Aufbau einer Datenvorbereitungsumgebung für Machine-Learning-Modelle zur Berechnung von kundenindividuellen Produktabschlüssen

Laura Offermanns

Technische Hochschule Mittelhessen

Fachbereich MND
Wilhelm-Leuschner-Straße 13
61169 Friedberg
E-Mail:
laura.offermanns@mnd.thm.de

Prof. Dr. Harald Ritz

Technische Hochschule Mittelhessen

Fachbereich MNI
Wiesenstraße 14
35390 Gießen
E-Mail:
harald.ritz@mni.thm.de

Dr. Michel Becker

Mittelstand.ai GmbH & Co. KG

Schiffenberger Weg 110
35394 Gießen
E-Mail:
michel.becker@mittelstand.ai

Kategorie

Masterarbeit

Schlüsselwörter

PySpark, SQL, Apache Beam, Datenvorbereitung, Machine Learning, Python, Google Cloud, Dataflow, Dataproc

Zusammenfassung

Große Datenmengen, auch Big Data genannt, können oftmals mit traditionellen Datenverarbeitungswerkzeugen nicht adäquat verarbeitet werden. Es stellt eine Herausforderung dar, diese zu transformieren und zu aggregieren. Bei der Mittelstand.ai GmbH & Co. KG in Gießen können Banken kundenindividuelle Produkte bestellen, welche auf Basis von Machine-Learning-Algorithmen eine Empfehlung geben, welches Bank- oder Versicherungsprodukt ein Kunde als Nächstes abschließen würde. Dazu wurde eine Analyse-Plattform entwickelt, in welcher Banken ihre individuellen Produkte bestellen und abrufen können. Weiterhin wird diese zum Hochladen der Daten verwendet. Die Machine-Learning-Algorithmen basieren auf den vergangenen Kundendaten der Banken und werden für jede Bank differenziert angewendet.

Die Transformation und Aggregation der einzelnen Daten wurde mit Python und den Bibliotheken Pandas und Numpy realisiert. Die Durchlaufzeit einer Datenvorbereitung für eine Bank ist sehr hoch und nicht automatisiert. Das bedeutet, bei jedem Durchlauf wird ein Mitarbeiter-PC benötigt, welcher in dieser Zeit aufgrund der hohen Rechenleistung nicht anderweitig verwendet werden kann. Außerdem werden die csv-Dateien mit den zu analysierenden Daten manuell heruntergeladen und nehmen somit viel Speicherplatz ein. Jede Bank lädt pro Monat vier Dateien hoch, welche über die Monate hinweg verarbeitet werden. Dadurch kommt die lokale Ausführung schnell an ihren Grenzen, in der Verarbeitung und auch im Datenspeicher.

Mit einer neuen Datenvorbereitungsumgebung soll die Laufzeit der Transformations-, Filter- und Aggregationsvorgänge verringert werden sowie eine Umgebung geschaffen werden, welche nicht lokal auf den PCs ausgeführt wird, sondern in der Google Cloud realisiert werden kann. Eine weitere Anforderung ist die Automatisierung der Umgebung. Durch die Einbindung von Google-Cloud-Produkten, welche bereits in der Mittelstand.ai verwendet werden, soll es möglich sein, geplante Datenvorbereitungen durchzuführen, da jede Bank monatlich ihre Daten in die Analyse-Plattform hochlädt.

Unter diesem Aspekt wurde eine Testumgebung entwickelt, welche mithilfe von Apache Beam und Dataflow die Daten der Banken verarbeitet. Dabei wurden Filterfunktionen, welche das Alter der Kunden, die Geschäftsform und beispielsweise die Einwilligung in die Verarbeitung der personenbezogenen Daten berücksichtigt, angewandt. Weiterhin konnten erste Transformationen durchgeführt werden, welche sich mit der Aufarbeitung des Datensatzes beschäftigen. Im Schritt der Datenaggregation konnte Apache Beam nicht überzeugen, da das vorgegebene Datenformat mit einzelnen Iterationen in Python verarbeitet werden musste und somit den Aspekt der Zeiteinsparung in einem Anwendungstest nicht erfüllte. Durch das Iterieren der Datensätze, welche durch die speziellen Anforderungen der Vorbereitung entstanden ist, mussten eigene Funktionen definiert werden. Diese Funktionen arbeiten nicht optimal mit dem Apache Beam Framework aufgrund der fehlenden Parallelisierung zusammen. Die Automatisierung mit Dataflow war jedoch erfolgreich und konnte planbar gestaltet werden.

Um auch Erfolge in der Effizienz zu erzielen, wurde

eine weitere Testumgebung eingerichtet. Diese nutzt als Basis das Framework PySpark und zusätzlich SQL-Abfragen, um die Daten vorzubereiten. Die SQL-Abfragen werden individuell für jede Bank generiert. Dazu dient ein String Builder, welcher in der PySpark Umgebung implementiert wurde. Dieser beachtet Informationen wie beispielsweise die eindeutige ID der Bank und die Konfigurationen, welche eine Bank bei einer Produktbestellung angeben kann. Dazu gehört ebenso die Tiefe der Analyse, da die Datengrundlage die Möglichkeit bietet, Kundenverhalten sehr genau zu analysieren. Eine weitere Möglichkeit, die Bestellung zu konfigurieren, liegt in der Einstellung der miteinzubeziehenden Daten. Das hat den Vorteil, dass eine Bank die Möglichkeit hat, die letzten 15 Monate oder auch nur sechs Monate in ihr Ergebnis miteinzubeziehen. Dadurch können bankenindividuelle Ereignisse, welche zu einem abweichenden Verhalten der Bankkunden geführt hat, ausgeschlossen werden und verfälschen somit das Ergebnis nicht.

PySpark ist ebenfalls in der Lage, in den Funktionen die Konfigurationen der Banken miteinzubeziehen. Die wichtigste PySpark-Funktion ist die Pivot-Tabelle, mit welcher die Daten auf bestimmte Kennzahlen transformiert und aggregiert werden. Im Gegensatz zur alten Umgebung arbeitet diese effizienter und kann tiefere Datenebenen verarbeiten. Zur Ausführung in der Google Cloud wurde Dataproc verwendet. Die Clusterkonfiguration entspricht den Bankenanforderungen und stellt somit sicher, dass die Daten ordnungsgemäß verarbeitet werden. Sobald ein Dataproc-Cluster hochgefahren wurde, kann ein PySpark-Job auf diesem ausgeführt werden. Mit der aktivierten Option des Autoscalings ist es möglich, die Clusterauslastungen automatisch zu regeln und somit eine optimale Auslastung der Rechenkapazität und der Kosten zu bieten. Des Weiteren besteht die Möglichkeit Durchläufe für Banken zu planen und diese regelmäßig durchzuführen, ohne dass ein Mitarbeiter eingreifen muss.

Die Kombination der PySpark-Funktion mit den SQL-Befehlen führte zu einer effizienten Verarbeitung. In einer Testumgebung konnte die Durchlaufzeit etwa halbiert werden. Neben der Implementierung eines Bankproduktes ist die modulare Erweiterung des Codes anzumerken. Durch die Modularität können weitere Filter, Transformationen, Aggregationen und Berechnungen eingeführt werden. Ebenfalls sind die Mitarbeiter vertraut mit der Nutzung von Python und den Datentypen des PySpark-Frameworks, da im Vorfeld bereits mit Pandas gearbeitet wurde und Tabellenstrukturen bekannt sind. Aufgrund einer parallelen Entwicklung konnte ein Data Warehouse verwendet werden und ermöglichte den Beginn einer Automatisierung der Datenvorbereitungsumgebung. Mit der Nutzung von SQL-Abfragen konnte die Anbindung an das Data Warehouse realisiert werden. Nachdem alle Dateien der Bank vorbereitet wurden, ist es möglich, diese wieder zusammenzuführen und weitere Aufbereitungen, beispielsweise der Spaltennamen, durchzuführen.

Eine Tabelle in dem Data Warehouse mit einem vorbereiteten Enddatensatz stellt die Schnittstelle zur Umgebung dar, auf welcher die Machine-Learning-Algorithmen anwendet werden. Auch diese bietet die Möglichkeit, automatisiert auf die neue Tabelle zuzugreifen.

Abschließend lässt sich erwähnen, dass erfolgreich eine Umgebung implementiert wurde, welche die vorhandenen Anforderungen an eine neue Datenvorbereitungsumgebung erfüllt und einen Ansatz für viele Erweiterungs- und Automatisierungsmöglichkeiten mit sich bringt.

Literatur

D'Onofrio, Sara; Meier, Andreas. Big Data Analytics. Wiesbaden. Springer Vieweg (2021)

Drabas, Tomasz; Lee, Denny. Learning PySpark. Birmingham. Packt Publishing (2017)

Frochte, Jörg. Maschinelles Lernen. München. Hanser (2021)