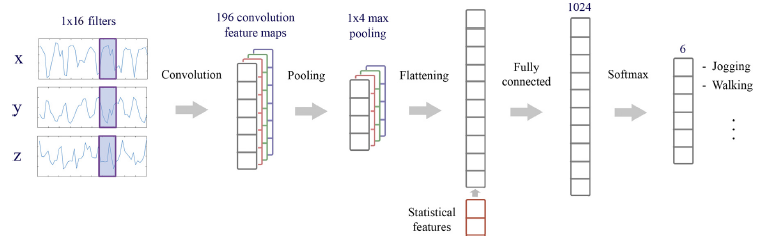
Literatuurstudie

# Real-time human activity recognition from accelerometer data using Convolutional Neural Networks

<https://www.sciencedirect.com/science/article/pii/S1568494617305665>

<https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>

Feature extraction gebaseerd op deep learning en convolutional neural networks. Een goed getunede oppervlakkige CNN kan met minder rekenkracht een accurate classificatie doen. De structuur van een CNN is als volgt: Convolutional layer. Hier zal een vector f glijden over input vector x en tijdens elke stap het scalair product berekenen. nonLinearity: om niet lineaire beslissingsgrenzen te leren wordt deze laag gebruikt. Door een niet lineaire activatie functie dat puntgewijs wordt toegepast op de output. Pooling layer: reduce en summarize de representatie. Fully-connected layer: flatten de output van convolutional en max-pooling layers in een 1 dimensionale vector en wordt gebruikt voor classificatie. De soft-max layer: berekent waarschijnlijkheidsdistributie.

Dropout technique to avoid overfitting. 

## Convolutional neural network

NOG BEKIJKEN <https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/>

Dit is een deep learning algoritme dat een input image kan ontleden en aan verschillende aspecten, verschillende gewichten kan toewijzen. De gewichten worden aangeleerd. Filters worden dus niet manueel gecreerd, maar worden door CNN zelf geleerd

# Deep Activity Recognition Models with Triaxial Accelerometers

<https://www.aaai.org/ocs/index.php/WS/AAAIW16/paper/viewPaper/12627>

Classificatie en feature engineering methodes voor activiteiten herkenning.

Zelf gemaakte features: statistisch, teken van ruwe data signalen.

Met deep learning is het mogelijk om discriminatieve features te leren uit historische data, dit is systematisch en automatisch.

Samples van volgende vorm: rt = r\*t + wt , wt is een noise vector

Deze samples worden gegroepeerd in windows.

## Data Preprocessing

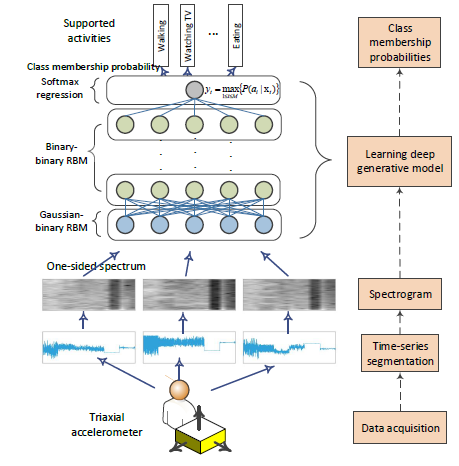
via een spectogram van een accelerometer signaal, dit is een representatie van veranderingen in de versnellingsenergie in functie van de frequentie en de tijd. CLASSIFICATION ACCURACY

de verschillen in intensiteit tussen bijelkaarliggende punten kunnen bekomen worden via de spectogram representatie.

COMPUTATIONAL COMPLEXITY

de spectogram representatie resulteerd in minder grootte data dimensionaliteit dan het tijd domein

## system and model



2 fasen: unsupervised, generatieve, pre-training stap en supervised, discriminatieve, fine-tuning stap

Eerst worden deep belief networks gebruikt. Dit zijn generatieve modellen bestaande uit meerde lagen van hidden units (restricted Boltzmann machines RBM)

Gaussian-binary RBM: model energie inhoud in continue accelerometer data

De volgende lagen zijn binary-binary RBMs

## Deep learning

<https://www.investopedia.com/terms/d/deep-learning.asp>

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 hierarchische levels van artificiele neurale netwerken 🡪 proces data in een niet lineaire manier. Elke laag van het neuraal netwerk bouwt verder op zijn vorige laag met extra data

# MATLAB Deep Learning

<https://link.springer.com/content/pdf/10.1007/978-1-4842-2845-6.pdf>

## Machine learning

Machine learning is een modeling techniek gebaseerd op data. Het systeem heeft als input een training set waaruit de model afgeleid wordt. Machine learning is nodig in situaties waarbij creeren van een model image herkenning of stem herkenning omvat, fysische wetten of wiskundige formules schieten hierbij te kort. Diversiteit van de training dataset is belangrijk. Een training dataset bestaande uit enkel geschreven notas van 1 persoon zal de cijfers/letters van een anders handschrift niet herkennen. De training dataset moet de karakteristieken van de field data reflecteren. Generalization 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.

Verschillende types machine learning. Supervised learning. Unserpervised Learning. Reinforcement Learning

# Personalized Physical Activity Coaching: A Machine Learning Approach

<https://www.mdpi.com/1424-8220/18/2/623/htm>

Deze studie heeft als doel om fysieke activiteit te verhogen tijdens werkdagen. Step count data werd verzameld met behulp van de Fitbit Flex. De data werd gecleaned, gereformat ne 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… toe te voegen. Het interval van 7 tot 18uur werd bekeken.

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. Accuracy is een metriek die weergeeft hoe dicht de voorspelde waarden bij de echte waarde liggen. De ratio tussen de correct geklassificeerde klassen en alle cases. Een laatste metriek is de F1-score. Deze gebruikt de precisie en recall van het model.

Het voorspellen of een bepaald doel zal behaald worden is een supervised 2 klasse classificatie probleem. Het is echter onmogelijk om op voorhand te bepalen welk algoritme het best zal performen, ook al bestaan er bepaalde richtlijnen voor welk algoritme goed is voor welke dataset. Ze moeten empirisch getest worden. De gebruikte algoritmes zijn: adaboost, desicion trees, KNeighborsClassifier, Logistic Regression, Neural Networking, Stochastic Gradient Descent, Random Forest en support Vector Classification. Deze werd gebaseerd op de flow chart van scikit learn en cheat sheet van microsoft azure.

# A data-driven approach to modeling physical fatigue in the workplace using wearable sensors

<https://www.sciencedirect.com/science/article/pii/S0003687017300261>

Metrieken om fysieke moeheid te meten kunnen bekomen worden via sensoren of door self reported fysieke moeheid. De rating scale die hier wordt gebruikt om fysieke moeheid te meten is Borg rating of perceived exertion. Andere metrieken voor het meten hiervan zijn: hartslag, kracht beschikbaar, tremor, veranderingen in postuur, multi joint coordination between diffrent segments.

Er wordt gewerkt in verschillende fasen. Eerst wordt de data verzameld, dan volg 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… Hierna werd de jerk berekent, dit is de verandering in versnelling en is een belangrijke indicator van fysieke moeheid. Dimension reduction zorgt voor verkleining van de dataset zonder dat gegevens verloren gaan. Hartslag werd bijvoorbeeld genormalizeerd naar de resting heart rate en age predicted maximum heart rate. …

## Borg rating of perceived exertion

<https://academic.oup.com/occmed/article/67/5/404/3975235>

PROBLEEM: hoe bekomen?

On the use of ensemble of classifiers for accelerometer-based activity

Recognition

<https://www.sciencedirect.com/science/article/pii/S1568494615000447>

De raw time series data werd onderverdeeld in 10 s segmenten en per segment werden verschillende features geidentificeerd. 6 basis feature types werden gebruikt: gemiddelde, standaard deviatie, gemiddel absoluut veschil, gemiddeld resulterend versnelling, tijd tussen pieken, binned distribution. In deze studie werden J48, logistic regression en MLP gebruikt als algoritmen.

## Window

<https://machinelearningmastery.com/deep-learning-models-for-human-activity-recognition/>

<https://towardsdatascience.com/activity-recognition-using-smartphones-machine-learning-application-a10e7b5578f9>

PROBLEEM: wat als transatie van ene nr andere activiteit in window?

# Automated e-Coaching System Architecture Framework for Promoting Physical Activity

<https://ieeexplore.ieee.org/abstract/document/8494163>

De database bevat waarden van alle concepten voor de patienten en vragen en berichten gebaseerd op de behavior change theory en practice. Dit wordt per uur geupdate according to regels in de diagnostic process. Rule based reasonings lost problemen op met feiten en regels afgeleid uit expert kennis.

De persistance layer slaat de nodige info op.

De service layer is verdeeld in componenten. Knowledge base bevat regels die dynamisch de attribuut gewichten aanpassen van de karakteristieken van een gebruiker. Om deze regels te creeren wordt informatie vanuit relevante literatuur gebruikt. Ook wordt kennis van een gezondheidspractinionar gebuikt. De regels worden gebruikt om een lijst van stage matched strategien en goal templates te genereren. De knowledge inference engine genereert recommandations op basis van de kennis. Rules transformation gebeurt in 2 modules. De expert driven rules module gebruikt kennis van domain experts gebruik makend van een data driven reasoning technique. De data driven regel module genereert regels automatisch.

De application layer bestaat ook uit verschillende componenten. De user profile and self reported questionnaire manager biedt de gebruiker een mogelijkheid om zijn info aan te passen. Gebaseerd hierop zal de service layer de rule execution uitvoeren.

# Thesis cas decloedt

## Telemedicine

<https://vsee.com/what-is-telemedicine/>

Dit is het uitoefenen van gezondheidszorg op afstand.

## Telehealth

<https://www.cchpca.org/about/about-telehealth>

## e-Health

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1550694/>

## m-health

<https://news.careinnovations.com/blog/what-is-mhealth-how-is-it-different-from-telehealth>

## privacy

## recommender systems

## gamification

## polar beat

<https://support.polar.com/e_manuals/Team_Pro/Polar_Team_Pro_user_manual_English/Content/Export_Data.htm>

deze app wordt gebruikt om de data gegeneert door de polar toestellen te kunnen exporteren naar csv formaat. De app heeft zeker nog andere doeleinden… De data bestaat in 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. Raw data bestaat uit samples per seconde met hartslag, snelheid, afstand, versnelling en running cadence.

## polar flow

<https://support.polar.com/e_manuals/M430/Polar_M430_user_manual_Nederlands/Content/Polar-Flow-Web-Service.htm>

zelfde functionaliteit (?) als polar beat, maar ook als webservice.

## Rscript

<https://paldhous.github.io/ucb/2018/dataviz/week7.html>

R wordt gebruikt om statistische analyse uit te voeren op big data en deze te visualiseren. RStudio kan hiervoor gebruikt worden.

## ANOVA test

<http://www.sthda.com/english/wiki/one-way-anova-test-in-r>

Dit is een test die meerdere gemiddelden met elkaar vergelijkt indien er meer dan 2 groepen zijn. De data is verdeeld in verschillende groepen gebaseerd op een groeperingsvariabele. Een null hypothesis betekent dat de gemiddelden hetzelfde zijn. Een alternative hypothesis betekent dat tenminste 1 ervan niet gelijk is aan de andere.

## Post hoc Tukey’s HSD test

## Bland-altman plot

## Google fit

<https://developers.google.com/fit/overview>

<https://developers.google.com/fit/android>

Google fit bestaat uit een aantal componenten. De fitness store slaat data op van verschillende toestellen en apps. Het is een cloud service. Het sensor framework zijn een set van high level representaties zodat werken met de store gemakkelijk is. Dit wordt gebruikt met de google Fit APIs. De high level representaties zijn data sources, deze stellen sensors voor. Data types zeggen welk soort data (step count, hartslag..). Data points zijn een timestamped array van waarden van een data type, komende van een data source. Dataset zijn data punten van hetzelfde type van dezelfde source binnen een bepaald tijdsinterval. Sessions stellen een tijdsinterval voor waarin de gebruiker een activteit beoefent. Fitness data is user specifiek, hiervoor moet dus toestemming gevraagd worden, de permissions and user controls doet dit voor u…... Met de google Fit APIs is het mogelijk om met de data te werken.

SENSORSERVICE: <https://developers.google.com/fit/android/new-sensors>

MEERDERE PROCESSEN: <https://developer.android.com/topic/performance/memory#MultipleProcesses>

* PROBLEEM: Custom sensor zal wss niet gaan want kan alleen luisteren naar datatypes vn google fit en accelerometer niet aanwezig

## GoogleApi

## Personal dynamic goals

<http://www.exercise4weightloss.com/activity-points.html>

FPoints

Borg exertion

# Personalizing Mobile Fitness Apps using Reinforcement Learning

<https://pdfs.semanticscholar.org/7824/05c860bf3812d9c78bd6bcef17adaf48c033.pdf>

Huidige fitness apps gebruiken geen features gerelateerd aan behavioral theory.

Goal setting is 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.

Parse API?

# The use of machine learning algorithms in recommender systems: A systematic review

<https://www.sciencedirect.com/science/article/pii/S0957417417308333>

Recommender systems gebruiken AI metgodes om de juiste recommendations te genereren. Er zijn 3 types: collabrative, content-based en hybrid filtering. Collabrative filtering kijkt naar de user data, recommendatios het item gekoppeld aan deze user kan dan gerocommended worden aan andere users met hetzelfde profiel. Content-based kijkt naar item data, gebaseerd op alle items van een bepaalde user worden gelijkaardige gerecommended. Hybrid filtering gebruikt beide methodes en geeft beide sets terug als recommendation. Collabrorative of hybrid heeft data van de user nodig, dit kan expliciet of impliciet bekomen worden. Er zijn verschillende soorten recommendations processen, naast het gewone: trust-based recommendations, context-aware recommendations, risk-aware recommendations. Content-based approaches gebruiken volgende strategieen: classifier based of neighbor methods.

# A Decision Tree Based Context-Aware Recommender System

<https://link.springer.com/chapter/10.1007/978-3-030-04021-5_27>

Ubiquitous computing zorgt voor persoonlijke recommandations die matchen met de gebruikers context, preferences, kennis, doelen en noden. Sommige recommandation technieken hebben user input nodig om interactie mogelijk te maken. Er zijn 2 categorien: 2D recommendation technieken, deze technieken bekijken enkel de dimensie user en item. Er wordt geen rekening gehouden met andere contextuele dimensies. MD recommandation technieken, naast de user en item dimensies wordt rekening gehouden met meerdere contextuele features. Een 2D model kan op volgende manieren uitgebreid worden. Contextuele pre filtering gaat de data set filteren op basis van huidige contextuele voorkeuren, daarna wordt 2D toegepast. Contextual post-filtering doet het omgekeerd. Contextual modeling integreert de huidige contextuele voorkeuren in de recommandation functie.

Beslissingsbomen worden gebruikt als classificatie model. Ze zijn flexibel in omgaan met demographische en contextuele features. ID3 is een simpel algoritme om zo’n boom op te maken. Dit algoritme werkt met informatie gain op basis van entropy-based measure.

Een beslissingsboom gebaseerd CARS framework: een afzonderlijke boom voor elke gebruiker (ID3 gebaseerd op content based filtering), deze boom houdt rekening met de context. Dan collaborative filtering. De classifier heeft als input data van de vom <user, item, contextual features -> rating>.

De beslissingsboom bestaat uit 2 stappen: inductie en inference. Informatie gain van elk contextueel feature wordt gebruikt als splitting criteria.

# Examining sensor-based physical activity recognition and monitoring forhealthcare using Internet of Things: A systematic review

NOG DOEN

# Statistical Analysis of Window Sizes and Sampling Rates in Human Activity Recognition

<https://www.scitepress.org/Papers/2017/61485/61485.pdf>

De sampling rate van de rauwe data en de window size zijn cruciale beslissingen voor activity recognition. Ook feature sets en classifier selection spelen een rol in de output van activity recognition.

Het aantal data objecten in een window hangt af van de window size en sampling rate. In deze studie werden partial windows verwijderd. Partial windows zijn windows met incomplete data en hebben een kleinere grootte.

246 features extracted en gereduced tot 32 via correlation-based feature selection. Er zijn features in tijds en frequentie domein. Random Forest classifier geeft goede resultaten.

Variantie neemt toe samen met window size, de gewicht functie bij normaliseren moet dus hiervan afhangen.

Er werd een vergelijking gemaakt op basis van geslacht, leeftijd en BMI.

SAMPLINGF: 50

WINDOW SIZE: 10s

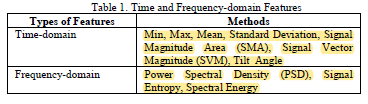
# Activity Recognition using Accelerometer Sensor and machine learning classifiers

<https://ieeexplore.ieee.org/abstract/document/8368718>

Enkel accelerometer wordt gebruikt en geen gyroscoop. Vroegere studies hebben bewezen dat meerdere sensoren nadelen heeft in identificeren van menselijke activiteiten. En er is bewezen dat accelerometer data genoeg is om de identificatie te doen.

Voor signaal segmentatie wordt gebruik gemaakt van een sliding window met 50% overlap. Er is bewezen dat dit goede resultaten levert.

Een volgende stap is feature selection. Hierbij worden feature berekent gebruik makend van time en frequentie domein features. Veel gebruikte zijn



Hierna doen we een normality test om na te gaan of de bekomen features passen in een normale distributie. Als we dit weten dan kan bepaald worden of we een parametische of niet parametische classificatie tools gebruiken. 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 geschikt.

Dimensionale reductie wordt gedaan met PCA.

Alle features zijn genormaliseerd in [0,1]

In deze studie werd een vergelijking gedaan tussen de performantie van de machine learning algoritmes op features voor en na dimensionale reductie. Ook werden performanties van algoritmes op time gebaseerde en frequentie gebaseerde features vergeleken. Hieruit wordt geconcludeert dat het frequentie domein betekeninsvollere info geeft.

# Integrating features for accelerometer-based activity recognition

<https://core.ac.uk/download/pdf/81111058.pdf>

FEATURE EXTRACTION

In het tijdsdomein worden 17 features berekend over elk window en voor elke as. Deze bevatten statistische features: gemiddelde, variantie, standaard deviatie. Envelope metrics: mediaan, interval maximum en minimum waarde, root main square. 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. De Fast Fourier Transform wordt gebruikt. De features zijn: 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 gebruikt. Features: power of signal, squared coefficients….?

DATA FILTERING

Om noise weg te filteren wordt een digitale filter gebruikt. High pass en low pass filter. (noise goed voor machine learning?)

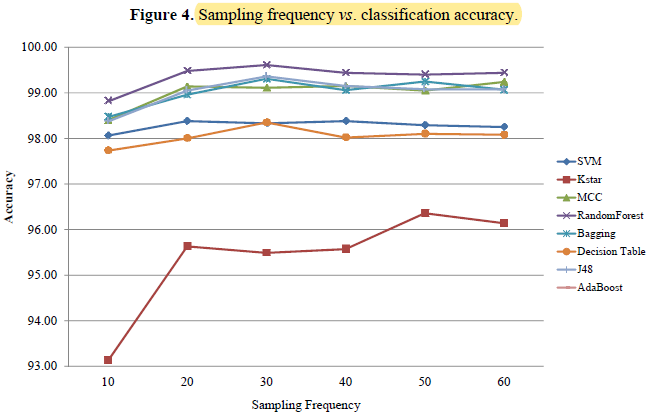
FEATURE SELECTION

Volgende selectiemodellen worden gebruikt: chi-square selection, correlation-based feature selection, reliefF selection, Information-gain based selection en gain-ratio-based selection.

# Real-Time Human Ambulation, Activity, and Physiological Monitoring: Taxonomy of Issues, Techniques, Applications, Challenges and Limitations

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3859040/>

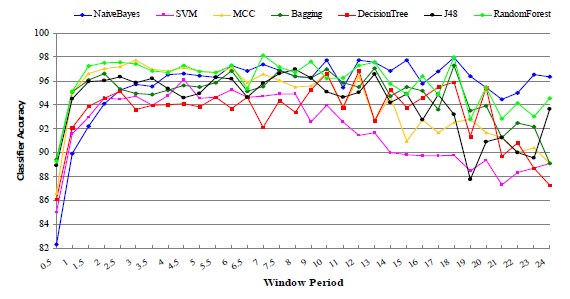
Classificatie accuracys krijgen een piek bij 20 Hz.



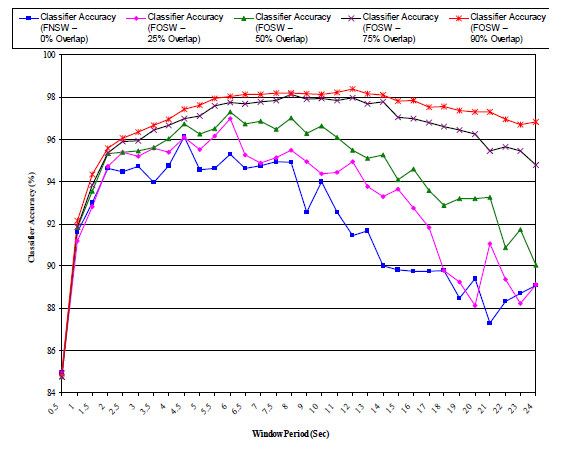
Data processing kan op het device gedaan worden maar hierbij zijn challenges door hardware limieten (limiet de data die kan gebufferd worden), de robuustheid van classificatie algoritmen en het aantal events dat kan geclassificeerd worden, energie consumptie.

Segmentatie algoritmen: BUp, SWAB

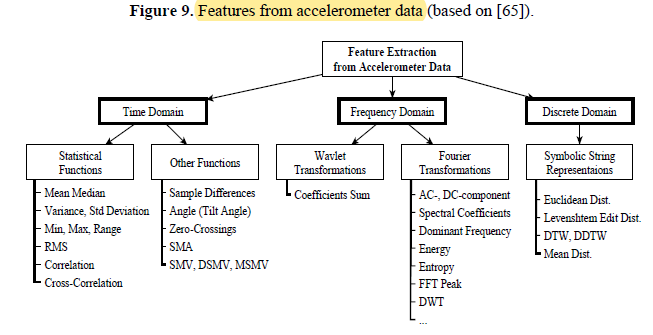
Window periode



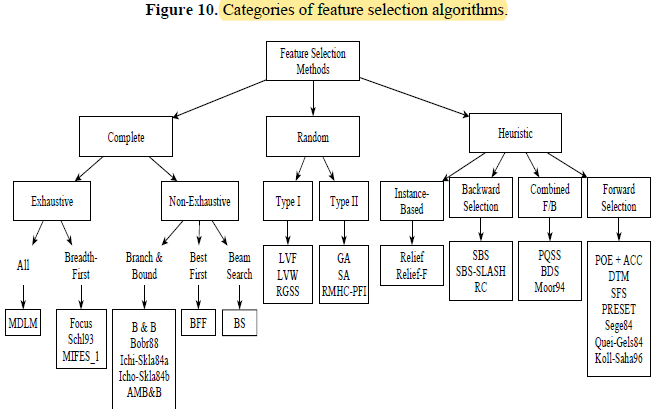
Overlap



Feature extraction

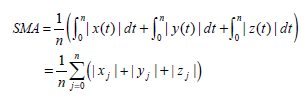


Feature selection algoritmen



SMA

Dit wordt gebruikt om een meetwaarde te bekomen die het level van activiteit weergeeft en dus het onderscheid kan maken tussen perioden van activiteit en inactiviteit. Eenheid = m/s²



SMV

Dit wordt gebruikt om de graad van bewegingsintensiteit te evalueren en wordt gebruikt om versnellings peak thresholds uit af te leiden. Eenheid = m/s²



# Validity of Accelerometry to Measure Physical Activity Intensity in Children With an Acquired Brain Injury

<https://journals.lww.com/pedpt/FullText/2017/10000/Validity_of_Accelerometry_to_Measure_Physical.8.aspx>

In deze studie werd zuurstof gebruik (VO2) gebruikt als metriek. Kinderen oefenden activiteiten uit met een indirect calorimeter (gas analyse systeem, breath by breath measure van respiratory volume en O2 en CO2 concentratie van uitgeademde lucht, het is een reliable measure of VO2), AG (accelerometer) en hart slag monitor. Verschillen in VO2 en AG vector magnitude werden gemeten. Het doel van deze studie was om specifieke activiteit instensiteits cut points af te leiden om acticiteitscounts in te delen in SED, LPA en MVPA.

Met deze data wordt MET berekend door het gemiddelde VO2 te delen door voorspelde MRM (resting metabolic rate). Adhv mets werd de classificatie uitgevoerd.

# Metabolic equivalent of task (METs) thresholds as an indicator of physical activity intensity

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6053180/>

Deze studie onderzoekt wat de thresholds zijn om de intensiteitsniveaus bij uitoegenen van activiteiten mee af te bakenen. De HRmax hier gebruikt is 208-(0,7\*age). Gemiddelde zuurstof inname werd gemeten en geconverteerd naar METs. Waarden voor afbakening werden gevonden die hoger lagen dan eerdere onderzoeken. Individuen met betere conditie hebben een hogere grens uitgedrukt in METs.

# A Comprehensive Study of Activity Recognition Using Accelerometers

NOG DOEN

# Comparison of Data Preprocessing Approaches for Applying Deep Learning to Human Activity Recognition in the Context of Industry 4.0

NOG DOEN

# The 220-age equation does not predict maximum heart rate inchildren and adolescents

NOG DOEN

# An Empirical Evaluation of the Prediction of Maximal Heart Rate

200- age geeft een lagere maxHR en is dus niet gevaarlijk.

NOG DOEN

# Age-predicted maximal heart rate revisited

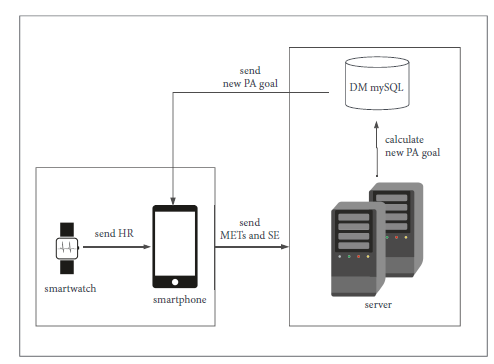
NOG DOEN

# Effects of Simple Personalized Goals on the Usage of a Physical Activity App

Deze studie ontwikkelde een app 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 calorien per dag. Met deze gegevens wordt het aantal calorien berekent die de persoon moet verliezen en het aantal calorien die moeten genomen worden om het uiteindelijke gewicht te behouden. Dit doel wordt niet bijgewerkt met gemonitoorde gegevens van calorie inname.

# Improving Physical Activity mHealth Interventions: Development of a Computational Model of Self-Efficacy Theory to Define Adaptive Goals for Exercise Promotion

<https://www.hindawi.com/journals/ahci/2019/3068748/>



De app 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 SE beliefs. Dit wekelijks doel kan gesplitst worden in dagelijkse doelen. Dit systeem houdt geen rekening met de context, wel laat het de user 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). . SE beliefs zijn vragen die gesteld worden na elke training sessie. De SE score is het gemiddelde van de gegeven antwoorden. De algemene richtlijnen zeggen dat 600 METs per week nodig is.. Er werd gebruik gemaakt van een Dynamic Descision Network (DDN), een opeenvolging van simpele Bayesian Networks. De DDN model heeft beslissingsvariabelen op elk training stage (= de PA goal voor die week). Het DDN model geeft een accurate weergave van de beslissingspunten.

# 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 het doel dus gezet door de gebruiker. Gebruikers starten te stappen met een duur afhankelijk van hun doen. Elke week worden 5 minuten bijgeteld.

# Context based Recommendation Methods: A Brief Review

NOG DOEN

Prefilteren, contextuel modeling, post filtering

# Context-Aware Mobile Recommender System Based on Activity Patterns

Apps die contectuele info in rekening brengen zijn context aware apps. Er zijn 2 manieren van computing: actief (de app adtapts onmiddellijk) en passief (de context info kan later opgehaald worden). In deze studie werd een widget gemaakt die user activiteit opslaat en hiermee voorspellen welke acties de user gaat ondernemen. De data die bekeken wordt is sms, calls en apps. Het recommendation algorimte is volledig gefocust op sql functionaliteit. Bij vijvoorbeeld het recommenden van de meest waarschijnlijke call wordt de totale duratie van elke oproep, gefilterd door dag vd week en gegroepeerd met contact nummer. De grootste dureatie wordt dan gegeven als voorspelling.

Opzoekingen

# Context aware reccommendations

CONTEXT: tijd, weer, huidige activiteit

* 1 keer per week sessiedata posten naar firestore (tijd, score, activiteit)
* 1 keer per week recommandations genereren in backend en posten naar firestore
* In app op tijdstip recommandation: controleren activiteit user en weer

= slecht 🡺 opnieuw recommendations oproepen (na tijdstip) en daaruit 1 kiezen

PROBLEEM: hoe recommendation als user te lang stilzit? Want kan niet op voorhand berekenen

PROBLEEM: hoe weer en huidige activiteit realtime in rekening brengen? Want algo moet op voorhand runnen want anders geen correcte optelling van aantal mets.

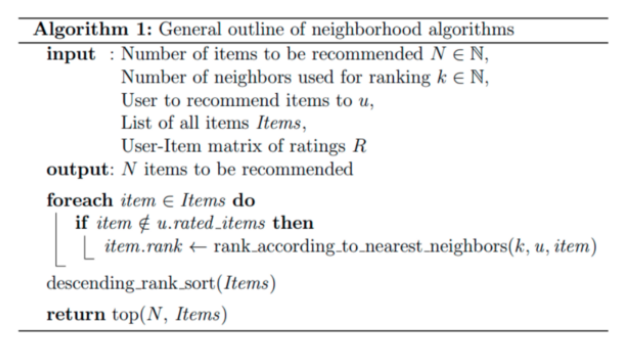
**1e poging:**

|  |  |
| --- | --- |
| **FIRESTORE** | collectie in user collectie met documenten per sessie:  start, end, activity, mets |
| **BACKEND** | * Bereken per activiteit gemiddeld aantal mets per min * Bereken per activiteit gemiddelde duur per sessie * Deel activiteiten in per interval (gemiddelde duur per sessie) * Rank activiteiten volgens aantal uur beoefent * Overloop activiteiten en neem tijdsinterval (gemiddelde duur per sessie) waarin ze het meest beoefend werd tot als mets goal voor die week bereikt is |

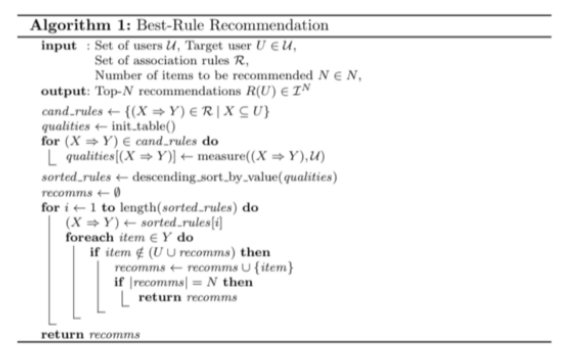
<https://medium.com/recombee-blog/machine-learning-for-recommender-systems-part-1-algorithms-evaluation-and-cold-start-6f696683d0ed>

* **content-based filtering:** cold start????
* **collaborative filtering:** historische data?

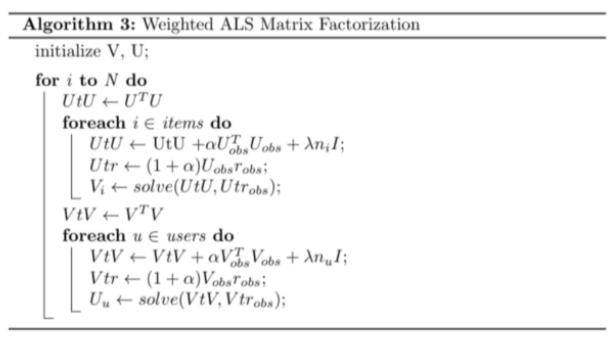
k nearest neighbors



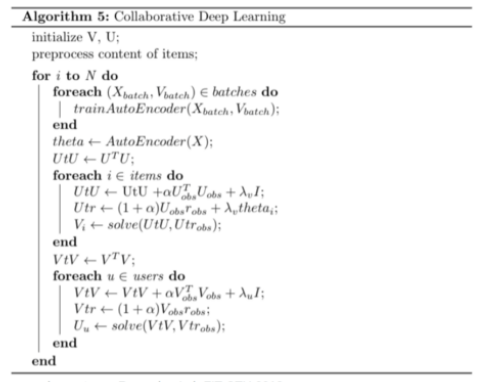
Best rule recommendation (Association Rules)



matrix factorization



Neural network



<https://developers.google.com/machine-learning/recommendation/overview/terminology>

* ITEMS = activities
* QUERY = historische data, userID, tijd, (weer, huidige activiteit)
* CANDIDATE GENERATION

op basis van query: content-based filtering (*items*) en collaborative filtering (*items* en *queries* dus ook userinfo in rekening houden)

user voorstellen in db met velden: kan zo kijken welke users similar zijn

Items (activiteiten met features) bijhouden in firebase

* SCORING
* RE RANKING

# Rope skipping moves

<https://buyjumpropes.net/resources/jump-rope-tricks-and-tips/>

**normal**

**side swing**

**crossover**

**full turn**

**caboose**

**leg-over**

# Machine learning

## Adaboost

<https://mccormickml.com/2013/12/13/adaboost-tutorial/>

adaboost combineert verschillende zwakke classifiers (maar toch beter dan random geussing) in 1 sterke classifier. Adaboost helpt met de keuze van de training set voor elke nieuwe classifier gebaseerd op de resultaten van de vorige classifier. Het bepaalt hoeveel gewicht wordt gegeven aan elk resultaat van een classifier.

## desicion trees

<https://medium.com/@chiragsehra42/decision-trees-explained-easily-28f23241248>

desicion tress leren van data door een sinus curve op te stellen met if then else rules.

## KNeighborsClassifier

<https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761>

Dit algorimte gaat ervan uit dat gelijkaardige datapunten dicht bij elkaar liggen.

## Logistic Regression

<https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python>

binary classes

## Neural Networking

<https://towardsdatascience.com/first-neural-network-for-beginners-explained-with-code-4cfd37e06eaf>

## Stochastic Gradient Descent

<https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31>

## Random Forest

<https://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/>

## support Vector Classification

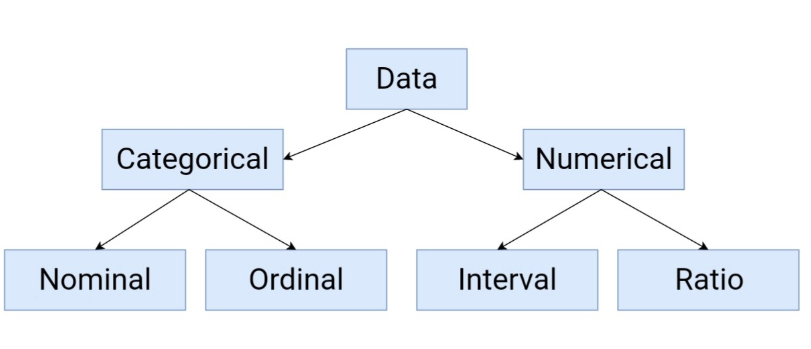
<https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496>

## preprocessing

<https://towardsdatascience.com/data-preprocessing-concepts-fa946d11c825>

Machines begrijpen rauwe data niet. Data preprocessing is de aaneenschakeling van stappen die de data zo transformeren zodat de features van de data kunnen geinterpreteerd 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. Andere benamingen voor features zijn variabelen, velden, attributen, dimensies… (WIKIPEDIA??). Een feature is een individueel meetbare caracteristiek van het event dat zich voordeed. Er zijn verschillende types features. Een eerste ondervedeling is categorisch en numeriek. Een categorische feature is discreet en heeft dus slechts een beperkt aantal mogelijke waarden. Een numerieke feature is continue en worden gerepresenteerd door nummers. Zoals te zien op de figuur kunnen deze ook nog eens ondererdeeld worden. Een categorische variabele kan nominaal of ordinaal zijn. Het verschil tussen beide is de aanwezigheid van ordering. Een numerieke variabele kan interval of ratio zijn. Het verschil tussen deze 2 is dat er bij ratio …. (ZIE GEGEVENSSTRUCTUREN)



Niet alle stappen van data preprocessing zijn toepasbaar op elk probleem. Dit is zeer afhankelijk van het soort data waarmee gewerkt wordt. Volgende stappen worden ondernomen bij preprocessing.

DATA QUALITY ASSESSMENT

Data komt meestal van verschillende bronnen die al dan niet betrouwbaar zijn. De data zal dus ook meestal in verschillende formaten binnenkomen. Problemen komend uit menselijke fouten, limieten van meet toestellen of flaws in data collectie proces. Volgende methoden proberen deze errors te maskeren.

* Missing values

Missig values komen voor door fouten tijdens data collectie of door een data validatie regel waardoor bepaalde punten niet geldig zijn. We kunnen volledige rijen met missing values verwijderen. Als veel datapunten incompleet zijn is deze strategie niet aan te raden. Als slechts een klein percentage van de data putnen te kampen heeft met missing values dan kan ook gekozen worden voor interpolatie methoden. Hiermee zal de waarde van de feature geschat worden aan de hand van beschikbare waarden voor die feature. Men kan ook kiezen om de missing value in te vullen met het gemiddelde, de mediaan of de modus van de feature.

* Inconsistente waarden

Dit komt vaak voor bij menselijke fouten. Bijvoorbeeld een telefoonnummer dat werd ingevuld in de adreskolom. Om dit te detecteren is het nodig om te weten welk datatype een bepaalde feature heeft en of dit hetzelfde is voor alle data objecten.

* Duplicate waarden

Sommige data objecten kunnen duplicaten zijn. Bijvoorbeeld als een persoon een formulier meerdere keren verzend. Deduplication is het proces van handelen met duplicaten. Duplicaten worden verwijderd om deze data objecen geen voordeel of bias te geven bij machine learning algoritmen.

FEATURE AGGREGRATION

Dit is het samennemen van grote hoeveelheden data objecten tot 1 samenvattend data object. Hierdoor is er een vermindering van geheugen consumptie en processortijd. Aggregaties bieden een high level view van de data die stabieler is dan 1 individueel data object.

FEATURE SAMPLING

Met sampling kan de dataset gereduced worden tot een punt waarbij we een beter maar duurder machine learning algoritme kunnen gebruiken. Sampling moet zo gedaan worden dat de eigenschappen van de originele dataset behouden worden, de sample is representatief. Dit wordt bekomen door de juiste sample size 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. Bij zonder vervanging wordt een geselecteerd data object verwijderd uit de dataset. Met vervanging steekt het terug. Bij ongebalanceerde datasets is deze 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* houdt rekening met deze distributie. Deze techniek verdeelt de dataset in groepen en neemt uit elke groep een gelijk aantal objecen ook al is de grootte verschillend.

FEATURE EXTRACTION

<https://www.kaggle.com/pmarcelino/data-analysis-and-feature-extraction-with-python>

FEATURE SELECTION

DIMENSIONALITY REDUCTION

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 maps de dataset op een lagere dimensionele ruimte. Dit wordt gedaan door nieuwe features te creeren die een combinatie zijn van de oude features. Volgende technieken worden hiervoor gebruikt: principal component analysis en singular value decomposition.

Data analyse algoritmes werken beter met lagere dimensionaliteit. Irrelevante features en noise zijn geelimineerd. Modellen gebouwd op basis van lage dimensionele data zijn makkelijker te verstaan. De data is makkelijker te visualiseren.

FEATURE ENCODING

Het doel van data preprocessing is om de data te encoderen zodat het naar een staat kan gebracht worden die de machine verstaat. Er zijn algemene normen en regels die gevolg 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 order preserving verandering van waarden ondergaan. Interval features kunnen getransformeerd worden met een wiskundige formule waarbij hun nulpunt behouden wordt (fahrenheit -> celsius). Ratio features kunnen geschaald worden met wiskundige formules (lengt: meters of feet…)

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/underfitting. De validatie data wordt gebruikt om het models hyperparameter (bij activity recognition: window size bijvoorbeeld) gekozen en verbeterd. 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 we hebben. 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. Dit ratio moet bekomen worden door alle details van het model en dataset te bekijken.

* In het geval van activity recognition op basis van accelerometer data hebben we te maken met continue features (x,y,z en time) van het type ratio ??? . Tijdens dit experiment komt data enkel van de polar M600 (wel van verschillende personen) dat via de messaging api wordt overgebracht naar een applicatie op een android gsm. De applicatie op de smartwatch die deze data verzamelt en verstuurt wordt via bluetooth gerunt. Dit kan eventueel een onstabiele connectie veroorzaken bij grote afstanden. Manipulatie van data objecten zou dus mogelijk zijn (?) . Hierdoor moet er aandacht bestaat worden aan het DATA QUALITY ASSESMENT PROCES (missing values, inconsistente waarden, duplicate waarden). FEATURE AGGREGATION is hier zeker van toepassing. 1 individueel datapunt biedt weinig informatie. SAMPLING sample uit window nemen? . Doordat hier slechts 3 originele features aanwezig zijn, is DIMENSIONALITY REDUCTION niet nodig (in artikel wordt eerst feature extraction gdn en dan dimensionality reduction) . FEATURE ENCODING?? .

Het model zal goed moeten getraind worden, dus een grote training dataset is gewenst. Er is (afhankelijk van het gebruikte algoritme) een aantal hyperparameters. Window size is hier sowieso 1 van, onafhankelijk van het gebruikte algoritme. De validation dataset zal bijgevolg waarschijnlijk niet enorm groot zijn. Testen is ook een belangrijke component.

Pandas + tensorflow har (normalize function + segmentation): <http://aqibsaeed.github.io/2016-11-04-human-activity-recognition-cnn/>

### Scikit-learn

<https://scikit-learn.org/stable/modules/preprocessing.html>

het sklearn.preprocessing package biedt functies en klassen aan die rauwe feature vectors transformeren in een representatie meer geschikt voor downstream estimators.

Als outliers aanwezig zijn, zijn robust scalers of transformers beter geschikt (deze data geen outliers?)

STANDARDIZATION OR MEAN REMOVAL AND VARIANCE SCALING BEKIJKEN

Algoritmes geimplementeerd in scikit learn hebben gestandaardiseerde data nodig, data die lijkt op een standaard normale distributie: gaussian met 0 gemiddelde en 1 variantie. In de praktijk wordt de data getransformeerd door het gemiddelde te verwijderen en het dan te schalen door deling van de standaard deviatie.

Een feature met een variantie niet in dezelfde order als de andere, kan dominant zijn. (SCALE). De preprocessing module heeft een klasse STANDARDSCALER die de transformer API implementeerd. Deze berekent het gemiddelde en standaard deviatie op een training set om dan dezelfde transformatie te kunnen doen op de test data.

SCALING FEATURES TO A RANGE

Een alternatieve manier is om features te schalen tussen een min en max. Dit gebruiken om robustness te hebben voor zeer kleine standaard deviaties/features en behouden van 0 entries in sparse data. (hier niet van toepassing?)

SCALING SPARSE DATA (geen sparse data hier)

Centreren van sparse data vernietigd de sparse eigenschap, maar schalen kan wel gedaan worden. MaxAbsScaler is hiervoor ontwikkeld.

SCALING DATA WITH OUTLIERS (hier geen outliers)

Schalen met het gemiddelde en variantie zal in dit geval niet werken. Robust\_Scale gebruikt meer robuuste estimated voor de center en range data.

SCALING vs WHITENING BEKIJKEN

Soms is het niet genoeg om de data individueel te centreren en te schalen aangezien een model een assumptie kan maken gebaseerd op de lineaire (in)relatie van de features. => sklearn.decomposition.PCA met whiten=true wordt gebruikt om lineaire correlatie te verwijderen.

NON-LINEAR TRANSFORMATION

2 types: quantile transforms en power transforms. Beide zijn gebaseerd op monotonic transformaties van de features en behouden dus de rang.

Quantile transforms steken alle features in dezelfde distributie.

Power transforms steken data in een distrubutie dicht bij een gaussian distributie.

….

NORMALIZATION BEKIJKEN

Dit is schalen van individuele samples naar de unit norm. Dit is handig als het dot product zal gebruikt worden of een andere kernel om de gelijkheid tussen individuele samples te bepalen.

ENCODING CATEGORICAL FEATURES

Converteren van categorische features naar int waarden (0 – n-1) 🡪 one hot encoding

DISCRETIZATION

Verdelen van een continue feature in discrete waarden. Preprocessing me een discretizizer kan nonlineariteit introduceren in lineaire modellen.

K-BINS DISCRETIZATION

Data verdelen in k bins, output is hot encoded

FEATURE BINARIZATION

Dit is thresholden van numerieke features om boolean features te bekomen. Dit is bruikbaar voor estimators die willen dat input data gedistribueerd is volgens de bernoulli distributie

IMPUTATION OF MISSING VALUES BEKIJKEN

GENERATING POLYNOMIAL FEATURES BEKIJKEN

Voeg complexiteit toe aan het model door nonlineaire features te gebruiken. Soms zijn enkel interactie features nodig.

Welk model: <https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html>

**Goal prediction**

|  |  |
| --- | --- |
| SGD REGRESSOR | Deze regressie estimator implementeert een plain stochastic gradient descent learning routine dewelke verschillende loss functies en penalties ondersteunt.  Het is goed voor training sets groter dan 10 000 samples.  De loss functie kan meegegeven worden via de loss parameter (squared\_loss, huber of epsilon\_insensitive). Huber en Epsilon kunnen gebruikt worden voor rubuste regressie. Parameter epsilon stelt de breedte van de intensieve regio in (hangt af van de schaal van de target variables)  Averaging kan aangezet worden  Voor regressie met squared loss and l2 penalty, is een andere variant met een avareging strategie beschikbaar.  SGD is efficient  Stopping criteria zijn beschikbaar.  VOORDELEN   * Verschillende loss functies en penalties * Groot aantal training samples ( > 10 000)   PRECONDITIES   * geschaalde data * regularizatie term alpha (10.0\*\*-np.arange(1,7))gridSearchCV * max\_iter = np.ceil(10\*\*6 / n)   Loss  Epsilon  Average  Early\_stopping  N\_iter\_no\_change  Tol  Max\_iter |
| LASSO | VOORDELEN   * afhankelijk van een kleiner aantal features   alpha  Eps  Fit\_intercept  Positive  CROSS VALIDATION: lassoCv, LassoLarsCV, LassoLarsCV |
| ELASTICNET | Als features gecorreleerd zijn  alpha  Eps  L1\_ratio  Fit\_intercept  CROSS VALIDATION: ElasticNetcv |
| RIDGEREGRESSION | Alpha  alphas=array([1.e-06, 1.e-05, 1.e-04, 1.e-03, 1.e-02, 1.e-01, 1.e+00, 1.e+01,  1.e+02, 1.e+03, 1.e+04, 1.e+05, 1.e+06] |
| SVR | VOORDELEN   * Werkt goed in hoog dimensionale ruimte   nvt   * Werkt goed in gevallen waar het aantal dimensies groter is dan het aantal datapunten   nvt   * Geheugen efficient: gebruikt slechts een subset van datapunten in de beslissingsfunctie (support vectors)   Support vectors = degene de dichtst bij hyperplane liggen. Deze worden gebruikt in de decision functie   * Veelzijdig: verschillende kernel functies (transformeren data van zodat hyperplane een lineaire vorm heeft) kunnen gespecifieerd worden voor de beslissingsfunctie   NADELEN   * Als het aantal features groter is dan het aantal datapunten, moet over fitting voorkomen worden in het kiezen van een kernel functie en een regulatie term (maat van belangrijkheid die gegeven wordt aan misclassificaties) is nodig   nvt   * SVMs geven geen directe kans estimates, ze worden berekend adhv een dure five fold cross validatie   PRECONDITIES   * Zowel dichte als ijle vectoren zijn mogelijk als input   Dicht, want om naar een ver punt te gaan, moeten de tussenliggende er ook zijn (acceleration)   * Schalen van X-train en X-test   Standardize want features schalen   * Y moet float zijn   **linearSVR**  = sneller dan SVR, maar alleen lineaire kernels  **SVR**   * data moet c ordered contiguous zijn anders wordt gekopieerd (flags van numpy array) * kernel cache size: cache\_size groter zetten om sneller te werken |
| ENSEMBLEREGRESSORS | **randomForestRegressor**  **ExtraTreesRegressor**  Max\_features = random subset of features, lager betekent meer vermindering in variantie + meer bias  N\_estimators = aantal trees  Max\_depth  Min\_samples\_split  Bootstrap  Oob\_score  max\_leaf\_nodes  min\_samples\_leaf  n\_jobs  kan gebruikt worden voor feature selection (MDI) : feature\_importances\_  **AdaBoostRegressor**  N\_estimators = aantal weak learners  Learning\_rate  Base\_estimator  Max\_depth  Min\_samples\_split |
| NEURAL NETWORK | VOORDELEN   * non lineaire boundary * realtime learning (partial fit)   NADELEN   * niet convexe loss functie met meer dan 1 lokaal min * tuning van hyperparameters * gevoelig voor schaling features   PRECONDITIES   * schalen data * regularizatie parameter (gridsearchCV)   **MLPRegressor**  Alpha = 10.0 \*\* -np.arange(1, 7)  solver:  L-BFGS voor kleine datasets  Adam voor grote  SGD met correcte learning rate tuning  hidden\_layer\_sizes  random\_state  predict\_proba  warm\_start  max\_iter |

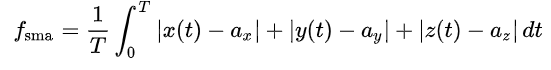
**Activity recognition**

|  |  |
| --- | --- |
| SVC | VOORDELEN   * Werkt goed in hoog dimensionale ruimte   nvt   * Werkt goed in gevallen waar het aantal dimensies groter is dan het aantal datapunten   nvt   * Geheugen efficient: gebruikt slechts een subset van datapunten in de beslissingsfunctie (support vectors)   Support vectors = degene de dichtst bij hyperplane liggen. Deze worden gebruikt in de decision functie   * Veelzijdig: verschillende kernel functies (transformeren data van zodat hyperplane een lineaire vorm heeft) kunnen gespecifieerd worden voor de beslissingsfunctie   NADELEN   * Als het aantal features groter is dan het aantal datapunten, moet over fitting voorkomen worden in het kiezen van een kernel functie en een regulatie term (maat van belangrijkheid die gegeven wordt aan misclassificaties) is nodig   nvt   * SVMs geven geen directe kans estimates, ze worden berekend adhv een dure five fold cross validatie   PRECONDITIES   * Zowel dichte als ijle vectoren zijn mogelijk als input   Dicht, want om naar een ver punt te gaan, moeten de tussenliggende er ook zijn (acceleration)   * Schalen van X-train en X-test   Standardize want features schalen  Eigenschappen van de support vectors in support\_vectors\_, support\_, n\_support (support vectors voor elke klasse).  **SVC en NuSVC** implementeren one-against-one voor multiclass classification. N\_class \* (n\_class-1)/2 classifiers worden aangemaakt die elk data van 2 klassen trainen.  Descicion function shape: transformeren output functie van aantal classifiers naar (n\_smaples, n\_classes)  **linearSVC** implementeert one-vs-the-rest multiclass strategy, wat trainen van n\_class modellen betekent. Me optie multi\_class='crammer\_singer' wordt een andere multiclass strategie geselecteerd. Beter one vs rest want minder tijd nodig. Coef\_ en intercept\_, dual\_coef\_  SCORES AND PROBABILITIES  Predict, predict\_proba, break\_ties  UNBALENCED PROBLEMS  Class\_weight, wample\_weight  Nvt  C is default 1 maar bij noisy data is het beter om dit te verminderen. Lage c maakt het beslissings vlak smooth en een hoge c probeert alle training samples te klasseren  Accelerometer = noisy data???  Gamma: hoe ver sample reikt om desicion boundary te bepalen  Random state  Loss, penalty, dual  **SVC, NuSVC:**   * data moet c ordered contiguous zijn anders wordt gekopieerd (flags van numpy array) * kernel cache size: cache\_size groter zetten om sneller te werken * SVC: unbalancet data 🡪 class\_weight=balenced   **linearSVC:**   * data wordt altijd gekopieerd (SGD classifier als niet wil) * niet zo gevoelig voor grote C |
| ENSEMBLE CLASSIFIERS | Averaging methods: forest of randomized trees  Boosting methods: adaBoost  AVERAGING METHODS  Diverse set van classifiers door randomness erin te steken. De uiteindelijke predictie is het gemiddelde van die classifiers.  **Random Forest & extra-tree**  Max\_features = random subset of features, lager betekent meer vermindering in variantie + meer bias  N\_estimators = aantal trees  Max\_depth  Min\_samples\_split  Bootstrap  Oob\_score  max\_leaf\_nodes  min\_samples\_leaf  n\_jobs  kan gebruikt worden voor feature selection (MDI) : feature\_importances\_  PRECONDITIES   * dicht of ijl   BOOSTING METHODS  **Adaboost**  N\_estimators = aantal weak learners  Learning\_rate  Base\_estimator  Max\_depth  Min\_samples\_split  Discrete vs real |
| NAIVE BAYES | VOORDELEN:   * hebben weinig data nodig   nvt   * snel * door decoupling van de klasse conditionele feature distributies betekent dat elke distributie kan benaderd worden door een 1 dimensionele distributie wat problemen voortkomend uit de curse of dimensionality tegengaat * nvt * slechte estimator   ?  **Gaussian Naive Bayes**  PRECONDITIES   * waarschijnlijkheidsdistributie features moet gaussian zijn * voor continue data?   **Multinomial Naive Bayes**  PRECONDITIES   * multinomially distributed data |
| KNEIGHBORS CLASSIFIER | VOORDELEN   * non parametric: goed voor problemen met irregular decision boundary   PRECONDITIES   * dicht of ijl   **kNeighborsClassifier**  hoge k is goed om noise te onderdrukken  hier noise?  Weights  Algorithm:   * *kd tree*   kleine D en grote N  intrinsic dimensionality en sparsity van de data  trager bij grotere k  groot aantal query points   * *ball tree*   grote D en grote N  intrinsic dimensionality en sparsity van de data  trager bij grotere k  groot aantal query points  Leaf size = wanneer veranderen naar brute-force |
| SGD CLASSIFIER | VOORDELEN   * efficientie * veel mogelijkheden voor code tuning * large scale en sparse datasets   NADELEN   * veel hyperparameters (regulation parameter en aantal iteraties) * gevoelig voor schaling   PRECONDITIES   * training data moet geshuffeld zijn voor fitting   shuffle = True   * geschaalde data * regularizatie term alpha gridSearchCV * max\_iter = np.ceil(10\*\*6 / n)   loss  penalty  coef\_ = model parameters  fit\_intercept  intercept\_ = bias  descision\_function = geef signed afstand tot hyperplane  predict\_proba  class\_weight  sample\_weight  average: beter veel groot aantal features  early\_stopping  n\_iter\_no\_change  max\_iter = np.ceil(10\*\*6 / n)  eta0  power\_t  MULTICLASS CLASSIFICATION: one versus all |
| KERNEL APPROXIMATION |  |
| NEURAL NETWORK | VOORDELEN   * non lineaire boundary * realtime learning (partial fit)   NADELEN   * niet convexe loss functie met meer dan 1 lokaal min * tuning van hyperparameters * gevoelig voor schaling features   PRECONDITIES   * schalen data * regularizatie parameter (gridsearchCV)   coefs\_  intercepts\_  solver:  L-BFGS voor kleine datasets  Adam voor grote  SGD met correcte learning rate tuning  alpha = regulizatie (vermijden van overfitten) 10.0 \*\* -np.arange(1, 7)  hidden\_layer\_sizes  random\_state  predict\_proba  warm\_start  max\_iter  MULTICLASS CLASSIFICATION: softmax |

### feature extraction

SMA <https://www.revolvy.com/page/Signal-magnitude-area>

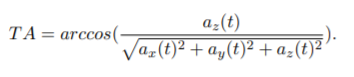
Dit is een statistische measure van de magnitude van een varanderende feature. De sma value is de genormaliseerde integraal. Magnitude is een maat die ordering aangeeft (is het object kleiner of groter dan andere). In dit geval geeft sma dus een ordering aan tussen de verschillende windows.



a is een offset correctie

TILT ANGLE

<http://www.univagora.ro/jour/index.php/ijccc/article/view/2787/1064>



FOURRIER TRANSFORMATIE

<https://ipython-books.github.io/101-analyzing-the-frequency-components-of-a-signal-with-a-fast-fourier-transform/>

De fourrier transformatie zal het signaal voorstellen in functie van de frequentie in plaats van de tijd.

POWER SPECTRAL DENSITY (PSD)

<https://link.springer.com/chapter/10.1007/978-981-10-4361-1_32>



<https://community.sw.siemens.com/s/article/what-is-a-power-spectral-density-psd>

PSD is een maat voor de kracht inhoud vs frequentie.

Dezelfde metingen gedaan met een andere frequentie resolutie resulteren in een signaal met verschillende amplitude. Bij hogere frequentie resolutie worden meer datapunten gebruikt. Hetzelfde signaal wordt onderverdeeld in kleinere stukjes maar het totaal blijft hetzelfde. Met de RMS metriek kan de totale inhoud van het signaal weergegeven worden. # Spectral lines = bandwidth/delta f = # datapunten. Spectral lines zijn discrete punten in het frequentie domein. Naarmate delta f stijgt zal het aantal spectral lines dalen en zal de zichtbare amplitude stijgen. Signaal met 1HZ heeft meer datapunten dan signaal met 8Hz (?). Power spectral density wordt gebruikt om het verschil ertussen weg te werken (representatie veranderen). PSD normaliseert de amplitudes by de frequentie resolutie. Eenheid = g²/Hz

<https://www.cygres.com/OcnPageE/Glosry/SpecE.html>

PSD toont de kracht van de variaties (energie) als functie van de frequentie. Op welke frequentie zijn variaties krachtig en op welke minder.

ENTROPY

<https://amethix.com/entropy-in-machine-learning/>

<https://stackoverflow.com/questions/49973537/shannons-entropy-on-an-array-containing-zeros>

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 zeker 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 werd 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).

Explanatory data analysis: 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 geen relationship hebben. 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. MI is eigenlijk de relatieve entropy tussen de joint distributie p(x,y) en de product distributie p(x) . p(y).

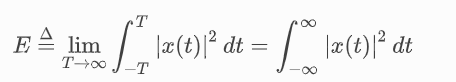
Bij feature selection wordt ook gebruikt gemaakt van entropy. Sommige features zijn zeer correlated, andere dragen enkel noise en bijna geen signaal. Hogere dimensionale vectoren lijken meer en meer op elkaar naarmate hun dimensionale ruimte vergroot (curse of dimensionality). Feature selection detecteert relevante features.

Subset extraction / sampling maakt ook gebruik van entropy. Bijvoorbeel 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 hidden bias in de originele dataset leid 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 het ok.

Modeling: nog doen

ENERGY

<https://scikit-dsp-comm.readthedocs.io/en/latest/nb_examples/Continuous-Time%20Signals%20and%20Systems%20using%20sigsys.html#Energy-and-Power-Signals>



GIBBS VERSCHIJNSEL

## Classification

<https://www.youtube.com/watch?v=0Lt9w-BxKFQ>

## Random forest classifier

## Classification report

Hoe goed zijn de voorspelde waarden. Testen tegenover juiste waarden

## Confusion matrix

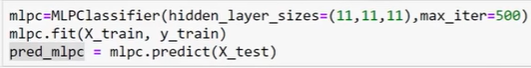
Hoeveel voorspellingen waren correct en hoeveel niet per klasse.

## SVM classifier

## Neural network

<https://scikit-learn.org/stable/modules/neural_networks_supervised.html>

<https://towardsdatascience.com/how-does-back-propagation-in-artificial-neural-networks-work-c7cad873ea7>



Een neural netwerk wek op een trainingset van m dimensies en dus m verschillende features. Er kunnen meerdere niet lineaire lagen gebruikt worden. Deze worden dan hidden layers genoemd. Elke neuron in een hidden layer transformeert de waarden van de vorige laag met een gewogen lineaire sommatie.



Gevolgd door een niet lineaire activatie functie. De output layer zal deze waarden transformeren naar output values

Ook is er sprake van een bias tussen lagen. Zonder deze bias vormen de waarden gewoon een rechte door de oorsprong.

Deze library kan modellen leren in real-time gebruikmakend van partial\_fit.

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

Parameters zoals aantal hidden neurons/lagen en iteraties moeten getuned worden.

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

Een MLP kan trainen gebruikmaken van backpropagation. Dit algoritme zal de gewichten fijntunen gebaseerd op de hoeveelheid errors in de vorige iteratie. De batch gradient descent optimization functie zal bepalen in welke richting de gewichten moeten aangepast worden om een minder grote error te hebben.

In elke laag moet de node die de meeste errors veroorzaakt heeft beboet worden door een kleiner gewicht eraan te koppelen.

Er zijn 2 datasets nodig. Array X bevat de training samples en array Y bevat de classification labels.

<https://scikit-learn.org/stable/modules/multiclass.html#multiclass>

Multi-class classification (meer dan 2 klassen) is mogelijk door de softmax functie te gebruiken.

<https://towardsdatascience.com/journey-to-the-center-of-multi-label-classification-384c40229bff>

Ook is multi-label classification ondersteund zodat een sample tot meer dan 1 klasse kan behoren. Er zijn verschillende technieken mogelijk

1. ONEVSREST

Onderverdelen in meerdere binaire classificatie problemen, 1 per categorie. Kies de classifier met de grootste waarschijnlijkheid. Er wordt hier verondersteld dat de labels niets met elkaar te maken hebben.

1. BINARY RELEVANCE

Verzameling van single label binaire classifiers is getraind. De unie van al deze klassen is de multi-label output. Ook hier wordt geen rekening gehouden met de relatie tussen labels

1. CLASSIFIER CHAINS

Hier wordt een reeks van classifiers gebruikt waarbij iedere classifier in de reeks rekening houdt met de voorgaande

1. LABEL POWERSET

Hier wordt rekening gehouden met relaties tussen class labels

1. ADAPTED ALGORITHM

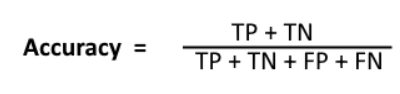
Adapting single label klassificatie algoritmes naar multi label door gebruik te maken van kost/beslissings functies. Bijvoorbeeld de lazy learning approach ML-KNN

## Regression

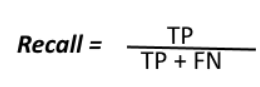
## Accuracy

<https://www.geeksforgeeks.org/confusion-matrix-machine-learning/>

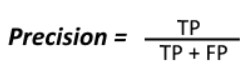
accuracy kan problemen geven omdat er evenveel gewicht gegevens wordt aan elk soort error.



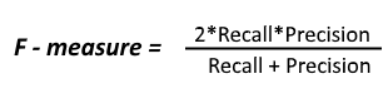
Recall is het ratio van het totaal aantal correct geclassificeerde positieve samples en het totaal aantal positieve samples.



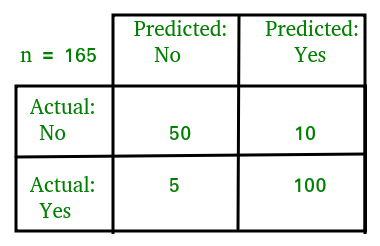
Precision is het ratio van het aantal correct geclassificeerde positieve samples en het totaal aantal voorspelde positieve samples.



F measure stelt recall en precision voor



Vb



# Reinforcement learning

<https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/>

<https://www.kaggle.com/mikebaik/openai-gym-env-reinforcement-learning-environment>

<https://gym.openai.com/>

<https://adventuresinmachinelearning.com/reinforcement-learning-tensorflow/>

Q-learning: reward tabel

|  |  |  |
| --- | --- | --- |
| **Hogere goal** | **Lagere goal** | **Zelfde goal** |
|  |  |  | **Goal achieved** |
|  |  |  | **Goal not achieved** |
|  |  |  | **Goal almost achieved** |

INVERSE REINFORCEMENT LEARNING

<https://thegradient.pub/learning-from-humans-what-is-inverse-reinforcement-learning/>

<https://thinkingwires.com/posts/2018-02-13-irl-tutorial-1.html>

# Dynamic descision network (bayes server)

<https://www.bayesserver.com/code/>

# Feature extraction vs feature selection

<https://quantdare.com/what-is-the-difference-between-feature-extraction-and-feature-selection/>

# Feature extraction techniques

PCA, ICA, LDA, LLE, t-SNE

# Scoring system

* Google fit: heart points
* Bereken met heart zones

# Calculating Physical Activity Intensity

<https://www.diversalertnetwork.org/health/heart/calculating-physical-activity-intensity>

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 gaan, top atleten kunnen tot 20 gaan. Activities kunnen worden geclasseerd in hoeveel mets ze vergen. Een individues peak aerobic capaciteit wordt uitgedrukt in maximum oxygen uptake terwijl hij excercised. VO2 max meten moet gebeuren door stricte protocollen te volgen in een sport ab. Dit is zeer time consuming en wordt dus enkel in speciale omstandigheden gedaan.

Het is mogelijk om een relatieve benadering van de intensiteit door de effecten op de hartslag en respiratie rate te meten. De talk test: 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. Heart rate test: 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 bmp. Submaximal excercise test: dit wordt gebruikt om de maximum excercise capaciteit te berekenen zonder over de grens van 0,85\*max heart rate te gaan. Door monitoren van hartslag kan de maximum excercise capaciteit gextrapoleerd worden mbv verschillende methoden. Deze waarde heeft gelimiteerde value nog verder

# MEASURING PHYSICAL ACTIVITY INTENSITY

<https://healthandfitnessohio.com/measuring-physical-activity-intensity/>

Voor gemiddeld intensieve activiteit is de hartslag tussen 50% en 70% van de maximum hartslag. Voor vigorous intensity activiteit is de hartslag tussen 70% en 85% van de maximum hartslag.

# Heart rate zones

<https://www.polar.com/blog/running-heart-rate-zones-basics/>

heart rate zones zijn een manier om te monitoren hoe hard je traint. De running heart rate 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 ook berekend worden. Tussen deze waarden liggen de 5 zones. Er zijn verschillende manieren om deze zones af te bakenen. Een simpele manier 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 je voor uren zou kunnen uithouden. Deze threshold is wanneer de hoeveelheid lactate in het bloed begint te stijgen en de level van intensiteit waaraan anaerobic energie pathways helpen in energie productie. 30 bpm aftrekken van je lactate threshold heart rate geeft je een benadering van de aerobic threshold.

Anaerobic threshold is de hoogste exercise threshold die je kan volhouden voor een periode zonder lactate opbouwing in je bloed. Spieren verbranden suiker op 2 manieren: aerobically (zuurstof) en anaerobically (zonder zuurstof)

Heart rate zone 1 (50-60% HRMAX) is de zeer lichte intensiteitszone. Trainen in deze zone zal de recovery boosten 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 capillary densiteit stijgen. Heart rate zone 3 (70-80% HRMAX) is de gemiddelde intensiteitszone. Deze zone bevordert de efficientie van de bloed circulatie in het hart en skelet spieren. Dit is de zone waar pesky lactic acid 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 lopen. In deze zone wordt snelheids uithoudingsvermogen getraint. Het lichaam wordt beter in het gebruik van carbohydraten voor energie. Je zal hogere levels van lactic acid 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 voor atleten.

# Overige bronnen

Accelerometer sensor: <https://examples.javacodegeeks.com/android/core/hardware/sensor/android-accelerometer-example/>

intentService:

<https://guides.codepath.com/android/starting-background-services#creating-an-intentservice>

# Google activity recognition client

<https://developers.google.com/android/reference/com/google/android/gms/location/ActivityRecognitionClient.html>

<https://developer.android.com/reference/android/app/PendingIntent.html>

Er zijn 2 APIs: *the activity recognition transition API* en *the activity recognition sampling API*.

The activity recognition transition API heeft betere accuracy en vergt minder energie van het toestel (+ engineering productivity). Er wordt een notificatie gegenereerd als de gebruiker zijn/haar activiteit veranderd. De API call requestActivityTransitionUpdates heeft als eerste parameter een activityTransactionRequest waarin alle activiteiten in vermeld staan die belangrijk zijn (bv VEHICLE, ENTER). 2e parameter is 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 call. De functionaliteit moet dus beschreven zijn in een service. FLAG\_UPDATE\_CURRENT = maar 1 pendingintent tegelijk actief.

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 grotere samplingperiode, hoe minder energieverbruik. Individuele voorspellingen kunnen noisy zijn. Het is dan ook aangewezen om een eigen filter te schrijven dat hiermee kan omgaan (omdat raw data). 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 waar kans beschikbaar

# MATLAB activity detection project with accelerometer data

<https://nl.mathworks.com/matlabcentral/fileexchange/51079-activity-detection-in-matlab>

pandas: <https://machinelearningmastery.com/how-to-load-and-explore-a-standard-human-activity-recognition-problem/>

# firebase

<https://firebase.google.com/docs/firestore/data-model>

<https://firebase.google.com/docs/firestore/quickstart>

android app moet toegevoegd worden aan het firebase project. In app gradle: database depencie toevoegen om database te kunnen gebruiken

file -> project structure -> Cloud -> enable firebase service

Firebase.setAndroidContext(this)

Firebase fire = new Firebase(“link tod b”)

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

# Workmanager

<https://developer.android.com/topic/libraries/architecture/workmanager/basics.html>

# pandas framework

<https://www.learndatasci.com/tutorials/python-pandas-tutorial-complete-introduction-for-beginners/>

Het panda framework maakt gebruik van 2 componenten: series en dataframes. Een series kan gezien worden als een kolom en een dataframe als een multidimensionale tabel, een collectie van series.

## Create dataframe from scratch

Met een dict. De keys zijn dan kolommen en values zijn arrays die de waarden in de kolommen voorstellen. Via pd.DataFrame(dict) kan hieruit een dataframe gemaakt worden. De index is de nummering van rijen. Standaard is dit een reeks getallen beginnend bij 0. Om zelf een index te genereren kan gebruik gemaakt worden van een tweede parameter in de dataframe functie aanroep: index. Hiermee kan een array meegegeven worden bestaande uit de index waarden.

## Locate rij

Een bepaalde rij ophalen kan gemakkelijk via loc ( df.loc[‘rijindex’] ). Hierbij wordt de index gebruikt

## How to read in data

De data kan ook uit een csv file gehaald worden.



CSVs hebben geen indexen dus moeten we de index\_col instellen. De kolom die hier gespecifieerd wordt, zal dan dienen als index.



Indien de data in een json formaat beschikbaar is, is een andere functie aanroep nodig.



Hier is het niet nodig om manueel een index te specifieren. Indexes werken in JSON dmv nesting.

Data kan ook uit een SQL database komen bv SQLite. Eerst moet connectie gemaakt worden via sqlite3. Via een select query kan dan een dataframe gemaakt worden.

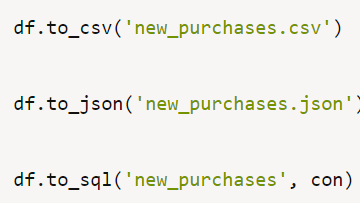


De index kan hierna gespecifieerd worden of op dezelfde manier als bij csvs.



Deze manier van achteraf een index instellen kan toegepast worden op elke dataframe.

## Converting back to a CSV, JSON or SQL



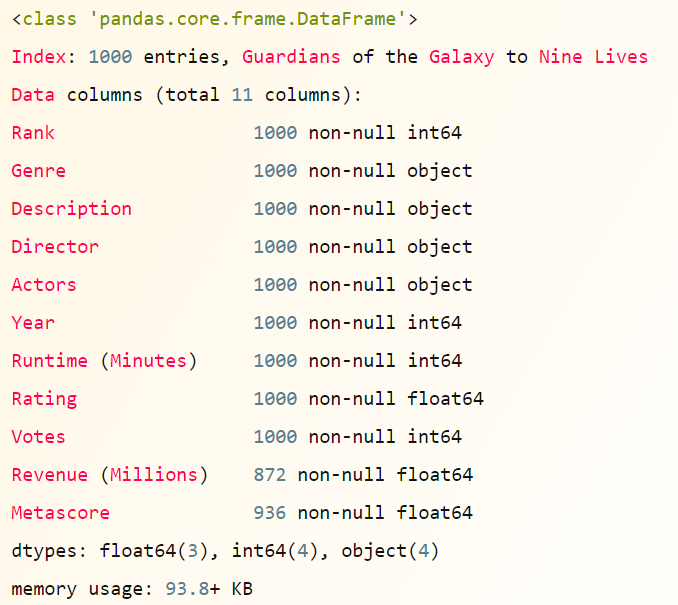
Bij deze operaties zal een nieuw file gecreerd worden of een nieuwe tabel in de SQL database.

## Viewing your data

Met head() kunnen de eerste rijen gevisualiseerd worden. Dit is beter dan het volledige dataframe visualiseren aangezien dit zeer groot kan zijn. Om de laatste rijen te zien kan tail() gebruikt worden.

## Getting info about your data

Met info() worden de essentiele details (aantal rijen/kolommen, aantal non-null waarden, welk type data in iedere kolom…) van de dataset getoond.



Shape heeft als output een tuple van (rijen, kolommen)

## Handling duplicates

Append(df) zal de entries van df onderaan toevoegen aan het originele dataframe. Deze functie returnt een kopie en heeft geen effect op het originele dataframe

Drop\_duplicates() gaat duplicate entries verwijderen en zal ook het originele dataframe ongedeerd laten en dus een kopie teruggeven. Met parameter inplace kan je ervoor zorgen dat het originele dataframe wel bewerkt wordt.



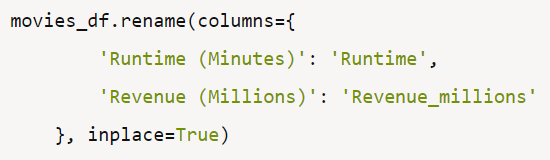
Parameter keep heeft 3 mogelijke opties. First zal alle duplicaten verwijderen behalve de eerste occurence. Last doet hetzelfde maar met de laatste occurence. False zal alle dupliaten verwijderen. De default waarde is first.

## Column cleanup

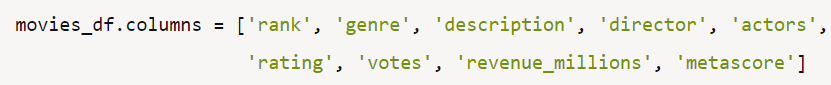
Printen van de kolomnamen wordt gedaan op volgende manier



Deze kolomnamen kunnen verandert worden van naam mbv de rename functie. Als parameter wordt een dict verwacht die de oude kolomnamen mapt op de nieuwe.



Indien alle kolommen moeten aangepast worden kan dit op een efficientere manier: door het columns attribuut gelijk te stellen aan een list met kolomnamen.

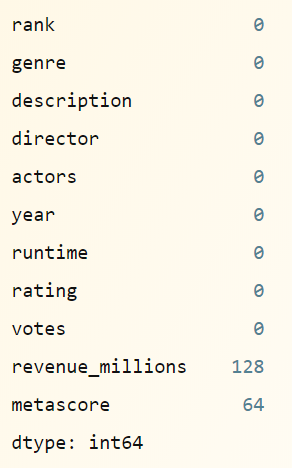




## How to work with missing values

Hiermee kan op 2 manieren om gegaan worden. Ofwel worden hele rijen/kolommen verwijdert met null waarden ofwel worden ze vervangen door non null waarden (dit heet imputation). Met de functie isnull() kan nagegaan worden welke cellen in de dataframe effectief een null waarde bevatten. Om de null waarden te tellen in elke kolom kan een aggregate functie gebruikt worden.





Het wordt aangeraden om enkel null data te verwijderen als het gaat om een kleine hoeveelheid. Dit wordt gedaan met volgende functie. Hiermee zal elke rij met minstens 1 null waarde verwijderd worden. Ook hier zal het originele dataframe niet aangepast worden indien parameter inplace=true niet aanwezig is.



Vewijderen van kolommen wordt gedaan door een parameter toe te voegen. Door axis gelijk te stellen aan 1 worden kolommen met minstens 1 null waarde verwijdert. Axis=1 komt van het feit dat shape een tuple teruggeeft, hierin is het aantal kolommen index 1.

Indien de null waarden vervangen worden, wordt dit meestal gedaan door het gemiddelde of mediaan te gebruiken. Selecteren van een kolom gebeurt door vierkante haakjes te gebruiken, analoog aan de manier waarop dit gebeurt bij keys in een dict. De index komt mee.

Om het gemiddelde van een kolom te bekomen wordt gebruik gemaakt van de mean() functie. Deze waarde wordt dan gebruikt om de null values te vervangen.



Het is meestal beter om niet de mean voor een hele kolom te gebruiken maar specifieker te werken. Bijvoorbeeld door het gemiddelde per genre te nemen als vervangingswaarde.

Met describe() wordt de distrubutie van continue variabelen weergegeven. Deze functie kan ook gebruikt worden om discrete kolomtypes om te zetten. Met value\_counts() wordt voor elke categorie in een discrete kolom het aantal waarden weergegeven.

## Relationships between continuous variables

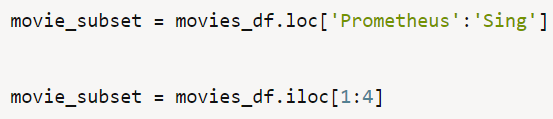
Door het genereren van de correlation matrix kan gezien worden hoe lineair de relatie is tussen verschillende continue variabelen. De tekens van de getallen hierin bepalen een dalende lineaire relatie (ene stijgt, andere daalt) of een stijgende (ene stijgt, andere stijgt ook). Indien de waarde 0 is, is er geen uitsluitsel over de relatie.

## Dataframe slicing, selecting, extracting

De operaties zijn verschillend bij series of dataframe. Met type kan nagegaan worden waarmee we te maken hebben. Het is mogelijk om meerdere kolommen te extracten.



Om rijen te selecteren wordt gebruikgemaakt van loc en iloc. Loc vraagt de index van de rij als parameter. Iloc de numerieke index. Ook hier is het mogelijk om meerdere rijen te selecteren met beide methodes.



Een verschil tussen beide is dat met iloc het laatste element in de slice niet zal geincludeerd worden.

Rijen selecteren op basis van een conditionele selectie wordt gedaan op volgende manier.



Dit heeft als output een series van true en false values. Om enkel de rijen te verkrijgen waarbij de conditie een true waarde heeft moet dit via het datafame gebeuren.



Met |en & kunnen condities gecombineerd worden.



Dit kan vereenvoudigt worden met de isin methode



Met de functie quantile() is het mogelijk om de verschillende kwartielen te verkrijgen.

## Applying functions

een functie kan toegepast worden op elke entry van de dataset. Bijvoorbeeld om een bepaalde waarde te vervangen door een andere. De nieuwe waarde wordt dan gereturned door de functie.

# Android sensor data to csv

<https://github.com/ejoebstl/Android-Sensor-Log/blob/master/app/src/main/java/io/iam360/sensorlog/MainActivity.java>

~~PROBLEEM~~: files worden opgeslagen op de emulator/gsm zelf

+ moet filestream afsluiten anders geen inhoud in file

# Android services

<https://mobologicplus.com/understanding-android-service-intentservice-and-bindservice-of-concurrency-api/>

Er zijn 2 types services. Started service/ unbound service is niet gebonden aan een activity. De service wordt alleen gekilld als er bv niet genoeg geheugen is om de taak uit te voeren Bound service

PROBLEEM: intentservice zal zichzelf killen als het niks meer te doen heeft, dus de sensorchanged events komen niet toe 🡪 gewone service gebruiken , maar runt op de main thread = ok?

# Polar m600 vs polar h7

# Verschillende machine learning algoritmes

# Activity recognition

<http://rstudio-pubs-static.s3.amazonaws.com/165795_92b97c49b5a74d04940670469a9a40f2.html>

<https://machinelearningmastery.com/how-to-model-human-activity-from-smartphone-data/>

keras api (neural network): <https://www.kaggle.com/machinoai/human-activity-recognition-using-lstm>

<https://towardsdatascience.com/human-activity-recognition-har-tutorial-with-keras-and-core-ml-part-1-8c05e365dfa0>

machine learning algos: <https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss>

# android permissions

<https://developer.android.com/training/permissions/requesting#perm-check>

# android bluetooth debugging

C:\Users\Elise\AppData\Local\Android\sdk\platform-tools

adb forward tcp:4444 localabstract:/adb-hub

adb connect 127.0.0.1:4444

# sensorevent timestamp + sampling frequency

= uptime in nanoseconden (time since start operating system)

Soms nanoseconden since epoch (nexus)??

SensorManager.SENSOR\_DELAY\_NORMAL (delay of 200000 microseceonds) 🡪 T = 0,2 sec => 5 HZ

Accelerometer, SENSOR\_DELAY\_FASTEST: 18-20 ms

Accelerometer, SENSOR\_DELAY\_GAME: 37-39 ms

Accelerometer, SENSOR\_DELAY\_UI: 85-87 ms

Accelerometer, SENSOR\_DELAY\_NORMAL: 215-230 ms

# Android in app vs backend

BACKEND: Activity recognition, point calculation, recommendations (netwerk verkeer)

FRONTEND: point calculation, recommendations (processor)

# Android datapoint (score berekening)

tijd tussen start en eind punt van een datapunt is niet groot genoeg om in milliseconden uit te drukken en is dus in feite verwaarloosbaar voor de berekening van een score. De tijd tussen het startpunt van vorig datapunt en eindpunt van huidig wordt gebruikt.