2016•2017
FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN
master in de industriële wetenschappen: elektronica-ICT

Masterproef

Toepassen van machine learning algoritmes op embedded systemen

Promotor :
Prof. dr. ir. Ronald THOELEN

Copromotor :
De heer Marijn LEMMENS

Promotor :
ing. VINCENT CLAES

Nick Goyens

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

Gezamenlijke opleiding Universiteit Hasselt en KU Leuven

2016•2017

Faculteit Industriële ingenieurswetenschappen

master in de industriële wetenschappen: elektronica-ICT

Masterproef

Toepassen van machine learning algoritmes op embedded systemen

Promotor :
Prof. dr. ir. Ronald THOELEN

Copromotor :
De heer Marijn LEMMENS

Promotor :
ing. VINCENT CLAES

Nick Goyens

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: elektronica-ICT

universiteit hasselt | KU LEUVEN

Woord vooraf

Deze thesis is het verslag mijn stageperiode als masterstudent industrieel ingenieur elektronica-ICT, een gemeenschappelijke opleiding aan de Universiteit Hasselt en de Katholieke Universiteit Leuven. Tijdens deze stageperiode heb ik de kans gehad een volledig academiejaar onderzoek te doen bij de Biomedical Device Engineering groep aan het IMO-IMOMEC onderzoekscentrum van Universiteit Hasselt. Deze masterthesis werd uitgewerkt in samenwerking met het "SmaCos" onderzoeksproject binnen Smart-ICT van de Hogeschool PXL.

Ik wil graag mijn interne promotor Prof dr. ir. Ronald Thoelen en mijn externe promotor ing. Vincent Claes van de Hogeschool PXL bedanken voor de hulp en steun die ik heb gekregen bij het uitwerken van deze thesis. Ik wil ook mijn begeleider drs. ing. Marijn Lemmens en bij uitbreiding de hele onderzoeksgroep bedanken om elke dag klaar te staan met hun hulp en advies. Ten slotte wil ik mijn medestudenten in het onderzoekscentrum bedanken om altijd klaar te staan met hun technische kennis.

Het uitwerken van deze masterthesis is een enorm leerrijke ervaring geweest en heeft sterk bijgedragen aan mijn ontwikkeling tot industrieel ingenieur. Ik hoop dat het ook voor u, als lezer een bron van inspiratie en kennis mag zijn.

Inhoudsopgave

Lijst van tabellen

Lijst van figuren

Abstract

De Biomedical Device Engineering groep aan het IMO-IMOMEC doet onderzoek naar de ontwikkeling van sensoren binnen de gezondheidszorg. Deze onderzoeksgroep zoekt naar een oplossing om het genezingsproces van wonden beter op te volgen met behulp van biosensoren. Deze masterproef onderzoekt de ontwikkeling van machine learning algoritmes voor de verwerking van de sensordata. Deze machine learning algoritmes worden specifiek toegepast op embedded systemen om verschillende soorten data in real-time met hoge nauwkeurigheid te verwerken.

Om de mogelijkheden van machine learning op embedded systemen te vergelijken worden er modellen getraind op basis van verschillende soorten datasets. Deze datasets omvatten onder andere impedantiemetingen op voeding en vloeistoffen. De data wordt verwerkt met classificatie en regressie algoritmes op twee verschillende embedded systemen, namelijk de Raspberry Pi 2 en de Intel Curie module. De soorten machine learning modellen die hierop worden toegepast zijn k-nearest neighbors, artificiële neurale netwerken en support vector regressie.

De nauwkeurigheid van de machine learning modellen varieert tussen 90 en 100%, afhankelijk van het gebruikte algoritme en de toegepaste dataset. De snelheid van de trainingsfase varieert tussen 0,74 en 107 seconden. Ten slotte wordt er op basis van de resultaten, betreffende nauwkeurigheid, snelheid en eenvoud van implementatie, een aanbeveling gemaakt afhankelijk van de toepassing.

Abstract in English

The biomedical device engineering group at IMO-IMOMEC conducts research into the development of sensors in the healthcare industry. This research group is searching for a solution to monitor the healing of wounds using biosensors. This master's thesis researches the development of machine learning algorithms for processing the sensor data. These machine learning algorithms are specifically applied to embedded systems to handle different types of data in real time with high accuracy.

To compare the possibilities of machine learning running on embedded systems, models are trained on different types of data sets. These data sets include impedance measurements on food and liquids. The data is processed with classification and regression algorithms on two different embedded systems, namely the Raspberry Pi 2 and the Intel Curie module. The types of machine learning models that are applied to this are k-nearest neighbors, artificial neural networks and support vector regression.

The accuracy of the machine learning models varies between 90 and 100%, depending on the used algorithm and the applied data set. The processing time of the training phase varies between 0.74 and 107 seconds. Finally, based on the results regarding accuracy, speed and simplicity of implementation, a recommendation is made depending on the application.

1 Inleiding

1.1 Situering

Het IMO-IMOMEC is een onderzoekscentrum van de Universiteit Hasselt gelegen op de universitaire campus van Diepenbeek. De belangrijkste werkzaamheden van het IMO-IMOMEC zijn het ontwikkelen en karakteriseren van nieuwe materialen en systemen om te gebruiken in micro-elektronica, bio-elektronica en nanotechnologie.

Dit project situeert zich in de *biomedical device engineering* groep van het onderzoekscentrum. Deze onderzoeksgroep focust zich vooral op het ontwikkelen van sensoren en prototypes voor de gezondheidszorg. Er wordt nauw samengewerkt met ziekenhuizen en zorgcentra om onderzoek te verrichten en prototypes te ontwikkelen.

FIGUUR 1: IMO-IMOMEC [1]

1.2 Probleemstelling

Voor de opvolging van het genezingsproces van een wonde zijn vandaag de dag weinig goede oplossingen beschikbaar. Op dit moment wordt het genezingsproces vooral subjectief en weinig nauwkeurig vastgesteld, namelijk door een visuele inspectie. Dit is een manuele en arbeidsintensieve methode omdat bijvoorbeeld verbanden of pleisters moeten afgenomen en opnieuw aangebracht worden om de opvolging uit te voeren. Hierdoor wordt mogelijk ook de optimale genezingsomgeving van de wonde verstoord.

Aangezien chronische en langdurige wonden vaak voorkomen heeft een slecht genezingsproces een grote impact, zowel op het welzijn van de patiënt als op economisch gebied. Een eerdere studie toont aan dat de kosten van wondverzorging in Europa kan geschat worden tussen 12,2 en 14,9 miljard euro op jaarbasis [2]. Belangrijke parameters bestaan onder andere uit de vochtigheid, oppervlakte en diepte van de wonde. Deze zijn cruciaal om een goed beeld te krijgen van het genezingsproces.

Binnen de onderzoeksgroep wordt er gewerkt aan een methode om door bio-impedantiemetingen dit genezingsproces beter op te volgen. Deze metingen zijn vaak omvangrijk en worden achteraf verwerkt en geclassificeerd op een ander systeem. De metingen kunnen op dit moment dus niet in real time verwerkt worden. Voor de verwerking van deze data wordt in deze masterproef een oplossing gezocht.

1.3 Doelstelling

Om de eerder vermelde parameters van de wonde beter te kunnen opvolgen kan er gebruik worden gemaakt van een bio-impedantie sensor. Bio-impedantie analyse is een veelgebruikte techniek om lichaamsparameters te meten in de medische sector. Dit soort metingen geven inzicht op de samenstelling van de cellen in de huid en kunnen dus ook gebruikt worden om het genezingsproces op te volgen [3].

De verkregen data van deze sensor geeft niet rechtstreeks een duidelijk resultaat over het genezen van de wonde. Door deze data te classificeren met een *machine learning* model wordt er echter wel een duidelijk beeld van de progressie gevormd. Om dit te doen wordt er gewerkt met een model dat constant wordt geüpdatet door nieuwe data.

Machine learning modellen worden traditioneel toegepast als post processing op computers met veel rekenkracht. Vooral het trainen van zulk model is vaak zeer rekenintensief. Met een machine learning model dat wordt toegepast op een *embedded systeem* kan de data effectiever geclassificeerd worden en meteen een resultaat geven. De verwachting is dat het embedded systeem een eenvoudigere, goedkopere en minder tijdrovende oplossing kan zijn. De belangrijkste doelstelling is om te onderzoeken als op zulk embedded systeem goede resultaten behaald worden om data te classificeren.

Om deze hoofddoelstelling te bereiken wordt er in verschillende fases gewerkt. Zo worden er eerst metingen gedaan op bijvoorbeeld vloeistoffen, voeding en weefsels om daarna pas over te gaan naar de hoofddoelstelling van de wonde opvolging. Op deze manier kan er gaandeweg geëxperimenteerd worden wat de mogelijkheden zijn van een machine learning model geïmplementeerd op deze embedded systemen.

Om samen te vatten gaat het hier dus om een verkennende studie waarbij de verschillende aspecten in fases onderzocht worden. Voor elk deel van het project wordt gekeken wat de mogelijkheden zijn, welke haalbaar zijn en wat de beste oplossing uiteindelijk is. De verschillende delen hierin zijn de sensor, de microcontroller en de verwerking van de data met behulp van machine learning.

1.4 Onderzoeksmethode

In de beginfase van het onderzoek wordt onderzocht welke systemen en methodes kunnen gebruikt worden om de doelstellingen te bereiken. Eerdere literatuur wordt onderzocht om na te gaan welke toepassingen er eerder zijn ontwikkeld rond machine learning en embedded systemen.

Vervolgens zal er een selectie worden gemaakt van een of meerdere embedded systemen om algoritmes op te testen. Bij de selectie wordt er ook gekeken naar de beschikbaarheid van *software libraries* die ondersteuning bieden voor het implementeren van machine learning modellen.

Om deze combinatie van hardware en software te testen is er veel data nodig. Zoals eerder vermeld bij de doelstellingen zullen de datasets opgebouwd worden in verschillende fases. Om te beginnen worden er datasets gebruikt van eerdere metingen op het IMO of datasets die vrij beschikbaar zijn op het internet. Vervolgens worden er eigen metingen gedaan op bijvoorbeeld voeding en vloeistoffen.

Op basis van de data die verkregen wordt uit de verschillende metingen zullen de eerdergenoemde platformen getest. Hierna zal een vergelijking worden gemaakt op basis van snelheid, implementatie en nauwkeurigheid van de systemen. Afhankelijk van de toepassing zal er een aanbeveling worden gemaakt over welke oplossing het meest geschikt is.

2 Literatuurstudie

Om deze masterproef beter te situeren en de verschillende onderdelen te onderzoeken, wordt in dit hoofdstuk de literatuurstudie besproken. Eerst wordt er een overzicht gegeven van verschillende machine learning algoritmes en worden de meest relevante voor dit onderzoek in detail besproken. Vervolgens wordt er literatuur onderzocht van verschillende toepassingen die met dit onderzoek te maken hebben. Namelijk metingen rond bio-impedantie, classificatie van impedantiemetingen en machine learning op embedded systemen.

2.1 Machine learning algoritmes

Het doel van machine learning is om een computer de mogelijkheid te geven een probleem op te lossen op basis van eerder verkregen data. Er wordt hierbij gebruik gemaakt van een model waarbij de classificatie van data continue verbeterd door te leren uit de oudere data. Bij genoeg training van het model zal er een nauwkeurige voorspelling kunnen gemaakt worden voor de classificatie van elk nieuw data punt.

2.1.1 Overzicht

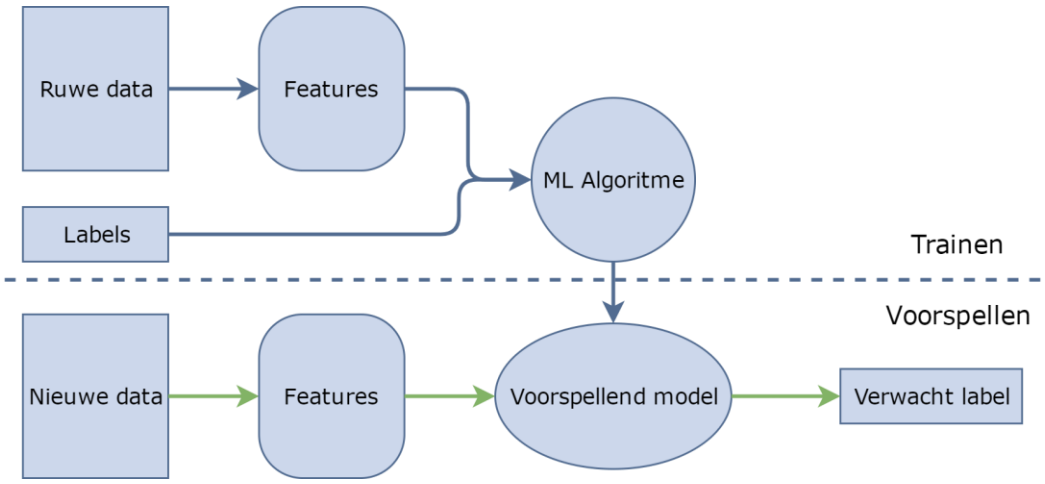
Er is een groot aanbod aan machine learning algoritmes voor allerlei complexe verwerking van data [4]. Het belangrijkste onderscheid in machine learning kan gemaakt worden op basis van de gebruikte dataset en hun toepassing. Er wordt onderscheid gemaakt tussen twee grote categorieën:

- *supervised learning*, er wordt gebruikt gemaakt van data waarvoor al categorieën of labels worden toegewezen;
- *unsupervised learning*, er zijn nog geen categorieën aan de data toegewezen, er wordt vooral naar patronen gezocht in de data of de data wordt gegroepeerd in clusters om zo een verband te zoeken.

Verder kan er ook een samenstelling van beide optreden, we spreken dan over *semi-supervised learning*.

In dit project wordt er enkel gewerkt met algoritmes in de categorie van supervised learning, omdat de data die gebruikt wordt om een model te trainen op voorhand al een categorie heeft. Dit soort algoritmes worden vooral gebruikt in voorspellende modelering. Een voorspellend model is een model dat getraind wordt met input data en bijbehorend output data.

Het model probeert dan een verband te leggen tussen de waarden, zodat de output waarden van nieuwe data voorspeld wordt. Figuur 2 toont een belangrijke flowchart van de basiswerkwijze van supervised learning.



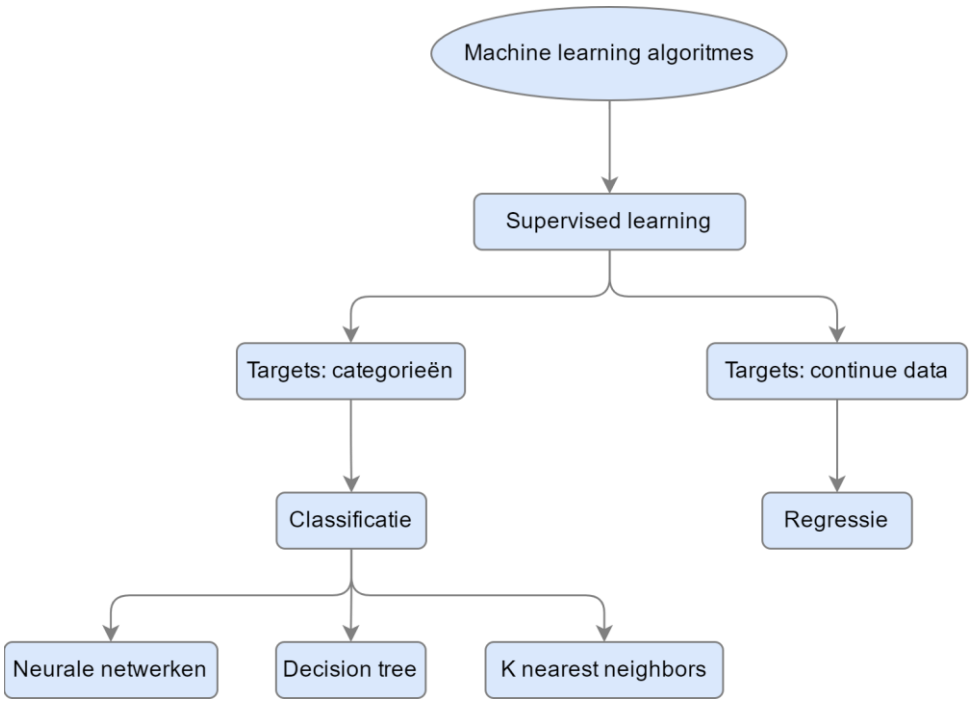
FIGUUR 2: FLOWCHART VAN DE WERKWIJZE VAN SUPERVISED LEARNING

In deze categorie wordt er ook een verder onderscheid gemaakt tussen twee grote soorten, namelijk classificatie en regressie.

Een classificatie algoritme maakt een voorspellend model van de input en de bijhorende klasse labels (output). Vervolgens wordt het model gebruikt om klasse labels te voorspellen op basis van nieuwe, onbekende data. In dit geval zijn de outputwaarden dus discreet(niet-continue). Belangrijke voorbeelden hiervan zijn *decision trees, K-nearest neighbors* en artificiële neurale netwerken.

Een regressie algoritme maakt ook een voorspellend model op basis van de input data maar werkt niet met categorieën. De output waarden die worden gebruikt voor de voorspelling zijn in dit geval continue en dus niet discreet. Er kan dus een unieke waarde worden voorspeld voor elke input waarde. Belangrijke voorbeelden hiervan zijn lineaire regressie en logaritmische regressie.

Zowel classificatie en regressie worden toegepast afhankelijk van het voorgelegde probleem. Figuur 3 geeft een overzicht van de meest relevante algoritmes voor dit onderzoek binnen de supervised learning categorie.



FIGUUR 3: OVERZICHT VAN RELEVANTE MACHINE LEARNING ALGORITMES VOOR DIT ONDERZOEK

2.1.2 K-nearest neighbors

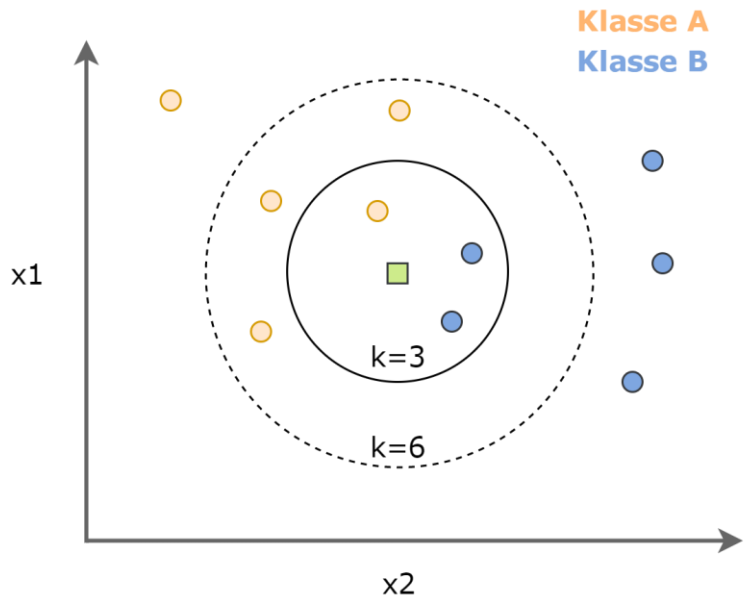
K-nearest neighbors (k-NN) is een van de meest gebruikte algoritmes. K-NN kan eigenlijk zowel voor classificatie als voor regressie problemen gebruikt worden, maar wordt in de meeste gevallen toegepast als classificatie algoritme. Het wordt in de literatuur vooral omschreven als een simpel, maar toch krachtig classificatie algoritme. K-NN wordt door zijn robuustheid en veelzijdigheid ook vaak gebruikt als een benchmark voor meer complexe classificatie algoritmes als artificiële neurale netwerken (ANN) en support vector machines (SVM). Het algoritme is ook niet parametrisch. Dit wilt zeggen dat er geen aanname wordt gemaakt van een zekere wiskundige verhouding tussen de data. Dit is vaak nuttig omdat er in realistische scenario's meestal niet met data wordt gewerkt die theoretische wiskundige modellen volgen zoals gaussische of lineaire functies.

Verder is het een 'lui' algoritme. Dit wil zeggen dat er geen expliciete trainingsfase plaats vind waarin een model wordt getraind door een trainingsdataset vele malen door een model aan te leren. De trainingsdata wordt wel opgeslagen en gebruikt als een soort geheugen om later een voorspelling op te maken. Een groot voordeel hiervan is dat de trainingsfase zeer snel is afgehandeld en er meteen kan overgegaan worden naar het testen van de data. Dit is een groot voordeel voor de toepassingen gebruikt voor dit project, waarbij beperkte rekenkracht beschikbaar is op de embedded systemen. Verder is de korte traingingstijd ook zeer voordelig om de data meteen in real time te verwerken. De minimale trainingsfase heeft echter het nadeel dat er een groter geheugen nodig is om de data van de trainingsdata op te slaan, vooral als er

met een grote dataset gewerkt wordt. Een ander nadeel is dat er in de test fase iets meer rekenkracht nodig is omdat de hele dataset moet afgelopen worden. We kunnen k-NN dus samenvatten als minimale training maar uitgebreide testing.

K-NN gaat uit van data als een vector, deze data wordt uitgezet in de ruimte. Hierdoor is er dus een bepaalde afstand tussen de data punten. De k is gelijk aan een getal dat bepaalt hoeveel buren er worden toegewezen aan elk nieuw data punt in deze ruimte. Indien k=1 wordt er simpelweg gekeken naar de enkele dichtstbijzijnde buur [5],[6].

Figuur 4 geeft een visuele weergave van het werkingsprincipe van k-NN. Het groene vierkant stelt hierbij een nieuw datapunt voor, en de cirkels stellen eerder geclassificeerde datapunten voor. Het toont aan dat wanneer k gelijk wordt gesteld aan 3, het nieuwe data punt zal geclassificeerd worden als klasse B. Echter wanneer k gelijk is aan 6 zullen er meet datapunten van klasse A zich binnen de cirkel bevinden en zal het nieuwe datapunt dus tot klasse A behoren. Dit toont aan dat de bepaling van waarde k zeer belangrijk is voor een correcte classificatie van het algoritme.

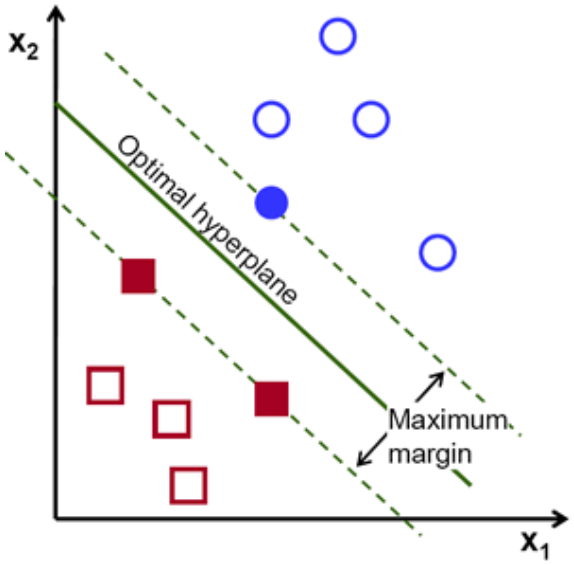


FIGUUR 4: GRAFISCHE VOORSTELLING VAN CLASSIFICATIE MET 2 VERSCHILLENDE K WAARDEN

Dit voorbeeld geeft aan hoe k-NN kan gebruikt worden als classificatie algoritme. Echter werd eerder ook vermeld dat k-NN ook kan gebruikt worden bij regressie problemen. Het enige verschil met de besproken methode is dat er een gemiddelde wordt genomen van de nearest neighbors in plaats van deze te tellen en een absolute waarde te nemen.

2.1.3 Support vector machine

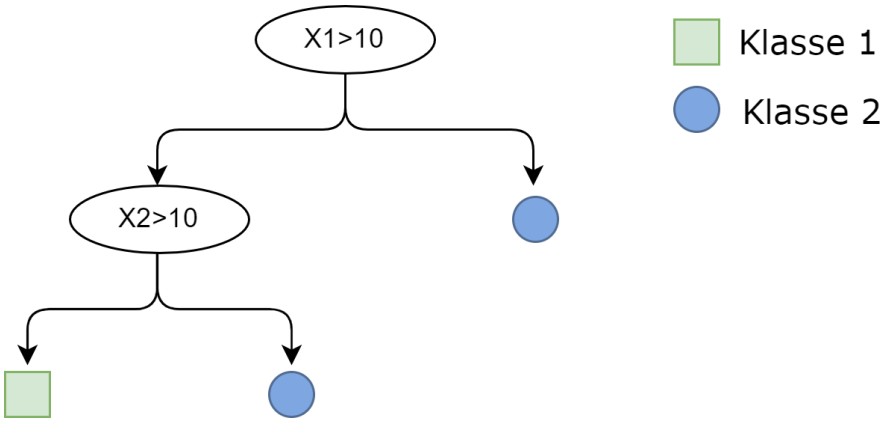
Support vector machine (SVM) is een ander veelgebruikt supervised learning model. SVM wordt vooral gebruikt voor classificatie en is gebaseerd rond het concept van lijnen te trekken in de ruimte om data op te delen in categorieën. Een SVM-model is een presentatie van de verschillende input waarden in de ruimte. Deze presentatie wordt zo opgesteld dat de verschillende categorieën worden gescheiden door een zo groot mogelijke kloof. In deze kloof wordt een lijn getrokken die de datapunten van elkaar scheidt. De lijn is optimaal als hij zo ver mogelijk passeert van alle datapunten. In Figuur 5 is een ideale voorstelling te zien waar de lijn (*optimal hyperplane*) zo ver mogelijk verwijderd is van alle datapunten [7].



FIGUUR 5: VOORSTELLING VAN SVM MET EEN OPTIMALE SCHEIDING [7]

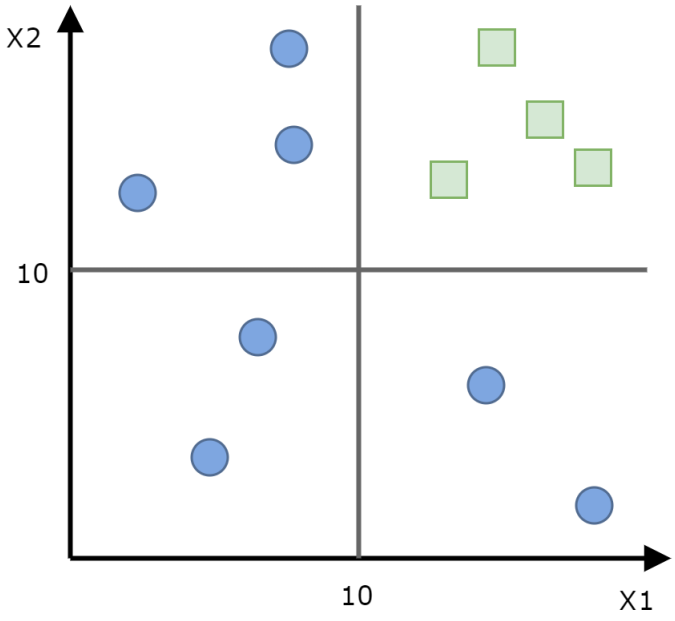
2.1.4 Decision tree

Een *decision tree* (DT) is ook een supervised learning algoritme en wordt vooral gebruikt voor classificatie, maar kan ook voor regressie gebruikt worden. Het doel is om een model te maken dat de waarde van de target variabele voorspelt door simpele keuzes te maken op basis van de inputdata [8]. Figuur 6 toont aan hoe een heel eenvoudige decision tree in zijn werk gaat. Voor elke nieuw datapunt worden de parameters getest als ze groter zijn dan 10. Op basis van deze informatie worden nieuwe datapunten geclassificeerd.



FIGUUR 6: GRAFISCHE VOORSTELLING VAN DE CLASSIFICATIE IN EEN DECISION TREE

In Figuur 7 is te zien in een plot hoe de datapunten geclassificeerd zijn op basis van bovenstaand model.



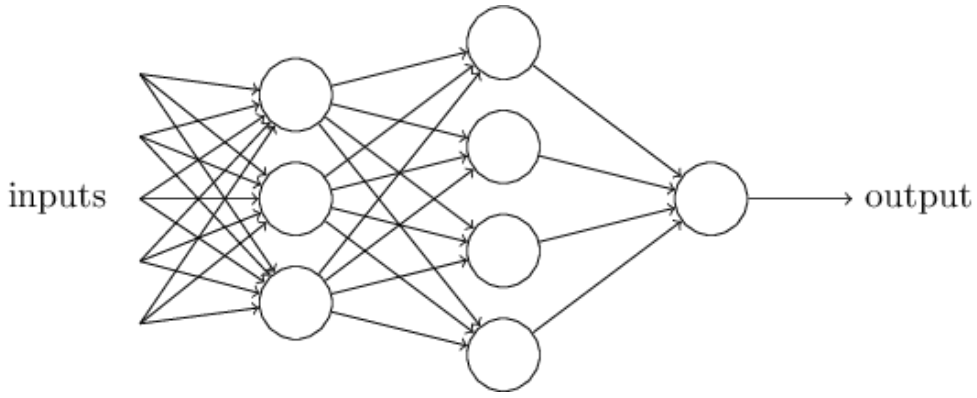
FIGUUR 7: CLASSIFICATIE VEN EEN DECISION TREE VOORGESTELD ALS 2D RUIMTE

Het grote voordeel van DT is dat ze simpel zijn om te begrijpen omdat de resultaten visueel kunnen voorgesteld worden, zoals hierboven te zien is. Maar decision trees kunnen zeker niet voor alle problemen toegepast worden en bij veel datafeatures bestaat het gevaar van overfitting.

2.1.5 Neurale netwerken

Een neuraal netwerk of artificieel neuraal netwerk (ANN) is een manier om data te verwerken geïnspireerd op biologische systemen, zoals het menselijk brein. Een neuraal netwerk is opgebouwd uit een hoog aantal elementen die met elkaar verbonden zijn, genaamd neuronen. Deze neuronen werken met elkaar samen om specifieke problemen op te lossen en leren bij uit de voorbeelddata die ze krijgen. Elk neuraal netwerk kan ontworpen worden om een specifiek probleem op te lossen, zoals patroonherkenning of data classificatie. Belangrijk bij deze taken is dat de connecties tussen de neuronen of gewichten bij het trainen worden aangepast.

In Figuur 8 is te zien hoe een neuraal netwerk typisch wordt voorgesteld, met verschillende inputwaarden die naar de neuronen worden gestuurd en een output als resultaat.



FIGUUR 8: BASISVOORSTELLING VAN DE VERSCHILLENDE LAGEN IN EEN NEURAAL NETWERK [9]

Dit soort neurale netwerken gaan ook geïmplementeerd worden in het vervolg van dit project. De voornaamste reden hiervoor is dat een neuraal netwerk de belangrijke eigenschap heeft een resultaat af te leiden uit grote blokken complexe data, en hieruit zelf kan leren. Er kunnen patronen herkend worden die zeer moeilijk te berekenen zijn met conventionele programmeertechnieken.

Een neuron is een element met verschillende inputwaarden en één output. Er wordt gewerkt in twee basis modes, namelijk training mode en test mode. In training mode worden neuronen aangeleerd om een output te geven op een bepaald input patroon. In de test mode wordt het input patroon gedetecteerd en wordt de bijhorende output waarde uitgestuurd. De waarde van neurale netwerken wordt echt duidelijk als er een input patroon wordt ingelezen dat nog niet gekend is. In dit geval zal de dichts bijhorende neuron uitgestuurd worden.

Feed forward neurale netwerken is een type netwerk waarbij de data enkel van input naar output kan gestuurd worden zonder feedback. Dit soort netwerken worden het vaakst gebruikt bij patroonherkenning en zullen ook in het vervolg van dit project vooral aan bod komen [10].

Radial basis function netwerk

Een *radial basis function* netwerk (RBF) is een type van neuraal netwerk waarbij de output gelijk is aan een lineaire combinatie van de parameters van de neuronen en een radial basis function. RBF wordt ook vooral gebruikt als classificatie netwerk. Een RBF-netwerk kan gezien worden

als een normaal neuraal netwerk met RBF-neuronen als de verborgen lagen. Elke RBF-neuron slaat een van de vectoren van de input data op als prototype vector. Vervolgens vergelijkt elke RBF-neuron de nieuwe inputwaarde met zijn prototype en geeft een waarde terug tussen 0 en 1 die overeenstemt met de gelijkheid tussen de twee vectoren. Deze output in functie van de afstand tussen de vectoren heeft de form van een klokkromme. In de output laag bevinden zich een aantal neuronen (1 voor elke categorie) die op basis van de waarde van de RBF-neuronen de input waarde classificeert [11].

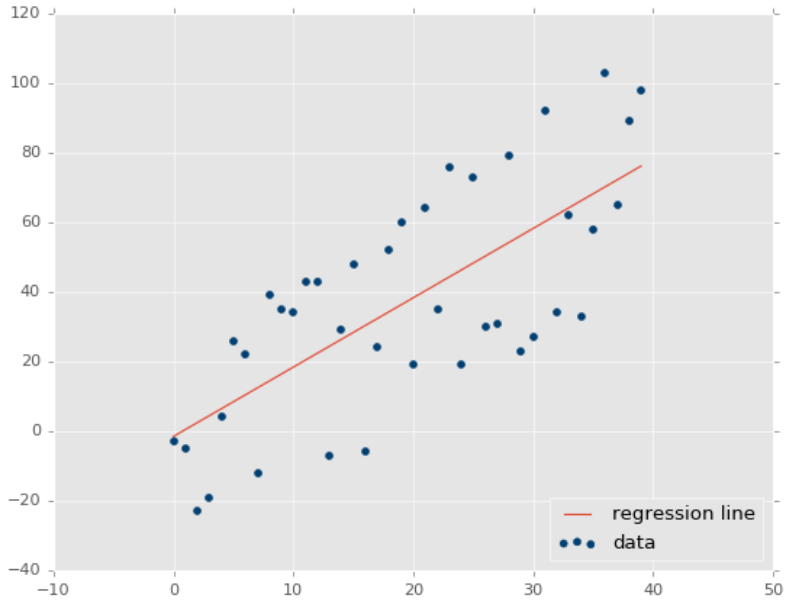
2.1.6 Regressieanalyse

Regressieanalyse is de tegenhanger van classificatie algoritmes en wordt gebruikt op continue data. Bij regressie is het doel om een verband te vinden tussen output- en inputvariabele. Er wordt een vergelijking gezocht waarmee zo goed mogelijk een nieuwe waarde kan voorspeld worden. Regressie is ook een vorm van supervised machine learning, omdat het model wordt getraind met output waarden die al gekend zijn. Er zijn verschillende soorten regressie technieken om voorspellingen te maken. De keuze van de techniek is afhankelijk van de vorm van de regressielijn.

Lineaire regressie

Bij lineaire regressie wordt er een simpel lineaire verband gezocht. De regressielijn is dus een rechte lijn die het nauwst aansluit bij alle variabelen. Een voorbeeld hiervan is te zien in Figuur 9. De lijn wordt voorgesteld door vergelijking:

$$y = a + b * x + e$$



FIGUUR 9: EENVOUDIG VOORBEELD VAN LINEAIRE REGRESSIE [12]

De lijn wordt gezocht door de kleinstekwadratenmethode. Hierbij wordt de kleinst mogelijke verticale afstand gezocht tussen de datapunten en de lijn. Het kwadraat van de afstanden wordt genomen om de negatieve waarden weg te werken.

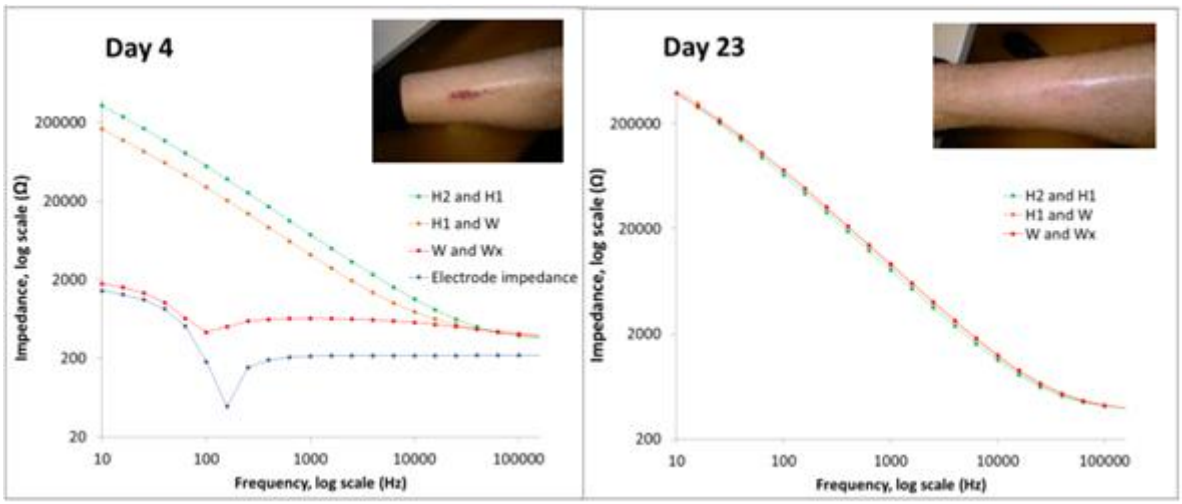
Vaak is een simpele lineaire regressie echter niet voldoende om een probleem op te lossen, omdat er niet zomaar een lineaire verband bestaat tussen de variabelen. Daarom zijn er nog vele andere technieken beschikbaar zoals polynomiale regressie en logistische regressie. Maar ook met veel classificatie algoritmes kan regressie worden toegepast, zoals SVM, k-NN of ANN.

2.2 Toepassingen

2.2.1 Bio-impedantiemeting

Zoals eerder vermeld wordt bio-impedantie vaak gebruikt als parameter bij medische toepassingen. Een onderzoek dat een overzicht maakt van deze techniek geeft aan dat impedantiemetingen gebruikt kunnen worden om schade in een celstructuur te meten [13]. Bio-impedantie wordt gedefinieerd als de mogelijkheid van cellen om een elektrische wisselstroom tegen te werken. De impedantie van biologische vezels is sterk afhankelijk van de gebruikte frequentie en bevat zowel capacitieve als resisitieve componenten. Vandaar dat er gesproken wordt over impedantie en niet over weerstand[14].

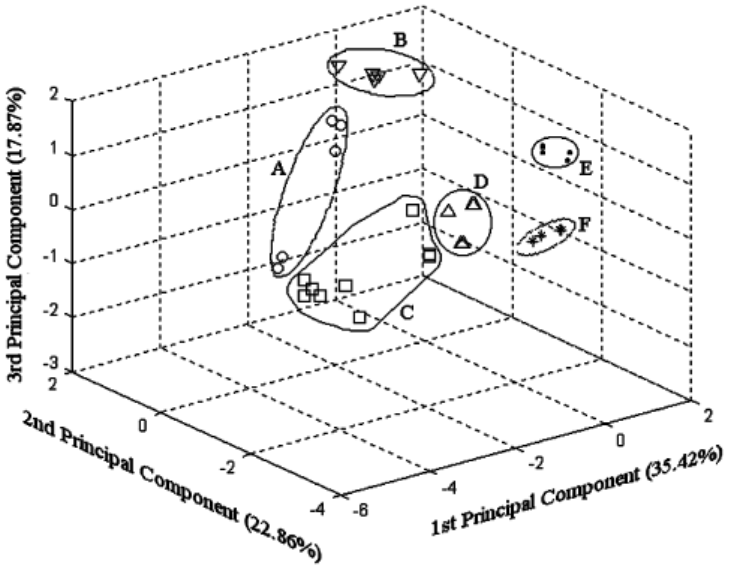
Uit eerder onderzoek naar bio-impedantie bij de genezing van een wonde zijn reeds veelbelovende resultaten gekomen. Uit een studie waarbij de impedantie van de huid werd gemeten met behulp van twee elektrodes kan een duidelijke evolutie worden vastgesteld in impedantie tussen enkele dagen na de wonde en bij complete genezing. In Figuur 10 is te zien hoe de impedantie waarden die rechtsreeks op de wonde gemeten worden op dag 4 nog ver afwijken van de referentiewaarden. Bij volledige genezing is echter te zien dat de waarden terug naar de referentie terugkeren [15]. Een andere studie toont aan dat het ook mogelijk is om verder analyse uit te voeren, zoals de graad van infectie van de wonde en de hoeveelheid cellen die al hersteld zijn [16]. Ander onderzoek toont ook de mogelijkheid aan om met impedantie spectroscopie borstkanker te detecteren. De nauwkeurigheid van de tumor classificatie op basis van deze data is 92% [17].



Figuur 10: Evolutie van impedantie bij een genezing tussen dag 4 en dag 23 [15]

2.2.2 Classificatie van impedantiedata

Er is al eerder onderzoek gedaan naar het toepassen van machine learning algoritmes op impedantiemetingen. Gebruik makend van impedantie spectroscopie kunnen bijvoorbeeld verschillende soorten wijn geclassificeerd worden. In deze studie wordt gebruik gemaakt van een impedantie elektrode gebaseerd op nanomateriaal. In Figuur 11 is een 3D spreiding te zien van de data van 6 verschillende soorten wijnen. Hier kunnen verschillende clusters uit afgeleid worden. Deze data wordt gebruikt voor classificatie in een artificieel neuraal netwerk. Met dit neuraal netwerk is een nauwkeurigheid van 100% behaald. Ook beginnende experimenten gedaan op verschillende soorten koffie en water beloven een hoge nauwkeurigheidsgraad te halen. De studie geeft aan dat de methode ook voor andere toepassingen kan gebruikt worden waarbij classificatie van vloeistoffen nodig is [18].



FIGUUR 11: 3D SPREIDING VAN 6 VERSCHILLENDE SOORTEN WIJN TER CLASSIFICATIE [18]

In ander onderzoek is er een methode ontwikkeld voor automatische herkenning van concentraties van verschillende gassen, gebruikmakend van impedantiemetingen. De classificatie van de gassen wordt getest met verschillende soorten machine learning algoritmes. Van de geteste classificatiemethodes haalt k-NN een goed resultaat bij een supervised dataset [19].

In de biomedische sector is er een ANN gebruikt om een schatting te maken van een spierziekte. Het impedantie spectrum wordt hier gemeten op 27 frequenties tussen 100 Hz en 1 MHz. Het ANN model haalt hierbij een nauwkeurigheid van 94.5% [20].

2.2.3 Machine learning modellen op embedded systemen

Voor toepassingen waarin data van sensoren in real time moet geclassificeerd worden, is het logisch om de data rechtsreeks te verwerken op een microprocessor. Voor heel wat toepassingen is het interessant om hiervoor machine learning classificatie te gebruiken zoals bijvoorbeeld een neuraal netwerk. Omdat neurale netwerken vaak ontworpen zijn om uitgevoerd te woorden op zware CPU's en GPU's, is het nodig om deze te optimaliseren om goed te werken op de beperkte rekenkracht van een microprocessor. Hieronder wordt een overzicht gegeven van de literatuur over het gebruik van machine learning op embedded systemen.

In eerder onderzoek zijn binaire neurale netwerken (BNN) uitgewerkt om het geheugen en vermogen gebruik te verlagen, alsook de benodigde rekenkracht. Een BNN gebruikt enkel binaire getallen als gewichten om de neurale netwerken te trainen. Het gebruik van enkel binaire getallen zorgt ervoor dat de meeste berekening met bit bewerkingen worden gedaan. Hierdoor zijn de algoritmes zeer efficiënt zonder nauwkeurigheidsverlies. Er worden goede resultaten behaald op verschillende datasets [21]. Het implementeren van deze BNN's levert veelbelovende resultaten op embedded systemen. Bij het gebruik van binaire getallen in plaats van floating-point variabelen is het geheugengebruik 32 keer verminderd. Dit terwijl er nog steeds een nauwkeurigheid van 95% wordt behaald op de MNIST dataset op de Intel Curie [22].

RBF neurale netwerken zijn ook al gebruikt om in real time data te classificeren. In een eerder onderzoek werd een RBF netwerk gebruikt om gezichtsherkenning toe te passen op drie verschillende embedded systemen. Er worden nauwkeurigheidscijfer gehaald van 92% op een *field programmable array* (FPGA), 85% op een *zero instruction set computer* (ZISC) en 98,2% op een *digital signal processor* (DSP). De verwerkingssnelheid voor de drie embedded systemen zijn respectievelijk 14 afbeeldingen/sec, 25 afbeeldingen/sec en 4,8 afbeeldingen/sec [23].

Een andere beeldverwerking toepassing die in real time moet verwerkt worden is de automatische herkenning van nummerplaten. Met een neuraal netwerk dat ontwikkeld is om weinig geheugen te gebruiken, kunnen nummerplaten op een snelle manier verwerkt worden op een kleine FPGA [24]. Ook in ander onderzoek is een neuraal netwerk succesvol geïmplementeerd op een FPGA. Hierbij wordt er aan multiplexing gedaan tussen de lagen van het neuraal netwerk om een effectievere classificatie te bereiken [25].

Een ANN is ook gebruikt om een huishoud rooksensor te ontwikkelen. Deze methode herkent niet enkel rook, maar classificeert ook de aard van het vuur aan de hand van verschillende gassensoren. Dit algoritme is succesvol geïmplementeerd op een kleine microcontroller [26]. Hetzelfde principe is ook gebruikt om een product te ontwikkelen om activiteiten te herkennen[27].

3 Onderzoeksproces

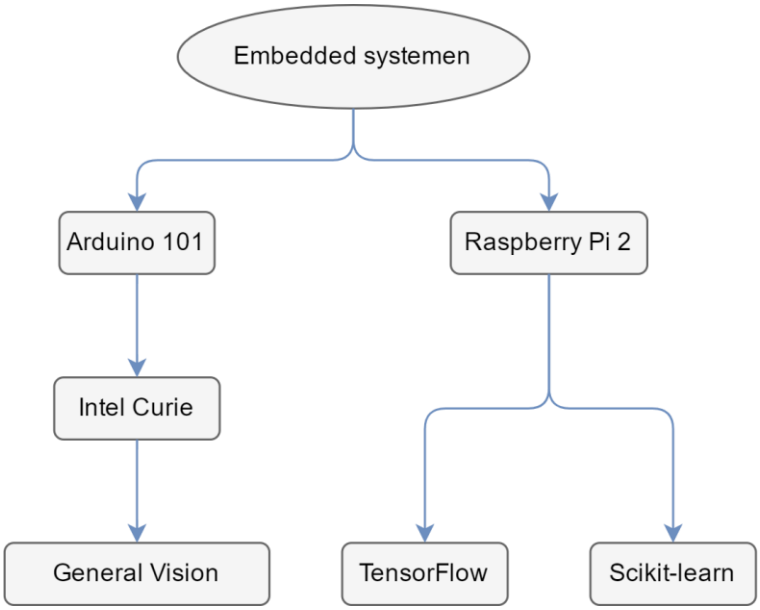
In dit hoofdstuk wordt het onderzoeksproces van deze masterproef omschreven. Eerst wordt er uitleg gegeven over de toegepaste hardware systemen en waarom dat deze gekozen zijn. Vervolgens wordt er toegelicht welke softwaremethodes en libraries er gebruikt worden om machine learning toe te passen. Vervolgens komen de verschillende metingen aan bod. Zoals eerder vermeld worden er verschillende metingen in fases uitgevoerd zodat er veel data kan verkregen worden. Deze data zal gebruikt worden om de verschillende embedded platformen en uitgeschreven algoritmes met elkaar te vergelijken. De belangrijkste parameters die bekeken worden zijn de snelheid en nauwkeurigheid van de platformen. Op basis hiervan zal later een conclusie gemaakt worden.

3.1 Implementatie machine learning algoritmes

Om machine learning algoritmes embedded te kunnen uit voeren zijn er microcontrollers nodig. De twee microcontrollers die hiervoor gekozen worden zijn de Arduino 101 en de Raspberry Pi 2.

De Arduino 101 is een uitbreiding op de wel bekende Arduino Uno. Deze versie is gebaseerd op de Intel Curie module met een ingebouwde Intel Quark system on chip (SoC). Unieke aan deze chip is de ingebouwde pattern matching engine waarmee machine learning algoritmes kunnen worden toegepast. Dit maakt de Intel Curie zeer interessant voor dit project. Om deze pattern matching engine aan te spreken wordt er gebruik gemaakt van de *General Vision* library. Verder is de module ontwikkeld om een zeer laag energieverbruik te hebben en is dus ideaal voor medische toepassingen waarbij er constant data moet worden gemeten. Ten slotte is de Arduino omgeving makkelijk te programmeren en kunnen vele soorten sensoren rechtstreeks worden ingelezen.

De Raspberry Pi 2 is ook een logische keuze voor deze applicatie. Net als Arduino is Raspberry een zeer breed ondersteund en gedocumenteerd platform. Verder is het een krachtig en compact systeem. Een grote troef is ook dat er rechtsreeks in de python programmeertaal kan worden geschreven. Voor deze programmeeromgeving zijn er reeds verschillende machine learning software libraries beschikbaar. Hieruit zijn er twee gebruikt en toegepast, namelijk TensorFlow en scikit-learn. Figuur 12 geeft een overzicht van de gebruikte embedded systemen en hun software libraries.



FIGUUR 12: OVERZICHT EMBEDDED SYSTEMEN EN GEBRUIKTE SOFTWARE LIBRARIES

3.1.1 Intel Curie

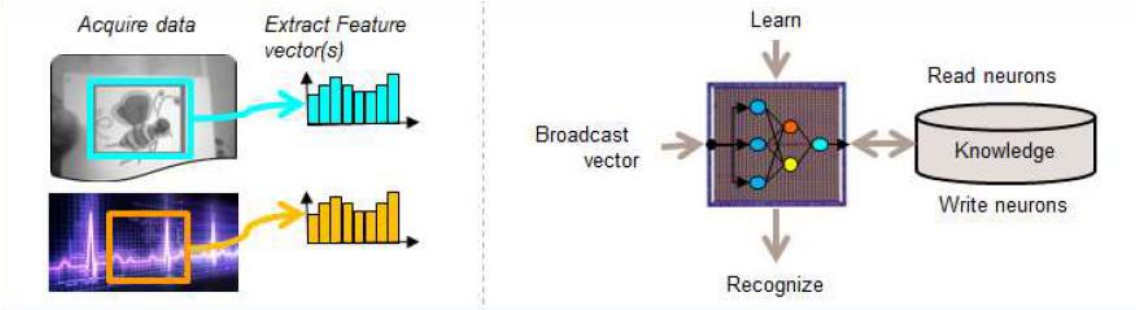
Het eerste platform dat onderzocht en getest is de Arduino 101 met de Intel Curie module aan boord. De belangrijkste reden hiervoor is dat de pattern matching engine chip op deze module unieke hardware neuronen bevat. Deze neuronen worden normaal steeds in software uitgeschreven en Intel is bij de eerste producenten om deze op hardware te implementeren op een toegewijde chip [28]. Deze toegewijde chip heeft als belangrijk voordeel minder processorkracht en vermogen te vereisen dan traditionele software algoritmes. Er kan dus sneller en energiezuiniger data verwerkt op embedded platformen.

De belangrijkste voordelen van de neuronen zijn:

- leren door voorbeelden,
    - weinig code nodig,
- continue dataverwerking op laagvermogen,
- minimale rekenkracht.

Er zijn verschillende methodes beschikbaar om de hardware neuronen van de Intel Curie aan te sturen. Enerzijds is er een library voor Arduino van Intel zelf, deze biedt een basis aansturing en gelimiteerde functies om de chip aan te sturen. Anderzijds is er een library geschreven door het bedrijf General Vision genaamd *CurieNeurons*, ook te gebruiken in de programmeeromgeving

van Arduino. Deze library voorziet een meer uitgebreide aansturing voor de neuronen en een betere documentatie om de code te implementeren. Na testen met beide libraries en vanwege eerder omschreven voordelen is er gekozen om verder te werken met de CurieNeurons library voor de aansturing van de chip. In Figuur 13 is er een schematische voorstelling te zien van de basisfuncties van deze chip.



FIGUUR 13: BASISFUNCTIES VAN DE CURIENEURONS [29].

3.1.2 TensorFlow

TensorFlow is een interface voor het implementeren en uitvoeren van machine learning algoritmes. TensorFlow werd ontwikkeld door Google om hun behoeften tegemoet te komen om neurale netwerken te maken om aan patroonherkenning te doen. Buiten voor onderzoek en ontwikkeling van Google producten is TensorFlow ook opensource en wordt de software gebruikt door een actieve community voor de ontwikkeling van machine learning modellen. TensorFlow wordt vooral, maar niet uitsluitend, gebruikt voor classificatie met neurale netwerken.

TensorFlow is opgebouwd rond zogenaamde 'tensors'. Een tensor bestaat uit een aantal waarden die in een array zijn geplaatst met verschillende dimensies. Deze tensors worden gebruikt als de input en output van de operaties. Rond deze tensors en operaties wordt een neuraal netwerk in TensorFlow opgebouwd. De software biedt verschillende hoge en lage level methodes aan om en model te ontwikkelen. De ontwikkelde modellen zijn ook flexibel en kunnen met weinig aanpassingen voor verschillende datasets worden ingezet. De software is eerder al gebruikt voor een grote aantal toepassingen zoals spraakherkenning, robotica en beeldherkenning [30].

Om volgende redenen wordt TensorFlow gebruikt in dit project:

- heeft een Python interface en is dus te programmeren op de Raspberry Pi;
- uitgebreide community en veel documentatie beschikbaar;
- interessante ondersteuning van neurale netwerken voor classificatie.

3.1.3 Scikit-learn

Scikit-learn (SKlearn) is een andere machine learning library uitgeschreven voor gebruik in python. Bij dit softwarepakket ligt de nadruk op gebruiksgemak, documentatie en prestatie. Het grootste verschil met bijvoorbeeld TensorFlow is de verscheidenheid van machine learning modellen. Waar TensorFlow zich vooral op neurale netwerken focust is er bij SKlearn een zeer groot aanbod van bijna alle classificatie en regressie algoritmes. Verder is SKlearn zeer snel en efficiënt. De meeste modellen worden veel sneller getraind in SKlearn dan met andere libraries [31].

Om deze redenen wordt ook Scikit-learn opgenomen in dit project:

- ook te gebruiken op Raspberry Pi door de python interface;
- groot aanbod aan verschillende modellen voor zowel regressie als classificatie;
- goed evenwicht tussen gebruiksgemak en snelheid van uitvoering.

3.2 Classificatie op iris dataset

Voor dat er aan de slag wordt gegaan met eigen data, worden twee classificatie methodes eerst getest op een dataset die beschikbaar is op het internet. De data die hiervoor gebruikt wordt komt uit de iris dataset. Dit is een dataset die vaak wordt toegepast om classificatie algoritmes te testen en vrij beschikbaar is op het internet [32]. De dataset bestaat uit 150 samples van 3 verschillende soorten bloemen. Voor elk van de datapunten zijn er telkens 4 kenmerken van de bloem opgenomen. De modellen die worden gebruikt om de data te classificeren op de Raspberry Pi worden hieronder kort toegelicht. Hierna worden gelijkaardige modellen toegepast om data uit eigen metingen te classificeren.

3.2.1 K-nearest neighbors op Raspberry Pi

De data is eerst geclassificeerd met een k-NN algoritme met behulp van de SKlearn software. De details over hoe dit model geïmplementeerd is worden later gegeven bij de eigen data. Van de in totaal 150 datapunten worden 30 datapunten genomen om het model later te testen. Met de overige 120 datapunten wordt het model getraind. Bij het testen van het model worden 27 van de 30 datapunten correct geclassificeerd. Dit komt dus neer op een nauwkeurigheid van 90%. Omdat er bij een geen lange trainingsfase bestaat maar de data wordt opgeslagen, is de snelheid zeer hoog en de verwerkingstijd dus verwaarloosbaar.

3.2.2 K-nearest neighbors op Intel Curie

Ook op de Intel Curie is deze dataset geclassificeerd met een k-NN model. Ook de details van dit model worden later verder toegelicht. Dezelfde verdeling wordt gebruikt om de data op te splitsen. In dit model werden 28 van de 30 datapunten correct geclassificeerd. Wat dus neerkomt op een nauwkeurigheid van 93,33%. Ook hier is de trainingstijd verwaarloosbaar.

3.2.3 Neuraal netwerk op Raspberry Pi

Dezelfde dataset is ook getraind in een neuraal netwerk model. Dit keer met behulp van de Tensor Flow library, die speciaal ontwikkeld is voor artificiële neurale netwerken. Opnieuw wordt dit model in detail uitgelegd wanneer het wordt toegepast op eigen data. Hier wordt enkel de uitvoering besproken. De dataset wordt opnieuw gesplitst in 120 trainingspunten en 30 testpunten. Het model wordt getraind in 2000 herhalingen voor een nauwkeurige voorspelling. Uiteindelijk wordt er hier een nauwkeurigheid gehaald van 96,67%. De verlieswaarde bedraagt 0,042. Ook deze wordt later verder toegelicht. De snelheid van de trainingsfase moet hier wel in rekening worden gebracht. Het netwerk wordt getraind aan een snelheid van 121 stappen/seconden uitgevoerd op de Raspberry Pi. Dit wilt zeggen dat de volledige traingingsfase ongeveer 16,5 seconden in beslag neemt. Eens het model voldoende getraind is gebeurt de classificatie wel onmiddellijk [33].

3.3 Classificatie van voeding

Een eerste toepassing die vanuit het onderzoekscentrum is voorgesteld is een automatische detectie van voeding. Het idee achter deze toepassing is om de uitgeschreven algoritmes toe te passen om een bepaald soort voedselproduct te detecteren op basis van data vergaard uit sensoren. Deze proef heeft in de praktijk niet meteen een concrete toepassing, maar is belangrijk om de algoritmes op nauwkeurigheid en snelheid te vergelijken. De volgende data is in deze proef opgenomen:

- weerstand,
- elektrolytmeting met koper en zink,
- impedantie (verschillende frequenties).

Uit de bovenstaande meetmethodes zal de meest efficiënte combinatie gekozen worden afhankelijk van het product dat wordt gemeten. De eerste proef bestaat uit het classificeren van verschillende soorten groeten en fruit. Volgende producten werden gebruikt:

- appels (3 soorten),
- wortels,
- aardappels.

In een volgende proef zijn er metingen uitgevoerd op verschillende concentraties van fruitsap in water om het verschil hiervan te classificeren. Dit is iets complexer omdat de waarden bij verschillende concentraties vermoedelijk dichter bij elkaar gaan liggen dan bij verschillende soorten groenten en fruit. Dit laat toe om de nauwkeurigheid van de neurale netwerken nog beter te onderzoeken. Deze proef is ook belangrijk omdat in een volgende fase van het onderzoek de doorspoeling van leidingen zal gemeten worden. Deze doorspoeling zal gebeuren van fruitsap naar water en ook hier zullen deze concentraties aan bod komen.

De laatste impedantiemeting is uitgevoerd op verschillende soorten melk. Namelijk, magere, halfvolle en volle melk. Ook hier worden de verschillen in impedantie gemeten om vervolgens classificatie op toe te passen.

De combinatie van alle bovenstaande voeding en vloeistoffen geeft 4 uitgebreide en gevarieerde dataset. Deze datasets worden gebruikt om de verschillende classificatie algoritmes te evalueren.

3.3.1 Gebruikte sensoren

De eerste gebruikte sensor voor deze proef meet de zuiver ohmse weerstand van het product. Deze wordt simpelweg gemeten door twee geleiders in het product te prikken. Deze twee geleiders worden samen met een voorschakelweerstand aangesloten op de Arduino. De weerstandswaarde van de substantie ertussen wordt vervolgens ingelezen op de analoge ingang van de Arduino. De twee geleiders werden in een vaste opstelling geplaatst zodat de afstand ertussen steeds gelijk is. Deze meetmethode is niet heel nauwkeurig maar geeft een goede eerste indicatie van het product en eenvoudige data om aan de slag mee te gaan in het algoritme.

Een tweede parameter die wordt gemeten is de sterkte van het elektrolyt. Dit wordt gedaan op basis van twee verschillende metalen. De werking hiervan steunt op het principe van een batterij cel. Als er twee verschillende soorten metalen aanwezig zijn in een elektrolyt, vindt er een chemische reactie plaats die een spanning opwekt. Dit elektrolyt is ook aanwezig in zuren zoals voeding of vloeistoffen. De spanning wordt het best opgewekt met twee verschillende metalen zoals koper en zink. In deze proef wordt dus een koper en zink electrode ingebracht in het product en wordt er een spanning opgewekt die evenredig is met de sterkte van het elektrolyt tussen de elektrodes. Deze spanning wordt gemeten op de analoge poort van de Arduino. Uit testen is gebleken dat de elektrolyt meting nauwkeurig genoeg is om verschil te zien tussen de verschillende stukken fruit en groenten. Echter bij testen op de verschillende vloeistoffen is gebleken dat de meting niet nauwkeurig genoeg is om onderscheid te maken tussen de concentraties. Vandaar de deze data enkel wordt gemeten bij de classificatie van groeten en fruit.

De derde meetmethode die wordt toegepast in deze proef is een impedantiemeting. Dit is eigenlijk een verbetering van de voorgaande weerstandsmeting. Enerzijds omdat de impedantie wordt gemeten op verschillende frequenties en anderzijds is dit veel nauwkeuriger dan de eerdere eenvoudige weerstandsmeting. Bij deze meting wordt voor elke frequentie twee grootheden uitgelezen. Namelijk, de grootte van de impedantie Z in Ohm (Ω) en de fase θ in graden (°). Deze extra datapunten zijn handig om nauwkeuriger classificatie te kunnen toepassen.

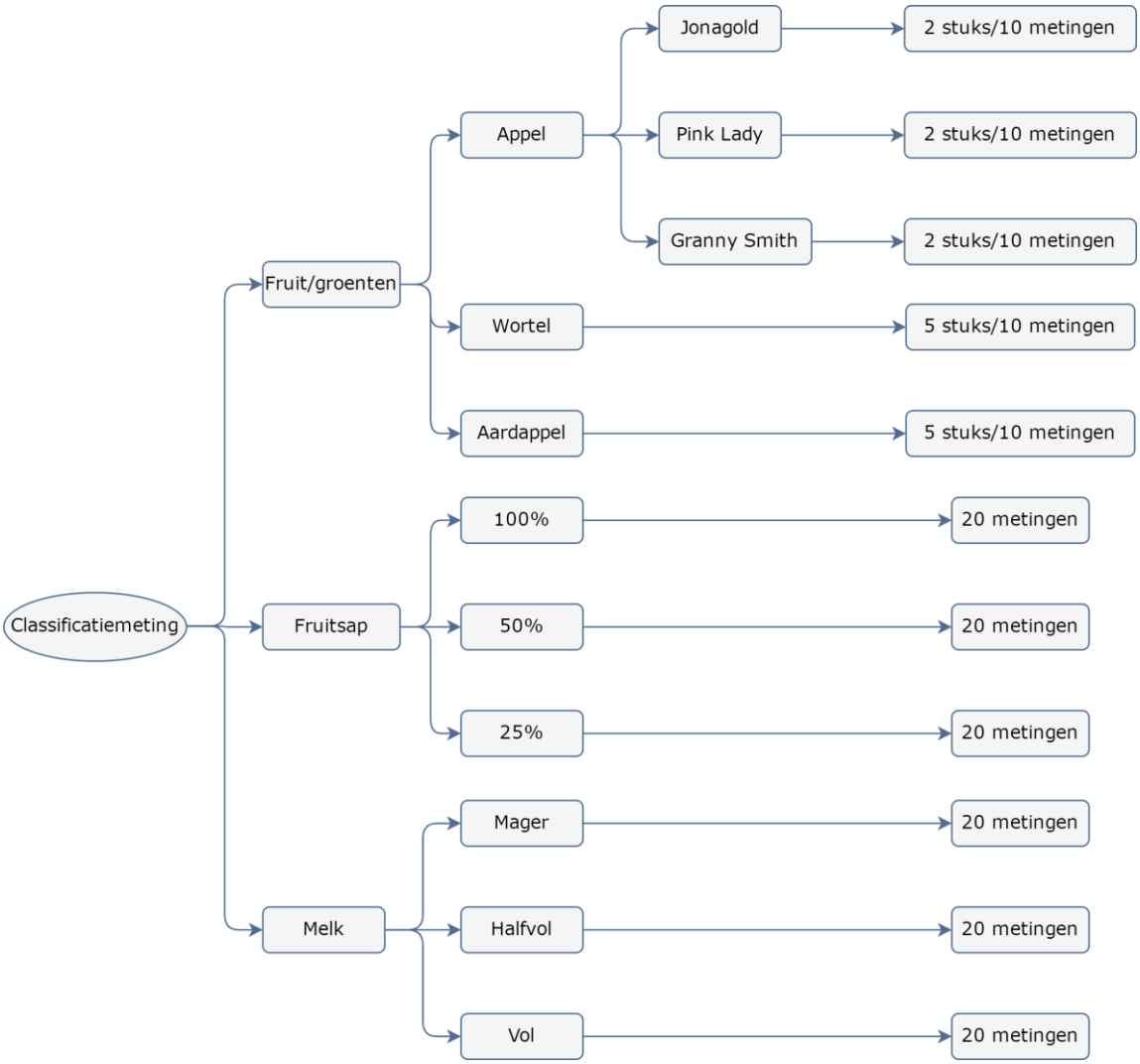
In tegenstelling tot voorgaande metingen wordt de impedantie niet rechtsreeks ingelezen op de Arduino maar via een impedantie meter en een *LabView* interface. De meting gebeurt op drie frequenties, namelijk: 100, 1000 en 10 000Hz. Dit geeft telkens drie extra meetwaarden om verder mee aan de slag te gaan.

3.3.2 Meetresultaten

Bij het testen van de verschillende meetmethodes is besloten om de weerstandsmeting te laten vallen en enkel verder te gaan met de impedantiemetingen en elektrolytmeting. De weerstandsmeting gaf een goede indicatie maar is niet in elke stof even constant. Verder heeft de impedantiemeting als groot voordeel meetwaarden te geven op verschillende frequenties. Deze extra meetpunten zorgen voor een nauwkeurigere classificatie. De weerstandsmeting blijft echter een interessante methode voor snelle, real time metingen op embedded systemen.

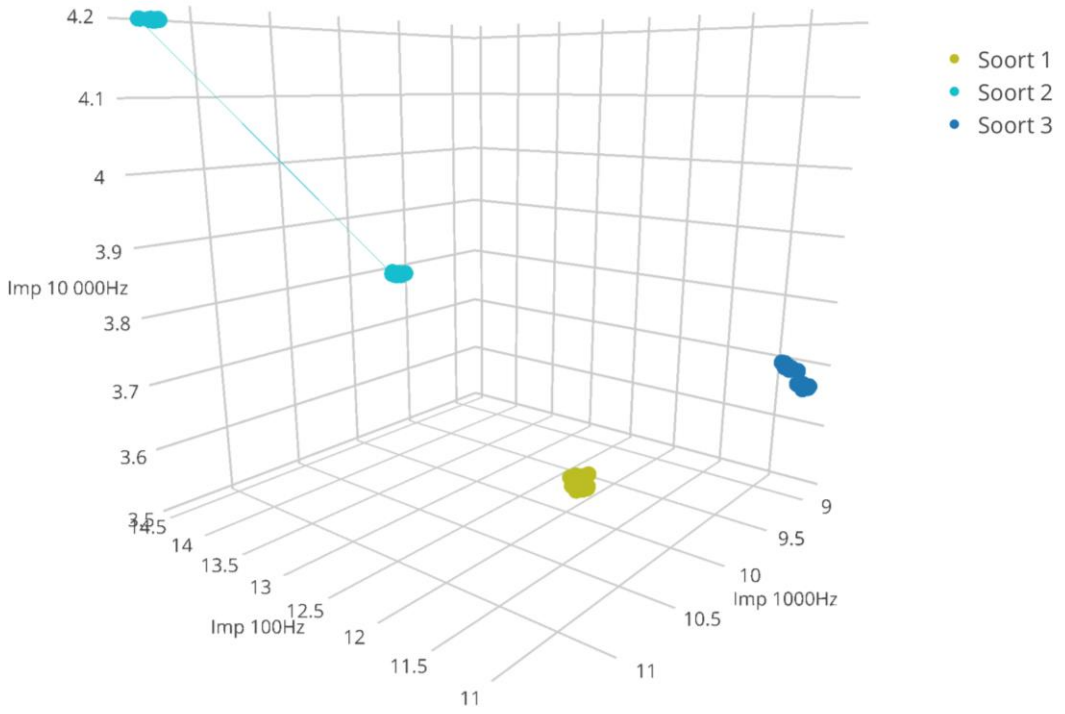
Figuur 14 toont een overzicht van alle uitgevoerde metingen van deze proef. Als extra uitdaging voor de classificatie algoritmes is er gekozen voor metingen te doen op drie verschillende soorten appels. Er werden 10 metingen gedaan op twee stuks per soort. Van de wortels en de aardappelen zijn telkens 5 stuks genomen en 10 metingen per stuk uitgevoerd. Op dit gedeelte zijn telkens de impedantiemeting op 3 frequenties en de elektrolytmeting uitgevoerd, dus 4 waarden per meetpunt.

Verder werden er metingen gedaan op de verschillende concentraties fruitsap en soorten melk. Hier werden telkens 20 metingen per vloeistof gedaan. Omdat de elektrolytmeting geen betekenisvolle resultaten geeft in vloeistoffen, wordt deze hier weggelaten.



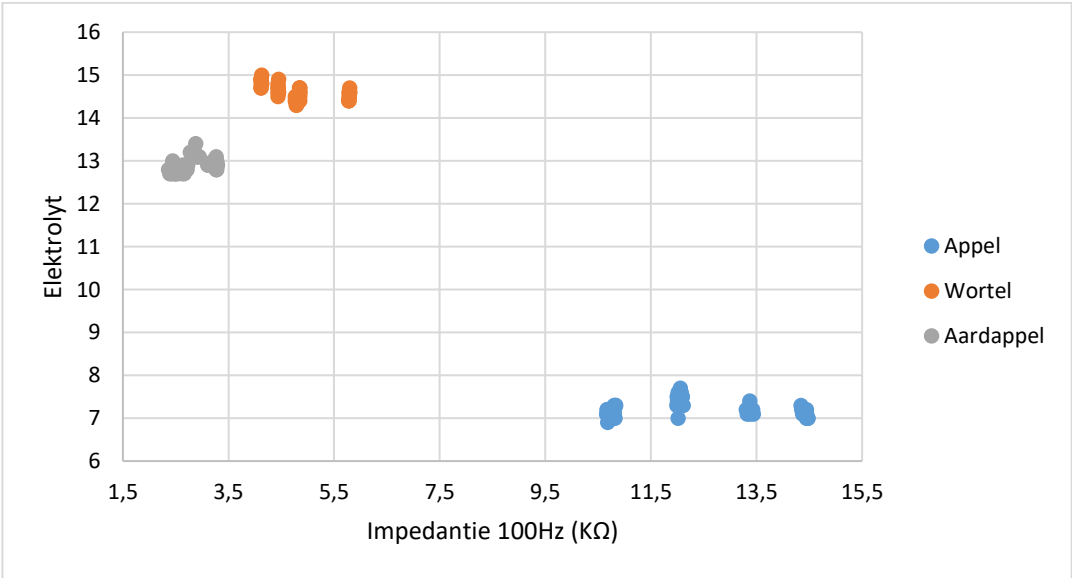
FIGUUR 14: OVERZICHT UITGEVOERDE IMPEDANTIEMETINGEN OP VOEDING EN VLOEISTOFFEN

Ter illustratie worden nu enkele van deze datapunten visueel voorgesteld. In Figuur 15 is een 3D spreiding te zien van de impedantiemeting op de verschillende soorten appels. Op de drie assen zijn de impedantiemetingen op verschillende frequenties uitgezet in kilo ohm. De figuur toont dat er een marge is tussen de verschillende soorten, dit geeft aan dat de data geschikt is om gebruikt te worden als trainingsdata voor een classificatie algoritme. De figuur toont ook dat er een vrij groot verschil is tussen de twee appels van soort 2, de afstand tussen de soorten is echter nog ruim voldoende. Hierdoor is de verwachting dat dit geen probleem zal opleveren voor classificatie.



FIGUUR 15: 3D SPREIDING VAN IMPEDANTIEMETING OP VERSCHILLENDE SOORTEN APPELS

Voorgaande figuur geeft echter enkel een indicatie van de meting op appels. Figuur 16 toont de meetwaarden van de meting op groenten en fruit, uitgezet in een 2D spreiding. Op de X-as is ter illustratie enkel de impedantie op 100Hz uitgezet, in de uiteindelijke dataset worden wel de drie frequenties gebruikt. De Y-as toont de waarden van de elektrolytmeting, deze is zonder eenheid. Er is duidelijk te zien dat de waarden van de wortels en aardappels dichter bij elkaar liggen, maar er is nog steeds voldoende marge voor classificatie.



FIGUUR 16: 2D SPREIDING VAN IMPEDANTIE OP 100 HZ EN ELEKTROLYTMETING OP VOEDING

3.3.3 Classificatie met machine learning

Uit voorgaande meting wordt er dus een uitgebreide dataset gemaakt, waarvan de marges groot genoeg lijken om de belangrijkste stap op toe te passen, het classificatie algoritme. Er worden twee vormen van classificatie gedaan op deze data. Enerzijds een k-NN algoritme op de Intel Curie en de Raspberry Pi, en anderzijds met een neuraal netwerk geschreven met behulp van TensorFlow op de Raspberry Pi.

K-nearest neighbors op Intel Curie

De hardware neuronen op de Intel Curie chip en het k-NN classificatie algoritme zijn eerder al aan bod gekomen. Zulk algoritme wordt nu toegepast op de neuronen om de dataset te classificeren. De code hiervoor wordt uitgeschreven in de Arduino IDE programmeeromgeving. Verder wordt ook gebruik gemaakt van de General Vision library.

Twee belangrijke regels code staan hieronder weergegeven. De 'learn' functie wordt gebruikt om enkele datapunten op te slaan in de neuronen, ook de categorie wordt meegegeven, zodat het k-NN model aanleert welke waarden bij welke categorie horen. Het model werd ervoor al aangemaakt en ingesteld. Vervolgens wordt de 'classify' methode gebruikt om nieuwe waarden, waarvan de categorie nog onbekend is te classificeren.

```
K-NN.learn(pattern,LEN, CAT_APPEL);

K-NN.classify(pattern, LEN, K, dists, cats, nids);
```

Figuur 17 toont de output van een van de testen van het k-NN model. Dit is een test die werd uitgevoerd op de dataset van fruit en groenten. Eerst worden 3 neuronen van het model aangeleerd met patronen van elke categorie. Elke neuron heeft een 'aif' parameter, deze geeft aan wat het invloedgebied is van elke categorie. Deze is groter voor categorie 1 (appel) dan de andere twee categorieën, dit bevestigt dus wat eerder te zien was in Figuur 16.

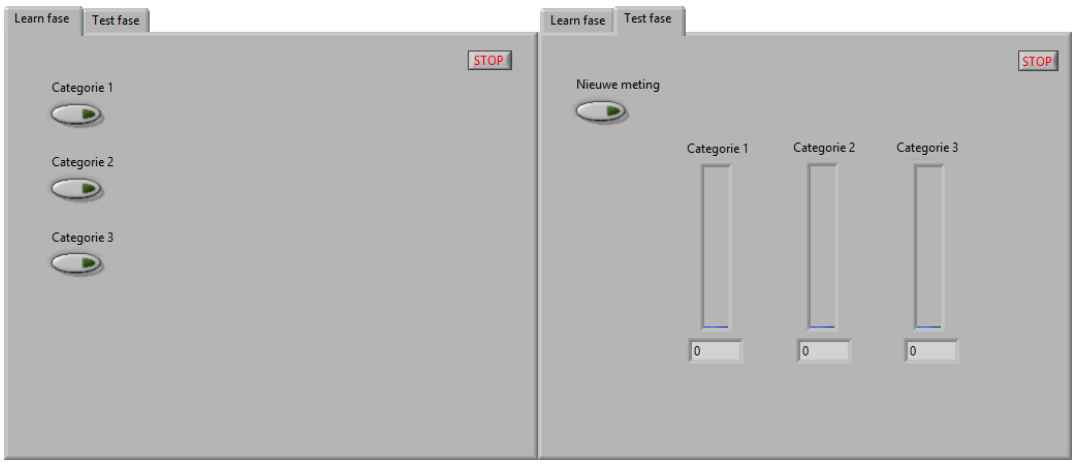
Vervolgens worden drie nieuwe datapunten ingeladen in het model, zonder categorie. Er is te zien dat deze categorie elke keer juist voorspeld wordt. Ook is de afstand tussen het datapunt en het model telkens klein. Dit alles geeft aan dat het model goed werkt.

```
Cure neuronen - Classificeren
Neuron size =128
Neurons available =128
Neurons committed =0

Gebruikte neuronen weergeven:, aantal=3
neuron 1        model=121, 98, 35, 77,  ncr=1   aif=215 cat=1
neuron 2        model=44, 39, 26, 147,  ncr=1   aif=55  cat=2
neuron 3        model=29, 26, 14, 132,  ncr=1   aif=55  cat=3
Gebruikte neuronen weergeven:, aantal=3
neuron 1        model=121, 98, 35, 77,  ncr=1   aif=181 cat=1
neuron 2        model=44, 39, 26, 147,  ncr=1   aif=50  cat=2
neuron 3        model=29, 26, 14, 132,  ncr=1   aif=50  cat=3

Nieuw patroon classificeren:
Neuron 1, Categorie=1, at Afstand=5

Nieuw patroon classificeren:
Neuron 2, Categorie=2, at Afstand=10

Nieuw patroon classificeren:
Neuron 3, Categorie=3, at Afstand=10
```

FIGUUR 17: SERIËLE OUTPUT VAN HET K-NN MODEL OP INTEL CURIE. FASE 1: TRAINING VAN DE NEURONEN, FASE 2: CLASSIFICATIE

Het k-NN model classificeert de nieuwe datasets altijd in de juist categorie, de nauwkeurigheid op alle gebruikte datasets is dus 100%. Zoals eerder vermeld is de trainingstijd bij k-NN zeer snel, omdat de data wordt opgeslagen in de neuronen en er geen herhalende trainingsfase moet worden doorlopen. Dat is ook het geval bij dit model op Intel Curie. De verwerkingstijd kan dus als praktisch onbestaande beschouwd worden

Om het inlezen en uitlezen van het model grafisch te maken is er een LabView interface ontwikkeld. Deze is te zien in Figuur 18. Het LabView programma communiceert over de seriële bus met behulp van LabView VISA met het Arduino programma. In de 'learn fase' kan met een druk op de knop nieuwe data aan elke categorie worden toegevoegd. Via deze knop krijgt het Arduino programma het commando om een nieuwe meting op te starten en de data in te lezen in het model.



FIGUUR 18: GRAFISCHE LABVIEW INTERFACE OM HET ARDUINO PROGRAMMA TE BEDIENEN

In de ‘test fase’ kan er met de knop een nieuwe meting worden gestart die door het model wordt geclassificeerd. Via onderstaande exponentiele functie, die geprogrammeerd in het LabView programma, worden de afstand (x) tot de categorie omgezet tot een percentage.

Dit percentage geeft aan hoe groot de kans is dat de nieuwe meetwaarde bij een bepaalde categorie hoort en wordt vervolgens weergegeven op de interface.

$$y = 2{,}58 + 102{,}59 * e^{(-0{,}052*x)}$$

K-nearest neighbors op Raspberry Pi

Ter vergelijking van het model op de Intel Curie is er ook een k-NN-algoritme uitgewerkt op de Raspberry Pi. Het model is geschreven met behulp de SKlearn library. Het model is zeer gelijkaardig aan dat van de Intel Curie. Het grootste verschil is dat de Intel Curie gebruikt maakt van de unieke hardware neuronen. Met een k waarde van 3 wordt ook hier de data van alle datasets correct geclassificeerd. De nauwkeurigheid is dus ook 100%. Zoals steeds bij een k-NN model is de verwerkingstijd te verwaarlozen.

Neuraal netwerk op Raspberry Pi

Op de Raspberry Pi is er ook een neuraal netwerk geschreven om de dataset te classificeren. Om dit algoritme te schrijven is de reeds eerder beschreven TensorFlow library van google gebruikt.

Alle data uit de voorgaande metingen is opgeslagen als een ‘.csv’ databestand. Deze bestanden zijn makkelijk in te lezen in de python programmeertaal, waar TensorFlow is op gebaseerd. Vervolgens werd deze data geconverteerd naar een classificatie dataset. Deze dataset wordt opgesplitst in een deel trainingsdata en een deel testdata. De gebruikte verhouding is 70% trainingsdata en 30% testdata. Dit is een verhouding die veel gebruikt wordt en goede resultaten geeft in eerdere experimenten en literatuur.

Hierna wordt er een model opgesteld. Dit is een diep neuraal netwerk (DNN) classificatie model. De term diep neuraal netwerk wordt gebruikt omdat er hier met meerdere verborgen lagen van neuronen wordt gewerkt in het model. Om het model op te stellen worden de volgende argumenten gebruikt:

- feature_columns, geeft door welke kolommen van het databestand bruikbare inputdata bevat. In dit geval 3 of 4, afhankelijk van de gebruikte data.
- hidden_units, wordt gebruikt om in te stellen hoeveel verborgen lagen er gebruikt worden. In dit geval 3, en deze bevatten respectievelijk 10,20 en 10 neuronen.
- n_classes , om in te stellen hoeveel categorieën er moeten geclassificeerd worden. Dit zijn voor alle datasets 3 categorieën.
- model_dir, wordt gebruikt om de plaats in te stellen waar het model wordt opgeslagen.

Nu het classificatie model is ingesteld kan het getraind worden met de trainingsdata. Om dit te doen wordt de data en een aantal herhalingen ingesteld in het model. Over het algemeen geldt, meer herhalingen geeft een nauwkeuriger model, maar hier wordt later dieper op ingegaan.

Wanneer het model voor een aantal herhalingen getraind is met de inputdata, is het tijd om te evalueren met de testdata. Aan de evaluatiemethode wordt de test dataset meegegeven en wordt éénmaal volledig doorlopen. Voor elk input vector zal het model een voorspelling doen. Deze voorspelling wordt dan vergeleken met de effectieve categorie waaraan de input data is toegewezen. Uiteindelijk wordt er dan een nauwkeurigheidsscore getoond van het model. Bijvoorbeeld, als de testdata 1000 meetpunten bevat en aan 950 hieraan wordt de juist categorie toegewezen, is de nauwkeurigheid 95%.

Tenslotte kan met de voorspellingsfunctie een voorspelling gemaakt worden op nieuwe datavectoren waaraan nog geen categorie is toegewezen. De voorspellingsmethode wijst dus een categorie toe aan de vector op basis van het getrainde model.

In Tabel 1 zijn de resultaten te zien van het model op de eerder vergaarde datasets. De verlies en nauwkeurigheid waarden zijn voor de verschillende metingen te zien tot 200 herhalingen. De testen worden telkens gedaan tot de nauwkeurigheid 100% bereikt. De nauwkeurigheid is eerder al uitgelegd. De verlieswaarde wordt berekend op basis van de trainingsfase en testfase en geeft aan hoe goed het model is aangeleerd. Het is geen percentage zoals de nauwkeurigheid maar wel de som van de error van elk datapunt in de dataset. Uiteraard is het de bedoeling bij het trainen van het model om deze waarde zo laag mogelijk te maken. In het ideale scenario daalt de verlieswaarde bij elke herhaling.

Allereerst valt op te merken dat voor elke dataset uiteindelijk een nauwkeurigheid van 100% wordt gehaald. Dit is een teken dat het model perfecte voorspellingen doet voor elk nieuw datapunt. Ook is het duidelijk dat de verlieswaarde het hoogste is voor de soorten appels. Dit is te verklaren omdat er weinig impedantie verschil is tussen de verschillende soorten appels.

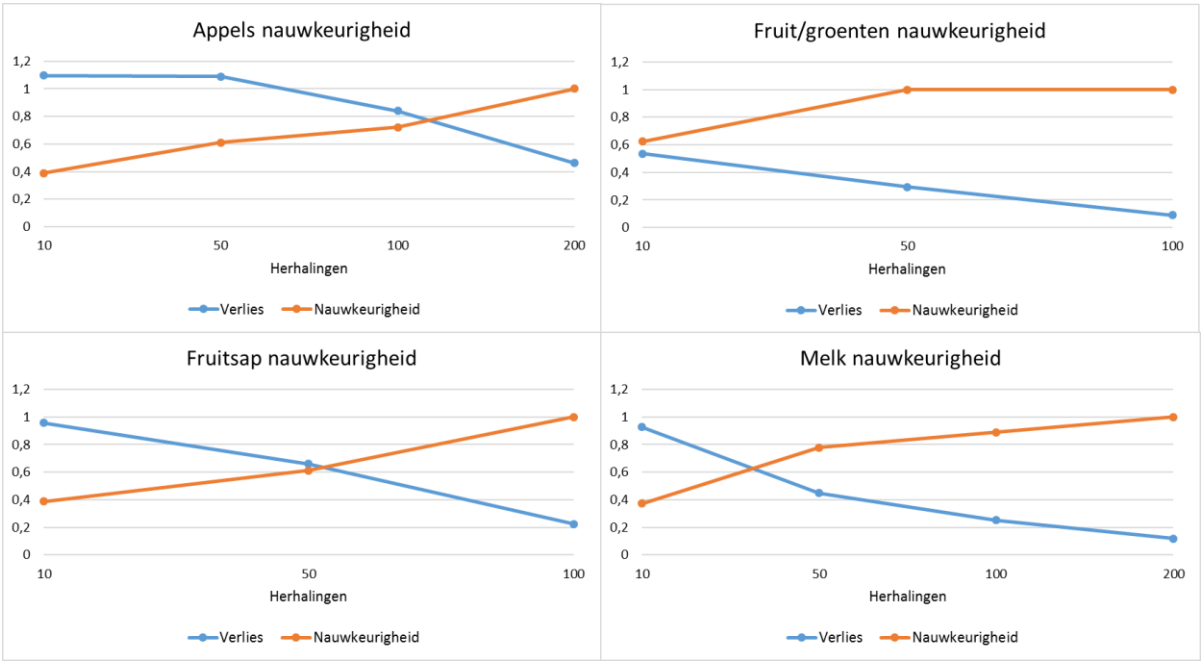
De verlieswaarde op verschillende soorten fruit en groenten is echter zeer laag en zakt ook snel, net zoals de nauwkeurigheid zeer snel stijgt en al snel zijn maximum bereikt. Dit is te verklaren doordat de elektrolytmeting als extra parameter werd opgenomen in deze dataset. Een extra inputwaarde verhoogt de nauwkeurigheid van een neuraal netwerk dus gevoelig.

TABEL 1: VERLIES EN NAUWKEURIGHEID VAN DE 4 DATASETS TOT 200 HERHALINGEN

Meting:	Soorten appels			
Herhalingen:	10	50	100	200
Verlies:	1,096	1,090	0,840	0,463
Nauwkeurigheid:	38,9%	61,1%	72,0%	100%
Meting:	Appel, wortel, aardappel			
Herhalingen:	10	50	100	200
Verlies:	0,535	0,295	0,090	/
Nauwkeurigheid:	62,5%	100%	100%	/
Meting:	Fruitsap			
Herhalingen:	10	50	100	200
Verlies:	0,956	0,659	0,226	/
Nauwkeurigheid:	38,9%	61,1%	100%	/
Meting:	Melk			
Herhalingen:	10	50	100	200
Verlies:	0,928	0,448	0,252	0,120
Nauwkeurigheid:	37,4%	77,8%	88,9%	100%

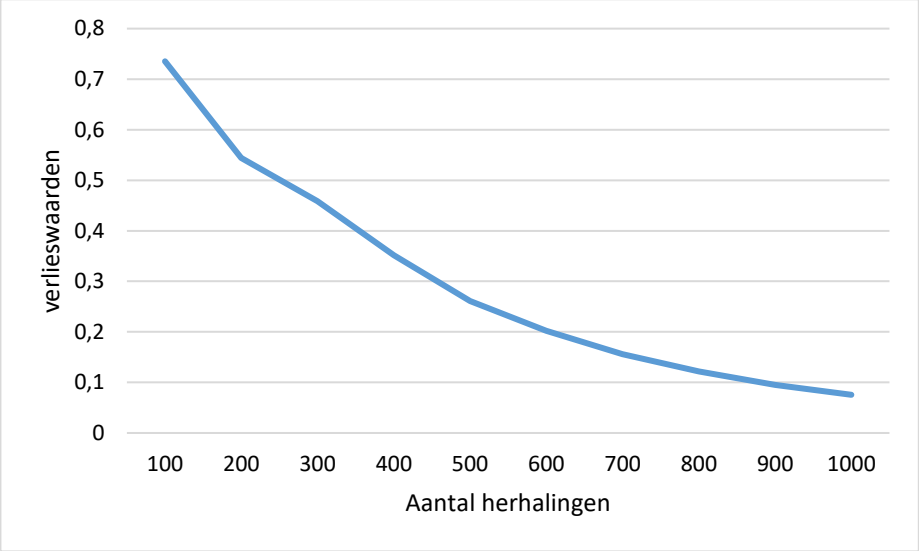
Ook de snelheid van dit model is uiteraard van belang. Bij de datasets met 3 inputwaarden wordt een snelheid behaald van 136 herhalingen per seconde op de Raspberry Pi. Met de iets grotere dataset van 4 inputwaarden daalt de snelheid naar 124 herhalingen per seconde. Voor het trainen van de appels dataset tot 100% is dus 1,61 seconden nodig. Bij de datasets van fruit/groenten en concentraties fruitsap is dit 0,74 seconden en de tijd bij soorten melk bedraagt de traingingstijd 1,47 seconden.

In Figuur 19 is een grafische voorstelling te zien van de verhouding van de verlies- en nauwkeurigheidswaarden voor de verschillende datasets. Dit geeft een goed beeld van hoe de waarden evolueren met meer herhalingen. De nauwkeurigheid is hier geschaald naar een waarde tussen 0 en 1 in plaats van een percentage.

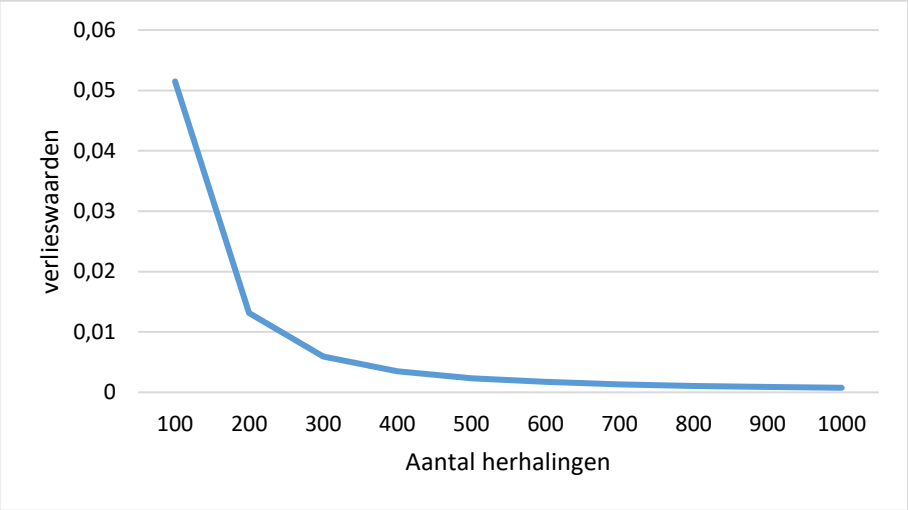


FIGUUR 19: VERLIES/NAUWKEURIGHEID GRAFIEK VOOR DE 4 VERSCHILLENDE DATASETS GETRAIND TOT 100% NAUWKEURIGHEID

Figuur 20 en Figuur 21 zijn de verlieswaarden van het neuraal netwerk verder uitgezet voor twee datasets. Dit toont aan dat zelf bij een nauwkeurigheid van 100% op de testdataset, het model toch nog beter kan getraind worden met meer herhalingen. Het valt ook op dat bij de fruit/groenten dataset die één extra input heeft, de verliesfunctie veel sneller richting 0 gaat.



FIGUUR 20: VERLIESWAARDEN VAN HET NEURAAL NETWERK MODEL BIJ 1000 HERHALINGEN VOOR APPELS



FIGUUR 21: VERLIESWAARDEN VAN HET NEURAAL NETWERK MODEL BIJ 1000 HERHALINGEN VOOR FRUIT/GROENTEN

3.3.4 Conclusie

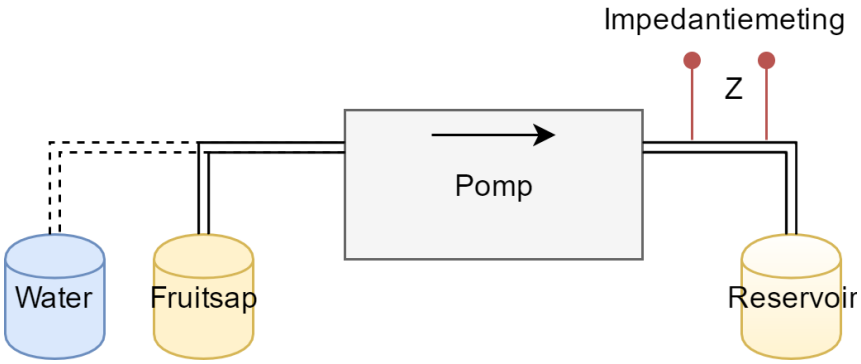
Er kan geconcludeerd worden dat zowel de twee k-NN modellen als het neurale netwerk de classificatie van alle datasets met 100% nauwkeurigheid kunnen uitvoeren. Beide softwaremodellen zijn ook zeer modulair geschreven zodat ze op allerhande andere datasets en toepassingen kunnen toegepast worden. De snelheid van het k-NN model is vrijwel simultaan. Bij het neuraal netwerk loopt de snelheid op, afhankelijk van de grootte en complexiteit van de dataset.

3.4 Doorspoeling leidingen

Een volgende toepassing om de algoritmes te testen is het doorspoelen van leidingen. Ook deze toepassing is voorgesteld vanuit het onderzoekscentrum. De doelstelling hier is om te detecteren wanneer een leiding van bijvoorbeeld een frisdrankproducent volledig uitgespoeld is met water. Met machine learning algoritmes kan er echter nog extra informatie verkregen worden zoals het detecteren van de concentratie water/fruitsap op een bepaal moment in het spoeling proces, of een voorspelling in de toekomst over wanneer de leiding volledig uitgespoeld zal zijn.

3.4.1 Meetopstelling

Om de doorspoeling van leidingen te simuleren is er een proefopstelling op kleine schaal gemaakt. Deze proefopstelling bestaat uit een kleine pomp met plastic leidingen om vloeistoffen door te pompen. In het tweede deel van de leiding, achter de pomp, zijn twee elektrodes aangebracht. Via deze elektrodes wordt de impedantie van de tussenliggende vloeistof gemeten. Het verschil met de voorgaande meting is dat de impedantie hier maar op één frequentie, namelijk 1000Hz, wordt gemeten in plaats van op drie frequenties. Dit komt doordat er een ander toestel moest worden gebruikt voor de meting. In Figuur 22 is een schema te zien van de opstelling. De resultaten van deze kleine opstelling geven een goede indicatie voor het gedrag van vloeistoffen in een groter pompsysteem.

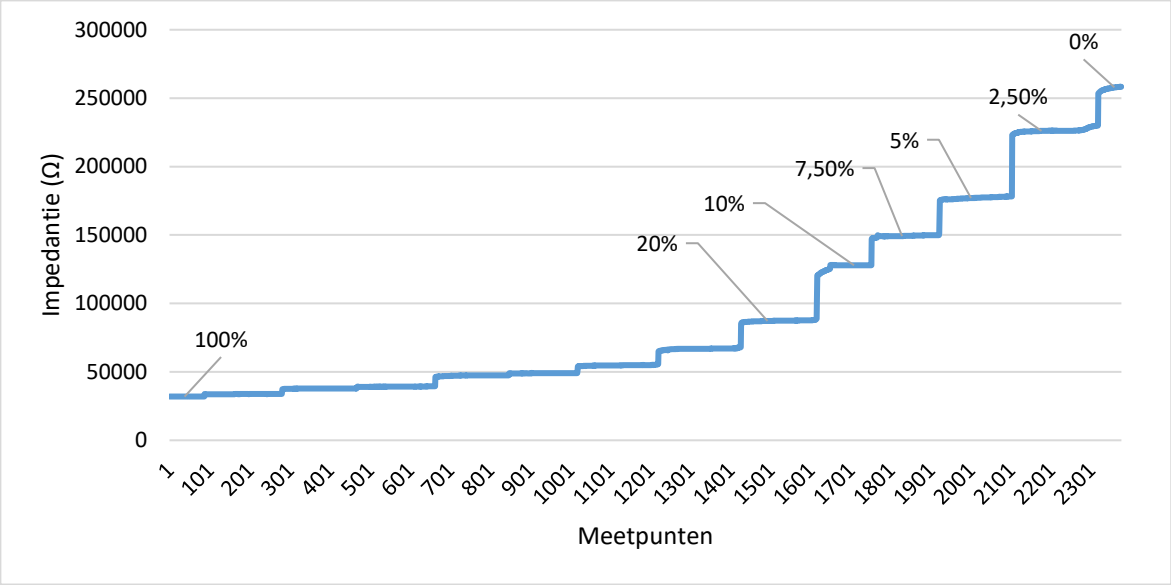


FIGUUR 22: MEETOPSTELLING VAN DE DOORSPOELING VAN LEIDINGEN

3.4.2 Meetresultaten

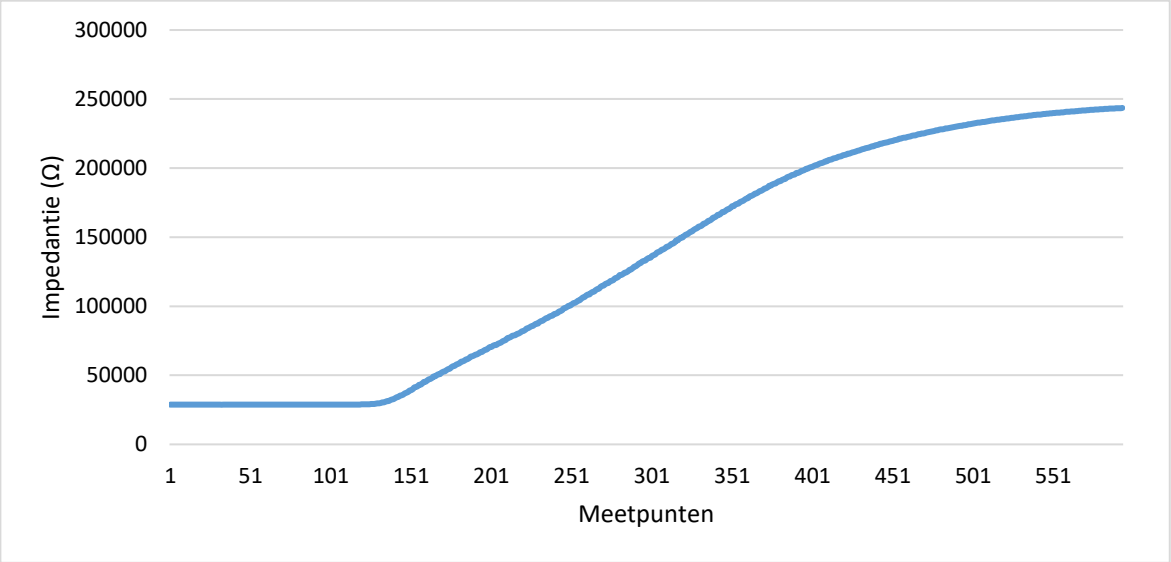
Om een inschatting te krijgen van de impedantiewaarden van een bepaalde concentratie van de vloeistof wordt er eerst een meting gedaan die gebruikt kan worden als vergelijkingspunt. Bij deze meting worden er eerst verschillende concentraties aangemaakt gaande van 100% fruitsap tot 0% in stappen van 10%, en nog extra concentraties van 7,5%, 5% en 2,5%. Dit wordt gedaan om de laatste sporen van het fruitsap voor de volledige uitspoeling goed te kunnen vergelijken. Zoals eerder vermeld wordt er maar op 1 frequentie gemeten en is er dus telkens maar één datapunt per meting.

In Figuur 23 zijn de resultaten te zien van de metingen op verschillende concentraties. Bij meetpunt 1 gaat er 100% fruitsap door de leiding en de impedantie dus het laagst. Vervolgens wordt concentratie telkens verlaagd tot er 100% water door de leiding gaat. Er is nu dus een referentiewaarde gemeten voor al de verschillende concentraties. Hierbij valt op dat de verschillen in impedantie het hoogst zijn bij de lage concentraties van de vloeistof. Dit is positief omdat vooral het overgangspunt van fruitsap naar water moet geclassificeerd worden.



FIGUUR 23: MEETRESULTATEN VAN ALLE CONCENTRATIES VAN 100% TOT 0% (TRAININGSDATA)

Figuur 24 toont de resultaten van een volledige spoeling met water. Hierbij gingen er geen verschillende concentraties door de leiding maar is er eerst 100% fruitsap door de leiding gegaan en vervolgens gespoeld met 100% water. Dit is dus een referentie meting van hoe de spoeling zou moeten gebeuren.



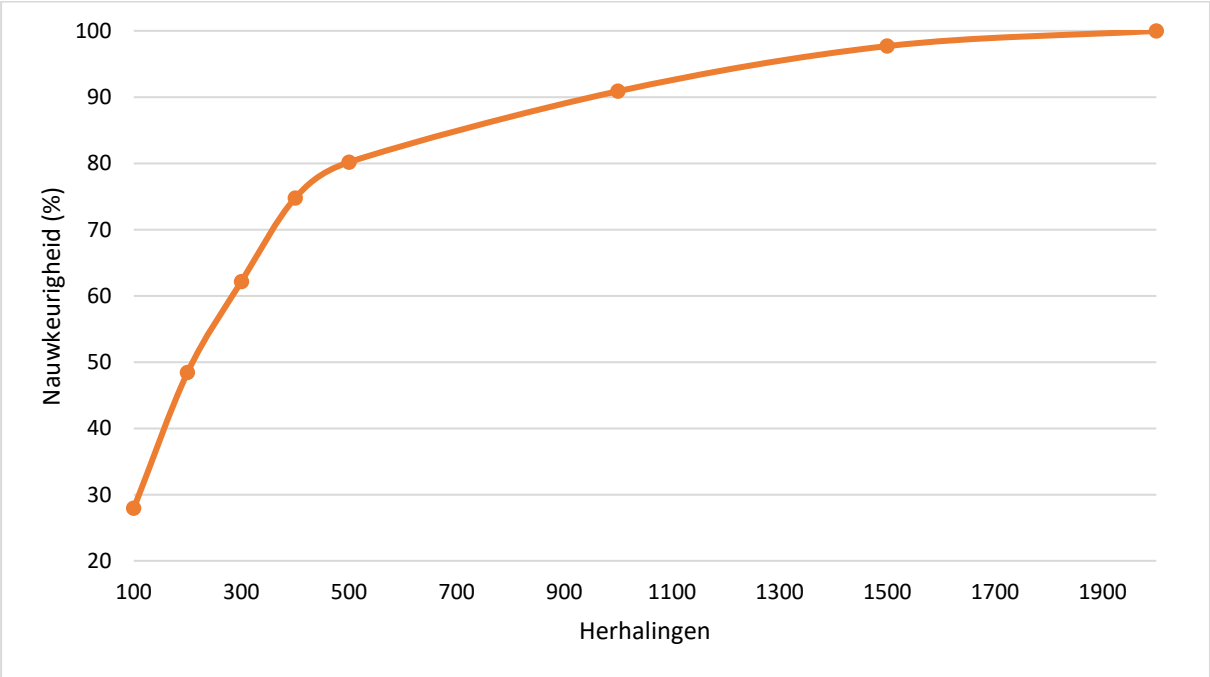
FIGUUR 24: MEETRESULTATEN VAN EEN VOLLEDIGE SPOELING MET WATER (VALIDATIEDATA)

3.4.3 Classificatie met machine learning

Het classificeren van deze dataset gebeurt enkel op Raspberry Pi en niet op de Intel Curie module. Voor de classificatie van de concentraties wordt hetzelfde neuraal netwerk gebruikt als bij de classificatie van voeding met slechts kleine aanpassingen. Dit toont aan dat eenzelfde neuraal netwerk heel universeel bruikbaar is en voor veel verschillende soorten datasets kan ingezet worden.

Het grootste verschil is dat er nu slechts één input wordt gebruikt, namelijk de impedantie op 1000Hz. Voor de classificatie zijn er 14 categorieën, namelijk de concentraties gaande van 100% fruitsap tot 0% in stappen van 10%, en nog extra concentraties van 7,5%, 5% en 2,5%. De metingen op verschillende concentraties, eerder te zien in Figuur 23, wordt gebruikt als trainingsdata om het model op te stellen. Omdat de meting over een vrij lange periode is uitgevoerd zijn er per concentratie veel meetpunten om het netwerk mee te trainen. Dit verhoogt de nauwkeurigheid van het model maar verhoogt ook de trainingstijd.

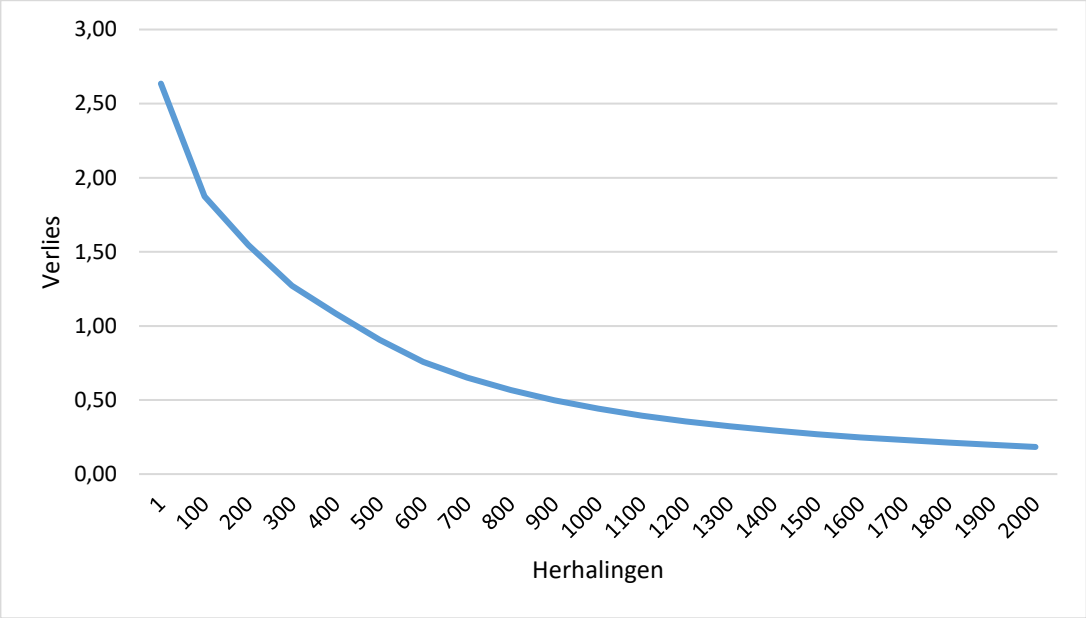
Opnieuw wordt deze dataset willekeurig opgesplitst in een trainingsdataset van 70% en een testdataset van 30%. De testdata wordt gebruikt om de nauwkeurigheid te controleren. In Figuur 25 is te zien hoe de nauwkeurigheid stijgt bij het aantal uitgevoerde herhalingen waarop het model getraind wordt. Hier valt meteen op dat het model veel vaker moet getraind worden om een goed resultaat te behalen dan bij de vorige dataset. Bij bijvoorbeeld 100 herhalingen wordt er slechts een nauwkeurigheid gehaald van ongeveer 28%. Dit komt doordat de dataset in dit geval veel groter is als bij de vorige proef en er veel meer categorieën moet aangeleerd worden. Bij 1500 herhaling stijgt de nauwkeurigheid echter naar een respectabele 97,7%. Bij 2000 herhalingen wordt uiteindelijk ook op deze dataset een nauwkeurigheid van 100% gehaald.



FIGUUR 25: NAUWKEURIGHEID VAN HET NEURAAL NETWERK MODEL VAN 100 TOT 2000 HERHALINGEN

In Figuur 26 Zijn voor dezelfde trainingsherhalingen de verlieswaarden te zien. Ook hierbij valt op dat deze aanvankelijk veel hoger liggen dan bij de vorige proeven. Deze daalt echter snel tot een acceptabele waarde bij veel herhalingen.

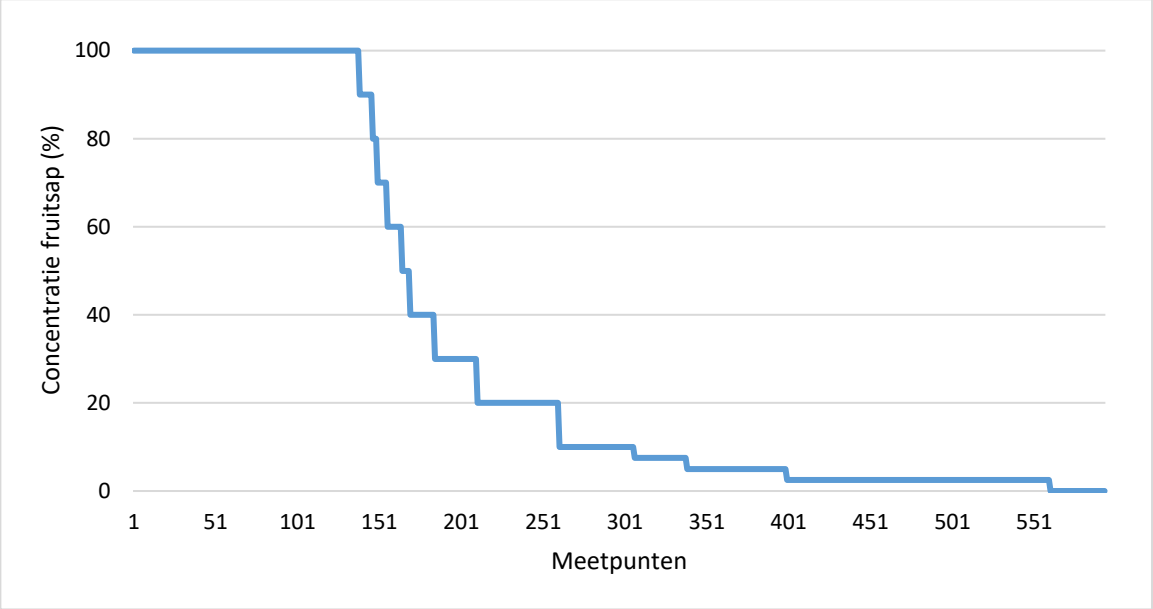
De snelheid bij het trainen van het model gebeurt aan gemiddeld 18,7 herhalingen/seconden. De snelheid ligt dus door de grootte van de dataset een stuk lager dan bij de vorige proeven. Om het model te trainen tot een nauwkeurigheid van 100% is dus een trainingstijd nodig van ongeveer 1min 47seconden. Bij een grotere dataset vallen de limieten van de rekenkracht van de Raspberry Pi dus wel op.



FIGUUR 26: VERLIESWAARDEN VAN HET NEURAAL NETWERK MODEL TOT 2000 HERHALINGEN

Het model is nu dus getraind en getest op nauwkeurigheid, maar is nog niet toegepast op nieuwe data. Daarvoor wordt de voorspellingsfunctie gebruikt waaraan een nieuwe dataset wordt doorgegeven. De data die hiervoor wordt gebruikt is de volledige doorspoeling, eerder te zien in Figuur 24 en is dus de validatiedata voor het model.

Figuur 27 toont de voorspelling van de concentraties op basis van deze validatiedata. Het valt op dat bij het spoelen van water de concentratie plots snel zal zakken tot een waarde van ongeveer 10%. Hierna zakt de concentratie relatief traag en worden er nog enige tijd een lage concentratie gedetecteerd tot de doorspoeling compleet is op 0%.



FIGUUR 27: VOORSPELLING VAN DE CONCENTRATIE OP BASIS VAN DE VALIDATIEDATA

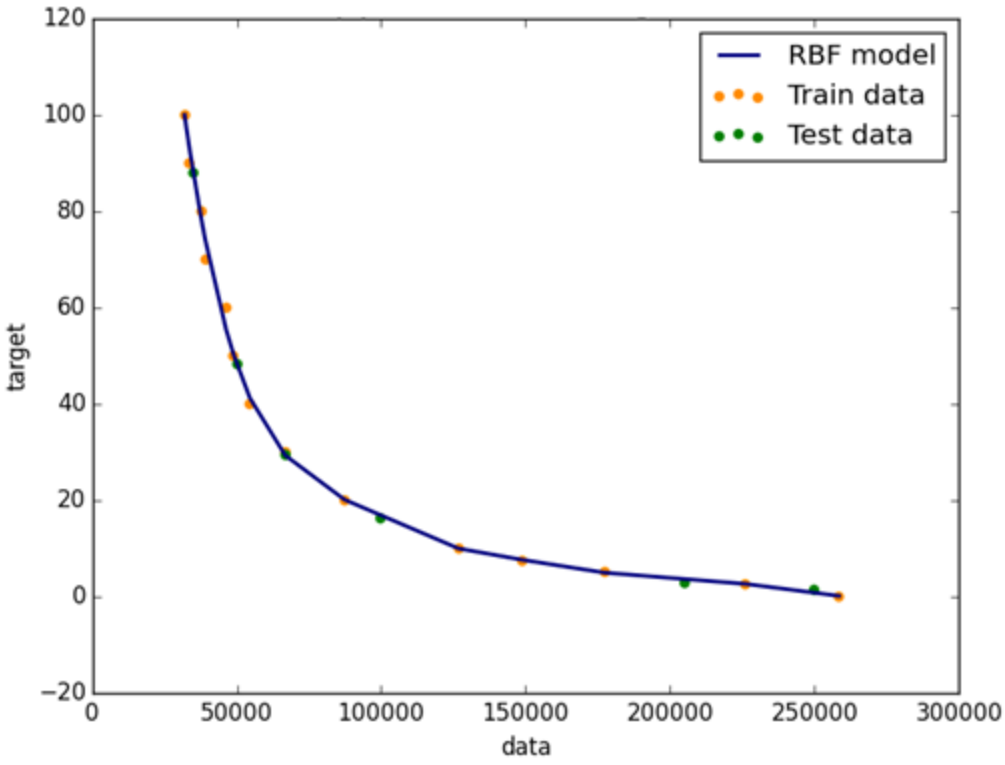
3.4.4 Regressie met machine learning

Er worden goede resultaten gehaald door de data in te delen in categorieën en op basis hiervan de concentraties te classificeren. Er is bij deze meting echter sprake van een continue dataset. Vandaar is het ook interessant om hier een regressie algoritme op toe te passen. Aan de data is meteen te zien dat er niet over een lineair verloop kan gesproken worden. Er kan namelijk geen rechte lijn worden getrokken om een nauwkeurig verband te zoeken tussen de data.

Om de regressielijn te berekenen is er gekozen om support vector regressie toe te passen met behulp van Scikit-learn. De methode van SVM wordt meestal gebruikt als classificatiemethode, maar kan ook aangepast worden om aan regressie te doen. Voor deze SVM regressie is er een keuze tussen een lineaire, polynomiale en RBF kernel. Aangezien deze data niet goed kan beschreven worden met een lineair of polynomiaal verloop, wordt er gekozen voor de RBF kernel om de beste fit te vinden [34].

In Figuur 28 is het regressiemodel te zien. Voor de trainingsdata van dit model is telkens één waarde per concentratie genomen, gebaseerd op de data van Figuur 23. Deze waarde is de minimum impedantie waarde voor 100% en de maximum impedantie waarde voor 0%, om het volledige bereik te hebben. Voor alle andere concentraties wordt telkens het gemiddelde genomen van alle impedantiewaarden. Op basis van deze waarden wordt de regressie berekend.

Deze lijn benadert op zeer nauwkeurige wijze alle waarden. Door de regressie te berekenen op basis van SVM met RBF kernel wordt er een veel nauwkeuriger resultaat behaald dan bijvoorbeeld met een lineaire of polynomiale benadering. Om de nauwkeurigheid uit te drukken wordt er gebruik gemaakt van de R^2 coëfficiënt. Om de R^2 te berekenen wordt er gekeken naar de afstand tussen de trainingsdatapunten en de best-fit lijn. De best mogelijke score hiervoor is 1,00. De score van onderstaand model is 0,9968, dit is dus een zeer nauwkeurige benadering. In het groen zijn enkele willekeurige test datapunten te zien, waarvoor een waarde is voorspeld door het model.



FIGUUR 28: RESULTAAT VAN REGRESSIE MET SVM ALGORITME

3.4.5 Conclusie

De classificatie van de meetdata via een neuraal netwerk uitgeschreven in TensorFlow is succesvol. Na 2000 trainingsherhalingen wordt een nauwkeurigheid van 100% gehaald. Het uitspoelen van de leiding kan dus nauwkeurig gedetecteerd worden. Het valt wel op dat er bij deze dataset, die meer datapunten en meer categorieën bevat, een langere traingingsperiode nodig is. Ook is het gebruikt van classificatie hier niet helemaal optimaal, omdat de doorspoeling maar in stappen van 2,5% kan opgedeeld worden.

Ook bij het toepassen van een regressie algoritme wordt er een goed resultaat gehaald. SVM-regressie met een RBF kernel geeft een nauwkeurige fit op basis van de trainingsdata en kan nieuwe punten nauwkeurig voorspellen. De R^2 coëfficiënt bij dit model is 0,9968. Op basis van deze regressie kan de concentratie fruitsap dus nauwkeurig voorspeld worden. Uit deze regressielijn kan er ook andere informatie gehaald worden. Zoals een voorspelling doen over wanneer de leiding volledig gespoeld zal zijn. Verder is het interessant dat deze vorm van regressie kan toegepast worden op bijna alle soorten data, onafhankelijk van het verloop van de functie.

4 Besluit

4.1 Conclusie

Dit onderzoek is van start gegaan met de doelstelling te onderzoeken wat de verschillende mogelijkheden zijn om machine learning principes toe te passen op embedded systemen. Om deze vraag te beantwoorden is er eerst onderzoek verricht naar welke toepassingen van machine learning er in de literatuur al gedaan werden op deze embedded systemen.

Vervolgens zijn er twee systemen gekozen om de testen op uit te voeren. De Intel Curie, vanwege de unieke ingebouwde hardware neuronen, en de Raspberry Pi, vanwege de veelzijdigheid en eenvoudige implementatie. Om verschillende algoritmes te implementeren en te vergelijken is er gebruik gemaakt van software libraries van General Vision, Scikit-learn en TensorFlow. Voor de classificatie van data is er gebruik gemaakt van k-nearest neighbors en neurale netwerk modellen. Voor regressie werd er gewerkt met een support vector machine model en een radial basis function kernel.

Om de verschillende algoritmes te vergelijken op nauwkeurigheid en snelheid, is er gebruik gemaakt van drie soorten datasets. Allereerst met de iris dataset die vrij te verkrijgen is op het internet. Vervolgens met eigen impedantiedata, gemeten op verschillende soorten groenten, fruit en vloeistoffen en ten slotte op impedantiedata van het doorspoelen van een leiding. In Tabel 2 is een overzicht te zien van alle resultaten.

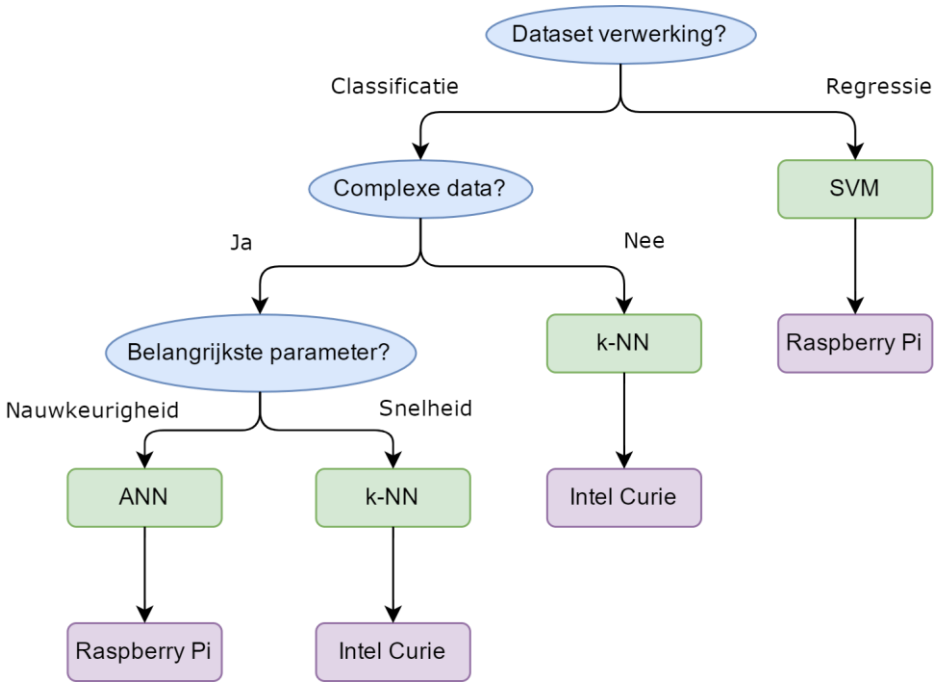
TABEL 2: OVERZICHT VAN DE NAUWKEURIGHEID EN SNELHEID VAN DE VERSCHILLENDE MODELLEN

	Nauwkeurigheid:	Trainingstijd (s):
Iris dataset		
KNN (Intel Curie):	93,33%	<1
KNN (RPi):	90%	<1
ANN (RPi):	96,60%	16,5
Voedsel data		
KNN (Intel Curie):	100%	<1
KNN (RPi):	100%	<1
ANN (RPi):	100%	1,61
Pomp data		
ANN (RPi):	100%	107
SVM-RBF (RPi):	R^2=0,9968	2

Het valt op dat alle modellen een goede nauwkeurigheid halen van 90% of meer. Verder is de trainingstijd sterk afhankelijk van het gebruikte model. Bij de resultaten van de iris dataset is het duidelijk dat er een afweging moet gemaakt worden tussen nauwkeurigheid en snelheid. Als de snelheid van uitvoering zeer snel moet zijn en de nauwkeurigheid minder van belang is kan er best gekozen worden voor een k-NN model. Het valt hier ook op dat de Intel Curie iets nauwkeuriger is vanwege de ingebouwde hardware neuronen. Bij een complexere dataset waarbij de nauwkeurigheid het belangrijkste is, is een neuraal netwerk echter aanbevolen.

Een extra argument voor k-NN op Intel Curie is de implementatie op de Arduino 101. Via de Arduino kan er sneller een programma ontwikkeld worden en kan er een rechtsreeks verbinding gemaakt worden met verschillende sensoren dankzij de ingebouwde analoog digitaal converter.

Voor regressieproblemen kan geconcludeerd worden dat de SVM-RBF methode een zeer nauwkeurige fit kan maken op de data van de pomp. Verder is het heel belangrijk met dit model een regressielijn kan worden gevonden voor alle soorten data, onafhankelijk van het verband. Wat bijvoorbeeld niet mogelijk is met lineaire of polynomiale regressie. In Figuur 29 is een overzicht gegeven in flowchart vorm, van de aanbevelingen die kunnen gemaakt worden op basis van dit onderzoek.



FIGUUR 29: FLOWCHART: AANBEVELING VAN EEN MACHINE LEARNING MODEL OP BASIS VAN DE TOEPASSING

Alle modellen die in deze thesis zijn uitgeschreven zijn modulair geschreven en makkelijk aan te passen voor andere toepassingen. Zo kunnen de algoritmes aangepast worden voor toepassingen als beeldverwerking of data van andere sensoren. Ook voor de aanvankelijke probleemstelling van het opvolgen van het genezingsproces van wonden zijn de modellen in de toekomst zeker inzetbaar.

4.2 Toekomstig werk

Dit onderzoek geeft een overzicht van enkele classificatie en regressie modellen op basis van twee systemen. Dit geeft een goed resultaat voor een breed aantal toepassingen en is dus zeker geslaagd. Andere systemen zoals FPGA's en DSP's zijn echter niet opgenomen in dit onderzoek. Uit literatuuronderzoek is echter wel gebleken dat hier zeker potentieel is voor machine learning.

Door het enorme aanbod aan verschillende machine learning algoritmes zijn deze niet allemaal aan bod kunnen komen. De focus in dit onderzoek heeft gelegen op k-nearest neighbors, neurale netwerken en support vector regressie. Er zijn echter nog verschillende andere algoritmes die kunnen getest en geïmplementeerd worden op embedded systemen.

De modellen die in dit onderzoek werden voorgesteld zijn modulair opgebouwd. Met slechts kleine aanpassingen kunnen ze dienen voor andere soorten data. De belangrijkste volgende stap is om data die afkomstig is van het genezingsproces van wonden in te laden. Op basis van deze resultaten kan er verder gewerkt worden naar een afgewerkt prototype voor de medische sector.

Literatuurlijst

[1] "IMO - IMOMEC." [Online]. Available: http://www.uhasselt.be/IMO. [Accessed: 29-May-2017].

[2] Dp. J. Posnett BA, Dms. F. Gottrup MD, H. L. MSc, and M. A. G. Saal BA, "The resource impact of wounds on health-care providers in Europe," *J. Wound Care*, vol. 18, no. 4, p. 154, 2009.

[3] S. F. Khalil, M. S. Mohktar, and F. Ibrahim, "The theory and fundamentals of bioimpedance analysis in clinical status monitoring and diagnosis of diseases," *Sensors (Switzerland)*, vol. 14, no. 6, pp. 10895–10928, 2014.

[4] "Types of machine learning algorithms | en.proft.me." [Online]. Available: http://en.proft.me/2015/12/24/types-machine-learning-algorithms/. [Accessed: 22-May-2017].

[5] Z. Yao and W. L. Ruzzo, "A Regression-based K nearest neighbor algorithm for gene function prediction from heterogeneous data," *BMC Bioinformatics*, vol. 7, no. Suppl 1, p. S11, 2006.

[6] "A Detailed Introduction to K-Nearest Neighbor (KNN) Algorithm." [Online]. Available: https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/. [Accessed: 23-May-2017].

[7] "Introduction to Support Vector Machines — OpenCV 2.4.13.2 documentation." [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html. [Accessed: 24-May-2017].

[8] T. Elomaa, "Tools and Techniques for Decision Tree Learning," 1996.

[9] M. A. Nielsen, "Neural Networks and Deep Learning." Determination Press, 2015.

[10] Christos Stergiou, "Neural Networks." [Online]. Available: https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html#Introduction to neural networks. [Accessed: 08-May-2017].

[11] C. McCormick, "Radial Basis Function Network (RBFN) Tutorial." [Online]. Available: http://mccormickml.com/2013/08/15/radial-basis-function-network-rbfn-tutorial/. [Accessed: 24-May-2017].

[12] "Python Programming Tutorials." [Online]. Available: https://pythonprogramming.net/regression-introduction-machine-learning-tutorial/?completed=/machine-learning-tutorial-python-introduction/. [Accessed: 30-May-2017].

[13] G. Park and D. J. Inman, "Structural health monitoring using piezoelectric impedance measurements," *Philos. Trans. R. Soc. London A Math. Phys. Eng. Sci.*, vol. 365, no. 1851, 2007.

[14] J. Hyttinen, "Bioimpedance Measurement Device for Chronic Wound Healing Monitoring," 2010.

[15] A. Kekonen, J.-E. Eriksson, M. Bergelin, H. Ylänen, and J. Viik, “A Quantitative Method for Monitoring Wound Healing,” pp. 2–5.

[16] H. C. Lukaski and M. Moore, “Bioelectrical impedance assessment of wound healing.,” *J. Diabetes Sci. Technol.*, vol. 6, no. 1, pp. 209–12, 2012.

[17] J. Estrela da Silva, J. P. Marques de Sá, and J. Jossinet, “Classification of breast tissue by electrical impedance spectroscopy,” *Med. Biol. Eng. Comput.*, vol. 38, no. 1, pp. 26–30, Jan. 2000.

[18] A. Riul *et al.*, “Wine classification by taste sensors made from ultra-thin films and using neural networks,” *Sensors Actuators, B Chem.*, vol. 98, no. 1, pp. 77–82, 2004.

[19] F. Li, W. Wlodarski, U. Marschner, S. Sauer, E. Starke, and W. J. Fischer, “ScienceDirect Development of a novel gas sensing algorithm based on impedance spectroscopy,” vol. 0, pp. 3–6, 2014.

[20] S. Kun, B. Ristic, R. A. Peura, and R. M. Dunn, “Algorithm for tissue ischemia estimation based on electrical impedance spectroscopy,” *IEEE Trans. Biomed. Eng.*, vol. 50, no. 12, pp. 1352–1359, Dec. 2003.

[21] M. Courbariaux *et al.*, “Binarized Neural Networks: Training Neural Networks with Weights and Activations Constrained to +1 or −1.”

[22] B. Mcdanel and H. T. Kung, “Embedded Binarized Neural Networks.”

[23] Fan Yang and M. Paindavoine, “Implementation of an rbf neural network on embedded systems: real-time face tracking and identity verification,” *IEEE Trans. Neural Networks*, vol. 14, no. 5, pp. 1162–1175, Sep. 2003.

[24] H. Caner, H. S. Gecim, and A. Z. Alkar, “Efficient Embedded Neural-Network-Based License Plate Recognition System,” *IEEE Trans. Veh. Technol.*, vol. 57, no. 5, pp. 2675–2683, Sep. 2008.

[25] S. Himavathi, D. Anitha, and A. Muthuramalingam, “Feedforward Neural Network Implementation in FPGA Using Layer Multiplexing for Effective Resource Utilization,” *IEEE Trans. Neural Networks*, vol. 18, no. 3, pp. 880–888, May 2007.

[26] S. Bashyal and G. K. Venayagamoorthy, “Scholars’ Mine Embedded Neural Network for Fire Classification Using an Array of Gas Sensors,” 2008.

[27] T. Choudhury *et al.*, “The Mobile Sensing Platform: An Embedded Activity Recognition System,” *IEEE Pervasive Comput.*, vol. 7, no. 2, pp. 32–41, Apr. 2008.

[28] “CurieNeurons_Tools - General Vision Inc.” [Online]. Available: http://www.general-vision.com/products/curieneurons_tools/.

[29] “CurieNeurons library,” 2016.

[30] M. Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.”

[31] F. Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[32] “The Iris Dataset — scikit-learn 0.18.1 documentation.” [Online]. Available: http://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html. [Accessed: 04-Jun-2017].

[33] “tf.contrib.learn Quickstart | TensorFlow.” [Online]. Available: https://www.tensorflow.org/get_started/tflearn. [Accessed: 04-Jun-2017].

[34] A. J. Smola and B. Sc, “A Tutorial on Support Vector Regression,” 2003.

Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Toepassen van machine learning algoritmes op embedded systemen

Richting: **master in de industriële wetenschappen: elektronica-ICT**
Jaar: **2017**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

Goyens, Nick

Datum: **6/06/2017**