technische universität
dortmund

Bachelorarbeit

Detektion von Zeitreihenanomalien in der Niederspannung

Joël Haubold

Juni 2020

Gutachter:
Prof. Dr. Rudolph
Dr.-Ing. Sebastian Ruthe

Technische Universität Dortmund
Fakultät für Informatik
Lehrstuhl für Computational Intelligence (LS-11)
https://ls11-www.cs.tu-dortmund.de/

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation und Hintergrund

1.1.1 Anomaliererkennung auf Zeitreihen

Anomalieerkennung auf Zeitreihen ist ein weitreichendes Forschungsgebiet, sowohl an der großen Zahl möglicher Vorgehensweise gemessen, als auch and der Vielfalt der Anwendungsgebiete. [11] Einige Beispiele für den Nutzen den die Erkennung von Anomalien darstellt sind:

- Finanzmärkte: Abrupte Einbrüche im Finanzmarkt müssen möglichst Frühzeitig erkannt werden um sich ausbreitenden Schaden zu verhindern oder einzudämmen.

- Benutzerhandlungen: Zeichnen sich Auffälligkeiten im Verhalten eines Benutzers ab so kann dies auf Situationen mit Handlungsbedarf hindeuten. So kann zum Beispiel etwaigen ungewollten Eingriffen in ein Computersystem entgegengewirkt werden.

- Biologische Daten: Zwar nicht direkt Zeitabhängig so können bestimmte biologische Forschungsprozesse, wie das platzieren einzelner Aminosäuren, analog zu temporalen Daten mit Methoden zur Zeitreihenanomalieerkennung unterstützt werden.

- Sensordaten: Viele physikalischen Anwendungen wird deren Verlauf anhand umfassender Sensordaten überwacht. Die hohe Quantität an Daten die kontinuierlich erfasst werden, macht es unmöglich diese alle per Hand auszuwerten und so kann automatisierte Anomalieerkenung dazu genutzt werden Ereignisse und Zusammenhänge in diesen Daten zu entdecken die ansonsten unbemerkt geblieben wären.

Diese Arbeit beschäftigt sich spezifisch mit dem in dem folgendem Abschnitt erläuterten Sensordatensatz.

1.1.2 Testdatensatz

Das deutsche Verteilnetz wurde ursprünglich mit dem Ziel gebaut, den in Großkraftwerken produzierten Strom und über das Transportnetz in die einzelnen Regionen Deutschlands transportiert wird, regional an die Endkunden (sowohl Industrie- und Gewerbekunden als auch Haushalte) zu verteilen. Das Verteilnetz ist dabei baumartig strukturiert und besteht aus der Hochspannungsebene die den Übergabepunkt des Transportnetz enthält und sich hin zur Mittelspannungsebene, Niederspannungsebene und schließlich den Endkunden verzweigt. Mit zunehmender Integration von Erneuerbaren Energien wie Wind- und PV-Anlagen in die Mittel- und Niederspannungsebene steigt auch die Dynamik in den unteren Spannungsebenen. Lastflüsse die vorher stets von oben (Hochspannung) nach unten (Mittel-, Niederspannung) gerichtet waren, kehren sich in Teilen um und können zu einer lokal höheren Auslastung des Netzes führen. Hinzu kommen neue Verbraucher wie z.B. Elektrofahrzeuge die insbesondere in den frühen Abendstunden und über die Nacht verteilt das Netz stärker belasten. Um diese Effekte erkennen und analysieren zu können, müssen die Niederspannungsebene zunächst messtechnisch erfasst werden. Ein Messgerätehersteller hat ein Messgerät entwickelt, welches sich in Ortsnetzstationen (Übergabepunkt von Mittel- zu Niederspannung) einbauen lässt und dort eine dreiphasige Spannungsmessung durchführen kann. Zusätzlich verfügt dieses Messgerät über eine Kommunikationsanbindung mit der sich die Daten abrufen und an einem zentralen Punkt aggregieren und auswerten lassen.

Zum Testen inwiefern sich diese Auswertung automatisieren lässt, sollen zuerst vordefinierte Anomalien automatisch auf dem Datensatz erkannt werden. In Kooperation mit der Firma logarithmo wurden dazu wurde eine Teilmenge dieser Daten, in Form von der gemessenen absoluten Spannung von 17 Messstationen als Grundlage dieser Arbeit freigegeben.

1.2 Aufbau der Arbeit

In dieser Arbeit wird nun für diese Anomalieerkennung eine Implementation des *Robust Random Cut Forest* Verfahrens [9] benutzt. Als Vergleichsverfahren wird das dem Random Forest zugrundeliegende *Isolation Forest* [13] Verfahren angewandt. Die Benutzung zweier unüberwachter Lernverfahren begründet sich durch den zu hohen Aufwand den ein manuelles Labelling des PPC-Datensatzes, aufgrund seiner Größe, sowie der relativen Knappheit an Vertretern der verschiedenen Anomalieklassen in diesem mit sich bringen würde.

In Kapitel 2 werden zuerst die Grundsätze und Herausforderungen von Anomalieerkennung erläutert. In Kapitel 3 und Kapitel 4 werden jeweils der Isolation Forest und der auf diesem aufbauende Random Forest beschrieben. In Kapitel 5 wird auf die Ergebnisse der beiden Verfahren eingegangen. In Kapitel 6 wird, auf Basis dieser Ergebnisse ein Fazit

gezogen, inwiefern auf Basis des PPC-Datensatzes der Random Forest eine Verbesserung gegenüber dem Isolation Forest verfahren darstellt, und generell für die Anwendung auf dem Datensatz geeignet ist.

Kapitel 2

Grundlagen

2.1 Anomalien

In einem gegebenen Datensatz an Punkten, wird einer dieser Punkte als Anomalie bezeichnet, falls er sich signifikant in einen oder mehreren seiner Merkmale von dem restlichen, nicht-anomalen *normalen* Punkten des Datensatzes abhebt.

Grundsätzlich lassen sich Anomalien darüber inwiefern sie sich aus ihrem Datensatz abheben in drei Klassen unterteilen [3] :

- *Punktanomalien*: Wenn ein Datenpunkt sich stark von den normalen Merkmalsausprägungen in seinem Datensatz unterscheidet. Beispielsweise wäre bei Beobachtung des Kraftstoffverbrauchs eines Autos pro Tag ein Verbrauch von 50 Litern, bei

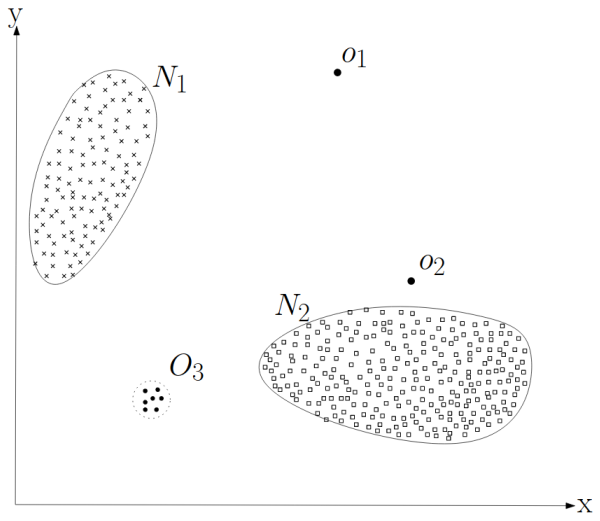


Abbildung 2.1: Ein zweidimensionaler Beispieldatensatz über zwei Merkmale x und y, dessen Struktur durch die Punktegruppen N_1 und N_2 gebildet wird. Im Kontext zu diesen sind die Punkte o_1 und o_2, sowie die Punktegruppe O_3 anomal. Quelle: [6]

einem normalen Verbrauch von 5 Litern pro Tag eine Punktanomalie. Die anomalen Punkte in Figur 2.1 entsprechen dieser Anomalieklasse.

- *Kontextanomalien*: Wenn ein Datenpunkt in einem bestimmten Kontext in seinem Datensatz hervorsticht, ohne diesen aber nicht als Anomalie zu erkennen wäre. Zum Beispiel können bei einer Anomalieerkennung auf den Finanzen einer Person, überdurchschnittlich hohe Ausgaben an einem Feiertag normal sein, im Kontext eines Arbeitstages allerdings eine Anomalie darstellen.

- *Kollektivanomalien*: Wenn mehrere, über ein oder mehrere ihrer Merkmale zusammenhängende Datenpunkte, welche alleine keine Besonderheit darstellen würden, zusammen eine Anomalie darstellen. Beispielsweise sind bei einem Elektrokardiogramm (EKG) einzelne niedrige Werte Teil einer der normalen Punkte, eine Reihe lange zeitlich aufeinanderfolgender Werte allerdings ist eine Anomalie.

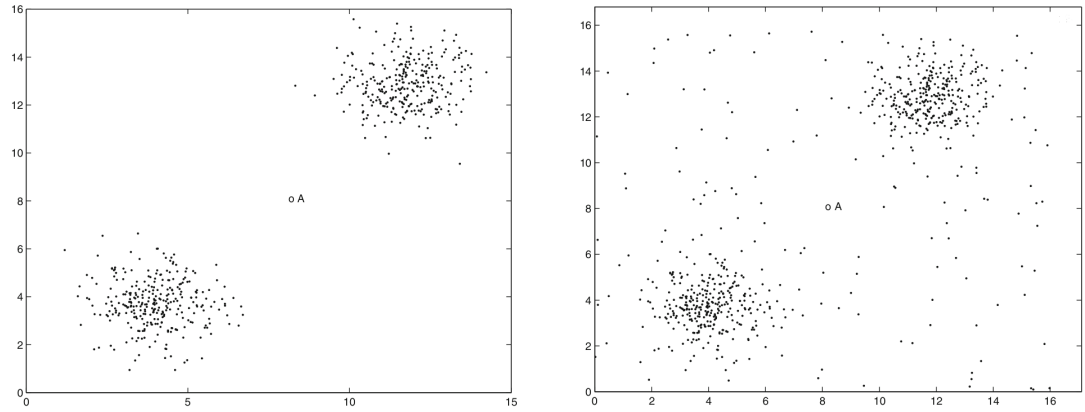
Kollektivanomalien setzen entsprechend ihrer Definition voraus, dass die Punkte des ihnen zugrunde liegendem Datensatz miteinander in Beziehung stehen, etwa wie in den oben aufgeführten Beispiel durch deren Zeitpunkt zu dem diese aufgenommen wurden. Ähnlich muss ein Datensatz über Attribute Verfügen, mit welcher für dessen Punkte Kontexte definiert werden können, damit in diesem Kontextanomalien existieren können.

2.1.1 Komplikationen

Die Diversität von möglichen Datensätzen und deren Merkmalen macht es generell nicht möglich, ein allgemeines Vorgehen für die Erkennung von Anomalien zu bestimmen. E Dazu kommen mögliche Eigenschaften von Datensätzen, welche Anomalieerkennung auf diesen weiter erschweren, oder bestimmten Vorgehensweisen sogar unmöglich machen, Anomalien zu klassifizieren. Ein Überblick über einige dieser möglichen erschwerenden Eigenschaften ist hier aufgeführt:

Kontextabhängigkeit

Es ist zu beachten das bei zwei anomalen Punkten nicht die gleichen Grenzwerte für die einzelnen Merkwerte gelten müssen, es kommt vielmehr auf die Kombination der Merkmale an. Ein einfaches Beispiel ist ein über die Zeit stetig zunehmender Messwert. Ein Punkt dessen Wert zu Beginn aus der Zeitreihe nach oben ausreißt, ist wahrscheinlich anomal. Die Punkte die später durch den Trend der Zeitreihe diesen Wert überschreiten, sind deswegen aber nicht zwingend selber anomal, noch invalidieren sie den Status des Ausreißers als Anomalie. [16]



(a) Ohne Rauschen **(b)** Mit Rauschen

Abbildung 2.2: Der Einfluss von Rauschen auf einen Datensatz bestehend aus zwei Ansammlung von Punkten und einem anomalen Punkt A. Quelle: [2]

Duplikate

Erschwerend für die Anomalieerkennung kann es sein falls sich mehrere Anomalien eines Datensatzes ähneln, wie in Abbildung 2.1. Während sich die Punkte in O_3 eindeutig von den beiden Inliner-Punktegruppen N_1 und N_2 abgrenzen, so haben sie alleinstehend betrachtet dennoch untereinander eine starke Ähnlichkeit., ein Modell des dargestellten Datensatzes vereinfacht sich durch die einzelne Entfernung eines Punktes aus O_3 nicht. [9] Sollen die Punkte in O_3 von einem Anomalieerkennungsverfahren als Anomalie eingestuft werden, so muss entweder dem Verfahren mitgeteilt werden das Inliner Ähnlichkeiten zu den Punkten in N_1 und N_2 haben müssen, oder es muss so kalibriert werden, dass eine Ansammlung von 7 ähnlichen Punkten noch nicht als Inlinergruppe gesehen wird. Mehr dazu in Sektion 2.2.1

Rauschen

Je nach generierenden Prozess des Datensatzes kann es sein das in diesem neben der zu beobachtenden Größe, weitere Punkte aufgenommen werden, welche sich in ihren Merkmalen stark von den Inlinern unterscheiden, aber nicht von Relevanz für den Beobachter des Prozesses sind. [2]

In den beiden Abbildungen 2.2 ist die Schwierigkeit die Rauschen bei der Anomalieerkennung mit sich bringt zu sehen. In Abbildung 2.2 (a) ist der Punkt A offensichtlich anomal. In 2.2 (b) könnte dieser allerdings Teil des Rauschens sein. Um den Punkt A als anomal markieren zu können, aber nicht den Rest des uninteressanten Rauschens, muss dem Anomalieerkennungsverfahren mitgeteilt werden das Punkte mit seinen Merkmalen als anomal gelten.

Mehrdimensionalität

Hat der zu untersuchende Datensatz eine hohe Dimensionalität in seinen Merkmalen, führt dies zu weiteren Problemen bei der Anomalieerkennung. Mit zunehmender Anzahl an Merkmalsdimensionen erhöhen sich die möglichen Kombinationen an Dimensionen auf denen nach anomalen Merkmalen gesucht werden kann exponentiell, womit der Aufwand der Anomalieerkennung ansteigen kann. Weiterhin führt diese Zunahme der möglichen Dimensionskombinationen auf denen gesucht werden kann, dass es immer wahrscheinlicher wird, für jeden Punkt mindestens eine solche Kombination zu finden, dass er auf dieser anomal ist. Umgekehrt wird es mit zunehmenden Dimensionen, auf denen man nach anomalen Ausprägungen suchen kann, schwieriger die relevanten Dimensionen zu finden. Es entsteht effektiv ein Rauschen, da die relevanten Dimensionen gegenüber den nicht relevanten untergehen. [7]

2.2 Anomalieerkennungsverfahren

Ein Anomalieerkennungsverfahren bietet generalisiert die Funktion auf einem Datensatz Anomalien zu erkennen und diese eventuell in mehrere Klassen zu kategorisieren. Nicht alle Verfahren für alle Datensätze, sei es weil sie für eine bestimmte Eigenschaft des Datensatzes nicht geeignet sind, oder umgekehrt weil sie zur eigenen Leistungsverbesserung bestimmte Eigenschaften im Datensatz voraussetzen [11]. Auch die Zielsetzung, welche Art von Anomalie man in dem Datensatz erkennen will hat Einfluss auf die Auswahl des entsprechenden Anomalieerkennungsverfahren.

2.2.1 Überwachtes und unüberwachtes Lernen

Es mag sein, dass für Teile eines Datensatzes auf dem ein Anomalieerkennungsverfahren laufen soll, bereits Label existieren die Punkte oder Ausschnitte des Datensatzes als anomal oder normal klassifizieren. Mithilfe dieser gelabelten Daten können einem Anomalieerkennungsverfahren die Eigenschaften der anomalen beziehungsweise der normalen Daten antrainiert werden, damit es diese besser auf ungelabelten Daten erkennen kann. Anomalieerkennungsverfahren lassen sich so über den Grad an Informationen den sie auf ihren Trainingsdaten benötigen in drei Klassen unterteilen [6]:

Überwachtes Lernen

Das überwachte Lernen setzt voraus das ein Trainingsdatensatz zur Verfügung steht, welcher gelabelte Instanzen von normalen sowie anormalen Daten enthält. Oftmals wird über diese Daten ein Prognosemodell erstellt, welches zwischen den normalen Daten, sowie den Anomalieklassen unterscheiden soll. Dies ermöglicht komplexere Anomaliedefinitionen, welche eventuell nicht alleinig über die Seltenheit der Eigenschaften eines Punktes oder einer

Gruppe von Punkten erkennbar wären. Bei dieser Art von Anomalieerkennung gibt es zwei grundlegende Probleme: Zuerst sind Anomalien, naheliegend aus ihrer Definition im Datensatz oft nur geringfügig vertreten, was dazu führen kann das bei der Erstellung des Prognosemodells die zugehörigen Anomalieklassen zu spezifisch auf diese im Trainingsdatensatz vorkommenden Ausprägungen dieser Klassen modelliert werden, was dazu führt das diese im ungelabelten Teil des Datensatzes nicht vollständig als Teil ihrer Klasse erkannt werden. Weiterhin entspricht das Vorhandensein eines Trainingsdatensatzes, welcher alle möglichen Ausprägungen aller Anomalieklassen darstellt oft nicht der Praxis. Einerseit, weil Anomalien als Abweichung von dem normalen Verhalten des Datensatzes oftmals in vielfältiger Form kommen können, und somit eventuell künstliche Trainingspunkte erzeugt werden müssen um die Ausmaße der Klassen, sowie den eventuell benötigten Kontext für diese Klassen ausreichend darzustellen. Andererseits müssen real vorkommende Anomalien im Trainingsdatensatz oftmals per Hand von einer Fachkraft als solche gelabeled werden, was unerwünschte Kosten mit sich bringt.

Semiüberwachtes Lernen

Beim semiüberwachtem Lernen wird nur von einem Trainingsdatensatz ausgegangen, welcher das Normalverhalten der Daten vollständig darstellt. Dies ist oft einfacher zu erfüllen, so kann der Trainingsdatensatz zum Beispiel bereits aus einer Aufzeichung eines normalen Ablaufs des datengenerierenden Prozesses gewonnen werden. Die einzige Voraussetzung bleibt, dass der Trainingsdatensatz das Normalverhalten ausreichend genug darstellt, sodass das Anomalieerkennungsverfahren nicht anormale Entwicklungen im datengenerierenden Prozess, als solche erkennen kann. Da das Normalverhalten eines Prozesses aber oft klarer definiert ist, als eine anomale Abweichung in beliebig komplexer Form, ist diese Voraussetzung oft verfügbarer als die für überwachtes Lernen.

Aufgrund dieses klaren Unterschied der Praktikabilität, kommen semiüberwachte Ansätze, welche einen Trainingsdatensatz aus gelabelten Anomalien nur limitiert vor.

Unüberwachtes Lernen

Der Ansatz des unüberwachten Lernens benötigt keinerlei vorgelabelten Testdaten, stattdessen wird die implizite Annahme getroffen, dass Anomalien im Datensatz wesentlich seltener auftreten als normale Daten. Unter dieser Annahme versucht ein unüberwachtes Verfahren diese selten auftretenden anomalen Daten von den zueinander konformen Daten abzugrenzen.

Viele semiüberwachte Verfahren können unüberwacht angewandt werden, indem sie in Abwesenheit des eigentlich benötigten, das Normalverhalten modellierenden Testdatensatzes, über einem Datensatz trainieren, bei dem davon ausgegangen wird das dieser eine sehr geringe Anzahl an Anomalien enthält. So verwendete Verfahren müssen robust genug

gegenüber den so eintrainierten Anomalien sein, um eine Verfälschung bei der späteren Beurteilung von Daten zu verhindern.

2.2.2 Input und Output von Anomalieerkennungsverfahren

Weiter Unterscheidungen lassen sich über Anomalieerkennungsverfahren darin machen, in welcher Form der Input auf Anomalien untersucht wird, und in Welcher Form das Anomalieerkennungsverfahren seine Ergebnisse ausgibt.

Arten von zu analysierenden Dateninstanzen

Auch darin in welcher Form die Anomalien erkannt werden sollen unterscheiden sich die möglichen Verfahren. Je nach Zielsetzung und den Eigenschaften der zugrundeliegenden Daten kann in diesen nach einzelnen oder unter einem bestimmten Kontext zusammenhängenden anomalen Datenpunkten gesucht werden. Auch mag es von Nutzen seien, ganze Datensätze aus einer Gruppe dieser als anomal oder normal zu bestimmen. [11]

Arten des Dateninputs

Grundsätzlich lassen sich verschiedene Verfahren darin unterscheiden, ob sie davon ausgehen alle Daten auf denen sie operieren sollen von Anfang an zur Verfügung haben, also diese Daten *offline* analysieren können. Ist dies nicht der Fall, sind mit der Anomaliedetektion auf diesen *gestreamten* Daten eine Reihe von Komplikationen verbunden [16] [2]: Solche Streams oftmals kein vordefiniertes Ende haben, ist es meist nicht praktisch diese abzuspeichern, um auf den so gesammelten Daten ein offline Verfahren auszuführen, da dieses zwangsweise nicht genug Speicher haben kann, und da es von Bedeutung sein mag die Anomalien möglichst nahe zu dem Zeitpunkt, an dem diese gestreamt wurden zu erkennen. Auch haben Streams, welche oftmals eine live Übertragung von Sensordaten darstellen oftmals das oben definierte Problem, nur wenige Anomalien zu enthalten, welche die Gesamtheit aller möglichen Anomalien nicht ausreichend darstellen, was das Antrainieren eines Verfahrens zum Erkennen der Anomalieklassen erschwert. Weiterhin muss darauf geachtet werden, dass mögliche Änderungen des Kontextes in welchem Anomalien auftreten, zum Beispiel in Form einer Änderung im Normalverhalten der Daten in die Anomalieerkennung miteinbezogen werden.

Ergebnisse des Anomalieerkennungsverfahren

Das Ergebnis eines Anomalieerkennungsverfahrens, stellt die Beurteilung des Verfahrens gegenüber den eingegebenen Datensatz dar, ob die Eingabe oder die Elemente die diese ausmacht anomal oder nicht sind, beziehungsweise um welche Art von Anomalie es sich handelt. Allgemein kann man zwischen zwei Ausgabearten der Ergebnisse unterscheiden: [3]

- *Bewertung*: Bei bewertenden Anomalieerkennungsverfahren wird jeder zu bewertenden Dateninstanz, ein Wert zugeordnet, dessen Größe darstellt wie sicher sich das Verfahren ist, ob die Instanz eine Anomalie ist. Entweder werden diese Werte dann einer genaueren Betrachtung unterzogen, oder es wird eine Grenze festgelegt, ab welchen Wert eine Dateninstanz als Anomalie interpretiert wird.

- *Kennzeichnung*: Bei einem kennzeichnenden Anomalieerkennungsverfahren bestimmt das Verfahren im Alleingang, ob eine Dateninstanz eine Anomalie ist oder nicht, beziehungsweise zu welcher Anomalieklasse es gehört.

2.2.3 Robustheit

Die Robustheit eines Algorithmus beschreibt seine Stabilität gegenüber Anomalien im Trainingsdatensatzes und gegenüber ungewollten Unterschieden zwischen dem Trainingsdatensatz und dem Testdatensatz. Weiterhin kann ein Anomalieerkennungsverfahren besonders Robust gegenüber einer Eigenschaft von Datensätzen, wie zum Beispiel Rauschen oder Mehrdimensionalität, sein, die sich allgemein negativ auf die Performance von auf ihrem Datensatz ausgeführten Algorithmen auswirkt. [18]

2.2.4 Kriterien zur Performancebeurteilung

Zur Beurteilung des Erfolges eines spezifischen Anomalieerkennungsverfahrens auf einem gelabelten Datensatz kann das Ergebnis von diesem mit den vorhandenen Labels abgeglichen werden. Aus diesem Vergleich können eine Reihe von Metriken gezogen werden, um eine quantitative Rangliste der Performance von verschiedenen Verfahren, beziehungsweise verschieden parametrisierte Verfahren erstellen zu können [5]. In den folgenden vorgestellten Verfahren werden die folgenden Notationen, basierend auf den englischen Begriffen $True(/\,False)$ $Positives(/\,Negatives)$ benutzt. Zur beispielhaften Darstellung wird hier von einer einzigen Anomalieklasse, und einer punktweisen Anomalie Erkennung ausgegangen:

$TP =$ Anzahl korrekt als Anomalien klassifizierter Punkte

$TN =$ Anzahl korrekt als nicht-Anomalie klassifizierter Punkte

$FP =$ Anzahl fälschlicherweise als Anomalien klassifizierter Punkte

$FN =$ Anzahl fälschlicherweise als nicht-Anomalie klassifizierter Punkte

Ein Punkt gilt als von seinem Verfahren korrekt klassifiziert, wenn das Ergebnis des Verfahrens in bezüglich mit seinem Label übereinstimmt.

Tabelle 2.1: Zwei mögliche Ergebnisse eines Anomalieerkennungsverfahrens auf einem Datensatz bestehend aus 950 normalen und 50 anomalen Punkten, mit der berechneten Genauigkeit für jedes Verfahren

Verfahren	TP	TN	FP	FN	Genauigkeit
V1	0	950	0	50	0.95
V2	50	900	50	0	0.95

Accuracy

Eine der einfachsten Formen des Vergleichs bietet sich dadurch, zu identifizieren zu welchen Raten Anomalien, beziehungsweise nicht-Anomalien korrekt identifiziert wurden. Diese ergeben sich jeweils folgendermaßen aus den richtig und falsch klassifizierten Anomalien, beziehungsweise nicht-Anomalien:

$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{TN}{TN + FP}$$

Aus dem Zusammenfügen dieser beiden Raten, für einen einfachen Vergleich ergibt sich die *Genauigkeit* (englisch: *Accuracy*) mit der ein Verfahren seine Daten klassifiziert:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Die möglichen Werte der Genauigkeit liegen dabei zwischen 0 und 1, wobei eine 0 dafür steht das kein einziger und eine 1 dafür, dass alle Punkte des zugrundeliegenden Datensatzes richtig klassifiziert wurden. Der Vorteil dieser Metrik zur Performancebeurteilung eines Verfahrens liegt neben der Intuitivität darin, dass sie alle möglichen Klassifizierung eines Datenpunktes miteinbezieht, jegliche positive oder negative Abänderung des Ergebnisses demnach durch sie abgebildet wird. Ein Problem mit der Genauigkeit eines Verfahrens ergibt sich allerdings sobald der zugrundeliegende Datensatz gegenüber einer Klasse sehr unbalanciert ist, er zum Beispiel wesentlich mehr anomale als nicht anomale Punkte enthält. Diese wie in Sektion 2.2.1 beschriebene häufig vorkommende Situation kann zu, wie in Tabelle 2.1 dargestellt missrepresentativen Genauigkeitswerten führen. Die beiden dargestellten Verfahren V1 und V2 unterscheiden sich stark darin, dass V1 keine einzige der 50 Anomalien als solche klassifiziert hat, V2 hingegen lediglich 50 Punkte fälschlicherweise als normal klassifiziert hat. In der Praxis würde V2 fast immer als produktiver angesehen werden, dies spiegelt sich jedoch keinesfalls in der Genauigkeit der beiden Verfahren wieder.

Tabelle 2.2: Zwei mögliche Ergebnisse eines Anomalieerkennungsverfahrens auf zwei Datensätzen, mit dem berechneten F_1-Maß für jedes Verfahren. Der Datensatz von V1 besteht aus 200 anomalen und 125 normalen Punkten, der von V2 aus 200 normalen und 1025 anomalen Punkten.

Verfahren	TP	TN	FP	FN	Präzision	Trefferquote	F_1
V1	100	0	25	100	0.8	0.5	~ 0.62
V2	100	1000	25	100	0.8	0.5	~ 0.62

F-Measure

Um einen Messwert zu definieren, dessen Fokus mehr auf dem Erkennen von Anomalien liegt eignen sich die Präzision (englisch: *Precision*) und die Trefferquote (englisch: *Recall*) eines Verfahrens, welche wie folgt definiert sind:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

Dabei stellt die Präzision im Bereich von 0 bis 1 den Anteil an wirklichen Anomalien den diese an den als Anomalie bestimmten Punkte eines Verfahrens haben dar. Dazu passend steht die Trefferquote im Bereich von 0 bis 1 für den Anteil an Anomalien die ein Verfahren erfolgreich klassifiziert hat. Über das harmonische Mittel zusammengefasst ergibt sich das F_1-Maß:

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Ein hoher F_1 Wert steht so für ein Verfahren, welches einen Großteil der Anomalien des zugrundeliegenden Datensatzes richtig klassifiziert und nur wenige normale Punkte fälschlicherweise als Anomalie klassifiziert. Soll auf eine dieser Eigenschaften mehr Wert gelegt werden bietet sich eine Variation des F_1-Maßes an:

$$F_\alpha = (1 + \alpha^2) * \frac{Precision * Recall}{\alpha^2 * Precision + Recall}$$

Für α größer null wird die Trefferquote mit wachsendem α im Vergleich zur Präzision stärker gewichtet. Alle F-Maße haben allerdings zwei Schwächen. Einmal setzen sie das vorhandensein von Anomalien im Datensatz voraus um angewandt weren zu können, da ansonsten mit einer garantieren Anzahl von richtig klassifizierten Anomalien TP von 0, sowohl die Genauigkeit als auch die Trefferquote, und somit auch jedes F-Maß immer gleich 0 sind. Der zweite Nachteil des F-Maßes ist die Nichteinbeziehung der Anzahl der von dem Verfahren richtig als normal klassifizierten Punkte FP, wie in der Tabelle 2.2 dargestellt [8]. Verfahren V1 hat gegenüber dem Verfahren V2 die Schwäche keine normalen Punkte als solche klassifizieren zu können. Dennoch sind beide Verfahren nach beliebigen F-Maßen gleich.

MCC

Eine Lösung sowohl des Problems der Nichtmiteinbeziehung der richtig klassifizierten normalen Punkte der F-Maße, als auch der Schwäche gegenüber stark unbalancierten Klassen der Genauigkeit, bildet der Mathews Correlation Coefficient (MCC):

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}}$$

Dieser reicht von -1 für ein Verfahren, welches alle Punkte jeweils umgekehrt zu ihren Labels klassifiziert über 0 für ein Verfahren, welches Punkte unabhängig von ihrem tatsächlichem Label klassifiziert zu 1 für ein Verfahren, welches alle Punkte entsprechend ihrer Labels klassifiziert. Auch für den MCC bilden sich jedoch zwei Probleme. Erstens zieht dieser zwar alle möglichen Klassifizierungen mit ein und vernachlässigt keine basierend auf ihrer Größe, dafür fehlt ihm aber die Flexibilität der F-Maße, einen Fokus auf die Reinheit, beziehungsweise auf die Vollständigkeit der Klassifizierungen eines Verfahrens zu legen. Weiterhin ist der MCC aufgrund der Miteinbeziehung der als normal klassifizierten Punkte FP sowohl für Datensätze, welche keine Anomalien, als auch für Datensätze welche nur Anomalien beinhalten nicht definiert.

Kapitel 3

Isolation Forest

Da das in dieser Arbeit auf dem Testdatensatz evaluierte Verfahren *Robust Random Cut Forest* auf dem benutztem Vergleichsverfahren *Isolation Forest* (**iForest**) basiert, wird zuerst in diesem Kapitel der Isolation Forest in seinen Grundzügen beschrieben. Das Kapitel orientiert sich dabei an dem Artikel [13], welcher das Verfahren vorstellte. Es werden die folgenden Notationen benutzt:

- x Ein Datenpunkt
- Xr Ein Datensatz aus n Punkten
- n Die Anzahl von Punkten in einem Datensatz, $n = |X|$
- Q Die Attribute eines Datensatzes X
- x_q Der Wert von x in $q \in Q$
- T Ein Baum oder ein Knoten eines Baumes
- t Die Anzahl von Bäumen in einem Forest
- $h(x)$ Die Weglänge eines Punktes x in einem Baum
- $hlim$ Tiefenmaximum der Wegevaluation
- ψ Größe einer Stichprobe
- s Eine Funktion welche die Annomalität eines Punktes bewertet

3.1 Isolation Forest Theory

Ein Großteil der existierenden Anomalieerkennungsverfahren, wie *Replicator Neural Network* [19], *One class Svm* [17], oder auf Klassifizierung [1] [15] beziehungsweise Clustering [12] basierenden Methoden erkennen Anomalien in einem Datensatz, indem sie ein Profil

der normalen Klasse an Punkten konstruieren und alle Punkte welche von diesem Profil abweichen als Anomalien identifizieren. Da solche Verfahren oftmals ursprünglich nicht zur Anomalieerkennung eingesetzt wurden ergeben sich zwei große Nachteile [13]: Zuerst ziehen nicht auf Anomalieerkennung spezifizierte Verfahren nicht den oftmals sehr geringen Anteil den anomale Punkte am Datensatz haben in Betracht, was zu einer erhöhten Zahl an fälschlicherweise als Anomalie klassifizierten Punkten führt. Weiterhin sind solche Verfahren oftmals nicht für hoch dimensionale oder sehr große Datensätze, welche zum Beispiel bei der Auswertung von Sensordaten vorhanden sein können optimiert, und brauchen daher auf diesen Datensätzen eine große Menge an Rechenleistung.

Als ein alternatives Anomalieerkennungsverfahren wurde in dem Artikel [13] die Methodik entwickelt anomale Instanzen eines Datensatzes zu isolieren, ohne dabei eine Distanz beziehungsweise Dichte zwischen den einzelnen Instanzen zu kalkulieren. Dabei wird von den beiden, in Sektion 2.2.1 beschriebenen grundlegenden Annahmen von unüberwachten Anomalieerkennungsverfahren über die Eigenschaften anomaler Instanzen ausgegangen:

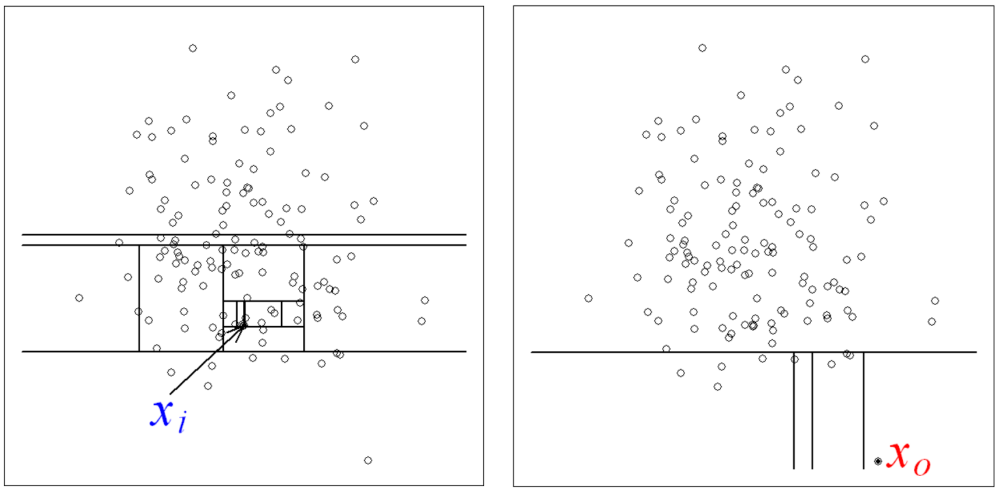
3.1.1 Theorem. *Anomale Punkte in einem Datensatz zeichnen sich durch zwei Eigenschaften aus:*

1. *Anomalien bilden eine Minderheit in ihrem Datensatz*
2. *Anomalien zeichnen sich durch von der Norm des Datensatzes stark abweichende Merkmalsausprägungen aus*

Das grundlegende Konzept von *Isolation* ist dabei, dass diese beiden Eigenschaften es einfacher machen einen anomalen Punkt von dem restlichem Datensatz zu trennen als einen normalen Punkt. Das zugrundeliegende Trennverfahren eines iForests sind eine Ansammlung von binären Bäumen, genannt *Isolation Tree* (**iTree**), aufgrund der Effizienz ihrer Konstruktion.

3.1.1 iForest Training

Um die Empfänglichkeit jedes Punktes eines Datensatzes X zu bestimmen, von X isoliert zu werden, partitioniert ein iTree X solange rekursiv bis alle Punkte von X voneinander getrennt sind. Die inneren Knoten stellen dabei, entsprechend Definition 3.1.2 jeweils eine Partition und die Blätter jeweils eine Stelle an der ein Punkt von X isoliert wurde dar. Zu jeder Partition wird dazu ein Merkmal des verbleibenden Datensatzes X', sowie ein zufälliger Trennwert zwischen den maximalen und minimalen Wert in X' in diesem Merkmal ausgewählt, worauf X' über diesen Grenzwert getrennt wird. Aufgrund der vorher definierten Annahmen zu den Eigenschaften anomaler Punkte, benötigen diese wesentlich weniger Partitionen und haben somit einen wesentlich kürzeren Pfad zur Wurzel als normale Punkte:



(a) Isolation eines normalen Punktes x_i über 12 Partitionen

(b) Isolation eines anomalen Punktes x_o über 4 Partitionen

Abbildung 3.1: Zwei gleiche Datensätze über zwei Dimensionen, in denen jeweils ein Punkt durch zufällige partitionierungen des Datensatzes isoliert wurde. Jede Schnittlinie stellt eine Isolation dar. Quelle: [13]

- Die geringe Anzahl an Anomalien führt zu mehr Platz um die Anomalien herum, wodurch weniger Partitionen gebraucht werden um diese zu isolieren.
- Die stark abweichenden Merkmalsausprägungen von Anomalien führt zu einer höheren Wahrscheinlichkeit, dass diese durch den zufällig gewählten Trennwert einer Partition von dem Rest des Datensatzes getrennt wird.

Diese beiden Effekte sind in dem Beispiel in Figur 3.1 dargestellt. Aufgrund der Abwesenheit von ähnlichen Punkten, sowie der hohen Entfernung von x_o zu dem Cluster an normalen Punkten, braucht es im Vergleich zu dem Punkt x_i merkbar weniger zufällige Partitionen um diesen von dem Rest des Datensatzes zu trennen.

Es ergibt sich die folgende Definition eines iTrees:

3.1.2 Definition (Isolation Tree). Sei T ein Knoten eines iTrees über eine Stichprobe der Größe ψ des Datensatz X mit Attributen Q. T ist entweder ein Blatt (*exNode*), oder ein interner Knoten (*inNode*) mit zwei Kindknoten (T_l,T_r) und einer Partition. Die Partition von T besteht aus einem Attribut $q \in Q$ und einem Trennwert p, sodass für $x \in X$ über $x_q < p$ bestimmt wird ob x zu T_l oder T_r partitioniert wird.

Es ergeben sich, für einen über ψ Punkte konstruierten iTree ψ externe und ,aus der zugrunde liegenden Struktur eines Binärbaums hervorgehend $\psi - 1$ interne Knoten.

Um die möglichen Schwankungen durch die zufällige Auswahl des Attributes sowie des Trennwerts zu jeder Partition auszugleichen, und um zu einem konsistenten Ausgabe zu kommen, wird beim iForest Verfahren nicht ein iTree verwendet, sondern eine Ansammlung

von t iTrees, deren Einschätzung gegenüber der Anomalität eines Punktes, kombiniert wird um zu der letztendlichen Einschätzung des iForests zu kommen.

3.1.2 iForest Auswertung

Unter Definition 3.1.2 ergibt sich für die Länge des Weges eines Punktes, welcher den Partitionen von der Wurzel eines iTrees aus folgt, bis er auf ein Blatt trifft:

3.1.3 Definition (Weglänge $h(x)$). Sei T ein iTree. Die Weglänge $h(x)$ eines Punktes x entspricht der Anzahl an Kanten von T, welche x den Partitionen der inneren Knoten von T folgend passiert, bis der Weg von x an einem externen Knoten endet.

Die Weglänge eines Punktes x kalkuliert durch einen iForest $E(h(x))$, ergibt sich dann als der Durchschnitt aller kalkulierten Weglängen der iTrees des iForests.

Da die erwartete Weglänge eines Punktes in einem iTree T, mit der Zunahme der Größe der Stichprobe über der T konstruiert wurde um die Größenordnung $log(\psi)$ zunimmt, wird die Weglänge $E(h(x))$ des Punktes x in einem iForest über die durchschnittliche Weglänge $c(\psi)$ eines beliebigen Punktes in einem iTree welcher über eine Stichprobe der Größe ψ konstruiert wurde. So kann die Performance von iForests mit unterschiedlichen Stichprobengrößen verglichen werden. Da die Struktur eines iTrees mit der von *Binary Search Trees* übereinstimmt, ist die durchschnittliche Weglänge einer nicht erfolgreichen Suche eines Binary Search Trees gleich $c(\psi)$. Somit gibt sich für $c(\psi)$ mit der harmonischen Zahl $H(i) \approx \ln(i) + 05772156649$ [14]:

$$c(\psi) = \begin{cases} 2H(\psi-1) - 2(\psi-1)/n & ,\psi > 2 \\ 1 & ,\psi = 2 \\ 0 & ,\text{otherwise} \end{cases} \tag{3.1}$$

Die genormte Einschätzung der Anomalität eines Punktes x durch einen iForest der Größe ψ ist nun definiert als:

$$s(x,\psi) = 2^{-\frac{E(h(x))}{c(\psi)}} \tag{3.2}$$

Die gesamte Spannweite von s liegt im Intervall $[0,1]$ und ist in Abbildung 3.2 abgebildet. Es stechen 3 besondere Werte von s hervor:

1. $E(h(x)) \to 0, s \to 1$: Ist s nahe 1, geht die durchschnittliche Weglänge von x gegen 0. x ist somit wesentlich anfälliger für Isolation als ein durchschnittlicher Punkt der zur Konstruktion des iForests benutzten Stichprobe, und somit nach Theorem 3.1.1 höchstwahrscheinlich eine Anomalie.

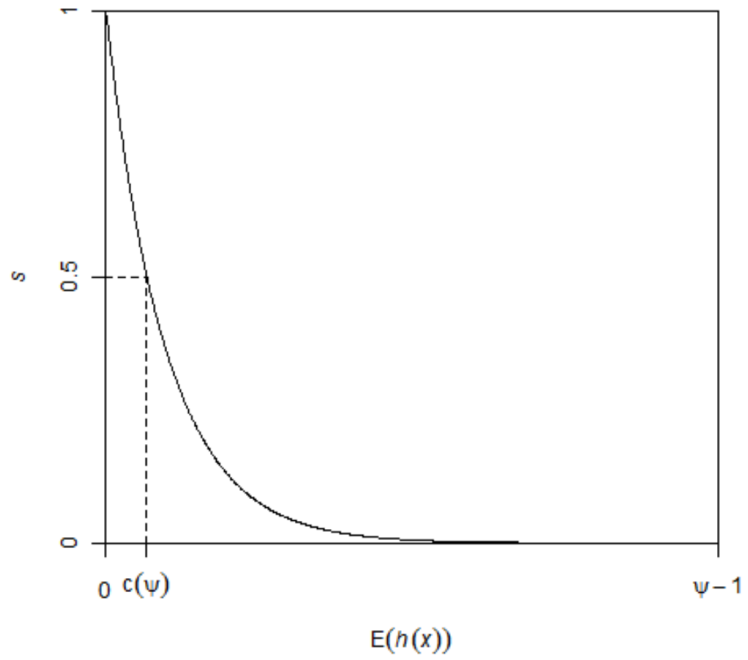


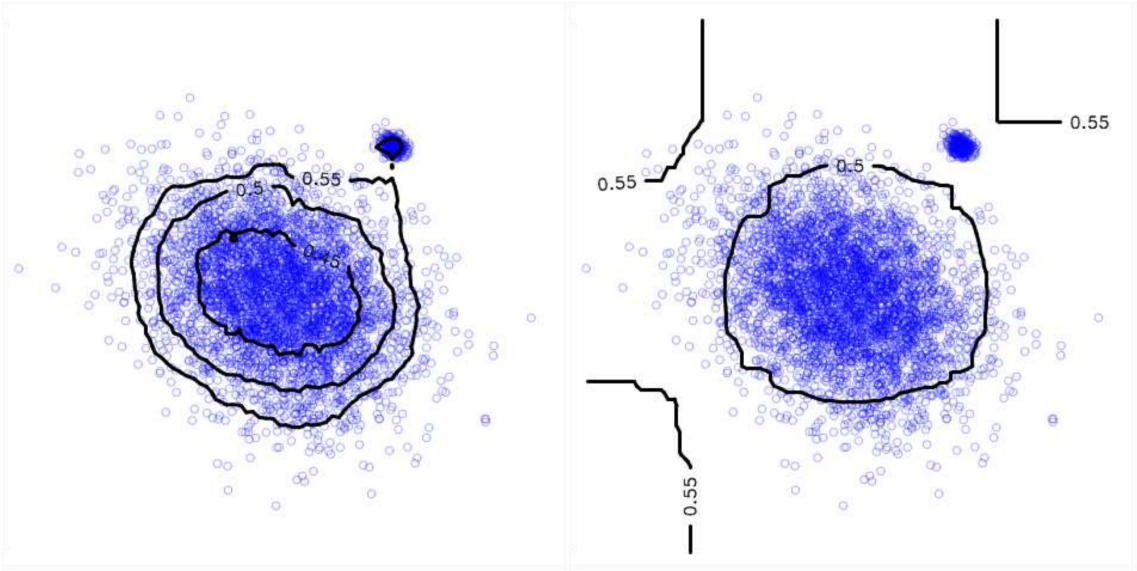
Abbildung 3.2: Der Verlauf des Anomalieindikators s eines iForests über die durchschnittliche Weglänge aller iTrees des iForests. Quelle: [13]

2. $E(h(x)) \to \psi - 1, s \to 0$: Analog dazu bedeutet ein s nahe 0, dass x erst nach unverhältnismäßig vielen Partitionen isoliert wurde, und somit somit nach Theorem 3.1.1 höchstwahrscheinlich keine Anomalie ist.

3. $E(h(x)) \to c(\psi), s \to 0.5$: Falls alle Punkte ein s nahe 0.5 zugewiesen bekommen, hat die gesamte bewertete Stichprobe keine distinkten Anomalien

Evaluationsmaximum der Weglänge

Eine Herausforderung bei der Anomalieerkennung ist zu bestimmen ab wann ein verhältnismäßig kleiner Cluster von Punkten in einem Datensatz eine Gruppe von normalen Punkten und ab wann er eine Gruppe von Anomalien darstellt. Bei einem iForest Verfahren kann dies über eine Beschränkung der zulässigen Tiefe *hlim* bei der Bestimmung der Weglänge eines Punktes parametrisiert werden. Wurden *hlim* Kanten bei der Bestimmung der Weglänge passiert, wird die Suche nach der genauen Weglänge abgebrochen und, basierend auf der bisherigen Anzahl passierter Kanten sowie der Anzahl der nach den so durchlaufenen Partitionen möglichen übrigen Blättern ψ' eine Abschätzung $e + c(\psi')$ anstatt ihrer ausgegeben. So lässt sich eine Granularität für die Abschätzung der Anomalität durch s festlegen, die in dem obigen Beispiel festlegen würde, ob der gesamte Cluster oder nur seine Ränder, als anomal gewertet werden. Dies ist illustriert durch Abbildung 3.3.



(a) hlim = 6 **(b)** hlim = 1

Abbildung 3.3: Ein Datensatz über zwei Dimensionen, über dem ein iForest konstruiert wurde. Die beiden Unterabbildungen stellen zwei Evaluationen durch diesen iForest mit unterschiedlich gesetzter Tiefenbegrenzung *hlim* dar. Die Trennlinien zeigen dabei Wertebereiche der s-Werte der Punkte an (Die Trennlinie innerhalb des kleinen Klusters in unterabbildung a) gehört dabei zu der 0.55 Linie). Zu sehen ist das $hlim = 6$ in Unterabbildung a) für eine wesentlich höhere Granularität der s-Werte, als $hlim = 1$ in b) sorgt. Dadurch gibt es in a) eine klare Abgrenzung zwischen den inneren Punkten des kleinen Kluster und den Äußeren Punkten, in Form der 0.55 Trennlinie. Diese Abggrenzung ist in b) mit $hlim = 1$ nicht vorhanden. Quelle: [13]

3.2 Implementation

Für das Trainieren eines iForests aus t iTrees über Stichproben der Größe ψ ergeben sich die folgenden Algorithmen:

Algorithmus 3.1 erhält den Trainingsdatensatz über den der iForest konstruiert wird, sowie die beiden einzigen Trainingsparameter des Verfahrens: Die Stichprobengröße ψ über die die einzelnen Bäume konstruiert werden, sowie die Anzahl der Bäume des iForests t. In Zeile 1 wird die Liste *Forest* initialisiert, welche in der Schleife in Zeile 2 mit t Bäumen gefüllt wird. Dazu wird jeweils zuerst in Zeile 3 eine zufällige Stichprobe X' der Größe ψ aus dem Datensatz X genommen, aus der in Zeile 4 von Algorithmus 3.2 ein iTree konstruiert und *Forest* hinzugefügt wird.

Algorithmus 3.2 erhält die ihm von Algorithmus 3.1 zugewiesene Stichprobe X', und konstruiert rekursiv aus dieser einen möglichen iTree. Zeile 1 stellt dabei das rekurisve Abbruchkriterium dar: Wenn ein Punkt beziehungsweise eine Gruppe von Duplikaten erfolgreich von dem restlichen Datensatz isoliert wurde. In Zeile 2 wird ein Blatt mit der Anzahl der verbliebenen Punkte in der Stichprobe als Attribut *Size* zurück an den rekur-

Algorithmus 3.1 *iForest(X,t,ψ)*

Eingabe: X - Trainingsdatensatz, t - Anzahl Bäume, ψ - subsampling size

Ausgabe: Ein Satz von t *iTrees*

1: **Initialize** $Forest$
2: **for** $i = 1$ to t **do**
3: $X' \leftarrow sample(X, \psi)$
4: $Forst \leftarrow Forest \cup iTree(X')$
5: **end for**
6: **return** $Forest$

Algorithmus 3.2 *iTree(X')*

Eingabe: X' - Input Datensatz

Ausgabe: Ein $iTree$

1: **if** $|unique(X')| = 1$ **then**
2: **return** $exNode\{Size \leftarrow |X'|\}$
3: **else**
4: Sei Q die Liste von Attibuten von X'
5: Wähle $q \in Q$ zufällig
6: Wähle $p \sim \text{Uniform}[\min_{x \in X'} x_q, \max_{x \in X'} x_q]$
7: $X_l \leftarrow filter(X', x_q < p)$
8: $X_r \leftarrow filter(X', x_q \geq p)$
9: **return** $inNode\{Left \leftarrow iTree(X_l),$
10: $Right \leftarrow iTree(X_r),$
11: $SplitAtt \leftarrow q,$
12: $SplitValue \leftarrow p\}$
13: **end if**

siven Vaterprozess gegeben. Entspricht der Datensatz nicht der Abbruchbedingung in Zeile 1 wird stattdessen eine weiter Partition durchgeführt: Dazu wird in den Zeilen 4 bis 6 ein Attribut des Datensatzes $q \in Q$ sowie ein Trennwert p aus der Wertespanne aller Punkte in q uniform zufällig gewählt. In den Zeilen 7 und 8 wird darauf die eigentliche Trennung von X' vorgenommen, Punkte mit einem Wert x_q im Attribut q kleiner als p werden in den Datensatz X_l des linken Teilknoten partitioniert, der Rest in den Datensatz X_r des rechten Teilknoten. Darauf wird in Zeile 9 ein innerer Knoten an den rekursiven Vaterprozess zurückgegeben, mit den beiden rekursiv berechneten Kindknoten als $Left$ und $Right$ Attribut, sowie die Partition die dieser Knoten darstellt, über das ausgewählten Attribut q und dem zugehörigen Trennwert p respektiv als $SplitAtt$ und $SplitValue$.

Zur Berechnung der Anomalieeinschätzung s eines Punktes x durch einen iForest, wird für jedem iTree T des iForests die Weglänge die x in T hat mithilfe von Algorithmus 3.3 berechnet. Darauf wird s über Formel 3.2 bestimmt.

Zur Evaluation der Weglänge eines Punktes über einem Baum wird der folgende Algorithmus 3.3:

Algorithmus 3.3 *PathLength(x,T,hlim,e)*

Eingabe: x - der zu evaluierende Punkt, T - ein iTree, *hlim* - eine Tiefenbegrenzung, e - momentane Weglänge; 0 zur Initialisierung

Ausgabe: Die Weglänge von x in T

1: **if** T ist Blatt oder $e \geq hlim$ **then**
2: **return** $e + c(T.size)$
3: **else**
4: $a \leftarrow T.splitAtt$
5: **if** $x_a < T.splitValue$ **then**
6: **return** $PathLength(x, T.left, hlim, e + 1)$
7: **else**
8: **return** $PathLength(x, T.right, hlim, e + 1)$
9: **end if**
10: **end if**

Algorithmus 3.3 erhält als Parameter einen Punkt x, einen iTree T , sowie den einzigen Evaluationsparameter des iForests Verfahrens, in Form der Begrenzung der Tiefe die Algorithmus 3.3 in T vordringt *hlim*. Weiterhin wird über e die, der momentan erreichten Weglänge entsprechenden Anzahl an Kanten die bereits von Algorithmus 3.3 passiert wurden gezählt. Entsprechend wird e zur Initialisierung auf 0 gesetzt. Ausgegeben wird die Weglänge von x in T, beziehungsweise die *hlim* plus die durchschnittliche Weglänge, basierend auf der Anzahl an, nach der verbleibenden Punkten die dem aktuellen Knoten untergeordnet sind, falls e den Wert von *hlim* erreicht. Zeile 1 stellt dabei die Abbruchbedingung des Algorithmus dar, welche Eintritt sobald ein Blatt von T oder die Tiefe *hlim* erreicht wurde. Darauf wird in Zeile 2 als Ergebnis der Berechnung der Weglänge, die Anzahl passierter Kanten e plus die nach Formel 3.1 berechnete, durchschnittliche Weglänge welche sich durch die verbleibenden Punkte ergibt. Ist die Abbruchbedingung nicht eingetreten, wird in Zeile 4 und 5 die Partition des aktuellen Knotens aus diesem, in der Form von *splitAtt* als a und $splitValue$ ausgelesen und es wird überprüft ob x nach seinem Wert x_a im Attribut a in die Teilmenge des linken oder des rechten partitioniert worden wäre. Dementsprechend wird die Suche nach der Weglänge von x in Zeile 6 beziehungsweise Zeile 8 in dem linken beziehungsweise dem rechten Teilknoten weitergeführt.

Kapitel 4

Robust Random Cut Forest

In diesem Kapitel wird einer der beiden, auf den Testdatensatz angewendeten Verfahren, der **Robust Random Cut Forest** (von hier an RRCF) in seinen Grundzügen beschrieben. Das Kapitel orientiert sich dabei an Artikel [9] und dem zugehörigen Supplement [10].

4.1 Notationen

Die in diesem Kapitel verwendeten Notationen lehnen sich an die in dem Papier [9] verwendeten an:

- $\mathbb{E}$ ddd
- $\mathbb{P}r$ ddd
- $\mathcal{T}$ ddd

4.2 RRCF Theory

Der RRCF basiert auf, und ähnelt somit vielerlei dem in Kapitel 2 vorgestellten Isolation Forest. So versucht der RRCF ebenfalls Anomalien direkt vom Datensatz zu isolieren statt ein Profil einer normalen Klasse zu definieren. Auch basiert der RRCF ebenfalls auf dem Zufallsprinzip, und mittelt sein Ergebnis aus den einzelnen Ergebnissen der unabhängig konstruierten Bäume aus denen er besteht. Unterscheiden tut sich der RRCF allerdings in zweierlei Hinsicht:

1. Bei der Konstruktion der Bäume des RRCFs, werden die Dimensionen über die der zugrundeliegende Datensatz geteilt wird nicht uniform-zufällig, sondern nach der Größe der in ihnen vorhandenen Unterschieden der Punkte des Datensatzes gewichtet ausgewählt. So kann der Einfluss von unwichtigen Dimensionen (siehe Sektion 2.1.1) reduziert werden, und die Zugrundeliegenden Wahrscheinlichkeiten jedes Baumes über einen Datensatz bleiben konstant, unabhängig davon wie dieser Baum zustande kam.

2. Das Kriterium nach dem die Ausgabe des RRCFs berechnet wird bezieht sich nicht auf die Tiefe der Punkte, sondern auf den Effekt die eine beliebige diesen Punkt beinhaltende Gruppe von Punkten, auf die gesamte Modellkomplexität des Baumes hat. Diese Metrik ist allgemein robuster, ins besonders können Duplikate einer Anomalie nicht mehr ihre Erkennung als solche verhindern

In den folgenden Sektionen werden diese Unterschiede, sowie die dem RRCF zugrunde liegenden Theoreme dargestellt.

4.2.1 RRCF Aufbau

Analog zu anderen Forest-Ansätzen aus dem Gebiet des maschinellen Lernens, besteht ein RRCF aus mehreren unabhängig voneinander konstruierten **Robust Random Cut Trees** (RRCT):

4.2.1 Definition (RRCT). Ein RRCT wird über ein Datensatz S mit j Dimensionen wie folgt generiert:

1. Wähle eine Dimension i aus den j Dimensionen. Dabei hat jede Dimension eine Wahrscheinlichkeit proportional zu $\frac{l_i}{\Sigma_j l_i}$, mit $l_i = \max_{x \in S} x_i - \min_{x \in S} x_i$ ausgewählt zu werden.

2. Wähle $X_i \sim \text{Uniform}[\min_{x \in S} x_i, \max_{x \in S} x_i]$

3. Teile S in $S_1 = \{x \mid x \in S, x_i \leq X_i\}$ und $S_2 = S \setminus S_1$ und fahre rekursiv auf S_1 und S_2 fort, solange $|S_1| > 1$ beziehungsweise $|S_2| > 1$.

In Schritt 1 wird die Dimension ausgewählt über die der Datensatz bei der Konstruktion des Baumes getrennt wird. Ein wichtiger Unterschied bei der Konstruktion eines RRCT zu der Konstruktion eines Baumes in einem Isolation Forest, wie in [13], ist dabei, dass die zur Trennung genutzte Dimension i nicht Uniform über alle Dimensionen j ausgewählt wird. Stattdessen werden die Dimensionen proportional dazu wie stark die Werte der einzelnen Punkte sich in den Dimensionen unterscheiden gewichtet bevor eine von ihnen gewählt wird.

In Schritt 2 wird darauf analog zum Isolation Forest Verfahren ein Trennwert X_i uniform aus der Wertespanne aller Punkte $x \in S$ der in Schritt 1 ausgewählten Dimension gewählt.
In Schritt 3 wird der Datensatz S dann über X_i partitioniert, sodass $S1$ die Datenpunkte enthält die in Dimension i größer oder gleich groß wie X_i sind und S_2 die verbliebenen Datenpunkte, welche in i einen kleiner als X_i sind.

Beispielhaft würden in dem Datensatz von Tabelle 4.1 bei dem ersten Durchlauf der Baumkonstruktion die Dimensionen 1, 2 und 3 mit einer jeweiligen Wahrscheinlichkeit von

Tabelle 4.1: Ein Beispiel Datensatz über 3 Dimensionen mit numerischen Werten mit $S = \{x, y, z\}$ sowie die von Definition 4.2.1 in Schritt 1 berechnete Wahrscheinlichkeit $\frac{l_i}{\Sigma_j l_i}$ das S in Schritt 3 über die jeweilige Dimension partitioniert wird

Dimension	**x**	**y**	**z**	$\frac{l_i}{\Sigma_j l_i}$
1	5	10	6	$\frac{5}{35}$
2	2	8	12	$\frac{10}{35}$
3	25	5	5	$\frac{20}{35}$

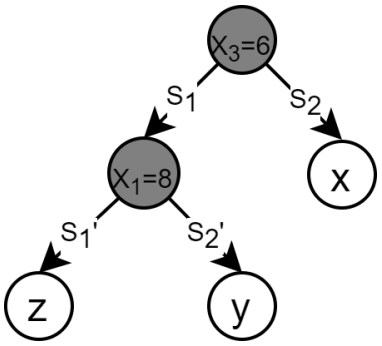


Abbildung 4.1: Ein möglicher, nach Definition 4.2.1 konstruierter RRCF über den in Tabelle 4.1 dargestellten Datensatz S. Die erste Partition erfolgte über die dritte Dimension mit einem nach Schritt 2 zufällig bestimmten Grenzwert von 6. Da S_1 darauf mehr als einen Punkt enthielt erfolgte eine weitere Partition über die erste Dimension und einen Grenzwert von 8

$\frac{1}{7}$, $\frac{2}{7}$ und $\frac{4}{7}$, als die Dimension über die S partitioniert wird, ausgewählt werden. Je nach gewählter Dimension wird X_i darauf aus den Wertespannen $[5, 10]$, $[2, 12]$ beziehungsweise $[5, 25]$ uniform-zufällig gewählt. Ein möglicher RRCF, welcher sich aus dem in Tabelle 4.1 dargestellten Datensatz ergibt ist in Abbildung 4.1 dargestellt.

Jeder innere Knoten eines RRCTs $T = \mathcal{T}(S)$, über einen Datensatz S, entspricht demnach einer Partition, und enthält die entsprechende Dimension und den Grenzwert für diese Partition. Die Blätter des RRCTs entsprechen den einzelnen Punkten in S, welche über eine Reihe von Partitionen, entsprechend der Knoten entlang des Pfades von der Wurzel von T zu dem jeweiligen Blatt, von allen anderen Punkten in S isoliert wurden.

4.2.2 Distanzbeibehaltung bei der RRCT Konstruktion

Damit ein RRCF zur Anomalieerkennung eingesetzt werden kann, muss gezeigt werden, dass die RRCTs in der er die Punkte des zu untersuchenden Datensatzes auf eine Art speichert, die die Distanz zwischen den Punkten Beibehält. Ein Datenpunkt der sich im Datensatz anomal abzeichnet muss, auch in einem aus diesem Datensatz gebauten RRCT als anomal erkennbar sein. Dies ist gegeben durch folgendes Theorem:

4.2.2 Theorem (Distanzbeibehaltung). *Sei ein RRCT $\mathcal{T}$ über einen Datensatz S mit d Dimensionen konstruiert. Sei das Gewicht eines Knotens von $\mathcal{T}$ die Summe der Länge der Kanten der minimal begrenzenden Box der diesem Knoten untergeordneten Punkte $\sum_i l_i$, und sei die Baumdistanz zwischen zwei Knoten $u, v \in S$ das Gewicht des letzten gemeinsamen Vorfahrens von u und v. Dann ist die Baumdistanz von u und v mindestens $L_1(u, v)$ und in Erwartung maximal ein Vielfaches von $L_1(u, v)$ um den Faktor:*

$$\mathcal{O}(d \log \frac{|S|}{L_1(u, v)}) \tag{4.1}$$

Beweis von Theorem 4.2.2

Sei für einen Datensatz S l_i erneut als die Wertespanne zwischen den niedrigsten und höchsten Wert von S in der Dimension i definiert. Sei $B(S)$ die $MinimalBoundingBox$(MBB) um alle Punkte in S. Sei dann $P(S) = \sum_i l_i$ die Summe der Seitenlängen von $B(S)$. Es ergibt sich:

4.2.3 Lemma. *Die Wahrscheinlichkeit das $u, v \in S$ durch eine Partition von S nach Definition 4.2.1 getrennt werden ist gegeben durch:*

$$\frac{1}{P(S)} \sum_i |u_i - v_i| \tag{4.2}$$

$P(S)$ entspricht der Summe der Länge aller Wertespannen l_i in denen in Schritt 1 und 2 von Theorem 4.2.1 ein Schnittpunkt gewählt wird. $\sum_i |u_i - v_i|$ entspricht der Summe der Wertespannen, auf denen die Wahl eines Schnittpunktes u und v trennen würde. Das Lemma folgt.

4.2.3 RRCF Instandhaltung

In diesem Abschnitt wird gezeigt das von einem RRCT $\mathcal{T}(S)$ effizient ein Punkt x gelöscht oder hinzugefügt werden kann, also die jeweiligen RRCTs $\mathcal{T}(S - \{x\})$ und $\mathcal{T}(S \cup \{x\})$ effizient erzeugt werden können.

Löschen eizelner Punkte

Soll ein Punkt u aus dem Baum $\mathcal{T}$ gelöscht werden, so muss lediglich der Elternknoten k von u, welcher die Trennung mithilfe der u isoliert wurde darstellt, mit gelöscht werden, und der Elternknoten von k bekommt als neues Kind, dass nun verwaiste Kind von k. Siehe Bild ???

4.2.4 Theorem (Konsistenz der inneren Probabilität). *Sei ein RRCT $\mathcal{T}$ welcher über einen Datensatz S konstruiert wurde. Wird ein Punkt $u \in S$ wie oben skizziert gelöscht, so hat der daraus resultierende Baum die gleiche Probabilität gegenüber über welche Dimensionen $\mathcal{T}$ bei seiner Konstruktion partitioniert wird, wie ein RRCT der über $S - u$ konstruiert wurde. Parallel dazu hat ein RRCT der über $S \cup \{v\}$ mit $v \notin S$ konstruiert wird, die gleiche Probabilität wie der RRCT der aus dem hinzufügen von v zu $\mathcal{T}$ resultiert*

Dieses natürliche Verhalten gegenüber dem hinzufügen und löschen von Punkten des RRCF Verfahrens, setzt es von vielen anderen Partitionierungsverfahren ab [9], insbesondere auch von anderen Baum konstruierenden Anomalieerkennungsverfahren Verfahren wie das Isolation Forest Verfahren, welche die über die zu partitionierende Dimension uniformzufällig auswählen. Dies zeigt sich durch folgendes Beispiel:

Unterschiede beim Löschen eines Punktes

Beispiel mit Bild pro Fall 4+2 :)

Die so ermöglichten dynamischen Änderungen an den durch das RRCF Verfahren konstruierten Bäumen, ermöglicht unter anderem die effiziente Anomalieerkennung auf gestreamten Daten, da die neu eintreffenden Punkte in die bestehenden Bäume mit eingefügt werden können, anstatt das diese von Grund auf neu gebaut werden müssten.

4.2.5 Theorem (Die RRCT Konstruktion ist Stichproben unabhängig). *Sei S eine Stichprobe eines Datensatzes. Es kann ein RRCF über S gebildet werden, selbst wenn S dynamisch aktualisiert wird.*

Das Theorem folt aus den bisher definierten. Theorem 4.2.2 sagt aus, dass der RRCT die in S gegebenen Abstände beibehält. Jedes auf S angewendete Stichprobenverfahren, welches die gewünschten Zusammenhänge beibehält, kann dementsprechend auch in einem RRCT abgebildet werden. Mit Theorem 4.2.4 ist der Prozess der RRCT Konstruktion unabhängig

von den angewendeten Stichprobenverfahren. Soll beispielsweise eine Stichprobe von S der Größe $\rho|S|$, mit $\rho < 1$ uniform-zufällig erstellt werden, so müssen kann entweder ein RRCT über $\rho|S|$ uniform-zufällig ausgewählte Punkte von S konstruiert werden, oder es können $|S| - \rho|S|$ Punkte uniform-zufällig bestimmte Punkte aus einem bestehenden RRCT über S gelöscht werden. Beide Vorgehensweisen resultieren in den selben Probabilitäten, gegenüber der Struktur und den ausgewählten Dimensionen über die die Stichprobe partitioniert wurde, für den resultierenden Baum. Parallel dazu kann jedes weiter Stichprobenverfahren vor oder auch abhängig von der Größe des resultierenden Baumes nach der Konstruktion des RRCT angewandt werden. Es folgt:

4.2.6 Theorem. *Existiert ein Verfahren welches eine Stichprobe des Datensatzes S per Downsampling erstellt dann existiert für jede Downsampling Rate ein Algorithmus der einen RRCT über die Stichprobe erzeugt indem er Punkte aus dem RRCT über S löscht.*

Somit ist es möglich die Menge an Punkten mit der ein RRCF konstruiert wurde, nach seiner Konstruktion anzupassen. Aus Theorem 4.2.5 ergibt sich weiterhin:

4.2.7 Theorem. *Sei ein RRCT über einen Datensatz S konstruiert. Sei $u \notin S$. Da wir effizient den RRCT über $S \cup \{p\}$ konstruieren können indem wir u zu $\mathcal{T}(S)$ hinzufügen, können wir effizient den erwarteten Effekt von u auf die Platzierung der anderen Punkte in S bestimmen, sowie die erwartete Tiefe die u in $\mathcal{T}(S \cup \{u\})$ hat.*

Diese Möglichkeit, kontrafaktische Fragen gegenüber dem Einfügen von u in $\mathcal{T}(S)$ effizient zu beantworten, eignet sich Intuitiv der Anomalieerkennung. So kann entweder die erwartete Tiefe von u bestimmt werden, um über Theorem 4.2.2 den Grad der Normalität von u abzuschätzen, oder es kann der Unterschied den u zwischen $\mathcal{T}(S)$ und $\mathcal{T}(S \cup \{u\})$ erzeugt, bemessen werden. Eine konkrete Metrik dazu wird in der nächsten Sektion in Form des $Codisplacements(CoDisps)$ vorgestellt.

4.3 Anomalieerkennung über RRCF

Um zu spezifizieren wie genau ein anomaler Punkt in einem $RRCF$ erkannt wird, sei hier auf das Beispiel in Kapitel 2, der Menge bestehend aus schwarzen Kugeln und Würfeln, sowie einer grünen Kugel, zurückgegriffen. Hier lassen sich 2 Arten der Anomalieausprägung definieren:

1. Eine Anomalie ist einfach zu beschreiben, die grüne Kugel unterscheidet sich zwar nicht im Merkmal der Länge, aber im Merkmal der Farbe stark von den anderen Objekten der Menge. Ihre Unterscheidung von der Menge ist leicht abzugrenzen. Diese Kategorisierung ist die in Kapitel 2 verwendete.

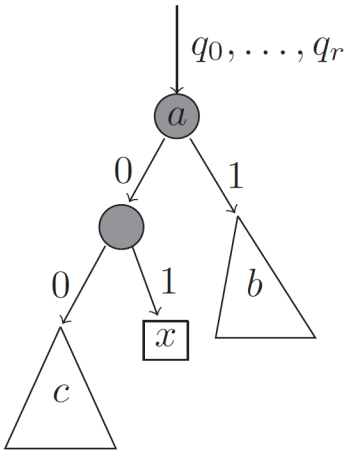


Abbildung 4.2: Ein Teilbaum T_1 über die Menge S_1, eines RRCTs T, dessen Wurzel in T die Tiefe $r+1$ hat. Der Knoten a stellt eine Partitionierung von S_1 in zwei Teilemengen da. $q_0, ..., q_r$ sind die Bits die die Position von a in T beschreiben. Quelle: [9]

2. Die Existenz einer Anomalie in einer Menge, macht es schwieriger diese Menge zu beschreiben. So müssen die Objekte der Menge nun nicht mehr nur noch nach Form, sondern auch nach Farbe differenziert werden. Der Fokus einer Beschreibung wird von einer Mehrzahl der Objekte zu einem einzigem verschoben.

Die beiden Anomalieausprägungen folgen auseinander. Das eine Anomalie über ihr hervorstechendes Merkmal einfach zu beschreiben ist, ist äquivalent dazu, dass die Beschreibung der Merkmale einer Menge einfacher wäre, würde diese Anomalie mit ihrem besonderen Merkmal beziehungsweise ihrem besonders ausgeprägtem Merkmal nicht existieren.

Der RRCF Algorithmus versucht die in Punkt 2 definierte, durch einen Punkt erzeugte Verschiebung (*Disp*) zu bestimmen. Dazu wird zuerst die Komplexität eines RRCTs definiert, um eine exakte Relation über den Effekt der im RRCT untergebrachten Punkte auf die Komplexität von diesem zu bestimmen.

4.3.1 Modellkomplexität eines RRCT

Sei jedem Zweig in einem RRCT ein Bit zugeordnet. Ein linker Zweig wird durch das Bit 0 und ein rechter Zweig durch das Bit 1 gekennzeichnet. Der Platz von jedem Punkt x in einem RRCT ist dann in diesem eindeutig durch die Folge an Bits entlang der Zweige von der Wurzel zu dem Punkt x, bestimmt. Siehe Abbildung 4.2, wo der Platz von x in T durch die Bitfolge $q_0, ..., q_r, 0, 1$ definiert ist. Es bietet sich die folgende Definition 4.3.2 der Modellkomplexität eines RRCTs an:

4.3.1 Definition (Tiefe eines Punktes in x). Gegeben sei ein Satz an Punkten S und sei $T = \mathcal{T}(S)$ ein RRCT über S. Sei ein Punkt $x \in S$, mit der zugehörigen Bitfolge b. Dann sei:

$$f(x, S, T) = |b| \tag{4.3}$$

die Tiefe von x in T.

Die Tiefe eines Knotens eines Binärbaumes entspricht der Anzahl der Zweige zwischen ihm und der Wurzel. Da sich pro Zweig ein Bit in der zugeordneten Bitfolge eines Knotens eines RRCTs ergibt, folgt die Gleichung 4.3.

4.3.2 Definition (Modellkomplexität). Gegeben sei ein Satz an Punkten S und sei $T = \mathcal{T}(S)$ ein RRCT über S. Sei $f(x, S, T)$ mit $x \in S$ die Tiefe des Punktes x in T. Dann ist die Modellkomplexität von T:

$$|M(T)| = \sum_{x \in S} f(x, S, T) \tag{4.4}$$

Die definierte Modellkomplexität $|M(T)|$ entspricht somit der Summe der Länge der Bitfolgen aller Punkte in dem RRCT T. Anomalien in einem Datensatz sorgen somit für eine höhere Modellkomplexität, da diese nach 4.2.4, durch ihre Hervorstechenden Merkmale früh im RRCT Konstruktionsprozess isoliert werden, die restlichen Punkte also einen gebündelt einen weiteren Zweig herunter schickt.

4.3.2 Verschiebung der Modellkomplexität durch einen Punkt x

Parallel zu der Modellkomplexität $|M(T)|$ ist die Modellkomplexität des RRCTs $T' = \mathcal{T}(S - \{x\})$, also des RRCTs der aus der Entfernung des Punktes x aus dem RRCT T nach Theorem 4.2.4 gegeben durch:

$$|M(T')| = \sum_{x \in S - \{x\}} f(x, S - \{x\}, T) \tag{4.5}$$

Der Effekt den x auf die Modellkomplexität von T hat ist demnach:

$$|M(T)| - |M(T')| \tag{4.6}$$

Dabei ist zu beachten das der Term 4.6 nur für den Effekt gilt den x auf $|M(T)|$ hat, da nach Theorem 4.2.4 mit gegebenen T und x der durch das Entfernen von x aus T produzierte RRCT T' deterministisch bestimmt ist. Umgekehrt kann aber jeder einzelne T' aus beliebig vielen möglichen T und x hergeleitet werden, es handelt sich um eine viele-zu-einem Beziehung. Somit trifft der Term 4.6 keine Aussage über den Effekt den x in T' haben würde.

Ausgeweitet auf alle möglichen RRCTs $T = \mathcal{S}$ und allen möglichen $T = \mathcal{S} - \{\S\}$ ergibt sich für die erwartete Verschiebung der Modellkomplexität, die x im durchschnitt in allen T verursacht:

$$\begin{aligned}
\mathbb{E}_T[|M(T)|] - \mathbb{E}_{T'}[|M(T')|] &= \sum_T \sum_{y \in S} \mathbb{P}r[T] f(y, S, T) \\
&\quad - \sum_{T'} \sum_{y \in S - \{x\}} \mathbb{P}r[T'] f(y, S - \{x\}, T') && (4.7) \\
&= \sum_T \sum_{y \in S - \{x\}} \mathbb{P}r[T] f(y, S, T) \\
&\quad - \sum_{T'} \sum_{y \in S - \{x\}} \mathbb{P}r[T'] f(y, S - \{x\}, T') \\
&\quad + \sum_T \mathbb{P}r[T] f(x, S, T) && (4.8) \\
&= \sum_T \sum_{y \in S - \{x\}} \mathbb{P}r[T] \Big(f(y, S, T) - f(y, S - \{x\}, T') \Big) \\
&\quad + \sum_T \mathbb{P}r[T] f(x, S, T) && (4.9)
\end{aligned}$$

Der Term 4.7 ergibt sich aus 4.3.2 und entspricht der durchschnittlichen Modellkomplexität aller über nach Definition 4.2.1 konstruierten RRCTs T und T'. In dem Term 4.8 ist die durchschnittliche Modellkomplexität des Punktes x getrennt von der des Rest des Baumes dargestellt. Wie oben dargestellt ist nach 4.2.4 mit gegebenen T und x, das Resultat T' der Entfernung des Punktes x aus T deterministisch gegeben und es gilt somit:

$$\sum_{T'} \sum_{y \in S - \{x\}} \mathbb{P}r[T'] f(y, S - \{x\}, T') = \sum_T \sum_{y \in S - \{x\}} \mathbb{P}r[T'] f(y, S - \{x\}, T') \qquad (4.10)$$

Woraus der Term 4.9 folgt und sich folgende Definition gibt:

4.3.3 Definition (Verschiebung (*Displacement*) eines Punktes). Sei ein Satz an Punkten S und sei ein Punkt $x \in S$. Seien $T = \mathcal{T}(S)$ und $T' = \mathcal{T}(S - \{x\})$ RRCTs über S. Die bitweise Verschiebung die der Punkt x im RRCT T verursacht ist:

$$Disp(x, S) = \sum_T \sum_{y \in S - \{x\}} \mathbb{P}r[T] \Big(f(y, S, T) - f(y, S - \{x\}, T') \Big) \qquad (4.11)$$

Zu bemerken gilt, dass die totale durch x durchnitlisch verursachte Vergrößerung der Modellkomplexität gegeben ist durch:

$$\mathbb{E}_T[|M(T)|] - \mathbb{E}_{T'}[|M(T')|] = Disp(x, S) + \sum_T \mathbb{P}r[T] f(x, S, T) \qquad (4.12)$$

, also der Summe der Bits die zu der Bit-Repräsentation der Punkte $y \in S - \{x\}$ durch x hinzukommen, plus der Bits die x selbst darstellen. Der Fokus der Anomalieerkennung

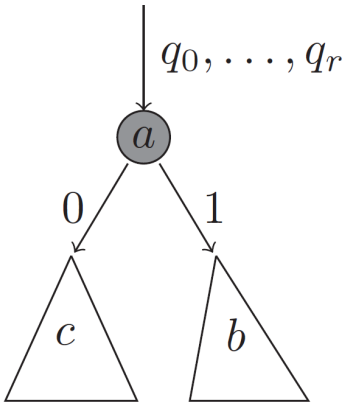


Abbildung 4.3: Ein Teilbaum T_2 über die Menge S_2, eines RRCTs T, dessen Wurzel in T die Tiefe $r+1$ hat. Der Knoten a stellt eine Partitionierung von S_2 in zwei Teilemengen da. $q_0, ..., q_r$ sind die Bits die die Position von a in T beschreiben. Quelle: [9]

durch RRCFs liegt demnach auf der Erkennung eines Steigens der Komplexität des Datensatzes den ein Punkt des Datensatzes hervorruft, anstatt auf das Hervorstechen des Punktes an sich. Die Benutzung des Wortes Verschiebung, ergibt lässt sich über folgendes Lemma herleiten:

4.3.4 Lemma. *Die in durch einen Punkt $x \in S$ verursachte Verschiebung in einem RRCT $T = \mathcal{T}$ entspricht der Menge an Punkten, die Geschwister von x sind*

Beweis Lemma 4.3.4 Orientiert an Abbildung 4.2, ist die Bitrepräsentation jedes Punktes in c, also jedes Punktes welcher in dem Baum T ein Geschwister von x ist, gegeben durch:

$$q_0, ..., q_r, 0, 0, ... \tag{4.13}$$

Repräsentiert in Abbildung 4.3, welche den Teilbaum darstellt der sich aus dem Entfernen von x aus dem in 4.2 dargestellten RRCT ergibt, fällt durch das Entfernen nach4.2.4, von x aus T ein Knoten auf dem Pfad der Wurzel von T zu den Punkten in dem Bereich c weg, womit sich für diese eine neue Bitepräsentation gibt:

$$q_0, ..., q_r, 0, ... \tag{4.14}$$

Da der Pfad von der Wurzel von T, zu allen Knoten außerhalb des Bereiches c durch das Löschen von x unverändert bleibt, ergibt sich beziehend auf die Definition 4.3.1, für den Effekt von x auf die Länge der Bitrepräsentation jedes anderen Punktes in T:

$$f(y, S, T) - f(y, S - \{x\}, T') = \begin{cases} 1, & y \in c \\ 0, & \text{otherwise} \end{cases} \tag{4.15}$$

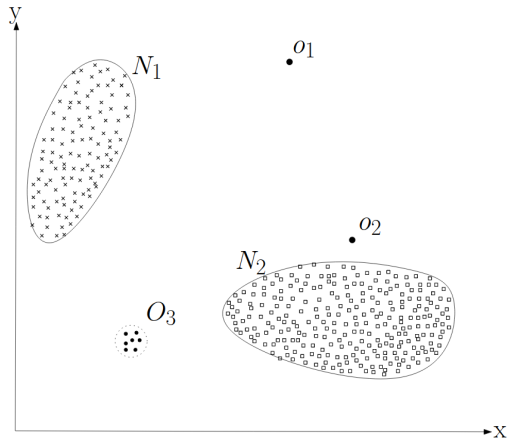


Abbildung 4.4: Ein Beispieldatensatz mit zwei Anomalien o_1 und o_2, sowie eine Punktegruppe O_3 von 7 Anomalien. Die Gruppen N_1 und N_2 stellen die Inliner des Datensatzes da. Quelle: [6]

Es folgt für die Verschiebung von x in einem gegeben Baum T:

$$Disp_T(x, S) = |c| \tag{4.16}$$

4.3.3 Codisp

Definition 4.3.3 bietet eine Möglichkeit der Anomaliedefinition. Diese ist allerdings stark anfällig gegenüber Duplikaten, wie in Sektion 2.1.1 definiert. Enthält die oben definierte Menge an Objekten 2 grüne Kugeln, so würde das Entfernen einer Kugel die Komplexität der Beschreibung der Menge nicht wesentlich vereinfachen. Ein genaueres Beispiel ergibt sich wie folgt:

Bei dem durch Abbildung 4.4 dargestellten Datensatz S, würde ein auf diesem konstruierten RRCT die Punkte o_1 und o_2, basierend auf Theorem 4.2.2, aufgrund ihrer hohen Distanz L_1 zu allen anderen Punkten des Datensatzes wahrscheinlich schnell isolieren. $Disp(o_1, S)$ sowie $Disp(o_2, S)$ wäre, aufgrund ihrer somit folgenden hohen Anzahl an Punkten in Geschwisterknoten, ebenfalls hoch im Vergleich zu den Punkten in N_1 und N_2. Die Punkte in O_3 würde in Erwartung, aufgrund ihrer hohen Distanz $L1$, ebenfalls schnell von allen Punkten nicht in O_3 getrennt werden. Aufgrund ihrer geringen Distanz $L1$ untereinander würde die dafür verantwortliche Partitionierung, in Erwartung alle Punkte in O_3, nach Schritt 3 der Definition 4.2.1 in eine Teilmenge partitionieren. In Erwartung ergibt sich ein RRCT wie in Abbildung 4.4. Da jedes Blatt, welches einen Punkt von O_3 enthält, eine geringe Anzahl an Blättern hat die von seinem Geschwisterknoten abstammen, ist somit $Disp(o_3, S)$ für alle $o_3 \in O_3$ gering. Die Punkte O_3 können über Definition 4.3.3 nicht als Anomalie erkannt werden.

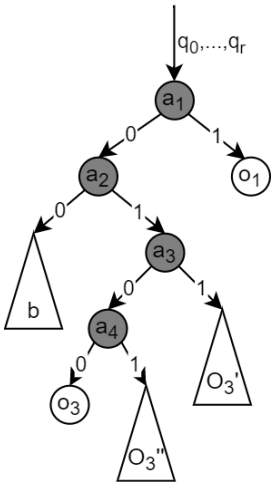


Abbildung 4.5: Ein Teilbaum, welcher

Robustheit gegenüber Duplikaten

Um über die Modellkomplexität einen anomalen Punkt $x \in S$ als solchen zu erkennen selbst wenn S Duplikate oder Beinah-Duplikate von x enthält, muss demnach das Vergleichsmodell betrachtet werden, bei dem ein Set an Punkten C, mit $x \in C$ entfernt wurden. Analog zu Term 4.9 ergibt sich für den erwarteten durchschnittlichen Unterschied in der Modellkomplexität aller RRCTs $T = \mathcal{T}(S)$ und $T'' = \mathcal{T}(S - C)$:

$$\mathbb{E}_T[|M(T)|] - \mathbb{E}_{T'}[|M(T'')|] = Disp(C, S) + \sum_T \sum_{y \in C} \mathbb{P}r[T] f(y, S, T) \tag{4.17}$$

, wobei $Disp(C, S)$ der erwarteten Bit-Verschiebung, die die Punkte C im Durchschnitt über alle T verursachen entspricht:

$$Disp(C, S) = \sum_T \sum_{y \in S - C} \mathbb{P}r[T] \Big(f(y, S, T) - f(y, S - C, T'') \Big) \tag{4.18}$$

Die Bit-Verschiebung von x entspricht damit, basierend auf Term 4.18 und der Annahme, dass alle Punkte in C die gleiche Bit-Verschiebung zugeschrieben werden sollte, da es sich bei diesen in Erwartung um Duplikate oder Beinah-Duplikate von x handelt, $Disp(C, S)/|C|$. Dementsprechend wäre eine Methodik C zu wählen die Ermittlung des folgenden Maximums:

$$\max_{x \in C \subseteq S} Disp(C, S)/|C| \tag{4.19}$$

Dieser Methodik folgen allerdings zwei Probleme:

1. Die mögliche Anzahl an Sets von Punkten $x \in C \subseteq S$ wächst exponentiell zu S, weshalb Anomalieerkennung über Methodik 4.19 ineffizient wäre.

2. Wird S gestreamed, und der RRCF live über den Stream konstruiert, sind zum Zeitpunkt der Bewertung von x noch nicht alle Punkte von S, also nicht alle möglichen Punkte von C, sondern nur ein Sample $S' \subset S$ bekannt. Somit ist Methodik 4.19 nicht für Streaming-Daten geeignet.

Zur Lösung dieser Probleme, darf C für unterschiedliche Samples S' verschieden gewählt werden. Es ergibt sich die folgende Definition des $CollusiveDisplacements(Codisp)$, oder der Bit-Verschiebung mithilfe einer Gruppe von Punkten, eines Punktes:

4.3.5 Definition (CoDisp). Sei ein Datensatz S gegeben. Die erwartete durchschnittliche Bit-Verschiebung eines Punktes x in allen möglichen RRCTs $T = \mathcal{T}(S')$ über ein Sample $S' \subset S$ und die darüber gegebenen $T'' = \mathcal{T}(S - C)$, ist gegeben durch:

$$CoDisp(x, S, |S'|) = \mathop{\mathbb{E}}_{S' \subseteq S, T} \left[\max_{x \in C \subseteq S} \frac{1}{|C|} \sum_{y \in S - C} f(y, S', T) - f(y, S' - C, T'') \right] \tag{4.20}$$

Dabei kann T'' wieder aufgrund von Theorem 4.2.4 deterministisch von allen Kombinationen von T und C abgeleitet werden. Mit der nun durch $Codisp()$ gegebenen, gegen Duplikate robusten Möglichkeit der Ermittlung des Effektes den ein Punkt innerhalb eines RRCTs auf die Modellkomplexität seines RRCTs hat, ergibt sich die zentrale Definition des RRCFs:

4.3.6 Definition. Die Ausreißer eines Datensatzes haben in einem über den Datensatz, oder über einem Sample über den Datensatz konstruierten RRCT in Erwartung einen hohen CoDisp()

Weiterhin gilt:

4.3.7 Lemma. *Die CoDisp(x,Z,|S|) kann effizient bestimmt werden*

Beweis von Lemma 4.3.7 Analog zu dem Beweis von Lemma 4.3.4 ist der U

Kapitel 5

Tests auf Niederspannungsdaten

Im Rahmen dieser Arbeit wurde die Performance von zwei Anomalieerkennungsverfahren auf dem ihr zugrundeliegendem Testdatensatzes beurteilt. In diesem Kapitel wird nun zuerst auf die Eigenschaften des Datensatzes eingegangen, und darauf auf die Eignung der angewandten Anomalieerkennungsverfahren für diesen, sowie auf die Details ihrer jeweiligen Implementierung.

5.1 Aufmachung der Testdaten

Der Testdatensatz wurde im Jahr 2018 von in 17 unterschiedlichen Stellen des deutschen Niederspannungsnetz angebrachten Messstationen aufgezeichnet. Bemessen wurde dabei die absolute Spannung aller drei Stromphasen in Abständen von 9.5 Sekunden, wobei jede Phase zeitgleich bemessen wurde. Je nach Station bilden die aufgenommenen Daten einen Zeitraum von mindestens 3 bis zu maximal 8 Monaten ab. Insgesamt enthält der Datensatz über alle Phasen aller Stationen 66 Millionen Punkte, welche sich in wie in Tabelle 5.1 dargestellt aufteilen.

Die absoluten Spannungswerte bewegen sich in einem Bereich von 182 V bis 236 V. Über den Gesamtmesszeitraum ergeben sich dabei starke saisonale Unterschiede, unter anderem Abhängig von der Jahreszeit und der Tagesart. **??**

Die Punkte des Testdatensatzes wurden in der Nachbearbeitung zu bis zu 5 Anomalieklassen zugeordnet. Diese sind:

1. **Sprunganomalien:** Punkte direkt nach einer Trafostufung, also eine drei Punkte Kombination einer Messung deren jeweilige Spannungen in der jeweilig gleichen Phase entweder jeweils ungewöhnlich größer oder jeweils ungewöhnlich kleiner sind, als die 3 Spannungen der 3 Punkte Kombination der vorherigen Messung der Messstation.

2. **Zeitanomalien** Punkte direkt nach einer Messlücke, also Punkte deren Zeitpunkt weit länger als die üblichen 10 Sekunden hinter dem Zeitpunkt ihres Vorgängerpunkts liegt.

Tabelle 5.1: Der Testdatensatz aufgeschlüsselt nach den Stationen und der Anzahl an Punkten aller drei Phasen der Station, welche den jeweiligen Anomalieklassen zugehören. Die letzten beiden Reihen stellen die Gesamtgröße der Anomalieklassen in dem Datensatz dar, sowie den prozentualen Anteil den diese insgesamt an dem Datensatz haben

Station	Punkte	Sprunga.	Zeita.	Phasena.	Saisona.	Stationsa.
4352	3798525	2853	6723	66	4085	98
0928	4854720	4377	84	12	5455	1300
0120	5521035	6867	54	0	0	782
0691	1974597	10563	4662	13560	3647	9205
4366	4814937	4497	129	9	5474	1158
0942	4032360	4965	8640	0	6037	94
4609	4365249	23865	10350	39414	17326	1547
0595	4276122	8310	12201	0	10	473
4623	2254374	5427	3300	0	0	81
0888	4864896	4677	99	375	4791	1382
0637	946380	3513	8346	6204	1689	3059
0993	5303775	7971	909	5589	1839	34259
3723	5767935	6999	57	0	0	2241
4367	4863876	4461	84	327	4930	1369
1035	4061799	5403	9180	564	2493	7790
1145	2194560	2478	657	0	3724	177
1146	2156118	2937	1569	48	4604	1062
gesamt	66051258	110163	67044	66168	66104	66077
anteil	1	0.0016678	0.001015	0.0010018	0.001	0.0010004

3. **Phasenanomalien:** Punkte welche sich von den jeweiligen Punkten der anderen beiden Phasen absetzen, also Punkte deren Spannung stark von der Spannung von mindestens einer Spannung der beiden zugleich aufgenommenen Punkte der beiden anderen Phasen unterscheidet.

4. **Saisonanomalien:** Punkte die mit dem saisonalen Trend der Zeitreihe brechen, also Punkte deren Spannungswerte sich stark von den Werten vorherigen Punkte unterscheiden, welche zu einer ähnlichen Uhrzeit und in der gleichen Tagesart gemessen wurden. Dabei wurde zwischen Werktagen und der Kombination aus Feier- und Wochenendtagen unterschieden.

5. **Stationsanomalien:** Punkte welche gegen den Trend des durchschnittlichen Verlaufs aller Zeitreihen verstoßen, also Punkte deren Spannung sich stark von dem Durchschnitt der Spannung der Punkte aller anderen Stationen unterscheiden.

Die Anomalieklassen sind nicht exklusiv, jeder Punkt kann in mehreren Anomalieklassen sein. Dies trifft allerdings nur auf x Punkte von x anomalen Punkten zu

Während die Zeitreihen jeder Messtation ähnliche Verhaltensweisen aufweist, treten die vorhandenen Anomalien je nach Zeitreihe in jeweils unterschiedlicher Stärke und Frequenz auf. Die Daten sind jeweils punktweise gelabeled, es zeichnen sich allerdings punktübergreifende Muster für jede Anomalieklasse ab:

- Aufgrund der Definition von Sprunganomalien, nach welcher jeder Punkt in einer Messung ein bestimmtes Verhalten gegenüber dem Punkt, der jeweilig gleichen Phase der vorherigen Messung haben muss, damit diese Punkte als anomal gekennzeichnet sind, sind in einer Messung entweder alle Punkte eine Sprunganomalie oder keiner von ihnen. Analog dazu sind entweder alle Punkte einer Messung als Phasenanomalie gekennzeichnet oder keine.

- Ähnlich dazu treten Zeitanomalien fast ausschließlich in dreier Paaren von Punkten auf, welche eine Messung einer Station darstellen, da die ihnen zugrundeliegenden Zeitlücken, fast immer Messlücken einer Messstation entsprechen und so für jede Phase einer Station sich zeitliche Lücken bilden.

- Saison-, Phasen- und Stationsanomalien weisen ein stark geklustertes Verhalten auf, wo der Großteil der ihnen zugehörigen Punkte direkt aufeinander folgen, da in diesem Bereich der Spannungsverlauf einer Phase nach dem zugehörigen Anomaliekriterium wesentlich höher oder niedriger als erwartet ist.

5.1.1 Eignung der Daten für überwachte und unüberwachte Anomalieerkennung

Während die Verhaltensformen der Anomalieklassen sich überwachten Lernen anbieten können, wurde sich in dieser Arbeit dennoch für zwei unüberwachte Verfahren entschieden. Wie in Tabelle 5.1 zu sehen ist, sind die Anomalieklassen sehr gering vertreten, wodurch sich eine Knappheit an Daten ergibt, mithilfe derer ein überwachtes Anomalieerkennungsverfahren die jeweiligen Anomalieklassen lernen könnte. Weiterhin entsprechen die Anomalien immer einer in einem bestimmten Kontext ungewöhnlich höheren oder niedrigeren Spannung als gewöhnlich, weshalb ein unüberwachtes Verfahren, diese über Bestimmung der jeweiligen Häufigkeit der eingegebenen Punkte als solche klassifizieren kann.

5.1.2 Benötigte Eigenschaften eines Anomalieerkennungsverfahrens

Es ergeben sich drei weitere Eigenschaften für die die gewählten Anomalieerkennungsverfahren geeignet sein müssen:

1. Aufgrund der oben beschriebenen Tendenz mancher Anomalien in Gruppen zu klustern muss ein auf dem Datensatz angewendetes Anomalieerkennungsverfahren robust gegenüber Duplikaten seien.

2. Da die Daten über die Messstationen live erfasst werden, sollte das Anomalieerkennungsverfahren in der Lage dazu seien, seinen Input als Stream zu empfangen. Weiterhin muss das Verfahren sich an Änderung des durchschnittlichen Spannungswerts anpassen können, möglichst bevor wegen dieser Änderungen eigentliche Inliner als Anomalien klassifiziert werden.

3. Aufgrund der numerischen Klassifizierungskriterien der Anomalien, also dem Fehlen einer klaren Abgrenzung zwischen Inlinern und Anomalien, muss das Verfahren in der Lage sein die Grenze zwischen diesen zu approximieren. Im Falle eines unüberwachten Verfahrens ohne das es sich diesen anlernen kann.

Aufbau der Testsätze

Aufgrund des hohen Umfangs der Testdaten wurden die Tests auf einem Subset der RRCF Daten ausgeführt, diese waren jeweils definiert über:

- Die Station der zu testenden Zeitreihe
- Der Startzeitpunkt des zu testenden Zeitfensters
- Die zu testende Phase
- Die zu testende Anomalieklasse

Die Gesamtheit der Testsätze wurde representativ über die Daten ausgewählt. Dabei wurde ein Fokus auf Testsätze, welche anomale Abschnitte enthalten gelegt. Testsätze, welche nur aus Inlinern bestehen wurden stellenweise hinzugefügt um die Klassifizierung von Inlinern als solche zu überprüfen.

5.2 Testen des RRCF Verfahrens

RRCF als unüberwachtes Anomalieerkennungsverfahren eignet sich zur Analyse des dieser Arbeit zugrunde legendem Datensatzes[4]:

- *Robust gegenüber Duplikaten*: Da der RRCF seine Anomalieeinschätzung in Form des CoDisps gibt, welches per Konzept Robust gegenüber Duplikaten ist, ist das Verfahren in der Lage auch mehrere sich nur schwach unterscheidende anomale Punkte in den Bäumen als solche zu klassifizieren.
- *Anwendbarkeit auf Streaming-Daten*: Neue Datenpunkte können in die konstruierten Bäume eingegliedert werden ohne das diese neu aufgebaut werden müssen.
- *Anpassung an Änderungen im Datensatz*: Da jeder RRCT eine endliche Anzahl an Punkten enthält, muss mit dem Einfügen von neuen Punkten in den RRCT, das Löschen von alten Punkten einhergehen. So kann das was der RRCT als Inliner klassifiziern würde, an das neuer normal angepasst werden
- *Ausgabe in vorm einer Bewertung*: Für das RRCF Verfahren muss ein Grenzwert ermittelt werden, um den Codisp der jedem Punkt zugeordnet wird zu der binären Klassifizierung zwischen Inliner und Anomalie zu transformieren. So kann die tatsächliche Grenze der vorhandenen Anomalielabel ermittelt werden.

Ein weiterer Vorteil des RRCF Verfahrens, die effiziente Handhabung von hochdimensionalen Daten, wird hier nicht benutzt, öffnet aber weitere Alternativen zu der Handhabung des Datensatzes.

5.2.1 Implementierung der Tests

Über die Tests sollen die Parameter für den auf den Testdaten leistungsfähigsten RRCF gefunden werden. Zur Ermittlung der Leistungsfähigkeit wird dabei der MCC verwendet, um eine Balance zwischen den richtigen Klassifizierung von Inlinern und Anomalien zu finden, eine detailliertere Begründung dazu folgt in Sektion **??**

Da in der Praxis der RRCF live auf den von den Messstationen aufgenommen Daten laufen soll, ist es Ziel der Tests diese Situation über Ausschnitte der Daten zu simulieren. Dazu wird wie folgt vorgegangen:

Ablauf der Testläufe

Jeder Testlauf testet, auf einem Testsatz die Leistungsfähigkeit einer Kombination der folgenden drei Parameter:

- **Baumgröße** (ts)**:** Die Anzahl an Punkten die in jeden RRCT des RRCFs passen
- **Baumanzahl** (nt)**:** Die Anzahl der RRCTs in dem konstruierten RRCF
- **Fenstergröße:** Die Größe der Fensterabschnitte, welche die Punkte aus denen die RRCTs gebaut werden ausmachen

Der Testlauf erfolgt in drei Schritten:

Schritt 1: Simulation der Praxis Es wird ein RRCF enstprechend der Parameter des Testlaufes erzeugt, um den in der Praxis bereits vorhanden, über die vorherig gestreamten Daten konstruierten RRCF zu simulieren. Dazu werden aus den letzten ts Punkten (wobei ein Punkt je nach Fenstergröße entweder ein alleinstehender Wert oder eine Reihe von Werten ist) vor dem von dem Testlauf definierten Startpunkt, nt Bäume konstruiert. Es genügt die einzelnen RRCTs nach Definition 4.2.1 zu konstruieren, anstatt die Punkte einzeln in die Bäume einzufügen, da es so nach Theorem 4.2.4 keinen Unterschied in den erwarteten Bäumen gibt.

Schritt 2: Streaming des Testsatzes Der Testsatz wird angefangen mit dem definierten Startpunkt durch jeden RRCT des RRCF gestreamed. Da die Größe jedes RRCTs nach Schritt 1 der definierten Baumgröße entspricht, muss mit jedem eingefügten Punkt ein Punkt aus dem RRCF entfernt werden. In den Testläufen wurde dabei immer der älteste Punkt, also der Punkt welcher am frühesten gemessen wurde gewählt. Mit dem Einfügen von jedem Punkt wird das CoDisp zu diesem vom RRCF berechnet und abgespeichert.

Schritt 3: Auswertung der Ergebnisse Basierend auf den generierten CoDisp Werten wird darauf der bestmögliche MCC über alle möglichen CoDisp-Grenzwerte, für ab wann ein Punkt als Anomalie klassifiziert wird ermittelt. Daraufhin wird für den ermittelten optimalen Grenzwert die Accuracy bestimmt, um eine Vergleichsmetrik mit Testabschnitten zu haben, welche ausschließlich Inliner enthalten, da der MCC, wie in Sektion 2.2.4 beschrieben dort nicht anwendbar ist.

Ergebnisse

Die ersten Testläufe, wurden gebündelt über alle Anomalieklassen ausgeführt:

Abweichungen, mcc vs accuracy, ergebnisse vs anomalie spezifische ergebnisse, reduzierung auf drei nachkommastellen vs volle nachkommastellen

Kapitel 6

Fazit

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

Literaturverzeichnis

[1] Abe, Naoki, Bianca Zadrozny und John Langford: *Outlier detection by active learning.* In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, Seiten 504–509, 2006.

[2] Aggarwal, Charu C: *Outlier analysis.* In: *Data mining.* Springer, 2015.

[3] Ahmed, Mohiuddin, Abdun Naser Mahmood und Jiankun Hu: *A survey of network anomaly detection techniques.* Journal of Network and Computer Applications, 60:19–31, 2016.

[4] Bartos, Matthew, Abhiram Mullapudi und Sara Troutman: *rrcf: Implementation of the Robust Random Cut Forest algorithm for anomaly detection on streams.* Journal of Open Source Software, 4(35):1336, 2019.

[5] Boughorbel, Sabri, Fethi Jarray und Mohammed El-Anbari: *Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric.* PloS one, 12(6):e0177678, 2017.

[6] Chandola, Varun, Arindam Banerjee und Vipin Kumar: *Anomaly detection: A survey.* ACM computing surveys (CSUR), 41(3):1–58, 2009.

[7] Erfani, Sarah M, Sutharshan Rajasegarar, Shanika Karunasekera und Christopher Leckie: *High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning.* Pattern Recognition, 58:121–134, 2016.

[8] Gu, Qiong, Li Zhu und Zhihua Cai: *Evaluation measures of the classification performance of imbalanced data sets.* In: *International symposium on intelligence computation and applications*, Seiten 461–471. Springer, 2009.

[9] Guha, Sudipto, Nina Mishra, Gourav Roy und Okke Schrijvers: *Robust random cut forest based anomaly detection on streams.* In: *International conference on machine learning*, Seiten 2712–2721, 2016.

[10] Guha, Sudipto, Nina Mishra, Gourav Roy und Okke Schrijvers: *Supporting Information for: Robust random cut forest based anomaly detection on streams.* In: *International conference on machine learning*, Seiten 2712–2721, 2016.

[11] Gupta, Manish, Jing Gao, Charu C Aggarwal und Jiawei Han: *Outlier detection for temporal data: A survey.* IEEE Transactions on Knowledge and Data Engineering, 26(9):2250–2267, 2013.

[12] He, Zengyou, Xiaofei Xu und Shengchun Deng: *Discovering cluster-based local outliers.* Pattern Recognition Letters, 24(9-10):1641–1650, 2003.

[13] Liu, Fei Tony, Kai Ming Ting und Zhi-Hua Zhou: *Isolation-based anomaly detection.* ACM Transactions on Knowledge Discovery from Data (TKDD), 6(1):1–39, 2012.

[14] Preiss, Bruno R.: *Data Structures and Algorithms with Object-Oriented Design Patterns in Java.* John Wiley & Sons Incorporated, 2000.

[15] Shi, Tao und Steve Horvath: *Unsupervised learning with random forest predictors.* Journal of Computational and Graphical Statistics, 15(1):118–138, 2006.

[16] Tan, Swee Chuan, Kai Ming Ting und Tony Fei Liu: *Fast anomaly detection for streaming data.* In: *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[17] Tax, David MJ und Robert PW Duin: *Support vector data description.* Machine learning, 54(1):45–66, 2004.

[18] Templ, Matthias, J Gussenbauer und P Filzmoser: *Evaluation of robust outlier detection methods for zero-inflated complex data.* Journal of Applied Statistics, 47(7):1144–1167, 2020.

[19] Williams, Graham, Rohan Baxter, Hongxing He, Simon Hawkins und Lifang Gu: *A comparative study of RNN for outlier detection in data mining.* In: *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, Seiten 709–712. IEEE, 2002.

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet sowie Zitate kenntlich gemacht habe.

Dortmund, den 15. Juli 2020

Joël Haubold