

**SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR GENERATING
COUNTERFACTUAL INSTANCES**CROSS REFERENCE TO RELATED APPLICATION

[0001] This application claims priority to United States Provisional Patent Application No. 63/523,959, filed June 29, 2023, the disclosure of which is hereby incorporated by reference in its entirety.

BACKGROUND

1. Technical Field

[0002] This disclosure relates generally to machine-learning and, in some non-limiting embodiments or aspects, to systems, methods, and computer program products for generating counterfactual instances.

2. Technical Considerations

[0003] Detecting anomalies is a significant and challenging task that has various practical applications. Since anomalies are rare occurrences, the labels used for detection often contain noise and are unreliable. Consequently, existing anomaly detection algorithms focus on modeling the majority distribution of data in an unsupervised manner. The objective of such algorithms is to identify data points that deviate from the established distribution, thus indicating anomalies. Given the complex structures present in real-world data, advanced techniques, such as deep neural networks and non-linear kernel methods, have emerged to address this problem.

[0004] However, as the models become more complex, the interpretability decreases. This means that users have limited ability to understand the decisions made by these models or to diagnose any issues they may have. For example, industries like healthcare and finance, which involve high-stakes applications, place great importance on model interpretability.

SUMMARY

[0005] Accordingly, provided are improved systems, methods, and computer program products for generating counterfactual instances.

[0006] According to non-limiting embodiments or aspects, provided is a system comprising at least one processor configured to: generate a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the

plurality of initial classifications is associated with a second class; generate a perturbation vector; perturb the data record based on the perturbation vector to generate a first perturbed data record; generate a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class; select a subset of the first plurality of data records; perturb the subset of the first plurality of data records to generate a plurality of synthesized data records; train a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and generate a second classification for the first perturbed data record based on the second machine-learning model.

[0007] According to non-limiting embodiments or aspects, provided is a computer-implemented method, comprising: generating, with at least one processor, a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class; generating, with at least one processor, a perturbation vector; perturbing, with at least one processor, the data record based on the perturbation vector to generate a first perturbed data record; generating, with at least one processor, a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class; selecting, with at least one processor, a subset of the first plurality of data records; perturbing, with at least one processor, the subset of the first plurality of data records to generate a plurality of synthesized data records; training, with at least one processor, a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and generating, with at least one processor, a second classification for the first perturbed data record based on the second machine-learning model.

[0008] According to non-limiting embodiments or aspects, provided is a system comprising at least one processor configured to: perturb a machine-learning model based on a plurality of input instances based on an outlier data point; determine that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point; in response to determining that the perturbation has failed, generate a plurality of query data points based on the

outlier data point; create a training dataset based on each query data point of the plurality of query data points; train a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models; perturb each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and generate a counterfactual output based on an output of at least one candidate model of the plurality of candidate models. In some non-limiting embodiments or aspects, the at least one processor is further programmed to configured to generate a perturbation vector for each input instance of a plurality of input instances, resulting in the plurality of perturbation vectors.

[0009] In some non-limiting embodiments or aspects, the plurality of query data points are determined based on the outlier data point and a clustering or distance algorithm. In some non-limiting embodiments or aspects, each candidate model of the plurality of candidate models is generated separately, and wherein each subsequent candidate model is generated in response to the counterfactual output of the candidate model resulting in an unchanged predictive output. In some non-limiting embodiments or aspects, the at least one processor is further configured to: determine that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual output changes the predictive output, store the candidate model in a data structure for testing. In some non-limiting embodiments or aspects, the at least one processor is further configured to: determine that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual output changes the predictive output, calculate a certainty of the candidate model by perturbing a next candidate model with a next query point. In some non-limiting embodiments or aspects, the plurality of candidate models are stored in a stack data structure.

[0010] According to non-limiting embodiments or aspects, provided is a computer-implemented method, comprising: perturbing, with at least one processor, a machine-learning model based on a plurality of input instances based on an outlier data point; determining, with at least one processor, that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point; in response to determining that the perturbation has failed,

generating, with at least one processor, a plurality of query data points based on the outlier data point; creating, with at least one processor, a training dataset based on each query data point of the plurality of query data points; training, with at least one processor, a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models; perturbing, with at least one processor, each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and generating, with at least one processor, a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.

[0011] In some non-limiting embodiments or aspects, the method further includes generating a perturbation vector for each input instance of a plurality of input instances, resulting in the plurality of perturbation vectors. In some non-limiting embodiments or aspects, the plurality of query data points are determined based on the outlier data point and a clustering or distance algorithm. In some non-limiting embodiments or aspects, each candidate model of the plurality of candidate models is generated separately, and each subsequent candidate model is generated in response to the counterfactual output of the candidate model resulting in an unchanged predictive output. In some non-limiting embodiments or aspects, the method further includes: determining that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual output changes the predictive output, storing the candidate model in a data structure for testing. In some non-limiting embodiments or aspects, the method further includes: determining that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual output changes the predictive output, calculating a certainty of the candidate model by perturbing a next candidate model with a next query point. In some non-limiting embodiments or aspects, the plurality of candidate models are stored in a stack data structure.

[0012] According to non-limiting embodiments or aspects, provided is a computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to perform any of the steps described herein.

[0013] According to non-limiting embodiments or aspects, provided is a computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: generate a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class; generate a perturbation vector; perturb the data record based on the perturbation vector to generate a first perturbed data record; generate a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class; select a subset of the first plurality of data records; perturb the subset of the first plurality of data records to generate a plurality of synthesized data records; train a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and generate a second classification for the first perturbed data record based on the second machine-learning model.

[0014] According to non-limiting embodiments or aspects, provided is a computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: perturb a machine-learning model based on a plurality of input instances based on an outlier data point; determine that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point; in response to determining that the perturbation has failed, generate a plurality of query data points based on the outlier data point; create a training dataset based on each query data point of the plurality of query data points; train a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models; perturb each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and generate a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.

[0015] Further non-limiting embodiments or aspects will be set forth in the following numbered clauses:

[0016] Clause 1: A system, comprising: at least one processor configured to: generate a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class; generate a perturbation vector; perturb the data record based on the perturbation vector to generate a first perturbed data record; generate a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class; select a subset of the first plurality of data records; perturb the subset of the first plurality of data records to generate a plurality of synthesized data records; train a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and generate a second classification for the first perturbed data record based on the second machine-learning model.

[0017] Clause 2: A computer-implemented method, comprising: generating, with at least one processor, a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class; generating, with at least one processor, a perturbation vector; perturbing, with at least one processor, the data record based on the perturbation vector to generate a first perturbed data record; generating, with at least one processor, a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class; selecting, with at least one processor, a subset of the first plurality of data records; perturbing, with at least one processor, the subset of the first plurality of data records to generate a plurality of synthesized data records; training, with at least one processor, a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and generating, with at least one processor, a second classification for the first perturbed data record based on the second machine-learning model.

[0018] Clause 3: A system, comprising: at least one processor configured to: perturb a machine-learning model based on a plurality of input instances based on an outlier data point; determine that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point; in response to determining that the perturbation has failed, generate a plurality of query data points based on the outlier data point; create a training dataset based on each query data point of the plurality of query data points; train a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models; perturb each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and generate a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.

[0019] Clause 4: The system of clause 3, wherein the at least one processor is further programmed to configured to generate a perturbation vector for each input instance of a plurality of input instances, resulting in the plurality of perturbation vectors.

[0020] Clause 5: The system of clauses 3 or 4, wherein the plurality of query data points are determined based on the outlier data point and a clustering or distance algorithm.

[0021] Clause 6: The system of any of clauses 3-5, wherein each candidate model of the plurality of candidate models is generated separately, and wherein each subsequent candidate model is generated in response to the counterfactual output of the candidate model resulting in an unchanged predictive output.

[0022] Clause 7: The system of any of clauses 3-6, wherein the at least one processor is further configured to: determine that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual output changes the predictive output, store the candidate model in a data structure for testing.

[0023] Clause 8: The system of any of clauses 3-7, wherein the at least one processor is further configured to: determine that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual

output changes the predictive output, calculate a certainty of the candidate model by perturbing a next candidate model with a next query point.

[0024] Clause 9: The system of any of clauses 3-8, wherein the plurality of candidate models are stored in a stack data structure.

[0025] Clause 10: A computer-implemented method, comprising: perturbing, with at least one processor, a machine-learning model based on a plurality of input instances based on an outlier data point; determining, with at least one processor, that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point; in response to determining that the perturbation has failed, generating, with at least one processor, a plurality of query data points based on the outlier data point; creating, with at least one processor, a training dataset based on each query data point of the plurality of query data points; training, with at least one processor, a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models; perturbing, with at least one processor, each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and generating, with at least one processor, a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.

[0026] Clause 11: The method of clause 10, further comprising generating a perturbation vector for each input instance of a plurality of input instances, resulting in the plurality of perturbation vectors.

[0027] Clause 12: The method of clauses 10 or 11, wherein the plurality of query data points are determined based on the outlier data point and a clustering or distance algorithm.

[0028] Clause 13: The method of any of clauses 10-12, wherein each candidate model of the plurality of candidate models is generated separately, and wherein each subsequent candidate model is generated in response to the counterfactual output of the candidate model resulting in an unchanged predictive output.

[0029] Clause 14: The method of any of clauses 10-13, further comprising: determining that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual output changes the predictive output, storing the candidate model in a data structure for testing.

[0030] Clause 15: The method of any of clauses 10-14, further comprising: determining that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and in response to determining that the counterfactual output changes the predictive output, calculating a certainty of the candidate model by perturbing a next candidate model with a next query point.

[0031] Clause 16: The method of any of clauses 10-15, wherein the plurality of candidate models are stored in a stack data structure.

[0032] Clause 17: A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to perform any of the steps recited in clauses 1-16.

[0033] Clause 18: A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: generate a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class; generate a perturbation vector; perturb the data record based on the perturbation vector to generate a first perturbed data record; generate a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class; select a subset of the first plurality of data records; perturb the subset of the first plurality of data records to generate a plurality of synthesized data records; train a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and generate a second classification for the first perturbed data record based on the second machine-learning model.

[0034] Clause 19: A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to: perturb a machine-learning model based on a plurality of input instances based on an outlier data point; determine that the perturbation based on the plurality of input instances

has failed to change a result of the machine-learning model for the outlier data point; in response to determining that the perturbation has failed, generate a plurality of query data points based on the outlier data point; create a training dataset based on each query data point of the plurality of query data points; train a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models; perturb each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and generate a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.

[0035] These and other features and characteristics of the present disclosure, as well as the methods of operation and functions of the related elements of structures and the combination of parts and economies of manufacture, will become more apparent upon consideration of the following description and the appended claims with reference to the accompanying drawings, all of which form a part of this specification, wherein like reference numerals designate corresponding parts in the various figures. It is to be expressly understood, however, that the drawings are for the purpose of illustration and description only and are not intended as a definition of the limits of the disclosed subject matter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0036] Additional advantages and details are explained in greater detail below with reference to the non-limiting, exemplary embodiments that are illustrated in the accompanying schematic figures, in which:

[0037] FIG. 1 is a schematic diagram of an example system for generating counterfactual instances, according to some non-limiting embodiments or aspects;

[0038] FIGS. 2A and 2B are flow diagrams of example methods for generating counterfactual instances, according to some non-limiting embodiments or aspects;

[0039] FIG. 3 is a diagram of an example payment processing network in which methods, systems, and/or computer program products, described herein, may be implemented, according to some non-limiting embodiments or aspects;

[0040] FIG. 4 is a schematic diagram of example components of one or more devices of FIG. 1 and/or FIG. 3, according to some non-limiting embodiments or aspects;

[0041] FIGS. 5A-5D are schematic diagrams of an example implementation of methods for generating counterfactual instances, according to some non-limiting embodiments or aspects; and

[0042] FIGS. 6A-6D are schematic diagrams of an example implementation of methods for generating counterfactual instances, according to some non-limiting embodiments or aspects.

DETAILED DESCRIPTION

[0043] For purposes of the description hereinafter, the terms “end,” “upper,” “lower,” “right,” “left,” “vertical,” “horizontal,” “top,” “bottom,” “lateral,” “longitudinal,” and derivatives thereof shall relate to the embodiments as they are oriented in the drawing figures. However, it is to be understood that the present disclosure may assume various alternative variations and step sequences, except where expressly specified to the contrary. It is also to be understood that the specific devices and processes illustrated in the attached drawings, and described in the following specification, are simply exemplary and non-limiting embodiments or aspects of the disclosed subject matter. Hence, specific dimensions and other physical characteristics related to the embodiments or aspects disclosed herein are not to be considered as limiting.

[0044] Some non-limiting embodiments or aspects are described herein in connection with thresholds. As used herein, satisfying a threshold may refer to a value being greater than the threshold, more than the threshold, higher than the threshold, greater than or equal to the threshold, less than the threshold, fewer than the threshold, lower than the threshold, less than or equal to the threshold, equal to the threshold, etc.

[0045] No aspect, component, element, structure, act, step, function, instruction, and/or the like used herein should be construed as critical or essential unless explicitly described as such. Also, as used herein, the articles “a” and “an” are intended to include one or more items and may be used interchangeably with “one or more” and “at least one.” Furthermore, as used herein, the term “set” is intended to include one or more items (e.g., related items, unrelated items, a combination of related and unrelated items, and/or the like) and may be used interchangeably with “one or more” or “at least one.” Where only one item is intended, the term “one” or similar language is used. Also, as used herein, the terms “has,” “have,” “having,” or the like are intended to be open-ended terms. Further, the phrase “based on” is intended to mean “based at least partially on” unless explicitly stated otherwise. In addition, reference to an

action being “based on” a condition may refer to the action being “in response to” the condition. For example, the phrases “based on” and “in response to” may, in some non-limiting embodiments or aspects, refer to a condition for automatically triggering an action (e.g., a specific operation of an electronic device, such as a computing device, a processor, and/or the like).

[0046] As used herein, the term “acquirer institution” may refer to an entity licensed and/or approved by a transaction service provider to originate transactions (e.g., payment transactions) using a payment device associated with the transaction service provider. The transactions the acquirer institution may originate may include payment transactions (e.g., purchases, original credit transactions (OCTs), account funding transactions (AFTs), and/or the like). In some non-limiting embodiments or aspects, an acquirer institution may be a financial institution, such as a bank. As used herein, the term “acquirer system” may refer to one or more computing devices operated by or on behalf of an acquirer institution, such as a server computer executing one or more software applications.

[0047] As used herein, the term “account identifier” may include one or more primary account numbers (PANs), tokens, or other identifiers associated with a customer account. The term “token” may refer to an identifier that is used as a substitute or replacement identifier for an original account identifier, such as a PAN. Account identifiers may be alphanumeric or any combination of characters and/or symbols. Tokens may be associated with a PAN or other original account identifier in one or more data structures (e.g., one or more databases, and/or the like) such that they may be used to conduct a transaction without directly using the original account identifier. In some examples, an original account identifier, such as a PAN, may be associated with a plurality of tokens for different individuals or purposes.

[0048] As used herein, the term “application program interface (API)” may refer to computer code or other data stored on a computer-readable medium that may be executed by a processor to facilitate the interaction between software components, such as a client-side front-end and/or server-side back-end for receiving data from the client. An “interface” refers to a generated display, such as one or more graphical user interfaces (GUIs) with which a user may interact, either directly or indirectly (e.g., through a keyboard, mouse, etc.).

[0049] As used herein, the terms “client” and “client device” may refer to one or more client-side devices or systems (e.g., remote from a transaction service provider)

used to initiate or facilitate a transaction (e.g., a payment transaction). As an example, a “client device” may refer to one or more POS devices used by a merchant, one or more acquirer host computers used by an acquirer, one or more mobile devices used by a user, and/or the like. In some non-limiting embodiments or aspects, a client device may be an electronic device configured to communicate with one or more networks and initiate or facilitate transactions. For example, a client device may include one or more computers, portable computers, laptop computers, tablet computers, mobile devices, cellular phones, wearable devices (e.g., watches, glasses, lenses, clothing, and/or the like), PDAs, and/or the like. Moreover, a “client” may also refer to an entity (e.g., a merchant, an acquirer, and/or the like) that owns, utilizes, and/or operates a client device for initiating transactions (e.g., for initiating transactions with a transaction service provider).

[0050] As used herein, the term “communication” may refer to the reception, receipt, transmission, transfer, provision, and/or the like of data (e.g., information, signals, messages, instructions, commands, and/or the like). For one unit (e.g., a device, a system, a component of a device or system, combinations thereof, and/or the like) to be in communication with another unit means that the one unit is able to directly or indirectly receive information from and/or transmit information to the other unit. This may refer to a direct or indirect connection (e.g., a direct communication connection, an indirect communication connection, and/or the like) that is wired and/or wireless in nature. Additionally, two units may be in communication with each other even though the information transmitted may be modified, processed, relayed, and/or routed between the first and second unit. For example, a first unit may be in communication with a second unit even though the first unit passively receives information and does not actively transmit information to the second unit. As another example, a first unit may be in communication with a second unit if at least one intermediary unit processes information received from the first unit and communicates the processed information to the second unit. In some non-limiting embodiments or aspects, a message may refer to a network packet (e.g., a data packet and/or the like) that includes data. It will be appreciated that numerous other arrangements are possible.

[0051] As used herein, the term “computing device” may refer to one or more electronic devices configured to process data. A computing device may, in some examples, include the necessary components to receive, process, and output data,

such as a processor, a display, a memory, an input device, a network interface, and/or the like. A computing device may be a mobile device. As an example, a mobile device may include a cellular phone (e.g., a smartphone or standard cellular phone), a portable computer, a wearable device (e.g., watches, glasses, lenses, clothing, and/or the like), a personal digital assistant (PDA), and/or other like devices. A computing device may also be a desktop computer or other form of non-mobile computer.

[0052] As used herein, the terms “electronic wallet” and “electronic wallet application” refer to one or more electronic devices and/or software applications configured to initiate and/or conduct payment transactions. For example, an electronic wallet may include a mobile device executing an electronic wallet application, and may further include server-side software and/or databases for maintaining and providing transaction data to the mobile device. An “electronic wallet provider” may include an entity that provides and/or maintains an electronic wallet for a customer, such as Google Pay®, Android Pay®, Apple Pay®, Samsung Pay®, and/or other like electronic payment systems. In some non-limiting examples, an issuer bank may be an electronic wallet provider.

[0053] As used herein, the term “issuer institution” may refer to one or more entities, such as a bank, that provide accounts to customers for conducting transactions (e.g., payment transactions), such as initiating credit and/or debit payments. For example, an issuer institution may provide an account identifier, such as a PAN, to a customer that uniquely identifies one or more accounts associated with that customer. The account identifier may be embodied on a portable financial device, such as a physical financial instrument, e.g., a payment card, and/or may be electronic and used for electronic payments. The term “issuer system” refers to one or more computer devices operated by or on behalf of an issuer institution, such as a server computer executing one or more software applications. For example, an issuer system may include one or more authorization servers for authorizing a transaction.

[0054] As used herein, the term “merchant” may refer to an individual or entity that provides goods and/or services, or access to goods and/or services, to customers based on a transaction, such as a payment transaction. The term “merchant” or “merchant system” may also refer to one or more computer systems operated by or on behalf of a merchant, such as a server computer executing one or more software applications.

[0055] As used herein, a “point-of-sale (POS) device” may refer to one or more devices, which may be used by a merchant to conduct a transaction (e.g., a payment transaction) and/or process a transaction. For example, a POS device may include one or more client devices. Additionally or alternatively, a POS device may include peripheral devices, card readers, scanning devices (e.g., code scanners), Bluetooth® communication receivers, near-field communication (NFC) receivers, radio frequency identification (RFID) receivers, and/or other contactless transceivers or receivers, contact-based receivers, payment terminals, and/or the like. As used herein, a “point-of-sale (POS) system” may refer to one or more client devices and/or peripheral devices used by a merchant to conduct a transaction. For example, a POS system may include one or more POS devices and/or other like devices that may be used to conduct a payment transaction. In some non-limiting embodiments or aspects, a POS system (e.g., a merchant POS system) may include one or more server computers configured to process online payment transactions through webpages, mobile applications, and/or the like.

[0056] As used herein, the term “payment device” may refer to a payment card (e.g., a credit or debit card), a gift card, a smartcard, smart media, a payroll card, a healthcare card, a wristband, a machine-readable medium containing account information, a keychain device or fob, an RFID transponder, a retailer discount or loyalty card, a cellular phone, an electronic wallet mobile application, a personal digital assistant (PDA), a pager, a security card, a computing device, an access card, a wireless terminal, a transponder, and/or the like. In some non-limiting embodiments or aspects, the payment device may include volatile or non-volatile memory to store information (e.g., an account identifier, a name of the account holder, and/or the like).

[0057] As used herein, the term “payment gateway” may refer to an entity and/or a payment processing system operated by or on behalf of such an entity (e.g., a merchant service provider, a payment service provider, a payment facilitator, a payment facilitator that contracts with an acquirer, a payment aggregator, and/or the like), which provides payment services (e.g., transaction service provider payment services, payment processing services, and/or the like) to one or more merchants. The payment services may be associated with the use of portable financial devices managed by a transaction service provider. As used herein, the term “payment gateway system” may refer to one or more computer systems, computer devices,

servers, groups of servers, and/or the like, operated by or on behalf of a payment gateway.

[0058] As used herein, the term “server” may refer to or include one or more computing devices that are operated by or facilitate communication and processing for multiple parties in a network environment, such as the Internet, although it will be appreciated that communication may be facilitated over one or more public or private network environments and that various other arrangements are possible. Further, multiple computing devices (e.g., servers, point-of-sale (POS) devices, mobile devices, etc.) directly or indirectly communicating in the network environment may constitute a “system.”

[0059] As used herein, the term “system” may refer to one or more computing devices or combinations of computing devices (e.g., processors, servers, client devices, software applications, components of such, and/or the like). Reference to “a device,” “a server,” “a processor,” and/or the like, as used herein, may refer to a previously-recited device, server, or processor that is recited as performing a previous step or function, a different device, server, or processor, and/or a combination of devices, servers, and/or processors. For example, as used in the specification and the claims, a first device, a first server, or a first processor that is recited as performing a first step or a first function may refer to the same or different device, server, or processor recited as performing a second step or a second function.

[0060] As used herein, the term “transaction service provider” may refer to an entity that receives transaction authorization requests from merchants or other entities and provides guarantees of payment, in some cases through an agreement between the transaction service provider and an issuer institution. For example, a transaction service provider may include a payment network such as Visa® or any other entity that processes transactions. The term “transaction processing system” may refer to one or more computer systems operated by or on behalf of a transaction service provider, such as a transaction processing server executing one or more software applications. A transaction processing server may include one or more processors and, in some non-limiting embodiments or aspects, may be operated by or on behalf of a transaction service provider.

[0061] Non-limiting embodiments or aspects of the disclosed subject matter are directed to methods, systems, and computer program products for generating counterfactual instances, including, but not limited to, generating counterfactual

instances based on instance-wise and model-wise perturbation. For example, non-limiting embodiments or aspects of the disclosed subject matter provide generating a plurality of initial classifications for a plurality of data records (e.g., instances) based on a first machine-learning model (e.g., target model). The data records may include a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class (e.g., non-anomalous, non-outlier, approved, and/or the like) and a data record for which a respective initial classification is associated with a second class (e.g., anomalous, outlier, denied, and/or the like). For instance-wise perturbation, a perturbation vector may be generated, and the data record (e.g., instance) may be perturbed based on the perturbation vector to generate a perturbed data record. A first classification for the first perturbed data record may be generated based on the first machine-learning model. For example, the first classification may be associated with the second class (e.g., the output of the target model failed to flip as a result of instance-wise perturbation). For model-wise perturbation, a subset of the first plurality of data records may be selected and perturbed to generate a plurality of synthesized data records, and a second machine-learning model (e.g., candidate model) may be trained based on the subset of the first plurality of data records and the plurality of synthesized data records. Thereafter, a second classification may be generated for the first perturbed data record based on the second machine-learning model. Additionally or alternatively, the process of instance-wise perturbation and/or model-wise perturbation may repeat until a termination condition is satisfied (e.g., the classification of the most recent perturbed data record is flipped). For example, the most recent perturbed data record (e.g., resulting in flipping the classification) may be a counterfactual instance and/or may be used to generate a counterfactual instance. As such, the disclosed subject matter provides using a unique and unconventional combination of instance-wise and model-wise perturbation to generate counterfactual instances. As described herein, the disclosed subject matter improves the validity of generating counterfactual instances compared to other techniques. Moreover, generating counterfactual instances as described herein increases the interpretability of machine learning models (e.g., complex and/or deep machine learning models). For example, by generative valid counterfactual instances, the behavior of the model may be interpreted and/or explained based on the difference(s) between the original data record and the counterfactual instance (e.g., most recent perturbed data record). This

may allow users to have improved ability to understand the decisions made by the model and/or to diagnose issues with the model. For the purpose of illustration, in industries like healthcare and finance, which involve high-stakes applications, model interpretability may be of great importance and help to inform decision making.

[0062] Referring to FIG. 1, a system 100 for generating counterfactual instances is shown according to some non-limiting embodiments or aspects. The system 100 may include model data 104, which may be one or more data storage devices having one or more machine-learning models stored thereon. The model(s) may have been previously executed during production. For example, the model(s) may be configured to generate predicted classifications for inputs (e.g., transaction data, account data, and/or the like). The outputs of the model(s) may also be stored as model data 104 and/or may be stored in one or more other data storage devices.

[0063] With continued reference to FIG. 1, a perturbation engine 101 may be in communication with the model data 104. The perturbation engine 101 may include one or more computing devices and/or software applications executed by one or more computing devices. The perturbation engine 101 may be configured to perturb one or more data instances and/or to perturb one or more machine-learning models based on data instances (e.g., input instances 118). For example, a plurality of input instances 118 may be based on an outlier data point from previous executions of the machine-learning model(s). For example, a previous execution of a model may have one or more data points that was classified in a particular way despite being an outlier (e.g., having at least a threshold distance from other data points similarly classified).

[0064] Still referring to FIG. 1, if the perturbation engine 101 fails to flip (e.g., change) an output (e.g., predicted classification) of one of the model(s), the perturbation engine 101 and/or another computing device may generate a plurality of query data points 116 (e.g., based on the outlier data point and/or based on non-outlier data points). For example, query data points 116 may be based on the outlier data point, such as one or more nodes identified based on a clustering algorithm, a distance algorithm, and/or the like. A training dataset 108 may then be created for each query of the query data points 116. Based on the training dataset 108, at least one candidate model 106 (e.g., a plurality of candidate models 106) may be generated and trained by the training engine 102. The training engine 102 may include one or more computing devices and/or software applications executed by one or more computing devices. The candidate models 106 may be based on the machine-learning model(s)

from the model data 104. The training engine 102, perturbation engine 101, and/or another system or application may generate the candidate models 106 in non-limiting embodiments.

[0065] The perturbation engine 101 may then perturb the candidate models 106 based on perturbation vectors. The perturbation engine 101 may perturb each candidate model of a plurality of candidate models 106 until it flips (e.g., changes) a predictive classification output of the model as compared to an output of the model(s) it was derived from in the model data 104. The plurality of candidate models may be perturbed successively, from a stack of candidate models, until a candidate model is identified that flips the predictive classification output.

[0066] The number and arrangement of components shown in FIG. 1 are provided as an example. There may be additional components, fewer components, different components, and/or differently arranged components than those shown in FIG. 1. Furthermore, two or more components shown in FIG. 1 may be implemented within a single component, or a single component shown in FIG. 1 may be implemented as multiple, distributed components. Additionally or alternatively, a set of components (e.g., one or more components) of system 100 may perform one or more functions described as being performed by another set of components of system 100.

[0067] Referring now to FIG. 2A, shown is a flow diagram for a method 200a for generating counterfactual instances, according to some non-limiting embodiments or aspects. The steps shown in FIG. 2A are for example purposes only. It will be appreciated that additional, fewer, different, and/or a different order of steps may be used in some non-limiting embodiments or aspects. In some non-limiting embodiments or aspects, a step may be automatically performed in response to performance and/or completion of a prior step. In some non-limiting embodiments or aspects, one or more of the steps of method 200a may be performed (e.g., completely, partially, and/or the like) by perturbation engine 101. In some non-limiting embodiments or aspects, one or more of the steps of method 200a may be performed (e.g., completely, partially, and/or the like) by another system, another device, another group of systems, or another group of devices, separate from or including perturbation engine 101, such as training engine 102 and/or the like.

[0068] As shown in FIG. 2A, at step 202, process 200 may include generating initial classifications based on a target model. For example, perturbation engine 101 may

generate a plurality of initial classifications for a plurality of data records (e.g., input instances 118) based on a target model (e.g., a first machine-learning model).

[0069] In some non-limiting embodiments or aspects, the data records (e.g., input instances 118) may include a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class (e.g., non-anomalous, non-outlier, approved, and/or the like) and at least one data record (e.g., a second plurality of data records) for which a respective initial classification of the plurality of initial classifications is associated with a second class (e.g., anomalous, outlier, denied, and/or the like).

[0070] In some non-limiting embodiments or aspects, the target model may include at least one of a neural network, a convolutional neural network (CNN), a recurrent neural network (RNN), an autoencoder (or portion thereof), a long short-term memory (LSTM), a bidirectional LSTM, an attention network, a transformer, a tree-based classifier, a random forest, a gradient boosting classifier, a support vector machine (SVM), a one-class support vector machine (OCSVM), a deep support vector data description (DeepSVDD), any combination thereof, and/or the like.

[0071] For the purpose of illustration, with reference to FIGS. 5A and 6A and with continued reference to FIG. 2A, perturbation engine 101 may generate a plurality of initial classifications for a plurality of data records (e.g., first data records 518a and second data records 518b, collectively referred to as “data records 518,” and individually referred to as “data record 518”) based on target model 504. In some non-limiting embodiments or aspects, the data records 518 may include first data records 518a for which respective initial classifications of the plurality of initial classifications are associated with a first class (e.g., non-anomalous, non-outlier, approved, and/or the like) and at least one data record 518 (e.g., second data records 518b) for which a respective initial classification of the plurality of initial classifications is associated with a second class (e.g., anomalous, outlier, denied, and/or the like).

[0072] Referring again to FIG. 2A, at step 204, process 200 may include generating a perturbation vector 519. For example, perturbation engine 101 may generate a perturbation vector.

[0073] For the purpose of illustration, with reference to FIGS. 5A and 6A and with continued reference to FIG. 2A, perturbation engine 101 may generate a perturbation vector 519.

[0074] Referring again to FIG. 2A, at step 206, process 200 may include perturbing a data record based on the perturbation vector. For example, perturbation engine 101 may perturb a data record (e.g., a selected data record associated with the second classification, such as anomalous, outlier, denied, and/or the like) based on the perturbation vector to generate a perturbed data record.

[0075] For the purpose of illustration, with reference to FIGS. 5A and 6A and with continued reference to FIG. 2A, perturbation engine 101 may perturb one of the second data records 518b based on the perturbation vector 519 to generate perturbed data record 520.

[0076] Referring again to FIG. 2A, at step 208, process 200 may include generating a classification for the perturbed data record. For example, perturbation engine 101 may generate a classification for the perturbed data record based on the target model (e.g., first machine-learning model).

[0077] In some non-limiting embodiments or aspects, the classification may be associated with the first class (e.g., non-anomalous, non-outlier, approved, and/or the like). For example, the perturbation of the data record may have resulted in flipping the output generated by the target model from the second class (before perturbation) to the first class (e.g., after perturbation). As such, the perturbed data record may be a counterfactual instance (and/or may be used to generate a counterfactual instance/output). In some non-limiting embodiments or aspects, process 200a may terminate if the perturbation of the data record successfully flipped the output of the target model.

[0078] In some non-limiting embodiments or aspects, the classification may be associated with the second class (e.g., anomalous, outlier, denied, and/or the like). For example, the perturbation of the data record may have failed to flip the output generated by the target model. As such, the perturbed data record may not be a counterfactual instance (and/or may not be used to generate a counterfactual instance/output). In some non-limiting embodiments or aspects, process 200a may proceed to step 210 if the perturbation of the data record failed to flip the output of the target model.

[0079] For the purpose of illustration, with reference to FIGS. 5A and 6A and with continued reference to FIG. 2A, perturbation engine 101 may generate a classification for perturbed data record 520 based on the target model 504. As shown in FIGS. 5A

and 6A, perturbed data record 520 may still be outside the boundary of target model 504 such that perturbed data record 520 is still associated with the second class.

[0080] Referring again to FIG. 2A, at step 210, process 200 may include selecting a subset of data records. For example, perturbation engine 101 may select a subset of the data records (e.g., a subset of input instances 118).

[0081] In some non-limiting embodiments or aspects, perturbation engine 101 may select a target number (e.g., k) of data records that are the nearest neighbors to the perturbed data record (e.g., k -nearest neighbors).

[0082] For the purpose of illustration, with reference to FIGS. 5B and 6B and with continued reference to FIG. 2A, perturbation engine 101 may select a subset of first data records 518a. For example, as shown in FIGS. 5B and 6B perturbation engine 101 may select the four first data records 518a that are nearest neighbors to perturbed data record 520.

[0083] Referring again to FIG. 2A, at step 212, process 200 may include perturbing the subset. For example, perturbation engine 101 may perturb the (selected) subset of the first plurality of data records to generate a plurality of synthesized data records (e.g., query data points 116).

[0084] For the purpose of illustration, with reference to FIGS. 5B and 6B and with continued reference to FIG. 2A, perturbation engine 101 may perturb the (selected) subset of first data records 518a to generate a plurality of synthesized data records 516.

[0085] Referring again to FIG. 2A, at step 214, process 200 may include training a candidate machine-learning model. For example, perturbation engine 101 and/or training engine 102 may train a candidate model (e.g., a second machine-learning model) based on the subset of the first plurality of data records (e.g., subset of input instances 118) and the plurality of synthesized data records (e.g., query data points 116). For example, the subset of input instances 118 and the query data points 116 may be collectively referred to as training dataset 108.

[0086] In some non-limiting embodiments or aspects, the candidate model may include at least one of a neural network, a convolutional neural network (CNN), a recurrent neural network (RNN), an autoencoder (or portion thereof), a long short-term memory (LSTM), a bidirectional LSTM, an attention network, a transformer, a tree-based classifier, a random forest, a gradient boosting classifier, a support vector machine (SVM), a one-class support vector machine (OCSVM), a deep support vector

data description (DeepSVDD), any combination thereof, and/or the like. In some non-limiting embodiments or aspects, the candidate model may be the same type of machine learning model and/or may have the same structure as the target model.

[0087] In some non-limiting embodiments or aspects, training may include inputting each respective data record of the training dataset (e.g., the subset of the first plurality of data records and the plurality of synthesized data records) to the candidate model to generate a predicted output for each respective data record. A loss may be determined based on each respective data record and the respective predicted output. For example, the losses may be calculated based on a loss function (e.g., based the predicted output), a reward function, an error, a mean error, a mean squared error (MSE), any combination thereof, and/or the like. The parameters of the candidate model may be updated based on the loss(es). For example, perturbation engine 101 and/or training engine 102 may update (e.g., adjust) the parameters of the candidate model based on back propagation (e.g., of the loss(es)), gradient calculations (e.g., based on the loss(es)), any combination thereof, and/or the like.

[0088] In some non-limiting embodiments or aspects, training may further include repeating the training process (e.g., repeating inputting each respective data record of the training dataset, determining the loss, and updating the parameters) until a termination condition is satisfied. For example, the termination condition may include a threshold level of accuracy, and the repetition may continue until the threshold level of accuracy is satisfied. Additionally or alternatively, the termination condition may include a target number of repetitions (e.g., a target number of epochs and/or the like), and the repetition may continue until the target number of repetitions is completed. In some non-limiting embodiments or aspects, there may be a plurality of termination conditions, and the repetition may continue until at least one of the plurality of termination conditions is satisfied.

[0089] For the purpose of illustration, with reference to FIGS. 5B and 6B and with continued reference to FIG. 2A, perturbation engine 101 and/or training engine 102 may train candidate model 506 (e.g., first candidate model M1) based on the subset of first data records 518a and the synthesized data records 516.

[0090] In some non-limiting embodiments or aspects, after step 214, method 200a may at least partially repeat. For example, method 200a may include repeating steps 204-214 until a termination condition is reached, as described herein.

[0091] Referring again to FIG. 2A, at step 216, process 200 may include generating a classification for the perturbed data record based on the candidate model. For example, perturbation engine 101 may generate a classification for the perturbed data record based on the candidate model (e.g., second machine-learning model).

[0092] In some non-limiting embodiments or aspects, the classification may be associated with the first class (e.g., non-anomalous, non-outlier, approved, and/or the like). For example, the candidate model may have resulted in flipping the output with respect to the target model from the second class (e.g., as generated by the target model) to the first class (e.g., as generated by the candidate model). As such, in some non-limiting embodiments or aspects, the candidate model may be used to generate a counterfactual instance/output.

[0093] In some non-limiting embodiments or aspects, the classification may be associated with the second class (e.g., anomalous, outlier, denied, and/or the like). For example, the candidate model may have failed to flip the output with respect to the target model.

[0094] For the purpose of illustration, with reference to FIGS. 5B and 6B and with continued reference to FIG. 2A, perturbation engine 101 may generate a classification for perturbed data record 520 based on candidate model 506 (e.g., first candidate model M1). As shown in FIGS. 5B and 6B, perturbed data record 520 may still be outside the boundary of candidate model 506 such that perturbed data record 520 is still associated with the second class.

[0095] In some non-limiting embodiments or aspects, after step 216, method 200a may at least partially repeat. For example, method 200a may include repeating steps 204-216 until a termination condition is reached, as described herein.

[0096] In some non-limiting embodiments or aspects, repeating steps 204 and 206 may be similar to the initial performance of steps 204 and 206, except instead of perturbing the initially selected data record (e.g., selected second data record 518b), perturbation engine 101 may perturb the most recent perturbed data record (e.g., perturbed data record 520).

[0097] For the purpose of illustration, with reference to FIGS. 5B and 6C and with continued reference to FIG. 2A, perturbation engine 101 may generate a (further) perturbation vector 519' (e.g., repeat step 204). Additionally, perturbation engine 101 may perturb the most recent perturbed data record 520 based on the (further)

perturbation vector 519' to generate a (further) perturbed data record 520' (e.g., repeat step 206).

[0098] Referring again to FIG. 2A, repeating step 208 may be similar to the initial performance of step 208, except instead of classifying the (further) perturbed data record 520' based on the target model (e.g., target model 504), the (further) perturbed data record 520' may be classified based on the most recent candidate model (e.g., candidate model 506/first candidate model M1). Additionally or alternatively, repeating steps 210 and 212 may be similar to the initial performance of steps 210 and 212, except instead of selecting the subset from the initial data records (e.g., first data records 518a), the subset may be selected from the training dataset (e.g., the subset of first data records 518a and the synthesized data records 516 that collectively form the training dataset for candidate model 506/first candidate model M1).

[0099] For the purpose of illustration, with reference to FIGS. 5C and 6C and with continued reference to FIG. 2A, perturbation engine 101 may select a subset of the most recent training dataset (e.g., the subset of first data records 518a and the synthesized data records 516 that collectively form the training dataset for candidate model 506/first candidate model M1) (e.g., repeating step 210). Additionally, perturbation engine 101 may perturb the (selected) subset to generate a (further) plurality of synthesized data records 516' (e.g., repeating step 212).

[0100] Referring again to FIG. 2A, repeating step 214 may be similar to the initial performance of step 214, except instead of training the (first) candidate model, perturbation engine 101 and/or training engine 102 may train a (further) candidate model (e.g., a third machine-learning model).

[0101] For the purpose of illustration, with reference to FIGS. 5C and 6C and with continued reference to FIG. 2A, perturbation engine 101 and/or training engine 102 may train (further) candidate model 506' (e.g., second candidate model M2).

[0102] In some non-limiting embodiments or aspects, the repetition of at least some of the steps of method 200a may continue until a termination condition is satisfied. For example, as shown in FIGS. 5C and 6C, after another repetition, the (further) perturbation vector 519' may result in perturbing the most recent (further) perturbed data record 520' across the boundary of second candidate model M2 (e.g., the classification of the resulting further perturbed data record 520' may flip from the second class to the first class). If such a condition is satisfied, in some non-limiting embodiments or aspects, second candidate model M2. For example, as shown in

FIGS. 5D and 6D, second candidate model M2 can be removed and another repetition of generating a (further) perturbation vector 519' from the most recent (further) perturbed data record 520' may be performed. The repetition of at least some of the steps of method 200a may continue until a most recent (further) perturbed data record 520' crosses the boundary of first candidate model M1, which may then be removed. The repetition of at least some of the steps of method 200a may continue until most recent (further) perturbed data record 520' crosses the boundary of target model 504, which may be a termination condition. For example, when this termination condition is satisfied, the most recent (further) perturbed data record 520' may be a counterfactual instance (and/or may be used to generate a counterfactual instance/output).

[0103] In some non-limiting embodiments or aspects, the most recent perturbed data record (e.g., further perturbed data record 520') may be used to generate a counterfactual instance. For example, perturbation engine 101 may generate a message based on the most recent (further) perturbed data record 520'. For the purpose of illustration, the message may include an explanation of behavior of the target model and/or an explanation of what would need to change to cause the target model to generate a different (e.g., flipped) output (e.g., based on the difference between the most recent (further) perturbed data record 520' and the initially selected second data record 518b). For example, if the initially selected second data record 518b was associated with a denied request (e.g., denied access request, denied credit application, denied authorization request, and/or the like), the differences between that initially selected second data record 518b and the most recent (further) perturbed data record 520' may explain what would need to change to result in the request being approved (e.g., approved access request, approved credit application, approved authorization request, and/or the like). As another example, if the initially selected second data record 518b was associated with a detected anomaly (e.g., fraud, attack, malicious network traffic, and/or the like), the differences between that initially selected second data record 518b and the most recent (further) perturbed data record 520' may explain why the initially selected second data record 518b was considered anomalous.

[0104] In some non-limiting embodiments or aspects, perturbation engine 101 may detect an anomaly (e.g., fraud, attack, malicious network traffic, and/or the like) based on the most recent perturbed data sample.

[0105] For the purpose of illustration, with reference to FIGS. 6A-6D and with continued reference to FIG. 2A, in some non-limiting embodiments or aspects, the

target model and candidate model(s) may be stored in a stack (e.g., list, table, data structure, and/or the like). For example, each time a new candidate model is trained, as described herein, the new candidate model may be added to the stack. For example, as shown in FIG. 6B, first candidate model M1 is added to the stack, and as shown in FIG. 6C, second candidate model M2 is added to the stack. When a candidate model is removed, as described herein (e.g., the classification of the most recent perturbed data record flips), the candidate model may be removed from the stack. For example, as shown in FIG. 6D, second candidate model M2 is removed from the stack.

[0106] In some non-limiting embodiments or aspects, another termination condition may include the stack being full of candidate models without achieving flipping of the classification of the most recent perturbed data record. For example, not all data records may be successfully flipped, and limiting the size of the stack may conserve resources and reduce waste.

[0107] For the purpose of illustration, Table 1 shows validity (e.g., number of successfully flipped data records divided by total number of data records) for different types of machine learning models (e.g., random forest, one-class support vector machines (OCSVM), and autoencoder) for the disclosed subject matter (Ours) compared to other counterfactual explanation techniques (e.g., LocalSearchCF, CFProto, and RELACE). As shown in Table 1, the disclosed subject matter outperforms the other techniques.

Table 1

Validity (Pima Dataset)	LocalSearchCF	CFProto	RELACE	Ours
RandomForest	0.38	0.44	0.72 ± 0.08	0.87
OCSVM	0.0	0.0	0.61 ± 0.1	0.70
AutoEncoder	0.0	0.0	0.74 ± 0.21	0.93

[0108] For the purpose of illustration, Table 2 shows validity for different types of machine learning models (e.g., OCSVM, autoencoder, and DeepSVDD) for the disclosed subject matter (Ours) compared to other counterfactual explanation techniques (e.g., LocalSearch, CFProto, and ReLaX) for three different datasets (Vowels, Thyroid, and Mammography). As shown in Table 2, the disclosed subject matter outperforms the other techniques.

Table 2

Dataset	Detector (Validity)	Explainers			
		LocalSearch	CFProto	ReLaX	Ours
Vowels	OCSVM	0.10	0.25	0.95	0.98
	AutoEncoder	0.15	0.26	0.93	1.00
	DeepSVDD	0.30	0.42	0.94	1.00
Thyroid	OCSVM	0.06	0.10	0.06	0.99
	AutoEncoder	0.11	0.22	0.03	0.98
	DeepSVDD	0.04	0.09	0.12	0.99
Mammography	OCSVM	0.15	0.35	0.89	1.00
	AutoEncoder	0.28	0.52	0.92	1.00
	DeepSVDD	0.55	0.63	0.91	0.99

[0109] For the purpose of illustration, Table 3 shows distance for different types of machine learning models (e.g., OCSVM, auto-encoder, and DeepSVDD) for the disclosed subject matter (Ours) compared to other counterfactual explanation techniques (e.g., LocalSeach, CFProto, and ReLaX) for three different datasets (Vowels, Thyroid, and Mammography). As shown in Table 3, the disclosed subject matter performs comparably to or better than the other techniques.

Table 3

Dataset	Detector (Distance)	Explainers			
		LocalSearch	CFProto	ReLaX	Ours
Vowels	OCSVM	7.25	8.51	0.95	0.72
	AutoEncoder	15.52	14.38	6.51	1.64
	DeepSVDD			0.94	1.00
Thyroid	OCSVM	-	-	12.68	7.31
	AutoEncoder	-	-	15.91	8.84
	DeepSVDD	17.81	16.45	11.62	7.13
Mammography	OCSVM	5.94	5.08	1.73	1.28
	AutoEncoder	4.87	3.53	1.59	1.15
	DeepSVDD	6.19	5.91	0.98	1.03

[0110] In some non-limiting embodiments or aspects, an implementation of process 200a may be represented by the following algorithm:

Algorithm 1 Training Procedure

```

1: Input: Training data  $X^{\text{train}}$ , Target Model  $\phi$ , number of training episode  $E$ , DRL training step  $S$ , reward coefficient  $\lambda$ , maximum perturbation iteration  $\hat{t}$ .
2: Initialize actor network  $\pi$ , critic network  $Q$ , memory buffer  $\mathcal{B}$ ,  $\epsilon = 0$ , model stack  $\mathcal{M}$ .
3: Pre-compute  $K$ -nearest neighborhood for  $X^{\text{train}}$ .
4: Push  $\phi$  into  $\mathcal{M}$ 
5: for  $e = 0, 1, 2, \dots, E$  do
6:   for  $i = 0, 1, 2, \dots, |X^{\text{train}}|$  do
7:     Obtain target data sample  $x_i$  from  $X^{\text{train}}$ .
8:      $t \leftarrow 0$ .
9:     Generate initial state  $s_t$ .
10:    while True do
11:      Get the action  $a_t = \pi(s_t)$  from actor network.
12:      Get instance-perturbation vector  $\hat{p}_t$  from  $a_t$ .
13:      Get top-k neighbors of the data sample  $i$  and the directional jitter vector for model-perturbation from  $a_t$ .
14:      Perform instance-perturbation:  $x_t = x_i + \hat{p}_t$ .
15:      while  $\mathcal{M} \neq \text{Empty}$  do
16:        Pop one model  $m$  from  $\mathcal{M}$ .
17:        Examine the validity of the perturbation and compute the perturbation information (e.g., distance between  $x_t$  and  $x_{t-1}$ , size of  $\mathcal{M}$ ...etc).
18:        if not valid perturbation then
19:          Create a proxy model  $m'$  based on the popped model  $m$  with model-wise perturbation.
20:          Push back the popped model  $m$  into  $\mathcal{M}$ .
21:          Push the proxy model  $m'$  into  $\mathcal{M}$ .
22:          Break
23:        end if
24:      end while
25:      Form the reward  $r_t$  with the computed information.
26:      Form the next state  $s_{t+1}$  based on  $x_t$  and  $\mathcal{M}$  and  $\phi$ .
27:      Store the triplet  $T_t = (s_t, a_t, s_{t+1}, r_t)$  into  $\mathcal{B}$ .
28:       $t \leftarrow t + 1$ .
29:      if  $t > \hat{t}$  then
30:        Break
31:      end if
32:    for step = 1, 2, ...,  $S$  do
33:      Use data in  $\mathcal{B}$  to update DRL model [2].
34:    end for
35:  end while
36: end for
37: end for

```

[0111] In some non-limiting embodiments or aspects, at least one counterfactual instance may be used for model explanation by providing example-based reasoning, where the counterfactual instance(s) (e.g., hypothetical data sample(s)) are provided to promote the understandings of model boundaries. For example, given a classifier $f_\theta : \mathcal{X}^d \rightarrow \mathcal{Y}$, generating a counterfactual x^* to explain the classifier may be formulated based on the following equation:

Equation 1

$$x^* = \arg \min_{x \in \mathcal{C}} l(x, q), \quad \text{s.t. } f_\theta(q) \neq f_\theta(x^*)$$

where q is a input query, l is a distance metric and $\mathcal{C}$ is the counterfactual universe that composed of counterfactual instances. For example, based on Equation 1, counterfactual instances may include data samples from certain distributions, which aim at flipping the model decisions, while keeping the similarity to the input query.

[0112] In some non-limiting embodiments or aspects, counterfactual instance generation may include a sequential decision making process that moves the selected data record from one side of the model decision boundary to another side of the boundary via perturbation(s). In some non-limiting embodiments or aspects, a Markov Decision Process (MDP) may formulate sequential decision making process by a quintuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}_T, \mathcal{R}, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are finite sets of states and actions, respectively, $\mathcal{P}_T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}^+$ is the state transition function that maps the current state s , action a and the next state s' to a probability value, $\mathcal{R} : \mathcal{S} \rightarrow \mathbb{R}$ is the immediate reward function that reflects the quality of action a , and $\gamma \in (0, 1)$ is a discount factor.

[0113] For the purpose of illustration, in some non-limiting embodiments or aspects, given a dataset X , an anomaly detection model $f_\theta : \mathcal{X}^d \rightarrow \mathbb{R}^+$, f_θ assesses the likelihood of individual data record (e.g., data points) $x \in X$ being anomalies such that $f_\theta(x) > 0.5$ indicates that the data point x is an anomaly. An example goal of counterfactual instance generation may include generating counterfactual instances $x^* = \pi_\phi(x | f_\theta)$ with a model-agnostic counterfactual generator π_ϕ , for a d -dimensional input data $x \in \mathbb{R}^d$ by minimizing the following counterfactual loss equation:

Equation 2

$$\mathcal{L}_{\text{CF}}(\pi_\phi) = \frac{1}{|X|} \sum_{x \in X} \mathcal{L}_{\text{flip}}(x, x^*, f_\theta) + \mathcal{L}_{\text{prox}}(x, x^*)$$

where $\mathcal{L}_{\text{flip}}$ evaluates successful flips on target model f_θ and $\mathcal{L}_{\text{prox}}$ encourages minimum perturbation for counterfactual generation. In this way, given a selected data record (e.g., query data) q , a valid counterfactual x^* may be obtained when $[f_\theta(q) - 0.5] \cdot [f_\theta(x^*) - 0.5] < 0$, and the corresponding counterfactual explanation may be provided by comparing the differences between x^* and q .

[0114] In some non-limiting embodiments or aspects, an instance-wise feature perturbation may aim at creating a perturbation vector $p \in \mathbb{R}^d$ that is added to the input x . At each time step t , the counterfactual generator π_ϕ may make an observation from both target model and an input instance x_{t-1} to generate a perturbation vector p_t . The perturbed instance is then obtained as $x_t = x_{t-1} + p_t$, where $x_t = q$ and $p_t = 0^d$ when $t = 0$. The datapoint x_t may then be fed to the target model f_θ to compute $\mathcal{L}_{\text{flip}}$, which may be represented by the following equation:

Equation 3

$$\mathcal{L}_{\text{flip}} = \mathbb{1}_{\{f_\theta(x_t) \neq f_\theta(q)\}}$$

where $\mathbb{1}_{\{\cdot\}}$ is an indicator function that assigns 1 if $f_\theta(x_t) \neq f_\theta(q)$ and 0 otherwise. In some non-limiting embodiments or aspects, the process of instance perturbation may be continued until the counterfactual outcome is obtained (i.e., $\mathcal{L}_{\text{flip}} = 1$) or step constraint is met.

[0115] In some non-limiting embodiments or aspects, a counterfactual with reduced (e.g., minimum) proximity to the target instance (e.g., selected data record) may be more useful than the ones with larger proximity. To achieve this, the distance metric $\mathcal{L}_{\text{prox}}$ may be formulated to encourage reduced (e.g., minimum) distance between the perturbed instance and the target instance. However, in some non-limiting embodiments or aspects, the out-of-distribution space may be enormous, while the number of anomalies may be extremely small in the dataset. Directly posing proximity limitation with an indicator objective may lead to extremely sparse data. This may make it difficult to train a counterfactual generator since the constraint of minimum proximity between a counterfactual and the target instance leads to a counterfactual instance approaching the decision boundary of the target model. Additionally, the data

points that lie near the decision boundary are often with higher uncertainty (i.e., lower prediction confidence). To overcome these issues, a model-aware $\mathcal{L}_{\text{prox}}$ may have prediction confidence quantification of an anomaly detector, for example, as represented in the following equation:

Equation 4

$$\mathcal{L}_{\text{prox}} = \alpha \left(-\frac{1}{n} \sum_{i=1}^n f_{\theta}(x_i) \cdot (1 - f_{\theta}(x_i)) \right) + (1 - \alpha) \cdot d(q, x_t)$$

where $x_i = x_t + \eta$ is a jittered x_t to provide a generalized proximity between the decision boundary of f_{θ} and the x_t where $\eta \in \mathbb{R}^d$ is a random noise, $d(q, x_t)$ measures the instance proximity between x_t and q with Euclidean distance, and α controls the weight of the two proximity. Here, higher α encourages the generation of counterfactual that is near the decision boundary of the target model f_{θ} while lower α leads to a counterfactual instance that is more similar to the target instance.

[0116] In some non-limiting embodiments or aspects, instance-wise perturbation as described herein may be effective when the target model is capable of generating meaningful prediction confidence. However, it may require the target instance to be sufficiently close to the decision boundary of the target model in the feature space. Otherwise, the target model may constantly generate extreme values as the prediction confidence and lead to a noisy $\mathcal{L}_{\text{prox}}$. Additionally, model-wise perturbation may guide the instance-perturbation procedure by creating a series of candidate models that may be derived from the target model. For example, a candidate model of f_{θ} may be created with training dataset generation (e.g., dataset collection) and candidate model training, as described herein.

[0117] In some non-limiting embodiments or aspects, training dataset generation (e.g., dataset collection) may include creating a realistic-synthetic dataset for training the candidate model. For example, in each iteration, the k -nearest neighbors of the selected data record (e.g., query instance) that are detected as normalities (e.g., classified in the first class, not abnormal, and/or the like) may be sampled to serve as base instances. Then, a synthesizing direction may be determined based on subtracting the query instance with the sampled neighbors. The sampled dataset (e.g., subset) may be augmented by adding a vector of jitters with the synthesizing directions to the sampled neighbors (e.g., to generate synthetic data records/instances). For example, the synthetic instances may be generated based on the following equation:

Equation 5

$$x_{\text{syn}} = x + \lambda \cdot \text{sgn}(q - x)$$

where x is the k -nearest neighbor of the query instance q and λ is the size of jitter that controls the level of deviation of the x_{syn} from the neighbor x .

[0118] In some non-limiting embodiments or aspects, candidate model training may include using the collected synthetic dataset to train the candidate model, as described herein. In some non-limiting embodiments or aspects, to initialize the candidate model, the target model may be duplicated. In each iteration, the candidate model f_{θ_n} may be initialized with the latest candidate $f_{\theta_{n-1}}$. This approach may avoid training from scratch, which may help to preserve the predictive power of the target model to the maximum extent possible. The candidate model may be trained using the collected data (e.g., training dataset), for example, to shift the decision boundary of the candidate model (e.g., with respect to the target model or latest candidate model from the previous iteration).

[0119] In some non-limiting embodiment or aspects, the search for a counterfactual instance may involve consecutive perturbations, which may be seen as a sequential decision-making problem, which can be formulated using an MDP. For example, the objective of the MDP may be to incorporate perturbations at both the instance and model levels, enabling the counterfactual generator π to learn the perturbation strategy. For the purpose of illustration, the MDP may be designed for a reinforcement learning (RL) setting, where an RL agent is employed to observe the current state s_t in a timestamp t that reflects the status of perturbation and make a corresponding decision a_t to perform the next perturbation that possibly leads to a successful counterfactual instance x^* .

[0120] In some non-limiting embodiment or aspects, with respect to the state, at time step t , the state $s_t \in \mathcal{S}$ may aim to reflect the status of both instance-wise and model-wise perturbation. To describe the status of model-wise perturbation, a stack may be maintained to store the candidate models, as described herein. For example, the bottom of the model stack may be the target model f_{θ} , and the top of the stack may be the latest perturbed model f_{θ_t} that has failed to flip the perturbed instance x_t . In some non-limiting embodiment or aspects, two attributes of the model stack may be provided in the state s_t : length of the model stack L_{stack} and the distance between the top and the bottom model, where model distance $\cos(f_{\theta_t}, f_{\theta})$ may be computed based

on the cosine similarity between the prediction probability of two models on the k -nearest neighbors of the query instance q in X , for example, as set forth in the following equation:

Equation 6

$$\cos(f_{\theta_t}, f_{\theta}) = \frac{\sum_{x_i \in X} f_{\theta_t}(x_i) \cdot f_{\theta}(x_i)}{\sqrt{\sum_{x_i \in X} f_{\theta_t}(x_i)^2} \cdot \sqrt{\sum_{x_i \in X} f_{\theta}(x_i)^2}}$$

[0121] In some non-limiting embodiments or aspects, to reflect the status of instance-wise perturbation, the latest status of the perturbed instance x_t and the aggregation of all perturbations $p_{\text{agg}} = x_t - q$ may be used. Additionally, $\mathcal{L}_{\text{prox}}$ may be computed to provide the perturbed instance's locality information with respect to the target model. As such, the state $s_t \in \mathbb{R}^{2d+3}$ in time step t may be determined as $s_t = (x_t, p_{\text{agg}}, \mathcal{L}_{\text{prox}}, l_{\text{stack}}, \cos(f_{\theta_t}, f_{\theta}))$.

[0122] In some non-limiting embodiment or aspects, with respect to the action, to drive the perturbation process for the numerical features, the action a_t at time step t may control both the instance-wise perturbation and model-wise perturbation. For the instance-wise perturbation, the RL agent may directly generate a continuous perturbation vector p_t . For the model-wise perturbation, the RL agent may control the k -nearest neighbor of the query instance q to collect data for training candidate models. As such, the action a_t at time step t may be defined as $a_t = (p_t, k)$.

[0123] In some non-limiting embodiment or aspects, with respect to the reward, a reward function may be tailored to facilitate both model-wise and instance-wise perturbation for counterfactual instance generation. For the purpose of illustration, the reward r_t for timestamp t may be determined based on the following equation:

Equation 7

$$r_t = \begin{cases} \frac{\mu - [d(q, x_t) - d(q, x_{t-1})]}{l_{\text{stack}}}, & \text{if } f_{\theta}(q) \neq f_{\theta}(x), \\ -[d(q, x_t) - d(q, x_{t-1})], & \text{otherwise.} \end{cases}$$

[0124] In some non-limiting embodiments or aspects, to reduce (e.g., minimize) the instance-wise perturbation, the distances between the query instance and the perturbed instance may be computed. Then, the differences between the distances from current perturbation and previous perturbation may be computed to minimize the perturbation cost before finding the counterfactual instance. When the counterfactual

instance is found, a threshold μ may be used to subtract the instance-wise perturbation cost and the cost may be divided with the size of model stack to encourage minimizing the model perturbation.

[0125] Referring now to FIG. 2B, shown is a flow diagram for a method 200b for generating counterfactual instances, according to some non-limiting embodiments or aspects. The steps shown in FIG. 2B are for example purposes only. It will be appreciated that additional, fewer, different, and/or a different order of steps may be used in some non-limiting embodiments or aspects. In some non-limiting embodiments or aspects, a step may be automatically performed in response to performance and/or completion of a prior step. In some non-limiting embodiments or aspects, one or more of the steps of method 200b may be performed (e.g., completely, partially, and/or the like) by perturbation engine 101. In some non-limiting embodiments or aspects, one or more of the steps of method 200b may be performed (e.g., completely, partially, and/or the like) by another system, another device, another group of systems, or another group of devices, separate from or including perturbation engine 101, such as training engine 102 and/or the like.

[0126] As shown in FIG. 2B, at step 222, method 200b may include perturbing a machine-learning model. For example, perturbation engine 101 may perturb a machine-learning model (e.g., associated with model data 104) based on a plurality of input instances 118 based on an outlier data point, as described herein.

[0127] As shown in FIG. 2B, at step 224, method 200b may include determining perturbation failed to change a result for the outlier data point. For example, perturbation engine 101 may determine that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point, as described herein.

[0128] As shown in FIG. 2B, at step 226, method 200b may include generating a plurality of query data points. For example, in response to determining that the perturbation has failed, perturbation engine 101 may generate a plurality of query data points 116 based on the outlier data point, as described herein.

[0129] In some non-limiting embodiments or aspects, the plurality of query data points may be determined based on the outlier data point and a clustering or distance algorithm.

[0130] As shown in FIG. 2B, at step 228, method 200b may include creating a training dataset based on each query data point. For example, perturbation engine

101 and/or training engine 102 may create a training dataset 108 based on each query data point 116 of the plurality of query data points 116, as described herein.

[0131] As shown in FIG. 2B, at step 230, method 200b may include training candidate model(s) based on the training dataset. For example, perturbation engine 101 and/or training engine 102 may train a candidate model 106 derived from the machine-learning model based on the training dataset 108, resulting in a plurality of candidate models 106.

[0132] In some non-limiting embodiments or aspects, each candidate model 106 of the plurality of candidate models 106 may be generated separately. For example, each subsequent candidate model may be generated in response to the counterfactual output of the candidate model resulting in an unchanged predictive output.

[0133] In some non-limiting embodiments or aspects, the plurality of candidate models are stored in a stack data structure.

[0134] As shown in FIG. 2B, at step 232, method 200b may include perturbing the candidate model(s). For example, perturbation engine 101 may perturb each candidate model 106 of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors, as described herein.

[0135] In some non-limiting embodiments or aspects, perturbation engine 101 may generate a perturbation vector for each input instance of a plurality of input instances, resulting in the plurality of perturbation vectors.

[0136] As shown in FIG. 2B, at step 234, method 200b may include generating a counterfactual output. For example, perturbation engine 101 may generate a counterfactual output based on an output of at least one candidate model 106 of the plurality of candidate models, as described herein.

[0137] In some non-limiting embodiments or aspects, perturbation engine 101 may determine that a counterfactual output of at least one candidate model 106 of the plurality of candidate models 106 changes a predictive output of the machine-learning model. In response to determining that the counterfactual output changes the predictive output, perturbation engine 101 may store the candidate model in a data structure for testing.

[0138] In some non-limiting embodiments or aspects, perturbation engine 101 may determine that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model. In response to determining that the counterfactual output changes the predictive output,

perturbation engine 101 may calculate a certainty of the candidate model by perturbing a next candidate model with a next query point.

[0139] Referring now to FIG. 3, depicted is a diagram of an example payment processing network 300, according to non-limiting embodiments or aspects. In some non-limiting embodiments or aspects, payment processing network 300 may be used in conjunction with the systems, methods, and/or computer program products described herein, and/or the systems, methods, and/or computer program products described herein may be implemented in payment processing network 300. As shown in FIG. 3, payment processing network 300 may include transaction processing system 301, payment gateway system 302, merchant system 304, issuer system 306, acquirer system 308, and/or consumer device 310. In some non-limiting embodiments or aspects, each of perturbation engine 101, training engine 102, model data 104, candidate model 106, training dataset 108, input instances 118, and/or query data points 116 of FIG. 1 may be implemented by (e.g., part of) transaction processing system 301. In some non-limiting embodiments or aspects, at least one of perturbation engine 101, training engine 102, model data 104, candidate model 106, training dataset 108, input instances 118, and/or query data points 116 of FIG. 1 may be implemented by (e.g., part of) another system, another device, another group of systems, or another group of devices, separate from or including transaction processing system 301, such as merchant system 304, issuer system 306, acquirer system 308, consumer device 310, and/or the like. For example, model data 104 and/or input instances 118 may be implemented by (e.g., part of) at least one of payment gateway system 302, merchant system 304, issuer system 306, acquirer system 308, and/or consumer device 310. For example, one of the aforementioned systems and/or devices may communicate the model data 104 and/or input instances 118 to transaction processing system 301.

[0140] Transaction processing system 301 may include one or more devices capable of receiving information from and/or communicating information to payment gateway system 302, merchant system 304, issuer system 306, acquirer system 308, consumer device 310, and/or the like (e.g., directly, indirectly, via a public and/or private communication network connection, and/or the like). For example, as shown in FIG. 3, transaction processing system 301 may be in communication with one or more issuer systems (e.g., issuer system 306), one or more acquirer systems (e.g., acquirer system 308), and/or one or more payment gateway systems (e.g., payment

gateway system 302). Although only a single issuer system 306, single acquirer system 308, and single payment gateway system 302 are shown, it will be appreciated that transaction processing system 301 may be in communication with a plurality of issuer systems, a plurality of acquirer systems, and/or a plurality of payment gateways. In some non-limiting embodiments or aspects, transaction processing system 301 may include a computing device, such as a server (e.g., a transaction processing server), a group of servers, and/or other like devices. In some non-limiting embodiments or aspects, transaction processing system 301 may be in communication with a data storage device, which may be local or remote to transaction processing system 301. In some non-limiting embodiments or aspects, transaction processing system 301 may be capable of receiving information from, storing information in, communicating information to, or searching information stored in the data storage device. In some non-limiting embodiments or aspects, transaction processing system 301 may be associated with a transaction service provider, as described herein. In some non-limiting embodiments or aspects, transaction processing system 301 may also operate as an issuer system such that both transaction processing system 301 and issuer system 306 are a single system and/or controlled by a single entity.

[0141] Payment gateway system 302 may include one or more devices capable of receiving information from and/or communicating information to transaction processing system 301, merchant system 304, issuer system 306, acquirer system 308, consumer device 310, and/or the like (e.g., directly, indirectly, via a public and/or private communication network connection, and/or the like). For example, as shown in FIG. 3, payment gateway system 302 may be in communication with one or more merchant systems (e.g., merchant system 304), one or more acquirer systems (e.g., acquirer system 308), and/or one or more transaction processing systems (e.g., transaction processing system 301). Although only a single merchant system 304, single acquirer system 308, and single transaction processing system 301 are shown, it will be appreciated that payment gateway system 302 may be in communication with a plurality of merchant systems, a plurality of acquirer systems, and/or a plurality of transaction processing systems. In some non-limiting embodiments or aspects, payment gateway system 302 may include a computing device, such as a server, a group of servers, and/or other like devices. In some non-limiting embodiments or aspects, payment gateway system 302 may be associated with a payment gateway, as described herein.

[0142] Merchant system 304 may include one or more devices capable of receiving information from and/or communicating information to transaction processing system 301, payment gateway system 302, issuer system 306, acquirer system 308, consumer device 310, and/or the like (e.g., directly, indirectly, via a public and/or private communication network connection, and/or the like). For example, as shown in FIG. 3, merchant system 304 may be in communication with one or more payment gateway systems (e.g., payment gateway system 302), one or more acquirer systems (e.g., acquirer system 308), and/or one or more consumer devices (e.g., consumer device 310). Although only a single payment gateway system 302, single acquirer system 308, and single consumer device 310 are shown, it will be appreciated that merchant system 304 may be in communication with a plurality of payment gateway systems, a plurality of acquirer systems, and/or a plurality of consumer devices. In some non-limiting embodiments or aspects, merchant system 304 may include a computing device, such as a server, a group of servers, a client device, a group of client devices, a POS device, a POS system, computers, computer systems, peripheral devices, and/or other like devices. In some non-limiting embodiments or aspects, merchant system 304 may be associated with a merchant, as described herein. In some non-limiting embodiments or aspects, merchant system 304 may include a device capable of receiving information from and/or communicating information to consumer device 310 via a short range communication connection (e.g., an NFC communication connection, an RFID communication connection, a Bluetooth® communication connection, a Zigbee® communication connection, and/or the like) with consumer device 310 and/or the like. In some non-limiting embodiments or aspects, merchant system 304 may include one or more client devices. For example, merchant system 304 may include a client device that allows a merchant to communicate information to transaction processing system 301 (e.g., via at least one of acquirer system 308 and/or payment gateway system 302). In some non-limiting embodiments or aspects, merchant system 304 (e.g., a client device thereof, a POS device thereof, and/or the like) may also operate as a payment gateway system such that both merchant system 304 and payment gateway system 302 are a single system and/or controlled by a single entity.

[0143] Issuer system 306 may include one or more devices capable of receiving information and/or communicating information to transaction processing system 301, payment gateway system 302, merchant system 304, acquirer system 308, consumer

device 310, and/or the like (e.g., directly, indirectly, via a public and/or private communication network connection, and/or the like). For example, as shown in FIG. 3, issuer system 306 may be in communication with one or more transaction processing systems (e.g., transaction processing system 301) and/or one or more consumer devices (e.g., consumer device 310). Although only a single transaction processing system 301 and a single consumer device 310 are shown, it will be appreciated that issuer system 306 may be in communication with a plurality of transaction processing systems and/or a plurality of consumer devices 310. In some non-limiting embodiments or aspects, issuer system 306 may include a computing device, such as a server, a group of servers, and/or other like devices. In some non-limiting embodiments or aspects, issuer system 306 may be associated with an issuer institution, as described herein. For example, issuer system 306 may be associated with an issuer institution that issued a credit account, debit account, credit card, debit card, a payment device, and/or the like to a user associated with consumer device 310.

[0144] Acquirer system 308 may include one or more devices capable of receiving information from and/or communicating information to transaction processing system 301, payment gateway system 302, merchant system 304, issuer system 306, consumer device 310, and/or the like (e.g., directly, indirectly, via a public and/or private communication network connection, and/or the like). For example, as shown in FIG. 3, acquirer system 308 may be in communication with one or more transaction processing systems (e.g., transaction processing system 301), one or more payment gateway systems (e.g., payment gateway system 302), and/or one or more merchant systems (e.g., merchant system 304). Although only a single transaction processing system 301, a single payment gateway system 302, and a single merchant system 304 are shown, it will be appreciated that acquirer system 308 may be in communication with a plurality of transaction processing systems, a plurality of payment gateway systems, and/or a plurality of merchant systems. In some non-limiting embodiments or aspects, acquirer system 308 may include a computing device, such as a server, a group of servers, and/or other like devices. In some non-limiting embodiments or aspects, acquirer system 308 may be associated with an acquirer institution, as described herein.

[0145] Consumer device 310 may include one or more devices capable of receiving information from and/or communicating information to transaction processing system

301, payment gateway system 302, merchant system 304, issuer system 306, acquirer system 308, and/or the like (e.g., directly, indirectly, via a public and/or private communication network connection, and/or the like). For example, as shown in FIG. 3, consumer device 310 may be in communication with one or more merchant systems (e.g., merchant system 304) and/or one or more issuer systems (e.g., issuer system 306). Although only a single merchant system 304 and a single issuer system 306 are shown, it will be appreciated that consumer device 310 may be in communication with a plurality of merchant systems and/or a plurality of issuer systems. In some non-limiting embodiments or aspects, consumer device 310 may be associated with a user to whom a credit account, debit account, credit card, debit card, a payment device, and/or the like has been issued. In some non-limiting embodiments or aspects, user device 310 may include a computing device, such as a computer, a portable computer, a laptop computer, a tablet computer, a mobile device, a cellular phone, a smartphone, a wearable device (e.g., watches, glasses, lenses, clothing, and/or the like), a PDA, a client device, and/or other like devices. In some non-limiting embodiments or aspects, user device 310 may include a payment device, as described herein. In some non-limiting embodiments or aspects, consumer device 310 may include a device capable of receiving information from and/or communicating information to other customer devices 310 (e.g., directly, indirectly, via a public and/or private communication network connection, a short range communication connection, and/or the like). In some non-limiting embodiments or aspects, consumer device 310 may include a device capable of receiving information from and/or communicating information to merchant system 304 via a short range communication connection (e.g., an NFC communication connection, an RFID communication connection, a Bluetooth® communication connection, a Zigbee® communication connection, and/or the like) with merchant system 304 and/or the like. In some non-limiting embodiments or aspects, consumer device 310 may include a client device.

[0146] In some non-limiting embodiments or aspects, transaction processing system 301 may communicate with merchant system 304 directly (e.g., via a public and/or private communication network connection and/or the like). Additionally or alternatively, transaction processing system 301 may communicate with merchant system 304 through payment gateway 302 and/or acquirer system 308. In some non-limiting embodiments or aspects, an acquirer system 308 associated with merchant system 304 may operate as payment gateway 302 to facilitate the communication of

transaction messages (e.g., authorization requests) from merchant system 304 to transaction processing system 301. In some non-limiting embodiments or aspects, merchant system 304 may communicate with payment gateway 302 directly (e.g., via a public and/or private communication network connection and/or the like). For example, a merchant system 304 that includes a physical POS device may communicate with payment gateway 302 through a public or private network to conduct card-present transactions. As another example, a merchant system 304 that includes a server (e.g., a web server) may communicate with payment gateway 302 through a public or private network, such as the Internet, to conduct card-not-present transactions.

[0147] For the purpose of illustration, processing a transaction (e.g., a payment transaction) may include generating a transaction message (e.g., authorization request and/or the like) based on an account identifier of a customer (e.g., accountholder associated with customer device 310 and/or the like) and/or transaction data associated with the transaction. For example, merchant system 304 (e.g., a client device of merchant system 304, a POS device of merchant system 304, and/or the like) may initiate the transaction, e.g., by generating an authorization request (e.g., in response to receiving the account identifier from a payment device and/or a portable financial device of the customer and/or the like). Merchant system 304 may communicate the authorization request to payment gateway 302 and/or acquirer system 308. In some non-limiting embodiments or aspects, payment gateway 302 may communicate the authorization request to acquirer system 308 and/or transaction processing system 301. Additionally or alternatively, acquirer system 308 (and/or payment gateway 302) may communicate the authorization request to transaction processing system 301. After receiving the authorization request from merchant system 304 that identifies the account identifier of the customer (e.g., the accountholder associated with consumer device 310 and/or the account identifier), transaction processing system 301 may communicate the authorization request to issuer system 306 (e.g., the issuer system that issued the payment device and/or account identifier). Issuer system 306 may determine an authorization decision (e.g., approve, deny, and/or the like) based on the authorization request, and/or issuer system 306 may generate an authorization response based on the authorization decision and/or the authorization request. Issuer system 306 may communicate the authorization response to transaction processing system 301. Transaction processing

system 301 may communicate the authorization response to acquirer system 308 and/or payment gateway 302. In some non-limiting embodiments or aspects, acquirer system 308 may communicate the authorization response to payment gateway 302 and/or merchant system 304. Additionally or alternatively, payment gateway 302 (and/or acquirer system 308) may communicate the authorization response to merchant system 304.

[0148] In some non-limiting embodiments or aspects, transaction processing system 301 and/or issuer system 306 may include at least one machine learning model (e.g., at least one of a fraud detection model, a risk detection model, a transaction authorization model, a credit approval model, a product recommendation model, a classifier model, an anomaly detection model, an authentication model, any combination thereof, and/or the like). For example, the machine learning model(s) may be trained and/or finetuned based on counterfactual instances, as described herein. Transaction processing system 301 and/or issuer system 306 may perform at least one task (e.g., generate a prediction and/or generate an embedding) based on the authorization request and the machine learning model(s). For example, performing the task(s) may include generating at least one prediction associated with fraud detection, risk detection, transaction authorization, credit approval, product recommendation, classification, anomaly detection, authentication, any combination thereof, and/or the like. In some non-limiting embodiments or aspects, transaction processing system 301 may communicate at least one message based on performing the task (e.g., generating the prediction and/or generate an embedding) to issuer system 306 (e.g., along with the authorization request). In some non-limiting embodiments or aspects, issuer system 306 may determine the authorization decision (e.g., approve, deny, and/or the like) based on the authorization request and the performance of the task (e.g., generation of the prediction and/or generation of the embedding).

[0149] For the purpose of illustration, clearing and/or settlement of a transaction may include generating a message (e.g., clearing message and/or the like) based on an account identifier of a customer (e.g., associated with customer device 310 and/or the like) and/or transaction data associated with the transaction. For example, merchant system 304 may generate at least one clearing message (e.g., a plurality of clearing messages, a batch of clearing messages, and/or the like). Merchant system 304 may communicate the clearing message(s) to acquirer system 308 (and/or payment gateway 302, which may communicate the clearing message(s) to acquirer

system 308). Acquirer system 308 may communicate the clearing message(s) to transaction processing system 301. Transaction processing system 301 may communicate the clearing message(s) to issuer system 306. Issuer system 306 may generate at least one settlement message based on the clearing message(s). In some non-limiting embodiments or aspects, issuer system 306 may communicate the settlement message(s) and/or funds to transaction processing system 301 (and/or a settlement bank system associated with transaction processing system 301), and transaction processing system 301 (and/or the settlement bank system) may communicate the settlement message(s) and/or funds to acquirer system 308. Additionally or alternatively, issuer system 306 may communicate the settlement message(s) and/or funds to acquirer system 308. In some non-limiting embodiments or aspects, acquirer system 308 may communicate settlement message(s) and/or funds to merchant system 304 (and/or an account associated with merchant system 304).

[0150] The systems and/or devices of FIG. 3 may communicate via one or more wired and/or wireless communication networks. For example, the communication network(s) may include a cellular network (e.g., a long-term evolution (LTE®) network, a third generation (3G) network, a fourth generation (4G) network, a fifth generation (5G) network, a code division multiple access (CDMA) network, and/or the like), a public land mobile network (PLMN), a local area network (LAN), a wide area network (WAN), a metropolitan area network (MAN), a telephone network (e.g., the public switched telephone network (PSTN)), a private network (e.g., a private network associated with a transaction service provider), an ad hoc network, an intranet, the Internet, a fiber optic-based network, a cloud computing network, and/or the like, and/or a combination of these or other types of networks.

[0151] The number and arrangement of systems, devices, and/or networks shown in FIG. 3 are provided as an example. There may be additional systems, devices, and/or networks; fewer systems, devices, and/or networks; different systems, devices, and/or networks; and/or differently arranged systems, devices, and/or networks than those shown in FIG. 3. Furthermore, two or more systems or devices shown in FIG. 3 may be implemented within a single system or device, or a single system or device shown in FIG. 3 may be implemented as multiple, distributed systems or devices. Additionally or alternatively, a set of systems (e.g., one or more systems) or a set of devices (e.g., one or more devices) of payment processing network 300 may perform

one or more functions described as being performed by another set of systems or another set of devices of payment processing network 300.

[0152] Referring now to FIG. 4, shown is a diagram of example components of a device 400 according to non-limiting embodiments. Device 400 may correspond to at least one of perturbation engine 101, training engine 102, model data 104, candidate model 106, training dataset 108, input instances 118, and/or query data points 116 in FIG. 1 and/or at least one of transaction service provider system 302, issuer system 304, customer device 306, merchant system 308, and/or acquirer system 310 in FIG. 3, as an example. In some non-limiting embodiments, such systems or devices may include at least one device 400 and/or at least one component of device 400. The number and arrangement of components shown are provided as an example. In some non-limiting embodiments, device 400 may include additional components, fewer components, different components, or differently arranged components than those shown. Additionally or alternatively, a set of components (e.g., one or more components) of device 400 may perform one or more functions described as being performed by another set of components of device 400.

[0153] As shown in FIG. 4, device 400 may include a bus 402, a processor 404, memory 406, a storage component 408, an input component 410, an output component 412, and a communication interface 414. Bus 402 may include a component that permits communication among the components of device 400. In some non-limiting embodiments, processor 404 may be implemented in hardware, firmware, or a combination of hardware and software. For example, processor 404 may include a processor (e.g., a central processing unit (CPU), a graphics processing unit (GPU), an accelerated processing unit (APU), etc.), a microprocessor, a digital signal processor (DSP), and/or any processing component (e.g., a field-programmable gate array (FPGA), an application-specific integrated circuit (ASIC), etc.) that can be programmed to perform a function. Memory 406 may include random access memory (RAM), read only memory (ROM), and/or another type of dynamic or static storage device (e.g., flash memory, magnetic memory, optical memory, etc.) that stores information and/or instructions for use by processor 404.

[0154] With continued reference to FIG. 4, storage component 408 may store information and/or software related to the operation and use of device 400. For example, storage component 408 may include a hard disk (e.g., a magnetic disk, an optical disk, a magneto-optic disk, a solid-state disk, etc.) and/or another type of

computer-readable medium. Input component 410 may include a component that permits device 400 to receive information, such as via user input (e.g., a touch screen display, a keyboard, a keypad, a mouse, a button, a switch, a microphone, etc.). Additionally, or alternatively, input component 410 may include a sensor for sensing information (e.g., a global positioning system (GPS) component, an accelerometer, a gyroscope, an actuator, etc.). Output component 412 may include a component that provides output information from device 400 (e.g., a display, a speaker, one or more light-emitting diodes (LEDs), etc.). Communication interface 414 may include a transceiver-like component (e.g., a transceiver, a separate receiver and transmitter, etc.) that enables device 400 to communicate with other devices, such as via a wired connection, a wireless connection, or a combination of wired and wireless connections. Communication interface 414 may permit device 400 to receive information from another device and/or provide information to another device. For example, communication interface 414 may include an Ethernet interface, an optical interface, a coaxial interface, an infrared interface, a radio frequency (RF) interface, a universal serial bus (USB) interface, a Wi-Fi® interface, a cellular network interface, and/or the like.

[0155] Device 400 may perform one or more processes described herein. Device 400 may perform these processes based on processor 404 executing software instructions stored by a computer-readable medium, such as memory 406 and/or storage component 408. A computer-readable medium may include any non-transitory memory device. A memory device includes memory space located inside of a single physical storage device or memory space spread across multiple physical storage devices. Software instructions may be read into memory 406 and/or storage component 408 from another computer-readable medium or from another device via communication interface 414. When executed, software instructions stored in memory 406 and/or storage component 408 may cause processor 404 to perform one or more processes described herein. Additionally or alternatively, hardwired circuitry may be used in place of or in combination with software instructions to perform one or more processes described herein. Thus, embodiments described herein are not limited to any specific combination of hardware circuitry and software. The term “configured to,” as used herein, may refer to an arrangement of software, device(s), and/or hardware for performing and/or enabling one or more functions (e.g., actions, processes, steps of a process, and/or the like). For example, “a processor configured to” may refer to a

processor that executes software instructions (e.g., program code) that cause the processor to perform one or more functions.

[0156] Although embodiments have been described in detail for the purpose of illustration, it is to be understood that such detail is solely for that purpose and that the disclosure is not limited to the disclosed embodiments or aspects, but, on the contrary, is intended to cover modifications and equivalent arrangements that are within the spirit and scope of the appended claims. For example, it is to be understood that the present disclosure contemplates that, to the extent possible, one or more features of any embodiment or aspect can be combined with one or more features of any other embodiment or aspect.

WHAT IS CLAIMED IS:

1. A system, comprising:

at least one processor configured to:

generate a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class;

generate a perturbation vector;

perturb the data record based on the perturbation vector to generate a first perturbed data record;

generate a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class;

select a subset of the first plurality of data records;

perturb the subset of the first plurality of data records to generate a plurality of synthesized data records;

train a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and

generate a second classification for the first perturbed data record based on the second machine-learning model.

2. A computer-implemented method, comprising:

generating, with at least one processor, a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class;

generating, with at least one processor, a perturbation vector;

perturbing, with at least one processor, the data record based on the perturbation vector to generate a first perturbed data record;

generating, with at least one processor, a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class;

selecting, with at least one processor, a subset of the first plurality of data records;

perturbing, with at least one processor, the subset of the first plurality of data records to generate a plurality of synthesized data records;

training, with at least one processor, a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and

generating, with at least one processor, a second classification for the first perturbed data record based on the second machine-learning model.

3. A system, comprising:

at least one processor configured to:

perturb a machine-learning model based on a plurality of input instances based on an outlier data point;

determine that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point;

in response to determining that the perturbation has failed, generate a plurality of query data points based on the outlier data point;

create a training dataset based on each query data point of the plurality of query data points;

train a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models;

perturb each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and

generate a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.

4. The system of claim 3, wherein the at least one processor is further programmed to configured to generate a perturbation vector for each input instance of a plurality of input instances, resulting in the plurality of perturbation vectors.

5. The system of claim 3, wherein the plurality of query data points are determined based on the outlier data point and a clustering or distance algorithm.

6. The system of claim 3, wherein each candidate model of the plurality of candidate models is generated separately, and wherein each subsequent candidate model is generated in response to the counterfactual output of the candidate model resulting in an unchanged predictive output.

7. The system of claim 3, wherein the at least one processor is further configured to:

determine that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and

in response to determining that the counterfactual output changes the predictive output, store the candidate model in a data structure for testing.

8. The system of claim 3, wherein the at least one processor is further configured to:

determine that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and

in response to determining that the counterfactual output changes the predictive output, calculate a certainty of the candidate model by perturbing a next candidate model with a next query point.

9. The system of claim 3, wherein the plurality of candidate models are stored in a stack data structure.

10. A computer-implemented method, comprising:

- perturbing, with at least one processor, a machine-learning model based on a plurality of input instances based on an outlier data point;
- determining, with at least one processor, that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point;
- in response to determining that the perturbation has failed, generating, with at least one processor, a plurality of query data points based on the outlier data point;
- creating, with at least one processor, a training dataset based on each query data point of the plurality of query data points;
- training, with at least one processor, a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models;
- perturbing, with at least one processor, each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and
- generating, with at least one processor, a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.

11. The method of claim 10, further comprising generating a perturbation vector for each input instance of a plurality of input instances, resulting in the plurality of perturbation vectors.

12. The method of claim 10, wherein the plurality of query data points are determined based on the outlier data point and a clustering or distance algorithm.

13. The method of claim 10, wherein each candidate model of the plurality of candidate models is generated separately, and wherein each subsequent candidate model is generated in response to the counterfactual output of the candidate model resulting in an unchanged predictive output.

14. The method of claim 10, further comprising:

determining that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and

in response to determining that the counterfactual output changes the predictive output, storing the candidate model in a data structure for testing.

15. The method of claim 10, further comprising:

determining that a counterfactual output of at least one candidate model of the plurality of candidate models changes a predictive output of the machine-learning model; and

in response to determining that the counterfactual output changes the predictive output, calculating a certainty of the candidate model by perturbing a next candidate model with a next query point.

16. The method of claim 10, wherein the plurality of candidate models

are stored in a stack data structure.

17. A computer program product comprising at least one non-

transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to perform any of the steps recited in claims 1-16.

18. A computer program product comprising at least one non-

transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to:

generate a plurality of initial classifications for a plurality of data records based on a first machine-learning model, the plurality of data records comprising a first plurality of data records for which respective initial classifications of the plurality of initial classifications are associated with a first class and a data record for which a respective initial classification of the plurality of initial classifications is associated with a second class;

generate a perturbation vector;

perturb the data record based on the perturbation vector to generate a first perturbed data record;

generate a first classification for the first perturbed data record based on the first machine-learning model, the first classification associated with the second class;

select a subset of the first plurality of data records;

perturb the subset of the first plurality of data records to generate a plurality of synthesized data records;

train a second machine-learning model based on the subset of the first plurality of data records and the plurality of synthesized data records; and

generate a second classification for the first perturbed data record based on the second machine-learning model.

19. A computer program product comprising at least one non-transitory computer-readable medium including program instructions that, when executed by at least one processor, causes the at least one processor to:

perturb a machine-learning model based on a plurality of input instances based on an outlier data point;

determine that the perturbation based on the plurality of input instances has failed to change a result of the machine-learning model for the outlier data point;

in response to determining that the perturbation has failed, generate a plurality of query data points based on the outlier data point;

create a training dataset based on each query data point of the plurality of query data points;

train a candidate model derived from the machine-learning model based on the training dataset, resulting in a plurality of candidate models;

perturb each candidate model of the plurality of candidate models based on a perturbation vector of a plurality of perturbation vectors; and

generate a counterfactual output based on an output of at least one candidate model of the plurality of candidate models.