HO
GENT

Faculteit Bedrijf en Organisatie

De invloed van artificiële intelligentie op de user experience van webapplicaties

Stefaan De Vylder

Scriptie voorgedragen tot het bekomen van de graad van
professionele bachelor in de toegepaste informatica

Promotor:
Lieven Smits
Co-promotor:
Joren Saey

Instelling: —

Academiejaar: 2021-2022

Tweede examenperiode

Faculteit Bedrijf en Organisatie

De invloed van artificiële intelligentie op de user experience van webapplicaties

Stefaan De Vylder

Scriptie voorgedragen tot het bekomen van de graad van
professionele bachelor in de toegepaste informatica

Promotor:
Lieven Smits
Co-promotor:
Joren Saey

Instelling: —

Academiejaar: 2021-2022

Tweede examenperiode

Woord vooraf

Deze bachelorproef kwam tot stand in het kader van de opleiding “Toegepaste informatica“ aan de hogeschool Gent, en strekt ertoe het diploma te behalen.

Het creëren van deze bachelorproef was een zeer uitgebreid werk. Desondanks ben ik wel blij dat ik deze kans heb gekregen om een onderzoek te doen tijdens mijn studies. Aan de hand van dit onderzoek heb ik meer inzicht gekregen in de termen artificiële intelligentie (AI), machine learning (ML) en neurale netwerken. Toen ik de opdracht kreeg om deze bachelorproef te schrijven, was ik niet zeker welke richting ik van plan was uit te gaan. Er zijn veel verschillende en complexe onderwerpen binnen informatica die mijn aandacht trekken. Het was een moeilijke keuze, maar ik ben zeer blij dat ik dit pad gekozen heb. Artificiële intelligentie leunt dicht aan bij mijn interesses en het was dan ook heel interessant om mij de afgelopen maanden hierin te verdiepen.

Allereerst wil ik graag mijn vriendin, familie en vriendenkring bedanken om mij telkens opnieuw moed en energie te geven als ik dit het nodig had.

Vervolgens zou ik zeker Joren Saey willen bedanken voor zijn kostbare tijd en nieuwe inzichten in deze bachelorproef. Zonder Joren was dit zeker niet mogelijk geweest.

Tot slot zou ik graag Lieven Smits willen bedanken voor de verdere correcte afhandeling van mijn bachelorproef, de talloze interessante lessen over Probleem Oplossend Denken (POD 1 en 2!) en natuurlijk artificiële intelligentie, machine learning en Classic Computer Science Algorithms!

Samenvatting

Webapplicaties worden complexer, dit maakt de user experience (UX) steeds belangrijker. In deze studie wordt bekeken hoe er aan de hand van artificiële intelligentie (AI) en machine learning (ML) verbeteringen kunnen worden gemaakt aan de user experience. De user experience kan op verschillende manieren verbeterd worden, een optie dat hier bekeken werd, was de user flow. Webapplicaties worden dagelijks gebruikt op verschillende manieren. Bijvoorbeeld een facturatie-/CRM-programma wordt dagelijks gebruikt om verschillende acties uit te voeren op verschillende momenten van de dag. Hierna is er bekeken om suggesties te genereren over welke actie de gebruiker vervolgens wilt uitvoeren. Er werd gevonden dat aan de hand van het suggereren van acties binnen de webapplicatie, de user flow sneller en efficiënter kan worden gemaakt. Om deze suggesties te berekenen werd er gebruik gemaakt van het Tensorflow JS framework. Dit laat toe om machine learning (ML) makkelijk te implementeren in een front-end framework zoals React. In deze studie werd er ook gekeken hoe Tensorflow JS concreet kan worden geïmplementeerd aan de hand van een proof-of-concept. In dit proof-of-concept werd het KNN Classifier model van de Tensorflow open-source community gebruikt. Dit model gebruikt onderliggend het K-means Clustering algoritme. Hierdoor konden er suggesties worden gegeven aan de gebruiker op basis van tijd, welke actie de gebruiker heeft ondernemen en eventuele andere input parameters. Er is geconcludeerd, aan de hand van dit proof-of-concept en de voorgaande studie, dat er zeker een verbetering is gemaakt in de user flow. Er kan dan ook gezegd worden dat dit proof-of-concept, webapplicatie ontwikkelaars de mogelijkheid biedt om een concreter beeld te schetsen hoe artificiële intelligentie en machine learning te implementeren valt in een bestaande of nieuwe webapplicatie. Als laatste wordt er een aanzet gegeven naar volgende studies die implementatie mogelijkheden van machine learning in mobiele applicaties kunnen bekijken.

Inhoudsopgave

1 Inleiding 15

1.1 Probleemstelling 15

1.2 Onderzoeksvraag 16

1.3 Onderzoeksdoelstelling 16

1.4 Opzet van deze bachelorproef 16

2 Stand van zaken 19

2.1 Artificial intelligence (AI) 19

2.2 Machine Learning (ML) 20

2.2.1 Natural Language Processing (NLP) 22

2.2.2 Planning, scheduling and optimization 22

2.2.3 Vision 22

2.2.4 Deep learning - Neuraal netwerk (Neural network) 23

2.2.5 Neuronen 24
2.2.6 Activatie functies (Activation functions) 24
2.3 Tensorflow 25
2.3.1 WebGL 25
2.3.2 KNN Classifier 25
2.3.3 ML5 26
2.4 Torch 26
2.5 Keras 26
2.6 User experience (UX) 26
2.7 De user flow 27
3 Methodologie 29
3.1 Fase 1 29
3.2 Fase 2 30
3.3 Fase 3 30
4 Suggestieve acties 33
4.1 Het doel 33
4.2 Requirements analyse 34
4.2.1 Input en output requirements 34
4.2.2 Must have's 34
4.2.3 Should have's 35
4.2.4 Nice to have 35
4.3 Gekozen algoritme 35

4.4 Datasets 35
4.4.1 GDPR 36
5 Proof-of-concept 37
5.1 Het proof-of-concept 37
5.2 Gebruik 37
5.3 Tensorflow 39
6 Conclusie 41
A Onderzoeksvoorstel 43
A.1 Introductie 43
A.2 State-of-the-art 43
A.2.1 Wat behoort tot UX 43
A.2.2 Het AI probleem 44
A.3 Methodologie 44
A.4 Verwachte resultaten 44
A.5 Verwachte conclusies 45
B Bijlagen 47
B.1 Broncode proof-of-concept 47
Bibliografie 55

Lijst van figuren

2.1 Verschillende AI onderwerpen Kar (2017) 20
2.2 K-means clustering algoritme 21
2.3 Representatie van een neuraal netwerk 23
2.4 Een neuron in een neuraal netwerk 24
2.5 Activatiefuncties: Sigmoid en ReLU 25

3.1 Top frontend frameworks 30

5.1 Proof-of-concept stap 1 38
5.2 Proof-of-concept 40

Lijst van tabellen

2.1 Tensorflow API's 28

1. Inleiding

Met de huidige transformatie van de online wereld zijn er steeds meer ondernemers die een probleem willen oplossen met een online applicatie of platform. De problemen worden steeds op een efficiënte wijze opgelost, maar toch blijven dezelfde complicaties opduiken. Elke gebruiker heeft een eigen beeld over elke webapplicatie. De gebruiker zou graag het zo makkelijk en efficiënt mogelijk willen maken voor zichzelf, maar als al deze zaken worden gecombineerd, voor meerdere gebruikers tegelijkertijd, ontstaan er zeker conflicten op vlak van layout en personalisatie. Het optimale punt tussen gebruiksvriendelijkheid, personalisatie en design is dan soms ver te zoeken. Deze studie zal, aan de hand van artificiële intelligentie (AI), data gaan verzamelen van een gebruiker om een webapplicatie zo makkelijk en persoonlijk mogelijk te maken. Hierdoor voelt de applicatie voor elke gebruiker weer iets meer gepersonaliseerd. Niet elke functie van het systeem wordt door elke gebruiker gebruikt, daarom wordt er bekeken om sommige functies te prioriteren, suggereren, eventueel onzichtbaar te maken of verbeteren aan de hand van AI.

1.1 Probleemstelling

Bedrijven en gebruikers willen steeds meer dat een webapplicatie op maat gemaakt is. Gebruikers zijn het reeds gewoon gemaakt om zo snel mogelijk te kunnen werken met een bepaalde webapplicatie. Maar hoe wordt dit nu bereikt? Het antwoord hierop is personalisatie. Een webapplicatie dient persoonlijker te worden gemaakt zodat de gebruiker beter zijn of haar weg kent. Echter meestal wenst de gebruiker geen persoonlijke informatie vrij te geven over zichzelf, dan wordt het natuurlijk moeilijker om personalisatie toe te passen. Aan de hand van het opslaan van vorige acties en AI kunnen gebruiker gegevens

verwerkt worden zonder dat de gebruiker directe input moet geven. Bijvoorbeeld klik of scroll gedrag van de gebruiker verteld al veel.

Dit onderzoek zal bruikbaar zijn voor SaaS (Software-as-a-Service) bedrijven die een webapplicatie hebben waarbij de gebruikerservaring een belangrijk punt is. Het is ook enkel voor bedrijven die heel wat features hebben waardoor de gebruiker soms het overzicht kan verliezen. Hiernaast is dit onderzoek ook bruikbaar voor bedrijven die de gebruikerservaring willen verbeteren aan de hand van layout.

1.2 Onderzoeksvraag

Er zijn talloze opties waarbij de gebruikerservaring van een webapplicatie kan worden verbeterd met artificiële intelligentie (AI). Hoe wordt de user experience (gebruikerservaring, UX) van webapplicaties verbeterd met artificiële intelligentie (AI)?

Verder specifiek zijn er nog een paar deelvragen:

- Welke aspecten van UX valt te verbeteren met AI?
- Kan AI gebruikt worden in front-end of enkel in back-end?
- Wat voor data heeft de AI nodig en wat voor data mag worden bijgehouden volgens de europese GDPR wetgeving?

1.3 Onderzoeksdoelstelling

Het doel van dit onderzoek is om een beeld te schetsen waar en wanneer er artificiële intelligentie (AI) valt te implementeren in een webapplicatie om de user experience (gebruikerservaring, UX) te verbeteren. Dit heeft als gevolg dat een ontwikkelaar van een webapplicatie, na deze studie, een concreter beeld krijgt om artificiële intelligentie (AI) te integreren in de applicatie. Er zal een proof-of-concept zijn om inspiratie uit op te nemen.

1.4 Opzet van deze bachelorproef

De rest van deze bachelorproef is als volgt opgebouwd:

In Hoofdstuk 2 wordt een overzicht gegeven van de stand van zaken binnen het onderzoeksdomein, op basis van een literatuurstudie.

In Hoofdstuk 3 wordt de methodologie toegelicht en worden de gebruikte onderzoekstechnieken besproken om een antwoord te kunnen formuleren op de onderzoeksvragen.

In Hoofdstuk 4 wordt er bekeken of het mogelijk is om artificiële intelligentie (AI) en machine learning (ML) te gebruiken om suggesties te maken voor gebruikers in een we-

bapplicatie.

In Hoofdstuk 5 wordt er een proof-of-concept gemaakt om machine learning (ML) makkelijk te integreren in front-end webapplicaties.

In Hoofdstuk 6, tenslotte, wordt de conclusie gegeven en een antwoord geformuleerd op de onderzoeksvragen. Daarbij wordt ook een aanzet gegeven voor toekomstig onderzoek binnen dit domein.

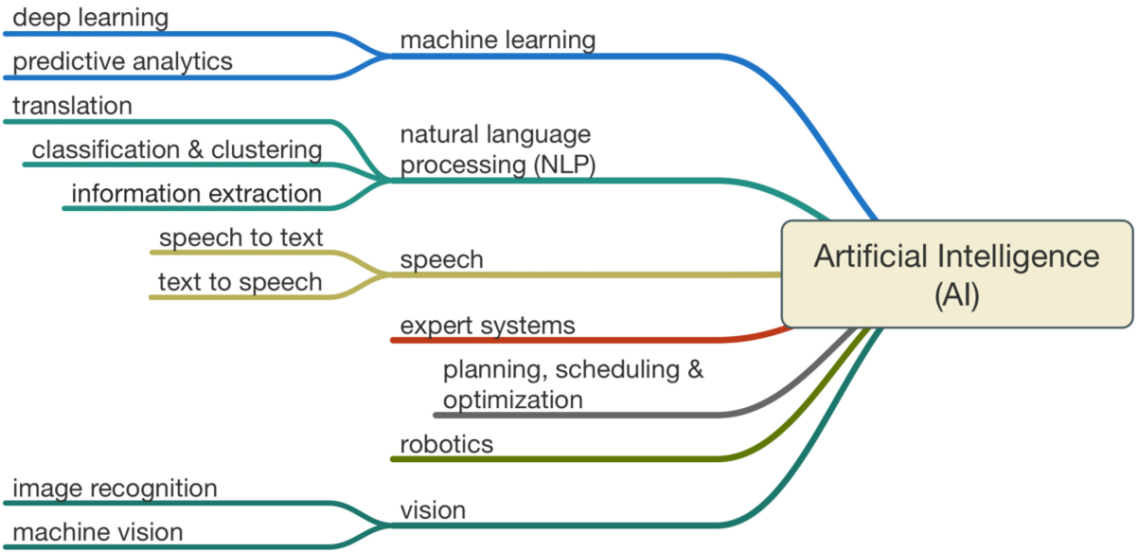
2. Stand van zaken

Voor er van start kan worden gegaan met dataverwerking, AI en andere besproken onderwerpen, zal er eerst wat meer informatie moeten worden gegeven over wat alles eigenlijk is. Bijvoorbeeld artificiële intelligentie (AI) heeft verschillende sub-onderwerpen. Ook voor font-end en back-end bestaan er manieren en frameworks om AI te implementeren. Ten laatste wordt er gekeken naar verschillende test scenario's en frameworks waarbij er AI kan worden geïmplementeerd. Dit allemaal met de user-experience (UX) of de gebruikerservaring als primair verbeteringspunt.

2.1 Artificial intelligence (AI)

Artificiële intelligentie (AI) of ook wel kunstmatige intelligentie genoemd is simpel gezegd een algoritme dat menselijke intelligentie probeert na te bootsten om taken uit te voeren die zichzelf tijdens dat proces kan verbeteren aan de hand van vorige informatie (Oracle, g.d.-a). Er bestaan verschillende sub-onderwerpen van (Een mind-map van de verschillende onderwerpen kan u zien in figuur 2.1). Voorbeelden van deze onderwerpen zijn (Sophia, g.d.):

- Machine learning (ML): Wordt gebruikt bij toepassingen zoals zelfrijdende auto's.
- Natural language processing (NPL): Wordt gebruikt voor vertaling software zoals Google Translate of DeepL
- Speech: Wordt gebruikt bij toepassingen zoals Siri of Google Home
- Planning, scheduling and optimization: Toepassingen zoals Facebook postplanner
- Robotics: Het namaken van een persoon op basis van verschillende karakteristieken



Figuur 2.1: Verschillende AI onderwerpen Kar (2017)

U zal zien dat onderwerpen bij AI zullen overlopen in elkaar, dit is omdat er algoritmes zijn die delen van andere algoritmes gebruiken. Er zullen verscheidene relevante onderwerp bekeken en geanalyseerd worden om te zien wat het beste past voor deze studie.

2.2 Machine Learning (ML)

Machine learning (ML) is een vorm van artificiële intelligentie (AI) dat gericht is op het bouwen van systemen die van de verwerkte data kunnen leren of data gebruiken om beter te presteren (Oracle, g.d.-b). De termen artificiële intelligentie (AI) en machine learning (ML) worden vaak door elkaar gebruikt, terwijl dit niet hetzelfde is. Artificiële intelligentie (AI) is de overkoepelende term waaronder machine learning (ML) valt. Er bestaan verschillende soorten van machine learning (ML):

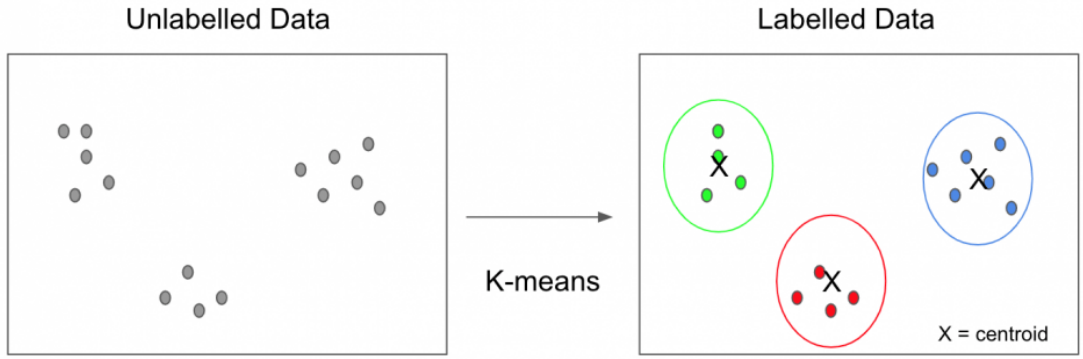
- Supervised machine learning
- Unsupervised machine learning

Supervised machine learning is de meest gebruikte vorm van machine learning (ML) tot nu toe. Dit werkt met een tussenpersoon, de AI wordt getraind door een mens, deze gaat labelen wat 'juist' of 'fout' is. Dit kan gedaan worden aan de hand van grote datasets maar vereist veel manueel werk (Oracle, g.d.-b).

Bij unsupervised machine learning zal de computer op zichzelf patronen gaan herkennen. Het werkt zonder dat er initieel een output wordt gedefinieerd. Dit model zorgt ervoor dat er meer complexe taken kunnen worden uitgevoerd. Clustering is een belangrijk begrip bij unsupervised learning algoritmes. Deze algoritmes zullen zich bezig houden met de data te groeperen (clusteren) om hier conclusies uit te trekken. Het is ook mogelijk om aan te passen hoeveel clusters het algoritme maakt, zo past u de granulariteit van het algoritme

aan (Johnson, 2022). Er bestaan verschillende unsupervised machine learning algoritmes, de meest gebruikte zijn:

- K-means clustering: De datapunten kunnen we op een grafiek plotten, hieruit worden de verschillende groepen geclusterd. Door gebruik van een parameter moet het aantal groepen eerst worden vastgelegd. Hierna pakt het algoritme het centrum (centroid) van de groep en bakent de randen af (DataScienceTeam, 2020).



Figuur 2.2: K-means clustering algoritme (DataScienceTeam, 2020)

- Hierarchical clustering: Dit algoritme werkt door data te groeperen (clusteren) in 3 clusters. Het werkt door elk data punt te behandelen als andere clusters. Vervolgens zal het 2 stappen blijven herhalen: 1. Vind de twee clusters die het dichts bijeen liggen. 2. Voeg de twee clusters tesamen. Herhaal dit tot alle clusters samengevoegd zijn. Vorige twee beschreven stappen heet Agglomerative Hierarchical clustering. Hiernaast bestaat het algoritme ook in een tweede vorm: Divisive Hierarchical clustering. Het algoritme zal de stappen herhalen maar in omgekeerde volgorde. De cluster zal worden opgedeeld in kleine sub-clusters (GeekForGeeks, 2020).

Een goede methode van machine learning (ML) hangt af of de behoeften van het doel meer leunen naar supervised machine learning of naar unsupervised machine learning (Oracle, g.d.-b). Hierna moet er bijvoorbeeld bij het K-means clustering algoritme nog andere input parameters bepaald worden, zoals hoeveel clusters er gemaakt worden. Voor dit onderzoek, mocht er worden gekozen voor machine learning, zal dit unsupervised learning zijn. Dit komt doordat er op een webapplicatie zo veel gebruikers zijn, er niet voor elke gebruiker manuele input kan worden gegeven. Tenzij dit aan de hand van A/B testen of een andere methode wordt vastgelegd per gebruiker.

2.2.1 Natural Language Processing (NLP)

Natural Language Processing (NLP) is een algoritme dat probeert om tekst of spraak data te verstaan. NLP combineert taal regels, gebaseerd op mensentaal, met statistische, machine learning (ML) deep learning modellen. Als deze modellen worden samengevoegd, laat dit een computer, tekst of spraak data behandelen en een response op genereren (IBM-CloudEductation, 2020). Onder NLP vallen verschillende onderdelen zoals:

- Vertaling software
- Machine learning (ML)
- Speech-to-text
- Text-to-speech

NLP is in dit onderzoek minder van toepassing maar zal zeker niet uitgesloten worden om het gebruik hiervan te analyseren.

2.2.2 Planning, scheduling and optimization

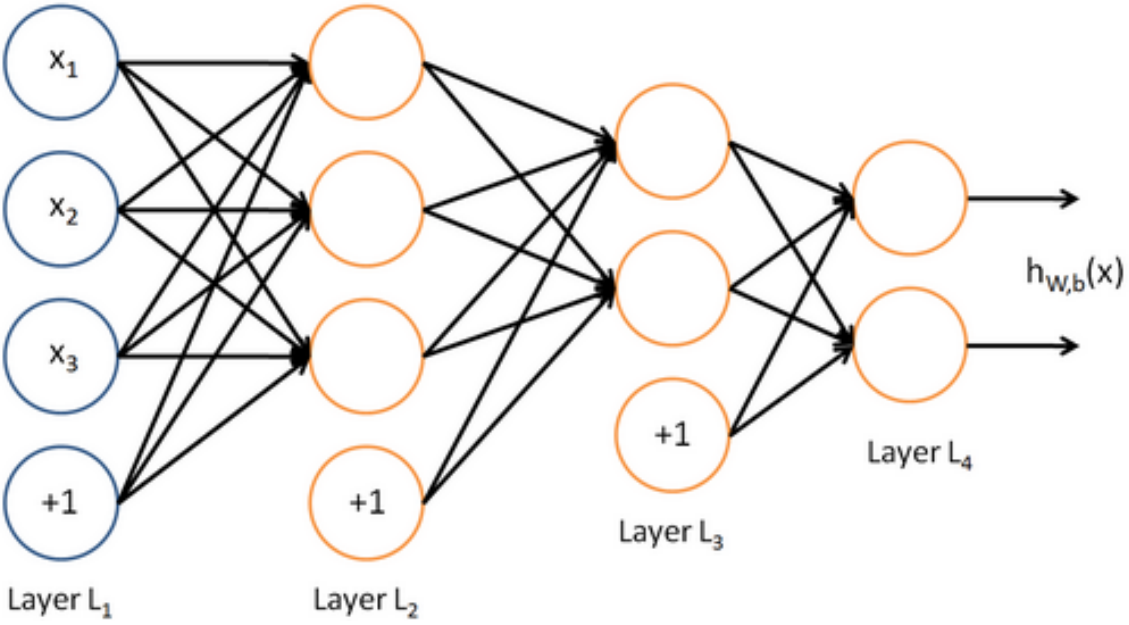
Een systeem gebaseerd op AI gebruikt planning om te bepalen welke stappen te nemen (planning) en wanneer een bepaalde stap uit te voeren (scheduling). Het planning proces bestaat uit het volgende (Adler, 2019):

- Een initiële toestand (Initial state)
- Een doel toestand (Goal state)
- Bepaalde acties gedefinieerd in een domein

Een plan zal een AI ervoor doen zorgen bovenstaande stappen uit te voeren. Een voorbeeld hiervan is een food-delivery robot. De initiële toestand kan zijn: de robot die een bestelling heeft staan om af te leveren bij een klant. Een doel toestand kan dan zijn: de klant die de bestelling heeft ontvangen. De bepaalde acties kunnen dan '100 meter naar voor', 'draai 180 graden' zijn. Veel modellen werden gemaakt om niet afhankelijk te zijn van een bepaald domein en kunnen dus worden ingezet in verschillende domeinen. Dit zorgt ervoor dat bedrijven AI sneller kunnen implementeren en niet alles van 0 moeten creëren. De gebruikelijke term dat wordt gegeven aan een systeem dat AI planning implementeert, is een agent. Er zijn twee soorten agenten, namelijk software en fysieke agenten. Het belangrijkste voor een agent is dat deze beslissingen voor zichzelf kan maken en intelligent genoeg is om zich te begeven in onvoorspelbare omgevingen. De uitvoerder moet de doelen dan ingeven aan een agent zonder dat er wordt gekeken over hoe dit doel wordt bereikt (Adler, 2019).

2.2.3 Vision

Vision is een onderdeel van AI dat toelaat om een computer beslissingen te maken gebaseerd op input zoals een afbeelding, video's of andere visuele input (IBM, g.d.-b). Een voorbeeld is de Google Vision AI. Hierbij kan er gekozen worden tussen twee opties:



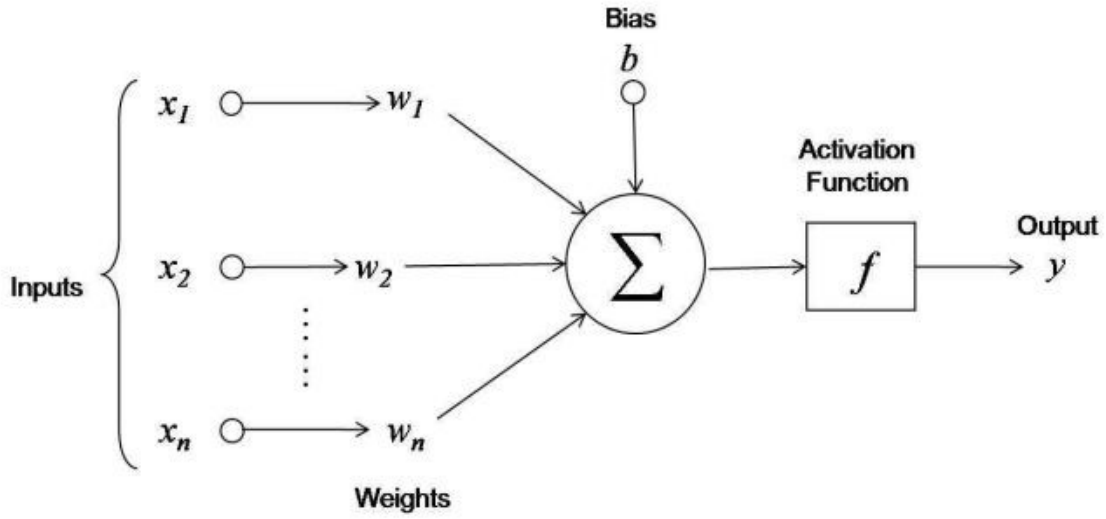
Figuur 2.3: Representatie van een neuraal netwerk (Arnx, 2019)

AutoML of een voorbereid Vision AI. Met AutoML kan er makkelijk input worden gegeven om een model te trainen. Bijvoorbeeld, u upload 500 foto's van een nieuwe ras kat, dan zal de AI dit altijd herkennen. De voorbereide Vision AI bevat meer voorgedefinieerde labels die de AI zal herkennen, zoals logo's van bedrijven of gezichtsuitdrukkingen (Google, g.d.).

Computer vision probeert een mensenvisie zo hard mogelijk na te bootsen. Een mens kan makkelijk objecten differentiëren en analyseren hoe ver, hoe zwaar of hoe snel een object is of gaat in een afbeelding. Met machine learning (ML) en een neuraal netwerk wordt dit dus mogelijk voor een computer. Machine learning gebruikt modellen om een computer zichzelf meer te leren over de context van de data. Als er genoeg data beschikbaar is, kan er een neuraal netwerk worden ingezet om de data de differentiëren (IBM, g.d.-b).

2.2.4 Deep learning - Neuraal netwerk (Neural network)

Deep learning is een type van machine learning. Dit gebruikt een neuraal netwerk om eigenschappen toe te kennen aan een bepaald object. Een neuraal netwerk is gebaseerd op de natuur en probeert een representatie te creëren van een brein, gelijkaardig met neuronen (Arnx, 2019). Een neuraal netwerk heeft een eenvoudig stappenplan, namelijk geef een input, na bepaalde calculaties moet dit dan een output geven. Bijvoorbeeld: geef een foto van een boot in, het neuraal netwerk geeft de tekst 'boot' terug. Een neuraal netwerk bestaat uit verschillende lagen en kolommen (Bekijk figuur 2.3). Elke neuron is geconnecteerd met verschillende andere neuronen van de vorige en volgende laag. Neurale netwerken kunnen meestal van links naar rechts worden gelezen. Bij figuur 2.3 zijn er 2 tussenlagen (hidden layers) die de calculaties zullen uitvoeren (Arnx, 2019).



Figuur 2.4: Een neuron in een neuraal netwerk (Arnx, 2019)

2.2.5 Neuronen

Per neuron worden er verschillende calculaties uitgevoerd. In figuur 2.4 krijgt de gegeven neuron een aantal inputs (x1, x2...xn). Deze inputs kunnen van vorige neuronen komen of van de initiële input. Elke input variabele wordt dan vermenigvuldigd met het gewicht (de weight). Het gewicht is een variabele dat de connectie indiceert tussen twee neuronen. Deze gewichten (weights) kunnen worden aangepast door het neuraal netwerk tijdens het volledige leer proces. Hierna wordt de som genomen van alle inkomende waarden en kan eventueel een 'bias' worden toegevoegd. Achter deze calculaties wordt de uitkomst geplaatst op een activatie functie (activation function) (Arnx, 2019).

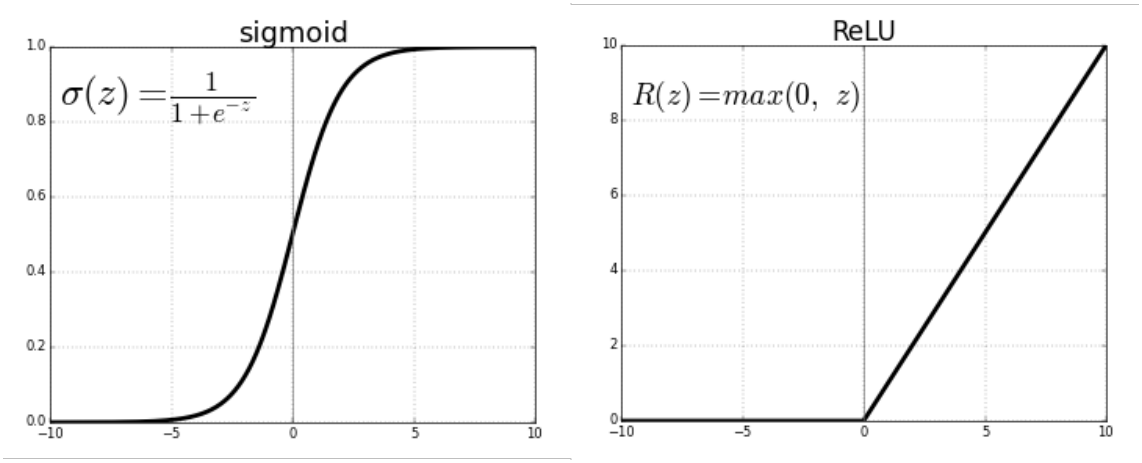
2.2.6 Activatie functies (Activation functions)

Er zijn verschillende activatie functies. Dit zijn wiskunde functies of formules die worden geplot op een grafiek. De meest gebruikte functies zijn (Sharma, 2017):

- Lineaire functie
- Niet-lineaire functie
- Sigmoid of logistische functie
- ReLU functie

De uitkomst van deze functies zorgen ervoor dat de uitkomst van het neuraal netwerk kan worden afgebeeld in een getal tussen 0 en 1. Andere functies kunnen de limieten veranderen maar het doel blijft hetzelfde.

Een neuron heeft dus een simpel stappenplan, neem alle waarden van vorige neurons vermenigvuldigd met de respectievelijke gewicht (weight), sommeer deze waarden en pas de activatie functie toe. Het leren van een neuraal netwerk gebeurd dus als volgt. We moeten



Figuur 2.5: De Sigmoid activatie functie en de ReLU activatie functie (Sharma, 2017)

onthouden dat een neuraal netwerk een gewilde output heeft. Maar om de juiste output te halen moeten we in het begin een label meegeven om te zeggen wat de gewenste output is. Hierdoor kan het neuraal netwerk dit herhalen en telkens de gewichten (weights) aanpassen tot dit in de buurt zijn van de gewenste output. De gewichten (weights) aanpassen en bekijken welk gewicht het beste is noemt 'backpropogation' (Arnx, 2019).

2.3 Tensorflow

Om machine learning concreet in neurale netwerken toe te passen, dient er gebruik te worden gemaakt van een framework. Voor dit onderzoek is er gekozen voor het Tensorflow framework. Meer specifiek het Javascript gedeelte van Tensorflow, genaamd Tensorflow JS 2.0. Dit is een machine learning framework gemaakt om in een browser uitgevoerd te worden. Tensorflow JS gebruikt C++ code dat herschreven is in Javascript, gebruikmakend van WebGL om via de GPU deze calculaties sneller te kunnen maken (Tensorflow, g.d.-c). Tensorflow stelt verschillende API's ter beschikking dat gebruikt kan worden door een developer. De huidige Tensorflow API's zijn te vinden in tabel 2.1 (Tensorflow, g.d.-b).

2.3.1 WebGL

Aan de hand van WebGL (Web Graphics Library) is het mogelijk om de computaties op de GPU uit te voeren in plaats van op de CPU. Dit is een Javascript API om de GPU aan te spreken vanuit de browser (Mozilla, g.d.-b).

2.3.2 KNN Classifier

De KNN Classifier package bevat een implmentatie van het K-means clustering algoritme. Deze API is een extensie op de huidige Tensorflow API en bevat makkelijk te gebruiken

functies om het algoritme toe te passen (Tensorflow, g.d.-a). De package kan worden gevonden onder de getrainde modellen dat in de Tensorflow open source repository staat, te vinden op GitHub.

2.3.3 ML5

ML5 is een framework dat Tensorflow JS onderliggend gebruikt. Het doet zich voor als een makkelijkere oplossing voor machine learning (ML) te gebruiken in de browser. Dit framework is geschreven en te gebruiken in Javascript. De reden dat ML5 gekomen is, is omdat het ML wilt leren kennen aan een breder publiek. Het wilt ook ML toegankelijker maken. Dit framework heeft tientallen getrainde modellen die gebruikt kunnen worden door iedereen (ML5, g.d.). ML5 is gericht op machine learning met afbeeldingen of een live camera.

2.4 Torch

Torch, of ook wel PyTorch genoemd, is een open source framework voor machine learning (ML) te implementeren. Dit framework kan niet gebruikt worden in een front-end applicatie zondig enige back-end connectie. Dit is een API dat kan worden geïmplementeerd in een backend applicatie. PyTorch is beschikbaar voor Linux, Mac en Windows en bestaat uit 3 packages: Conda, Pip en LibTorch. Momenteel kan dit framwork worden gebruikt in Python, C++ of Java. Hier worden ook alle calculaties gedaan op de GPU in plaats van de CPU, hoewel er ook een CPU versie bestaat wordt dit niet aangeraden. Een GPU heeft meer kracht om wiskundige calculaties uit te voeren (PyTorch, g.d.).

2.5 Keras

Keras is een Python framework dat gemaakt is om makkelijker deep learning/machine learning (ML) te implementeren in Python applicaties. Dit framework zorgt ervoor dat de GPU wordt gebruikt om calculaties uit te voeren. Het framework focust zich vooral op de snelheid om projecten te kunnen creëren. Het wenst zich toegankelijk te maken voor iedereen dat Python gebruikt en geïnteresseerd is in AI. (Keras, g.d.).

2.6 User experience (UX)

De user experience (UX) of de gebruikerservaring in een webapplicatie is een breed onderwerp. Wat behoort nu eigenlijk allemaal tot de gebruikerservaring en wat niet? UX focust zich op het verstaan van de gebruikers, wat nodig is, wat er gewaardeerd wordt, de capaciteiten van de gebruiker en hun limieten. Het neemt de business doeleinden van het project of bedrijf in acht (Usability.gov, g.d.).

UX is het proces van het maken van een product (digitaal of fysiek) dat praktisch en bruikbaar is. Dit kan opgedeeld worden in verschillende karakteristieken (Babich, 2020):

- Bruikbaar: Een product moet makkelijk en bruikbaar zijn.
- Nuttig: Een product moet alle noden invullen. Als een product niet alle noden vervuldigd, heeft de gebruiker geen reden om het te gebruiken.
- Gewenst: Een product moet er goed uitzien.
- Vindbaar: Als er een probleem is met het product, moet de oplossing makkelijk te vinden zijn.
- Toegankelijk: Een product moet voor iedereen beschikbaar zijn.
- Geloofwaardig: Een product en het bedrijf moeten geloofwaardig zijn voor alle klanten.

2.7 De user flow

Een user flow is een representatie van alle paden dat een gebruiker kan nemen in een webapplicatie. De flow begint met een startpunt bij het product, zoals een onboarding scherm of een homepagina, en eindigt met een actie, zoals 'de betaling is afgerond' of 'de factuur is aangemaakt'. Door deze flow te representeren kan het proces geëvalueerd en veranderd worden om verbeteringen toe te voegen. Een user flow wordt meestal aan de hand van een flowchart uitgebeeld. Er bestaan verschillende types van flow charts. Enkele voorbeelden zijn (Browne, 2021):

- Task flows: Deze flow focust zich op hoe een gebruiker zich verplaatst binnen een systeem. Deze flows hebben in het algemeen geen meerdere vertakkingen of paden.
- Wire flows: Dit is een combinatie van wireframes en flowcharts. Een wireframe is een blauwdruk van een applicatie waarbij enkel lijnen en tekst worden gebruikt om de layout voor te stellen (Hannah, 2021). De combinatie van een wireframe en een flowchart is dan een wire flow. Het toont de flow van de applicatie aan de hand van de layout.
- User flow: Deze flow focust zich op het doelpubliek en hoe deze de applicatie zouden gebruiken. Een user flow benadrukt dat niet elke gebruiker elke actie zal uitvoeren. Het legt alle paden vast dat mogelijk zijn maar niet noodzakelijk zijn om te volgen.

Naam	**Functie**
Tensors	Dit is de core structuur van Tensorflow. Het is een manier om matrices en vectors te abstraheren.
Models	Modellen kunnen getraind, geëvalueerd en gebruikt worden voor voorspellingen te maken. Modellen zijn een verzameling van Layers (zie volgende API).
Layers	Dit is het basis component om een model te creëren. Het is te vergelijken met een neuron in een neuraal netwerk. Het heeft een input en berekent hierna een output.
Operations	Om wiskundige formules uit te voeren op Tensors (vectors of matrices) kan deze API worden gebruikt. Dit bevat bijvoorbeeld sommeren of vermenigvuldigen van matrices.
Training	Een API om sneller een machine learning algoritme te trainen met data.
Performance	Hierbij worden er functies open gesteld om de snelheid te meten van een bepaald algoritme.
Environment	Tensorflow kan op verschillende plaatsen wiskundige formules uitvoeren. Met de environment API kan er geselecteerd worden waar dit gebeurd, op de CPU of op de GPU.
Constraints	Dit zijn attributen toegevoegd aan de Layers API, bijvoorbeeld de gewichten (weights) of biases.
Initializers	Gebruikmakend van deze functies kunnen er start waarden worden meegegeven aan de Constraints API.
Regulizers	Door deze API kan er worden gezorgd dat de de Layer API een nieuwe 'scoring' functie krijgt zodat deze output binnen bepaalde waarden blijft. Hierdoor worden er limieten opgesteld en houd de API zich aan een 'simpeler' model.
Data	Dit is een API dat het makkelijk maakt om data in te laden in het Tensorflow JS framework. Er worden functies opengesteld zoals 'csv', hiermee kan er een CSV bestand worden ingeladen als dataset.
Visualisatie	Deze API maakt het mogelijk om alle calculaties, data en andere objecten te visualiseren aan de hand van grafieken.
Util	Een paar handige utility's die kunnen worden gebruikt in Javascript.
Browser	Dit laat toe om een Tensor te creëren van een afbeelding of een element in de HTML file (DOM element).
Backends	Een API voor de implementatie van een backend.
Metrics	Dit stelt extra functies ter beschikking om te gebruikten met Tensors.
Callbacks	Deze functie wordt enkel uitgevoerd indien er eerder wordt gestopt met de calculaties. Dan kan er een callback functie worden aangesproken. Deze functie wordt voorop gedefinieerd.

Tabel 2.1: Tensorflow API's en hun functies

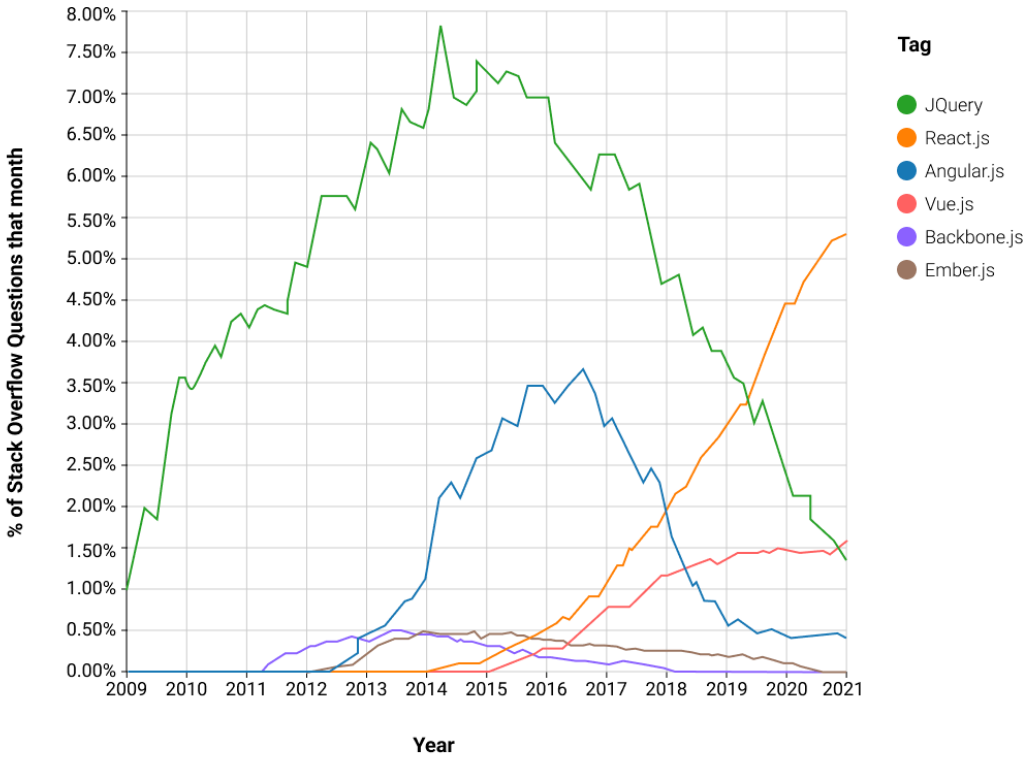
3. Methodologie

Voor het onderzoek werd gestart, diende er eerst een literatuur studie te worden gedaan. Deze kan u vinden in 2. Hierna werd er gekeken welke aspecten in een webapplicatie er best de user experience (gebruikerservaring, UX) kan worden verbeterd. Als de juiste aspecten gevonden waren, kon er verder worden gegaan naar het verzamelen van de juiste data. Bij artificiële intelligentie (AI) en machine learning (ML) is er veel training data nodig. Hierna werden de verschillende frameworks bekeken en hoe er AI en ML kon worden geïmplementeerd in een webapplicatie om de UX te verbeteren. Als laatste werd er een proof-of-concept opgemaakt om te bewijzen dat machine learning niet zo moeilijk te implementeren valt in een front-end applicatie.

3.1 Fase 1

Allereerst was het de bedoeling om te kijken hoe de UX kon worden verbeterd met AI. In fase 1 is er dan geanalyseerd geweest hoe dit het best werd gedaan. Er is een conclusie getrokken dat de UX meer aanleunt bij de front-end kant. Door deze conclusie werd de studie dus beperkt tot enkel het front-end gedeelte van ML en AI. Hierna is er gezocht geweest naar een passend framework om AI te kunnen toepassen in een front-end applicatie.

In figuur 3.1 kan u zien dat het meest gebruikte framework React is. Hierdoor zal er later in deze studie, een proof-of-concept worden gemaakt in React.



Figuur 3.1: The most used frontend frameworks (Gairola, 2022)

3.2 Fase 2

In fase 2 is er geanalyseerd geweest hoe er dan best AI kan worden geïmplementeerd in een React applicatie. Er zijn veel punten waarbij AI kan worden gebruikt om de UX te verbeteren. Enkele UX verbeteringspunten kunnen zijn:

- Het verbeteren van de user flow
- Het verbeteren van de layout
- Het verbeteren van de communicatie tussen de webapplicatie en de gebruiker, dit kan zijn: Email, SMS, ...

In deze studie is er gekozen voor het eerste verbeteringspunt, het verbeteren van de user flow.

3.3 Fase 3

In fase 3 werd er concreet gekeken welk AI framework er best wordt gekozen en welk punt van de user flow er kan worden verbeterd. Om AI te implementeren in een frontend applicatie zijn er niet veel opties. Het enige 'production ready' front-end framework is namelijk Tensorflow. Er bestaan dan ook meerdere vertakkingen van dit framework zoals ML5. Deze studie houdt zich bij beperkt bij de implementatie van Tensorflow aangezien

het ML5 framework meer voor afbeeldingen en video's van toepassing is.

Als verbeteringspunt in de user flow was er gekozen voor acties te suggereren in een webapplicatie. Dit kan bijvoorbeeld een facturatie programma zijn waarbij een gebruiker dagelijks in werkt. Eerst moeten op voorhand bepaalde acties worden gelogd. Na een bepaalde tijdsperiode zal er genoeg data zijn voor een AI om suggesties te maken over welke actie de gebruiker zou willen uitvoeren. Dit is op basis van vorige acties en de tijd van uitvoeren van deze acties.

4. Suggestieve acties

Als eerste werd er in deze studie gekeken naar het suggereren van acties (suggestieve acties) met machine learning (ML). Hoe kan er een suggestie worden gegeven aan de gebruiker om zijn of haar volgende actie te bepalen? Wordt dit best enkel in front-end of ook in back-end gedaan? Dit werd onderzocht door een proof-of-concept te maken, te bekijken en testen. Allereerst werd er een requirements analyse uitgevoerd. Hierdoor werden de mogelijke datasets of data input punten ontdekt. Voor ML zijn er grote datasets nodig om bijvoorbeeld het neuraal netwerk of zelfs een simpeler algoritme te kunnen trainen. Hierbij moest er zeker worden gekeken welke data er mag worden bijgehouden volgens de Europese GDPR wetgeving.

4.1 Het doel

Om de gebruikerservaring te optimaliseren, was er gekozen voor suggestieve acties weer te geven aan de gebruiker. Het doel was om in applicaties op voorhand bepaalde acties te gaan loggen van een gebruiker op een bepaald tijdstip van een weekdag. Als de gebruiker eventueel de dag erna terug komt, kan er een suggestie worden gegeven om een bepaalde actie uit te voeren. Ook kan dit worden toegepast op de layout. Als de gebruiker dagelijks het systeem gebruikt en vaste taken uitvoert, kan de layout automatisch aanpassen op maat van de gebruiker. Er kan dan bijvoorbeeld een knop of een navigatie knop verzet worden dichter in het gezichtsveld van de gebruiker. Er werd bekeken of dit best gedaan wordt via een algoritme in een font-end framework of in een algoritme in een back-end framework.

4.2 Requirements analyse

Voor er van start werd gegaan, was het nodig om duidelijk af te bakenen wat nodig is (must have) en wat leuk is om te hebben (nice to have). In de requirements analyse gebeurd dit in perspectief van een gebruiker van een webapplicatie. Het is belangrijk dat dit perspectief werd doorgetrokken naar alle aspecten zodanig dat de user experience of gebruikerservaring altijd voorop werd gesteld.

4.2.1 Input en output requirements

Het proof-of-concept heeft zeker een aantal punten nodig qua input en output, namelijk:

- De eerste input(s): Gebaseerd op de gebruiker, zonder dat de deze hier weet van heeft. Alle input moet komen van vorig gedrag op de webapplicatie, data vanuit de browser of databank van de webapplicatie. Het is niet de bedoeling dat de gebruiker zelf 'user preferences' gaat instellen.
- De tweede input: Gebaseerd op tijd. Per webapplicatie zal er moeten worden uitgemaakt op welke tijdspanne de applicatie wordt gebruikt. Bijvoorbeeld een facturatie programma wordt dagelijks gebruikt, dan kunnen de tijdstippen van de dag worden gebruikt als een tweede input.
- De derde input: De actie zelf. De actie zal worden gedefinieerd door de ontwikkelaar van de webapplicatie. Deze acties kunnen gaan van 'Ga naar deze pagina' tot 'Druk op deze knop 3 keer'.
- De output: De gesuggereerde actie. Dit kan worden getoond in verschillende vormen in een webapplicatie. Bijvoorbeeld een popup, layout veranderingen of andere mogelijkheden.

4.2.2 Must have's

De belangrijkste punten dat het proof-of-concept nodig had:

- Bovenstaande inputs zijn een must, het gebruikte framework moet dus minimaal een input kunnen behandelen van 2 tot 3 items.
- Calculaties voor het trainen van het model moeten gebeuren via de GPU.
- Alle toevoegingen die gebeuren, mogen niet ten koste gaan van de snelheid van de webapplicatie zelf.
- De gebruikte frameworks of modellen moeten open source zijn.
- Het proof-of-concept moet functioneel klaar zijn voor productie.
- De gebruikte frameworks of modellen moeten goed onderhouden zijn zodat in de toekomst dit geen last kan worden.

4.2.3 Should have's

- Mogelijkheid om in front-end te worden geïmplementeerd. Als alle calculaties worden gedaan in de browser neemt dit veel data verwerking weg van de server. Dit kan dan weer als gevolg hebben dat er geen extra server kost is om machine learning te implementeren in een bestaande webapplicatie.
- Het framework dient alles asynchroon te doen.

4.2.4 Nice to have

Een nice to have is dat de documentatie goed is uitgeschreven van het gebruikte framework. Hierdoor kan er ook worden afgeleid dat het goed onderhouden is. Dit is natuurlijk weer een must have voor naar de toekomst.

4.3 Gekozen algoritme

Om suggestieve acties weer te geven, is het duidelijk dat het K-means clustering, of het K-nearest neighbors algoritme van toepassing was. Het was de bedoeling om acties te kunnen classificeren en hierna een voorspelling uit op te maken. Als de data in een twee dimensionale grafiek wordt voorgesteld, kan worden gezegd dat op de X-as de tijd en op de Y-as de actie kan worden weergegeven. Aan de hand van het Tensorflow framework (Tensorflow, g.d.-c) kunnen er meerdere inputs worden gegeven als Tensors en kan er worden gekeken naar 3, 4, tot zelfs 6 dimensies.

Tensorflow heeft een grote gemeenschap van developers en hieruit is al een bestaand K-means clustering (KNN Classifier) basis API gecreëerd. In dit onderzoek werd dit model gebruikt om een proof-of-concept te creëren.

4.4 Datasets

Er werden geen bestaande datasets gekozen voor deze studie. Dit kwam omdat er heel veel personalisatie in zit per gebruiker. Het was dus de bedoeling dat er zo veel mogelijk data werd verzameld van een gebruiker om hier conclusies uit te kunnen trekken. In deze studie was er een beperkte dataset. De structuur van 1 waarde ziet er als volgt uit:

```
[
    {
        time: 1653229957352,
        action: 1
    }
]
```

Zoals u kan zien word er hier maar gebruikt gemaakt van 2 objecten. Dit kan dan ook worden gezien als een Tensor van de tweede dimensie. De tijd kan worden gecalculeerd door een nieuw datum object te instantieren en hiervan het jaartal, de maand en de dag te resetten naar nul. Dan de functie 'getTime()' uit te voeren. Dit geeft als output de tijd in milliseconden sinds Jan 1, 1970, 00:00:00.000 GMT (Mozilla, g.d.-a). Als deze werkwijze wordt gebruikt om de tijd te plotten in de Tensor, wordt er vanuit gegaan dat de gebruiker de webapplicatie meerdere dagen per week gebruikt. Het is mogelijk om een andere tijdsvoorstelling te gebruiken zodat deze aan een andere webapplicatie kan worden toegewezen. Bijvoorbeeld als er gewenst wordt de suggesties per uur op te slaan dan wordt er best enkel het huidige uur gebruikt als tijdsindicatie. In het proof-of-concept zal enkel de weekdag als integer worden gebruikt.

Als tweede object is de actie nodig. De actie moet worden getransformeerd naar een getal zodat deze kan worden ingeladen in een Tensor. Indien gewenst kunnen er meerdere input velden worden gemaakt. Let op, hoe meer inputvelden, hoe meer data er nodig zal zijn om een eerste suggestie te creëren.

Indien er voor meerder inputvelden wordt gekozen kan het zeker handig zijn om eventueel de beeldscherm instellingen te gebruiken. Hier zit in: de breedte en hoogte van het scherm, de schaal dat het scherm heeft, de maximum waarden van vorige items, ... Zo kan de suggesties van acties meer op maat worden gemaakt van het device en de gebruiker.

4.4.1 GDPR

Voor er verder kan worden gekeken naar meerdere input velden voor het algoritme, is het belangrijk dat de Europese data wetgeving in acht wordt genomen. Het is namelijk niet toegelaten om data op te slaan als de gebruiker dit niet wenst. Natuurlijk hangt dit van use case tot use case af. Indien er toch data vanuit een back-end applicatie komt zal dit zeker moeten worden opgenomen in de algemene voorwaarden. Als het een publieke applicatie is dan gaat dit in de cookie banner moeten worden opgenomen. Omdat er zo veel data wordt gebruikt, dient er voor een productie klare omgeving zeker een analyse te gebeuren om te zien welke data gebruikt mag worden en hoe ver de gebruiker gevolgd mag worden.

5. Proof-of-concept

5.1 Het proof-of-concept

Voor het proof-of-concept werd er gebruik gemaakt van het React framework (React, g.d.). Er was een simpele webapplicatie opgesteld om te tonen dat machine learning vrij simpel is toe te passen is bij een front-end applicatie. Bij dit proof-of-concept werd er gebruik gemaakt van het CNN Model dat te vinden is in het Tensorflow Javascript API.

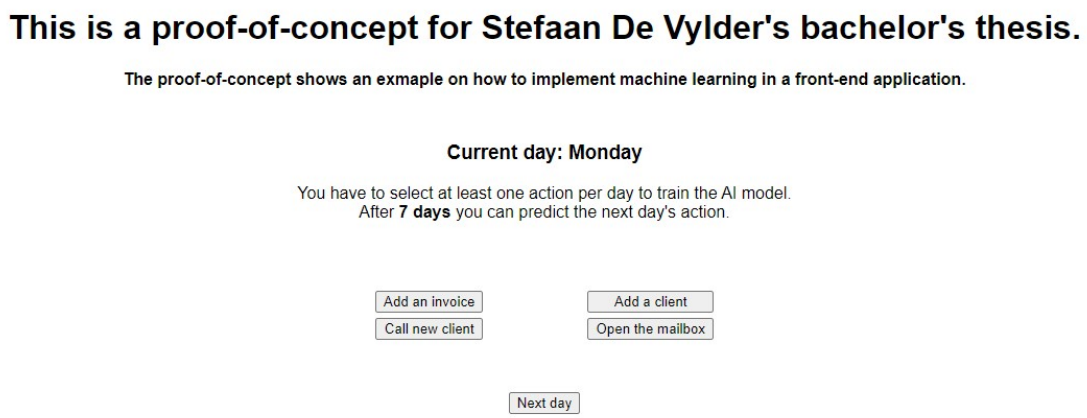
Het proof-of-concept kan worden bekeken via onderstaande GitHub repository:

https://github.com/stefaandevylder/react-suggestive-actions

Figuur 5.2 is een voorbeeld weergave nadat alle stappen zijn voltooid. Dit bevat het uitvoeren van de acties, dus ook het trainen en het maken van voorspellingen gebruikmakend van het CNN algoritme.

5.2 Gebruik

Allereerst was er gebruik gemaakt van de Create-react-app (CreateReactApp, g.d.) API om een 'lege' React (React, g.d.) applicatie te maken die al klaar is voor productie. Dit is gebeurd op basis van Typescript om concreter voorbeelden te kunnen weergeven. Dit maakt het ook mogelijk dat bestaande Typescript en Javascript applicaties geen probleem ondervinden om dit als basis of inspiratie te gebruiken. In figuur 5.1 kan u de initiële toestand zien van de webapplicatie. Deze bevat een aantal knoppen (buttons) met acties en een knop (button) voor de volgende dag te selecteren. Alles in dit proof-of-concept werd



Figuur 5.1: De eerste initiale status van de webapplicatie

gedaan volgens de best-practices van React (React, g.d.). Er werd geen stijl toegepast om de focus te leggen op het functionele werking van de applicatie. Indien deze basis wordt toegepast in een webapplicatie, wordt best de functionaliteit van het volledige algoritme in een store gestopt zodat niet alles via props moet worden doorgegeven. Het algoritme mag maar 1 keer geïnstantieerd worden.

Om het concept juist te kunnen voorstellen zijn er eerst een paar opgestelde regels nodig, namelijk:

- De gebruiker gebruikt dagelijks een webapplicatie om acties in uit te voeren
- De gebruiker zal meerder acties per dag uitvoeren voor een minimum van 7 dagen
- De gebruiker kan in elke gewenste volgorde een actie selecteren
- Indien gewenst, kan de gebruiker een dag overslaan, het is niet erg dat er lege toevoegingen zijn
- Er is geen rekening gehouden met tijd, enkel de dag van de week

Om dit proof-of-concept te gebruiken, begint u met acties te selecteren. U ziet de huidige dag van de week, hieronder 4 knoppen met mogelijke acties en als laatste een knop om naar de volgende dag te gaan. Er dienen eerst een paar acties te worden geselecteerd, minimaal 2, voor er naar de volgende dag kan worden gegaan. In een 'production-ready' webapplicatie zal de tijd worden voorgesteld aan de hand van de dag, uren, minuten eventueel seconden of milliseconden. Het is ook mogelijk om helemaal geen acties in te geven maar dit is niet optimaal voor een AI te trainen. Hoe meer input data er wordt gegeven hoe betere voorspellingen de AI kan maken.

Als er genoeg data is gegeven, voor minimaal 7 dagen, dan wordt een knop zichtbaar gemaakt om een voorspelling weer te geven. Deze voorspelling, of ook wel de suggestie actie genoemd, zal worden gemaakt aan de hand van het KNN Classifier algoritme. Dit algoritme is open source gesteld door Tensorflow JS. Er kan ook gebruik worden gemaakt van het ML5 framework, maar dit is niet gemaakt om in productie te gaan.

5.3 Tensorflow

Gebruikmakend van Tensors en het Tensorflow framework kunnen er suggesties worden gegenereerd in de front-end webapplicatie. Concreet kunnen de gebruikte Tensor gaan voorstellen in een array. Let op, elke input in de array moet altijd een getal zijn. Indien dit niet mogelijk is, moet de input worden voorgesteld als een getal.

```
[ 1, 3 ]
```

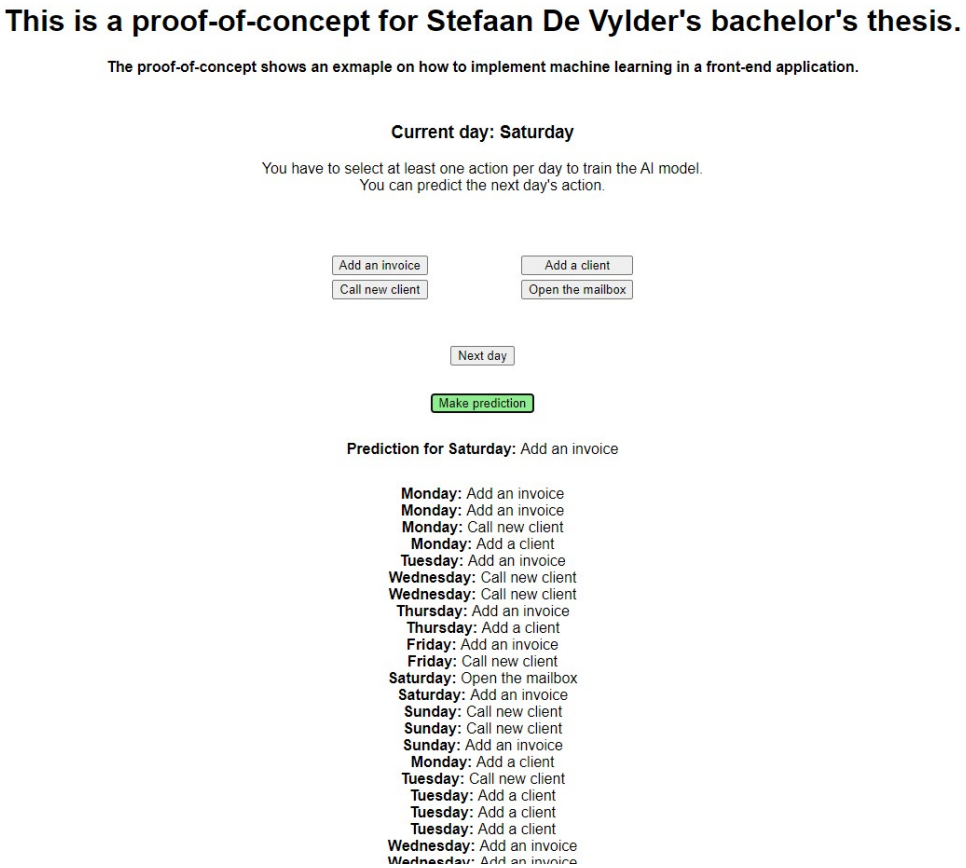
De eerste parameter is het gebruikers identificatie nummer, bij deze webapplicatie zal het altijd op 1 staan, aangezien dit maar door 1 gebruiker werd gebruikt. De tweede is de dag van de week, voorgesteld als een getal.

Als de Tensor is opgemaakt, dient dit enkel nog worden ingegeven in de classifier. Dit gebeurd met de ‘addExample‘ functie. Deze neemt 2 parameters: als eerste de opgebouwde Tensor zoals hierboven beschreven, als tweede zal deze een label opnemen. Door dit label kunnen de Tensors worden geclassificeerd. Bij het proof-of-concept werd de actie als label gekozen. Dit maakt het dan wel een harde vereiste om een consistente naam te kiezen voor het label, en hier natuurlijk niet van af te wijken.

Indien er genoeg training data beschikbaar is, kan er een voorspelling worden opgemaakt. Dit kan worden gedaan met de t’predictClasst’ functie. Deze functie neemt 1 parameter, een Tensor. Het algoritme maakt berekeningen met de Tensors om te bekijken bij welke groep deze het dichtste ligt. Een voorbeeld van een voorspelling zag er als volgt uit:

```
{
    "classIndex": 0,
    "label": "Add an invoice",
    "confidences": {
        "Add an invoice": 0.3333333333333333,
        "Call new client": 0.3333333333333333,
        "Add a client": 0.3333333333333333,
        "Open the mailbox": 0
    }
}
```

Hier kan u zien dat er verschillende output variabelen zijn. Beginnende bij de ‘classIndex‘, dit is de index van de classificatie. Hierna werd het ‘label‘ getoond. Het label werd gebruikt als actie in een string formaat. Als laatste gaf dit ook een object genaamd ‘confidences‘ weer. Dit bevatte alle in rekening genomen acties met daarbij een variabele hoe zelfverzekerd het algoritme is dat dit de juiste waarde is. Hoe meer input data het algoritme krijgt, hoe beter de schatting kan worden gemaakt.



Figuur 5.2: Proof-of-concept van ML in front-end

6. Conclusie

In deze studie werd een antwoord gegeven op de onderzoeksvraag 'Hoe wordt de user experience (gebruikerservaring, UX) van webapplicaties verbeterd met artificiële intelligentie (AI)?'. Er werd geconcludeerd dat de user experience vooral valt onder front-end, wat de back-end zeker niet onbelangrijk maakt. Onder de term user experience valt ook de term user flow. Er zijn verschillende framworks om binnen het front-end gedeelte machine learning toe te passen. De berekeningen kunnen in de browser gebeuren maar ook op een server die daarna de output doorstuurt naar de gebruiker zijn of haar browser. Om zo weinig mogelijk computatie vermogen van andere dingen af te nemen, is het best om dit te doen in de browser in plaats van de server. Tensorflow heeft een fantastisch framework uitgebracht, namelijk Tensorflow JS. Met dit framework valt machine learning makkelijk te implementeren in een front- of een back-end toepassing. Als er naar een concreet voorbeeld wordt gekeken, zoals suggesties geven aan de gebruiker om een volgende actie uit te voeren, kan dit een enorme verbetering zijn voor de gebruiker zelf. Hierdoor kan deze de webapplicatie veel efficiënter gebruiken. Er werd ook geconcludeerd dat dit de user experience van het systeem heeft verbeterd. Aangezien er wel heel wat data nodig is om een machine learning algoritme te trainen, is het belangrijk dat er voldoende data wordt bijgehouden. Dit kan gebeuren in een databank of eventueel lokaal, het hangt af van de use case zelf. De voornaamste data, om suggesties te geven wat de gebruiker vervolgens wilt doen, is tijd en de actie zelf. Hierbij kunnen nog verschillende parameters worden toegevoegd zoals informatie over de gebruiker dat verzameld is doorheen het systeem of informatie over het device dat gebruikt wordt om de webapplicatie te gebruiken. Na deze studie ontstond er een concreter beeld over machine learning en classificatie in een webapplicatie voor ontwikkelaars. Het proof-of-concept toont aan hoe makkelijk het is om dit te implementeren en uit te voeren.

Verder werd er gekeken naar de deelvraag 'Welke aspecten van UX valt te verbeteren met

AI?'. Natuurlijk is er, met de gesuggereerde acties, gezien dat de user flow verbeterd was, maar dit was zeker niet het enige aspect. Aan de hand van machine learning in front-end werden er zeer veel opties opengesteld. Eventuele layout veranderingen met machine learning is zeker ook een mogelijkheid.

Hierna werd de vraag 'Kan AI gebruikt worden in front-end of enkel in back-end?' zeer duidelijk beantwoord. In het back-end gedeelte van een webapplicatie zal het altijd mogelijk zijn om iets van artificiële intelligentie of machine learning te implementeren. Nu er ook een implementatie is dat werkt in een browser, opent dit zeker mogelijkheden voor nieuwe frameworks om zich te gaan spiegelen of nieuwe inzichten creëren bovenop de huidige frameworks.

Als laatste werd de vraag 'Wat voor data heeft de AI nodig, wat mag worden bijgehouden volgens de GDPR wetgeving?' beantwoord. Hier kan er geconcludeerd worden dat dit afhangt van use case tot use case. Er moet zeker eerst een analyse worden gedaan voor er artificiële intelligentie wordt geimplementeerd. De basis data dat gebruikt werd bij het proof-of-concept was tijd en de actie. Ook kwam hier de gebruikers ID aan bod, dit was om aan te tonen dat er verschillende input variabele mogelijkheden zijn. De GDPR was hier niet van toepassing omdat alles in front-end wordt opgeslagen. Indien er een connectie wordt gemaakt tussen de front-end AI en een back-end, zal er zeker een tweede analyse moeten gebeuren over welke data mag worden bijgehouden en eventueel of deze zaken mee in de algemene voorwaarden moeten worden opgenomen. Indien dit een publieke applicatie zonder accounts is, zal er een cookie disclaimer moeten worden gebruikt om aan te tonen dat deze data wordt bijgehouden.

Er werd geconcludeerd dat dit een succesvolle studie was. Er zijn nieuwe en vooral concrete inzichten gecreëerd over hoe machine learning wordt gebruikt in een front-end applicatie. Met behulp van verschillende frameworks konden er nieuwe analyses worden gemaakt om de term artificiële intelligentie verder te zetten.

In verdere studies kan er worden gekeken naar een verdere implementatie van machine learning of neurale netwerken in de browser. Aangezien tegenwoordig, met zaken zoals een Progressive Web App (PWA), de kloof tussen een mobiele applicatie en een webapplicatie kleiner is geworden, kan er worden gekeken naar implementaties van machine learning in applicaties voor een mobiele telefoon. Hoe minder calculaties gebeuren in een back-end omgeving, hoe beter dit uitkomt voor de webapplicatie of het bedrijf zelf. Er is een groot potentieel van calculatie vermogen door elke gebruiker, dit moet dan ook zeker worden benut.

A. Onderzoeksvoorstel

Het onderwerp van deze bachelorproef is gebaseerd op een onderzoeksvoorstel dat vooraf werd beoordeeld door de promotor. Dat voorstel is opgenomen in deze bijlage.

A.1 Introductie

Webapplicaties worden ontwikkeld met een vaste user-flow in gedachte. Echter is dit niet altijd het pad dat een normale gebruiker volgt. De user-flow is het pad dat wordt genomen door een stereotypische gebruiker op een website of app om een bepaald doel te bereiken. (Optipedia, g.d.) Het is één van de belangrijkste parameters bij de user experience (UX) in een webapplicatie. Als de flow aan de hand van artificiële intelligentie kan worden gestuurd door gebruik van persoonlijke voorkeuren, zonder dat de user dit merkt, kan dit een verbetering vormen voor bedrijven zoals dat voor elke user een andere structuur kan worden getoond, optimaal is voor deze user. Niet enkel in het visuele gedeelte kan AI worden gebruikt, maar ook in automated testen.

A.2 State-of-the-art

A.2.1 Wat behoort tot UX

User experience (UX) of gebruikerservaring is niet 1 onderwerp, het bestaat uit talloze verschillende onderwerpen. UX gaat meer over het gevoel van een gebruiker op de website. Het concludeert alle taken dat de gebruiker uitvoert op een website. Het doel van

UX is van zichzelf sprekend:

- Een effectieve website
- Een efficiënte website
- Een hogere tevredenheid

(Divbyte, g.d.)

A.2.2 Het AI probleem

Artificiële intelligentie (AI) is een complex begrip. AI heeft verschillende aftakkingen zoals machine learning (ML). ML betekent dat een computer zich gaat spiegelen aan een mens om bepaalde taken bij te leren. (IBM, g.d.-a) Aan de hand van ML kunnen we gaan berekenen hoe de optimale UX er kan uit zien voor een specifieke gebruiker. Natuurlijk heeft de AI dan genoeg input data nodig, hier rond zal een volledig onderzoek gebeuren. (Speelman, 2017)

A.3 Methodologie

In de eerste fase wordt bekeken welke AI software of framework het best wordt gebruikt. We analyseren welke input het beste is voor welke output. De AI moet op de vraag kunnen antwoorden: "Wat is de optimale user experience voor deze specifieke gebruiker?". Het is een gecompliceerde vraag waarbij voldoende input variabelen nodig zijn. Deze worden ook gezocht in de eerste fase. In de tweede fase wordt er bekeken hoe men de data van deze AI kan omzetten naar de juiste oplossing. Wat gaan we concreet gaan aanpassen? Dit kan zowel front-end als back-end. In front-end gaan we onderzoeken of het mogelijk is om deze dynamisch te maken aan de hand van de gebruiker. We bekijken tot welke vlakken de AI zich moet beperken en welke deze volledig mag gaan configureren. Bijvoorbeeld alles rond privacy. Dit zal enorm voorzichtig moeten worden behandeld. In back-end gaan we kijken welke data er wordt teruggegeven en of deze ook getransformeerd kan worden volgens de AI. Zelfs op vlak van testen kunnen we nu volledig gaan werken met de AI. In testen kunnen we dan met 'real-life' data gaan werken.

A.4 Verwachte resultaten

AI zal altijd een toepassing hebben in webapplicaties en user experience, dit zien we al gebeuren bij de grootste webapplicaties zoals Meta (Facebook). De verwachtingen zijn dat dit een positieve impact kan hebben op gebruikers. Ze zullen zich meer thuis voelen in de webapplicatie.

A.5 Verwachte conclusies

We kunnen concluderen dat UX wel eens kan veranderen. We gaan niet meer kijken hoe een probleemstelling best wordt opgelost, maar een bedrijf gaat het kunnen 'vragen' aan een AI. Bedrijven zullen enorme analyses moeten doen voor ze AI kunnen toepassen, maar door deze werkwijze toe te passen kan de gebruikerservaring (UX) weer een stapje dichter bij het optimale worden gebracht.

B. Bijlagen

B.1 Broncode proof-of-concept

Alle broncode is ook terug te vinden op de GitHub repository:

https://github.com/stefaandevylder/react-suggestive-actions

Package.json

```
{
    "name": "my-app",
    "version": "0.1.0",
    "private": true,
    "dependencies": {
        "@tensorflow-models/knn-classifier": "^1.2.4",
        "@tensorflow/tfjs": "^3.18.0",
        "@testing-library/jest-dom": "^5.14.1",
        "@testing-library/react": "^13.0.0",
        "@testing-library/user-event": "^13.2.1",
        "@types/jest": "^27.0.1",
        "@types/luxon": "^2.3.2",
        "@types/node": "^16.7.13",
        "@types/react": "^18.0.0",
        "@types/react-dom": "^18.0.0",
        "luxon": "^2.4.0",
        "react": "^18.1.0",
        "react-dom": "^18.1.0",
        "react-scripts": "5.0.1",
```

```
        "typescript": "^4.4.2",
        "web-vitals": "^2.1.0"
    },
    "scripts": {
        "start": "react-scripts start",
        "build": "react-scripts build",
        "test": "react-scripts test",
        "eject": "react-scripts eject"
    },
    "eslintConfig": {
        "extends": [
        "react-app",
        "react-app/jest"
        ]
    },
    "browserslist": {
        "production": [
        ">0.2%",
        "not dead",
        "not op_mini all"
        ],
        "development": [
        "last 1 chrome version",
        "last 1 firefox version",
        "last 1 safari version"
        ]
    }
}
```

Suggestive actions.tsx

```
import { KNNClassifier } from "@tensorflow-models/knn-classifier";
import * as ts from "@tensorflow/tfjs";

/**
* Generate the KNN classifier. This should be done only once in the
    application.
*/
const classifier = new KNNClassifier();

export default class SuggestiveActions {
    /**
    * Track an action as a suggestion.
    * @param {Tensor} action The action to track.
    * @param {string} userId The user id.
    */
    trackAction(userId: number, action: string, date: number) {
        classifier.addExample(ts.tensor([userId, date]), action);
    }
```

```
    /**
    * Predict an action based on the user id.
    * @param userId The user id.
    * @returns The predicted action.
    */
    predictAction(userId: number, date: number) {
        return classifier.predictClass(ts.tensor([userId, date]));
    }

    /**
    * Get the actions.
    */
    getActions() {
        return classifier.getClassifierDataset();
    }

    /**
    * Set the actions from a source.
    */
    setActions(source: any) {
        classifier.setClassifierDataset(source);
    }
}
```

App.tsx

```
import { useState } from "react";
import { DateTime } from "luxon";
import SuggestiveActions from "./SuggestiveActions";

/**
* Only need to initialize the suggestive actions once.
*/
const SUGGESTIVE_ACTIONS = new SuggestiveActions();

/**
* We currently hardcode the user id.
*/
const USER_ID = 1;

/**
* Lets say the current date it the start of the week.
* This way the prototype can be better presented.
*/
const CURRENT_DATE = DateTime.now().startOf("week");

/**
* The minimum date that a prediction can be made for.
```

```
*/
const MINIMUM_DATE = CURRENT_DATE.plus({ days: 7 });

interface Action {
    date: string;
    label: string;
}

interface Prediction {
    label: string;
    classIndex: number;
    confidences: {
        [label: string]: number;
    };
}

const App = () => {
    const [currentDate, setCurrentDate] = useState(CURRENT_DATE);
    const [allowPredictions, setAllowPredictions] = useState(false);
    const [actionsByWeek, setActionsByWeek] = useState([] as Action[]);
    const [prediction, setPrediction] = useState<Prediction>();

    /**
    * When next day gets pushed, the current date is updated.
    * Then we clear any prediction.
    * At last we allow predictions again.
    */
    const onNextDayHandler = () => {
        setCurrentDate((date) => date.plus({ days: 1 }));
        setPrediction(undefined);

        if (currentDate.diff(CURRENT_DATE, "days").toObject().days! > 6) {
            setAllowPredictions(true);
        }
    };

    /**
    * Convert the current day to a day of the week integer.
    * Integer is needed for the AI model.
    * @returns The current date as an integer.
    */
    const currentWeekToInteger = () => {
        return parseInt(currentDate.toFormat("E"));
    };

    /**
    * When a button is clicked, we track the action for the local state,
    * after that we register the action in the AI model.
    * This uses KNN classifier.
```

```
*/
const trackButtonClickAction = (action: string) => {
    setActionsByWeek((actions) => [
    ...actions,
    {
        date: currentDate.toFormat("EEEE"),
        label: action,
    },
    ]);
    SUGGESTIVE_ACTIONS.trackAction(USER_ID, action,
        currentWeekToInteger());
};

/**
* Calculate a next action from the AI model.
*/
const calculateSuggestedAction = () => {
    SUGGESTIVE_ACTIONS.predictAction(
    USER_ID,
    parseInt(currentDate.toFormat("E"))
    ).then((result) => setPrediction(result));
};

return (
<div style={{ textAlign: "center", fontFamily: "arial" }}>
<div>
<h1>
This is a proof-of-concept for Stefaan De Vylder's bachelor's thesis.
</h1>
<h4>
The proof-of-concept shows an exmaple on how to implement machine
learning in a front-end application.
</h4>
</div>
<div style={{ marginTop: "50px" }}>
<h3>Current day: {currentDate.toFormat("EEEE")}</h3>
<p>
You have to select at least one action per day to train the AI model.
<br />
{MINIMUM_DATE.diff(currentDate, "days").days >= 0 ? (
    <>
    After <b>{MINIMUM_DATE.diff(currentDate, "days").days} days</b>{"
        "}
    you can predict the next day's action.
    </>
    ) : (
    <>You can predict the next day's action.</>
    )}
</p>
```

```
</div>
<div style={{ display: "flex", justifyContent: "center" }}>
<div
style={{
        display: "flex",
        justifyContent: "center",
        margin: "50px",
        flexDirection: "column",
        gap: "5px",
}}
>
<button onClick={() => trackButtonClickAction("Add an invoice")}>
Add an invoice
</button>
<button onClick={() => trackButtonClickAction("Call new client")}>
Call new client
</button>
</div>
<div
style={{
        display: "flex",
        justifyContent: "center",
        margin: "50px",
        flexDirection: "column",
        gap: "5px",
}}
>
<button onClick={() => trackButtonClickAction("Add a client")}>
Add a client
</button>
<button onClick={() => trackButtonClickAction("Open the mailbox")}>
Open the mailbox
</button>
</div>
</div>
<div>
<button onClick={onNextDayHandler}>Next day</button>
</div>

{allowPredictions && (
    <div style={{ marginTop: "30px" }}>
    <button
    onClick={calculateSuggestedAction}
    style={{ backgroundColor: "lightgreen" }}
    >
    Make prediction
    </button>
    </div>
    )}
```

```
    {prediction && (
        <div style={{ marginTop: "30px" }}>
        <b>Prediction for {currentDate.toFormat("EEEE")}: </b>
        <span>{prediction.label}</span>
        </div>
        )}

    <div style={{ marginTop: "30px" }}>
    {actionsByWeek.length > 0 &&
        actionsByWeek.map((action, index) => (
        <div key={index}>
        <b>{action.date}: </b>
        <span>{action.label}</span>
        </div>
        ))}
    </div>
    </div>
    );
};

export default App;
```

index.tsx

```
import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
import reportWebVitals from "./reportWebVitals";

const root = ReactDOM.createRoot(
document.getElementById("root") as HTMLElement
);
root.render(
<React.StrictMode>
<App />
</React.StrictMode>
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

Bibliografie

Adler, S. (2019). A Quick Guide to Planning, Scheduling, and Optimization. https://www.intelligentautomation.network/decision-ai/articles/a-basic-guide-to-planning-scheduling-and-optimization

Arnx, A. (2019). First neural network for beginners explained. https://towardsdatascience.com/first-neural-network-for-beginners-explained-with-code-4cfd37e06eaf

Babich, N. (2020). What You Should Know About User Experience Design. https://xd.adobe.com/ideas/career-tips/what-is-ux-design/

Browne, C. (2021). What are User Flows in User Experience (UX) Design? https://careerfoundry.com/en/blog/ux-design/what-are-user-flows/

CreateReactApp. (g.d.). Adding typescript. https://create-react-app.dev/docs/adding-typescript/

DataScienceTeam. (2020). K-means clustering in machinaal leren. https://datascience.eu/nl/machine-learning/k-middelen-clustering-in-machinaal-leren/

Divbyte. (g.d.). The Importance of Artificial Intelligence in Web Development. https://divbyte.com/importance-artificial-intelligence-web-development/

Gairola, A. (2022). https://www.bacancytechnology.com/blog/best-frontend-framework

GeekForGeeks. (2020). Hierarchical Clustering in Data Mining. https://www.geeksforgeeks.org/hierarchical-clustering-in-data-mining/

Google. (g.d.). Vision AI. https://cloud.google.com/vision

Hannah, J. (2021). What Exactly Is Wireframing? A Comprehensive Guide. https://careerfoundry.com/en/blog/ux-design/what-is-a-wireframe-guide/

IBM. (g.d.-a). Machine Learning. https://www.ibm.com/cloud/learn/machine-learning

IBM. (g.d.-b). What is computer vision? https://www.ibm.com/topics/computer-vision

IBMCloudEductation. (2020). Natural Language Processing (NLP). https://www.ibm.com/cloud/learn/natural-language-processing

Johnson, D. (2022). Unsupervised Machine Learning: Algorithms, Types with examples. https://www.guru99.com/unsupervised-machine-learning.html

Kar, S. (2017). AI Mind Map. https://medium.com/ml-ai-study-group/ai-mind-map-a70dafcf5a48

Keras. (g.d.). Keras. https://keras.io/

ML5. (g.d.). ML5. https://ml5js.org/

Mozilla. (g.d.-a). Date.prototype.getTime(). https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Date/getTime

Mozilla. (g.d.-b). WebGL: 2D and 3D graphics for the web. https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API

Optipedia. (g.d.). User flow. https://www.optimizely.com/optimization-glossary/user-flow/#:~:text=User%20flow%20is%20the%20path,such%20as%20purchasing%20a%20product.

Oracle. (g.d.-a). Wat is AI? https://www.oracle.com/nl/artificial-intelligence/what-is-ai/

Oracle. (g.d.-b). Wat is machine learning? https://www.oracle.com/nl/data-science/machine-learning/what-is-machine-learning

PyTorch. (g.d.). Torch. https://pytorch.org/

React. (g.d.). React. https://reactjs.org/

Sharma, S. (2017). Activation functions in neural networks. https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

Sophia. (g.d.). Sophia. https://hansonrobotics.com/sophia/

Speelman, B. (2017). Wat is user experience? https://bramspeelman.nl/user-experience-betekenis/

Tensorflow. (g.d.-a). KNN Classifier. https://github.com/tensorflow/tfjs-models/tree/master/knn-classifier

Tensorflow. (g.d.-b). Tensorflow API. https://js.tensorflow.org/api/latest/

Tensorflow. (g.d.-c). Tensorflow JS. https://www.tensorflow.org/js

Usability.gov. (g.d.). User Expeirence Basics. https://www.usability.gov/what-and-why/user-experience.html