KAPITEL 1

Data Science mit AWS – eine Einführung

In diesem Kapitel erörtern wir die Vorteile der Erstellung von Data-Science-Projekten in der Cloud. Zunächst werden die Vorzüge von Cloud Computing dargelegt. Anschließend beschreiben wir einen typischen Arbeitsablauf beim Machine Learning und die allgemeinen Herausforderungen bei der Überführung unserer Modelle und Anwendungen vom Prototyp in die Produktion. Wir gehen auf die allgemeinen Vorteile der Entwicklung von Data-Science-Projekten mit *Amazon Web Services* (AWS) ein und stellen die relevanten AWS-Services für jeden Schritt des Modellentwicklungsworkflows vor. Darüber hinaus zeigen wir bewährte Ansätze zur Gestaltung der Architektur auf, insbesondere bezogen auf Operational Excellence, Sicherheit, Zuverlässigkeit und Leistungs- sowie Kostenoptimierung.

Vorzüge des Cloud Computing

Cloud Computing ermöglicht eine bedarfsgerechte Bereitstellung von IT-Ressourcen über das Internet, wobei sich die Kosten nach dem jeweiligen Bedarf bemessen. Anstatt eigene Rechenzentren und Server zu kaufen, sie zu betreiben und zu warten, können wir Technologien wie Rechenleistung, Speicherplatz, Datenbanken und andere Dienste je nach Bedarf erwerben. Ähnlich wie ein Stromversorger sofort Strom liefert, wenn wir einen Lichtschalter in unserem Haus betätigen, stellt die Cloud IT-Ressourcen bei Bedarf per Mausklick oder durch den Aufruf einer API bereit.

»Es gibt keinen Algorithmus, der Erfahrung kompensiert«, lautet ein berühmtes Zitat von Andy Jassy, ehemals CEO von Amazon Web Services und inzwischen CEO von Amazon. Das Zitat drückt die langjährige Erfahrung des Unternehmens beim Aufbau zuverlässiger, sicherer und leistungsstarker Dienste seit dem Jahr 2006 aus.

AWS hat sein Leistungsportfolio fortwährend erweitert, um praktisch jede Art von Arbeit in der Cloud zu unterstützen, einschließlich zahlreicher Dienste und Funktionen auf dem Gebiet der künstlichen Intelligenz und des maschinellen Lernens. Viele dieser KI- und Machine-Learning-Dienste gehen auf Amazons Pionierarbeit in den Bereichen Empfehlungssysteme, Computer Vision, Sprach- bzw. Textverarbei-

tung und neuronale Netze in den letzten 20 Jahren zurück. Ein Forschungsbeitrag aus dem Jahr 2003 mit dem Titel »Amazon.com Recommendations: Item-to-Item Collaborative Filtering« (*https://oreil.ly/UlCDV*) wurde kürzlich vom Institute of Electrical and Electronics Engineers als ein Beitrag ausgezeichnet, der den »Test der Zeit« überstanden hat. Lassen Sie uns die Vorzüge von Cloud Computing im Zusammenhang mit Data-Science-Projekten in der AWS-Cloud betrachten.

Agilität

Cloud Computing ermöglicht uns, Ressourcen bedarfsgerecht bereitzustellen, was uns wiederum die Möglichkeit bietet, zügig und häufig Experimente durchzuführen. Vielleicht möchten wir eine neue Bibliothek testen, um die Qualität unseres Datensatzes zu überprüfen, oder ein Modell mithilfe der neuesten Generation von Grafikprozessoren (GPUs) schneller trainieren. Innerhalb von Minuten können wir Dutzende, Hunderte oder sogar Tausende von Servern in Betrieb nehmen, die derartige Aufgaben für uns erledigen. Führt ein Experiment nicht zum gewünschten Erfolg, können wir die entsprechenden Ressourcen jederzeit und ohne jegliches Risiko wieder abstellen.

Kosten einsparen

Cloud Computing ermöglicht uns, auf Investitionen mit hohem Kapitaleinsatz zu verzichten und dafür Kosten zu tragen, die variabler Natur sind: Wir zahlen nur für das, was wir tatsächlich in Anspruch nehmen, und müssen keine Vorabinvestitionen in Hardware tätigen, die in ein paar Monaten veraltet sein könnte. Wenn wir Rechenressourcen im Rahmen unserer Datenqualitätsprüfungen und -transformationen oder unserem Modelltraining einsetzen, zahlen wir ausschließlich für die Zeit, in der diese Rechenressourcen genutzt werden. Wir können weitere Kosteneinsparungen erzielen, indem wir Amazon-EC2-Spot-Instanzen für unser Modelltraining nutzen. Mit Spot-Instanzen können wir ungenutzte EC2-Kapazitäten in der AWS-Cloud mit einem Preisnachlass von bis zu 90 % im Vergleich zu On-Demand-Instanzen verwenden. Mit reservierten Instanzen (*Reserved Instances*) und Sparplänen (*Savings Plans*) können wir Geld sparen, indem wir für einen bestimmten Zeitraum im Voraus bezahlen.

Elastizität

Cloud Computing ermöglicht uns, unsere Ressourcen automatisch herauf- oder herunterzuskalieren (engl. *Scaling-out* bzw. *Scaling-in*), um sie an die Bedürfnisse unserer Anwendung anzupassen. Nehmen wir an, wir hätten unsere Data-Science-Anwendung in die Produktion überführt und unser Modell diente der Echtzeitvorhersage. Sollten wir feststellen, dass es zu einem Anstieg der Modellanfragen kommt, können wir die Ressourcen, die das Modell hosten, automatisch erhöhen. Ebenso können wir die Ressourcen automatisch reduzieren, wenn die

Anzahl der Modellanfragen sinkt. Es ist nicht mehr notwendig, unnötig viele Ressourcen bereitzustellen, um Lastspitzen zu bewältigen.

Schneller innovieren

Cloud Computing ermöglicht uns, Innovationen schneller einzuführen, da wir uns auf die Entwicklung von Anwendungen konzentrieren können, die dazu verhelfen, unser Unternehmen von anderen abzuheben, anstatt uns mit der aufwendigen Verwaltung der Infrastruktur zu beschäftigen. Die Cloud hilft uns, mit neuen Algorithmen, Frameworks und Hardware in Sekunden statt Monaten Experimente anzustellen.

Globales Deployment in Minutenschnelle

Cloud Computing ermöglicht uns, unsere Data-Science-Anwendungen innerhalb von Minuten weltweit zur Verfügung zu stellen. In der heute global vernetzten Wirtschaft ist es wichtig, nahe bei unseren Kunden zu sein. AWS hat das Konzept von Regionen etabliert, die physischen Standorten auf der ganzen Welt entsprechen und AWS-Rechenzentren zu Clustern zusammenfasst. Jede Gruppe von logischen Rechenzentren wird als *Availability Zone* (AZ) bezeichnet. Jede AWS-Region besteht aus mehreren isolierten und physisch getrennten AZs innerhalb eines geografischen Gebiets. Die Anzahl der verfügbaren AWS-Regionen und AZs wächst stetig (*https://oreil.ly/qegDk*).

Wir können die weltweite Verfügbarkeit der AWS-Regionen und AZs dazu nutzen, unsere Data-Science-Anwendungen in der Nähe unserer Kunden bereitzustellen, die Anwendungsleistung mit extrem schnellen Reaktionszeiten zu verbessern und die Datenschutzbestimmungen der einzelnen Regionen einzuhalten.

Reibungsloser Übergang vom Prototyp zur Produktion

Einer der Vorteile der Entwicklung von Data-Science-Projekten in der Cloud ist der reibungslose Übergang vom Prototyp zur Produktion. Wir können innerhalb von Minuten von der Programmierung eines Modellprototyps in unserem Notebook zur Überprüfung der Datenqualität oder zum verteilten Modelltraining mit Petabytes an Daten übergehen. Im Anschluss daran können wir unsere trainierten Modelle für Echtzeit- oder Batch-Vorhersagen für Millionen von Nutzerinnen und Nutzern auf der ganzen Welt einsetzen.

Die Erstellung von Prototypen erfolgt häufig in Entwicklungsumgebungen mit nur einem Rechner unter Verwendung von Jupyter Notebook, NumPy und Pandas. Dieser Ansatz funktioniert gut für kleine Datensätze. Bei der Arbeit mit großen Datensätzen werden wir schnell die CPU- und RAM-Ressourcen eines einzelnen Rechners überschreiten. Außerdem möchten wir vielleicht GPUs – oder mehrere Rechner – verwenden, um unser Modelltraining zu beschleunigen. Dies ist mit einer einzelnen Maschine bzw. einem einzelnen Rechner in der Regel nicht möglich.

Die nächste Herausforderung wartet auf uns, wenn wir unser Modell (bzw. unsere Anwendung) in der Produktionsumgebung zum Einsatz bringen bzw. deployen möchten. Zudem müssen wir sicherstellen, dass unsere Anwendung Tausende oder Millionen gleichzeitiger Benutzer global bedienen kann.

Das Deployment in die Produktionsumgebung erfordert oft eine enge Zusammenarbeit zwischen verschiedenen Teams, wie etwa Data Science, Data Engineering, Anwendungsentwicklung und DevOps. Und sobald unsere Anwendung erfolgreich bereitgestellt wurde, müssen wir die Modellleistung und die Datenqualität kontinuierlich überwachen und auf Probleme reagieren, die nach der Überführung des Modells in die Produktion auftreten können.

Die Entwicklung von Data-Science-Projekten in der Cloud ermöglicht uns, unsere Modelle reibungslos – vom Prototyp ausgehend – in die Produktion zu überführen, ohne dass wir eine eigene physische Infrastruktur aufbauen müssen. Verwaltete Cloud-Dienste geben uns die Werkzeuge an die Hand, um unsere Arbeitsabläufe zu automatisieren und Modelle in einer skalierbaren und äußerst leistungsfähigen Produktionsumgebung bereitzustellen.

Data-Science-Pipelines und -Workflows

Data-Science-Pipelines und -Workflows umfassen viele komplexe, multidisziplinäre und iterative Schritte. Nehmen wir als Beispiel einen typischen Workflow im Rahmen der Entwicklung eines Machine-Learning-Modells. Wir beginnen mit der Datenaufbereitung und gehen dann zum Training und zum Feintuning eines Modells über. Schließlich stellen wir unser Modell (bzw. unsere Anwendung) in einer Produktionsumgebung bereit. Jeder dieser Schritte besteht wiederum aus mehreren Teilschritten, wie in Abbildung 1-1 dargestellt.

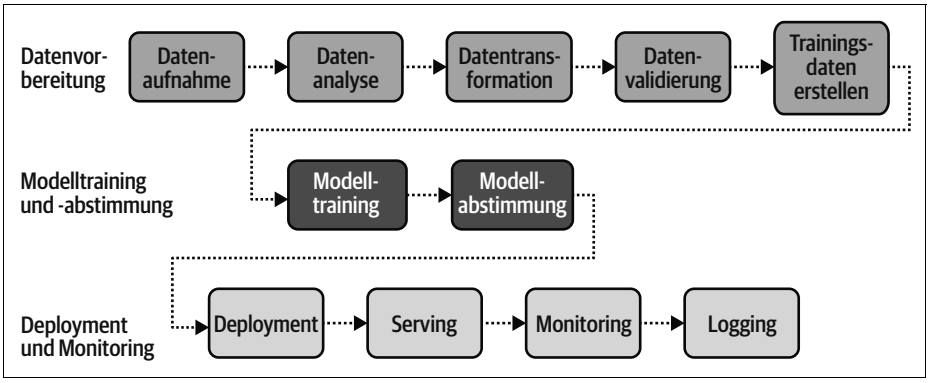

Abbildung 1-1: Ein typischer Machine-Learning-Workflow umfasst viele komplexe, multidisziplinäre und iterative Schritte.

Wenn wir AWS verwenden, befinden sich unsere Rohdaten wahrscheinlich bereits im *Amazon Simple Storage Service* (Amazon S3) und sind als CSV-Datei, als Apache Parquet oder in einem ähnlichen Format gespeichert. Mit den Amazon-KI- oder

-AutoML-Diensten können wir in kürzester Zeit damit beginnen, Modelle zu trainieren, um eine Baseline für die Modellleistung zu erhalten, indem wir direkt auf unseren Datensatz verweisen und auf eine einzige Schaltfläche *train* klicken. Die KI-Dienste und AutoML werden in den Kapiteln 2 und 3 ausführlich behandelt.

Für kundenspezifisch gestaltete Machine-Learning-Modelle – den Hauptschwerpunkt dieses Buchs – können wir mit der manuellen Datenaufnahme und der explorativen Analyse beginnen, einschließlich Datenanalyse, Überprüfung der Datenqualität, zusammenfassender Statistiken, fehlender Werte, Quantilsberechnungen, Analyse der Schiefe der Daten, Korrelationsanalyse usw. Die Kapitel 4 und 5 behandeln ausführlich die Datenaufnahme und die explorative Datenanalyse.

Dann sollten wir die Art der maschinellen Lernaufgabe definieren – Regression, Klassifikation, Clustering usw. Sobald wir die Art des zu lösenden Problems bestimmt haben, können wir einen Machine-Learning-Algorithmus auswählen, der sich am besten für die Lösung des jeweiligen Problems eignet. Je nachdem, welchen Algorithmus wir wählen, müssen wir eine Teilmenge unserer Daten auswählen, um unser Modell zu trainieren, zu validieren und zu testen. Unsere Rohdaten müssen in der Regel in mathematische Vektoren umgewandelt werden, um die numerische Optimierung und das Modelltraining zu ermöglichen. Wir könnten uns zum Beispiel dafür entscheiden, kategoriale Datenspalten in One-Hot-codierte Vektoren umzuwandeln oder textbasierte Datenspalten in Worteinbettungsvektoren, sogenannte Word Embeddings, umzuwandeln. Nachdem wir einen Teil der Rohdaten in Features umgewandelt haben, sollten wir die Features bzw. Daten in Trainings-, Validierungs- und Testdatensätze aufteilen, um sie für das Modelltraining, das Feintuning und das Testen vorzubereiten. Die Auswahl und Transformation von Features – die auch als Merkmale bezeichnet werden – wird in den Kapiteln 5 und 6 detaillierter behandelt.

In der Phase des Modelltrainings wählen wir einen Algorithmus aus und trainieren unser Modell mit unserem Trainingsdatensatz, um zu prüfen, ob unser Programmcode und unser Algorithmus geeignet sind, das gegebene Problem zu lösen. In Kapitel 7 werden wir uns ausgiebig mit dem Modelltraining beschäftigen.

In der Phase der Modellabstimmung bzw. -optimierung stimmen wir die Hyperparameter des Algorithmus ab und bewerten die Leistung des Modells anhand des Validierungsdatensatzes. Wir wiederholen diese Schritte, fügen weitere Daten hinzu oder ändern je nach Bedarf die Hyperparameter, bis das Modell die erwarteten Ergebnisse auf dem Testdatensatz erzielt. Bevor wir das Modell in die Produktion überführen, sollten wir sicherstellen, dass sich diese Ergebnisse mit unserem Geschäftsziel decken. In Kapitel 8 werden wir uns ausführlich mit der Abstimmung von Hyperparametern beschäftigen.

Die letzte Phase – die Überführung von Prototypen in die Produktion – stellt für Data Scientists und Machine-Learning-Experten oft die größte Herausforderung dar. In Kapitel 9 nehmen wir genauer unter die Lupe, wie wir Modelle deployen können.

In Kapitel 10 führen wir alles zu einer automatisierten Pipeline zusammen. In Kapitel 11 widmen wir uns Streaming-Daten und zeigen, wie sich diese analysieren lassen und wie sie in Machine-Learning-Modellen verarbeitet werden können. Kapitel 12 beschreibt die besten Praktiken zur Sicherung von Data-Science-Projekten in der Cloud.

Sobald wir jeden einzelnen Schritt unseres Machine-Learning-Workflows aufgebaut haben, können wir damit beginnen, die Schritte in einer einzelnen, wiederverwendbaren Machine-Learning-Pipeline zu automatisieren. Wenn neue Daten in S3 abgelegt werden, wird unsere Pipeline mit den neuesten Daten neu gestartet, und das neueste Modell wird in die Produktion geschickt, um unsere Anwendungen zu bedienen. Es gibt zahlreiche Tools zur Workflow-Orchestrierung und AWS-Dienste, die uns beim Aufbau automatisierter Pipelines für maschinelle Lernmodelle helfen.

Amazon SageMaker Pipelines

Amazon SageMaker Pipelines bieten die standardmäßige, voll funktionsfähige und vollständigste Methode zur Implementierung von KI- und Machine-Learning-Pipelines in Amazon SageMaker. SageMaker Pipelines sind in SageMaker Feature Store, SageMaker Data Wrangler, SageMaker Processing Jobs, SageMaker Training Jobs, SageMaker Hyperparameter Tuning Jobs, SageMaker Model Registry, SageMaker Batch Transform und SageMaker Model Endpoints integrierbar, die wir im Laufe des Buchs besprechen werden. In Kapitel 10 befassen wir uns eingehend mit verwalteten SageMaker Pipelines und besprechen, wie Sie Pipelines mit AWS Step Functions, Kubeflow Pipelines, Apache Airflow, MLflow, TFX sowie Human-in-the-Loop-Workflows erstellen können.

AWS Step Functions Data Science SDK

Step Functions, ein verwalteter AWS-Service, ist eine großartige Möglichkeit, komplexe Workflows zu erstellen, ohne dass wir eine eigene Infrastruktur aufbauen und warten müssen. Wir können das Step Functions Data Science SDK verwenden, um Machine-Learning-Pipelines aus Python-Umgebungen wie Jupyter Notebook zu erstellen. Step Functions in Bezug auf maschinelles Lernen werden wir uns in Kapitel 10 genauer ansehen.

Kubeflow Pipelines

Kubeflow ist ein relativ neues Ökosystem, das auf Kubernetes aufbaut und ein Orchestrierungssubsystem namens *Kubeflow Pipeline*s enthält. Mit Kubeflow können wir fehlgeschlagene Pipelines neu starten, die Ausführung von Pipelines planen, Trainingsmetriken analysieren und den Verlauf der Entwicklung von Pipelines nachverfolgen bzw. tracken. In Kapitel 10 werden wir uns näher mit der Verwal-

tung eines Kubeflow-Clusters auf *Amazon Elastic Kubernetes Service* (Amazon EKS) befassen.

Managed Workflows for Apache Airflow in AWS

Apache Airflow ist eine sehr ausgereifte und beliebte Lösung, die in erster Linie für die Orchestrierung von Data-Engineering- und ETL-Pipelines (Extract – Transform – Load) entwickelt wurde. Wir können Airflow verwenden, um Workflows als gerichtete azyklische Graphen von Aufgaben zu erstellen. Der Airflow-Scheduler führt unsere Aufgaben auf einem Verbund von Workern aus und folgt dabei den angegebenen Abhängigkeiten. Über die Benutzeroberfläche von Airflow können wir die in der Produktion befindlichen Pipelines visualisieren, den Fortschritt überwachen und bei Bedarf Probleme beheben. In Kapitel 10 werden wir *Amazon Managed Workflows for Apache Airflow* (Amazon MWAA) näher beleuchten.

MLflow

MLflow ist ein Open-Source-Projekt, das sich ursprünglich auf die Rückverfolgbarkeit (Tracking) von Experimenten konzentrierte, jetzt aber auch Pipelines namens *MLflow Workflows* unterstützt. Wir können MLflow auch dazu verwenden, Experimente mit Kubeflow- und Apache-Airflow-Workflows zu tracken. MLflow erfordert jedoch, dass wir unsere eigenen Amazon-EC2- oder Amazon-EKS-Cluster aufbauen und warten. Wir werden MLflow noch ausführlicher in Kapitel 10 vorstellen.

TensorFlow Extended

TensorFlow Extended (TFX) ist eine Open-Source-Sammlung von Python-Bibliotheken, die in einem Pipeline-Orchestrator wie AWS Step Functions, Kubeflow Pipelines, Apache Airflow oder MLflow verwendet werden. TFX ist spezifisch für TensorFlow und hängt von einem anderen Open-Source-Projekt, Apache Beam, ab, um über einen einzelnen Verarbeitungsknoten hinaus zu skalieren. In Kapitel 10 werden wir TFX ausführlicher besprechen.

Human-in-the-Loop – Workflows, die den Menschen einbeziehen

Während die auf KI und Machine Learning basierenden Dienste unser Leben einfacher machen, ist der Mensch noch lange nicht überflüssig. Tatsächlich hat sich das Konzept des *Human-in-the-Loop* zu einem wichtigen Eckpfeiler in vielen KI- bzw. ML-Workflows entwickelt. Die Einbindung des Menschen trägt wesentlich zur Qualitätssicherung von sensiblen und regulierten Modellen in der Produktion bei.

Amazon Augmented AI (Amazon A2I) ist ein vollständig verwalteter Service zur Entwicklung von Human-in-the-Loop-Workflows, die eine übersichtliche Benutzeroberfläche, eine rollenbasierte Zugriffskontrolle mit *AWS Identity and Access*

Management (IAM) und eine skalierbare Datenspeicherung unter Verwendung von S3 umfassen. Amazon A2I ist in zahlreiche Amazon-Services integriert, darunter Amazon Rekognition für die Handhabung von Medieninhalten (Content-Moderation) und Amazon Textract für die Extraktion von Formulardaten. Des Weiteren können wir Amazon A2I mit Amazon SageMaker und jedem unserer benutzerdefinierten ML-Modelle verwenden. Wir werden uns in Kapitel 10 eingehender mit Human-in-the-Loop-Workflows befassen.

Best Practices für MLOps

Der Bereich *Machine Learning Operations* (MLOps) hat sich in den letzten zehn Jahren herausgebildet, um den einzigartigen Herausforderungen beim Betrieb von KI- und ML-basierten Systemen zu begegnen, die aus der Kombination von Software und Daten resultieren. Mithilfe von MLOps entwickeln wir die End-to-End-Architektur für ein automatisiertes Modelltraining, das Hosten von Modellen und die Überwachung der Pipeline. Indem wir von Beginn an eine vollständige MLOps-Strategie verfolgen, bauen wir Fachwissen auf, reduzieren die Wahrscheinlichkeit menschlicher Fehler, verringern das Risiko unseres Projekts und gewinnen Zeit, um uns auf die eigentlichen Herausforderungen der Data Science zu konzentrieren.

MLOps hat drei verschiedene Entwicklungsphasen durchlaufen:

MLOps 1.0
: Modelle manuell entwickeln, trainieren, optimieren und deployen.

MLOps 2.0
: Modell-Pipelines manuell erstellen und orchestrieren.

MLOps 3.0
: Pipelines werden automatisch ausgeführt, wenn neue Daten eintreffen oder der Code durch deterministische Auslöser (Trigger) wie GitOps verändert wird oder wenn die Leistung der Modelle aufgrund statistischer Auslöser wie Drift, Bias (Verzerrung) und Abweichungen in der Erklärbarkeit nachlässt.

AWS und Amazon SageMaker Pipelines unterstützen die komplette MLOps-Strategie, einschließlich des automatischen Pipeline-Retrainings mit sowohl vorab bestimmten bzw. deterministischen GitOps-Triggern als auch statistisch basierten Triggern infolge einer Drift der Daten, eines Bias des Modells oder Veränderungen in der Erklärbarkeit. In den Kapiteln 5, 6, 7 und 9 werden wir uns eingehend mit statistischer Drift, statistischem Bias (bzw. statistischer Verzerrung) und Erklärbarkeit (*Explainability*) beschäftigen. Außerdem implementieren wir kontinuierliche (*Continuous*) und automatisierte Pipelines in Kapitel 10 mit verschiedenen Pipeline-Orchestrierungs- und Automatisierungsoptionen, darunter SageMaker Pipelines, AWS Step Functions, Apache Airflow, Kubeflow und andere Optionen, einschließlich Human-in-the-Loop-Workflows. Lassen Sie uns nun einige Best Practices für Operational Excellence, Sicherheit, Zuverlässigkeit, Leistungseffizienz und Kostenoptimierung von MLOps besprechen.

Operational Excellence

Im Folgenden finden Sie einige Best Practices in Bezug auf Machine Learning, die uns helfen, Data-Science-Projekte erfolgreich in der Cloud aufzubauen:

Datenqualitätsprüfungen.
: Da alle unsere ML-Projekte mit Daten beginnen, sollten Sie sicherstellen, dass Sie Zugang zu hochwertigen Datensätzen haben und wiederholt Prüfungen der Datenqualität durchführen können. Eine unzureichende Datenqualität führt häufig dazu, dass Projekte scheitern. Behalten Sie diese Probleme schon früh in Ihrer Pipeline im Auge.

Fangen Sie einfach an und verwenden Sie bestehende Lösungen wieder.
: Beginnen Sie mit der einfachsten Lösung, denn es gibt keinen Grund, das Rad neu zu erfinden, wenn es nicht zwingend erforderlich ist. Wahrscheinlich gibt es bereits einen KI-Dienst, der unsere Aufgabe bewältigen kann. Greifen Sie auf verwaltete Dienste wie Amazon SageMaker zurück, die über eine Vielzahl integrierter Algorithmen und vortrainierter Modelle verfügen.

Legen Sie Gütemaße bzw. Leistungsmetriken für das Modell fest.
: Ordnen Sie die Leistungsmetriken des Modells den Geschäftszielen zu und überwachen Sie diese Maße kontinuierlich. Wir sollten eine Strategie entwickeln, um bei nachlassender Leistung Modelle für unzureichend zu erklären und neu zu trainieren.

Tracken und versionieren Sie alles.
: Tracken Sie die Modellentwicklung mithilfe von Experimenten und dokumentieren Sie den vollständigen Verlauf (*Lineage*) in nachvollziehbarer Weise. Sie sollten ebenfalls die Datensätze, den Code für die Feature Transformation, die Hyperparameter und die trainierten Modelle versionieren.

Wählen Sie eine geeignete Hardware für das Modelltraining und den Betrieb des Modells aus.
: In vielen Fällen stellt das Training des Modells andere Anforderungen an die Infrastruktur als der Betrieb des Modells zur Erstellung von Vorhersagen. Wählen Sie für die einzelnen Phasen die entsprechenden Ressourcen aus.

Überwachen Sie die im Einsatz befindlichen Modelle fortlaufend.
: Erkennen Sie eine Drift in den Daten oder im Modell und ergreifen Sie geeignete Maßnahmen, wie z. B. das Modell neu zu trainieren (Retraining).

Automatisieren Sie ML-Workflows.
: Bauen Sie konsistente, automatisierte Pipelines auf, um menschliche Fehler zu reduzieren und Zeit für die eigentlichen Kernaufgaben zu gewinnen. Die Pipelines können Schritte umfassen, in denen die Modelle von Menschen genehmigt werden müssen, bevor sie in die Produktion überführt werden.

Sicherheit

Die Verantwortung für Sicherheit und Compliance liegt sowohl bei AWS als auch beim Kunden. AWS sorgt für die Sicherheit »der« Cloud, während der Kunde für die Sicherheit »in der« Cloud verantwortlich ist.

Die häufigsten Sicherheitsüberlegungen für den Aufbau sicherer Data-Science-Projekte in der Cloud betreffen die Bereiche Zugriffsverwaltung, Isolierung von Rechnern und Netzwerken, Verschlüsselung, Governance und Auditierbarkeit.

Wir benötigen umfassende Sicherheits- und Zugriffskontrollfunktionen für unsere Daten. Dementsprechend gilt es, den Zugriff auf Aufträge bzw. Jobs wie das Labeln von Daten, auf Skripte, die der Datenverarbeitung dienen, auf Modelle, Endpoints für die Inferenz oder auch auf Jobs für Batch-Vorhersagen zu beschränken.

Außerdem sollten wir eine Daten-Governance-Strategie verfolgen, die die Integrität, Sicherheit und Verfügbarkeit unserer Datensätze gewährleistet. Implementieren und erzwingen Sie eine Datenabfolge, die die auf unsere Trainingsdaten angewandten Datentransformationen überwacht und trackt. Stellen Sie sicher, dass die Daten im Ruhezustand und bei der Übertragung verschlüsselt sind. Des Weiteren sollten Sie bei Bedarf die Einhaltung gesetzlicher Vorschriften gewährleisten.

In Kapitel 12 werden wir einige Best Practices für den Aufbau sicherer Data-Science- und Machine-Learning-Anwendungen auf AWS noch genauer erörtern.

Reliabilität

Der Begriff *Reliabilität* bzw. Zuverlässigkeit eines Systems beschreibt seine Fähigkeit, Störungen bzw. Unterbrechungen der Infrastruktur oder des Diensts auszugleichen, dynamisch Rechenressourcen zu beziehen, um der Nachfrage gerecht zu werden, und Störungen wie Fehlkonfigurationen oder vorübergehende Netzwerkprobleme abzufedern.

Wir sollten das Tracking von Änderungen und die Versionskontrolle für unsere Trainingsdaten automatisieren. Auf diese Weise können wir im Fall eines Fehlers exakt die gleiche Version eines Modells neu erstellen. Wir werden das Modell einmal erstellen und anschließend die Modellartefakte verwenden, um das Modell in mehreren AWS-Konten und -Umgebungen bereitzustellen.

Leistungseffizienz

Die *Leistungseffizienz* (engl. *Performance Efficiency*) bezieht sich auf die effiziente Nutzung von Computerressourcen zur Erfüllung des Bedarfs und auf die Frage, wie diese Effizienz aufrechterhalten werden kann, wenn sich der Bedarf ändert oder Technologien weiterentwickelt werden.

Wir sollten die für unseren Machine-Learning-Workload geeignete Rechenleistung verwenden. So können wir zum Beispiel GPU-basierte Instanzen nutzen, um Deep-

Learning-Modelle effizienter zu trainieren, indem wir eine längere Warteschlangentiefe, höhere arithmetische Logikeinheiten oder mehr Register verwenden.

Machen Sie sich mit den Leistungsanforderungen der Modelle in Bezug auf Latenz und Netzwerkbandbreite vertraut und deployen Sie jedes Modell bei Bedarf möglichst nah am Kunden. Es gibt Situationen, in denen wir unsere Modelle direkt vor Ort einsetzen möchten, um die Leistung zu verbessern oder die Datenschutzbestimmungen einzuhalten. Mit »direkt vor Ort« ist gemeint, dass das Modell auf dem Gerät selbst ausgeführt wird und die Vorhersagen somit lokal ermittelt werden. Außerdem möchten wir die wichtigsten Leistungsmetriken unseres Modells kontinuierlich überwachen, um frühzeitig Veränderungen der Leistung zu erkennen.

Optimierung der Kosten

Wir können die anfallenden Kosten optimieren bzw. minimieren, indem wir verschiedene Preisoptionen für unsere Amazon-EC2-Instanzen nutzen. Zum Beispiel bieten Sparpläne, sogenannte *Savings Plans*, erhebliche Einsparungen im Vergleich zu den Preisen für On-Demand-Instanzen. Im Gegenzug verpflichten Sie sich, eine bestimmte Menge an Rechenleistung für eine bestimmte Zeit zu nutzen. Savings Plans sind eine gute Wahl, wenn Ihnen die Arbeitslasten bekannt sind oder sich nicht ändern, wie z. B. bei festen/stabilen Inferenzarbeitslasten.

Bei On-Demand-Instanzen zahlen wir für die Rechenkapazität stundenweise oder sekundenweise, je nachdem, welche Instanzen wir verwenden. On-Demand-Instanzen eignen sich am besten für neue oder zustandsbehaftete Workloads mit hohem Bedarf, wie z. B. kurzfristige Aufträge im Rahmen des Modelltrainings.

Schließlich können wir mit Amazon-EC2-Spot-Instanzen freie Amazon-EC2-Rechenkapazitäten zu einem Preis abrufen, der bis zu 90 % unter dem On-Demand-Preis liegt. Spot-Instanzen können flexible, fehlertolerante Arbeitslasten abdecken, wie z. B. Modelltrainingsaufträge, die nicht zeitkritisch sind. Abbildung 1-2 veranschaulicht die durch die Savings Plans, On-Demand- und Spot-Instanzen resultierende Kombination.

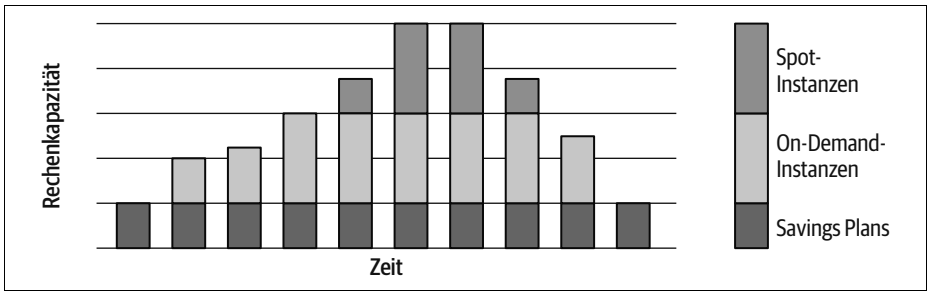

Abbildung 1-2: Optimieren Sie die Kosten, indem Sie eine Kombination aus Savings Plans, On-Demand- und Spot-Instanzen wählen.

Bei vielen der verwalteten Services können wir von dem Modell »Zahlen Sie nur für das, was Sie nutzen« profitieren. Bei Amazon SageMaker zahlen wir zum Beispiel lediglich für die Zeit, in der unser Modell trainiert wird, oder auch nur für die Zeit, in der wir unsere automatische Modelloptimierung durchführen. Beginnen Sie bei der Entwicklung von Modellen mit kleineren Datensätzen, um schneller und sparsamer iterieren zu können. Sobald wir ein gut funktionierendes Modell haben, können wir das Training auf den gesamten Datensatz ausweiten. Ein weiterer wichtiger Aspekt ist die Wahl der geeigneten Größe der Instanzen für das Training und Hosting von Modellen.

In vielen Fällen profitiert das Modelltraining von der GPU-Beschleunigung. Jedoch benötigt die Modellinferenz möglicherweise nicht die gleiche höhere Rechenleistung. Tatsächlich handelt es sich bei den meisten Machine-Learning-Workloads um Vorhersagen. Während das Trainieren des Modells mehrere Stunden oder Tage dauern kann, läuft das eingesetzte Modell wahrscheinlich 24 Stunden am Tag und sieben Tage die Woche auf Tausenden von Servern für Vorhersagen, die Millionen von Kunden unterstützen. Wir sollten entscheiden, ob unser Anwendungsfall einen 24 × 7-Echtzeit-Endpoint oder eine Batch-Vorhersage (*Batch Transformation*) auf Spot-Instanzen am späten Abend erfordert.

Amazons KI-Services und AutoML mit Amazon SageMaker

Wie wir wissen, umfassen datenwissenschaftliche Projekte viele komplexe, multidisziplinäre und iterative Schritte. Wir benötigen Zugang zu einer ML-Entwicklungsumgebung, die die Modellprototyping-Phase unterstützt und gleichzeitig einen reibungslosen Übergang zur Vorbereitung unseres Modells auf die Produktion ermöglicht. Wir werden wahrscheinlich mit verschiedenen ML-Frameworks und -Algorithmen experimentieren und einen benutzerdefinierten Code für das Modelltraining und die Inferenz entwickeln wollen.

In anderen Fällen möchten wir vielleicht einfach nur ein sofort verfügbares, vorab trainiertes Modell verwenden, um eine einfache Problemstellung zu lösen. Oder wir möchten AutoML-Techniken nutzen, um eine erste Ausgangsbasis für unser Projekt zu schaffen. AWS bietet eine breite Palette an Diensten und Funktionen für jedes Szenario. Abbildung 1-3 zeigt den gesamten KI- und ML-Stack von Amazon, einschließlich der AI-Services und Amazon SageMaker Autopilot für AutoML.

Amazons KI-Services

Für viele gängige Anwendungsfälle wie personalisierte Produktempfehlungen, Inhaltsmoderation oder Bedarfsprognosen können wir auch die verwalteten KI-Services von Amazon mit der Option des Feintunings auf unsere eigenen Datensätze anwenden. Wir können diese »1-Click«-KI-Services über einfache API-Auf-

rufe in unsere Anwendungen integrieren, ohne dass wir viel Erfahrung mit maschinellem Lernen benötigen (wenn überhaupt).

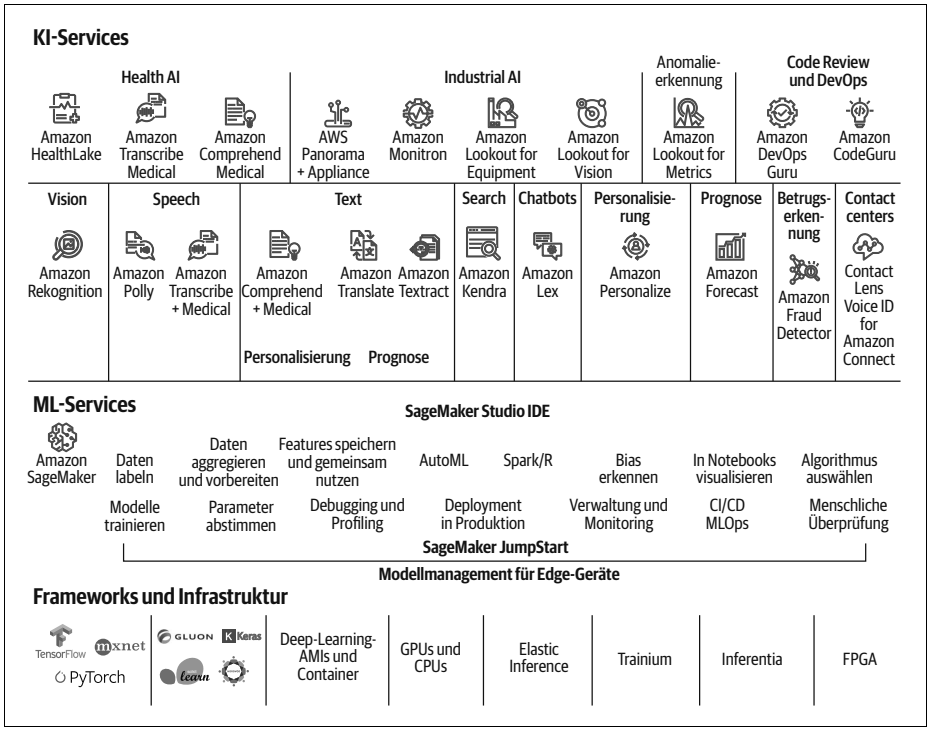

Abbildung 1-3: Der KI- und ML-Stack von Amazon

Die vollständig verwalteten KI-Services von AWS sind die schnellste und einfachste Möglichkeit, unseren Anwendungen mithilfe einfacher API-Aufrufe Intelligenz zu verleihen. Die KI-Services bieten vorab trainierte oder automatisch trainierte maschinelle Lernmodelle für Bild- und Videoanalyse, erweiterte Text- und Dokumentenanalyse, personalisierte Empfehlungen oder Bedarfsprognosen.

Zu den KI-Services zählen Amazon Comprehend für die Verarbeitung natürlicher Sprache, Amazon Rekognition für Computer Vision, Amazon Personalize für die Erstellung von Produktempfehlungen, Amazon Forecast für die Nachfrage- bzw. Bedarfsprognose und Amazon CodeGuru für die automatische Überprüfung vom Quellcode.

AutoML mit SageMaker Autopilot

In einem anderen Zusammenhang möchten wir vielleicht die sich wiederholenden Schritte der Datenanalyse, der Datenaufbereitung und des Modelltrainings für einfache und bekannte Fragestellungen aus dem Bereich des maschinellen Lernens automatisieren. Dies hilft uns, den Fokus auf komplexere Anwendungsfälle zu legen. AWS bietet AutoML als Teil des Amazon-SageMaker-Service an.

AutoML ist nicht auf SageMaker beschränkt. Viele der Amazon-KI-Services führen AutoML durch, um das beste Modell und die besten Hyperparameter für den gegebenen Datensatz zu finden.

AutoML bezieht sich im Allgemeinen auf die Automatisierung der typischen Schritte eines Modellentwicklungsworkflows, die wir zuvor beschrieben haben. Amazon SageMaker Autopilot ist ein vollständig verwalteter Service, der AutoML-Techniken auf unsere Datensätze anwendet.

SageMaker Autopilot analysiert zunächst unsere tabellarischen Daten, identifiziert die Art des maschinellen Lernproblems (z. B. Regression, Klassifikation) und wählt Algorithmen (z. B. XGBoost) zur Lösung der Aufgabenstellung aus. Außerdem wird der erforderliche Datenumwandlungscode erstellt, um die Daten für das Modelltraining vorzuverarbeiten. Autopilot erstellt dann eine Reihe von verschiedenen Pipelines mit Kandidaten für maschinelles Lernen, die verschiedene Variationen von Datentransformationen und ausgewählten Algorithmen darstellen. Er wendet die Datentransformationen in einem Feature-Engineering-Schritt an und trainiert und optimiert dann jeden dieser Modellkandidaten. Anschließend wird eine Rangliste (ein *Leaderboard*) der Modellkandidaten erstellt, die auf einer bestimmten Zielmetrik wie der Treffergenauigkeit auf dem Validierungsdatensatz basiert.

SageMaker Autopilot ist ein Beispiel für eine transparente Form von AutoML. Autopilot gibt nicht nur den Datenumwandlungscode an uns weiter, sondern generiert auch zusätzliche Jupyter Notebooks, die die Ergebnisse des Datenanalyseschritts und die Modellkandidaten-Pipelines zur Reproduktion des Modelltrainings dokumentieren.

SageMaker Autopilot kann in vielen verschiedenen denkbaren Szenarien Einsatz finden. Zudem können wir mehr Personen im Unternehmen in die Lage versetzen, Modelle zu erstellen, z. B. Softwareentwicklerinnen und -entwickler, die unter Umständen nur sehr wenig Erfahrung mit maschinellem Lernen haben. Wir können die Modellerstellung für einfach zu lösende Probleme des maschinellen Lernens automatisieren und unsere volle Aufmerksamkeit auf neue, komplexere Anwendungsfälle richten. Ebenso können die ersten Schritte der Datenanalyse und -aufbereitung automatisiert werden, und das Ergebnis kann dann als der Ausgangspunkt genutzt werden, an dem unser Fachwissen und unsere Erfahrung darüber entscheiden, ob die Modelle weiter optimiert und verbessert werden müssen. Die von Autopilot generierten Modellmetriken geben uns darüber hinaus einen guten Anhaltspunkt für die Modellqualität, die mit dem bereitgestellten Datensatz erreicht werden kann. Wir werden SageMaker Autopilot in Kapitel 3 noch genauer unter die Lupe nehmen.

Datenaufnahme, -exploration und -aufbereitung in AWS

In den Kapiteln 4, 5 und 6 werden wir jeweils noch gesondert die Datenaufnahme bzw. -eingabe (engl. *Ingestion*) sowie die Datenexploration und -aufbereitung (engl. *Preparation*) behandeln. Doch verschaffen wir uns zunächst einmal einen Überblick über diesen Teil des Modellentwicklungsworkflows, um zu erfahren, welche AWS-Services und Open-Source-Tools wir im Rahmen der einzelnen Schritte überhaupt nutzen können.

Datenaufnahme und Data Lakes mit Amazon S3 und AWS Lake Formation

Alles beginnt mit Daten. Und wenn wir in den letzten Jahrzehnten einen beständigen Trend beobachten konnten, dann ist es die anhaltende Explosion von Daten. Die Menge an Daten wächst exponentiell, und ihre Vielfalt wird immer größer. Heutzutage hängt der geschäftliche Erfolg oft eng mit der Fähigkeit eines Unternehmens zusammen, schnell einen Nutzen aus seinen Daten zu ziehen. Es gibt immer mehr Menschen, Teams und Anwendungen, die auf die Daten zugreifen und sie analysieren müssen. Aus diesem Grund gehen viele Unternehmen zu einem hoch skalierbaren, verfügbaren, sicheren und flexiblen Datenspeicher über, der oft als *Data Lake* bezeichnet wird.

Ein Data Lake ist ein zentrales und sicheres Repository, das es uns ermöglicht, Daten in beliebigem Umfang zu speichern, zu verwalten, zu erkunden und zu teilen. Mit einem Data Lake können wir jede Art von Analysen effizient durchführen und mehrere AWS-Services nutzen, ohne unsere Daten umwandeln oder verschieben zu müssen.

Data Lakes können sowohl strukturierte relationale Daten als auch halb strukturierte und unstrukturierte Daten enthalten. Wir können sogar Daten in Echtzeit einspeisen. Data Lakes bieten Data-Science- und Machine-Learning-Teams Zugang zu großen und vielfältigen Datensätzen, um genauere Modelle trainieren und einsetzen zu können.

Amazon Simple Storage Service (Amazon S3) ist ein Objektspeicher, der zum Speichern und Abrufen beliebiger Datenmengen von jedem Ort aus und in jedem Format entwickelt wurde. Wir können unsere Daten mit ausgefeilten Zugriffskontrollen organisieren, um unsere Geschäfts- und Compliance-Anforderungen zu erfüllen. Das Thema Sicherheit werden wir in Kapitel 12 noch eingehender behandeln. Amazon S3 ist für eine Dauerhaftigkeit (engl. *Durability*) von 99,999999999 % (elf Neunen) sowie für eine starke Read-after-Write-Konsistenz ausgelegt. S3 stellt eine beliebte Wahl für Data Lakes in AWS dar.

Wir können den AWS-Service *Lake Formation* nutzen, um unseren Data Lake zu erstellen. Der Service hilft beim Sammeln und Katalogisieren von Daten aus Datenbanken und Objektspeichern. Mit Lake Formation lassen sich nicht nur unsere Daten verschieben, sondern auch bereinigen und klassifizieren, und der Zugriff auf unsere sensiblen Daten wird mithilfe von ML-Algorithmen geschützt.

AWS Glue kann zur automatischen Erkennung und zum Profiling neuer Daten eingesetzt werden. Es handelt sich bei AWS Glue um einen skalierbaren und serverlosen Datenkatalog- und Datenaufbereitungsservice, der aus einer ETL-Engine, einem Apache-Hive-kompatiblen Service und einem Datentransformations- und Analyseservice besteht. Wir können Daten-Crawler erstellen, um regelmäßig neue Daten zu erkennen und zu katalogisieren. *AWS Glue DataBrew* ist ein Service, der durch seine benutzerfreundliche Oberfläche besticht und der die Datenaufnahme, -analyse, -visualisierung und -transformation vereinfacht.

Datenanalyse mit Amazon Athena, Amazon Redshift und Amazon QuickSight

Bevor wir mit der Entwicklung eines Machine-Learning-Modells beginnen, müssen wir zunächst die vorhandenen Daten verstehen. Im Rahmen der Datenanalyse untersuchen wir unsere Daten, sammeln Statistiken, prüfen auf fehlende Werte, berechnen Quantile und ermitteln Korrelationen zwischen den verschiedenen Daten.

Manchmal möchten wir einfach eben mal die verfügbaren Daten in unserer Entwicklungsumgebung analysieren und ein erstes Modell als Prototyp entwerfen. Vielleicht möchten wir auch schnell einen neuen Algorithmus ausprobieren. Wir nennen dies »Ad-hoc«-Exploration und Prototyping, bei dem wir einen Teil unserer Daten abfragen, um ein erstes Verständnis des Datenschemas und der Datenqualität für unser spezifisches maschinelles Lernproblem zu erhalten. Anschließend entwickeln wir den Modellcode und stellen sicher, dass er korrekt funktioniert. Diese Ad-hoc-Exploration und das Prototyping können in Entwicklungsumgebungen wie SageMaker Studio, AWS Glue DataBrew und SageMaker Data Wrangler durchgeführt werden.

Amazon SageMaker bietet uns eine gehostete sowie verwaltete Jupyter-Umgebung und mit SageMaker Studio eine integrierte Entwicklungsumgebung. Mit Tools wie Pandas (*https://pandas.pydata.org*), einer beliebten Open-Source-Bibliothek zur Datenanalyse und -verarbeitung in Python, können wir unmittelbar in unserer Notebook-Umgebung mit der Analyse von Datensätzen beginnen. Beachten Sie, dass Pandas In-Memory-Datenstrukturen (sogenannte DataFrames) verwendet, um Daten zu speichern und zu bearbeiten. Da viele Entwicklungsumgebungen nur über begrenzte Speicherressourcen verfügen, müssen wir vorsichtig dabei sein, wie viele Daten wir in Pandas in einen DataFrame laden.

Um Daten in unserem Notebook zu visualisieren, können wir beliebte Open-Source-Bibliotheken wie Matplotlib (*https://matplotlib.org*) oder Seaborn (*https://*

seaborn.pydata.org) nutzen. Mit Matplotlib können wir statische, animierte und auch interaktive Visualisierungen in Python erstellen. Seaborn baut auf Matplotlib auf und unterstützt zusätzliche statistische Diagramme – und bietet eine vergleichsweise einfach zu nutzende Syntax. Beide Datenvisualisierungsbibliotheken sind direkt mit den Pandas-Datenstrukturen kompatibel.

Die Open-Source-Bibliothek AWS Data Wrangler (*https://oreil.ly/Q7gNs*) erweitert die Leistungsfähigkeit von Pandas innerhalb von AWS. AWS Data Wrangler ermöglicht es, Pandas-DataFrames an AWS-Services wie Amazon S3, AWS Glue, Amazon Athena und Amazon Redshift anzubinden.

AWS Data Wrangler bietet optimierte Python-Funktionen zur Durchführung gängiger ETL-Aufgaben, um Daten zwischen Data Lakes, Data Warehouses und Datenbanken zu laden und auszulesen. Nachdem wir AWS Data Wrangler mit `pip install awswrangler` installiert und anschließend importiert haben, können wir unseren Datensatz direkt aus S3 in einen Pandas-DataFrame einlesen, wie hier gezeigt:

```
import awswrangler as wr


# Daten direkt aus Amazon S3 abrufen.
df = wr.s3.read_parquet("s3://<BUCKET>/<DATASET>/"))
```

AWS Data Wrangler verfügt auch über zusätzliche Speicheroptimierungsfunktionen, wie z. B. das Einlesen von Daten in Blöcken, sogenannten Chunks. Dies ist insbesondere dann hilfreich, wenn wir große Datensätze abfragen müssen. Wenn die Chunking-Funktion aktiviert ist, liest AWS Data Wrangler jede Datensatzdatei im Pfad ein und gibt sie als separaten Pandas-DataFrame zurück. Wir können auch die Chunk-Größe festlegen, um die Anzahl der Zeilen in einem DataFrame zurückzugeben, die dem numerischen Wert entspricht, den wir als Chunk-Größe definiert haben. Eine vollständige Übersicht über die Möglichkeiten finden Sie in der Dokumentation (*https://oreil.ly/4sGjc*). In Kapitel 5 werden Sie die Funktionen von AWS Data Wrangler noch ausführlicher kennenlernen.

Mit verwalteten Diensten wie Amazon Athena können wir von unserem Notebook aus interaktive SQL-Abfragen für die Daten in S3 durchführen. Amazon Athena ist eine verwaltete, serverlose, dynamisch skalierbare verteilte SQL-Abfrage-Engine, die für schnelle parallele Abfragen auf extrem großen Datensätzen konzipiert ist. Athena basiert auf Presto, der beliebten Open-Source-Abfrage-Engine, und erfordert keine Wartung. Bei Amazon Athena zahlen wir nur für die Abfragen, die wir ausführen. Außerdem können wir Daten in ihrer Rohform direkt aus unserem S3 Data Lake ohne zusätzliche Transformationen abfragen.

Amazon Athena nutzt auch den Service AWS Glue Data Catalog zum Speichern und Abrufen der für unsere SQL-Abfragen benötigten Schema-Metadaten. Wenn wir unsere Athena-Datenbank und -Tabellen definieren, verweisen wir auf den Datenspeicherort in S3. Athena speichert dann diese Zuordnung von Tabellen zu S3 (*Table-to-S3 Mapping*) im AWS Glue Data Catalog. Mit PyAthena, einer beliebten

Open-Source-Bibliothek, können wir Athena von unseren Python-basierten Notebooks und Skripten aus abfragen. In den Kapiteln 4 und 5 werden wir tiefer in Athena, AWS Glue Data Catalog und PyAthena eintauchen.

Amazon Redshift ist ein vollständig verwalteter Cloud-Data-Warehouse-Service, mit dem wir komplexe analytische Abfragen für strukturierte Daten in Petabyte-Größenordnungen durchführen können. Dabei werden unsere Abfragen über mehrere Knoten verteilt und parallelisiert. Im Gegensatz zu relationalen Datenbanken, die dafür optimiert sind, Daten in Zeilen zu speichern und hauptsächlich transaktionale Anwendungen zu bedienen, bietet Amazon Redshift eine spaltenbasierte Datenspeicherung, die für analytische Anwendungen optimiert ist, bei denen wir hauptsächlich an den zusammenfassenden Statistiken jener Spalten interessiert sind.

Amazon Redshift umfasst auch Amazon Redshift Spectrum, mit dem wir SQL-Abfragen von Amazon Redshift auf Exabytes an unstrukturierten Daten in unserem Amazon S3 Data Lake direkt ausführen können, ohne die Daten physisch verschieben zu müssen. Amazon Redshift Spectrum skaliert die benötigten Rechenressourcen automatisch auf der Grundlage der empfangenen Datenmenge, sodass Abfragen (engl. *Queries*) an Amazon S3 unabhängig von der Größe unserer Daten schnell ausgeführt werden.

Wenn wir Dashboard-artige Visualisierungen unserer Daten benötigen, können wir Amazon QuickSight nutzen. QuickSight ist ein benutzerfreundlicher, serverloser Geschäftsanalysedienst, mit dem wir schnell leistungsstarke Visualisierungen erstellen können. Wir können interaktive Dashboards und Berichte erstellen und sie sicher mit unseren Mitarbeitenden über Browser oder mobile Geräte teilen. QuickSight verfügt bereits über eine umfangreiche Bibliothek, die zahlreiche Funktionen zur Erstellung von Visualisierungen, Diagrammen und Tabellen bereithält.

QuickSight bietet zusätzlich Funktionen für maschinelles Lernen und für die Verarbeitung natürlicher Sprache, damit wir tiefergehende Erkenntnisse aus unseren Daten gewinnen können. Mit ML Insights können wir Trends und Ausreißer in unseren Daten entdecken, die sonst nicht direkt ersichtlich wären. Mithilfe dieses Tools kann jeder Was-wäre-wenn-Analysen und Prognosen durchführen, ohne dass er zwingend Erfahrung mit Machine Learning haben muss. Überdies können wir auch Prognose-Dashboards erstellen, indem wir QuickSight mit unseren in Amazon SageMaker erstellten Machine-Learning-Modellen verbinden.

Die Datenqualität mit AWS Deequ und SageMaker Processing Jobs validieren

Um qualitativ hochwertige Modelle entwickeln zu können, sind wir auf qualitativ hochwertige Daten angewiesen. Bevor wir unseren Trainingsdatensatz erstellen, müssen wir sicherstellen, dass unsere Daten bestimmte Qualitätsanforderungen erfüllen. Bei der Softwareentwicklung führen wir sogenannte Unit-Tests durch, um

sicherzustellen, dass unser Code den Design- und Qualitätsstandards entspricht und sich wie erwartet verhält. In ähnlicher Weise können wir Unit-Tests für unseren Datensatz durchführen, um sicherzustellen, dass die Daten unseren Qualitätsanforderungen entsprechen.

AWS Deequ (*https://oreil.ly/a6cVE*) ist eine Open-Source-Bibliothek, die auf Apache Spark aufbaut und mit der wir Unit-Tests für Daten definieren und die Datenqualität in großen Datensätzen erfassen können. Mithilfe der Unit-Tests können wir Anomalien und Fehler frühzeitig finden, bevor die Daten zum Trainieren von Modellen verwendet werden. Deequ ist für die Arbeit mit sehr großen Datensätzen (Milliarden von Zeilen) ausgelegt. Die Open-Source-Bibliothek unterstützt tabellarische Daten, d. h. CSV-Dateien, Datenbanktabellen, Logdateien oder vereinfachte (*flattened*) JSON-Dateien. Alle Daten, die in einen Spark-DataFrame passen, können auch mit Deequ validiert werden.

In einem späteren Beispiel werden wir AWS Deequ dafür nutzen, die Datenqualität unseres Beispieldatensatzes zu überprüfen. Da wir unsere Deequ-Unit-Tests möglichst in großem Maßstab durchführen möchten, profitieren wir davon, dass SageMaker Processing Jobs Apache Spark unterstützt. Im Rahmen dieses Setups sind wir nicht dazu gezwungen, selbst ein Apache-Spark-Cluster bereitzustellen – SageMaker Processing erledigt diese schwierige Aufgabe für uns. Wir haben es quasi mit »serverlosem« Apache Spark zu tun. Sobald wir die Hochwertigkeit unserer Daten bestätigt sehen, können wir unseren Trainingsdatensatz erstellen.

Trainingsdaten mit SageMaker Ground Truth labeln

In vielen datenwissenschaftlichen Projekten wird überwachtes Lernen (engl. *Supervised Learning*) eingesetzt. Beim Supervised Learning lernen unsere Modelle anhand von Beispielen. Zunächst müssen wir diese Beispiele sammeln und bewerten. Anschließend stellen wir die korrekten Labels bzw. Annotationen bereit, d. h., wir kennzeichnen, welcher Kategorie die jeweiligen Beispiele angehören. Wenn die bereitgestellten Labels nicht korrekt sind, lernt unser Machine-Learning-Modell auf Basis fehlerhafter Beispiele, was letztendlich zu ungenauen Vorhersagen führt. *SageMaker Ground Truth* hilft uns dabei, in Amazon S3 gespeicherte Daten effizient und akkurat zu labeln, wobei die Daten sowohl automatisch als auch von Menschenhand gelabelt werden.

Für gängige Aufgaben im Zusammenhang mit dem Labeln von Daten bietet SageMaker Ground Truth vorgefertigte Workflows und Schnittstellen. Wir legen die Labeling-Aufgabe fest und weisen den Labeling-Auftrag entweder externen (über Amazon Mechanical Turk) oder internen Arbeitskräften zu, z. B. unseren Mitarbeitenden. Ebenso können wir auch auf Drittanbieter in Form von Labeling-Dienstleistern zurückgreifen, die im AWS Marketplace aufgeführt sind und von Amazon vorab geprüft werden.

SageMaker Ground Truth sieht die Nutzung von Active-Learning-Techniken vor, die im Rahmen bereits vorgegebener Workflows verwendet werden können. Basierend auf den von den Anwenderinnen und Anwendern bzw. Mitarbeitenden vergebenen Labels, erstellt es ein Modell zur automatischen Klassifizierung einer Teilmenge der Daten. Da das Modell kontinuierlich von den gelabelten Daten der Anwender lernt, verbessert sich die Genauigkeit, und es müssen weniger Daten an die Anwender gesendet werden. Im Laufe der Zeit und bei ausreichender Datenmenge ist das Active-Learning-Modell von SageMaker Ground Truth in der Lage, qualitativ hochwertige und automatische Annotationen zu liefern, die insgesamt zu geringeren Kosten für das Labeln von Daten führen. Einen ausführlicheren Einblick in SageMaker Ground Truth erhalten Sie in Kapitel 10.

Datentransformationen mit AWS Glue DataBrew, SageMaker Data Wrangler und SageMaker Processing Jobs

Lassen Sie uns nun zum Thema Datentransformation übergehen. Wir gehen davon aus, dass wir unsere Daten in einem S3 Data Lake bzw. S3-Bucket haben. Außerdem haben wir durch die Datenanalyse eine grundlegende Vorstellung von unserem Datensatz gewonnen. Der nächste Schritt besteht nun darin, unsere Daten für das Modelltraining vorzubereiten.

Denkbare Datentransformationen könnten das Löschen oder Kombinieren von Daten in unserem Datensatz sein. Möglicherweise müssen wir Textdaten in Worteinbettungen (engl. *Word Embeddings*) konvertieren, um sie in Modellen, die auf natürlicher Sprache basieren, zu verwenden. Oder wir müssen Daten in ein anderes Format konvertieren, beispielsweise von einer numerischen in eine Textdarstellung oder umgekehrt. Es gibt zahlreiche AWS-Services, die uns dabei helfen können.

AWS Glue DataBrew ist ein Tool zur visuellen Datenanalyse und -aufbereitung. Mit 250 integrierten Transformationen kann DataBrew Anomalien erkennen, Daten zwischen Standardformaten konvertieren und ungültige oder fehlende Werte korrigieren. DataBrew kann Profile unserer Daten erstellen, zusammenfassende Statistiken berechnen und Korrelationen zwischen Spalten visualisieren.

Wir können mit Amazon SageMaker Data Wrangler auch benutzerdefinierte Datentransformationen in einem größeren Maßstab entwickeln. SageMaker Data Wrangler bietet Low-Code-Datentransformationen, die sich über die Benutzeroberfläche steuern lassen. Wir können Daten aus verschiedenen Quellen einlesen, darunter Amazon S3, Athena, Amazon Redshift und AWS Lake Formation. Ähnlich wie AWS DataBrew SageMaker verfügt Data Wrangler über vorkonfigurierte Datentransformationen, um den Datentyp von Spalten zu ändern, die Daten in eine 1-aus-n-Codierung (engl. *One-Hot-Encoding*) zu überführen und Textfelder zu verarbeiten. SageMaker Data Wrangler unterstützt benutzerdefinierte Funktionen, die auf Apache Spark basieren, und generiert sogar Code, einschließlich Python-Skripten und SageMaker Processing Jobs.

Mit SageMaker Processing Jobs können wir für die Datentransformation, Datenvalidierung oder Modellevaluierung einen benutzerdefinierten Code zur Verarbeitung von Daten, die in S3 liegen, ausführen. Bei der Konfiguration von SageMaker Processing Jobs legen wir die benötigten Ressourcen fest, einschließlich der Instanztypen und der Anzahl der Instanzen. SageMaker nimmt unseren benutzerdefinierten Code entgegen, kopiert unsere Daten aus Amazon S3 und greift dann auf einen Docker-Container zur Ausführung des Verarbeitungsschritts zurück.

Für die Datenverarbeitung mit Apache Spark und scikit-learn bietet SageMaker vorgefertigte Container-Images. Wir können bei Bedarf auch ein benutzerdefiniertes Container-Image bereitstellen. SageMaker stellt dann die von uns spezifizierten Cluster-Ressourcen für die Dauer des Auftrags bereit und beendet sie, sobald der Auftrag erledigt ist. Die Verarbeitungsergebnisse werden nach Beendigung des Auftrags in einen Amazon-S3-Bucket zurückgeschrieben.

Modelle mit Amazon SageMaker trainieren und feintunen

Nun möchten wir die Schritte im Rahmen des Modelltrainings und der Modellabstimmung in unserem Modellentwicklungsworkflow genauer betrachten und herausfinden, welche AWS-Services und Open-Source-Tools wir nutzen können.

Modelle mit SageMaker Experiments trainieren und tracken

Amazon SageMaker Training Jobs bieten eine Vielzahl von Funktionen, die wir für unser Modelltraining benötigen. Mit SageMaker Experiments können wir unsere individuellen Modelltrainingsdurchläufe organisieren, nachverfolgen und auswerten. Der SageMaker Debugger trägt zu einer höheren Transparenz während unseres Modelltrainings bei: Er erfasst automatisch Echtzeitmetriken während des Trainings und bietet eine visuelle Schnittstelle zur Analyse der Debugging-Daten. Außerdem erstellt er Profile, überwacht die Auslastung der Systemressourcen und identifiziert Ressourcenengpässe wie überlastete CPUs oder GPUs.

Für den SageMaker Training Job geben wir einfach den Amazon-S3-Speicherort unserer Daten, den Container zur Ausführung unseres Codes für den Algorithmus, der zum Trainieren des Modells genutzt werden soll, und die Art und Anzahl der benötigten SageMaker-ML-Instanzen an. SageMaker kümmert sich um die Initialisierung der Ressourcen und führt unser Modelltraining durch. Sofern es zuvor von uns festgelegt worden ist, startet SageMaker ein verteiltes Compute-Cluster. Sobald das Modelltraining abgeschlossen ist, speichert SageMaker die Ergebnisse in S3 und beendet die ML-Instanzen.

SageMaker unterstützt auch Managed Spot Training. Managed Spot Training nutzt zur Durchführung des Modelltrainings Amazon-EC2-Spot-Instanzen. Mit Spot-

Instanzen können wir die Kosten für das Modelltraining im Vergleich zu On-Demand-Instanzen um bis zu 90 % senken.

Neben SageMaker Autopilot können wir aus einem der in Amazon SageMaker integrierten Algorithmen wählen oder das Training des Modells individuell anpassen, indem wir unseren eigenen Modellcode (Script Mode) oder unseren eigenen Algorithmus bzw. Framework-Container einbinden.

Vorab integrierte (Build-in-)Algorithmen

SageMaker enthält viele bereits vorab integrierte Algorithmen, die es den Entwicklern von Machine-Learning-Modellen erleichtern, schnell mit dem Training und dem Einsatz von Machine-Learning-Modellen zu beginnen. Die integrierten Algorithmen erfordern keinen zusätzlichen Code. Wir müssen lediglich die Daten und alle Modelleinstellungen (Hyperparameter) bereitstellen und die Rechenressourcen angeben. Die meisten der integrierten Algorithmen unterstützen auch verteiltes Training, damit große Datensätze, die nicht auf eine einzelne Maschine passen, ebenfalls verarbeitet werden können.

Für Aufgaben im Zusammenhang mit Supervised Learning stehen Regressions- und Klassifikationsalgorithmen wie Linear Learner und XGBoost zur Verfügung. Für Empfehlungssysteme bieten sich vor allem sogenannte Factorization Machines an.

Für Aufgaben des unüberwachten Lernens (engl. *Unsupervised Learning*), wie z. B. Clustering, Dimensionsreduktion, Mustererkennung und Erkennung von Anomalien, sind zusätzliche integrierte Algorithmen verfügbar. Zu dieser Gattung von Algorithmen gehören beispielsweise die Hauptkomponentenanalyse (engl. *Principal Component Analysis*, PCA) und K-Means-Clustering.

Wir können auch integrierte Algorithmen für Textanalyseaufgaben wie Textklassifikation und Topic Modeling nutzen. Zu diesen Algorithmen gehören BlazingText und Neural Topic Model.

Für die Bildverarbeitung stehen integrierte Algorithmen zur Bildklassifikation und Objekterkennung, einschließlich Semantic Segmentation (semantische Segmentierung), zur Verfügung.

Eigene Skripte einsetzen (Script Mode)

Sollten Sie mehr Flexibilität benötigen oder gibt es keine integrierte Lösung, die für Ihren Anwendungsfall geeignet ist, können Sie Ihren eigenen Code für das Training Ihres Modells erstellen. Dieses Vorgehen wird oft als *Script Mode* (skriptbasierter Modus) bezeichnet. In diesem Modus können wir uns auf unser Trainingsskript konzentrieren, während SageMaker hoch optimierte Docker-Container für jedes der bekannten Open-Source-Frameworks wie TensorFlow, PyTorch, Apache MX-Net, XGBoost und scikit-learn bereitstellt. Wir können alle benötigten Codeabhängigkeiten über eine Requirements-Datei hinzufügen, und SageMaker kümmert

sich um die Ausführung des benutzerdefinierten Codes für das Modelltraining mit einem der integrierten Framework-Container, die sich je nach Wahl des Frameworks unterscheiden.

Eigene Container einsetzen – »Bring Your Own Container«

Für den Fall, dass weder die integrierten Algorithmen noch der Script Mode unseren Anwendungsfall abdecken, können wir unser eigenes benutzerdefiniertes Docker-Image zum Hosten des Modelltrainings verwenden. Docker ist ein Softwaretool, das Build- und Laufzeitunterstützung für isolierte Umgebungen bietet, die als *Docker-Container* bezeichnet werden.

SageMaker verwendet Docker-Images und -Container, um Funktionen für die Datenverarbeitung, das Modelltraining und die Vorhersagen bereitzustellen.

Der Ansatz von *Bring Your Own Container* (BYOC) bietet sich vor allem dann an, wenn das Paket oder die Software, die wir benötigen, nicht in einem der unterstützten Frameworks enthalten ist. Dies eröffnet uns unbegrenzte Möglichkeiten und bietet uns die größte Flexibilität, da wir unseren eigenen Docker-Container erstellen und problemlos alle benötigten Komponenten selbst installieren können. Diese Lösung wird üblicherweise dann gewählt, wenn individuelle Sicherheitsanforderungen bestehen oder wenn Bibliotheken im Docker-Container vorinstalliert werden müssen, um Abhängigkeiten von Drittanbietern zu vermeiden (z. B. mit PyPI, Maven oder Docker Registry).

Wenn wir die BYOC-Variante nutzen, um unser eigenes Docker-Image zu verwenden, müssen wir das Docker-Image zunächst in eine Docker-Registry wie DockerHub oder *Amazon Elastic Container Registry* (Amazon ECR) hochladen. Die BYOC-Variante sollte jedoch nur dann gewählt werden, wenn Sie mit der Entwicklung, Wartung und Unterstützung benutzerdefinierter Docker-Images mit einer effizienten Docker-Image-Pipeline vertraut sind. Andernfalls sollten Sie auf die integrierten SageMaker-Container zurückgreifen.

Wir müssen unseren Code zum Zeitpunkt der Erstellung nicht in ein Docker-Image »brennen«. Wir können einfach vom Docker-Image aus auf unseren Code in Amazon S3 verweisen und den Code dynamisch laden, wenn ein Docker-Container gestartet wird. Auf diese Weise lässt sich vermeiden, dass jedes Mal, wenn sich unser Code ändert, unnötigerweise erneut ein Docker-Image erstellt wird.

Vorkonfigurierte Lösungen und vortrainierte Modelle mit SageMaker JumpStart

Mit SageMaker JumpStart haben wir Zugriff auf gebrauchsfertige Machine-Learning-Lösungen und vortrainierte Modelle von AWS, TensorFlow Hub und PyTorch Hub. Die vorgefertigten Lösungen decken viele gängige Anwendungsfälle wie z. B. Betrugserkennung (engl. *Fraud Detection*), vorausschauende Wartung (engl. *Predictive*

Maintenance) und Bedarfs- bzw. Nachfrageprognosen (engl. *Demand Forecasting*) ab. Die vorgefertigten Modelle umfassen die Bereiche Natural Language Processing, Objekterkennung und Bildklassifikation. Wir können die Modelle mit unseren eigenen Datensätzen feintunen und sie mit nur wenigen Klicks in die Produktionsumgebung unseres AWS-Kontos übertragen. In Kapitel 7 werden wir tiefer in SageMaker JumpStart eintauchen.

Modelle mit SageMaker Hyperparameter Tuning feintunen und validieren

Ein weiterer wichtiger Schritt bei der Entwicklung qualitativ hochwertiger Modelle ist die Suche nach der richtigen Modellkonfiguration bzw. den Modell-Hyperparametern. Im Gegensatz zu den Modellparametern, die vom Algorithmus erlernt werden, steuern Hyperparameter, wie der Algorithmus die Parameter erlernt.

Amazon SageMaker verfügt über Funktionen zur automatischen Modellabstimmung und -validierung, die dazu dienen, die besten Modell-Hyperparameter für unser Modell und unseren Datensatz zu finden. Zunächst gilt es, eine Zielmetrik zu definieren, die optimiert werden soll, z.B. die Treffergenauigkeit (engl. *Accuracy*) in Bezug auf den Validierungsdatensatz, sowie die zu untersuchenden Hyperparameterbereiche. SageMaker führt dann zahlreiche Modelltrainingsaufträge aus, um die von uns festgelegten Hyperparameterbereiche zu untersuchen und die Ergebnisse auf Grundlage der vorgegebenen Metrik zu evaluieren, d.h., welches Modell hinsichtlich der gewählten Metrik am erfolgreichsten abschneidet.

Es gibt verschiedene Strategien, um Hyperparameterbereiche zu untersuchen: die Rastersuche (engl. *Grid Search*), die Zufallssuche (engl. *Random Search*) und die bayessche Optimierung zählen zu den gängigsten. In Kapitel 8 werden wir ausführlicher auf den entsprechenden SageMaker-Service Hyperparameter Tuning Job eingehen.

Modelle mit Amazon SageMaker und AWS Lambda Functions deployen

Sobald wir unser Modell trainiert, validiert und optimiert haben, sind wir bereit, es in Betrieb zu nehmen und zu überwachen. Es gibt im Allgemeinen drei Möglichkeiten, unsere Modelle mit Amazon SageMaker bereitzustellen bzw. zu deployen, je nach unseren Anwendungsanforderungen: SageMaker Endpoints für REST-basierte Vorhersagen, AWS Lambda Functions (bzw. AWS-Lambda-Funktionen) für serverlose Vorhersagen und SageMaker Batch Transform Jobs für Batch-Vorhersagen.

SageMaker Endpoints

Wenn wir den Betrieb des Modells für Vorhersagen mit geringer Latenz und in Echtzeit optimieren müssen, können wir unser Modell mithilfe von SageMaker-Hosting-Services bereitstellen bzw. deployen. Diese Services bzw. Dienste richten

einen SageMaker Endpoint ein, um unser Modell zu hosten, und stellen eine REST-API für Vorhersagen zur Verfügung. Die REST-API können wir von unseren Anwendungen aus aufrufen, um Modellvorhersagen zu erhalten. SageMaker Model Endpoints unterstützen eine automatische Skalierung zur Anpassung an den aktuellen Datenverkehr und werden für eine hohe Verfügbarkeit über mehrere AZs hinweg eingesetzt.

SageMaker Batch Transform

Wenn wir Vorhersagen für einen ganzen Datensatz benötigen, können wir SageMaker Batch Transform verwenden. Batch Transform ist für einen hohen Durchsatz optimiert – für den Fall, dass keine Vorhersagen in Echtzeit mit geringer Latenz benötigt werden. SageMaker aktiviert die vorgegebene Anzahl von Ressourcen, um umfangreiche Batch-Vorhersagen für unsere Daten, die in S3 gespeichert sind, durchzuführen. Sobald der Job abgeschlossen ist, speichert SageMaker die Daten erneut in S3 und fährt die Rechenressourcen wieder herunter.

Serverloses Modell-Deployment mit AWS Lambda

Eine weitere Möglichkeit zur Bereitstellung unserer Modellvorhersagen sind AWS-Lambda-Funktionen für serverlose Modellserver. Nach dem Training des Modells mit SageMaker können wir eine AWS-Lambda-Funktion einsetzen, die das Modell aus S3 abruft und die Vorhersagen bereitstellt. AWS Lambda hat Speicher- und Latenzbeschränkungen. Testen Sie diese Option also unbedingt in größerem Stil, bevor Sie sich endgültig für diesen Ansatz des Deployments entscheiden.

Streaming-Analysen und Machine Learning mit AWS

Bisher sind wir davon ausgegangen, dass alle unsere Daten an einem zentralen, statischen Ort verfügbar sind, z. B. in unserem S3-basierten Data Lake. In der Realität strömen jedoch kontinuierlich zahlreiche Daten zeitgleich aus diversen verschiedenen Quellen auf der ganzen Welt. In vielen Fällen möchten wir für diese Streaming-Daten Echtzeitanalysen durchführen und Machine-Learning-Modelle in Anwendung bringen, bevor sie in einem Data Lake landen. Um Wettbewerbsvorteile erzielen und schnell auf sich ändernde Kunden- und Markttrends reagieren zu können, ist eine kurze Zeitspanne bis zum Erhalt von (Geschäfts-)Erkenntnissen erforderlich.

Streaming-Technologien geben uns die Tools zum Erfassen, Verarbeiten und Analysieren von Datenströmen in Echtzeit an die Hand. AWS bietet eine Vielzahl von Optionen im Bereich der Streaming-Technologien, darunter *Amazon Kinesis* und *Amazon Managed Streaming for Apache Kafka* (Amazon MSK). Wir werden uns eingehend mit den Themen Streaming-Analysen und Machine Learning in Kapitel 11 beschäftigen.

Amazon Kinesis Streaming

Amazon Kinesis ist ein Streaming-Datenservice, der uns hilft, Daten in Echtzeit zu sammeln, zu verarbeiten und zu analysieren. Mit Kinesis Data Firehose können wir Echtzeitdaten vorbereiten und kontinuierlich in verschiedenen Zielsystemen wie Amazon S3 und Amazon Redshift laden. Mit Kinesis Data Analytics können wir die Daten verarbeiten und analysieren, sobald sie ankommen. Und mit Amazon Kinesis Data Streams können wir die Eingabe bzw. Integration von Datenströmen für benutzerspezifische Anwendungen verwalten.

Amazon Managed Streaming for Apache Kafka (Amazon MSK)

Amazon MSK ist ein Streaming-Datenservice, der die Infrastruktur und den Betrieb von Apache Kafka verwaltet. Apache Kafka ist eine beliebte, hochleistungsfähige, fehlertolerante und skalierbare Open-Source-Plattform für die Erstellung von Echtzeit-Streaming-Daten-Pipelines und -Anwendungen. Mit Amazon MSK können wir unsere Apache-Kafka-Anwendungen auf AWS ausführen, ohne dass wir selbst Apache-Kafka-Cluster verwalten müssen.

Streaming-Vorhersagen und Erkennung von Anomalien

Im Kapitel über Streaming-Daten werden wir uns auf die Analyse eines kontinuierlichen Stroms von Produktbewertungsnachrichten konzentrieren, die wir über die verfügbaren Onlinekanäle sammeln. Wir werden Streaming-Vorhersagen durchführen, um die Stimmungslage unserer Kunden zu ermitteln, sodass wir letztlich Aufschluss darüber erhalten, welche Kunden vorrangig Aufmerksamkeit benötigen.

Im Anschluss daran führen wir eine kontinuierliche Streaming-Analyse der eingehenden Produktbewertungsnachrichten durch, um die durchschnittliche Stimmung pro Produktkategorie zu erfassen (engl. *Sentiment Analysis*). Wir visualisieren fortlaufend das aktuelle durchschnittliche Stimmungsbild in einem Metrik-Dashboard, das an die Geschäftsbereichsverantwortlichen (engl. *Line of Business*, LOB) adressiert ist.

Die Geschäftsbereichsverantwortlichen können dadurch Stimmungstrends schneller erkennen und Maßnahmen einleiten. Wir berechnen auch einen Anomalie-Score der eingehenden Nachrichten, um Anomalien im Datenschema oder in den Datenwerten zu erkennen. Im Fall eines steigenden Anomalie-Scores können wir die zuständigen Anwendungsentwickler entsprechend informieren, um die Ursache unverzüglich zu untersuchen.

Als letzte Metrik berechnen wir auch kontinuierlich eine ungefähre Anzahl der eingegangenen Nachrichten. Diese Anzahl an Onlinenachrichten kann vom digitalen Marketingteam zur Messung der Wirksamkeit von Social-Media-Kampagnen verwendet werden.

AWS-Infrastruktur und individuell zusammengestellte Hardware

Ein wesentlicher Vorteil beim Cloud Computing ist die Möglichkeit, Infrastrukturoptionen zu nutzen, die speziell auf unsere Arbeitslast abgestimmt sind. AWS bietet viele Optionen für Hochleistungsberechnungs-, Netzwerk- und Speicherinfrastrukturen für unsere Data-Science-Projekte (siehe Abbildung 1-4). Sehen wir uns die einzelnen Optionen an, auf die wir im Folgenden Bezug nehmen werden.

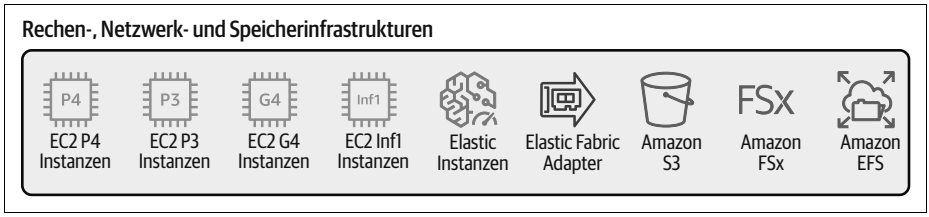

Abbildung 1-4: AWS-Infrastrukturoptionen für Data-Science- und Machine-Learning-Projekte

Varianten der SageMaker-Recheninstanzen

Bei AWS können wir je nach Arbeitslast aus einer Vielzahl von Instanztypen (*Compute Instances*) wählen. Im Folgenden finden Sie eine Übersicht der Instanztypen (*Instance Types*), die am häufigsten für Data-Science-Anwendungsfälle in Betracht gezogen werden:

Instanztyp T
: Universell einsetzbare Instanzen mit hoher temporärer Spitzenleistung, wenn wir keine konstant hohe CPU-Auslastung benötigen, aber von schnellen CPUs profitieren, sofern wir sie benötigen.

Instanztyp M
: Universell einsetzbare Instanzen mit einem ausgewogenen Verhältnis zwischen Rechenleistung, Speicher und Netzwerkbandbreite.

Instanztyp C
: Rechenoptimierte Instanzen, ideal für rechenintensive Workloads mit hohen CPU-Anforderungen.

Instanztyp R
: Speicheroptimierte Instanzen, die für Arbeitslasten optimiert sind, die von der Möglichkeit profitieren, große Datensätze im Speicher vorhalten zu können, wie z. B. Apache Spark.

P-, G-, Inferentia- und Trainium-Instanztypen
: Hochleistungsrecheninstanzen mit Hardwarebeschleunigung oder Koprozessoren wie GPUs oder von Amazon speziell entwickelte Hardware wie AWS Inferentia für die Inferenz und AWS Trainium für Arbeitslasten während des Trainings.

Amazon Elastic Inference Accelerator

An das Netzwerk angeschlossene Koprozessoren, die von anderen Instanztypen verwendet werden, wenn zusätzliche Rechenleistung für bestimmte Arbeitslasten benötigt wird, z. B. für die Verarbeitung von Batches und die Inferenz.

GPUs und kundenspezifische Rechenhardware von Amazon

Ähnlich wie Amazon S3 Speicherplatz in der Cloud bereitstellt, bietet die *Amazon Elastic Compute Cloud* (Amazon EC2) Rechenressourcen. Wir können aus über 350 Instanzen für unsere geschäftlichen Anforderungen und Arbeitslasten wählen. AWS bietet auch eine Auswahl an Intel-, AMD- und ARM-basierten Prozessoren. Die hardwarebeschleunigten P4-, P3- und G4-Instanztypen sind eine beliebte Wahl für hochleistungsfähiges, GPU-basiertes Modelltraining. Amazon bietet außerdem benutzerdefinierte Hardware an, die sowohl für das Modelltraining als auch für die Inferenz optimiert ist.

P4d-Instanzen bestehen aus acht NVIDIA-A100-Tensor-Core-GPUs mit 400 GBit/s Instanznetzwerk und Unterstützung für *Elastic Fabric Adapter* (EFA) mit NVIDIA GPUDirect RDMA (*Remote Direct Memory Access*). P4d-Instanzen werden in Hyperscale-Clustern mit der Bezeichnung *Amazon EC2 UltraClusters* bereitgestellt, die ML-Entwicklern, Forschern und Datenwissenschaftlern im Alltag eine Leistung der Kategorie Supercomputer bieten. Jedes EC2 UltraCluster mit P4d-Instanzen bietet Zugriff auf mehr als 4.000 NVIDIA-A100-GPUs, blockierungsfreie Netzwerke in der Größenordnung von Petabytes und Speicher mit hohem Durchsatz und geringer Latenz mittels Amazon FSx for Lustre.

P3-Instanzen bestehen aus bis zu acht NVIDIA-V100-Tensor-Core-GPUs und bieten einen Netzwerkdurchsatz von bis zu 100 GBit/s. P3-Instanzen liefern bis zu einem Petaflop an Mixed-Precision-Leistung pro Instanz. P3dn.24xlarge-Instanzen unterstützen auch EFA.

Die G4-Instanzen sind eine großartige Option für kosteneffiziente, kleine Trainings- oder Inferenzworkloads. G4-Instanzen bestehen aus NVIDIA-T4-GPUs mit einem Netzwerkdurchsatz von bis zu 100 GBit/s und bis zu 1,8 TB lokalem NVMe-Speicher.

Außerdem bietet AWS mit dem AWS-Trainium-Chip maßgeschneiderte Siliziumchips für das Training von Machine-Learning-Modellen und mit dem AWS-Inferentia-Chip das Pendant für Inferenzworkloads. Sowohl AWS Trainium als auch AWS Inferentia zielen darauf ab, die Performance beim Einsatz von Machine Learning zu steigern und die Infrastrukturkosten zu senken.

AWS Trainium wurde für Arbeitslasten beim Training von Deep-Learning-Modellen optimiert. Hierzu zählen die Bildklassifikation, die semantische Suche (engl. *Semantic Search*), Übersetzungsaufgaben, Spracherkennung, *Natural Language Processing* (NLP), also die natürliche Sprachverarbeitung, und Empfehlungsmaschinen.

AWS-Inferentia-Prozessoren unterstützen viele gängige Machine-Learning-Modelle, darunter Single-Shot-Detektoren und ResNet für Computer Vision sowie Transformer- und BERT-basierte (*Bidirectional Encoder Representations from Transformers*) NLP-Anwendungen.

AWS Inferentia ist über die Amazon-EC2-Inf1-Instanzen verfügbar. Wir können zwischen einem und bis zu 16 AWS-Inferentia-Prozessoren pro Inf1-Instanz wählen, die bis zu 2.000 Billionen (Tera-)Operationen pro Sekunde bewältigen. Das AWS Neuron SDK können wir so verwenden, um unsere TensorFlow-, PyTorch- oder Apache-MXNet-Modelle für die Ausführung auf Inf1-Instanzen zu kompilieren.

Inf1-Instanzen können dazu beitragen, bis zu 45 % der pro Inferenz anfallenden Kosten zu senken und den Durchsatz um 30 % im Vergleich zu Amazon-EC2-G4-Instanzen zu erhöhen. Tabelle 1-1 zeigt einige mögliche Instanztypen für die Modellinferenz, einschließlich des von Amazon speziell entwickelten Inferentia-Chips sowie CPUs und GPUs.

Tabelle 1-1: EC2-Instanz-Optionen für die Modellinferenz

Modellanforderungen	EC2 Inf1	EC2 C5	EC2 G4
Erfordert niedrige Latenzzeiten und einen hohen Durchsatz bei niedrigen Kosten.	X		
Geringe Bedeutung von Latenzzeiten und Durchsatz.		X	
Erfordert NVIDIAs CUDA-, CuDNN- oder TensorRT-Bibliotheken.			X

Amazon Elastic Inference stellt eine weitere Option dar, um eine beschleunigte Datenverarbeitung für die Modellinferenz zu nutzen. Mit Elastic Inference können wir einen Teil der GPU-Beschleunigung für jeden Amazon-EC2-Instanztyp (CPU-basiert) zuordnen. Außerdem können wir mit Elastic Inference die Intensität der Inferenzbeschleunigung unabhängig von der Wahl des Instanztyps für die Modellinferenz steuern.

Die Wahl von Elastic Inference anstelle von Inf1 kann sinnvoll sein, wenn wir andere Instanzeigenschaften als die von Inf1-Instanzen benötigen oder wenn unsere Leistungsanforderungen geringer sind als die der kleinsten Inf1-Instanz.

Elastic Inference skaliert von Single-Precision TFLOPS (Billionen Gleitkommaoperationen pro Sekunde) bis zu 32 Mixed-Precision TFLOPS an Inferenzbeschleunigung.

Graviton-Prozessoren sind von AWS speziell entwickelte ARM-Prozessoren. Die CPUs nutzen 64-Bit-Arm-Neoverse-Kerne und spezielle Siliziumprozessoren, die von AWS mit fortschrittlicher 7-nm-Fertigungstechnologie entwickelt wurden. ARM-basierte Instanzen können ein attraktives Preis-Leistungs-Verhältnis für viele in Amazon EC2 ausgeführte Workloads bieten.

Die erste Generation der Graviton-Prozessoren wird im Rahmen von Amazon-EC2-A1-Instanzen angeboten. Graviton2-Prozessoren bieten im Vergleich zur ersten Generation 7x mehr Leistung, 4x mehr Rechenkerne, 5x schnelleren Speicher

und doppelt so große Caches. Den Graviton2-Prozessor finden wir in Amazon-EC2-T4g-, -M6g-, -C6g- und -R6g-Instanzen.

Die AWS-Graviton2-Prozessoren bieten eine verbesserte Leistung für Videocodierungsarbeitslasten, Hardwarebeschleunigung für Komprimierungsarbeitslasten und Unterstützung für Prognosen mittels Machine Learning.

GPU-optimierte Netzwerktechnologie und kundenspezifische Hardware

AWS bietet fortschrittliche Netzwerklösungen, die uns bei der effizienten Durchführung von verteiltem Modelltraining und der Skalierbarkeit (*Scaling-out*) der Inferenz helfen können.

Die Amazon EFA ist eine Netzwerkschnittstelle für Amazon-EC2-Instanzen, die die Kommunikation zwischen den Knoten (engl. *Internode Communication*) in großem Umfang optimiert. EFA verwendet eine benutzerdefinierte OS-Bypass-Hardwareschnittstelle, die die Performance der Internode-Kommunikation steigert. Wenn wir die NVIDIA Collective Communications Library für das Modelltraining verwenden, können wir mit EFA auf Tausende von GPUs skalieren.

Wir können das Setup mit einer Netzwerkbandbreite von bis zu 400 GBit/s pro Instanz und NVIDIA GPUDirect RDMA für die Kommunikation zwischen GPUs mit geringer Latenz zwischen den Instanzen kombinieren. Dadurch erhalten wir bei Bedarf die Leistung von GPU-Clustern – bei gleichzeitiger bedarfsgerechter Elastizität und Flexibilität der Cloud.

Optimierte Speicheroptionen für das Modelltraining im großen Maßstab

Wir haben bereits die Vorteile des Aufbaus unseres Data Lake auf Amazon S3 kennengelernt. Wenn wir einen schnelleren Speicherzugriff für das verteilte Modelltraining benötigen, können wir Amazon FSx for Lustre verwenden.

Amazon FSx for Lustre bietet das Open-Source-Dateisystem Lustre als vollständig verwalteten Dienst an. Lustre ist ein hochleistungsfähiges Dateisystem, das Latenzzeiten von unter einer Millisekunde, einen Durchsatz von bis zu Hunderten von Gigabytes pro Sekunde und Millionen von IOPS *(Input/Output Operations per Second*, also Ein-/Ausgabeoperationen pro Sekunde) bietet.

Wir können FSx-for-Lustre-Dateisysteme mit Amazon-S3-Buckets verknüpfen. Dadurch können wir über das FSx-Dateisystem und von Amazon S3 aus auf die entsprechenden Daten zugreifen und sie verarbeiten. Mit FSx for Lustre können wir unsere Recheninstanzen für das Modelltraining so einrichten, dass sie über einen gemeinsamen Hochleistungsspeicher auf denselben Datensatz zugreifen.

Amazon Elastic File System (Amazon EFS) ist ein weiterer Dateispeicherservice, der eine Dateisystemschnittstelle (engl. *Filesystem Interface*) für bis zu Tausende von Amazon-EC2-Instanzen bietet. Die Dateisystemschnittstelle bietet standardmäßige Schnittstellen für die Ein- und Ausgabe von Dateien des Betriebssystems (I/O-APIs) und ermöglicht Dateisystemzugriffssemantiken, wie etwa starke Konsistenz und Sicherheitsschutz der Dateien.

Kosten mit Tags, Budgets und Alerts einsparen

Im Verlauf des Buchs geben wir einige wertvolle Tipps dazu, wie Sie die Kosten für Data-Science-Projekte mit Amazons KI- und Machine-Learning-Stack senken können. Insgesamt sollten wir unsere Ressourcen immer mit dem Namen des Geschäftsbereichs, der Anwendung, der Umgebung und des Benutzers kennzeichnen. Wir sollten Tags verwenden, die Aufschluss darüber geben, wofür wir unser Geld ausgeben. Zusätzlich zu den in AWS integrierten Tags, die der Kostenzuweisung dienen, können wir unsere eigenen benutzerdefinierten Zuweisungstags, die spezifisch für unseren Bereich sind, bereitstellen. Mit AWS Budgets können wir Warnungen (engl. *Alerts*) erstellen, wenn sich die Kosten einem bestimmten Schwellenwert nähern oder diesen überschreiten.

Zusammenfassung

In diesem Kapitel haben wir die Vorzüge der Entwicklung von Data-Science-Projekten in der Cloud erörtert, wobei der Schwerpunkt insbesondere auf der AWS-Cloud lag. Wir haben gezeigt, wie wir unseren Anwendungen schnell Intelligenz verleihen können, indem wir Amazons AI- und Machine-Learning-Stack nutzen. Zudem haben wir das Konzept von AutoML vorgestellt und erklärt, wie SageMaker Autopilot einen transparenten Ansatz für AutoML bietet. Anschließend haben wir einen typischen Workflow für Machine Learning in der Cloud besprochen und die relevanten AWS-Services vorgestellt, die bei jedem Schritt des Workflows helfen. Außerdem haben wir einen Überblick über die verfügbaren Tools zur Workflow-Orchestrierung gegeben, um Machine-Learning-Pipelines zu erstellen und zu automatisieren. Wir haben erläutert, wie Streaming-Analysen und Machine Learning auf Echtzeitdaten angewendet werden können. Abschließend haben wir einen Überblick gegeben über die verschiedenen Optionen, die die AWS-Infrastruktur für unsere Data-Science-Projekte bereithält.

In Kapitel 2 besprechen wir bekannte Data-Science-Anwendungsfälle in Branchen wie der Medienindustrie, der Werbebranche, im Bereich des Internet der Dinge (*Internet of Things*, IoT) und der Fertigung.