Machine Learning Pipeline Automation

Amar Kisoensingh

27 mei 2021

Voorwoord

Samenvatting

Summary

Afkortingen

PaaS Platform as a Service 15, 16

Lijst van figuren

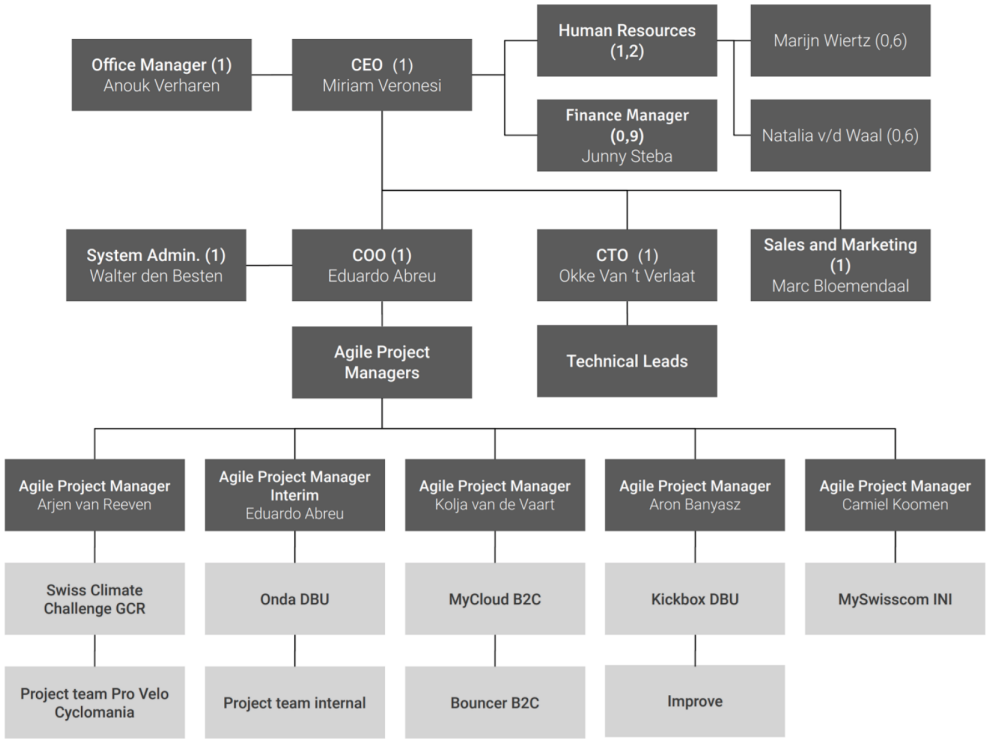
Lijst van tabellen

Inhoudsopgave

1 Inleiding

NGTI is een software ontwikkelbedrijf dat gevestigd is in Rotterdam. Opgericht in 2012 maakt NTGI applicaties (apps) voor mobiel en/of webgebruik. Naast het ontwikkelen werkt NGTI aan het hele traject om een app heen, namelijk de probleemstelling, mockups, wireframes en prototyping. Daarnaast levert NGTI ook support en lost bugs op nadat een app live is gegaan [1]. Ook maakt NGTI white label apps en frameworks [2].

Het bedrijf heeft, zoals de meeste bedrijven, een organogram (Figuur 1.1). In de praktijk is dit echter niet terug te vinden en wordt de structuur gezien als "plat". Collega's kunnen elkaar laagdrempelig benaderen waardoor niemand een onbekende is en verschillende disciplines makkelijk met elkaar samen kunnen werken.



Figuur 1.1: Organogram van NGTI op 29-03-2021 [3].

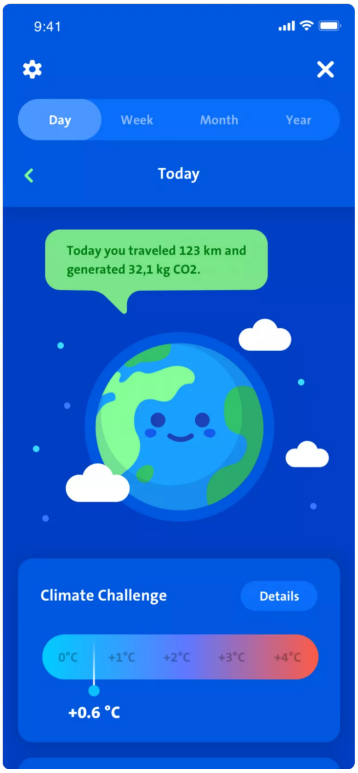
NGTI is een dochterbedrijf van Swisscom [4]. Sinds maart 2021 is het bekend gemaakt dat Swisscom van plan is om een afdeling, Swisscom DevOps Center, te fuseren met NGTI. Omdat de fusie onzekerheid met zich meebrengt voor de structuur en manier hoe NGTI werkt, zal de situatie vóór de fusie aangehouden worden gedurende het afstuderen.

1.1 Projecten van NGTI

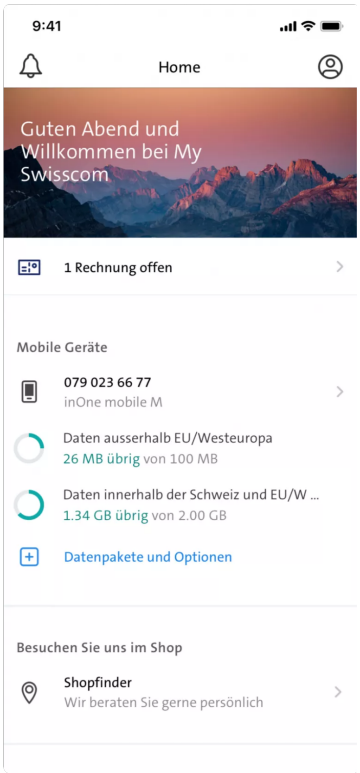
NGTI heeft een vrij breed portfolio met apps voor verschillende doeleinden. Een van deze apps is de Climate Challenge App [5]. Met deze app kunnen gebruikers hun C02-voetafdruk en impact in kaart brengen. Er wordt bijgehouden hoeveel kilometer de gebruiker reist en met welk vervoersmiddel. De app is onderdeel van twee bestaande nieuwsapps, Blick en Bluewin [6]. Het

doel is om de gebruiker aan te sporen om groener te reizen. Een screenshot van de app is te zien in Figuur 1.2.

Een andere oplossing is de My Swisscom App [7]. Dit is een native app voor Android en iOS waarbij Swisscom-klanten hun contract kunnen bestellen, wijzigen of beëindigen. In de app kunnen klanten ook de dataverbruik zien en instellingen voor abonnementen wijzigen. Een screenshot van de app is te zien in Figuur 1.3.



Figuur 1.2: Screenshot van de Swiss Climate Challenge app [5].



Figuur 1.3: Screenshot van de My Swisscom App [7]

1.2 Tools die worden gebruikt

Om productief te zijn, gebruikt NGTI een aantal tools en programma's om producten te maken en te communiceren met zowel collega's als klanten. De meest gebruikte en belangrijkste zijn Slack, Google Workspace, Zoom en Microsoft Teams.

1.2.1 Slack

Interne communicatie gaat via Slack. Het programma faciliteert collega's om elkaar met een lage instap te benaderen en berichten die voor het hele bedrijf relevant zijn te versturen. Ook zijn er 'channels' beschikbaar over specifieke onderwerpen, zoals: *#dev*, *#ios* en *#test-automation*.

1.2.2 Google Workspace

Met Google Workspace kunnen bestanden en documenten gemaakt, opgeslagen en gedeeld worden. Dit is mogelijk via een browser waardoor werknemers geen software hoeven te installeren. NGTI gebruikt het ook om collaboratief en parallel te werken aan hetzelfde document.

1.2.3 Microsoft Teams en Zoom

Voorheen werd Zoom alleen gebruikt om te videobellen met collega's en geïnterviewden. In de tijd van de pandemie is Zoom echter een belangrijke speler geworden om effectief samen te werken. Meetings zoals introducties van nieuwe collega's of demo's van producten worden online gehouden.

1.3 Aanleiding opdracht

NGTI wilt de gebruikerservaring van haar apps verbeteren en een voorsprong hebben op haar concurrenten. Dit kan NGTI op een aantal manieren doen waarvan apps "slimmer" maken er een van is. Het slimmer maken houdt voor NGTI in dat de app bijvoorbeeld beter kan anticiperen wat de gebruiker wilt en nodig heeft op een gegeven moment. Dit kan onder andere door het toepassen van machine learning (ML). De opdracht kan verdeelt worden in twee onderdelen:

- Onderzoek naar hoe een pipeline opgezet kan worden op een cloud computing platform door middel van een framework
- Onderzoek naar het maken van een platform-agnostische oplossing

Daarnaast zal een proof-of-concept (PoC) gemaakt worden om aan te tonen of het haalbaar is in de praktijk. Een diepere duik in het probleem en het definiëren van de onderdelen is te vinden in hoofdstuk 2.

1.4 Leeswijzer

TODO: Schrijf leeswijzer voor elk hoofdstuk

2 Probleemanalyse

Zoals beschreven in paragraaf 1.3 wilt NGTI ML toepassen om haar apps slimmer te maken. Hier zijn een aantal redenen voor, onder andere om de gebruikerservaring te verbeteren en om een voorsprong te hebben op concurrenten.

Het 'slimmer' maken van applicaties kan op verschillende manieren, maar met machine-learning kan een platform gebouwd worden waarmee elke richting op gegaan kan worden. Om machine-learning te implementeren in haar applicaties loopt NGTI tegen een aantal obstakels aan, namelijk: expertise vereist in het ML domein, benodigde tijd om een pipeline op te zetten en vendor lock-in.

2.1 Obstakels voor NGTI

NGTI loopt tegen de voorgenoemde obstakels aan omdat NGTI weinig ervaring heeft met ML. In samenwerking met een extern bedrijf worden modellen getraind die vervolgens gebruikt worden in apps van NGTI. Om zelf modellen te trainen heeft NGTI kennis over ML, ML pipelines en tijd nodig. Bovendien wilt NGTI niet bij één cloud computing platform haar infrastructuur opzetten maar gemakkelijk kunnen schakelen tussen platformen. De obstakels worden in de volgende koppen verduidelijkt.

2.1.1 Expertise Machine Learning

ML is ingewikkeld en diepgaand onderwerp. Om een model te trainen is kennis nodig van verschillende domeinen: data mining, software engineering en statistieken. Doordat er voorkennis nodig is om een model te trainen en een pipeline goed op te zetten, is het vaak te hoogdrempelig voor developers om een start te maken met ML. De expertise is daarnaast niet in een korte tijd te vergaren.

2.1.2 Opzetten pipeline

Bovenop de complexiteit van ML zelf bestaan er verschillende manieren om een model te trainen. Het opzetten van ML pipelines is daar een van. Een pipeline is een workflow dat bestaat uit een aantal stappen die doorgelopen worden om een model te trainen. In elke stap worden acties uitgevoerd, zoals het verwijderen van onbruikbare data of de prestatie van modellen vergelijken en een rapport met uitslagen genereren. Het opzetten van zo een pipeline én de actie(s) in de stappen definiëren kost tijd en vereist specifieke kennis. Daarnaast zijn de stappen en acties vaak hetzelfde voor verschillende pipelines. Het automatiseren en hergebruiken van stappen en acties tussen pipelines zou tot onder andere tijdwinst en het verminderen van herhaling van code kunnen leiden.

2.1.3 Vendor lock-in

Er bestaan een aantal diensten, zogenoemde Platform as a Service (PaaS), waarbij je een pipeline kan opzetten en acties kan definiëren. Een van de problemen met een PaaS is vendor lock-in. Dit betekent dat, als er eenmaal een pipeline is opgezet, de overdraagbaarheid van de pipeline naar een andere PaaS vrijwel onmogelijk is. Ook zijn de opties en mogelijkheden om uit te breiden in de toekomst gelimiteerd.

2.2 Vooronderzoek naar bestaande oplossingen

Het gebruik van ML pipelines is geen nieuwe techniek en is mogelijk bij PaaSs en bedrijven die zich specialiseren in ML pipelines. Het gemakt met zulke services is dat de gebruiker gelijk kan beginnen met het trainen van ML modellen en zich niet zorgen hoeft te maken over infrastructurele details. Echter is er sprake van vendor lock-in en kunnen services van deze bedrijven niet gebruikt worden.

Gedurende de vooronderzoek zijn "Infrastructure as Code"frameworks naar voren gekomen. Dit zijn frameworks waarmee, middels code, een infrastructure binnen een Platform as a Service (PaaS) opgezet kan worden. Dit kan gebruikt worden als onderdeel van de PoC en wordt in een van de deelvragen verder onderzocht.

2.3 Doelstelling

Om haar doel, de gebruikerservaring van apps verbeteren en een voorsprong hebben op concurrent, te bereiken is NGTI van plan ML pipelines te gebruiken om ML toe te passen. De gewenste oplossing is een systeem waarbij developers met weinig tot geen kennis een model kunnen trainen. Het systeem moet de infrastructurele taken voor zich nemen, zoals het opzetten van een pipeline en de stappen en acties automatiseren. Daarnaast moet een ML pipeline pijnloos doorlopen kunnen worden op verschillende platformen zodat het systeem platform-agnostisch is.

2.4 Hoofd- en deelvragen

Uitgaand van de drie obstakels kan de hoofdvraag als volgt worden geformuleerd:

In welke mate kan een machine learning pipeline worden geautomatiseerd onafhankelijk van het onderliggende cloud computing platform?

De hoofdvraag kan worden onderbouwd met vier deelvragen. Om te beginnen is het verstandig om te weten welke stappen er in een machine learning pipeline zit:

Waar bestaat een machine learning pipeline uit?

Daarnaast is een framework nodig dat, door middel van code, verschillende cloud computing platformen kan beheren:

Hoe kan een framework verschillende cloud computing platformen beheren om een machine learning pipeline op te zetten?

Ten slotte wordt een PoC gemaakt om te laten zien of het probleem oplosbaar is. Hiervoor is een doordachte voorbereiden onmisbaar:

Hoe ziet de architecturale blauwdruk van een applicatie, waarmee een platform-onafhankelijk machine learning pipeline opgezet kan worden, eruit?

3 Onderzoeksmethoden en scope

Om elke hoofd- en deelvraag te beantwoorden, wordt er bij elk gebruik gemaakt van een onderzoeksmethode. Volgens Scribbr [8] zijn er twee onderzoeksmethoden: kwantitatief en kwalitatief. Bij een kwantitatief onderzoeksmethode wordt data verzameld waarmee bijvoorbeeld grafieken of tabellen gemaakt kunnen worden. De focus bij een kwalitatief onderzoeksmethode ligt bij het verzamelen van verschillende interpretaties en opvattingen. Hierop kan optioneel een eigen interpretaties op gemaakt worden. [9].

Onder kwantitatief en kwalitatief vallen verschillende dataverzamelingsmethoden. Deze beschrijft simpelweg de manier hoe data wordt verzameld. Dit kan bijvoorbeeld met een enquête, literatuuronderzoek op websites en in boeken of een onderzoek over een lange periode [9].

Elke hoofd- en deelvraag is gekoppeld aan een onderzoeksmethoden. Vervolgens is beschreven welk(e) dataverzamelingsmethode(n) wordt gebruikt met een korte toelichting. Daarnaast wordt op een hoog niveau de scope bepaald.

D1: Waar bestaat een machine learning pipeline uit?	
Methode(s)	Kwalitatief
Dataverzamelings methode(n)	Literatuuronderzoek, fundamenteel onderzoek, toegepast onderzoek
Scope	Bijlage 11

Tabel 3.1: Onderzoeksmethode deelvraag 1

D2: Hoe kan een framework verschillende cloud computing platformen beheren om een machine learning pipeline op te zetten?	
Methode	Kwalitatief
Dataverzamelings methode(n)	Literatuuronderzoek, vergelijkend onderzoek
Scope	Tabel 11

Tabel 3.2: Onderzoeksmethode deelvraag 2

D3: Hoe ziet de architecturale blauwdruk van een applicatie, waarmee een platform-onafhankelijk machine learning pipeline opgezet kan worden, eruit?	
Methode	Kwalitatief
Dataverzamelings methode(n)	Literatuuronderzoek
Scope	Tabel 11

Tabel 3.3: Onderzoeksmethode deelvraag 3

H: In welke mate kan een machine learning pipeline worden geautomatiseerd onafhankelijk van de onderliggende cloud computing platform?	
Methode	Kwalitatief
Dataverzamelings methode(n)	Literatuuronderzoek
Scope	Tabel 11

Tabel 3.4: Onderzoeksmethode hoofdvraag

4 Stappen in een Machine Learning pipeline

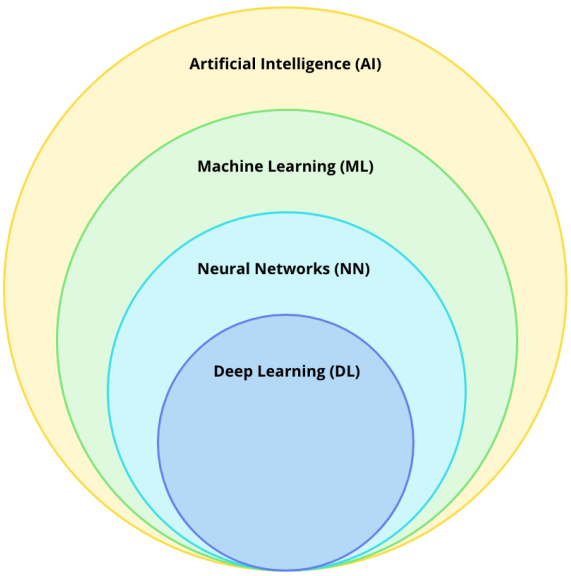
Een ML pipeline is zoals beknopt beschreven in deelparagraaf 2.1.2 een collectie van stappen dat wordt doorlopen om een model te trainen. Elke stap bevat een aantal acties dat wordt uitgevoerd, zoals onbruikbare data weghalen of de prestatie analyseren. De stappen en acties wordt in **??** uitgelegd. Een van de stappen in een pipeline is het trainen van het model. Om een idee te krijgen van ML en modellen trainen wordt dit kort uitgelegd.

4.1 Wat is machine learning?

ML houdt in dat een computer een taak kan uitvoeren zonder ervoor expliciet geprogrammeerd te zijn. Dit wordt gedaan door een ML model te laten leren van een gegeven dataset. Vervolgens kan er een voorspelling worden gemaakt [10, p. 1-3].

De domeinen deep learning (DL), neural networks (NN) en artificial intelligence (AI) komen vaak voor als het over ML gaat. Zoals weergegeven in Figuur 4.1 is te zien dat ML een subset is van AI, NN een subset van ML en als laatste DL dat een subset is van NN.

Bij AI wordt niet alleen ML toegepast, maar ook concepten zoals beredeneren, plannen, voor-uitdenken, onthouden en terug refereren. Een voorbeeld hiervan is dat een ML model kan voorspellen wat het volgende woord in een zin kan zijn, maar een AI kan beredeneren waarom de zin gebouwd is zoals het is en hoe het binnen de context van de alinea past [11].



Figuur 4.1: Machine learning in de context van andere domeinen

Een NN bestaat uit een collectie van nodes dat gemodelleerd is naar de hersenen. Een NN heeft minimaal 3 lagen: een input laag, een verborgen laag en een output laag. Elke laag bevat neuronen dat data als input kan krijgen en data als output aan de volgende laag meegeeft. DL is een NN dat meerdere verborgen lagen bevat [12].

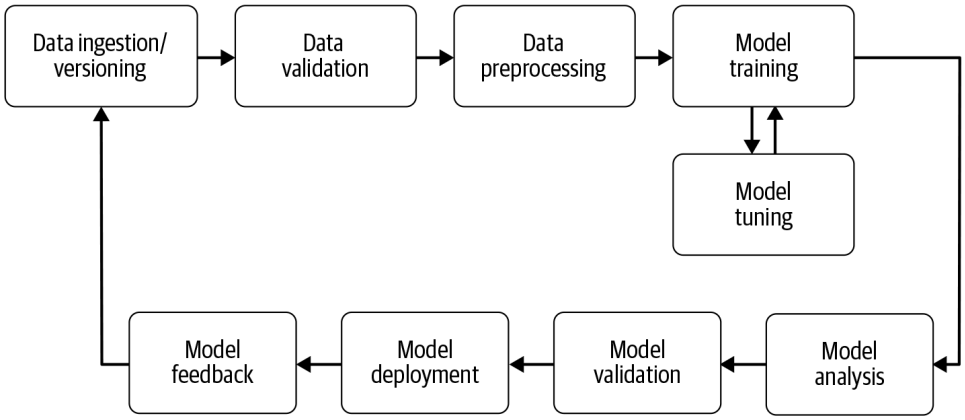
In het ML domein bestaan talloze algoritmes om voorspellingen te maken. In de [BIJLAGE LINK] is een mind-map te vinden van de algoritmes die gedurende de scriptie naar voren zijn gekomen. Het grotendeels kan gegroepeerd worden in vier stijlen: supervised, unsupervised,

semi-supervised en reinforcement learning. In de [BIJLAGE LINK] is meer informatie over de stijlen te vinden.

Het trainen van een model is een proces waarbij vooraf data wordt opgeschoond en achteraf de prestatie van het model wordt gevalideerd. Normaliter wordt dit gedaan door specifieke code te schrijven voor deze taken. Een valkuil is dat de code niet bij elke developer werkt en schaalt niet in alle gevallen naar een productie omgeving. Een manier om dit wel te behalen is om te werken met ML pipelines. Zoals kort uitgelegd in deelparagraaf 2.1.2 bestaat een ML pipeline uit stappen en acties. Er is echter geen consensus binnen het ML domein over wat de juiste stappen en acties in een pipeline zijn. Voor de scriptie is gekozen voor een pipeline van Hapke en Nelson uit het boek "Building Machine Learning Pipelines". Het boek is gemaakt en gepubliceerd door O'Reilly; een bekend en gecrediteerd bedrijf dat boeken maakt binnen het software engineering domein.

4.2 De stappen in een machine learning pipeline

Een ML pipeline begint met het opnemen van data en eindigt met het ontvangen van feedback om de prestatie van het model te verbeteren. De pipeline bevat een aantal stappen zoals data voorbereiden, het model trainen en het uitrollen van het model (Figuur 4.2). In totaal zijn er, zonder de feedback loop stap, acht stappen dat elke keer doorlopen moeten worden om een model te trainen.



Figuur 4.2: Lifecycle van een model volgens Hapke en Nelson [13, p. 4].

In de volgende subkoppen zullen de stappen worden doorlopen met een korte uitleg over wat er gebeurd in een stap. Om een praktisch beeld te schetsen wordt na de uitleg een ML probleem opgelost. Dit wordt gedaan met behulp van de iris dataset in [REFERENCE].

4.2.1 Stap 1: Data opname en versiebeheer (Data ingestion/versioning)

De eerste stap in de pipeline is het opnemen van data. Met deze data zal het model getraind, gevalideerd en getest worden. De dataset kan van een of meerdere bronnen komen, zoals lokaal, een online opslag locatie of van een database. Zodra de data is ingeladen, moet het verdeeld worden tussen een train, validatie en test dataset. Normaal gebeurt dit met een split ratio van

6:2:2. De train dataset is 60% en de validatie en test datasets zijn allebei 20% van de originele dataset [13, p. 27-37].

Een use case van een pipeline is dat een nieuw model getraind kan worden door een geüpdatet dataset te gebruiken. Dit wordt gedaan door de voorgaande dataset te gebruiken waarbij nieuwe data is toegevoegd. Door het gebruik van verschillende datasets is het verstandig om versiebeheer toe te passen. Zo is goed te zien welk dataset welk model produceert. Een versie geven aan een dataset gebeurt voordat de dataset wordt ingeladen [13, p. 39-40]. Versiebeheer voor datasets kan bijvoorbeeld met DVC [14] of Pachyderm [15].

4.2.2 Stap 2: Data validatie (Data validation)

Nu de dataset verdeeld is, een versie heeft en op een bereikbare plek is, kan de data gevalideerd worden. Deze stap is vooral belangrijk om te voorkomen dat een model wordt getraind dat niet nuttig is aangezien het trainen veel tijd in beslag kan nemen. Een bekende uitdrukking is "garbage in = garbage out". Dit betekent dat als de dataset niet goed is, het model ook niet goed zal presteren [13, p. 43]. Tijdens de validatie stap wordt gecontroleerd op het volgende:

- Afwijkingen in de dataset
- Wijzigingen in de structuur
- Algemene statistieken in vergelijkingen met voorgaand datasets [13, p. 44]

Bij het controleren van afwijkingen in de dataset wordt gekeken naar waardes die opmerkelijk zijn. Afwijkende waardes liggen te ver van het gemiddelde en kan een verkeerd beeld schetsen bij het trainen van het model. Deze uitschieters kunnen simpelweg uit de dataset gefilterd worden.

Het kan voorkomen dat bij een nieuw dataset de type van waardes zijn gewijzigd. Een *int* kan bijvoorbeeld veranderd zijn in een *string* of *boolean*. Er is dan sprake van een wijziging in de structuur van het dataset. Dit is problematisch omdat er een vertaalstap gemaakt moet worden naar iets bruikbaars. Als dit niet mogelijk is moeten de waardes uitgefilterd worden wat de prestatie van het model negatief kan beïnvloeden.

De algemene statistieken is een hulpmiddel om te controleren op afwijkingen en wijzigingen. Vaak kan de controle in een oogopslag gedaan worden.

4.2.3 Stap 3: Data voorbereiden (Data preprocessing)

Het voorbereiden van het dataset is een stap dat de prestatie van het model verbetert en het proces van het trainen versneld. Deze stap kan verdeeld worden in twee sub-stappen: het opschonen en het optimaliseren van het dataset.

Bij het opschonen worden bijvoorbeeld duplicaten of waardes die onbruikbaar zijn uit het dataset weggehaald. Onbruikbare waardes zijn waardes die simpelweg niet kloppen of verkeerd zijn ingevoerd. Hierbij kan bijvoorbeeld gedacht worden aan een medewerken die de interactietijd met een klant moet bijhouden, maar is vergeten om de eindtijd te noteren. Voor het algoritme zal het dan lijken alsof de medewerker een klant tot sluitingstijd heeft geholpen.

Datasets kunnen geoptimaliseerd worden om twee redenen: algoritmes werken sneller met waardes

die dichter bij 0 liggen en algoritmes kunnen niet met elke waarde in het dataset omgaan. Om de efficiëntie van het algoritme te verbeteren kunnen een aantal technieken gebruikt worden:

- Schalen

  Bij het schalen van data worden de waardes getransformeerd waardes dat ligt tussen een schaal, zoals tussen 0 en 100 of 0 en 1. Bij het schalen wordt het **bereik** getransformeerd [16].

- Normaliseren

  Als een dataset wordt gestandaardiseerd, worden de waarden getransformeerd in een standaard normale verdeling waarbij het gemiddelde 0 en de afwijking 1 is. Hierbij wordt de **vorm** van het dataset getransformeerd [16]. Het normaliseren is belangrijk als het algoritme waardes vergelijkt met verschillende eenheden [**feature-scaling-standardization**]. In sommige gevallen is normalisering een vereiste bij een aantal algoritmes [17].

- Categorisch codering

  Een groot aantal algoritmes kan alleen omgaan met numerieke waardes. Het kan voorkomen data datasets categorische waardes bevatten. Dit zijn waardes dat een label voorstellen, zoals *cat* of *dog*. Om deze waardes te gebruiken moeten ze worden gecodeerd als een numerieke waarde. De label kan als de waarde 1 of 2 gecodeerd worden. Deze techniek heet integer encoding. Een andere techniek om waardes te coderen heet hot encoding. Deze techniek maakt gebruikt van meerdere waardes om een label te beschrijven.

Na het opschonen en optimaliseren kan het dataset gebruikt worden om een model te trainen. Dit is een goed moment om het getransformeerde dataset op te slaan als "tussen-dataset"zodat deze stap niet herhaalt hoeft te worden. Het voorbereiden van grote datasets kan een grote hoeveelheid tijd in beslag nemen.

4.2.4 Stap 4 en 5: Model trainen en tunen (Model training and Model tuning)

Een ML algoritme is vergelijkbaar met een conventionele algoritme in software engineering. Het algoritme slaat na het trainen met een dataset de regels, nummers en algoritme specifieke data structuren op in de vorm van een model. Het model is als het ware een programma waarmee, gegeven een input, voorspellingen mee gedaan kan worden [https://machinelearningmastery.com/difference-between-algorithm-and-model-in-machine-learning/].

Een algoritme is geoptimaliseerd om een specifieke taak te doen, zoals het voorspellen van woorden of voorwerpen in een afbeelding herkennen. Een mindmap met algoritmes is te vinden in [BIJLAGE]. Zo is goed te zien hoeveel algoritmes hetzelfde bereiken. Hierom is het verstandig om vooraf te testen welke het beste werkt voor het gegeven ML probleem.

Over het algemeen gebeurt er hetzelfde, ongeacht welk algoritme is gebruikt [p84]:

- Laad de train en validatie dataset
- Definieer het model
- Train het model

- Sla het model op

[Hyperparameters uitleg]

[Uitleg hyperparameters SVC] [https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html]

4.2.5 Stap 6 en 7: Model analyse en validatie (Model analysis and Model validation)

De prestatie valideren en vergelijken met voorgaande modellen is het analyseren.

Ethische verantwoording van het model is validatie. Waarom het belangrijk is om het model te analyseren. Hoe behandeld het model voorspellingen van:

- mensen met verschillende variabelen zoals geslacht, locatie of leeftijd.
- image recognition met verschillende licht condities.

Meetinstrumenten voor algoritme groepen.

Voor voorbeeld kan classification metrics (confusion matrix) gebruikt worden.

4.2.6 Stap 8: Model uitrollen (Model deployment)

Het opgeslagen model kan simpelweg geladen worden in een webserver zoals Flask. Dit schaalt echter niet en het is lastig om bij te houden met welke versie van het model een voorspelling wordt gedaan.

Tensorflow Serving maakt dit makkelijker. Korte uitleg hoe TS Serving dit makkelijker maakt. Eventueel andere oplossingen

4.2.7 Stap 9: Feedback loop (Model feedback)

Impliciet en expliciet feedback

- Impliciet

  Een gebruiker kijkt een gesuggereerde film of koopt een gesuggereerde item in een webshop.

- Expliciet

  De applicatie vraagt aan de gebruiker of de suggestie waarde heeft.

[REFERENCE]

4.2.8 De machine learning pipeline toegepast op een probleem

- Om een beeld te geven van de ML pipeline wordt een ML probleem opgelost
- Het probleem is een bekend probleem binnen het ML domein omdat de focus op de pipeline ligt en niet of het ML probleem opgelost kan worden of niet
- Het probleem is gekozen omdat het een bekend probleem is binnen het ML domein en het doel is om de stappen in de ML pipeline te beproeven en niet om te experimenteren met onopgeloste problemen.

- Het doel is om een ML model te trainen dat, aan de hand van de afmetingen van de kelk- en bloemblad, kan identificeren tot welke familie een iris bloem hoort.
- Uitleg dataset Tabel 4.1

Kelkblad - lengte	**Kelkblad - breedte**	**Bloemblad - lengte**	**Bloemblad - breedte**	**Soort**
5.1	3.5	1.4	0.2	Iris-setosa

Tabel 4.1: Voorbeeld van de iris dataset

4.3 Conclusie

- Een ML pipeline is een manier om gestructureerd en reproduceerbaar een ML model te trainen
- Er is geen consensus over wat de stappen in een ML pipeline zijn
- Gedurende de pipeline wordt er gefocust op de kwaliteit van de dataset en prestatie van het model

4.4 Advies

- Het opzetten van een pipeline neemt tijd in beslag, maar het is het waard op de lange termijn
- Door de gestructureerde werkwijze is het goed bij te houden hoe het model tot stand is gekomen
-

5 Frameworks dat platformen beheert

6 Architecturale ontwerp van de oplossing

6.1 Literatuur

6.2 Design

6.3 Conclusie

7 Oplossing

7.1 Scope definiëren

7.2 Research techstack

7.3 Wireframe

7.4 Mockup

7.5 POC

7.6 Conclusie

8 Conclusie

9 Aanbeveling

10 Discussie

11 Reflectie

Bibliografie

[1] 22 mrt 2021. URL: https://www.ngti.nl/diensten/.

[2] 29 mrt 2021. URL: https://www.ngti.nl/oplossingen/.

[3] NGTI B.V. „Organogram".

[4] 29 mrt 2021. URL: https://www.swisscom.ch/en/about/beteiligungen-swisscom-uebersicht.html.

[5] 29 mrt 2021. URL: https://www.ngti.nl/cases/swiss-climate-challenge/.

[6] 29 mrt 2021. URL: https://www.swissclimatechallenge.ch.

[7] 29 mrt 2021. URL: https://www.ngti.nl/cases/my-swisscom-app/.

[8] 25 mrt 2021. URL: https://www.scribbr.nl/scriptie-structuur/methodologie-in-je-scriptie/.

[9] 25 mrt 2021. URL: https://www.scribbr.nl/onderzoeksmethoden/kwalitatief-vs-kwantitatief-onderzoek/.

[10] E. Alpaydin. *Introduction to Machine Learning*. 4de ed. 2020. ISBN: 9780262043793.

[11] 16 mrt 2021. URL: https://machinelearningmastery.com/think-machine-learning/.

[12] 16 mrt 2021. URL: https://wiki.pathmind.com/neural-network.

[13] H. Hapke en C. Nelson. *Building Machine Learning Pipelines*. 1ste ed. 2020. ISBN: 9781492053194.

[14] 6 apr 2021. URL: https://dvc.org.

[15] 6 apr 2021. URL: https://www.pachyderm.com.

[16] 27 mei 2021. URL: https://www.kaggle.com/rtatman/data-cleaning-challenge-scale-and-normalize-data.

[17] 27 mei 2021. URL: https://www.kdnuggets.com/2020/04/data-transformation-standardization-normalization.html.

Bijlagen

Scope hoofd- en deelvragen

Scope deelvraag 1

D1: Uit welke stappen bestaat een machine learning pipeline?	
Scope	Binnen de scope: • In kaart brengen uit welke stappen een pipeline bestaat • Acties die in een stap worden uitgevoerd • Of het mogelijk is om stappen te versimpelen / abstraheren voor developers • Of het mogelijk is om stappen en acties te automatiseren Buiten de scope: • Automatisering van stappen en acties • Een versimpeling van machine learning
Toelichting	Er wordt gekeken naar welke stappen er in een pipeline zitten. De theorie wordt vervolgens toegepast in een experiment. De nadruk ligt vooral of de mogelijkheid er is om stappen en acties te automatiseren en of machine learning versimpeld kan worden, niet dat er een uitwerking is.

Tabel 1: Scope deelvraag 1

Scope deelvraag 2

D2: Hoe kan een framework verschillende cloud computing platformen beheren om een machine learning pipeline op te zetten?	
Scope	Binnen de scope: • High-level uitleg over hoe een framework werkt • Inventarisatie met de "knock-out" methode • Criteria lijst voor frameworks • Opmerkelijke features die relevant zijn voor de PoC • Ervaring opdoen doormiddel van een pipeline te maken op twee cloud computing platformen Buiten de scope: • Performance en snelheid
Toelichting	Frameworks dat cloud computing platformen beheert worden in kaart gebracht. Met knock-out criteria wordt de lijst verkort. Met een framework wordt een pipeline opgezet.

Tabel 2: Scope deelvraag 2

Scope deelvraag 3

D3: Hoe ziet de architecturale blauwdruk van een applicatie, waarmee een platform-onafhankelijk machine learning pipeline opgezet kan worden, eruit?	
Scope	Binnen de scope: • Technische tekeningen Buiten de scope: -
Toelichting	De literatuuronderzoek slaat op of de technische tekeningen gemaakt zijn volgens een standaard zoals UML. Dit komt niet terug als theorie maar de bronnen worden wel vermeld.

Tabel 3: Scope deelvraag 3

Scope hoofdvraag

H: In welke mate kan een machine learning pipeline worden geautomatiseerd onafhankelijk van de onderliggende cloud computing platform?	
Scope	De scope wordt bepaald na de requirement analyse.
Toelichting	Onderzoek naar documentatie van gebruikte framework(s).

Tabel 4: Scope hoofdvraag

Criteria met toelichting voor cloud computing platformen

Criteria	**Toelichting**
ML Pipeline aanmaken	Het platform moet het aanmaken van een machine learning pipeline kunnen ondersteunen.
Serverbeheer	Aanmaken, wijzigen en verwijderen van een server. Een server kunnen aanmaken bij het platform is praktisch omdat een server voor meerdere doeleinde gebruikt kan worden.
Database beheer	Aanmaken, wijzigen en verwijderen van een database. De PoC moet data ergens kunnen opslaan en vandaan halen; dit is mogelijk in een database.
Storage bucket beheer	Aanmaken, wijzigen en verwijderen van een storage bucket. In een storage bucket kan grote hoeveelheden data opgeslagen worden zoals bestanden en afbeeldingen. Dit kan handig zijn om bijvoorbeeld train en test data op te slaan.
Uptime 99.9%	Het platform moet een vorm van uptime kunnen garanderen waarbij het percentage zo dicht mogelijk bij 99.9 ligt. Hierdoor is de kans dat NGTI door een storing bij een platform niet productief kan zijn zo klein mogelijk.
Regionale beschikbaarheid	Waar data wordt opgeslagen en de servers draaien is vrij belangrijk. Het platform moet ten minste West-Europa ondersteunen in verband met de gegevensbescherming in de EU (GDPR). In een ideale situatie zou het platform ook specifiek Zwitserland ondersteunen aangezien de meeste klanten van NGTI vandaar komen.
Toegankelijkheid documentatie ML pipeline	In het geval dat de PoC wordt uitgebreid tot een applicatie is het belangrijk om onderhoud uit te voeren en eventueel nieuwe functies toe te voegen. Het is daarom van belang dat het platform documentatie heeft over ML pipelines.
APIs	Om het platform programmatisch aan te sturen zijn APIs dat het platform beschikbaar stelt onmisbaar. Op deze manier kunnen frameworks het platform beheren en kan eventueel een op maat gemaakte oplossing gebruikt worden.
Inhoud ML	Voor NGTI is het belangrijk dat het platform niet alleen ondersteuning heeft om modellen te trainen, maar meer mogelijkheden biedt zoals het schrijven van een eigen algoritme of out-of-the-box oplossingen.

Tabel 5: Criteria voor cloud computing platformen

Gedetailleerd overzicht cloud computing platformen

AWS

Criteria	Toelichting
ML Pipeline aanmaken	Met Amazon SageMaker kunnen pipelines gemaakt worden. Daarnaast heeft Amazon specifieke services om bijvoorbeeld vooroordelen te detecteren, statistieken te meten en data te verzamelen (overzicht).
Serverbeheer	Met Amazon EC2 kunnen servers worden opgestart, gewijzigd of verwijderd worden. Een overzicht wat Amazon EC2 allemaal kan
Database beheer	Amazon biedt een database service aan waarbij een database aangemaakt, gewijzigd of verwijderd kan worden. De databases zijn gesorteerd op type in dit overzicht.
Storage bucket beheer	Wat betreft opslag biedt Amazon twee soorten aan: Amazon S3 en Amazon Elastic Block Store. Het verschil is dat S3 data opslaat als een object ten opzicht van Elastic Block Storage, dat data op de conventionele manier opslaat.
Uptime 99.9%	De uptime staat beschreven in de Service Level Agreement (SLA) en is voor elke service anders. Over het algemeen biedt Amazon voor elke service een uptime van 99.9%: Amazon SageMaker, Amazon EC2 en Elastic Block Store, Amazon Relational Databases en Amazon S3.
Regionale beschikbaarheid	Amazon heeft in totaal zes fysieke datacenters in Europa (overzicht). Ook is het mogelijk om een specifiek regio te kiezen zoals het hier beschreven staat. Helaas is Zwitserland geen beschikbare regio.
Toegankelijkheid documentatie ML pipeline	Amazon heeft een uitgebreide services voor ML binnen AWS. Alle services zijn te vinden in dit (overzicht).
APIs	Voor al haar services stelt Amazon een API beschikbaar. Een overzicht voor de documentatie is hier te vinden. API documentatie voor ML en SageMaker is ook beschikbaar.
Inhoud ML	Amazon heeft een groot aantal ML specifieke services waar gebruik van gemaakt kan worden. Een paar voorbeelden daarvan is Amazon Translate, Amazon Healthlake en Amazon Forecast.

Tabel 6: AWS tegenover criteria op 06-04-2021

Azure

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 7: Azure tegenover criteria op 06-04-2021

Google Cloud

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 8: Google Cloud tegenover criteria op 06-04-2021

DigitalOcean

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 9: DigitalOcean tegenover criteria op 06-04-2021

IBM Cloud

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 10: IBM Cloud tegenover criteria op 06-04-2021

Alibaba

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 11: Alibaba tegenover criteria op 06-04-2021

Oracle Cloud Infrastructure

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 12: Oracle Cloud Infrastructure tegenover criteria op 06-04-2021

Kamatera Cloud

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 13: Kamatera Cloud tegenover criteria op 06-04-2021

Cloudways

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 14: Cloudways tegenover criteria op 06-04-2021

Vultr

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 15: Vultr tegenover criteria op 06-04-2021

BigML Inc.

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 16: BigML Inc. tegenover criteria op 06-04-2021

H2O.ai Inc.

Criteria	**Toelichting**
ML Pipeline aanmaken	
Serverbeheer	
Database beheer	
Storage bucket beheer	
Uptime 99.9%	
Regionale beschikbaarheid	
Toegankelijkheid documentatie ML pipeline	
APIs	
Inhoud ML	

Tabel 17: H2O.ai Inc. tegenover criteria op 06-04-2021