[0051]; STRATEGY-1003, ¶296.	The Johnson-Lin-Purcell-Mihaylov combination displays the impact on machine learning results from the different predictive program. See Section XI.C.[21.5], Section XI.D.[Claim 19]. In the combination, the impact determined using the features discussed for claim 19 is displayed in a user interface such as is

	disclosed by Mihaylov. STRATEGY-1007, FIGS. 10-12, ¶¶[0050]-[0051].
--	---