Ontwerp en ontwikkeling van een gezondheidsaanbevelingssysteem gericht op rope skipping

Design and development of a health recommender system focussing on rope skipping

Elise Thienpont

faculteit ingenieurswetenschappen en architectuur
Universiteit Gent
België
31 Maart 2020

Dankwoord

Abstract

Fysieke activiteit is broodnodig in onze sedentaire samenleving. Zonder enige vorm van persoonlijke coaching is dit echter moeilijk te realiseren. Tegenwoordig heeft iedereen wel een soort mobiel toestel op zak. Dit is een bron van mogelijkheden op vlak van fysieke activiteit coaching. Een smartwatch geeft hier nog een extra dimensie aan door de fysieke activiteit rechtstreeks te monitoren op basis van hartslagsensoren. Deze persoonlijke data maakt het mogelijk om ook persoonlijke aanbevelingen te produceren. Één van de argumenten in het voordeel van te weinig beweging is het gebrek aan tijd. Hiervoor is rope skipping de ideale oplossing. Deze sport is namelijk de ideale conditietraining waardoor gebruikers optimaal en efficiënt bewegen. Ook kan deze sport eender waar uitgeoefend worden mits een beetje plaats. Qua activity recognition van specifieke rope skipping bewegingen is nog te weinig onderzoek gebeurd. Door de bewegingen te herkennen en eventuele tekortkomingen te detecteren, kan voor extra aanmoediging gezorgd worden. Deze paper beschrijft een Android applicatie ontwikkeld om de mens op conditie te krijgen. Dit met toevoeging van het leuke element rope skipping. In een eerste deel wordt bestaande literatuur bekeken met betrekking tot activity recognition, bepalen van inspanningsniveaus, goal prediction en recommender systems. Een tweede deel gaat dieper in op de gebruikte technologieën. Een goed inzicht in het materiaal/de technologieën waarmee gewerkt wordt is namelijk vereist. Vervolgens wordt meer verteld over het activity recognition proces. Door verzameling van data afkomstig van verschillende proefpersonen wordt een model ontwikkeld. Dit model is in staat om 5 rope skipping bewegingen te classificeren. In een laatste deel wordt de gezondheidsapplicatie toegelicht. Deze applicatie gaat, gebaseerd op het inspanningsniveau bij de verschillende rope skipping bewegingen, aanbevelingen genereren. Het inspanningsniveau wordt bepaald aan de hand van de Metabolic Equivalent Task (MET) metriek. Het aantal METs is afhankelijk van de tijd die in een bepaalde heart rate zone doorgebracht werd. Aanbevelingen worden berekend aan de hand van enerzijds de frequentie van uitvoering en het gemiddeld aantal METs verbruikt per minuut per beweging. Door te werken met een doel wordt een bovengrens gecreëerd voor het aantal aanbevelingen. Dit doel wordt bepaald door historische data tot 10 weken in het verleden te bekijken en hier het 60ste percentiel van te nemen. De aanbevelingen zijn naast de frequentie waarmee een activiteit uitgevoerd wordt, ook afhankelijk van het aantal fouten tijdens een beweging.

Trefwoorden: rope skipping, gezondheidsapplicatie, wear OS, android, recommender system

Inhoudsopgave

Figurenlijst

Tabellenlijst

Afkortingen

Introductie

Probleemstelling

Sport en beweging wint steeds aan populariteit in onze samenleving. Meer mensen gebruiken dan ook mobiele applicaties om hun sportprestaties en algemene gezondheid te verbeteren.

Huidige gezondheidsapplicaties geven echter vaak enkel statische aanbevelingen. Ze houden geen rekening met de persoonlijke vooruitgang van de gebruiker, zijn of haar conditie en fysieke capaciteiten. Gebruikers moeten zelf hun gewenste doelstellingen opgeven. De applicatie zal dan, gebaseerd op deze parameters, bijhorende aanbevelingen geven.

Rope skipping is een sport die voor een goede conditie zorgt en weinig plaats of tijd vergt. De sport is dus uitermate geschikt om mensen aan te zetten tot meer beweging. Een diepgaande opvolging van deze activiteit werd nog niet eerder op punt gezet.

Stress en een slecht of variërend slaappatroon zijn mogelijk belangrijke parameters in het bepalen van een bewegingspatroon. Hier wordt nog onvoldoende mee rekening gehouden.

Bij de meeste applicaties wordt ook geen rekening gehouden met externe factoren zoals bijvoorbeeld het weer en het tijdstip. Dit zorgt ervoor dat het advies niet altijd aangepast is aan de omstandigheden en men dus het voorgesteld schema niet altijd ten volle kan opvolgen.

Indien de gebruiker zijn/haar fysieke grenzen overschrijdt, omwille van vrijwillige forcering of ongezonde omstandigheden, wordt dit meestal niet gemeld. Hierdoor kunnen langdurige klachten ontstaan, wat de gezondheid zeker niet ten goede komt. Een classificatie van de gebruikerscontext in termen van gezondheidstoestand is noodzakelijk.

Doelstelling

Het doel van deze thesis is het ontwikkelen van een mobiele applicatie die gebruikers een gezonde levensstijl aanleert door aanbevelingen in de vorm van slimme meldingen te genereren. Deze meldingen zullen duidelijk maken wat de gebruiker moet doen om in optimale gezondheid te blijven. De inhoud van deze meldingen is gebaseerd op de fysieke activiteiten van de gebruiker en wat

zijn/haar huidige conditie toelaat (gepersonaliseerd). Ook is melden op een gepast tijdstip van belang. In de probleemstelling werd aangegeven dat dit bij huidige health applicaties nog niet ten volle gerealiseerd wordt. De applicatie zal zich vooral focussen op rope skipping aangezien dit de doelstelling van de thesis goed verwezenlijkt.

De thesis zal een antwoord bieden op volgende onderzoeksvraag: "Hoe kan vanuit ruwe data (hartslag, beweging) een gepersonaliseerd bewegingspatroon ontwikkeld worden?"

Het probleem van statische, niet gepersonaliseerde aanbevelingen wordt aangepakt door een op maat gemaakt bewegingspatroon te creëren voor iedere gebruiker. Via algoritmen die ruwe data van sensoren (hartslag, beweging) omzetten in specifieke activiteiten of inspanningsniveaus zal dit ontwikkeld worden.

Ook zal er aandacht besteed worden aan bepaalde deelonderzoeksvragen. Een eerste van deze deelonderzoeksvragen luidt als volgt: "Welke invloed heeft beweging op het stressniveau en slaappatroon?" Eerst en vooral zal een algoritme ontwikkeld worden om het stressniveau alsook het slaappatroon te bepalen. Er zal onderzoek gedaan worden naar hoe dit mogelijk is vanuit hartslagmetingen en metingen bekomen uit de accelerometer. Verder zal de vergelijking gemaakt worden tussen stressniveau/slaappatroon voor en na meer beweging.

De tweede deelonderzoeksvraag luidt als volgt: "Op welke momenten worden aanbevelingen, gebaseerd op het bewegingspatroon, best gemeld?" Om ervoor te zorgen dat het advies ten allen tijde kan opgevolgd worden, moet rekening gehouden worden met externe factoren zoals het weerbericht, lokale nieuwsberichten etc. Op die manier is het zeker dat in de huidige omstandigheden het advies kan opgevolgd worden. Indien het regent is een aanbeveling om buiten te gaan lopen bijvoorbeeld niet gepast.

De derde en laatste deelonderzoeksvraag omvat: "Wanneer wordt intensief bewegen ongezond?" Een classificatie van de gebruikerscontext in termen van gezondheidstoestand kan bekomen worden door constante monitoring van de hartslag. Via deze data kunnen ongewone patronen gedetecteerd worden. Uit het weerbericht kan ook info over bijvoorbeeld de temperatuur gehaald worden. Al deze gegevens samen geven een beeld van de gebruikerscontext in termen van gezondheidstoestand. Hieruit kan afgeleid worden of de fysieke toestand van de gebruiker en/of de omgeving verdere intensieve beweging toelaat.

Chapter 1

Literatuurstudie

1.1 Activity recogntion

Op het vlak van activity recognition is al veel onderzoek gebeurd. In wat volgt worden relevante onderzoeken in meer detail besproken wat betreft de keuze van algoritme, hyperparameters, feature extraction...

[38] In "Deep Activity Recognition Models with Triaxial Accelerometers" probeert men aan de hand van deep learning een accurater activity recognition model te creëren.

Gebruik maken van deep learning methoden heeft namelijk vele voordelen in termen van systeem performantie en flexibiliteit. oppervlakkige modellen blijven vaak hangen in lokale optima. Handmatige feature extraction is hierbij niet meer nodig. Handgemaakte features zoals statistische berekeningen zijn probleem specifiek en ze kunnen niet veralgemeend worden naar andere domeinen. menselijke interventie voor selecteren van de meest effectieve features is nodig. Data driven approaches kunnen onderscheidende features leren vanuit historische data, dit is automatisch en systematisch. deep generative models zijn meer bestand tegen overfitten.

De studie kaart aan dat spectrogram analyse van accelerometer data nodig is aangezien accelerometers multi-frequentie, aperiodische en fluctuerende signalen genereren. Deze transformatie helpt het deep activity model. Een spectrogram van een signaal is de representatie van verandering in de acceleratie energie inhoud als functie van de frequentie en tijd. voordelen hiervan zijn verhoging van de classificatie accuraatheid. Het spectrogram geeft namelijk de intensiteitsverschil tussen punten weer. Hierdoor is de classificatie van activiteiten gebaseerd op de variantie van spectrale densiteit. Dit verminderd de de classificatie complexiteit. De lengte van het signaal na de transformatie is kleiner (minder dimensionaliteit) en dus minder computationele complexiteit. Er kan ook ruis op de data zitten door temperatuur, elektromagnetische velden... Dit wordt eruit gefilterd met een ruis-vector.

Het deep learning model leert de gewichten van het neuraal netwerk en de informatieve features uit onbewerkte data. Het model bestaat uit 2 stappen: een unsupervised, pre training stap en een supervised, fine-tuning stap. De pre training stap leert features aan de hand van deep belief networks en restricted boltzmann machines. Eerst worden deep belief networks gebruikt. Dit zijn generatieve modellen bestaande uit meerdere lagen van hidden units (restricted Boltzmann machines RBM). De volgende lagen zijn binary-binary RBMs.

Data verzamelen en training van het model gebeurt offline (niet op de mobiele applicatie). Bij de training wordt gebruik gemaakt van een sliding window. Met het gemaakte model kan dan aan online activity recognition gedaan worden.

[47] "On the use of ensemble of classifiers for accelerometer-based activity Recognition" is een andere studie van activity recognition op basis van accelerometer data. De onbewerkte time series data werd onderverdeeld in 10 s segmenten en per segment werden verschillende features geïdentificeerd. 6 basis feature types werden gebruikt: gemiddelde, standaard deviatie, gemiddeld absoluut verschil, gemiddelde resulterende versnelling, tijd tussen pieken en binned distribution. Op basis van deze types wordt een groot aantal features bekomen per segment. In deze studie werden J48, logistic regression en MLP gebruikt als algoritmen. Dit zijn geen deep learning methodes waardoor handmatige feature extraction inderdaad vereist is. De studie toont aan dat een ensemble van classifiers kan gebruikt worden voor complexe activiteiten herkenning. Voor de train-test split werd 10-Fold cross validation gebruikt.

[12] In "Statistical Analysis of Window Sizes and Sampling Rates in Human Activity Recognition" wordt onderzocht wat de meest optimale window size en sampling frequentie is. Dit is echter zeer afhankelijk van het soort activiteit. Windows van 0.5 tot 17 zijn mogelijk. Deze studie brengt menselijke reactietijd in rekening. Een minimum sampling frequentie van 32Hz is nodig om correcte data te bekomen. Partiële windows worden verwijderd. Dit zijn windows waarin niet genoeg data aanwezig is om de volledige tijd te vullen. Het aantal data objecten in een window hangt logischerwijs af van de window grootte en sampling rate. 246 features werden geëxtraheerd en werden gereduceerd tot een aantal van 32 via correlation-based feature selection. Deze features komen uit zowel het tijds- en frequentiedomein. Random Forest classifier geeft goede resultaten op de gebruikte dataset. De variantie neemt toe samen met de window size, de gewichtsfunctie bij normaliseren moet dus hiervan afhangen. De ideale combinatie van window grootte en frequentie werd gevonden op basis van geslacht, leeftijd en BMI. Deze onderverdelingen gaven dezelfde resultaten, namelijk een window grootte van 10 seconden en een sampling frequentie van 50 Hz.

[9] In "Activity Recognition using Accelerometer Sensor and machine learning classifiers" wordt een afweging gemaakt tussen het aantal sensoren en de gebruikte classifier. Er werd geconcludeerd dat gebruik van meerdere sensoren (bijvoorbeeld pols en borst) resulteert in slechtere resultaten en minder gebruiksgemak. Meerdere sensoren resulteren in belemmeren van bewegingen en het niet

praktisch zijn bij langdurig gebruik. Enkel een accelerometer sensor wordt gebruikt en geen gyroscoop sensor. Vroegere studies hebben bewezen dat meerdere sensoren nadelig zijn bij het identificeren van menselijke activiteiten. Er werd eveneens bewezen dat accelerometer data voldoende is om de classificatie tot een goed eind te brengen. Voor signaal segmentatie wordt gebruik gemaakt van een sliding window met 50% overlap. Dit levert goede resultaten. Een volgende stap is feature extraction. Hierbij worden features berekend gebruikmakend van het tijd- en frequentiedomein. Hierna wordt een normality test uitgevoerd om na te gaan of de bekomen features passen in een normale distributie. Als we dit weten kan bepaald worden of parametrische of niet-parametrische classificatie tools moeten gebruikt worden. Volgende 3 testen worden gebruikt: shapiro-Wilk, kolmogorov-smirnov en anderson-darling. Als de features niet in een normale distributie passen zijn niet-parametrische tools beter geschikt. Dimensionale reductie wordt gedaan met behulp van PCA. Alle features zijn genormaliseerd in het [0,1] interval In deze studie werd een vergelijking gedaan tussen de performantie van de machine learning algoritmes voor en na dimensionale reductie. De gemiddelde accuraatheid ligt hoger bij het toepassen van dimensionality reduction. Ook werd de performantie van algoritmes op tijds- en frequentiedomein gebaseerde features vergeleken. Hieruit wordt geconcludeerd dat het frequentie domein betekeninsvollere info geeft.

[2] "Integrating features for accelerometer-based activity recognition" is nog een voorbeeld van activity recognition en de mogelijke feature extraction hierbij. Het is nodig om normalisatie van de input data te doen nog voor andere preprocessing technieken uitgevoerd worden. In verband met feature extraction worden volgende zaken toegepast. In het tijdsdomein worden 17 features berekend over elk window en voor elke as. Deze bevatten statistische features (gemiddelde, variantie, standaard deviatie), envelope metrieken (mediaan, interval maximum en minimum waarde, root main square) en andere features (signaal magnitude area, indexes van minimum en maximum waarde, kracht, energie, entropie, skewness, kurtosis, interquartile range en mean absolute deviation van het signaal) In het frequentie domein worden 6 features bekomen per window voor elke as. Dit domein krijgt men door de Fast Fourier transform uit te voeren. Het gaat over volgende features: band power of signal, energie, magnitude, gemiddelde, maximum en minimum waarden van het signaal. In het time-frequency (wavelet) domein worden 9 sets van features bekomen, ook weer voor elk window en elke as. De discrete wavelet transform wordt hiervoor gebruikt. Hier wordt beweerd dat tijdsdomein features betere resultaten geven. Dit is in contradictie met een eerder besproken studie.

[43] "Real-Time Human Ambulation, Activity, and Physiological Monitoring: Taxonomy of Issues, Techniques, Applications, Challenges and Limitations" is een voorbeeld van een online activity recognition systeem. Data processing kan op het device gedaan worden maar hierbij zijn beperkingen door gelimiteerde hardware (minder data die kan gebufferd worden), de robuustheid van classificatie algoritmen en het aantal events dat kan geclassificeerd worden (en-

ergie consumptie). Deze studie zag een piek van classificatie accuraatheid bij 20 Hz. Volgende segmentatie technieken werden gebruikt: BUp, SWAB.

1.2 Inspanningsniveau

Verschillende personen kunnen dezelfde activiteit uitoefenen maar het inspanningsniveau hierbij kan sterk verschillen. In volgende studies wordt gezocht naar een metriek om dit niveau voor te stellen.

[45] De studie getiteld "A data-driven approach to modeling physical fatigue in the workplace using wearable sensors" doet onderzoek naar het bepalen van het niveau van fysieke uitputting aan de hand van accelerometer data of een zelf aangegeven niveau van moeheid. De schaal die hier wordt gebruikt is de Borg rating of perceived exertion. Andere metrieken voor het meten hiervan zijn: hartslag, beschikbare kracht, tremor, veranderingen in postuur, multi joint coördinatie tussen verschillende segmenten. Er wordt gewerkt in verschillende fasen. Eerst wordt de data verzameld, dan volgt een data preprocessing fase. In fase 3 worden verschillende penalized regression models toegepast op de data en in fase 4 volgt een model evaluatie en test fase. Data cleaning is een eerste taak in de data preprocessing fase. Hierbij wordt gecontroleerd op verkeerde sensor data, noisy data... Hartslag werd bijvoorbeeld ook genormaliseerd naar het interval met als ondergrens de resting heart rate en als bovengrens de age predicted maximum heart rate. Hierna werd de jerk berekend, dit is de verandering in versnelling en is een belangrijke indicator van fysieke uitputting.

[10] In "Validity of Accelerometry to Measure Physical Activity Intensity in Children With an Acquired Brain Injury" werd zuurstof gebruik (VO2) gebruikt als metriek voor fysieke uitputting. Kinderen oefenden activiteiten uit met een indirect calorimeter. Dit is een gas analyse systeem dat het volume van uitgeademde lucht meet alsook de O2 en C02 concentratie hierin. Een accelerometer (AG) en een hart slag monitor zijn eveneens gebruikte sensoren. Verschillen in VO2 en AG vector magnitude werden gemeten. Het doel van deze studie is om specifieke intensiteits-afbakeningspunten af te leiden om activiteiten in te delen in volgende categorieën: SED (sedentary), LPA (light PA) en MVPA (moderate, vigourous PA). Met deze data wordt het aantal METs berekend door het gemiddelde VO2 verbruik te delen door de voorspelde RMR (Resting Metabolic Rate). Aan de hand van deze berekende METs werd de classificatie uitgevoerd in SED, LPA en MVPA.

[44] In "Metabolic equivalent of task (METs) thresholds as an indicator of physical activity intensity" wordt gebruik gemaakt van METs om de mate van inspanning op te delen in bepaalde niveaus. Hiervoor moet schatting voor de maximum hartslag bepaald worden. In deze studie gebruikt men volgende formule: 208-(0,7*age). Gemiddelde zuurstof inname werd gemeten en geconver-

teerd naar METs. Waarden voor afbakening werden gevonden die hoger lagen dan eerdere onderzoeken. Individuen met betere conditie hebben een hogere grens uitgedrukt in METs.

[40] "Improving Physical Activity mHealth Interventions: Development of a Computational Model of Self-Efficacy Theory to Define Adaptive Goals for Exercise Promotion" De applicatie ontwikkelt in deze studie gaat persoonlijke suggesties geven voor fysieke activiteit. Voor elke week wordt een goal berekend op basis van al dan niet bereiken van doelen in vorige weken en op basis van SE beliefs. SE beliefs zijn antwoorden op vragen die aan de gebruiker gesteld werden over de uitgevoerde activiteit. De SE score is het gemiddelde van de gegeven antwoorden. Het wekelijks doel kan gesplitst worden in dagelijkse doelen. Dit systeem houdt geen rekening met de context, wel laat het de gebruiker handmatig suggesties verplaatsen op basis van werkuren of weer. Het doel wordt uitgedrukt in METs op basis van hartslag. Er wordt gekeken hoelang men heeft doorgebracht in de gemiddelde intensiteitszone (6*MAXHR/10, 7 * MAXHR/10) en hoelang in de intensieve zone (7*MAXHR/10, 8*MAXHR/10). De algemene richtlijnen zeggen dat 600 METs per week nodig is om een gezond leven te leiden. Voor de berekening van het wekelijks doel werd gebruik gemaakt van een Dynamic Descision Network (DDN). Dit is een opeenvolging van simpele Bayesian Networks. Het DDN model heeft beslissingsvariabelen op elk trainingsniveau.

1.3 Goal setting

Dynamisch doelen zetten kan gebeuren op basis van complexe machine learning algoritmen of simpelere heuristieken. Volgende studies geven enkele voorbeelden.

[42] De studie "Personalized Physical Activity Coaching: A Machine Learning Approach" heeft als doel om fysieke activiteit te verhogen tijdens werkdagen. Hiervoor wordt een stappenteller ontwikkeld in de vorm van een doelgerichte applicatie. De gebruiker voert een doel in, de applicatie zal dan een schatting geven van de waarschijnlijkheid dat het doel gehaald wordt. Data werd verzameld met behulp van de Fitbit Flex. Deze werd gecleand, gereformat en gepreprocessed. Incomplete dagen werden verwijderd, net zoals dagen met geen stappen en weekend dagen. Er werd aan feature constructing gedaan door uur van de werkdag, aantal stappen tijdens dat uur enz. toe te voegen. Het interval van 7 tot 18uur werd bekeken aangezien dit de uren zijn die de meesten doorbrengen op hun werk. Het voorspellen of een bepaald doel zal behaald worden is een binair supervised classificatie probleem. Het is echter onmogelijk om op voorhand te bepalen welk algoritme het best zal presteren, ook al bestaan er bepaalde richtlijnen. Ze moeten dus empirisch getest worden. De gebruikte algoritmes zijn: adaboost, decision trees, KNeighborsClassifier, Logistic Regres-

sion, Neural Network, Stochastic Gradient Descent, Random Forest en support Vector Classification. Deze keuze werd gebaseerd op de flow chart van scikit learn en de cheat sheet van microsoft azure.

[19] "Personalizing Mobile Fitness Apps using Reinforcement Learning" bestudeert doelen stellen aan de hand van reinforcement learning. Goal setting is namelijk een belangrijke factor in veroorzaken van verandering in gedrag. Dynamisch doelen stellen doet dit nog veel meer. Dit kan gebeuren door simpele heuristieken zoals het nemen van de 60ste percentile van de stappen genomen in de voorbije 10 dagen. Een complexere benadering in de vorm van machine learning maakt gebruik van reinforcement learning. Het behavioral analytics algoritme gebruikt inverse reinforcement learning om een model te bekomen vanuit historische data. Hierna wordt dit model gebruikt om realistische goals te genereren.

[5] "Effects of Simple Personalized Goals on the Usage of a Physical Activity App" Deze studie ontwikkelt een applicatie die gebruikers ertoe moet aanzetten om te vermageren. Dit wordt gedaan door bij registratie een aantal parameters te vragen aan de gebruiker: leeftijd, geslacht, lengte, gewicht en een schatting van de ingenomen calorieën per dag. Met deze gegevens wordt het aantal calorieën berekend die de persoon moet verliezen en het aantal calorieën die moeten genomen worden om het uiteindelijke gewicht te behouden. Dit doel wordt niet bijgewerkt met gemonitorde gegevens van calorie-inname.

[6] "WalkWithMe: Personalized Goal Setting and Coaching for Walking in People with Multiple Sclerosis" Self set goals zijn direct gerelateerd aan self-efficacy, aangezien mensen meer geneigd zijn om een doel te zetten die ze denken aan te kunnen. In deze studie wordt hiervan gebruik gemaakt. Het doel wordt dus gezet door de gebruiker. Gebruikers starten te stappen met een duur afhankelijk van hun zelfgekozen doel. Elke week worden 5 minuten bijgeteld bij deze duur. De bedoeling is om uiteindelijk het vooropgestelde doel te bereiken in de laatste week van het programma.

1.4 Recommendations

Recommender systems zijn een hot topic bij grote bedrijven zoals netflix, facebook enz. Er is hier dan ook veel onderzoek naar gedaan.

[46] "The use of machine learning algorithms in recommender systems: A systematic review" gaat onderzoeken welke algoritmes met machine learning vooral gebruikt worden in aanbevelingssystemen. Studies werken meestal met neighborhood gebaseerde collaborative filtering, gevolgd door classifier gebaseerd content-based filtering en model gebaseerd collaborative filtering op respectievelijk de tweede en derde plaats. Indien gebruikt gemaakt wordt van machine learning 80% van de gevallen supervised learning.

[12] Het is nodig dat aanbevelingssystemen rekening houden met de context. Een ordinair systeem doet dit echter niet.

"A Decision Tree Based Context-Aware Recommender System" implementeert context-aware recommendations aan de hand van decision trees. Een beslissingsboom gebaseerd CARS framework wordt als volgt geïmplementeerd. Een afzonderlijke boom is aanwezig voor elke gebruiker (ID3 gebaseerd op content-based filtering) Deze boom houdt rekening met de context. Hierna gebeurt collaborative filtering. De classifier heeft input data van de vorm ¡user, item, contextual features -¿ rating¿. De beslissingsboom bestaat uit 2 stappen: inductie en inference. Informatie gain van elk contextueel feature wordt gebruikt als splitting criterium.

[8] "Context-Aware Mobile Recommender System Based on Activity Patterns" In deze studie werd een widget gemaakt die gebruikersactiviteit opslaat en hiermee voorspelt welke acties de gebruiker gaat ondernemen. De data die bekeken wordt zijn sms-berichten, telefoongesprekken en gebruikte applicaties. Het aanbevelingsalgoritme is volledig gefocust op SQL functionaliteit. Bij bijvoorbeeld het aanbevelen van de meest waarschijnlijke telefoonoproep wordt de totale duur van elke oproep, gefilterd door dag van de week en gegroepeerd met contact nummer. De grootste duur wordt dan gegeven als voorspelling.

Chapter 2

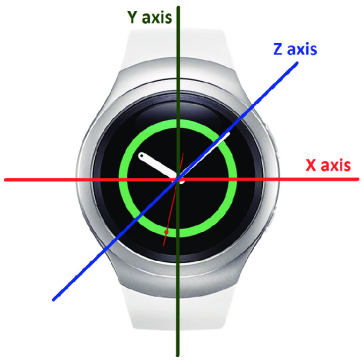
Gebruikte technologie

2.1 Polar M600

[25]

De Polar M600 is een smartwatch die wear OS en de sportfuncties van polar samenbrengt. Het sporthorloge is compatibel met de meeste android smartphones en iphones. Wear OS maakt het mogelijk om meldingen komende van de gekoppelde smartphone op de smartwatch te ontvangen alsook muziek en email communicatie beheren. De polar M600 heeft een ingebouwe hartslagmeter en geïntegreerde GPS Polar smartwatches kunnen gebruik maken van een aantal platformen voor visualisatie van trainingsdata. [26] De polar beat applicatie wordt gebruikt om data gegeneerd door de polar toestellen te kunnen exporteren naar csv formaat. Deze data is beschikbaar onder 2 vormen: session data en raw data. Session data bestaat uit de hartslag, tijd in iedere hartslagzone, afstand, snelheid, afstand in iedere snelheidszone, training load en recovery time. Onbewerkte data bestaat uit samples per seconde met hartslag, snelheid, afstand, versnelling en running cadence. Met de polar M600 is het echter jammer genoeg niet mogelijk om de onbewerkte data te extraheren. [24] Polal flow is de web equivalent hiervan en biedt dezelfde functionaliteit als polar beat.

Figure 2.1: assenstelsel smartwatch



Het assenstelsel van een smartwatch ziet eruit zoals op bovenstaande figuur. Dit is nuttige info voor de analyse van accelerometer data (zie volgend hoofdstuk).

2.2 Google fit

[4] Google fit bestaat uit een aantal componenten. De fitness store slaat data op van verschillende toestellen en applicaties. Google fit is een cloud service. Via de google fit APIs kunnen verschillende high level representaties van elementen bekomen worden. Werken met de fitstore is hierdoor gemakkelijk. Data sources zijn één van deze high level representaties, deze stellen sensoren voor. Data types, een andere high level representatie, bevatten info over het soort data (aantal stappen, hartslag..). Data points zijn een array van waarden met een bepaald data type, komende van een data source. Datasets zijn data punten van hetzelfde type van dezelfde source binnen een bepaald tijdsinterval. Sessions stellen een tijdsinterval voor waarin de gebruiker een activiteit beoefent. Fitness data is zeer gebruiker specifiek, hiervoor moet dus toestemming gevraagd worden. Dit wordt door het framework geautomatiseerd.

2.3 Scikit learn

Scikit learn is een machine learning library voor python. Hierin zijn diverse algoritmes geïmplementeerd voor classificatie, regressie, unsupervised learning enz. Ook is het mogelijk om de preprocessing functionaliteit te gebruik. Dit bevat onder andere normalisatie implementaties, standaard schalers enz. Algoritmes voor de automatisering van alle stappen in het data analyse proces zijn aanwezig (dimensionality reduction, Feature selection, feature extraction en feature engineering).

2.4 Google SignIn

In deze thesis wordt Google SignIn gebruikt als middel om authenticatie mogelijk te maken. Via deze weg kan de gebruiker inloggen met zijn/haar google account. Deze service behoort tot de google APIs. Er moet hiervoor dus een google API console project aangemaakt worden. Indien authenticatie met een backend server mogelijk moet zijn, is een OAuth 2.0 client ID vereist. Deze moet bij de Google SignIn options meegegeven worden. Voor authenticatie met Firebase is dit bijvoorbeeld nodig.

2.4.1 OAuth

[48] OAuth 2.0 is het standaard protocol voor authorizatie. Dit protocol zorgt voor (gelimiteerde) toegang van third party applicaties naar een backend server. Er zijn 4 rollen in OAuth. De eerste is resource owner, dit is de insantie die de informatie aanvraagt. Resource server is de server die uw persoonlijke data beheert. Cliënt is de applicatie die toegang vraagt tot de server. Authorisation server is de server die access tokens uitdeelt aan de clients. resource en auth server kunnen dezelfde zijn. Er wordt gebruik gemaakt van Access tokens en refresh tokens. Een access token geeft de toegang en moet

vertrouwelijk gehouden worden. Refresh tokens worden gebruikt om access token te vernieuwen aangezien deze een beperkte levensduur hebben.

2.5 Google Activity Recognition API

Via deze interface is het mogelijk om te detecteren welke activiteit de gebruiker uitoefent. Er zijn 2 APIs: de activity recognition transition API en de activity recognition sampling API. The activity recognition transition API heeft betere accuraatheid en vergt minder energie van het toestel. Er wordt een notificatie gegenereerd als de gebruiker zijn/haar activiteit verandert. De API call requestActivityTransitionUpdates heeft als eerste parameter een activityTransactionRequest waarin alle activiteiten in vermeld staan die belangrijk zijn (bv VEHICLE, ENTER). De 2e parameter is een pendingIntent. Een pendingintent is een actie die op een later tijdstip kan afgehandeld worden, in dit geval door de activity recognition client. Een pending intent bekomen kan via de getService aanroep. De gewenste functionaliteit moet dus beschreven zijn in een service.

The activity recognition sampling API wordt gebruikt als de sampling periode moet gecontroleerd worden of als ongefilterde data gewenst is. Via de detectionInterval parameter in de requestActivityUpdates API call kan het energieverbruik gecontroleerd worden. Hoe groter de samplingperiode, hoe minder energieverbruik. Individuele voorspellingen kunnen ruis bevatten. Het is dan ook aangewezen om een eigen filter te schrijven dat hiermee kan omgaan. Ook hier is de 2e parameter een pendingIntent. De verschillende activiteiten die kunnen gedetecteerd worden zijn: IN_VEHICILE, ON_FOOT, RUNNING, WALKING, ON_BICYCLE, STILL. Ook is telkens de waarschijnlijkheid beschikbaar.

2.6 Wearable Data Layer API

[3] Met behulp van Wear OS kan een smartwatch direct communiceren met een netwerk zonder gekoppeld te zijn aan een mobiel toestel. Indien er een connectie met een gsm is via bluetooth zal het netwerkverkeer geproxied worden langs deze weg. In het andere geval maakt de smartwatch gebruik van Wi-Fi en cellulaire netwerken. De Wearable Data Layer API biedt een een afzonderlijk communicatie kanaal tussen applicaties aan. De API bevat een aantal data objecten die communicatie mogelijk maken. Een Data Item slaat info op en zorgt voor automatische syncing tussen gsm en smartwatch. Een Asset zorgt voor verzending van binaire blobs van data. Een messageClient verzendt berichten, de 2 apparaten moeten hiervoor geconnecteerd zijn. Een ChannelClient wordt gebruikt voor verzenden van grotere bestanden. Deze thesis maakt enkel gebruik van de messageClient. Hiervoor is bluetooth connectie noodzakelijk.

2.7 Firebase

Firebase is een NoSQL document georiënteerde databank, data wordt opgeslagen in documenten die samen zitten in collections. Elk document heeft key-value pairs. De collections en documents worden impliciet gecreëerd, cloud firestore doet dit voor u. Cloud Firestore maakt gebruik van NoSQL en is dus schemaless.

2.8 Pandas

[18] Pandas is een data analyse library voor python. Het maakt gebruik van dataframe objecten voor data manipulatie. Deze datastructuur maakt het mogelijk om de data voor te stellen in rijen van observaties en kolommen van variabelen.

Chapter 3

Rope skipping

In deze studie wordt gefocust op rope skippen als middel om beweging toegankelijk te maken. Rope skipping is een ideale sport om de conditie te trainen. Aan de hand van de herkenning en classificatie van rope skipping bewegingen, met gelijktijdige monitoring van hartslag data, kan een schatting gemaakt worden van het inspanningsniveau. Gebruikers krijgen persoonlijke aanbevelingen te zien om een optimale conditietraining te verzekeren. minder frequente en/of minder lange trainingsessies zullen aangeboden worden bij onervaren rope skippers. Gebruikers worden aangemoedigd om bewegingen waarin fouten gemaakt werden meer in te oefenen. In wat volgt wordt beschreven hoe bovenstaande technieken verwezenlijkt worden. Machine learning ligt hierbij aan de basis. Verschillende machine learning algoritmen komen dan ook uitvoerig aan bod alsook voorafgaande preprocessing van de data. Om nuttige statistieken te kunnen tonen, is data manipulatie van onbewerkte accelerometer data essentieel.

3.1 Bewegingen

Dit onderdeel heeft als doel de onderzochte bewegingen toe te lichten. De bewegingen op zich worden uitgelegd aan de hand van afbeeldingen. Om een zo correct mogelijk beeld te hebben van deze bewegingen, wordt gewerkt met data afkomstig van verschillende proefpersonen met verschillend vaardigheidsniveau. De gelijkenissen en verschillen tussen bewegingen worden aangekaart. Er wordt telkens gebruik gemaakt van een 10 seconden interval tijdens de analyse.

3.1.1 Springen met tussensprong

Bij springen met tussensprong zal er met iedere draaiing van het touw twee keer gesprongen worden. In bovenstaande figuren is het verloop te zien van de verschillende accelerometerassen tijdens het springen. Er is ook een duidelijk periodiek verloop merkbaar. 1 periode staat gelijk aan 1 draaiing. Door de tussensprong is de periode op alle drie de grafieken schijnbaar opgedeeld in 2

Figure 3.1: accelerometer signaal van springen met tussensprong

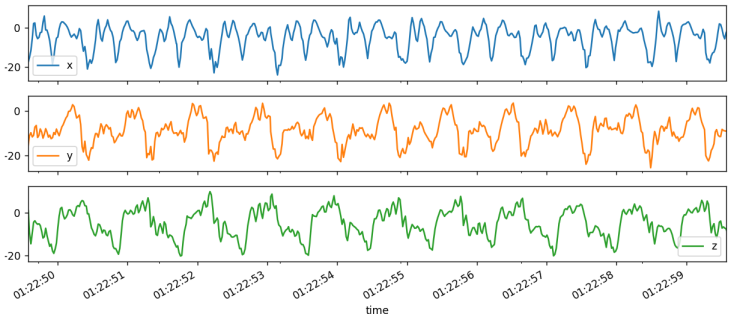
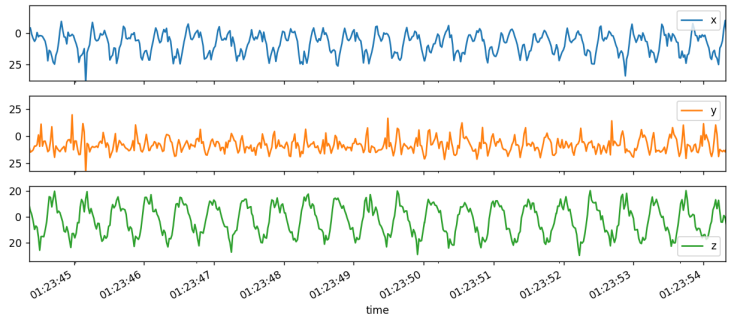


Figure 3.2: accelerometer signaal van springen zonder tussensprong



stukken. Een tussensprong veroorzaakt namelijk een extra beweging langs alle drie de assen. Ook is de versnelling grotendeels negatief wat aangeeft dat er een grotere snelheidsverandering is bij beweging in de richting van de negatieve assen.

3.1.2 springen zonder tussenprong

Bij deze beweging vindt er bij elke draaiing van het touw 1 sprong plaats in plaats van 2. Dit resulteert in een kleinere periode en grotere frequentie. Een merkbaar verschil is dat de amplitude minder groot is. De gemaakte bewegingen zijn bijgevolg kleiner om sneller rond te geraken. Het signaal op de y-as is eveneens veel grilliger. Deze as staat namelijk loodrecht op de richting van de arm in hetzelfde vlak en door de snellere bewegingen verandert de richting van de versnellingsvector langs deze as sneller. Het signaal op de x-as is ook weer voor het grootste deel negatief. Dit is niet het geval voor de andere assen.

3.1.3 Double under

3.1.4 Cross over

Figure 3.4: accelerometer signaal van cross over

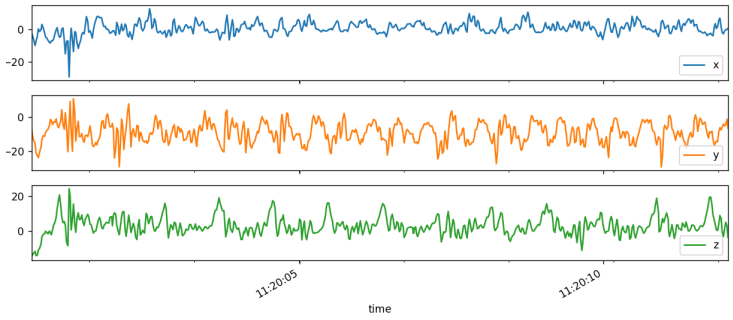


Figure 3.3: cross over

Cross over is een beweging waarbij tijdens het springen de armen gekruist worden. Dit is te zien op bovenstaande figuur. Op bovenstaande figuur is duidelijk te zien dat de amplitude langs de x as veel kleiner is. De x as loopt namelijk evenwijdig met de arm en aangezien de armen in een nogal geklemde positie zitten is beweging langs deze as moeilijker. Verder is er ook weer een periodiek verloop merkbaar, al is dit wat minder uitgesproken. De data is namelijk afkomstig van een beginnend rope skipper. X- en y-as ondergaan vooral een versnelling in negatieve richting. De z-as daarentegen ondergaat doorgaans een positieve versnelling.

3.1.5 Side swing

Side swing is een beweging waarbij de uitvoerder het touw 1 maal rechts van hem/haar ronddraait en meteen hierna 1 maal links. Er kan ook voor gekozen worden om links te beginnen en rechts te eindigen. Ook in deze data is een periodiek verloop merkbaar. De verandering in snelheid langs de verschillende assen verloopt trager. De armen en polsen zijn tijdens deze beweging namelijk compleet vrij zodat op elke as duidelijke veranderingen in richting zichtbaar zijn. Aan de waarden is te zien dat deze kleiner zijn. Dit komt omdat side swing een rustige beweging is met minder bruuske snelheidsveranderingen. Ook dit

Figure 3.5: side swing

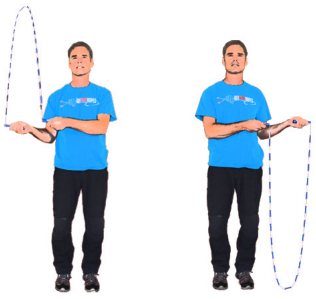
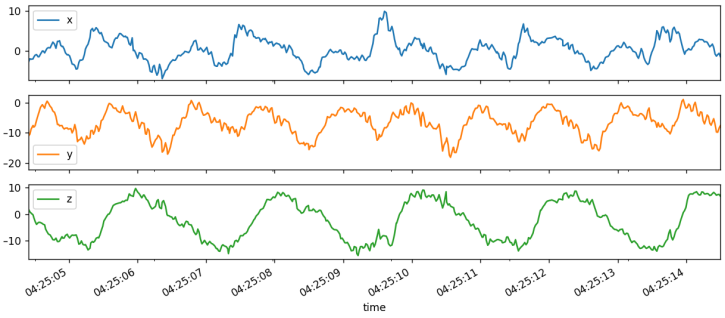


Figure 3.6: accelerometer signaal van side swing

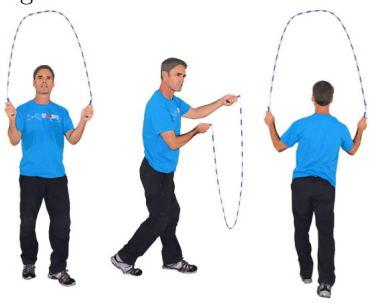


verloop is in 2 stukken verdeeld op x- en y-as
doordat dezelfde beweging langs beide kanten van het lichaam moet uitgevoerd worden. De z-as daarentegen vertoond een ander soort patroon. Deze as staat loodrecht op de pols, de versnellingsvector volgens deze as keert bij transitie niet volledig om.

3.1.6 Forward 180

Deze beweging zorgt ervoor, zoals de naam zegt, dat de uitvoerder zichzelf 180 graden draait tijdens het springen. Dit wordt verwezenlijkt door met behulp van een side swing tijdens het draaien het touw op de juiste plaats te krijgen. Wanneer men gedraaid is kan achterwaarts verder gesprongen worden. Het meten van deze beweging was iets moeilijker aangezien dit niet constant achter elkaar kan uitgevoerd worden. Een oplossing was om na iedere achterwaartse sprong, het touw van richting te veranderen. Het periodiek verloop is terug zichtbaar doordat dezelfde beweging telkens opnieuw werd uitgevoerd. De ruis die te zien is tussen 2 perioden door (vooral langs de y-as) heeft te maken met het feit dat na 1 beweging, zoals vermeld, van richting moet veranderd worden De handen moeten bijgevolg terug in positie gebracht worden. Ook hier is de grafiek weer in 2 delen verdeeld. Er zijn tijdens deze beweging eveneens 2 sprongen nodig met daartussen een side swing als overgangsbeweging. De richting van de versnellingsvector is hoofdzakelijk negatief langs de x- en y-as.

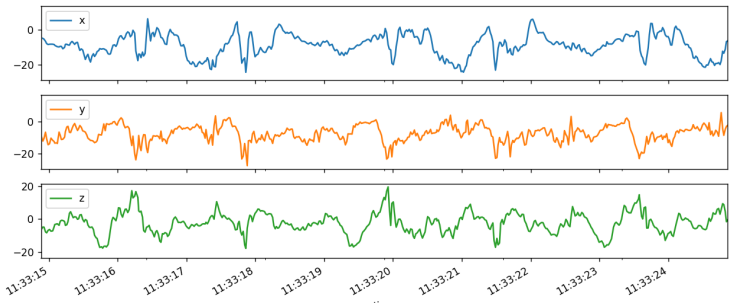
Figure 3.7: forward 180



3.2 Android applicaties

[27]

Figure 3.8: accelerometer signaal van forward 180



Het polar flow platform laat niet toe om voor een sessie de onbewerkte accelerometerdata te downloaden. Hierdoor werd een eigen applicatie ontwikkeld op de smartwatch en gsm. De smartwatch applicatie zal via sensoren de accelerometer data opvangen. Dit met een sample frequentie van ongeveer 52 Hz. Dit is echter zeer systeem en tijd afhankelijk. Het android systeem en/of andere applicaties kunnen de frequentie aanpassen. De sampling frequentie kan, afhankelijk van de lifecycle waarin de applicatie zit, zelfs verlaagd worden tot 18 Hz. Het aantal datapunten per seconde is daarom dus niet stabiel. Deze sampling configuratie wordt bekomen met de 'normal' delay. Andere mogelijkheden zijn 'fastest', 'UI' en 'game'. Er wordt via een knop een sessie gestart, hierna begint de data verzameling. Telkens er een accelerometer datapunt binnenkomt wordt dit via de wear OS messageClient verzonden. De datapunten worden op deze manier verzonden omdat ze dan via een applicatie op de android smartphone kunnen opgeslagen worden naar een bestand. Dit is namelijk efficiënter op een smartphone als op een smartwatch. Door een andere knop kan de sessie beëindigd worden. Het bestand met accelerometer data kan dan teruggevonden worden in de interne opslag van de gsm. Elke applicatie heeft hier in de android/data folder een eigen map. Deze wordt gebruikt voor het opslaan van de bestanden.

3.3 preprocessing

De data komende van de smartwatch applicatie bestaat uit 4 kolommen: tijd (ns), x-, y- en z-component van de versnelling (m/s). Dit zonder enige wijziging meegeven met een machine learning algoritme zal geen goede resultaten geven. Machines begrijpen onbewerkte data namelijk niet. Data preprocessing biedt hier een oplossing voor. Data preprocessing is de aaneenschakeling van stappen die de data zo transformeren dat de features ervan kunnen geïnterpreteerd worden door het algoritme. Een dataset is een collectie van data objecten, ook wel records/punten/vectoren.. genoemd. Data objecten worden beschreven met een aantal features die de basis karakteristieken van het object weergeven. Een feature is een individueel meetbare karakteristiek van het event dat zich vo-

ordeed. Bij het bepalen van een activiteit uit accelerometer data worden als features de x- , y- en z-component van het signaal genomen. Dit zijn namelijk de waarden die het signaal beschrijven. Er zijn verschillende types van features. Een eerste onderverdeling wordt gedaan op basis van het al dan niet numeriek zijn van het feature. Een categorische feature is discreet en heeft dus slechts een beperkt aantal mogelijke waarden. Een feature die categorisch is kan nominaal of ordinaal zijn. De aanwezigheid van een zekere ordening onderscheid deze van elkaar. Een numerieke feature is continue en wordt gerepresenteerd door nummers. Deze waarden kunnen een interval of een ratio beschrijven. Bij ratio is er sprake van een true zero, wat niet het geval is voor interval waarden. Negatieve getallen zijn niet mogelijk in geval van een ratio feature. Accelerometer data is bijgevolg een numeriek feature van het type interval. Dit betekent dat er bij deze data sprake is van ordening en dat de verschillen tussen datapunten een betekenis hebben. Het ratio van 2 datapunten heeft dan weer geen betekenis door de afwezigheid van een true zero.

Niet alle stappen van data preprocessing zijn toepasbaar op elk probleem. Dit is zeer afhankelijk van het soort data waarmee gewerkt wordt. In wat volgt worden de stappen die doorgaans ondernomen worden bij preprocessing besproken. Telkens zal vermeld worden of dit toepasbaar is op de preprocessing van accelerometer data.

3.3.1 Data Quality Assessment

Met onbewerkte data van eender welke oorsprong kunnen veel zaken verkeerd zijn met betrekking tot datatype, ontbrekende waarden enz. Een eerste preprocessing stap is het type van de tijd feature converteren naar een datetime object. Op die manier kan met de effectieve tijdsverschillen gewerkt worden. Door de onstabiele samplingfrequentie is het aantal samples per seconde onzeker. Daarom is het beter om met tijdsverschillen te werken en alle samples in een tijdsinterval te laten horen bij hetzelfde segment.

Om de data corresponderend met het aan- en uitzetten van de smartwatch tijdens een meting eruit te kunnen filteren, worden de eerste en laatste datapunten in een interval van 3 seconden verwijderd.

Data komt meestal van verschillende bronnen die al dan niet betrouwbaar zijn. De data zal in dat geval dus ook in verschillende formaten binnenkomen. Bij dit experiment wordt slechts 1 sensor gebruikt. Deze complicatie is dus niet aanwezig. De betrouwbaarheid van de sensor kan echter wel nog in vraag gesteld worden.

Bij observatie van de data waarden zijn namelijk duplicaten merkbaar. Dit is een fout in het datacollectie proces dat te wijten is aan het niet idempotent zijn van de ontwikkelde android applicatie. Dit is echter geen probleem aangezien duplicaten via het pandas framework makkelijk kunnen gedetecteerd en verwijderd worden.

Duplicaten worden verwijderd om deze data objecten geen voordeel of bias te geven bij machine learning algoritmen. Eveneens zal meer dan 1 record met dezelfde data een slecht visueel beeld geven bij plotting.

Ontbrekende waarden werden niet geobserveerd, maar hier moet het systeem toch tegen bestand zijn. Missing values komen voor door fouten tijdens de data collectie of door een data validatie regel waardoor bepaalde punten niet geldig zijn. Een eerste manier om hiermee om te gaan is volledige rijen met ontbrekende waarden verwijderen. Als veel datapunten incompleet zijn is deze strategie niet aan te raden. Als slechts een klein percentage van de data punten ermee te kampen heeft dan kan ook gekozen worden voor interpolatie methoden. Hierbij zal de waarde geschat worden aan de hand van de andere beschikbare waarden voor die feature. Men kan ook kiezen om de ontbrekende waarde in te vullen met het gemiddelde, de mediaan of de modus van de feature. Data verzameling door sensoren in een smartwatch vergt geen menselijke interactie, waardoor de kans op ontbrekende waarden klein is. Toch is het aangewezen om het zekere voor het onzekere te nemen. NaN waares worden daarom vervangen door een geinterpoleerde waarde van de feature waartoe de NaN behoort. Deze manier is beter dan verwijderen, want zo kan veel data verloren gaan. Ook is het gemiddelde of de mediaan niet zo'n goede keuze aangezien dit een vertekent beeld kan geven. De datapunten hebben vaak een groot bereik waardoor een gemiddelde waarde opgeven tijdens een piek een grote daling met vervolgens een stijging zal veroorzaken.

Een laatste mogelijk probleem waarmee data te kampen heeft zijn inconsistente waarden. Om dit te detecteren is het nodig te weten welk datatype een bepaalde feature heeft en of dit hetzelfde is voor alle data objecten. Dit komt vaak voor bij menselijke fouten en is dus hier niet van toepassing. Toch worden enige voorzorgen genomen door ook in dit experiment de x, y en z kolom te geconverteerd naar float objecten.

3.3.2 feature aggregation

De resulterende dataset wordt onderverdeeld in segmenten van 1 seconde elk met 50 procent overlap. 1 volledige sprong zal namelijk gemiddeld iets minder dan 1 seconde duren. Het is echter niet geweten of de beweging langer dan 1 sprong zal duren. Een interval van minder dan 1 seconde zal de accuraatheid in gedrang brengen. Vandaar de keuze voor een segmentgrootte van 1 seconde. Partiële segmenten waarin onvoldoende datapunten aanwezig zijn, worden genegeerd. Deze aggregatie wordt gedaan omdat uit 1 enkel datapunt te weinig info kan gehaald worden. Het verloop in de tijd is belangrijk. Aggregaties bieden namelijk een high level view van de data die stabieler is dan 1 individueel data object. Door het samennemen van grote hoeveelheden data objecten tot 1 samenvattend data object is er ook een vermindering van geheugen consumptie en processortijd. De bekomen segmenten moeten als laatste stap gelabeld worden. Het targetlabel in geval van activity recognition is de uitgevoerde beweging.

3.3.3 feature sampling

Met sampling kan de dataset gereduceerd worden waarbij een beter maar duurder machine learning algoritme kan gebruikt worden. Sampling moet zo gedaan

worden dat de eigenschappen van de originele dataset behouden worden: de samples zijn representatief. Dit wordt bekomen door de juiste sample grootte en sampling strategie te kiezen. Bij simple random sampling wordt een gelijke waarschijnlijkheid van selecteren van een entiteit verondersteld. Hierbij zijn 2 variaties: zonder vervanging en met vervanging. simple random sampling zonder vervanging verwijdert een geselecteerd data object uit de dataset. simple random sampling met vervanging steekt dit object terug zodat het bij volgende sampling iteraties kans heeft om nog eens gekozen te worden. Bij ongebalanceerde datasets is de simple random sampling techniek niet goed. Dit betekent namelijk dat de zeldzame data objecten evenveel kans hebben om geselecteerd te worden als de andere. De bekomen dataset zal dus niet meer overeenkomen met de originele. Stratified sampling daarentegen houdt rekening met deze distributie. Deze techniek verdeelt de dataset in groepen en neemt uit elke groep een gelijk aantal objecten ook al is de grootte verschillend.

Sampling is niet toepasbaar bij activity recognition op basis van accelerometer data. Hoe groter de dataset en hoe meer variatie hierin, hoe accurater de output van het machine learning algoritme zal zijn.

3.3.4 feature extraction

Enkel de x-, y- en z-component van een acceleratie-vector is voor de meeste algoritmen niet genoeg om correcte voorspellingen te kunnen maken. Daarom worden hier, op basis van de datapunten in een segment, nieuwe features ontwikkeld. Hierbij horen onder andere statistische features zoals het gemiddelde, de mediaan, het minimum, maximum.. Een aantal complexere features worden in wat volgt meer toegelicht.

Signal Magnitude Vector

SMV wordt gebruikt om de graad van bewegingsintensiteit te evalueren. Dit kan een onderscheid leveren tussen verschillende bewegingen op vlak van intensiteitsniveau. Met volgende formule wordt dit berekend:

$$\sum \sqrt{x^2 + y^2 + z^2}$$

Signal Magnitude Area

SMA wordt gebruikt om een meetwaarde te bekomen die het niveau van activiteit weergeeft en dus het onderscheid kan maken tussen perioden van activiteit en inactiviteit. Het is een statistische meetwaarde die de omvang (magnitude) weergeeft van een veranderende feature. Magnitude is een maat die ordering aangeeft. In dit geval geeft SMA dus een ordering aan tussen de verschillende segmenten. SMA wordt berekend door de genormaliseerde integraal te nemen van het signaal. Praktisch is het moeilijk om een integraal te berekenen in python. Daarom wordt gekozen voor een benaderende berekening. Per

tijdstip wordt de absolute waarde van de x-, y- en z-component opgeteld, deze waarde wordt opgeteld bij een accumulator.

$$\frac{\sum |x| + |y| + |z|}{aantaldatapunten}$$

tilt angle

Tilt angle is de hoek die een vector maakt met de x-as. Dit wordt berekend met volgende formule

$$cos\alpha = \frac{ab}{|a||b|}$$

Fourrier transformatie

De fourrier transformatie zal het signaal voorstellen in functie van de frequentie in plaats van de tijd. Uit deze representatie kunnen een aantal zinvolle features gehaald worden.

Power Spectral Density

PSD is een maat voor de kracht-inhoud in functie van de frequentie. Het toont de kracht van de variaties (energie) als functie van de frequentie. Anders gezegd, op welke frequentie zijn variaties krachtig en op welke minder.

Entropy

Entropy is een maat voor de wanorde in een fysisch systeem. De entropy groeit als het aantal mogelijke staten in een systeem groeit. Entropy is ook een maat voor de verwachte hoeveelheid informatie. Dit wordt shannon entropy genoemd, Maximum likelihood. Random variables zijn onzeker want ze kunnen eender welke waarde aannemen (continu signaal) of specifieke waarden met een zekere waarschijnlijkheid (discreet signaal). Self information/surprisal is de observatie van een unlikely outcom van een random variabele. Er werd meer info verwerft over het gedrag van de random variabele. Shannon entropy is de verwachte waarde van de self information van een random variabele. Voor een discrete variabele is dit de gewogen som van de self information geassocieerd met elke mogelijke output. De gewichten zijn de kans op voorkomen. Voor een continue variabele wordt de som vervangen door een integraal. Continue variabelen met een brede distributie hebben grotere entropie.

Andere entropie maten zijn: joint entropy (entropie van 2 random variabelen die samen voorkomen), conditional entropy (entropie van een random variabele afhankelijk van een andere), relative entropy (meet de afstand tussen 2 probabiliteitsfuncties) en cross-entropy (verwachte self information in respect to de probabiliteitsfunctie van een andere random variabele).

Entropy wordt ook gebruikt bij het analyseren van de relaties tussen data features. Dit wordt meestal gedaan met de pearson correlation wat een lineaire

relatie voorstelt tussen 2 features. Als een paar features geen relatie hebben wil dit niet zeggen dat ze niet op 1 of andere manier verwant zijn. Om niet lineaire afhankelijkheid te detecteren wordt mutual information gebruikt. Dit meet de hoeveelheid informatie die een variabele bevat over een andere. Dus de vermindering in onzekerheid van een variabele door de kennis van een andere.

Bij feature selection wordt eveneens gebruikt gemaakt van entropy. Sommige features zijn zeer gecorreleerd, andere dragen enkel ruis en bijna geen signaal. Hogere dimensionale vectoren lijken ook meer en meer op elkaar naarmate hun dimensionale ruimte vergroot (curse of dimensionality). Feature selection detecteert relevante features zodat de dimensionaliteit niet onnodig groot is.

Subset extraction / sampling maakt gebruik van entropy bij bijvoorbeeld het splitsen van de dataset in een training en test deel. De training set moet een goede representatie hebben van de distributie van de originele data set. Meestal werkt random sampling goed, maar het kan zijn dat een verdoken bias in de originele dataset leidt tot inaccurate samples. Om hiermee om te gaan wordt rekening gehouden met de entropy van de origineel dataset. Als deze overeenstemt met het sample dan is de splitsing geslaagd.

Energy

3.3.5 Feature selection

3.3.6 Dimensionality reduction

Na feature extraction is een groot aantal features aanwezig. Dit is voor veel algoritmes een te groot aantal. Daarom worden deze features via PCA gemapt op een ruimte met minder grote dimensies. Er wordt gekozen voor 6 componenten.

Vaak hebben datasets een groot aantal features. Een image processing probleem kan bijvoorbeeld duizenden features hebben. Dimensionality reduction probeert het aantal featuren te verminderen, maar niet door simpelweg een subset te nemen (feature subset selection). Hoe hoger het aantal dimensies, hoe complexer de dataset. Datasets kunnen voorgesteld worden in een assenstelsel met het aantal assen gelijk aan het aantal dimensies. Een groot aantal dimensies is moeilijk te modellen en te visualiseren. Dimensionality reduction mapt de dataset op een lagere dimensionele ruimte. Dit wordt gedaan door nieuwe features te creëren die een combinatie zijn van de oude features. De techniek die hiervoor gebruikt wordt is Principle Component Analysis. Data analyse algoritmes werken beter met lagere dimensionaliteit. Irrelevante features en noise zijn geëlimineerd. Dit resulteert in een dataset met lage dimensionaliteit, wat makkelijker te interpreteren en te visualiseren is.

3.3.7 Feature encoding

Het doel van data preprocessing is om de data te encoderen zodat het naar een staat kan gebracht worden die de machine begrijpt. Er zijn algemene normen en regels die gevolgd worden bij feature encoding. Nominale features kunnen een 1 op 1 mapping ondergaan zoals bijvoorbeeld een permutatie van waarden (one

hot encoding). Ordinale features kunnen een rangbehoudende verandering van waarden ondergaan. Interval features kunnen getransformeerd worden met een wiskundige formule waarbij hun nulpunt behouden wordt (fahrenheit naar celsius). Ratio features kunnen geschaald worden met wiskundige formules (lengte naar meters of feet...). Accelerometer data bestaat uit interval features zoals eerder gezegd. Hierop kan dus normalisatie of een andere schaling toegepast worden (zie verder).

Algoritmes geïmplementeerd in scikit learn hebben vaak gestandaardiseerde data nodig, data die lijkt op een standaard normale distributie: gemiddelde gelijk aan 0 en variantie gelijk aan 1. In de praktijk wordt de data getransformeerd door het gemiddelde te verwijderen en het dan te schalen door deling van de standaard deviatie. Dit is nodig omdat een feature met een variantie niet in dezelfde order als een andere feature, dominant kan zijn. De preprocessing module van scikit learn heeft een klasse standardScalar die de transformer API implementeert. Deze berekent het gemiddelde en standaard deviatie op een training set om dan dezelfde transformatie te kunnen doen op de test data. Standardisatie gaat zoals gezegd per feature de datapunten schalen. In het geval van detectie gebaseerd op accelerometer data is dit niet wenselijk omdat het gemiddelde en variantie van elke nieuwe dataset die op het model wordt toegepast steeds anders zal zijn. De nieuwe datapunten worden dus geschaald met getallen die niks met die distributie te maken hebben. Toch is er een zeker schaling nodig. Bij het herkennen van activiteiten moeten individuele datapunten met elkaar vergeleken worden. Om geen appelen met peren te vergelijken, moeten de datapunten zo getransformeerd worden dat ze een gelijke distributie hebben. Dit kan verwezenlijkt worden door elk datapunt te normaliseren.

3.3.8 Feature engineering

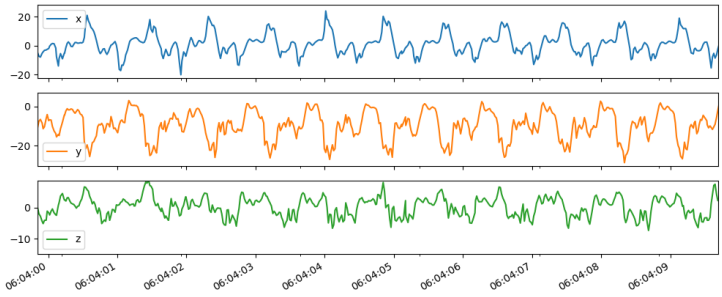
3.3.9 Balance data

Omdat het model geen voorkeur mag ontwikkelen voor een bepaalde klasse, moet de data gebalanceerd zijn. Dit wil zeggen dat er van elke klasse evenveel samples moeten aanwezig zijn. Indien de data ongebalanceerd is en hier wordt geen rekening mee gehouden dat zou het model elke klasse kunnen bestempelen met hetzelfde label en toch meer dan 50% accuraatheid krijgen. Elke klasse beweging heeft variaties zoals een andere draairichting van het touw en de smartwatch dragen op verschillende polsen. Data afkomstig van deze variaties worden eveneens gebalanceerd. Alsook de data afkomstig van verschillende proefpersonen.

3.3.10 Train/validation/test split

Machine learning algoritmes moeten eerst getraind worden, daarna gevalideerd en getest. Op basis van de training data wordt het model gebouwd. Hierbij moet opgepast worden voor overfitting. Dit is het verschijnsel waarbij het model perfect traint op de training data, maar hierdoor realistische data niet correct

Figure 3.9: andere pols



kan classificeren. Underfitting is het omgekeerde en doet zich voor wanneer te grote tolerantie gegeven wordt aan misclassificaties. De validatie data wordt gebruikt om de hyperparameters van het model (window grootte bijvoorbeeld) te kiezen en te verbeteren. De test data heeft als functie om het getrainde en gevalideerde model te testen. Het split ratio is afhankelijk van de dataset en het type model. Als veel training nodig is, dan is het training data gedeelte groter. Zijn er veel hyperparameters die moeten getuned worden, dan is de validation data groter.

Het model zal goed moeten getraind worden, dus een grote training dataset is gewenst. Er zijn (afhankelijk van het gebruikte algoritme) een aantal hyperparameters. Window grootte is hier sowieso 1 van, onafhankelijk van het gebruikte algoritme. Om over/underfitting te voorkomen moet een testgedeelte aanwezig zijn. Hiermee kan bekeken worden hoe goed het model presteert met nieuwe data.

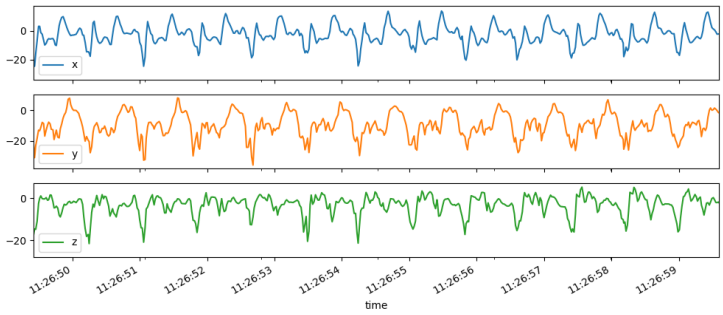
3.4 Soorten datasets

eenzelfde beweging kan op verschillende manieren gemeten worden. Zo kan het onderscheid gemaakt worden tussen de pols waaraan de smartwatch gedragen wordt of tussen de draairichting. Deze variaties includeren in de training data zal resulteren in een beter model. Aangezien deze variaties eigenlijk dezelfde beweging voorstellen. In wat volgt wordt de data verder geanalyseerd.

3.4.1 Pols

Door de smartwatch aan een andere pols te dragen worden de x en y as omgekeerd. Ook zullen ze onder een iets andere hoek komen te staan. De z-component zal enkel onder een andere hoek komen te staan en niet omkeren. Dit resulteert in andere meetwaarden en dus een andere grafiekvorm.

Figure 3.10: andere draairichting



3.4.2 Draairichting

Een andere draairichting zal de versnellingsvector zelf omkeren terwijl de assen gelijk blijven.

3.5 Machine learning algoritmes

[13] [30] [17] Machine learning is een modeling techniek gebaseerd op data. Het systeem heeft als input een training set waaruit de model afgeleid wordt. Diversiteit van de training dataset is belangrijk. Een training dataset bestaande uit enkel geschreven nota's van 1 persoon zal de cijfers/letters van een ander handschrift niet herkennen. De training dataset moet de karakteristieken van de field data reflecteren. Veralgemening is het proces waarmee de performantie van een model constant gemaakt wordt onafhankelijk van de training dataset of field dataset. Overfitting doet zich voor wanneer geen rekening gehouden wordt met ruis in de training dataset. Het bekomen model voldoet dan niet aan de werkelijkheid. Er zijn verschillende types machine learning: supervised learning, unsupervised learning en reinforcement learning.

Het is belangrijk om het bekomen model correct te evalueren. Zo is geweten of nog meer data, verdere tuning nodig is. Er wordt een score gegeven aan het model. Hier zijn verschillende metrieken voor beschikbaar. Accuracy is het ratio van correct geclassificeerde datapunten en het totaal aantal datapunten. Dit kan problemen geven omdat er evenveel gewicht gegeven wordt aan elk soort error. Recall is het ratio van het totaal aantal correct geclassificeerde positieve samples en het totaal aantal positieve samples. Een positief geclassificeerd datapunt (true positive) is een datapunt dat wordt bestempeld met het juiste label. Precision is het ratio van het aantal correct geclassificeerde positieve samples en het totaal aantal voorspelde positieve samples. Vergelijken van machine learning algoritmes wordt gedaan met confusion matrices. Deze matrices geven een beeld van het aantal true positives/negatives en false positives/negatives.

Voor de implementatie van de algoritmes werd grotendeels scikit learn gebruikt. Enkel voor CNN wordt gebruik gemaakt van tensorflow.

3.5.1 Support Vector Classification

[34] SVM zorgt voor het maken van een decision boundary tussen de verschillende klassen. In het geval van lineaire scheiding moet ervoor gezorgd worden dat de afstand tussen de boundary en het dichtstbijzijnde datapunt zo groot mogelijk is. Deze decision boundary is een hyperplane in een n-dimensionale ruimte. Waarbij n het aantal features zijn die een datapunt beschrijven. Het hyperplane heeft telkens een dimensie van n-1. Er bestaan veel zulke hyperplanes maar hiervan moet degene gekozen worden met maximale marge. Bij non lineaire gevallen wordt gebruik gemaakt van 2 concepten: soft margin en kernel tricks. Soft margin wil zeggen dat er een hyperplane gekozen wordt met een aantal verkeerd geclassificeerde datapunten. Hierbij worden 2 soorten misclassificaties getolereerd. Een datapunt bevindt zich aan de verkeerde kant van het hyperplane maar wel nog binnen de marge. Een datapunt bevindt zich aan de verkeerde kant van het hyperplane en niet binnen de marge. SVM zal dus de balans moeten vinden tussen maximale marge en minimaal misgeclassificeerde datapunten. De hoeveelheid tolerantie die gekozen wordt is een belangrijke hyperparameter. Deze wordt voorgesteld door C in scikit learn. Hoe groter C, hoe meer penalty het model krijgt bij misclassificatie. De margin zal bij grote C kleiner zijn en er zullen minder support vectors zijn. Bij noisy data is het dus beter om C niet al te groot te nemen. Support Vector Classifiation doet zijn classificatie door een hyperplane te creëren die de data scheidt in klassen. Hierbij moet de afstand met de support vectors (dit zijn datapunten dichtst bij het hyperplane) maximaal gemaakt worden. De kernel wordt gebruikt om datapunten te herschalen naar een grotere dimensionale ruimte waarin het mogelijk is om de punten met een hyperplane te scheiden. Zo wordt een non-lineair descision boundary gevonden.

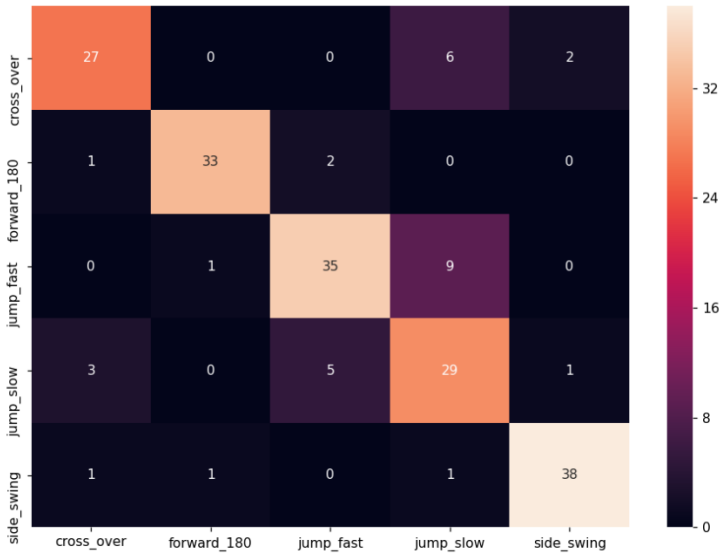
SVC implementeert het one-against-one algoritme voor multiclass classification. Classifiers worden aangemaakt die elk data van 2 klassen trainen. Er zijn dus n_class * (n_class-1)/2 classifiers nodig. De vorm van de decision functie: transformeren output functie van aantal classifiers naar (n_samples, n_classes).

3.5.2 Linear Support Vector Classification

[23] [33] [28]

Dit is vergelijkbaar met SVC met parameter kernel = 'lineair', maar het heeft meer flexibiliteit bij het kiezen van penalties en loss functies en zou beter moeten schalen naar grote aantallen samples. Support Vector Classifiation doet zijn classificatie, net zoals SVC, door een hyperplane te creëren die de data scheidt in klassen. Hierbij moet de afstand met de support vectors (dit zijn datapunten dichtst bij het hyperplane) maximaal gemaakt worden. De kernel wordt gebruikt om datapunten te herschalen naar een grotere dimensionale ruimte waarin het mogelijk is om de punten met een hyperplane te scheiden. Zo wordt een non lineaire descision boundary gevonden. LinearSVC gebruikt een lineaire kernel en vindt dus enkel lineaire decision boundaries. De loss functie, ook wel cost functie genoemd, zal aan elke classificatie een kost toewijzen. Door

Figure 3.11: confusion matrix van SVC



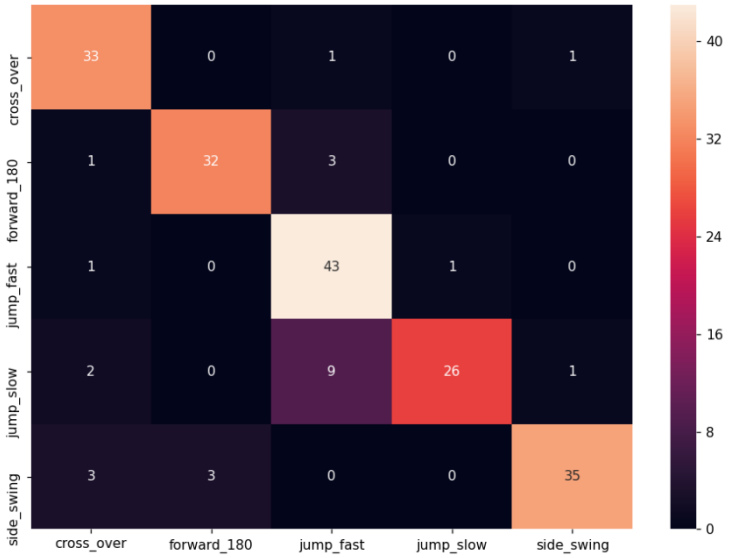
deze minimaal te maken kan een optimaal hyperplane gekozen worden waarbij de marge en dus de afstand tot de support vectors maximaal is. Door de afgeleide te nemen van deze functie kan gevonden worden in welke richting de gewichten moeten aangepast worden. De gewichten zijn de coördinaten van een vector die loodrecht op het hyperplane staat. Een regularization term kan toegevoegd worden aan de loss functie, deze zal de mate waarin loss wordt toegewezen aan verkeerd geclassificeerde datapunten aanpassen. Op die manier kan overfitting voorkomen worden.

linearSVC implementeert de one-vs-the-rest multiclass strategie, wat trainen van n$_c lassmodellenbetekent.Metoptiemulti_c lasswordteenanderemulticlassstrategiegeselecteerd.Hetisechter$

3.5.3 Random Forest Classifier

[36] Beslissingsbomen zijn de bouwstenen van het random forest model, deze bomen zullen vertakkingen opbouwen aan de hand van features. Er zijn evenveel takken als er mogelijkheden zijn voor waarden van die feature. Random Forest bestaat uit verschillende beslissingsbomen die samenwerken. Elke beslissingsboom in het forest zal een predictie doen. De klasse die het meest voorkomt bij deze predicties wordt de definitieve voorspelling van het model. Er mag geen of geen grote correlatie zijn tussen de verschillende beslissingsbomen. Is dit wel zo dan wordt de kracht van het forest teniet gedaan aangezien het zich nu zal gedragen als 1 geheel. De verschillende bomen vangen hun individuele fouten op. Random Forest zorgt ervoor dat de trees niet al te gecorreleerd zijn. Via "bagging" kan dit gerealiseerd worden. Beslissingsbomen zijn zeer gevoelig voor de data waarop ze getraind hebben. Een kleine wijziging hieraan resulteert al

Figure 3.12: confusion matrix van linearSVC

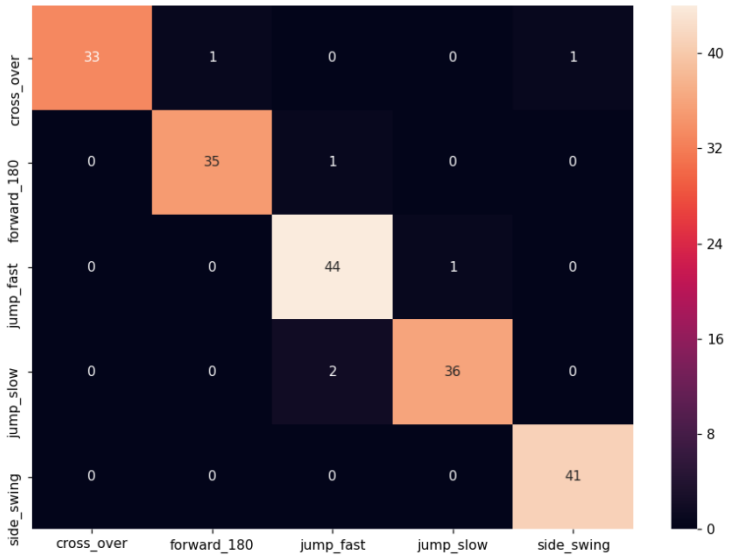


in zeer verschillende bomen. Elke boom zal random sampelen met replacement, hierdoor is de training data van elke tree anders. Feature randomness doet zich voor door bagging. Elke boom neemt, bij splitsing van een knoop, de feature die voor het meeste onderscheid zorgt. Dit zal dus ook sterk verschillen tussen bomen. De belangrijkste hyperparameters zijn de volgende. Max features geeft de random subset van features weer. Hoe lager dit getal, hoe meer vermindering in variantie en ook hoe meer bias. N_estimators stelt het aantal trees in het bos voor. Naarmate het aantal bomen toeneemt, neemt over het algemeen de fout tot een bepaald punt af. De nauwkeurigheid neemt vanaf een bepaald punt niet meer toe. Enkel de trainingsduur neemt hierbij toe. Max_depth geeft de maximale diepte van iedere boom in het forest aan. Hoe dieper een boom is, hoe meer hoe meer splits/beslissingen er gebeurd zijn en hoe meer informatie over de data aanwezig is. Min_samples_split is het minimum aantal datapunten die nodig zijn om een interne node op te splitsen.

3.5.4 AdaBoost

[35] Adaboost combineert verschillende zwakke classifiers, die slechts lichtjes beter zijn dan random guessing, in 1 sterke classifier. Elke zwakke classifier wordt getraind op een subset van de totale trainingsdata. Deze subsets mogen overlappen. Elk sample krijgt een gewicht toegewezen. Samples met een hoger gewicht hebben meer kans om gekozen te worden. Na training zal Adaboost het gewicht van misgeclassificeerde samples verhogen. Op die manier kan er meer aandacht aan besteed worden in de volgende iteratie. Classifiers met grotere accuraatheid krijgen meer gewicht. Door de resultaten en gewichten van alle

Figure 3.13: confusion matrix van random forest



classifiers in rekening te brengen, wordt het Adaboost model bekomen.

Hoe de zwakke learners geïmplementeerd worden in adaboost is afhankelijk van hyperparameter base_estimator. Vaak worden echter beslissingsbomen met 1 vertakkingsgraad (decision stumps) gebruikt. decision stumps splitsen de dataset in 2 gebaseerd op 1 feature aan de hand van een threshold.

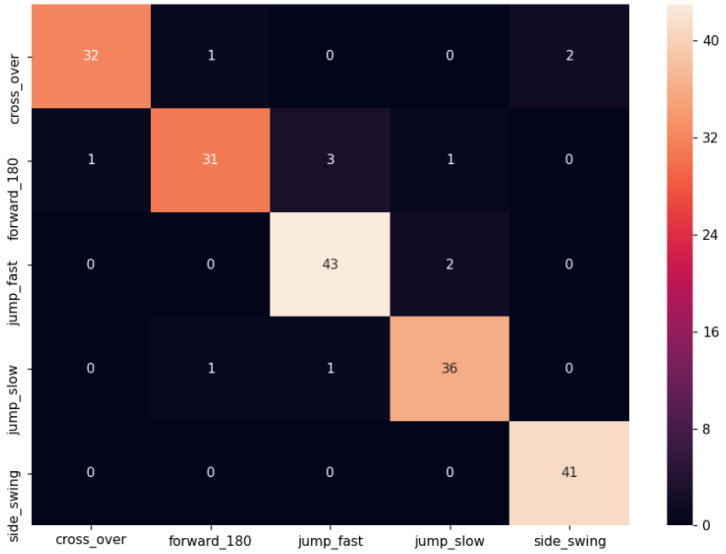
De implementatie in scikit learn kan getuned worden met de volgende hyperparameters. Het aantal zwakke learners kan aangegeven worden met de n_estimators parameter. Hoe meer er van deze aanwezig zijn, hoe complexer de decision boundary is. De learning rate zwakt de contributie van opeenvolgende iteraties af met een factor. Op die manier zal de classifier minder snel leren.

3.5.5 Naive bayes

[39] [22]

Naive Bayes is gebaseerd op Bayes theorie waarbij verondersteld wordt dat de features niet gecorreleerd zijn. Naive bayes berekent de kans dat een feature een bepaalde waarde heeft, gebaseerd op de waarde van een andere feature. De klasse met de grootste posterior probability is de voorspelling voor het datapunt. Voor elke klasse wordt dus P(c—f), waarbij c de klasse voorstelt, berekend en dit met elke mogelijke waarde van feature f. Naive Bayes is een simpel algoritme dat grote voordelen heeft. Er is weinig training data nodig om zeer snel goede resultaten af te leveren. Doordat Naive Bayes geen correlatie veronderstelt tussen de k features moet er geen rekening gehouden worden met de 2^k mogelijke feature interacties. Hierdoor is naive bayes sneller en even performant met kleine datasets in vergelijking met een complex neuraal netwerk.

Figure 3.14: confusion matrix van Adaboost



Het algoritme presteert goed met categorische variabelen aangezien voor elke waarde van een feature een berekening moet uitgevoerd worden. In geval van numerieke features wordt een normale distributie verondersteld. Een ongeziene categorische variabele zal het model echter niet kunnen voorspellen aangezien deze waarden een probabiliteit heeft van 0. Het wordt verondersteld dat de features onafhankelijk zijn, wat in het echte leven vaak niet zo is.

Parameter alpha geeft aan hoeveel smoothing moet toegepast worden op de berekening van de posterior probability. Dit zorgt ervoor dat deze kan nooit nul kan worden.

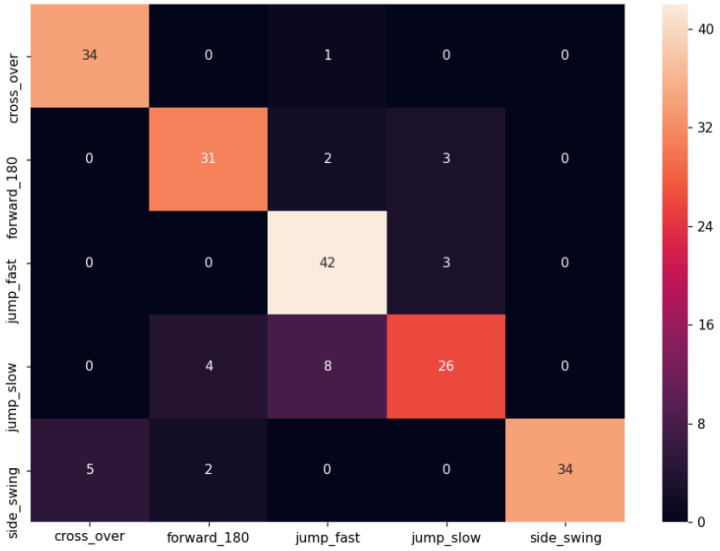
3.5.6 K-nearest neighbors

[31]

K-nearest neighbors gaat ervan uit dat gelijkaardige zaken zich in elkaars buurt bevinden. Het algoritme berekent de afstand tussen datapunten gebruikmakend van een bepaalde afstandsmetriek. De euclidische afstand is hierbij de meest gebruikte. Het algoritme werkt als volgt. Voor elk datapunt wordt de afstand berekend tot alle andere datapunten. Deze afstanden worden vervolgens gesorteerd. De meest voorkomende klasse van de k datapunten met de kleinste afstand wordt als voorspelling teruggegeven. K is hier de enige en dus ook belangrijkste hyperparameter.

Dit simpel algoritme heeft zo zijn voordelen. Er is slechts 1 hyperparameter waardoor geen tijd verloren gaat aan het tunen van vele parameters. Ook is het bouwen van een complex model niet nodig. De keerzijde van de medaille is het feit dat bij grote datasets KNN minder performant wordt.

Figure 3.15: confusion matrix van Naive bayes



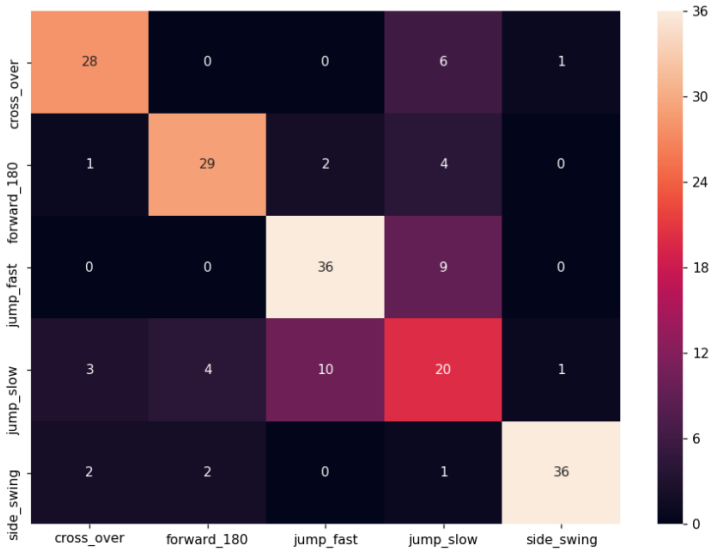
Belangrijke parameters zijn n_neighbors, weights, algorithm, leaf_size en metric. n_neighbors stelt k voor. Deze parameter kan een waarde hebben tussen 1 en n-1, waarbij n het aantal datapunten is in de dataset. Een kleine waarde kiezen voor k kan soms onstabiele resultaten geven. Stel dat een datapunt omringt is door datapunten van bijna exclusief 1 bepaalde klasse met slechts een klein aantal behorende tot een tweede verschillende klasse. Liggen de datapunten van deze minder voorkomende klasse toevallig dichterbij dan de rest, dan kan het resultaat met kleine k een vertekent beeld gegeven worden. Maken we k te groot dan is het mogelijk dat meer misclassificaties ontstaan aangezien ook verder liggende punten in rekening worden gebracht. Een grote k is echter wel in staat om ruis te onderdrukken. K is best een oneven nummer om geen ex aequos te krijgen.

weights kan ingesteld worden met 'uniform' of 'distance'. Uniform zal aan alle datapunten evenveel gewicht toekennen. Distance gaat de dichtstbijzijnde punten zwaarden laten doorwegen dan punten op een grotere afstand.

De parameter algorithm stelt het gebruikte zoekalgoritme in. Doordat telkens alle samples moeten overlopen worden bij classificatie van een nieuw datapunt, is het nuttig om met een efficiënte datastructuur te werken. Hiervoor zijn enkele opties. Een kd tree is nuttig bij datasets met grote dimensionaliteit maar met een klein aantal sampels. Zoeken in deze structuur verloopt echter trager bij grote k. Een ball tree is nuttig bij zowel grote dimensionaliteit als groot aantal samples. Ook deze structuur presteert slechter bij grote k.

Er kan voor gekozen worden om bij een bepaald aantal datapunten over te schakelen op het brute force algoritme. Dit wordt ingesteld met behulp van de

Figure 3.16: confusion matrix van K-nearest neighbors



leaf size parameter.

Als laatste kan de gebruikte afstandsmetriek meegegeven worden met behulp van de parameter metric.

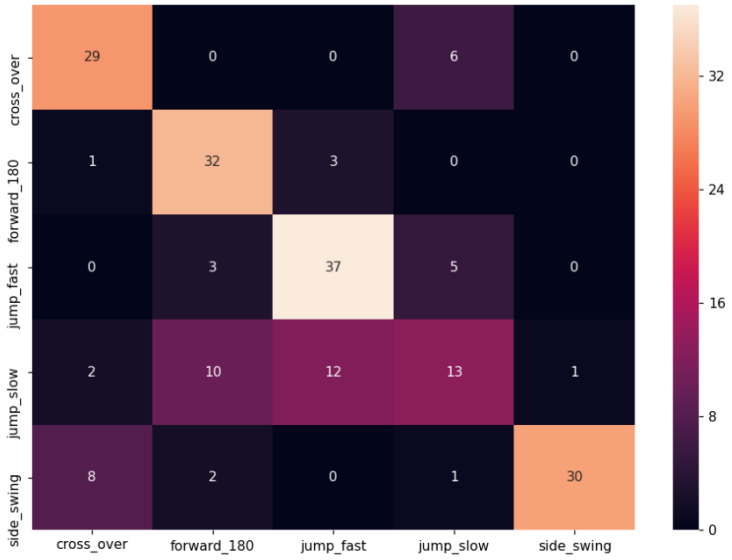
3.5.7 Stochastic Gradient Descent classifier

[20] [32] [7]

De decision boundary wordt voorgesteld door een vector loodrecht op een hyperplane. Door de gewichten van deze vector aan te passen kan de positie van deze boundary veranderd worden.Dit om een optimale classificatie te bekomen. Aan de hand van een loss functie wordt berekend wat de totale loss van het model is. Bij misclassificatie wordt aan een datapunt een kost toegekend. De loss functie bepaalt hoe groot deze kost is. Door deze functie dus minimaal te maken kan de meest optimale decision boundary gevonden worden. Door de afgeleide te berekenen ten op zichtte van elke feature wordt gekeken in welke richting het hyperplane, in die dimensie, moeten aangepast worden om de minimale kost te bereiken. Via een parameter alpha wordt de learning rate meegegeven. Dit geeft aan hoe groot de stappen zijn bij elke update van parameters. Stochastic gradient descent berekent een benadering voor de gradiënt aan de hand van slechts 1 datapunt. Dit is efficiënter dan alle datapunten hiervoor te overlopen en pas dan een update door te voeren. Voor grote datasets is dit een zeer goede oplossing. De gewichten van het model worden aangepast overeenstemmend met de berekende gradiënt in dat datapunt.

SGD maakt gebruik van randomness, de training data wordt dus best geshuffeld voor fitting. Ook wordt de data best geschaald aangezien datapunten met

Figure 3.17: confusion matrix van Stochastic Gradient Descent



elkaar moeten kunnen vergeleken worden.

belangrijke paramaters om te tunen bij dit algoritme zijn alpha, learning rate en max_iter. Zoals eerder vermeld geeft alpha weer hoe groot de stappen zijn bij elke update van de model parameters. Alpha is dus een getal tussen 0 en 1 waarbij 1 betekent dat de update totaal niet wordt afgezwakt. 0 betekent dat er geen updates worden doorgevoerd. Een te grote alpha zorgt ervoor dat het model te snel convergeert naar een niet optimale toestand. Een te kleine alpha kan ervoor zorgen dat het learning proces vast komt te zitten.

De parameter learning rate zegt welk learning rate schedule we gebruiken. Het soort schedule geeft weer hoe de learning rate verandert in de tijd.

Het maximum aantal iteraties hangt gedeeltelijk samen met alpha. Bij een kleine learning rate zijn ook meer iteraties nodig.

3.5.8 Multilayer Perceptron classifier

[1] [14] [11] [21]

MLP is 1 van de gemakkelijkst te implementeren neurale netwerken. Een perceptron is de kleinste eenheid van een neuraal netwerk. Een perceptron zal een aantal inputs met hun corresponderende gewichten vermenigvuldigen en hierbij een bias optellen. Via een activatie functie wordt de output bekomen van de perceptron. Een multilayer perceptron bestaat uit minstens 3 nodes waarvan elk (naast de input node) een non lineaire activatie functie gebruikt. De nodes tussen de input en output node worden ondergebracht in lagen, hidden layers genoemd.

De loss functie meet de performantie van het netwerk. Een hoge loss betekent dat de voorspelde klasse voor veel datapunten niet correct is. Het is de kunst om een set W te vinden die de loss functie minimaliseert. De activiatiefunctie beschrijft een non lineaire relatie tussen input en output. Populaire functies zijn sigmoid, relu en TanH. Het trainen van het model gebeurt in 3 fasen: forward pass, loss calculation en backward pass. Bij de forward pass wordt de input via de lagen doorgegeven en een output wordt berekend. Hiermee wordt de loss berekend. Door de partiële afgeleiden te nemen van de loss functie met betrekking tot de verschillende parameters worden de gewichten aangepast (backward pass). In elke laag moet de node die de meeste errors veroorzaakt heeft beboet worden door een kleiner gewicht eraan te koppelen. Voordelen van deze classifier zijn het feit dat de decision boundary niet lineair moet zijn. Nadelen zijn dat de loss functie convex is en meer dan 1 lokaal minimum bevat. Er is sprake van een grote hoeveelheid hyperparameters. Schaling van de data is nodig. Voor multiclass classificatie kan een laag softmax gebruikt worden.

Een aantal van de belangrijkste hyperparameters zijn: de gekozen activation functie, alpha en hidden_layer_sizes. Alpha stelt de mate van regulizatie in. Hiermee wordt ingesteld in welke mate misgeclassificeerde datapunten toegelaten worden. De hidden layer sizes stelt in van hoeveel hidden layers gebruik gemaakt wordt en hoeveel neuronen in elke laag zitten. Het aantal hidden layers is afhankelijk van de data. Indien deze lineair afscheidbaar is, dan zijn geen hidden layers nodig. Meestal is 1 hidden layer genoeg. Het aantal neuronen is best het gemiddelde tussen het aantal in de output en input layer.

Het is mogelijk dat er meerdere lokale minima voorkomen, verschillende weight initialisaties kunnen dus andere resultaten hebben.

MLP is ook gevoelig voor feature scaling. Het is dus beter om de data eerst te normaliseren zodat elke feature een gelijke bijdrage heeft. Hiervoor wordt gebruik gemaakt van de StandardScaler.

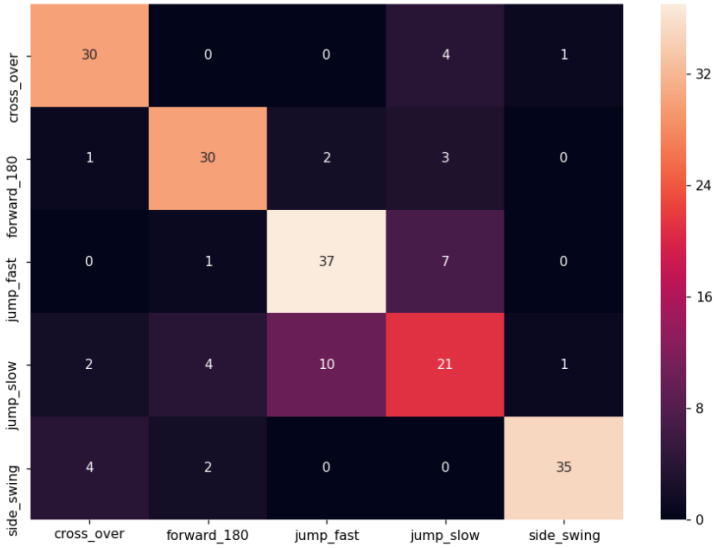
3.5.9 CNN

[41] Deep learning imiteert de werking van het menselijke brein in de manier van processen van data en maken van patronen voor gebruik in beslissingen. Het is een subset van machine learning en gebruikt hiërarchische levels van artificiële neurale netwerken. Data wordt geprocessed in een niet lineaire manier. Elke laag van het neuraal netwerk bouwt verder op zijn vorige laag met extra data. CNN wordt vooral gebruikt bij image processing. In dit geval zijn er veel inputs (het aantal pixels). In MLP wordt voor elke input 1 perceptron voorzien wat bij veel inputs niet echt efficiënt is.

[16] [37]

De convolutional layer is de bouwsteen van een CNN. Convolutie gaat een filter laden glijden over de input 2D array en telkens de convolutie nemen hiervan. De convolution layer bestaan uit 6 afzonderlijke filters. Dit resulteert in 6 feature maps waarin de berekende convolutie waarden zitten. Het doel hiervan is om features te extraheren. De matrix gevormd door de convolutie uit te voeren met de filter en het dot product te berekenen met de gewichten

Figure 3.18: confusion matrix van Multilayer Perceptron classifier



wordt de activation map of feature map genoemd. Parameter sharing is het delen van gewichten met alle neuronen in een feature map. Local connectivity gaat elk neuroon enkel connecteren met een subset van de input image. Dit zal het aantal parameters verminderen en de berekening dus ook sneller maken.

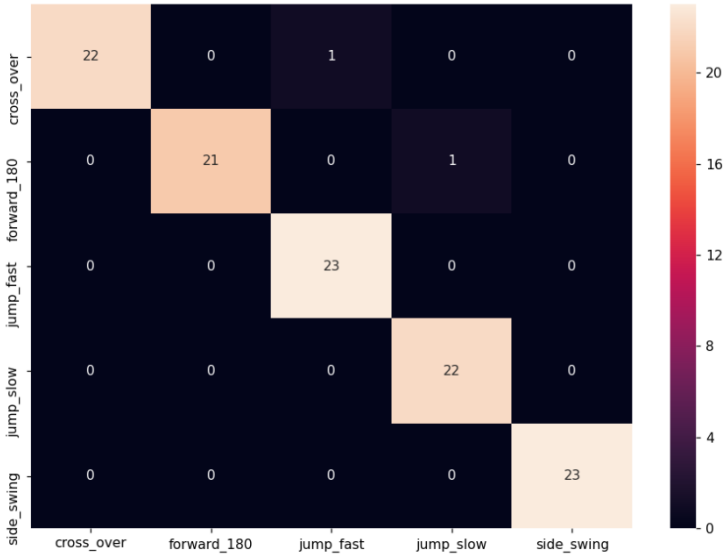
Een pooling layer is een tweede bouwsteen van CNN. Hiermee wordt de grootte van de representatie gereduceerd aangezien dit resulteert in minder parameters en dus minder rekenwerk. Dit is een soort dimensionality reduction. Max pooling wordt het meest gebruikt. Hierbij wordt een oppervlak bekeken en hieruit wordt de maximum waarde genomen. Ook kan gekozen worden om het gemiddelde te nemen. Door het aantal parameters te reduceren wordt ook overfitting voorkomen. Ook maakt het het netwerk robuust tegen kleine variaties in input data.

Een ander soort laag is RELU. Dit is een non lineaire operatie. De data die het model moet leren zal hoogstwaarschijnlijk non lineair zijn daarom is het introduceren van non lineariteit nodig. Er bestaan ook andere non lineaire operatie (tanh en sigmoid). RELU geeft echter betere resultaten.

Een fully connected layer is een traditionele MLP met een softmax activatie functie in de output layer. Fully connected wil zeggen dat elk neuron uit de vorige laag geconnecteerd is met elke neuron in de volgende laag. De fully connected layer gebruikt de high level features van de pooling en convulation layers om deze te classificeren. Deze laag zorgt ook voor het leren van een non lineaire combinatie van de features.

De convolution en pooling layers zijn feature extractors. De fully connected layer is de classifier. Hoe meer convolutie lagen, hoe complexer de geleerde

Figure 3.19: confusion matrix van CNN



features kunnen worden.

In eerste instantie worden de filters en gewichten random geïnitialiseerd. Na de forward propagation wordt de error (loss) van het model berekend. De gewichten worden ge-update door de gradiënt ten opzichte van elke parameter te berekenen. Zo is geweten in welke richting deze moet aangepast worden. In de backpropagation fase worden de gewichten effectief ge-update.

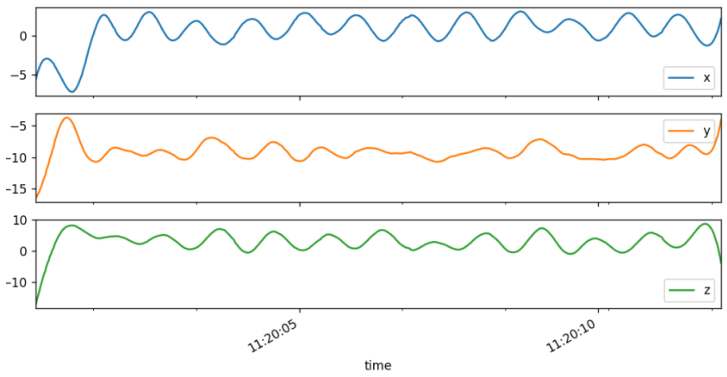
[15]

Om overfitting te voorkomen kan gebruik gemaakt worden van dropout. Deze teckniek zal bij iedere iteratie bepaalde neuronen negeren.

3.6 Hyperparameter tuning

Elk algoritme heeft een eigen set van hyperparameters. Via GridSearch kunnen mogelijke waarden voor deze parameters opgegeven worden. Het gridSearch algoritme zal dan elke combinatie uitvoeren en degene met de beste accuraatheid teruggeven. De classifier met deze combinatie van parameters kan dan gebruikt worden. Andere hyperparameters onafhankelijk van het algoritme zoals de window grootte worden via een empirisch onderzoek bepaalt. Waarden tussen 0.5 en 2 seconden werden getest. Deze grenzen werden gekozen met de gemiddelde duur van een sprong in het achterhoofd.

Figure 3.20: gefilterd signaal



3.7 Berekeningen

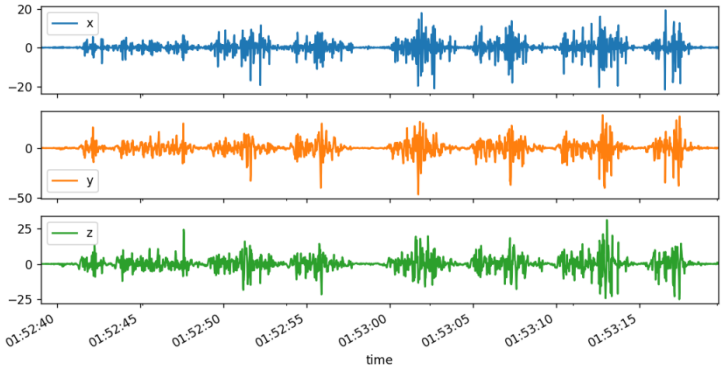
3.7.1 Aantal draaiingen

Het aantal draaiingen in een rope skipping sessie kan bekomen worden door naar de periode te kijken van het signaal. Deze periode zal gelijk zijn over x, y en z as. Om het aantal periodes eruit te halen, kan 1 periode manueel uitgesneden worden. Deze wordt dan geconvolueerd met het volledige signaal. De convolutie operatie zal grote pieken vertonen wanneer het signaal overeenkomt. Door deze pieken te tellen, is het aantal periodes en dus het aantal draaiingen geweten. Aangezien de periode voor verschillende sprongen uitgevoerd door verschillende proefpersonen niet telkens een gelijkaardig patroon zal vertonen, is deze manier niet toepasbaar. Ook kan gekozen worden om de data in het 95% kwartiel te bekijken. Elke periode heeft namelijk 1 uitgesproken piek, wat dus voor een hoge waarde zorgt. De amplitude van het signaal kan echter tijdens een sprong afzwakken of versterken zodat niet elke piek in het 95% kwartiel voorkomt. Hierdoor is ook dit ook geen optimale methode. Een laatste poging is om een filter op het signaal toe te passen. Dit zal ervoor zorgen dat er per periode slechts 1 lokaal maxima aanwezig is. Door op dit geëffend signaal een piek detectie uit te voeren kan een vrij goede benadering gemaakt worden van het effectieve aantal draaiingen. Het gemiddelde wordt genomen tussen de 3 assen. Dit omdat het signaal langs bepaalde assen soms een minder grote amplitude heeft wat kan resulteren in gewijzigde piekdetectie. Ook kan een extra piek te wijten aan ruis langs een bepaalde as zorgen voor te veel gedetecteerde draaiingen.

3.7.2 Fouten tijdens een beweging

Wanneer een fout zich voordoet tijdens een sessie in de vorm van een haperring van het touw, dan is dit duidelijk merkbaar in het verloop van het signaal. Er zal voor enkele seconden geen of amper versnelling zichtbaar zijn. De periodes kunnen gedetecteerd worden door te filteren op datapunten in een interval

Figure 3.21: signaal met fout



dichtbij 0. Indien een opeenvolgende reeks genoeg waarden bevat, wordt dit bestempeld als fout.

Chapter 4

Gezondheidsapplicatie

Met dit systeem is het de bedoeling dat de gebruiker via zijn/haar smartwatch sessies kan starten waarvan nadien statistieken weergegeven worden op de bijhorende smartphone applicatie. De aangeboden sporten zijn lopen, fietsen, badminton en rope skipping. Data processing van rope skipping zal gebeuren aan de hand van een eigen model (chapter 3). Voor de registratie en persistentie van de andere activiteiten wordt gebruik gemaakt van google fit sessies. Een sessie rope skipping zal meer info geven aan de gebruiker in de vorm van timestamps met betrekkings tot bewegingen, het aantal draaiingen en eventuele tekortkomingen. Het is de bedoeling dat gebaseerd op deze data persoonlijke aanbevelingen gegeven worden.

4.1 Backend - Frontend

Het model werd getraind in python. Ook hier werden berekeningen uitgevoerd op de onbewerkte accelerometer data (aantal draaiingen en fouten). Er moet nu een afweging gemaakt worden tussen behouden van de scheiding van backend en frontend of migreren van de backend naar de frontend. Beide opties hebben voordelen en nadelen. Laten we eerst kijken naar de backend - frontend optie. Hierbij kan de code in python behouden worden. Python is namelijk zeer efficiënt om aan data analyse te doen. Het beschikt over handige libraries zoals pandas waarmee data door middel van reeds geïmplementeerde functionaliteit kan bewerkt worden. De server waarop de backend zou draaien heeft ook meer rekenkracht dan een android smartphone of smartwatch. Hierdoor kunnen zwaardere maar accuratere machine learning methodes gebruikt worden. Er is echter wel constant internet connectie nodig om updates te krijgen. Een louter frontend aanpak zal het probleem van offline werken aanpakken. De berekeningen worden nu op de applicatie zelf uitgevoerd. De resultaten zijn dus meteen beschikbaar voor de gebruiker. Een android smartphone heeft echter minder rekencapaciteit (alhoewel dit vandaag de dag nog meevalt) waardoor zeer zware berekeningen niet mogelijk zijn. Ook is het minder intuïtief om in java aan data

analyse te doen.

4.2 Smartwatch applicatie

De smartwatch applicatie staat in voor het starten van sessies en het verzamelen van data tijdens deze sessies. De accelerometer en hartslag datapunten worden via de MessageClient API verstuurd naar de android applicatie draaiende op een smartphone. Tijdens een sessie wordt eveneens de hartslag gemonitord, indien deze gevaarlijk hoog wordt zal een melding gegenereerd worden.

4.3 Smartphone applicatie

De smartphone applicatie staat in voor het ontvangen van datapunten en de verwerking hiervan. Het machine learning model is lokaal beschikbaar en wordt dan ook gebruikt om bewegingen te herkennen in de ontvangen accelerometer data. Ook worden het aantal draaiingen en eventuele fouten berekend. Per beweging zal het aantal MET minuten berekend worden aan de hand van hartslag datapunten binnen een tijdsinterval. Deze worden later gebruikt om aanbevelingen te genereren. Aan de hand van de Google Activity API kan gedetecteerd worden wanneer de gebruiker te lang stilzit. Op dat moment moet een aanbeveling gegeven worden. Verder moet ook rekening gehouden worden met de huidige activiteit van de gebruiker. Als de gebruiker zich in de auto bevindt zal geen melding gegenereerd worden. Ook heeft de gebruiker de mogelijkheid om meldingen te negeren. Deze info wordt dan gebruikt om te leren uit het wekelijks patroon van de gebruiker. Indien meerdere keren op hetzelfde tijdstip (marge van een uur) genegeerd wordt, dan wordt deze periode niet meer gebruikt voor aanbevelingen.

4.3.1 Aanbevelingen

[12] Sommige aanbevelingstechnieken vereisen gebruikersinput om interactie mogelijk te maken. 2D aanbevelingstechnieken bekijken enkel de dimensie gebruiker en item. Er wordt geen rekening gehouden met andere contextuele dimensies. MD aanbevelingstechnieken daarentegen gaan naast de gebruiker en item dimensies rekening houden met meerdere contextuele features. Een 2D model kan op volgende manieren uitgebreid worden. Contextuele pre filtering gaat de dataset eerst filteren op basis van huidige contextuele voorkeuren. Contextuele post-filtering doet deze filtering als laatste. Contextuele modeling integreert de huidige contextuele voorkeuren in de aanbevelingsfunctie. Het aanbevelingssysteem, waarvan de applicatie gebruik maakt, is een context-aware systeem op basis van gebruikersinput. Er zal geleerd worden uit het al dan niet negeren van gegeven meldingen. De contextuele filtering wordt achteraf toegepast door te checken of de gebruiker al vaak een melding genegeerd heeft op het gekozen tijdstip van de melding.
[29]

Vooraleer een 2D aanbevelingssysteem uit te breiden naar een context-aware systeem moet dit 2D systeem op punt gezet worden. Er zijn verschillende manieren waarop dit kan geïmplementeerd worden. Collaborative filtering methoden bekijken enkel interacties tussen gebruiker en item. Deze worden opgeslagen in een gebruiker-item interactie matrix. Deze methode wordt nog eens onderverdeeld in een geheugen gebaseerde en een model gebaseerde aanpak.

Geheugen gebaseerd veronderstelt geen model en maakt in essentie gebruik van nearest neighbors. De items gekoppeld aan de gebruikers die het dichtst liggen worden aanbevolen. Een gebruiker-gebruiker methode gaat gebruikers proberen te identificeren met het meest gelijkaardige interactie profiel om zo nieuwe items aan te bevelen, dit is gecentreerd rond gebruikers. Gebruikers worden voorgesteld met een vector van hun interacties met items. Een item-item methode gaat items zoeken gelijkaardig aan de items waarmee de gebruiker al positief interageerde. Items worden gezien als gelijkaardig als de 2 gebruikers er op een gelijkaardige manier mee interageerden. Deze manier is gecentreerd rond het item.

Wanneer een model aanwezig is dan is er een relatie tussen gebruiker en de gekoppelde items. Deze relatie wordt gebruikt bij voorspellingen. matrix factorisatie gaat een grote ijle gebruiker-item interactie matrix ontleden in een product van 2 kleinere dichte matrices (user factor matrix en item factor matrix).

Deze methode heeft als voordeel dat geen additionele info nodig is over gebruikers of items en kan dus in vele situaties gebuikt worden. Aanbevelingen worden ook accurater naarmate de gebruikers meer intrageren met items. Een nadeel is het cold start probleem. Dit doet zich voor door het feit dat enkel historische data bekeken wordt. Een nieuw item aanbevelen is onmogelijk en aan een nieuwe gebruiker kan niks aanbevolen worden.

Content gebaseerde methoden gebruiken bijkomende info over de gebruiker en/of item objecten. Deze worden behandeld als features en kunnen helpen in verklaren waarom er juist een relatie is tussen die gebruiker en item. Deze methoden hebben geen last van cold start aangezien informatie altijd aanwezig zal zijn (leeftijd, geslacht..). Enkel nieuwe gebruikers of items met voorheen onbekende features hebben er last van.

content gebaseerde methoden vervormen het probleem naar ofwel een classificatie ofwel een regressie probleem. Als de classificatie gebaseerd is op gebruikersfeatures dan is het item gecentreerd omdat de berekeningen per item gebeuren. Indien de classificatie gebeurt op de item features dan is het gebruiker gecentreerd. Het model wordt per gebruiker gemaakt. Dit model is gepersonaliseerder omdat het niet gebruik maakt van data afkomstig van alle gebruikers. Het model is wel minder robuust omdat 1 gebruiker met minder items interageerd.

Het systeem ontwikkeld tijdens deze thesis legt de nadruk vooral op het persoonlijke aspect. Daarom wordt gekozen voor een content gebaseerde methode gecentreerd rond de gebruiker. Het nadeel bij deze methode is hier ook niet echt van toepassing. Er zijn slechts een beperkt aantal bewegingen. De kans is dus groot dat 1 gebruiker deze allemaal uitvoert.

Elke week zal een nieuwe reeks aanbevelingen berekend worden. Het ge-

bruikte algoritme houdt rekening met het aantal keer een bepaalde activiteit uitgevoerd werd. Ook met het aantal fouten gemaakt werd tijdens een beweging. Deze aanbevelingen krijgen dan meer gewicht. De duur van een aanbeveling wordt random gekozen tussen 0.2 en 3 uur. Het totaal aantal mets van de berekende aanbevelingen moet gelijk of groter zijn dan het doel voor die week. Dit totaal aantal mets wordt bekomen door gebruik te maken van het gemiddeld aantal mets per minuut voor de corresponderende activiteit.

4.3.2 Inspanningspunten

De hoeveelheid energie die gebruikt wordt is proportioneel tot de hoeveelheid zuurstof in je lichaam. De resting metabolic rate is de hoeveelheid zuurstof bij rust. De energie gevraagd van een fysieke activiteit kan uitgedrukt worden als een veelvoud van deze resting metabolic rate. Dit is de metabolic equivalent of task (MET). Een individu met gemiddelde fitness capaciteiten kan tot 12 METS aan, top atleten kunnen zelfs tot 20 gaan. Activiteiten kunnen worden geclassificeerd in hoeveel mets ze vergen. De peak aerobic capaciteit van een individu wordt uitgedrukt in maximum zuurstof inname tijdens sportactiviteiten. VO2 max meten moet gebeuren door strikte protocollen te volgen in een sport laboratorium. Dit is zeer tijdrovend en wordt dus enkel in speciale omstandigheden gedaan. Het is mogelijk om een relatieve benadering van de intensiteit te bekomen door de effecten op de hartslag en respiratie gehalte te meten. De talk test is 1 van de gebruikte methoden. Wanneer een persoon kan praten maar niet kan zingen, dan is de activiteit gemiddeld intensief. Wanneer de persoon ook niet meer kan praten dan is de activiteit vigorous intensief. De heart rate test gaat ervan uit dat de hartslag stijgt in een reguliere manier naarmate de intensiteit van de activiteit stijgt. De gemiddelde maximum hartslag kan berekend worden door de leeftijd af te trekken van 220. De Submaximal excercise test wordt gebruikt om de maximum fitness capaciteit te berekenen zonder over de grens van 0,85*max heart rate te gaan. Door monitoren van hartslag kan de maximum fitness capaciteit geëxtrapoleerd worden met behulp van verschillende methoden. Deze waarde heeft echter gelimiteerde waarde.

heart rate zones zijn een manier om te monitoren hoe hard getraind wordt. Hartslag wordt gesplitst in 5 heart rate zones gebaseerd op de intensiteit van de training en rekening houdend met de maximum heart rate. De minimum heart rate is de resting heart rate en de maximum heart rate kan berekend worden met verschillende methoden. Tussen deze waarden liggen de 5 zones. Er zijn verschillende manieren om deze zones af te bakenen. 1 ervan is om percentages van de maximum heart rate te gebruiken. Heart rate zones zijn gerelateerd aan de aerobic en anaerobic thresholds. Aerobic threshold is de steady state effort die voor uren kan uitgehouden worden. De threshold van dit niveau is bereikt wanneer de hoeveelheid lactose in het bloed begint te stijgen samen met het level van intensiteit waaraan anaerobic energie bronnen helpen in energie productie. Anaerobic threshold is de hoogste training threshold die kan volgehouden worden voor een periode zonder lactaat opbouwing in het bloed. Spieren verbranden suiker namelijk op 2 manieren: aerobically (zuurstof) en

anaerobically (zonder zuurstof).

Volgende paragraaf licht de verschillende zones toe. Heart rate zone 1 (50-60% HRMAX) is de zeer lichte intensiteitszone. Trainen in deze zone zal de herstelling bevorderen en je klaarstomen om te trainen in hogere zones. Heart rate zone 2 (60-70% HRMAX) is de lichte intensiteitszone. Dit is de zone die je uithoudingsvermogen verbetert. Het lichaam wordt beter in oxideren (verbranden) van vet en de musculaire fitness zal samen met de capillaire densiteit stijgen. Heart rate zone 3 (70-80% HRMAX) is de gemiddelde intensiteitszone. Deze zone bevordert de efficiëntie van de bloed circulatie in het hart en skelet spieren. Dit is de zone waar melkzuur begint te verbranden in de bloedstroom. Heart rate zone 4 (80-90% HRMAX) is de harde intensiteitszone. In deze zone zal de ademhaling moeilijker gaan en begin je aerobically te sporten. In deze zone wordt snelheidsuithoudingsvermogen getraind. Het lichaam wordt beter in het gebruik van carbohydraten voor energie. Je zal hogere levels van melkzuur in de bloedsomlopen voor langere tijd kunnen uithouden. Heart rate zone 5 (90-100% HRMAX) is de maximum intensiteitszone. Lactic acid zal zich ophopen in de bloedsomloop. Dit is het niveau waarop atleten trainen.

Aan de hand van de tijd gespendeerd in iedere heart rate zone wordt het aantal METs verbruikt tijdens die activiteit berekend. Dit geeft een beeld van hoe lastig de activiteit was voor de gebruiker.

4.3.3 Goal

Het doel voor de volgende week wordt berekend door historische data tot 10 weken in het verleden te bekijken. Deze data geeft weer hoeveel METs de gebruiker per week verbruikt heeft. Hiervan wordt het 60e percentiel genomen. Dit zodat het doel lichtjes verhoogd wordt ten opzichte van de mediaan, maar toch nog binnen de grenzen van mogelijkheden blijft. Wanneer niet genoeg data aanwezig is, dan wordt de historische data aangevuld met 600 mets, wat volgens literatuur het aan te raden aantal van MET verbruik is per week.

4.4 Authenticatie

Aangezien bij onder andere de berekening van het aantal mets gebruik gemaakt wordt van persoonlijke gegevens, is authenticatie belangrijk. Ook de fitness data zelf is vertrouwelijk. Er wordt gebruik gemaakt van Google Sign In. Voor Google Fit is deze authenticatie zeker nodig aangezien gewerkt wordt met een backend in de vorm van een fitstore en de persoonlijke data het toestel dus zal verlaten.

4.5 Permissies

Verschillende van de gebruikte Google APIs hebben vertrouwelijke informatie nodig zoals locatie en fitness statistieken. Hiervoor moet de gebruiker eerst toestemming geven.

Conclusie

Bibliography

[1] D. Adams. *https://analyticsindiamag.com/a-beginners-guide-to-scikit-learns-mlpclassifier/.* San Val, 1995.

[2] D. Adams. *https://core.ac.uk/download/pdf/81111058.pdf.* San Val, 1995.

[3] D. Adams. *https://developer.android.com/training/wearables/data-layer.* San Val, 1995.

[4] D. Adams. *https://developers.google.com/fit/overview.* San Val, 1995.

[5] D. Adams. *https://dl.acm.org/doi/abs/10.1145/2851581.2892366?casa*$_t$*oken* = $7ksHxSGebdYAAAAA : dIvk9oQdXZjMoAcpNTJT5J38BBzJPpY5RS5csOo9m9MkTp002xH5gpB$

[6] D. Adams. *https://dl.acm.org/doi/abs/10.1145/3320435.3320459?casa*$_t$*oken* = $gJlBqorChr0AAAAA : d2PKZQhfwngzfGL26ByBED56M6Xe8hTtt_kdfWAwggBUtN7B6OPKUxA$
$BJrwURNn92SHiqFE.SanVal$, 1995.

[7] D. Adams. *https://gluon.mxnet.io/chapter06*$_o$*ptimization/gd* $-$ *sgd* $-$ *scratch.html.*$SanVal$, 1995.

[8] D. Adams. *https://ieeexplore.ieee.org/abstract/document/7551575.* San Val, 1995.

[9] D. Adams. *https://ieeexplore.ieee.org/abstract/document/8368718.* San Val, 1995.

[10] D. Adams. *https://journals.lww.com/pedpt/FullText/2017/10000/Validity*$_o f_A$*ccelerometry*$_t o_M$*easure*$_P$*hy*

[11] D. Adams. *https://kindsonthegenius.com/blog/2018/01/basics-of-multilayer-perceptron-a-simple-explanation-of-multilayer-perceptron.html.* San Val, 1995.

[12] D. Adams. *https://link.springer.com/chapter/10.1007/978-3-030-04021-*$5_2 7.SanVal$, 1995.

[13] D. Adams. *https://link.springer.com/content/pdf/10.1007/978-1-4842-2845-6.pdf.* San Val, 1995.

[14] D. Adams. *https://medium.com/@AI*$_w ith_K$*ain/understanding* $-$ *of* $-$ *multilayer* $-$ *perceptron* $-$ *mlp* $-$ $8f179c4a135f.SanVal$, 1995.

[15] D. Adams. *https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5*. San Val, 1995.

[16] D. Adams. *https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8*. San Val, 1995.

[17] D. Adams. *https://ogrisel.github.io/scikit-learn.org/sklearn-tutorial/modules/multiclass.html*. San Val, 1995.

[18] D. Adams. *https://pandas.pydata.org/about/index.html*. San Val, 1995.

[19] D. Adams. *https://pdfs.semanticscholar.org/7824/05c860bf3812d9c78bd6bcef17adaf48c033.pdf*. San Val, 1995.

[20] D. Adams. *https://scikit-learn.org/stable/modules/sgd.html*. San Val, 1995.

[21] D. Adams. *https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw*. San Val, 1995.

[22] D. Adams. *https://stats.stackexchange.com/questions/379383/why-does-naive-bayes-work-better-when-the-number-of-features-sample-size-comp*. San Val, 1995.

[23] D. Adams. *https://stats.stackexchange.com/questions/39243/how-does-one-interpret-svm-feature-weights*. San Val, 1995.

[24] D. Adams. *https://support.polar.com/e_manuals/M430/Polar_M430_user_manual_Nederlands/Content/Pola*
$Flow - Web - Service.htm. SanVal$, 1995.

[25] D. Adams. *https://support.polar.com/e_manuals/M600/wear − os/polar − m600 − user − manual − nederlands/Content/introduction.htm.* $SanVal$, 1995.

[26] D. Adams. *https://support.polar.com/e_manuals/Team_Pro/Polar_Team_Pro_user_manual_English/Content/*

[27] D. Adams. *https://thesai.org/Downloads/Volume6No3/Paper*$_2$ − $Android_Application_to_Assess_Smartphone_Accelerometers.pdf. SanVal$, 1995.

[28] D. Adams. *https://towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989*. San Val, 1995.

[29] D. Adams. *https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15adaColl*. San Val, 1995.

[30] D. Adams. *https://towardsdatascience.com/journey-to-the-center-of-multi-label-classification-384c40229bff*. San Val, 1995.

[31] D. Adams. *https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761*. San Val, 1995.

[32] D. Adams. *https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31*. San Val, 1995.

[33] D. Adams. *https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47*. San Val, 1995.

[34] D. Adams. *https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496*. San Val, 1995.

[35] D. Adams. *https://towardsdatascience.com/understanding-adaboost-2f94f22d5bfe*. San Val, 1995.

[36] D. Adams. *https://towardsdatascience.com/understanding-random-forest-58381e0602d2*. San Val, 1995.

[37] D. Adams. *https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/*. San Val, 1995.

[38] D. Adams. *https://www.aaai.org/ocs/index.php/WS/AAAIW16/paper/viewPaper/12627*. San Val, 1995.

[39] D. Adams. *https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/*. San Val, 1995.

[40] D. Adams. *https://www.hindawi.com/journals/ahci/2019/3068748/*. San Val, 1995.

[41] D. Adams. *https://www.investopedia.com/terms/d/deep-learning.asp*. San Val, 1995.

[42] D. Adams. *https://www.mdpi.com/1424-8220/18/2/623/htm*. San Val, 1995.

[43] D. Adams. *https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3859040/*. San Val, 1995.

[44] D. Adams. *https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6053180/*. San Val, 1995.

[45] D. Adams. *https://www.sciencedirect.com/science/article/pii/S0003687017300261*. San Val, 1995.

[46] D. Adams. *https://www.sciencedirect.com/science/article/pii/S0957417417308333*. San Val, 1995.

[47] D. Adams. *https://www.sciencedirect.com/science/article/pii/S1568494615000447*. San Val, 1995.

[48] D. Adams. *http://www.bubblecode.net/en/2016/01/22/understanding-oauth2/*. San Val, 1995.