The Model Strikes Back

Warum künstliche Intelligenzen mit der Zeit an Genauigkeit verlieren und wie MLOps damit umgeht

NEXGEN

NEXGEN Whitepaper
Juni 2022

Tristan Poetzsch
Patrick Gschwendtner

Einführung

Das Thema künstliche Intelligenz (KI) und Modellierung ist so langsam in unserem Alltag angekommen. Inzwischen hat jede Finanzinstitution ihren Robo-Advisor und intern gibt es ganze Abteilungen, die sich auf das Thema spezialisieren. Doch so einfach ist das ganze Thema dann doch nicht, denn mit dem klassischen Dreiklang Trainieren, Testen und Bereitstellen hat man gerade einmal den Startschuss für den Betrieb von künstlicher Intelligenz gegeben, wenn man denn überhaupt bis zur Bereitstellung kommt.

Einen der größten Fehler, den Data Scientists und Analytics-Profis im Allgemeinen bei der Entwicklung von künstlichen Intelligenzen machen, ist, dass sie davon ausgehen, dass ihre Modelle nach der Bereitstellung für immer gut funktionieren. Aber was ist mit den Daten, die sich zwangsläufig ständig ändern werden? Ein in der Produktion eingesetztes und nicht regelmäßig überprüftes Modell kann sich nicht selbstständig an Änderungen von Daten anpassen.

Und genau hier setzt der Gedanke des **Continuous Training Cycles** im Bereich **Machine Learning Operations (MLOps)** ein. Mit dem kontinuierlichen Retraining meiner Modelle sorge ich dafür, dass meine Modelle auch morgen noch genauso effektiv und effizient laufen wie zum Zeitpunkt der initialen Erstellung. In diesem Whitepaper beschreiben wir also, warum sich Daten verändern, wie man die Veränderung bemerkt und auch, wie man ein Setup aufbaut, das diesen Änderungen auch automatisch Rechnung trägt.

Warum sollte man Modelle neu trainieren?

Die Welt um uns herum ist stetig im Wandel. Was heute noch aktuell ist, ist schon morgen wieder ganz anders. Und genau das ist der Grund, weswegen ein einmal entwickeltes, trainiertes und für gut empfundenes Model nicht für immer den Ansprüchen genügen kann. Und nicht nur die Welt um uns kann sich ändern, sondern auch unsere Ziele und Erwartungen an die Modelle, die zum Einsatz kommen. Dieses Phänomen bezeichnet man im Allgemeinen als Model Drift. Und die zwei größten Unterkategorien des Model Drifts wollen wir im Folgenden näher erläutern:

- **Data Drift**

  Unter Data Drift subsummiert man die **Änderung von Eigenschaften und Verteilungen in den Input-Daten**. Deutlich wird das anhand eines Beispiels aus der Geldwäsche: Initial lernt unser Modell zum Beispiel, Transaktionen zu erkennen, bei denen viel Geld auf einmal ins Ausland überwiesen wird. Da wir entsprechende Transaktionen nun nachverfolgen können und im Zweifelsfall unterbinden wollen, ändern die Geldwäscher ihr Verhalten daraufhin und überweisen kleinere Beträge auf viele verschiedene inländische Konten und anschließend ins Ausland. Solche Änderungen in den Datenmustern soll nun unser Modell aber auch selbstständig erkennen und darauf reagieren können.

- **Concept Drift**

  Concept Drift beschreibt die **Änderung der Ziele des Modells und der externen Faktoren**. Es kann sein, dass technologische oder gesellschaftliche Änderungen dazu zwingen, unser Verständnis der Welt und damit auch unserer Machine Learning Modelle zu überdenken. Ein Beispiel ist die Vorhersage von Aktienkaufverhalten vor und während der Corona-Pandemie. Vor der Pandemie hielten sich Käufe und Verkäufe in DAX-Unternehmen die Waage. Kurz nach Einbruch der Kurse überstiegen die Käufe von DAX-Aktien die Verkäufe um ca. 60%. Eine Entwicklung, die weder Experten noch Modelle vorhersagen konnten. Aber genau solche Änderungen sind wichtig zu berücksichtigen.

Nicht alle Änderungen lassen sich automatisch auffangen, aber insbesondere im subtileren Bereich des Data Drifts lässt sich mit einem Continuous Training Cycle ein Modell signifikant besser betreiben.

Experten im Interview

José Parra-Moyano
Assistant Professor @ Copenhagen BS

Wo siehst Du die größten Herausforderungen im produktiven Einsatz von Machine Learning?

José: *Ich würde klar sagen, dass Data Privacy die größte Herausforderung ist. Die meisten Modelle brauchen viel mehr Daten, um zu funktionieren als verfügbar sind. Daher braucht es Kollaborationen mit Eignern ähnlicher Daten. Das geht aufgrund von Data Privacy nur selten einfach so. Einen der großen Ansätze zur Lösung sehe ich im Federated Learning. Dort werden Algorithmen von ihrem Eigentümer an andere Dateneigner zurückgegeben, der das Modell anhand seiner privilegierten Daten trainiert und dann nur das fertige Modell zurückgibt. Der Algorithmus-Eigner muss die Daten also selbst nie sehen.“*

Welche Skills sind aktuell das größte Gap für ML-Einheiten?

José: *„Grundsätzlich fehlt es in der Breite an fähigen Ressourcen, die sowohl die Mathematik wie auch die Informatik in genügendem Maße verstehen. Insbesondere fehlt es aber auch an den kreativen Köpfen, die neben den harten Skills auch genügend neue Blickwinkel mitbringen, um den Daten einen Mehrwert abzuringen. Ich nenne diese Rolle den Creative Data Scientist.“*

Was sind die größten Herausforderungen im Model Retraining?

José: *„Veränderungen in den Daten und damit den Modellen passieren nicht immer in derselben Frequenz. Deswegen muss man wach bleiben, sowohl was die Veränderung der Muster in den Daten angeht wie auch was Veränderungen in den Quellen und deren Verfügbarkeit angeht. So hat zum Beispiel das Apple Privacy Update in der ML-Welt viel geändert. Und auch hier ist wieder das Problem, dass es wenige Leute gibt, die ein Gefühl dafür haben.“*

Wann sollte man ein Modell neu trainieren?

Nachdem die Notwendigkeit von Model Retraining klar geworden ist, stellt sich als Nächstes die Frage, wann man ein solches Retraining anstoßen sollte. Dabei haben wir die verschiedenen Ansätze zum Retraining von Modellen in unserem **Model Retraining Trigger Framework** zusammengefasst:

1. Die einfachste Variante ist das Retraining nach festen **Zeitintervallen** (zum Beispiel alle sechs Monate). Das bietet sich zum Beispiel für Modelle an, bei denen in regelmäßigen Abständen neue Daten ankommen. Zum Beispiel: Quartalsabschlüsse, die alle drei Monate kommen.

2. Eine weitere Möglichkeit ist es, basierend auf **Ereignissen** neu zu trainieren. Ein Beispiel für ein ereignisbasiertes Retraining wäre, in der Betrugserkennung nach 10.000 nicht erkannten und nachträglich gemeldeten Betrugsfällen das Modell neu zu trainieren.

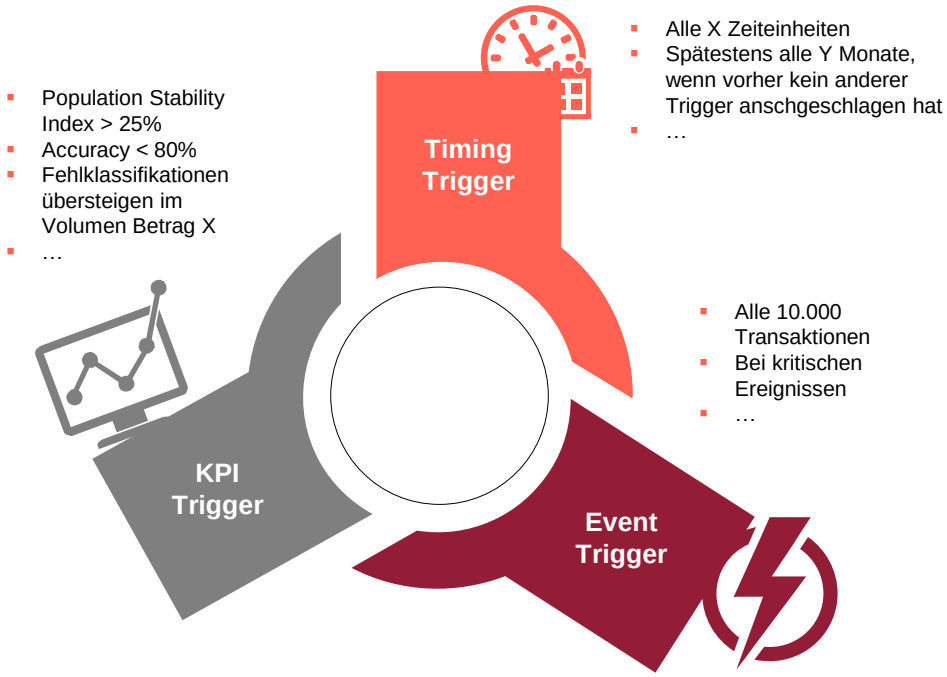
3. Die wahrscheinlich sensitivste Variante ist ein auf **Kennzahlen basiertes Retraining**, bei dem bestimmte Werte wie den Population Stability Index (PSI), Confidence Scores oder andere statistische Kennzahlen im Produktivbetrieb beobachtet und ab einem bestimmten Schwellenwert das Modell neu trainiert wird. KPI-basierte Trigger sind aus unserer Erfahrung häufig das sinnvollste Zielbild, um Data Drifts oder Model Degradation frühzeitig zu erkennen.

Population Stability Index

Der Population Stability Index (PSI) ist ein vor allem in der Finanzindustrie anerkannter Score, der anhand der Abweichung in den Verteilungen der Variablen von den Trainingsdaten zu den Produktivdaten einen möglichen Model Drift erkennt.

Als Daumenregel gilt: Unter 10% ist kein Retraining erforderlich. Zwischen 10% und 25% ist ein Retraining möglich, es sollten aber zumindest zukünftige Abweichungen genauer beobachtet werden. Bei einem **PSI über 25% sollte unbedingt ein neues Model mit aktuellen Daten trainiert werden**. Der PSI kommt häufig zum Einsatz, wenn andere Gütemaße wie die Accuracy nicht zugänglich sind.

Model Retraining Trigger Framework



Doch selbst nach Auswahl der gewünschten Trigger besteht die Herausforderung, sinnvolle Zeiträume und Schwellenwerte zu wählen. Ziel ist immer eine Optimierung, bei der Modelle weder zu früh noch zu spät neu trainiert werden. Wird das Retraining zu früh angestoßen, stehen meist nicht genügend neue Daten bereit, um „die neue Welt“ akkurat abzubilden. Ein zu lange herausgezögertes Retraining lässt die Ungenauigkeit des produktiven Modells zu groß werden. Darüber hinaus können auch mehrere Trigger angelegt werden. Aus Erfahrung empfehlen wir zum Beispiel, unabhängig der anderen Trigger spätestens nach einem Jahr zu prüfen, ob neue Daten das genutzte Model verbessern könnten.

Grundsätzlich raten wir dazu, die **Retraining-Parameter und Prinzipien bereits im Initialen Aufsatz des Machine Learning Modells mitzudenken** und zu entwickeln, um einen sinnvollen Modelleinsatz überhaupt erst zu ermöglichen.

Use Case: Ein Modell zur Erkennung von Kreditkartenbetrug neu trainieren

Kreditkartenbetrug ist eine sehr häufige, vielgestaltige und lästige Betrugsform für Financial Service Provider. Eine manuelle Erkennung von Betrugsfällen ist dabei schon lange nicht mehr ausreichend, aber KI-Modelle können dabei helfen, den Erkennungsprozess zu beschleunigen. Doch egal wie gut die initiale Entwicklung, das Training und das Testen des Modells war, zu einem bestimmten Zeitpunkt wird das Modell aufgrund von Data Drift neu trainiert werden müssen. Das ist zum Beispiel der Fall, wenn Betrüger keine Auslandskonten mehr nutzen oder ihre Volumina ändern. Dabei stellt sich besonders in diesem Fall die Frage, wann konkret man das Modell neu trainieren sollte. Dafür kommen je nach Budget und internen Gegebenheiten verschiedene Möglichkeiten zu Auswahl:

1. Man verwendet ein moving average, die Anzahl der erkannten Betrugsfälle der letzten drei Monate und vergleicht dies mit den ersten drei Monaten. Bei einer Abweichung von mehr als 20% sollte das Model neu trainiert werden. Vorteil ist die einfache Implementierung, Nachteil aber, dass keine neuen Trainingsdaten vorhanden sind und unklar ist, ob das Model wirklich an Genauigkeit verloren hat oder nur die Anzahl der Betrugsfälle zurückgegangen ist.

2. Man erstellt einen neuen Testdatensatz mit aktuellen Daten und bei einer Abweichung im F1-Score von mehr als 5% nach unten sollte das Model neu trainiert werden. Der F1-Score ist ein statistisches Maß für Präzision und Recall-Güte eines Modelles.

3. Man verwendet ein moving average nicht erkannter, aber von Kunden gemeldeten Betrugsfällen der letzten drei Monate und vergleicht dies mit den ersten drei Monaten. Bei einer Abweichung von mehr als 10% sollte das Model neu trainiert werden. Vorteil dieser Lösung ist, dass bereits neue Trainingsdaten durch eine Feedbackfunktion der Nutzer integriert sind.

4. Man vergleicht statistische Werte wie den PSI aus dem Betrieb mit denen der Trainingsdaten. Sollte der PSI über dem Grenzwert liegen (s. Seite 6), ist ein Model Retraining sehr empfehlenswert. Vorteil hier ist die verhältnismäßig frühe Erkennung, verlangt aber auch ein aufwendiges Monitoring.

Experten im Interview

Jannik Podlesny
Data & Analytics Principal

Wird Machine Learning den Erwartungen der Praxis für Financial Services gerecht?

Jannik: *„Financial Services sind traditionell sehr konservativ und hoch reguliert. Auch deswegen habe ich bisher wenige produktionsreife Use Cases neben den Leuchtturmprojekten und PoCs gesehen. Dazu kommt, dass zum Beispiel für Recommender Systemen die Produktpalette gar nicht so breit ist wie zum Beispiel im eCommerce, sodass ausgefeilte Algorithmen gar nicht notwendig sind. Und für andere Fälle wie zum Beispiel Dynamic Pricing gibt es viele regulatorische Hürden.“*

Welche Skills im Bereich Automatisierung und Machine Learning sind am schwersten in der Menge zu gewinnen?

Jannik: *„Wenn man eine Machine Learning Capability aufbaut, braucht man Leute, die das Thema wirklich verstehen – also echtes technisches, mathematisches und geschäftliches Verständnis. Und genau die treffen im Markt auf riesige Nachfrage. Das Problem ist aber, dass die meisten Aufgaben gar nicht anspruchsvolle ML-Cases sind, sondern am Ende des Tages meist nur Standard Business Intelligence sind. Das führt zu enormer Unzufriedenheit bei den hoch qualifizierten Leuten und einer hohen Fluktuation.“*

Wie "Erwachsen" ist MLOps deiner Meinung nach?

Jannik: *„MLOps ist im Vergleich zu DevOps noch ein Baby. Das merkt man zum Beispiel daran, dass die ausgeschriebenen Stellen häufig noch fünf Rollen in einer völlig unsinnigen Kombination verbinden. Gepaart mit dem fehlenden Grundverständnis sowohl für agile Entwicklungsprinzipien wie auch für DevOps Best Practices ist hier also noch einiges an Weg zurückzulegen, bis MLOps wirklich tiefgründig verstand und wertgeschätzt wird.“*

Wie implementiert man Retraining für Modelle?

Nachdem bestimmt wurde, auf Basis welcher Parameter ein Retraining angestoßen werden soll, steht die konkrete Implementierung aus. Die Herausforderung dabei ist, dass ein Retraining kleinschrittig und mit viel Aufwand verbunden ist. Die Komplexität rührt daher, dass im Retraining viele Faktoren und Modellparameter gemonitort und berücksichtigt werden müssen. Ein gutes Beispiel dafür ist die Reproduzierbarkeit von Modellergebnissen. Ein Datensatz sollte von ein und demselben Modell immer gleich klassifiziert werden – zum Beispiel sollte eine Transaktion immer als potentieller Betrug oder eben nicht klassifiziert werden. Daher ist auch das Testen eines ML-Systems aufwendiger als das Testen anderer Softwaresysteme. So ist zusätzlich zu den typischen Einheits- und Integrationstests auch die Datenvalidierung, Qualitätsbewertung trainierter Modelle wie auch eine Modellvalidierung erforderlich.

Aufgrund dieser Komplexität starten die meisten Projekte mit einem manuellen Retraining-Prozess, bei dem jeder Schritt einschließlich Datenanalyse und Vorbereitung einzeln durchgeführt wird. Trainierte Modelle werden als Artefakte an Operations gegeben, welche diese Modelle dann produktiv bereitstellen. Durch diesen hohen manuellen Aufwand werden Modelle meist weniger häufig neu trainiert, was zu dem genannten Problem führt, dass Modelle nicht mehr den Anforderungen entsprechen.

Ziel ist es also, den Trainingsprozess zu automatisieren und immer dann anzustoßen, wenn unsere vorher definierten Trigger anschlagen. Diese Automatisierung wird im Allgemeinen unter dem Begriff **Continuous Retraining** zusammengefasst.

Continuous Training (CT) Pipeline vs. CI/CD

Beim Continuous Training und dessen Pipelines geht es um das automatische, wiederholte Trainieren und Bereitstellen von Modellen. Der Unterschied zu einer CI/CD-Pipeline besteht darin, dass nicht neuer Code der Trigger für den Start der Pipeline ist, sondern neue Daten. Das bedeutet auch, dass die Konzepte unabhängig voneinander agieren können. Eine CI/CD Pipeline kann aber zum Beispiel eine neue Implementierung einer CT-Pipeline bereitstellen oder neue Artefakte für selbige deployen.

Für ein vollautomatisches Continuous Training Setup sind mehrere Komponenten nötig. Im Folgenden beschreiben wir einen solchen Aufbau beispielhaft im Google Cloud Plattform Umfeld.

1. Zuerst müssen die **Trigger automatisiert** werden. So kann zum Beispiel ein Cloud Monitoring den PSI kontinuierlich prüfen und ein Event auslösen, wenn der definierte Schwellwert überschritten ist. Auch Vertex AI kann selbst Trigger auslösen, zum Beispiel wenn die Accuracy des Modells zu niedrig wird, hier muss jedoch je nach Implementierungsdetails noch mal eine Hilfsfunktion z. B. als Kombination von Pub/Sub und Cloud Function aufgerufen werden.

2. Wenn einer der Wege den Trigger für das Retraining absetzt, soll nun **automatisch ein neues Modell berechnet** werden. Dafür eignet sich eine selbst gebaute Kubeflow Pipeline hervorragend. Dabei ist es unterschiedlich, ob und wie man tatsächlich auch eine Validierung durchführen kann, bevor das neue Modell eingesetzt wird.

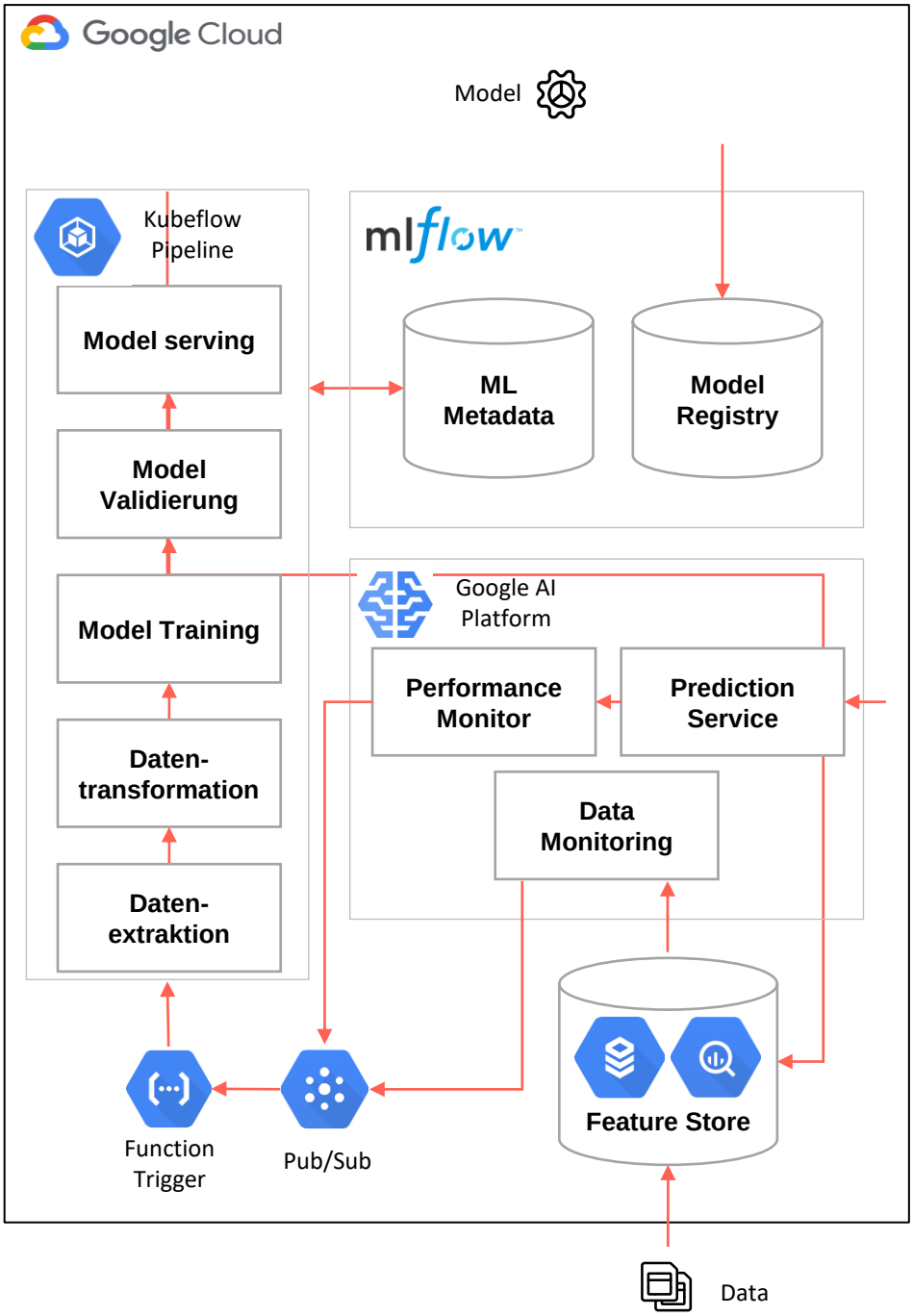
3. Das Retraining erfordert zusätzlich einen **Metadata Store** für die Parameter und Grenzwerte. Hier werden zum Beispiel die Akzeptanzkriterien für neue Modelle und Ergebnisse der Modell Evaluation festgehalten.

4. Auch aus regulatorischen Gründen ist es zwingend notwendig, die einzelnen Modellversionen in einem **Model Repository** bereitzuhalten. Von hier aus ist immer sicher dokumentiert, welche Modelle wann entwickelt wurden und produktiv waren.

Für den **Metadata Store** und das **Model Repository** ist die Empfehlung eine SaaS Lösung wie mlfow zu verwenden. Diese verbindet beide Komponenten und erlauben es so sich mehr auf das Model und die Pipeline selbst zu konzentrieren.

Grundsätzlich ist noch wichtig anzubringen, dass ein ausgeprägtes Maß an Vertrautheit mit DevOps Prinzipien notwendig ist und die Organisation diese Prinzipien auch bereits aktiv verfolgen sollte. Anderenfalls ist die Umsetzung durchaus schwierig.

Referenzarchitektur CT-Pipeline auf GCP



Interessiert an mehr?

Möchten Sie den erfolgreichen Einsatz von Machine Learning Modellen in ihrem Unternehmen fördern?

Suchen Sie nach Referenzarchitekturen und Blueprints für den Aufsatz ihrer MLOps Pipeline und den zugehörigen Prozessen?

Oder interessiert Sie vielleicht der Einsatz von Natural Language Processing?

Wir freuen uns darauf, mit Ihnen Ihr individuelles Vorhaben zu besprechen und mit unserer Erfahrung zu unterstützen.

Ihre Ansprechpartner:

Tristan Poetzsch

Manager

tristan.poetzsch@nexgenbc.com

Patrick Gschwendtner

Consultant

patrick.gschwendtner@nexgenbc.com

Übrigens – wir sind Google Cloud Partner!

Google Cloud Partner

NEXGEN

NEXGEN Business Consultants GmbH
Grüneburgweg 101
60323 Frankfurt am Main