KU LEUVEN

Interpretatie en modellering van multi instrumentele analytische data met Deep Learning

Marjolein SAELENS

Promotor(en): Prof. E. Courtijn
Co-promotoren: Ing. Simon Lambert
Ing. Delphine Verstraete

Masterproef ingediend tot het behalen van de graad van master of Science in de industriële wetenschappen: *Chemie, Chemie*

Academiejaar 2019-2020

Voorwoord

Deze thesis vormt het hoogtepunt en een mooie afsluiter van mijn opleiding master in de industriële wetenschappen: chemie op de Technologiecampus van de KU Leuven in Gent. Het schrijven van deze thesis is een zeer leerrijke maar zeker ook uitdagende periode geweest en dit op verschillende vlakken.

In de eerste plaats was het bijzonder uitdagend om me te verdiepen in een domein waar ik totaal geen ervaring in had, namelijk programmeren en dataverwerking. Het leerproces verliep zeker niet altijd van een leien dakje, maar het onderwerp leek me wel meteen enorm te boeien.

Een tweede aspect wat voor extra uitdaging zorgde bij het schrijven van deze thesis waren de uitzonderlijke omstandigheden als gevolg van het virus Covid-19. Door het grootste deel van de thesis thuis te moeten afleggen, miste ik op een bepaalde manier wel een eerste ervaring in een industrieel bedrijf waar ik zo naar uitgekeken had. Desondanks bleef er nog steeds goede communicatie en begeleiding, waardoor het mogelijk was om zelfs in deze nare omstandigheden toch een volwaardige thesis te schrijven.

Met dit dankwoord wil ik graag even stil staan bij alle mensen die hebben bijgedragen aan de totstandkoming van deze thesis. Zonder hun steun, hulp en begeleiding had het nooit tot dit eindresultaten kunnen komen.

In de eerste plaats zou ik graag professor Dorine Bruneel bedanken, het is dankzij haar dat ik voor mijn thesis terecht kon bij het bedrijf Christeyns NV. Daarnaast bedank ik ook mijn promotor professor Eddy Courtijn voor de goede begeleiding en ondersteuning tijdens het schrijven van deze thesis, in toch wel uitzonderlijke omstandigheden.

Speciale dank gaat uit naar mijn copromotoren Simon Lambert en Delphine Verstraete voor de uitstekende begeleiding gedurende het volledige traject. Ondanks hun eigen drukke agenda kon ik steeds met al mijn thesis gerichte vragen en problemen bij hen terecht. Ook hun kritische en snelle feedback bij het nalezen van mijn thesis waren zeer waardevol. Zonder hun deskundige kennis en adviezen zou dit nooit het eindresultaat geweest zijn. Hartelijk bedankt voor de tijd en moeite die jullie aan mij besteed hebben.

Mijn ouders en mijn zus wil ik ook nog graag bedanken voor alle steun, hulp en wijze raad die ik van hun kreeg, niet alleen gedurende deze laatste uitdaging, maar gedurende mijn volledige studietraject. Als laatste wil ik toch ook nog mijn vriend Dries Brabants bedanken voor zijn onderdak, goed gezelschap en steun tijdens de soms wel moeilijke lockdown periode.

Marjolein Saelens

16 juni 2020, Gent

Samenvatting

Bij Christeyns wordt textiel geanalyseerd aan de hand van drie verschillende analysetechnieken, namelijk witheidsmetingen met een spectrofotometer, Fourier-transformatie infraroodspectroscopie (FTIR) en high-performance liquid chromatography (HPLC). Per staal wordt dus een relatief grote hoeveelheid gegevens gegenereerd. De gegevens van alle geanalyseerde stalen door de jaren heen worden opgeslagen in een steeds groter wordende database. In deze thesis wordt onderzoek gedaan naar verschillende methoden voor de verwerking van data afkomstig van de analyse van textielstalen.

Verschillende dataverwerking methoden zoals principale-componentenanalyse (PCA), principale-componentregressie (PCR) en partiële kleinste kwadraten regressie (PLS) lijken geschikt voor het onthullen van verborgen informatie in de dataset. Uiteindelijk wordt toegelegd op artificiële intelligentie en in het bijzonder deep learning voor de verwerking van de dataset. Met behulp van verschillende testen wordt onderzocht wat de mogelijkheden zijn van deze techniek. Vervolgens wordt de techniek toegepast voor het zoeken van correlaties in de gegevens van de drie verschillende analysetoestellen.

In dit onderzoek wordt de invloed van verschillende parameters, zoals de grote van de dataset, de invloed van pre-processing en de parameterinstellingen van de neurale netwerken nagegaan. Vervolgens wordt met behulp van deep learning verschillende classificaties en regressies uitgevoerd met bepaalde gegevens uit de beschikbare dataset. Er wordt getracht om aan de hand van de X-, Y- en Z-waarden enerzijds en de volledige emissie-spectra anderzijds verschillende kenmerken van het textiel zoals de basiswitheid, de witheidindex en de concentratie aan optische witmaker te voorspellen.

Uit de resultaten blijkt het mogelijk om met deep learning stalen te classificeren naargelang hun basiswitheid en hun witheidindex en dit zowel met behulp van de X-, Y- en Z-waarden als met de volledige spectra. Na het verwijderen van enkele afwijkende stalen in de dataset is het ook mogelijk om de witheidindex van de stalen zeer nauwkeurig te voorspellen.

Meer moeilijkheden worden ondervonden bij de regressies en classificaties van de concentratie optische witmakers. Hier worden de gegevens van HPLC gecombineerd met de gegevens van de witheidmetingen. De combinatie van twee verschillende analyseapparatuur blijkt niet evident met de huidige dataset. In verder onderzoek kunnen pre-processing methoden zoals PCA en PLS worden toegepast om alleen de relevante informatie van de dataset over te houden, wat de prestaties van de opgebouwde modellen kan verbeteren.

Trefwoorden: Chemometrie, Deep learning, Artificiële neurale netwerken, Optische witmakers, Python.

Mededeling COVID-19

Initieel zou mijn thesis uit twee grote delen bestaan. In de eerste plaats zou onderzoek gedaan worden naar verwerking van de beschikbare dataset afkomstig van verschillende toestellen voor de analyse van textiel. Naast een theoretische studie over de verschillende multivariate analysetechnieken die hiervoor gebruikt kunnen worden, zou de techniek deep learnig in de praktijk worden toegepast voor het verwerken van de dataset.

De gegevens uit de dataset zijn afkomstig van Fourier-transformatie infraroodspectroscopie (FTIR), high-performance liquid chromatography (HPLC) en witheidsmetingen met een spectrofotometer. Witheidsdata zijn er voorlopig voldoende aanwezig om met deep learning aan te slag te kunnen. Bij de FTIR-spectroscopie is dit niet het geval, daarenboven is er nog niet echt gekend welke informatie hier allemaal kan uitgehaald worden. Vooraleer multivariate data-analyses op de FTIR-spectra worden toegepast, is het interessant om te onderzoeken bij welke golfgetallen in het spectrum de aanwezigheid van bepaalde additieven kan aangetoond worden. Een tweede deel van deze thesis bestond erin om na te gaan of er een mogelijkheid bestaat om bepaalde additieven op het textiel te detecteren met behulp van FTIR-spectroscopie, bijvoorbeeld de aanwezigheid van optische witmakers of wasverzachter. Naast het verwerken van de dataset zou ik dus vooral FTIR-analyses en witheidsmetingen uitvoeren in het labo en hiervoor zou ik verschillende additieven voor textiel gebruiken zoals een aantal wasverzachters en optische witmakers.

Voor beide onderdelen was ik al ver gevorderd in literatuuronderzoek en ook al begonnen met enkele experimenten toen er na amper drie weken bericht van de KU Leuven kwam dat, omwille van het virus COVID-19, masterproeven enkel nog vanop afstand mochten afgelegd worden en dit al zeker tot aan de paasvakantie. Gelukkig gebeurt een groot deel van de thesis op de computer, namelijk de verwerking van de dataset met multivariate analysetechnieken. Mits enkele ingrepen kon dit onderdeel volledig van thuis uit worden verdergezet. Het experimentele deel met onder anderen FTIR-spectroscopie zou later worden uitgevoerd wanneer er terug naar de bedrijven mag gegaan worden.

Al snel werd duidelijk dat terugkeren naar Christeyns niet voor juni zou gebeuren en dat er jammer genoeg dus geen tijd meer zou zijn voor het experimentele deel rond FTIR-spectroscopie. Ik kon me daarom volledig focussen op het verwerken van de beschikbare dataset.

Abstract

At Christeyns, textiles are analysed using three different analysis techniques: whiteness measurements with a reflectance spectrophotometer, Fourier-transformation infrared spectroscopy (FTIR) and high-performance liquid chromatography (HPLC). Therefore, a large amount of data is generated for each sample. The data of all analysed samples over the years is stored in an ever-increasing database. In this thesis, research is conducted into various methods for processing data from the analysis of textile samples.

Various data processing methods such as principal component analysis (PCA), principal component regression (PCR) and partial least squares (PLS) seem suitable for revealing information in the data. Ultimately, the focus in this thesis is artificial intelligence and in particular deep learning methods for the processing of the data. The possibilities of the technique were examined based on various tests. Subsequently, the technique was applied to search for correlations in the data of the three different analysis techniques.

In this research, the influence of various parameters is investigated, such as the size of the data set, the influence of data pre-processing and the parameter settings of the neural networks. Subsequently, with the aid of deep learning, various classifications and regressions are carried out. It is investigated whether it is possible to predict different characteristics of the textile, such as the basic whiteness, the whiteness index and the concentration of optical brightener, using the X, Y and Z values on the one hand and the full emission spectra on the other hand.

The results show that it is possible to classify samples with deep learning according to their basic whiteness and their whiteness index, using the X, Y and Z values as well as the full spectra. After removing some abnormal samples from the data set, it is also possible to accurately predict the whiteness index of the samples.

More difficulties are encountered in the classifications and regressions of the concentration of optical brighteners. Here, the data from HPLC is combined with the data from the whiteness measurements, which is clearly a bigger challenge. In further research, pre-processing methods such as PCA and PLS can be applied to retain only the relevant information from the dataset, which can improve the performance of the predictive models.

Keywords: Chemometrics, Deep learning, Artificial neural network, Optical Brighteners, Python.

Extended abstract

In this thesis, research is carried out on behalf of Christeyns into suitable data processing methods for the data of textile analyses. In the professional textile care department at Christeyns, textile samples are analysed using three different analysis techniques: Fourier-transformation infrared (FTIR) spectroscopy, high-performance liquid chromatography (HPLC) and whiteness measurements with a spectrophotometer. A large amount of data is therefore generated for each sample. The data of all analysed samples over the years is stored in an ever-increasing database.

This thesis contains a theoretical study on how to handle such large data sets. Some multivariate techniques are discussed and are examined whether they are suitable for processing the data set from the textile analyses. Ultimately, the focus is on artificial intelligence and in particular deep learning methods for the processing of the data. This is a technique that uses so-called neural networks in which the best possible connection is detected between input and output data. Optimization is made to a minimal difference between fair values and predicted value. Neural networks are able to “learn” patterns in the dataset themselves and to build their own model for interpretation and predictions of unknown samples. Neural networks are suitable for performing classifications of samples in discrete groups as well as for regressions, predicting continuous values.

The influence of various parameters when conducting deep learning is investigated. The parameters discussed are the size of the data set, the importance of pre-processing methods and the parameter settings of each neural network. Many of these parameters strongly depend on the complexity of the proposed goal and can only be determined by trial and error. This makes building a suitable neural network model for classifying or predicting unknown data a highly iterative process.

Attempts are being made to predict certain properties of the textile, such as the basic whiteness, the whiteness index and the amount of optical brightener, based on the X, Y and Z values on the one hand and the entire emission spectrum on the other. Once a suitable neural network model is found, it is possible to classify the samples into different classes based on their basic whiteness. In this classification 99 % of the samples are classified in the correct class. The accuracy appears to be slightly lower in the classification according to the whiteness index. With the most suitable model, 93 % of the samples are classified in the correct class. The results could be improved by removing the deviating samples from the dataset.

When performing the regression of the whiteness index, the abnormal samples with a negative whiteness index were removed from the data set. The mean absolute percentage error between the predicted and the actual whiteness index is 1,5 %. The model built with a neural network is therefore able to accurately predict the whiteness index of unknown samples.

In the classifications and regressions of the amount of optical brightener, the spectrophotometer data is used to predict data from HPLC. This is more difficult than predicting spectrophotometer data using other data from the same spectrophotometer. At first, a model was obtained for predicting the amount of optical brightener with a mean percent absolute error (MAPE) of 1500%. This means that, on average, the predicted concentration of optical brighteners is 15 times more or 15 times less than the actual concentration. The data set was optimized, several abnormal samples were removed from the data set and new insights were

obtained into the effects of the concentration of optical brighteners on the emission spectrum. This improved the results of the model, the MAPE is 20 %. The Rearon for this deviation is the measurement error on both devices and low response curves for optical brightener concentrations higher than 1500 ppm.

This research shows that deep learning can be a promising technique for processing the data set from various textile analysis. Further research is possible on this subject. This technique can be expanded to the FTIR-spectra, which, combined with reflectance spectra and HPLC data can be used to predict the concentration of other additives such as fabric softeners and other surfactants. What is important in further research is to combine deep learning with different multivariate features such as PCA and PLS as pre-processing methods for the optimization of the dataset and thereby improving the performance of the neural network.

INHOUD

Symbolenlijst

$\bar{x}(\lambda)$	Kleur-matching functie rood	[-]
$\bar{y}(\lambda)$	Kleur-matching functie groen	[-]
$\bar{z}(\lambda)$	Kleur-matching functie blauw	[-]
L*	Grijsschaal	[-]
a*	Rood-groen schaal	[-]
b*	Geel-blauw Schaal	[-]
X, Y en Z	Tristimulus waarden van bepaald object	[-]
X_n, Y_n en Z_n	Tristimulus waarden van perfect reflecterende diffusor	[-]
x en y	Chromaticiteitscoördinaten van bepaald object	[-]
x_n en y_n	Chromaticiteitscoördinaten van perfect reflecterende diffusor	[-]
c	Voortplantingssnelheid	[m/s]
λ	Golflengte	[m]
υ	Frequentie	[Hz]
$\bar{\nu}$	Golfgetal	[cm^{-1}]
E	Energie	[J]
h	Constante van Planck	[J.s]
w	Weight	[-]

Lijst met afkortingen

FTIR	Fourier-transformatie infrarood spectroscopie
HPLC	High-performance liquid chromatography
OAS	Oppervlakte actieve stoffen
DAS	Diamino stilbeen derivaten
DSBP	Distyryl bifenyl derivaten
CIE	Commission on Illumination
WI	Witheidindex
IR	Infrarood
ATR	Attenuated total reflection (Verzwakte totale reflectie)
DoE	Design of Experiments
OVAT	One-variable-at-time
PCA	Principle component analyses
PC	Principle component
LDA	Lineair discriminant analyses
MLR	Multiple linear regression
PCR	Principal component regression
PLSR	Partial least square regression
AI	Artificiële intelligentie
Relu	Rectified linear activation
Tanh	Hyperbolic tangent
MSE	mean squared error
SGD	Stochastic gradient descent
RMSProp	Root mean square propagation
GPU	Graphics processing unit
CPU	Central processing unit
VNC	Virtual network computing
FWA	Fluorescent whitening agent
MAPE	Gemiddelde absolute procentuele fout

Lijst met figuren

Lijst met tabellen

1 INLEIDING

1.1 Christeyns NV

Christeyns NV is een Belgisch chemisch bedrijf dat in 1946 als een zeepfabriek werd opgericht door Jules Robert Christeyns. De fabriek was gevestigd in Gent, waar zich tot op de dag van vandaag nog steeds de hoofdzetel bevindt. Er werden toen voornamelijk zepen en detergenten geproduceerd voor de professionele reinigingsindustrie en de wasserijmarkt. In 1989 werd het bedrijf overgenomen door de familie Bostoen en halverwege de jaren negentig zorgde verscheidende overnames en allianties voor de start van een Europese expansie. De activiteiten binnen Christeyns werden steeds meer uitgebreid. In 2010 kwam er een nieuwe afdeling met name voedselhygiëne en twee jaar later nog één met name ziekenhuishygiëne. Samen met de twee zusterbedrijven Govi en Oscrete is de lokale zeepfabriek uitgegroeid tot een internationale speler op gebied van hygiënechemie (Christeyns), Bouwchemie (Oscrete) en proceschemie (Govi). (1)

Bij Christeyns worden naar eigen zeggen hoogwaardige chemische hygiëneproducten geleverd met oog op kwaliteit, doeltreffendheid en duurzaamheid. Het bedrijf bestaat uit vier verschillende afdelingen, namelijk professionele textielverzorging, voedselindustrie en food retail, professionele reiniging en life science. (1)

Bij de afdeling professionele textielverzorging wordt intensief op zoek gegaan naar effectieve en innovatieve oplossingen voor textielverzorging. Hierbij worden detergenten, desinfecterende middelen en apparatuur voor industriële wasserijen en textielreinigers geïnnoveerd en geproduceerd. (1)

1.2 Objectief

De moderne analytische toestellen en technieken zorgen ervoor dat bedrijven en onderzoekers steeds meer te maken krijgen met een grote kwantiteit aan data. De dataset wordt daarbij ook steeds uitdagender om mee te werken. Dit is ook het geval bij de afdeling professionele textielverzorging van Christeyns. De textielstalen worden er met behulp van verschillende analytische toestellen zowel kwalitatief als kwantitatief geanalyseerd. Door het steeds groter wordend aantal geanalyseerde textielstalen en de grote hoeveelheid gegevens die per staal met de analysetoestellen verkregen worden, werd bij de R&D-afdeling van de textielverzorging al een behoorlijk grote dataset met gegevens verworven. Deze thesis is volledig toegewijd aan het verwerken van die dataset.

De dataset bevat gegevens afkomstig van drie verschillende analysetechnieken namelijk witheidsmetingen met een spectrofotometer, Fourier-transformatie infraroodspectroscopie (FTIR) en high-performance liquid chromatography (HPLC). Het doel van de thesis is om te onderzoeken hoe het best moet worden omgegaan met deze dataset, welke technieken hiervoor het meest geschikt zijn en hoeveel informatie, die op het eerste gezicht niet zichtbaar is, met deze technieken uit de dataset gehaald kan worden. Het hoofddoel is om uiteindelijk correlaties tussen de gegevens van de drie verschillende analysetoestellen op te sporen.

Omwille van de grote hoeveelheid bedrijven en onderzoekers die te maken krijgen met een grote kwantiteit aan data, werd de laatste jaren uitgebreid onderzoek gedaan naar methoden om met grote datasets om te gaan. Het gevolg hiervan is dat er een breed scala aan

technieken werd ontwikkeld waarmee het analyseren en combineren van gegevens steeds toegankelijker werd.

In deze thesis worden de mogelijkheden van verschillende technieken voor de verwerking van de beschikbare dataset theoretisch nagegaan. Uiteindelijk wordt toegelegd op de techniek *deep learning*. Met behulp van verschillende testen wordt onderzocht wat de mogelijkheden zijn van deze techniek. Vervolgens wordt de techniek toegepast voor het zoeken van correlaties in de gegevens van de drie verschillende analysetoestellen.

2 LITERATUURSTUDIE

2.1 Inleiding

Vooraleer gestart wordt met het verwerken van grote datasets, moet eerst een grondige literatuurstudie worden uitgevoerd. Er wordt onderzocht welke technieken geschikt kunnen zijn voor het vinden van correlaties tussen gegevens van verschillende meettoestellen. Deze technieken moeten goed begrepen zijn vooraleer ze kunnen toegepast worden op de aanwezige datasets. Vooraleer een uiteenzetting van de verschillende analysetechnieken gegeven wordt, wordt kort ingegaan op het textiel dat geanalyseerd wordt en de toestellen waarmee dit gebeurt.

Er bestaan veel verschillende soorten textiel, waarbij katoen en polyester de meest voorkomende zijn. Elk textiel kan bovendien op verschillende manieren behandeld worden met allerlei additieven om de eigenschappen ervan te verbeteren. In een eerste deel worden de verschillende additieven kort besproken.

Kwalitatieve analysemethoden worden toegepast om na te gaan over welk textiel het gaat en welke additieven erop aanwezig zijn. Daarnaast zijn er ook nog kwantitatieve methoden die aangeven hoeveel van een bepaald additief er aanwezig is op het textiel. De dataset die moet worden verwerkt bevat meetgegevens die afkomstig zijn van drie verschillende analysemethoden, namelijk witheidsmetingen met behulp van een spectrofotometer, Fourier-analyse infraroodspectroscopie (FTIR) en high-performance liquid chromatography (HPLC). Witheidsmetingen en FTIR-spectroscopie zijn kwalitatieve methoden om de aanwezigheid van een specifiek additief aan te tonen. HPLC daarentegen is een kwantitatieve methode. Deze drie methoden worden behandeld in een tweede deel.

Hierna volgt een theoretische bespreking over het verwerken van grote datasets. Verschillende multivariate data-analyse methoden worden opgesomd en kort besproken. De geschiktheid voor het verwerken van de gegeven dataset zal worden nagegaan. Uiteindelijk wordt er toegelegd op de deep learning, een techniek binnen het veld van artificiële intelligentie.

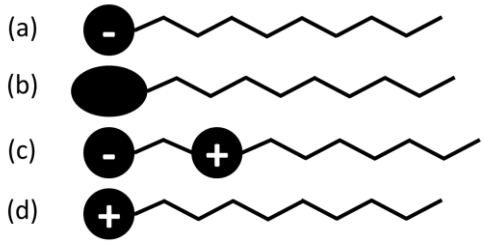
2.2 Additieven voor textiel

Textiel kan behandeld worden met verschillende additieven om de eigenschappen ervan te verbeteren. Zo wordt textiel gewassen met detergenten om vuil te verwijderen. Tijdens de wasbeurt kunnen wasverzachters worden toegevoegd om het textiel aangenamer te laten aanvoelen. Optische witmakers, *bluing agents* of bleekmiddel kunnen dan weer worden toegevoegd om het textiel er extra wit te laten uitzien. In wat volgt wordt de werking van enkele van deze additieven besproken.

2.2.1 Oppervlakte actieve stoffen

Oppervlakte actieve stoffen (OAS) of surfactants zijn organische moleculen die bestaan uit een polaire hydrofiele kop en een apolaire hydrofobe staart. Als gevolg van die structuur hebben ze de neiging om te absorberen aan het scheidingsoppervlak tussen twee fasen, wat zorgt voor een verlaging van de oppervlaktespanning van de twee fasen. De hydrofobe staart van de surfactants bestaat doorgaans uit een lange koolstofketen met op zijn minst acht koolstoffen.

Naargelang de moleculaire structuur en de lading van het hydrofiele deel worden ze onderverdeeld in vier grote groepen namelijk anionogene surfactants, kationogene surfactants, niet-ionogene surfactants en amfotere surfactants. Aan elke groep kunnen enkele specifieke eigenschappen toegeschreven worden, waardoor elke groep gebruikt kan worden voor verschillende toepassingen. In Figuur 1 wordt er van elke groep een schematisch weergave gegeven. (2)(3)(4)



Figuur 1: Schematische voorstelling van (a) Anionogene surfactant.
(b) Niet-ionogene surfactants. (c) Amfotere surfactants. (d) Kationogene surfactants.

2.2.1.1 Anionogene surfactants

Anionogene surfactants zijn de meest gebruikte oppervlakte actieve stoffen. Het zijn ook deze surfactants die het beste klei, modder en olievlekken van textiel of andere oppervlakken kunnen verwijderen. Ze hebben dus vooral hun toepassing in wasmiddelen, afwasmiddelen en shampoos. Ze bevatten in hun hydrofiele groep een negatieve lading die afkomstig kan zijn van een hydroxyl-, sulfaat-, sulfonaat of fosfaatgroep. In Figuur 2 wordt een specifiek voorbeeld van een anionogene surfactant weergegeven, hierin is de negatieve lading afkomstig van een sulfaatgroep. (4)(5)

Figuur 2: N-dodecyl sulfate. (3)

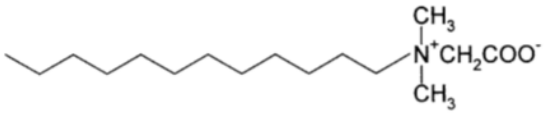
2.2.1.2 Niet-ionogene surfactants

Niet-ionogene surfactants hebben geen lading in het hydrofiele deel, waardoor ze in waterig midden niet kunnen dissociëren in ionen. De belangrijkste niet-ionogene surfactants zijn de ethoxylaten, daarnaast zijn er ook nog verschillende andere soorten. Door de grote diversiteit van hun chemische structuur kunnen ze gebruikt worden voor heel veel verschillende toepassingen. In Figuur 3 wordt een specifiek voorbeeld van een ethoxylaat weergegeven. (4)(5)(6)

Figuur 3: Dodecanol 9-mole ethoxylate. (3)

2.2.1.3 Amfotere surfactants

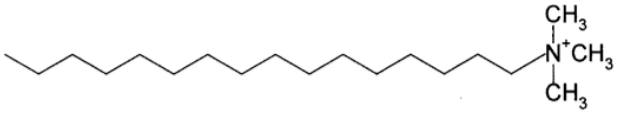
Amfotere surfactants hebben een zwitter-ion in hun hydrofiele groep. Dit wil zeggen dat er zowel een positieve als een negatieve lading in het hydrofiele deel aanwezig is. Deze groep van surfactanten wordt het minst gebruikt en zou wereldwijd maar voor 1 % van de totale consumptie van surfactants instaan. In Figuur 4 wordt een specifiek voorbeeld gegeven van een amfoteer surfactant. (5)(6)



Figuur 4: Dodecyldimethylammononiomethane carboxylate. (3)

2.2.1.4 Kationogene surfactants

Kationogene surfactants bezitten een positief geladen ion in hun hydrofiele groep, waardoor ze makkelijk absorberen aan negatief geladen oppervlakken. De negatieve lading is meestal een quaternair ammonium kation. Ze worden vooral gebruikt voor wasverzachters. Hier wordt in de volgende paragraaf dieper op ingegaan. In Figuur 5 wordt een specifiek voorbeeld van een kationogene surfactant met een quaternair ammonium kation weergegeven. (4)(5)(6)



Figuur 5: Af-Hexadecyltrimethylammononium. (3)

Wasverzachters zijn oppervlakte actieve stoffen die tijdens de spoelcyclus van een was proces aan de was worden toegevoegd om de was zachter te laten aanvoelen. Ze zorgen voor een beter gevoel wanneer de stof over de huid wrijft. Daarnaast bieden wasverzachters ook nog een verscheidenheid aan andere gewenste eigenschappen aan het textiel. Zo zorgen ze onder meer voor een frisse geur, een vermindering van de droogtijd en een vermindering van statische elektriciteit. Ze verminderen bovendien de hoeveelheid rimpels in kledingstukken en maakt hen gemakkelijker te strijken. (7)

Verschillende klassen van chemicaliën kunnen enige mate van wasverzachting veroorzaken. De meest gebruikte wasverzachters zijn echter kationogene oppervlakte actieve stoffen. Ze bestaan meestal uit twee vetketens die elk 16 tot 18 koolstofatomen bevatten, met daartussen de kationogene lading. Deze kationogene lading neutraliseert de nog aan het wasgoed klevende anionogene oppervlakte-actieve stof uit het wasmiddel, waardoor de statische elektriciteit verminderd wordt. De lange koolstofketens die naar de buitenkant gericht zijn zorgen dan weer voor het zachte gevoel. (7)(8)

2.2.2 Optische witmakers

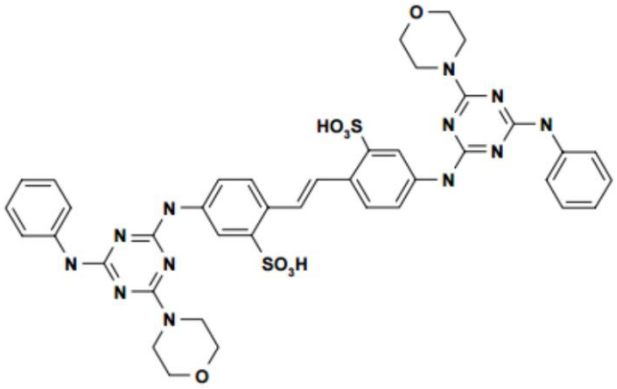
Optische witmakers worden aan wasproducten toegevoegd om de schijnbare witheid van textiel te verhogen. Het woord ‘schijnbaar’ wordt gebruikt omdat ze omwille van een optisch effect het textiel witter laten lijken dan dat het eigenlijk is. Dit effect wordt uitgeoefend doordat optische witmakers UV-straling (golflengtes van 330 nm tot 360 nm) uit het daglicht absorberen, vervolgens geven ze het grootste deel van de geabsorbeerde energie opnieuw af in de vorm van zichtbaar blauw licht (golflengtes tussen 400 en 500 nm). Dit zorgt voor een

verhoging van de reflectie van het textiel in het blauwe deel van het zichtbaar spectrum. De blauwe toon in wit licht geeft een gevoel van verhoging van witheid, waardoor het textiel dus witter lijkt. Optische witmakers zijn typisch stilbeen gebaseerde componenten en hun derivaten. (9)(10)

Twee belangrijke types stilbeen gebaseerde optische witmakers zijn de diamino stilbeen derivaten (DAS) en de distyryl bifenyl derivaten (DSBP). Beiden worden geproduceerd door BASF onder de naam Tinopal.

2.2.2.1 Diamino stilbeen derivaten (DAS)

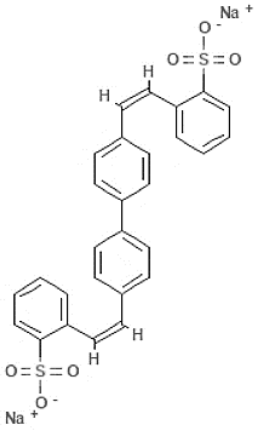
Tinopal DMA-X is een voorbeeld van een diamino stilbeen derivaat (Disodium 4,4'-bis[(4-anilino-6-morpholino1,3,5-triazin-2-yl) amino] stilbene -2,2'-disulphonate). Het product zorgt bij cellulose vezels voor een hoge witheid en helderheid bij een koude tot gemiddelde was temperatuur, zelfs bij een korte was tijd. Tinopal DMA-X is stabiel in zowel alkalisch als zuur milieu en dus ook in bleekmiddelen zoals hypochloriet en peroxide. De stof heeft een molaire massa van een 924,91 g/mol, een dichtheid van 1,54 g/cm^3 en een pH-waarde van 7 tot 9,5 bij een 1 g/l oplossing. De oplosbaarheid van dit product bij 20 °C bedraagt 2 g/L. De structuur van Tinopal DMA-X wordt weergegeven in Figuur 6. (11) (12)(13)



Figuur 6: Chemische structuur van Tinopal DMA-X. (12)

2.2.2.2 Distyryl bifenyl derivaten (DSBP)

Tinopal CBS-X is een voorbeeld van een anionionogeen distyryl bifenyl derivaat (Disodium 4,4'-bis(2-sulfostyryl) biphenyl). Deze optische witmaker wordt ook in poedervorm door BASF aangeboden. De stof heeft een hoge oplosbaarheid (25 g/L water bij 30 °C) en geeft uitstekende witheidseffecten op cellulose materiaal, dit vooral bij koude tot gemiddelde was temperaturen. De stof is net zoals Tinopal CBS-X stabiel in zowel alkalisch als zuur milieu en dus ook in bleekmiddelen zoals hypochloriet en peroxide. Tinopal CBS-X heeft een molaire massa van 562,6 g/mol, een dichtheid van 1,49 g/cm^3 en een pH-waarde van 7 tot 8 bij een 1 g/l oplossing. De chemische structuur van Tinopal CBS-X wordt weergegeven in Figuur 7. (13)(14)



Figuur 7: Chemische structuur van Tinopal CBS-X. (15)

2.3 Technieken voor het analyseren van textiel

Voor het analyseren van textiel zijn verschillende technieken nodig. Allereerst moet bepaald worden om welk textiel het gaat. Dit kan aan de hand van kwalitatieve technieken zoals bijvoorbeeld infraroodspectroscopie. Kwalitatieve methoden maken het ook mogelijk om aan te tonen welke producten verder nog aanwezig zijn op het textiel. Zo kan het belangrijk zijn om de aard van bepaalde vlekken op het textiel te kennen, om dan te onderzoeken op welke manier ze verwijderd kunnen worden. Daarnaast zijn ook kwantitatieve methoden nodig om de hoeveelheid van de aanwezige componenten te achterhalen. Een geschikte techniek hiervoor is HPLC.

Naast de kwalitatieve en kwantitatieve analyses worden ook nog andere analyses uitgevoerd. Zo wordt de kleur van textiel bepaald met behulp van een spectrofotometer. Deze spectrofotometer is ook in staat om de witheid van textiel in absolute waarde weer te geven. Bovendien kan met deze techniek worden bepaald of er een optische witmaker aanwezig is op het textiel, waardoor deze techniek ook voor een stuk kwalitatieve info over het textiel kan geven.

2.3.1 Spectrofotometer voor witheidsmetingen

Een methode die veel wordt gebruikt bij de analyse van wit textiel is het meten van de witheid. De witheid van het textiel is een belangrijke factor. Het is bijvoorbeeld niet de bedoeling dat onze stralend witte bloezen, na enkele wasbeurten al grijs en grauw zijn of dat de handdoeken in luxehotels een gelige schijn hebben. Als het textiel echt grijs of geel is, is het meestal al te laat. Daarom worden witheidsmetingen wascyclus na wascyslus steekproefsgewijs uitgevoerd. Een spectrofotometer kan namelijk al veel sneller dalingen in witheid waarnemen dan het menselijke oog, daarenboven wordt er met deze spectrofotometer een absolute waarde om witheid uit te drukken verkregen. Een witheidsmeting is dus niet enkel een momentopname maar kan eigenlijk de volledige historie van het textiel weergeven.

In de volgende paragrafen wordt uitgelegd hoe het meten van de witheid van textiel met behulp van een spectrofotometer in zijn werk gaat. Vooraleer de werking van de meting wordt besproken, moet eerst gekeken worden naar de manier waarop de witheid van het textiel wordt uitgedrukt. Dit gebeurt aan de hand van kleurenschalen waarmee een kleur op een subjectieve manier kan worden uitgedrukt.

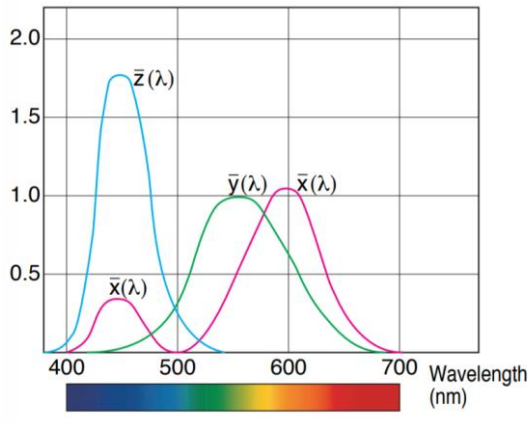
2.3.1.1 Kleurschalen

Kleur ontstaat wanneer bepaalde golflengtes, uitgezonden door het ingestraalde licht, worden geabsorbeerd, waardoor ze verdwijnen uit het spectrum van de lichtstraal. Enkel de stralen met golflengtes die niet worden geabsorbeerd bereiken onze ogen. Deze golflengtes worden waargenomen als een specifieke kleur. De beschrijving van een kleur hangt sterk af van de waarnemer en zijn subjectieve interpretatie. Daarnaast is het moeilijk om verbaal de kenmerken van een kleur te beschrijven. Om die redenen is het beschrijven van een bepaalde kleur buitengewoon moeilijk en vaag.

Om de kleurencommunicatie te vergemakkelijken werden er manieren ontwikkeld om de kleur dat we met onze ogen ervaren uit te drukken in absolute waarden. Kleuren en tinten kunnen hierdoor door iedereen nauwkeurig en eenduidig beschreven of begrepen worden. In 1931 werd er voor het eerst een gestandaardiseerd systeem ontwikkeld voor de kwantificering van kleur door de International Commission on Illumination (CIE). CIE ontwikkelde meerdere schalen waarmee een kleur wiskundig gedefinieerd kan worden met behulp van 3 variabelen. (16)(17)(18)(19)

XYZ-schaal

De eerste schaal was de zogenaamde tristimulus schaal, waarbij kleuren worden uitgedrukt in termen van X-, Y- en Z-coördinaten. Deze schaal is gebaseerd op de drie-componenten theorie, die stelt dat het oog receptoren bezit voor drie primaire kleuren (rood, groen en blauw). Alle kleuren worden waargenomen als een mengsel van deze drie primaire kleuren. De XYZ-tristimuluswaarden worden berekend met behulp van de kleur-matching functies $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ en $\bar{z}(\lambda)$, weergegeven in Figuur 8. (17)(19)



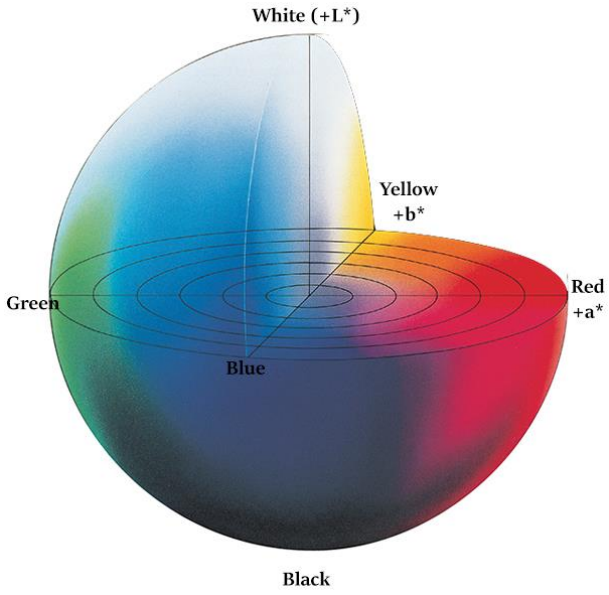
Figuur 8: Kleur matching functies $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ en $\bar{z}(\lambda)$. (19)

In deze figuur is te zien dat $\bar{x}(\lambda)$ een hoge respons heeft in het rode gebied, $\bar{y}(\lambda)$ heeft een hoge respons in het groene gebied en $\bar{z}(\lambda)$ in het blauwe gebied. Door verschillende verhoudingen van $\bar{x}(\lambda)$, $\bar{y}(\lambda)$ en $\bar{z}(\lambda)$ in het licht dat door een bepaald object wordt gereflecteerd naar de ogen, wordt een specifieke kleur waargenomen. (21)(19)

De berekening van de XYZ-tristimulus-waarden met behulp van de drie kleur-matching functies gebeurt als volgt. Het object waarvan de kleur bepaald wordt reflecteert licht met een bepaalde spectrale distributie. Dit licht valt in op een sensor waarvan de filters het licht verdelen in golflengtegebieden die overeenkomen met de drie primaire kleuren. Door deze gebieden te vermenigvuldigen met hun kleur-matching functies en de integraal te nemen van die nieuwe functies worden de X-, Y- en Z-waarden bepaald. De Z-waarde representeert dus het deel van het spectrum van ongeveer 400 nm tot 500 nm en is dus een maat voor blauwheid. De Y-waarde geeft het deel van het spectrum tussen 500 nm en 600 nm weer, wat de groenheid aangeeft. De X-waarde staat voor het spectrum tussen 600 nm en 740 nm, wat de rode kleur representeert. (21)

L*a*b* schaal

De X, Y en Z geven een eenduidige waarde aan alle verschillende kleuren, maar ze blijven moeilijk om te visualiseren. Wanneer de X-, Y- en Z-waarden gegeven zijn, kan nog niet direct een beeld gevormd worden over de exacte kleur van het voorwerp. Om dit op te lossen werd in 1976 een nieuwe schaal gedefinieerd door de CIE namelijk de L*a*b* schaal of de CIELAB. Tegenwoordig is dit een van de meest gebruikte kleurschalen. Bij het CIELAB-systeem wordt elke kleur beschreven aan de hand van zijn cartesiaanse coördinaten L*, a* en b*. Er wordt een 3D-kleurruimte gecreëerd waarmee men kleuren wiskundig kan karakteriseren en ruimtelijk kan weergeven. L* geeft de helderheid aan en wordt de helderheids-as of grijsschaal genoemd, a* en b* geven de kleur weer. In Figuur 9 wordt een representatie gegeven van de 3D-kleurruimte van het CIELAB-systeem.



Figuur 9: 3D-kleurruimte van CIELAB-systeem. (19)

De grijsschaal (L*) gaat van 0 tot +100. Bij een L*-waarde van 0 gaat het om een volledig zwart object, wanneer de L*-waarde +100 is, is het object wit. Bij alle L*-waarden daartussen bevat het object kleur. Deze kleur wordt gespecifieerd aan de hand van de a*-as en de b*-as. a* is de rood-groene component, hoe negatiever de a*-waarde hoe groener de kleur, hoe positiever de a*-waarde, hoe roder. b* is de blauw-gele component, het object heeft een blauwe kleur bij negatieve waarden van b* en een gele kleur bij positieve waarden van b*. Wanneer de L*, a* en b* waarde gekend zijn kan onmiddellijk een idee gevormd worden van de werkelijke kleur van het object. (19)

De L*, a*, b* -coördinaten kunnen met volgende formules op een eenvoudige manier berekend worden. Deze vergelijkingen maken gebruik van de X-, Y-, en Z-waarden die al eerder besproken werden.

$$L^* = 116\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 \tag{2.1}$$

$$a^* = 500\left[\left(\frac{X}{X_n}\right)^{\frac{1}{3}} - \left(\frac{Y}{Y_n}\right)^{\frac{1}{3}}\right] \tag{2.2}$$

$$b^* = 200\left[\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - \left(\frac{Z}{Z_n}\right)^{\frac{1}{3}}\right] \tag{2.3}$$

In deze formules staan X, Y en Z voor de tristimulus waarde van het object. X_n, Y_n en Z_n zijn de tristimulus waarde van een perfect reflecterende diffusor, deze laatste zijn dus constante waarden. (19)

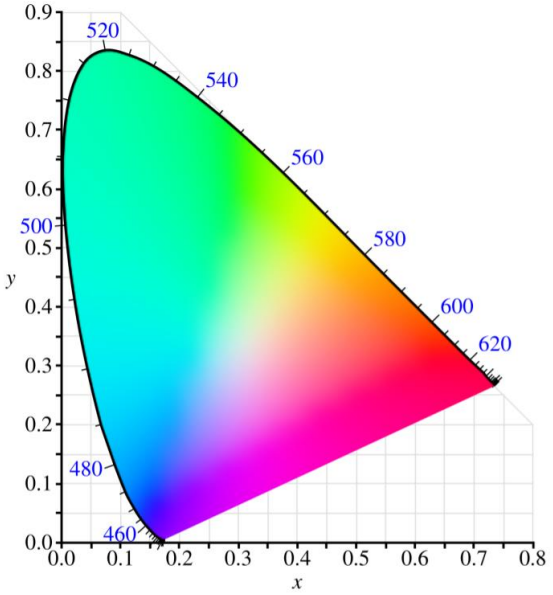
Yxy schaal

Bij de Yxy schaal wordt de Y tristimulus waarde gebruikt om de helderheid weer te geven. De kleur wordt aangegeven door de x en y coördinaten. Deze twee waarden zijn in functie van de drie Y tristimulus waarde en worden berekend met volgende formules. (20)(19)

$$x = \frac{X}{X+Y+Z} \tag{2.4}$$

$$y = \frac{Y}{X+Y+Z} \tag{2.5}$$

In Figuur 10 wordt het CIE x-y chromaticiteitsdiagram getoond. Elke kleur kan in dit diagram met een x en y waarde worden gedefinieerd.

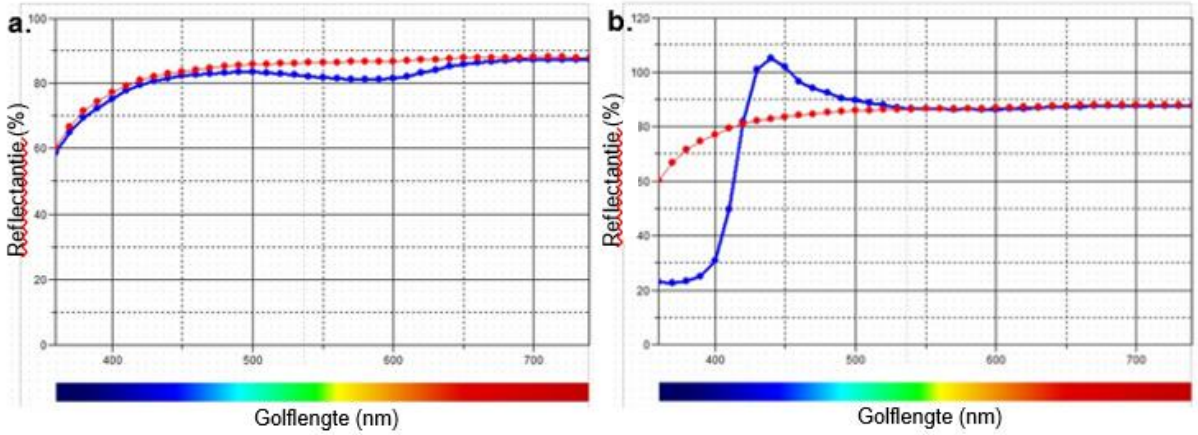


Figuur 10: CIE x-y chromaticiteitsdiagram. (20)

2.3.1.2 Witheid

Voor de beschrijving van de witheid van bepaalde objecten zoals textiel of papier werden op basis van de eerder vermelde kleurenschalen verschillende formules ontworpen. In eerste instantie kan de witheid gekwantificeerd worden met behulp van de Y- en de L*-waarde afkomstig van respectievelijk de XYZ-schaal en de L*a*b*-schaal, beschreven in bovenstaande paragrafen. Deze twee waarden geven de basiswitheid van het textiel (of andere objecten) aan.

Vooraf werd al beschreven dat de L*-waarde de helderheid van het object weergeeft. Een hoge L*-waarde geeft dus een hoge basiswitheid aan. Wanneer teruggekeken wordt naar vergelijking 2.1 voor het bereken van deze L*-waarde, blijkt dat deze enkel afhankelijk is van de Y-waarde. Een object zal dus een hoge L*-waarde hebben als gevolg van een hoge Y-waarde. Op die manier is de Y-waarde een maat voor de basiswitheid van objecten zoals textiel en papier. De Y-waarde geeft echter geen volledig beeld van de witheid van een object. Voor textiel bijvoorbeeld bestaan er verschillende hulpmiddelen om het textiel er witter te laten uitzien zonder dat de Y-waarde hierbij zal stijgen. Een voorbeeld hiervan zijn de optische witmakers. Zoals eerder uitgelegd absorberen deze stoffen Uv-straling uit het daglicht, waarna ze deze energie terug afgeven in de vorm van zichtbaar blauw licht. De blauwe toon in wit licht geeft een gevoel van witter dan wit, waardoor het textiel dus witter lijkt zonder dat de basiswitheid van het textiel verandert. Een tweede voorbeeld van zo een hulpmiddel zijn de bluing-agents. Dit zijn blauwe kleurstoffen die worden toegevoegd aan het textiel om zo de gele ondertoon van textiel te camoufleren. In Figuur 11 worden spectra afgebeeld met de aanwezigheid van een bluing-agent en een optische witmaker. (16)



Figuur 11: Vergelijking van spectrum zonder hulpmiddelen en spectrum in aanwezigheid van (a) Bluing agent en (b) Optische witmaker. (16)

De rode curves in Figuur 11 (a) en (b) geven het spectrum van een wit textiel zonder bluing agent of optische witmaker. De Y-waarde op deze spectra wordt afgelezen tussen 500 nm en 600 nm. In Figuur 11 (a) is duidelijk te zien dat de blauwe curve (met de aanwezigheid van een bluing-agent) in dit gebied lager is dan de rode curve (zonder bluing-agent). Dit betekent dus dat de Y-waarde en dus ook de basiswitheid van textiel zonder bluing-agent hoger is, hoewel dat de bluing-agent net zorgt voor een witter uitzicht.

In Figuur 11 (b) wordt de werking van een optische witmaker duidelijk weergegeven. De golflengtes lager dan 400 nm worden geabsorbeerd en extra energie wordt geëmitteerd als zichtbaar blauw licht. Op die manier ontstaat zelfs een reflectie van meer dan 100%. Uit deze figuur blijkt ook dat de basiswitheid van textiel onafhankelijk is van de aanwezigheid van een

optische witmaker, terwijl het textiel er visueel duidelijk witter uitziet in aanwezigheid van een optische witmaker. De Y-waarde alleen geeft dus niet altijd een juist beeld over de witheid van een textiel.

In een poging om de witheid die werkelijk ervaren wordt te kunnen karakteriseren met behulp van één getal werden meerdere witheid indexen (WI) gedefinieerd. Sommigen worden nu veel gebruikt in de industrie zoals de CIE-witheidindex en de Ganz witheidindex. (21)

De meest gebruikte witheidindex is de WI(CIE). Wanneer gewerkt wordt met de X-, Y- en Z-waarde wordt deze witheidindex berekend door volgende formule.

$$WI_{CIE-XYZ} = Y + 800(x_n - x) + 1700(y_n - y) \quad (2.6)$$

Hierin staat Y voor de Y tristimulus -waarde en x en y zijn de chromaticiteitscoördinaten van het monster in het x, y chromaticiteitsdiagram van CIE. De x_n en y_n zijn de chromaticiteitscoördinaten van een perfect reflecterende diffusor. Door over te schakelen op L*a*b* waarden wordt de volgende formule bekomen. (21)

$$WI_{CIE-Lab} = 2{,}41.L^* - 4{,}45.b^*.\big(1 - 0{,}0090 * (L^* - 96)\big) - 141{,}4 \quad (2.7)$$

Uit deze vergelijking blijkt dat de witheid van textiel bepaald wordt door de L* (de helderheid) en de b* (de blauw/gele kleur). Door deze twee parameters samen in rekening te brengen kan de werkelijke witheid van textiel in één getal gekarakteriseerd worden. (16)(22)

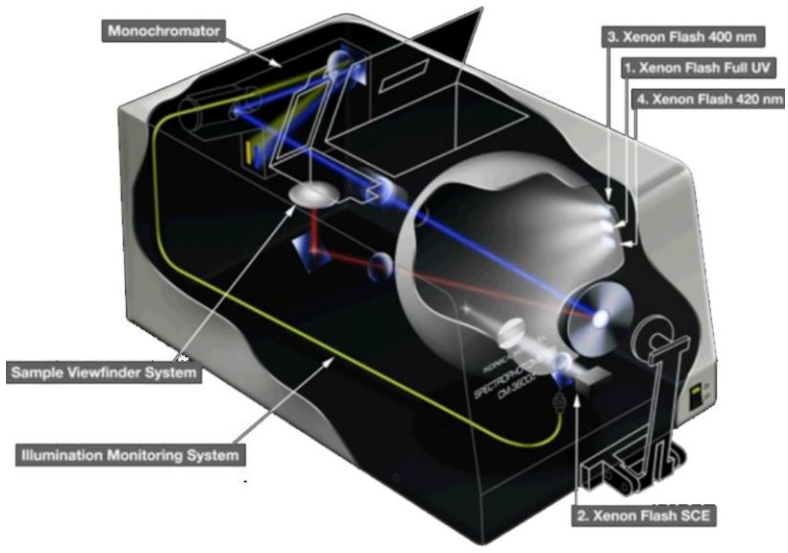
De GanzGriesser-witheidsmetriek (WI_{Ganz}) wordt bepaald met volgende formule.

$$\mathrm{WI_{Ganz}} = (DY) + (Px) + (Qy) + C$$

Y is ook hier weer de Y tristimulus-waarde en x en y de chromaticiteitscoördinaten van het monster in het x, y chromaticiteitsdiagram van CIE. D, P, Q en C zijn parameters die bepaald worden door de GanzGriesser-kalibratiemethode en zijn specifiek voor een bepaald instrument. (21)

2.3.1.3 Spectrofotometer

Spectrofotometers zijn erg geschikt voor het meten van kleuren. Ze kunnen kleuren zowel numeriek uitdrukken als via een spectrale reflectiegrafiek. Aan de hand van Figuur 12 wordt de werking van zo een spectrofotometer voor het meten van de kleur van textiel uitgelegd.



Figuur 12: Schematische voorstelling van de spectrofotometer. (16)

Bij deze spectrofotometer wordt als lichtbron een Xenonlamp gebruikt. Deze zendt licht over het volledig spectrum uit naar een integrerende sfeer. De sfeer is aan de binnenkant gecoat met een wit materiaal zoals bariumsulfaat, zodat het binnenkomend licht gelijkmatig wordt verspreid. In de sfeer is een opening voorzien waartegen het textiel kan bevestigd worden.

Textiel is geen glad oppervlak, wanneer er licht opvalt zal er dus diffusieve reflectie optreden. Het licht dat gereflecteerd wordt zal in alle richtingen terug stralen, wat het moeilijk maakt om het totale gereflecteerde licht te meten. Met behulp van de witte sfeer wordt al het gereflecteerde licht gecapteerd, waardoor het meten ervan wel mogelijk wordt. Het gereflecteerde licht wordt naar een monochromator gestuurd, waar de lichtbundel wordt gesplitst naar kleine bundels met een specifieke golflengte. De lichtbundels vallen vervolgens in op een receptor die verschillende fotodiodes bevat, elk gevoelig voor een specifieke golflengte. De fotodiodes genereren een stroom op het moment dat er een foton opkomt en op die manier wordt er een spectrum gevormd. (16)(19)

2.3.2 Fourier-Transformatie infraroodspectroscopie

Verzwakte totale reflectie Fourier-transformatie-infraroodspectroscopie (ATR-FTIR) is een techniek die snel en gemakkelijk karakteristieke informatie over organische componenten en dus ook over textiel geeft. Ook maakt deze techniek het mogelijk om de aard van aanwezige vlekken op het textiel te identificeren of om de aanwezigheid van bepaalde additieven te controleren. (23)

2.3.2.1 Infraroodspectroscopie

Infrarood (IR) spectroscopie is een veelgebruikte techniek waarbij het IR-gebied van elektromagnetische straling gebruikt wordt voor de bepaling en de identificatie van de moleculaire structuur van bepaalde materie. In wat volgt wordt uitgelegd hoe deze identificatie precies in zijn werk gaat. (24)

Elektromagnetische straling vertoont een dualistisch karakter. Het kan enerzijds beschreven worden als een golf en anderzijds als een deeltje. De golftheorie van elektromagnetische straling bestudeert de stralengang door verschillende media, terwijl de deeltjestheorie de interactie van de straling met bepaalde materie behandelt. Volgens de golftheorie wordt de relatie tussen de voortplantingssnelheid c, de golflengte λ, golfgetal $\bar{v}$ en de frequentie ν gegeven door volgende formule.

$$c = \lambda.\nu = \frac{\nu}{\bar{\nu}} \tag{2.8}$$

De deeltjestheorie geeft aan dat elektromagnetische straling bestaat uit een stroom van fotonen (energiepakketjes) die een bepaalde hoeveelheid energie E bevatten, afhankelijk van de frequentie v.

$$E = h.\nu \tag{2.9}$$

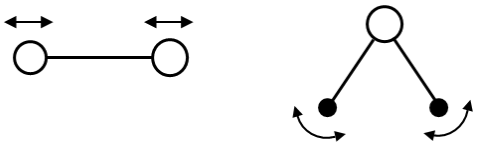
Met h = constante van Planck: $6{,}625.10^{-34}$ J.s.

Vergelijking 2.9 geeft aan dat energie niet continu is, maar enkel discrete waarde kan aannemen. (24)

Ook een molecule bevat een bepaalde energie. Net zoals bij elektromagnetische energie is deze discontinu, waardoor de molecule slechts kan voorkomen in bepaalde discrete energietoestanden. Een molecule kan verschillende soorten energie bezitten, namelijk

rotationele energie, vibrationele energie, elektronische energie en translationele energie. Voor IR-spectroscopie is vooral de vibrationele energie belangrijk. Deze energie is het gevolg van periodieke verplaatsing van de atomen ten opzichte van elkaar, vanuit de evenwichtspositie.

Bindingen in organische moleculen kunnen twee soorten vibraties ondergaan, namelijk rek- en deformatie vibraties. Deze worden weergegeven in Figuur 13.



Figuur 13: Rek- en deformatie vibraties in organische moleculen. (25)

Bij een rekvibratie verandert de bindingslengte, terwijl bij een deformatie vibratie de bindingshoek tussen twee atomen verandert. Wanneer deze vibraties een verandering van het elektrisch dipoolmoment van de moleculen veroorzaken is er interactie mogelijk met het fluctuerend elektrisch veld van de elektromagnetische straling. Elke rek- en deformatievibratie van een specifieke binding treedt op bij een karakteristieke frequentie (de eigen trillingsfrequentie). Enkel wanneer een molecule bestraald wordt met een elektromagnetische straal die exact dezelfde frequentie bevat, absorbeert de molecule de bijbehorende energie. Na deze absorptie wordt er een vermindering in intensiteit van de elektromagnetische lichtstaal waargenomen voor die welbepaalde frequentie. Door de golfgetallen van de elektromagnetische energie die door de molecule geabsorbeerd werd te analyseren, kan achterhaald worden welke type bindingen er aanwezig zijn in de molecule. (24)(26)(27)

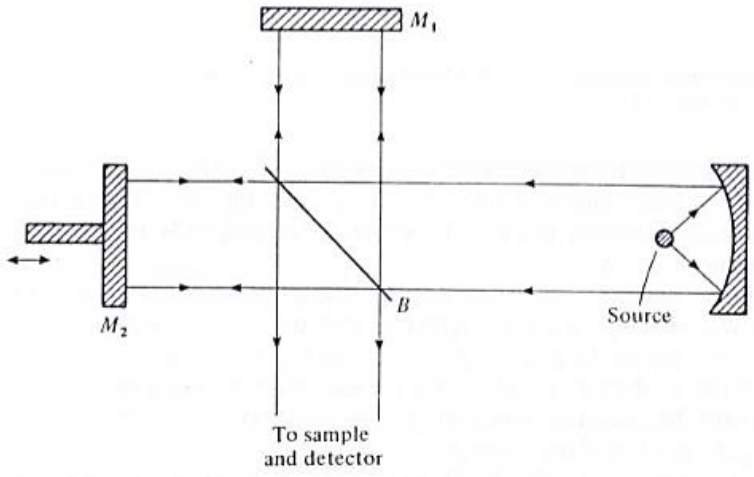
Voor deze techniek wordt gebruik gemaakt van IR-stralen aangezien de frequenties die nodig zijn voor de overgang van de ene vibrationele toestand naar de volgende, goed overeenkomen met de frequenties van het IR-gebied, meer bepaald met de frequenties van het midden IR-gebied. In Tabel 1 worden de eigenschappen van de domeinen binnen het IR-gebied weergegeven. (24) (26)

Tabel 1: Golflengten, frequenties, golfgetallen, foton energie en respons van IR-gebied.

Stimulans	λ (µm)	ν (s^{-1})	$\bar{\nu}$ (cm^{-1})	Foton energie (J)	Respons
Nabij IR	0,78 – 2,5	$3{,}8.10^{14}$ - $1{,}2.10^{14}$	12820 - 4000	1,56 – 0,5	Boventonen van vibraties
Midden IR	2,5 – 50	$1{,}2.10^{14}$ - $6{,}0.10^{12}$	4000 – 200	0,5 – $2.5.10^{-2}$	Vibratie
Verre IR	50 – 1000	$6{,}0.10^{12}$ - $3{,}0.10^{11}$	200 – 10	$2{,}5.10^{-2}$ – $1{,}24.10^{-1}$	Moleculaire rotaties

2.3.2.2 Fourier-Transformatie IR-spectroscopie

Bij Fourier Transform (FT) spectroscopie wordt gebruik gemaakt van Fourier Transformaties. Dit is een wiskundige techniek die een functie ontbindt in een continu spectrum van frequenties. Deze techniek heeft als groot voordeel dat alle golflengten tezamen gemeten worden, waardoor op elk moment een volledig spectrum verkregen wordt. Dit in tegenstelling tot de oude spectrometers, waarbij steeds het hele frequentiedomein, frequentie per frequentie moest doorlopen worden. De werking van een FT-IR-spectrometer wordt besproken aan de hand van Figuur 14, waarin een schematische weergave van de FT-IR-spectrometer wordt afgebeeld.
(26) (27)



Figuur 14: Schematische voorstelling van de FT-IR-spectrometer. (25)

Vanuit de bron wordt een evenwijdige bundel IR-stralen door een interferometer gestuurd. Deze bestaat uit een *beam splitter* (B) en twee spiegels (M_1 en M_2). De Beam splitter bestaat uit een dunne film germanium, gecoat op een plaatje van kaliumbromide of cesiumjodide. 50 % van de invallende IR-straling wordt door de beam splitter doorgelaten en de andere 50 % wordt door de beam splitter gereflecteerd. Door de pijlen in Figuur 14 te volgen wordt duidelijk wat er verder met deze twee bundels gebeurt. De bundel die wordt doorgelaten gaat naar de beweegbare spiegel M_2, waardoor ze volledig terug wordt gereflecteerd en weer bij de beam splitter terecht komt. Hier zal opnieuw 50 % van deze lichtbundel worden doorgelaten en dus teruggestuurd naar de bron, terwijl de andere 50 % wordt gereflecteerd in de richting van het staal. Het deel van de oorspronkelijke lichtbundel dat wordt gereflecteerd door de beam splitter, wordt door een vaste spiegel M_1 eveneens teruggestuurd naar de beam splitter. Ook hier zal 50 % terug keren naar de bron en de andere 50 % doorgaan naar het staal.

De twee lichtbundels die naar het staal worden gestuurd zullen ofwel constructief ofwel destructief met elkaar interfereren, afhankelijk van het optisch weglengteverschil tussen de twee stralen. Het optische weglengte verschil tussen deze twee lichtbundels is gelijk aan tweemaal de verplaatsing van de spiegel M_2. Wanneer het optisch weglengteverschil gelijk is aan 0 of een geheel aantal keer de golflengte, dan zijn de twee bundels exact in fase en zullen ze constructief interfereren. Dit geeft een maximum detectorsignaal. De twee bundels zullen destructief interfereren met elkaar wanneer het optisch weglengteverschil gelijk is aan een geheel aantal keer de helft van de golflengte. Door de spiegel M_2 met een constante snelheid te laten verplaatsten, varieert de signaalintensiteit dus als een oneindig lange cosinusfunctie. (26) (27)(28)

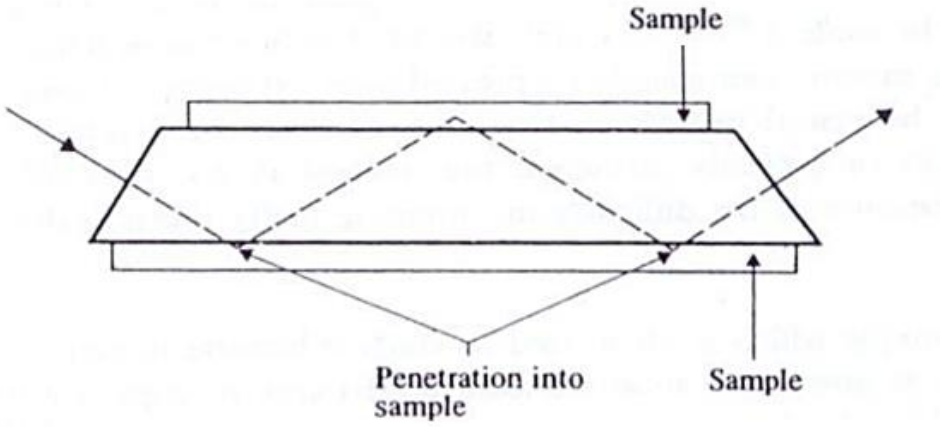
De lichtbundel die het staal bereikt zal interageren met de materie van het staal. Een deel van de bundel wordt gereflecteerd, het andere deel zal door het staal binnen dringen. De materie zal een deel van de binnendringende stralen absorberen, zoals uitgelegd in paragraaf 2.4.2.1. Het deel van de bundel dat volledig door het staal gaat (het deel dat niet gereflecteerd en niet geabsorbeerd wordt) bevat de moleculaire informatie van de materie. Deze bundel wordt verder naar de detector geleid waar het wordt omgezet naar een elektrisch signaal. Dit levert een interferogram op. Dit is een ruw signaal, dat de lichtintensiteit weergeeft als functie van de positie van een spiegel. Dit interferogram wordt vervolgens via Fourier-Transformatie omgezet naar een spectrum dat de intensiteit geeft in functie van het golfgetal.

Het verkregen spectrum is te vergelijken met een vingerafdruk. De meeste stoffen vertonen een karakteristiek spectrum waardoor ze met deze techniek kunnen geïdentificeerd worden. Daarnaast is het met deze techniek ook mogelijk om functionele groepen zoals bijvoorbeeld C = O, C-H of N-H te identificeren. (25)(26)(27) (28)

2.3.2.3 Attenuated total reflection

ATR (attenuated total reflection) is één van de verschillende opnametechnieken voor FT-IR-spectroscopie. Het grote voordeel van deze techniek is dat ze minder tijdrovend en omslachtig is dan andere opnametechnieken. In Figuur 15 wordt de ATR-techniek voorgesteld.

.



Figuur 15: Schematische voorstelling van de ATR-techniek. (25)

Bij de ATR-techniek wordt er een kleine hoeveelheid van het staal aangebracht op het zogenaamde ATR-kristal. Dit kristal bestaat uit een bepaald IR-transparant materiaal, bijvoorbeeld diamant, Zink Selenide of germanium. De brekingsindex van het ATR-kristal is steeds hoger dan de brekingsindex van het staal.

De infraroodstraal komt het ATR-kristal binnen onder een hoek die typisch 45 ° is en wordt volledig weerspiegeld aan het interfasevlak van het kristal en het staal. Vanwege het golfkarakter van het licht zal de IR-straal niet direct gereflecteerd worden aan het grensvlak, maar zal het enigszins doordringen in het optisch minder dichte staal (Goos-Hänchen effect). De fractie van de lichtgolf die in het monster reikt wordt de vluchtende golf genoemd. De penetratiediepte is afhankelijk van de golflengte, de brekingsindex van ATR-kristal, het staal en de hoek van de invallende lichtstraal. Typisch bedraagt deze enkele microns (ca. 0,5 - 3 µm). In die spectrale regio's waar het staal energie absorbeert, wordt de vluchtende golf verzwakt. Doordat de straal 10 tot 20 keer gereflecteerd wordt, wordt het effect versterkt. Na een meerdere interne reflecties verlaat de IR-straal het ATR-kristal en wordt het naar de IR-detector gestuurd.

De grootste voordelen van deze techniek zijn de snellere bemonstering zonder voorbereiding, de uitstekende reproduceerbaarheid van monster tot monster en de minimale door de operator veroorzaakte variaties. (25)(26)(27)

2.3.3 High performance liquid chromatography

Door een deel van het waswater tijdens het was proces van het textiel op te vangen en te analyseren met high performance liquid chromatography kan een kwantitatieve analyse worden uitgevoerd voor de aanwezige additieven op het textiel. High performance liquid chromatography is een veelgebruikte en goedgekende techniek voor kwantitatieve analyses. De werking van deze techniek zal hier om die reden niet meer herhaald worden.

2.4 Verwerking van grote datasets

De moderne analytische toestellen en technieken zorgen ervoor dat bedrijven en onderzoekers te maken krijgen met een grote kwantiteit aan data die steeds uitgebreider wordt en daarbij ook uitdagender om mee te werken. De laatste jaren werd uitgebreid onderzoek gedaan naar manieren om met deze grote datasets om te gaan. Er werd hierdoor een breed scala aan technieken en technologieën ontwikkeld, waardoor de mogelijkheid om gegevens te aggregeren, manipuleren, combineren, analyseren en visualiseren steeds toegankelijker werd. In de volgende paragrafen worden verschillende technieken en technologieën aangehaald voor de verwerking van grote datasets. Daarnaast zal ook extra aandacht gaan naar methoden om data en resultaten visueel voor te stellen. Deze visualisaties kunnen een belangrijk hulpmiddel zijn voor het begrijpen van grootschalige gegevens en complexe analyses. (29)

Er bestaan twee verschillende types van data, namelijk univariate data en multivariate data. Bij univariate data wordt er slechts één variabele gemeten. Een voorbeeld hiervan is wanneer bij spectroscopie de verandering in absorbantie over de tijd wordt gemeten bij één bepaalde golflengte. Bij multivariate data daarentegen worden meerdere variabele tegelijkertijd gemeten. Zo kan bij spectroscopie bijvoorbeeld de verandering van absorbantie over de tijd gemeten worden over het volledige spectrum. Deze data kan dus bijzonder nuttig zijn voor het vinden van verbanden tussen verschillende variabelen. Omdat gemeten data in de meeste gevallen multivariate van aard is, zal deze studie zich beperken tot de analyse van multivariate data. (30)

Er bestaan heel wat statistische en wiskundige methoden om goede gegevens te produceren en om er vervolgens relevante informatie uit te halen. De wetenschap die zich op gebied van chemie hiermee bezighoudt wordt Chemometrie genoemd. Het leveren van goede gegevens kan bereikt worden door gebruik te maken van *design of experiments* (DoE). Multivariate data-analyses kunnen vervolgens op deze gegevens worden uitgevoerd om complexe gegevens te vereenvoudigen, wat ook de visualisatie van de meetgegevens vergemakkelijkt. (31)

2.4.1 Design of Experiments

De beste manier om problemen bij het gebruik van grote datasets te voorkomen is om grote datasets te voorkomen. Design of Experiments (DoE) is een techniek met als doel om een zo klein mogelijke set representatieve gegevens te bekomen bij elk probleem met een zekere mate van experimentele controle. Tegenwoordig is DoE dan ook een zeer populaire methode voor het plannen en uitvoeren van experimenten. In deze aanpak wordt de invloed van elke variabele op een proces gelijktijdig gemeten op verschillende niveaus van alle andere variabelen. Hierdoor kunnen alle interacties tussen de verschillende variabelen in rekening gebracht worden. Er worden resultaten verkregen die een betere representatie geven dan in

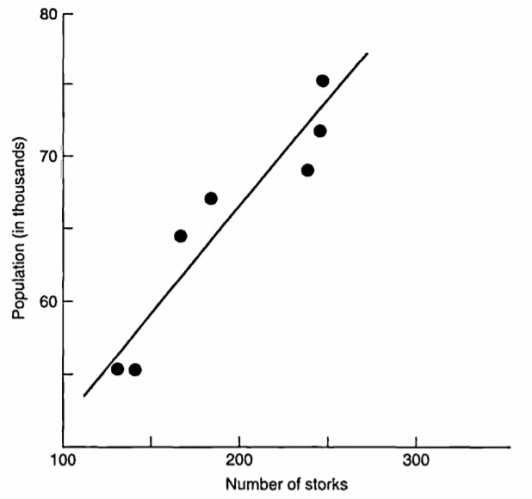
het geval van een *One-Variable-At-Time*-(OVAT)benadering. Met behulp van DoE kunnen dus uitgebreide en betrouwbare resultaten bekomen worden en dit terwijl tijd en middelen worden bespaard. (32)

Bij het uitvoeren van experimenten moeten een aantal stappen doorlopen worden. Om te beginnen moet er een doelstelling geformuleerd worden, vervolgens wordt er een experimenteel programma opgesteld, waarna deze experimenten kunnen uitgevoerd worden. Tot slot wordt de gevonden data geanalyseerd en worden er conclusies getrokken. In elke stap kunnen fouten gemaakt worden die de resultaten kunnen beïnvloeden. Fouten die in een vroeg stadium van het onderzoek gemaakt worden leiden meestal tot grotere fouten, omdat de gevolgen ervan worden versterkt in elke volgende fase van het experiment. Dit kan uiteindelijk leiden tot bevooroordeelde of onjuiste antwoorden. Het uitvoeren van een juist experimenteel plan is dus noodzakelijk om betrouwbare resultaten te bekomen. (32)

2.4.2 Multivariate data-analyse

Vooraleer kan gestart worden met de bespreking van multivariate data-analyses, wordt kort aangehaald hoe multivariate data er juist uitziet. Multivariate data wordt vaak voorgesteld in een matrix X die bestaat uit n rijen en m kolommen. Elke rij stelt een *object* voor, bijvoorbeeld een staal of een voorbeeld. De kolommen geven de verschillende *variabelen (x)* of kenmerken van het object weer. Dit wordt ook wel de x-data genoemd. Bij elk object uit de x-data kan een *respons (y)* gekend zijn. Deze eigenschap kan zowel een continue reële waarde zijn, zoals bijvoorbeeld een concentratie, als een discrete waarde dat duidt op een bepaalde categorie. Deze responsen worden ook wel de y-data genoemd. (33)

Bij het gebruik van grote hoeveelheid x-data is vaak een combinatie van verschillende multivariate analysetechnieken, samen met enkele visualisatie methoden nodig om op een efficiënte manier zinvolle informatie te extraheren. Multivariate data-analyses omvatten statistische, wiskundige of grafische technieken waarbij meerdere variabelen tegelijkertijd worden geanalyseerd. Met behulp van verschillende multivariate data-analyse technieken worden op basis van experimentele gegevens empirische modellen opgebouwd, die vervolgens gebruikt kunnen worden voor interpretatie en voorspellingen van onbekende stalen. Er moet wel rekening mee gehouden worden dat zo een model nooit 100 % correct is, aangezien er altijd ruis en andere irrelevante kenmerken in de gegevens aanwezig zullen zijn. Bovendien is de kans reëel dat bij deze empirische onderzoeken correlaties worden gevonden tussen twee variabelen terwijl er geen oorzakelijk verband bestaat. Correlaties zijn nodig voor het opstellen van een model, maar mogen om deze reden dus nooit worden geïnterpreteerd als een directe causaliteit. Een gekend voorbeeld hiervan is de correlatie tussen het aantal inwoners in de Duitse stad Oldenburg en het aantal ooievaars, te zien in Figuur 16.



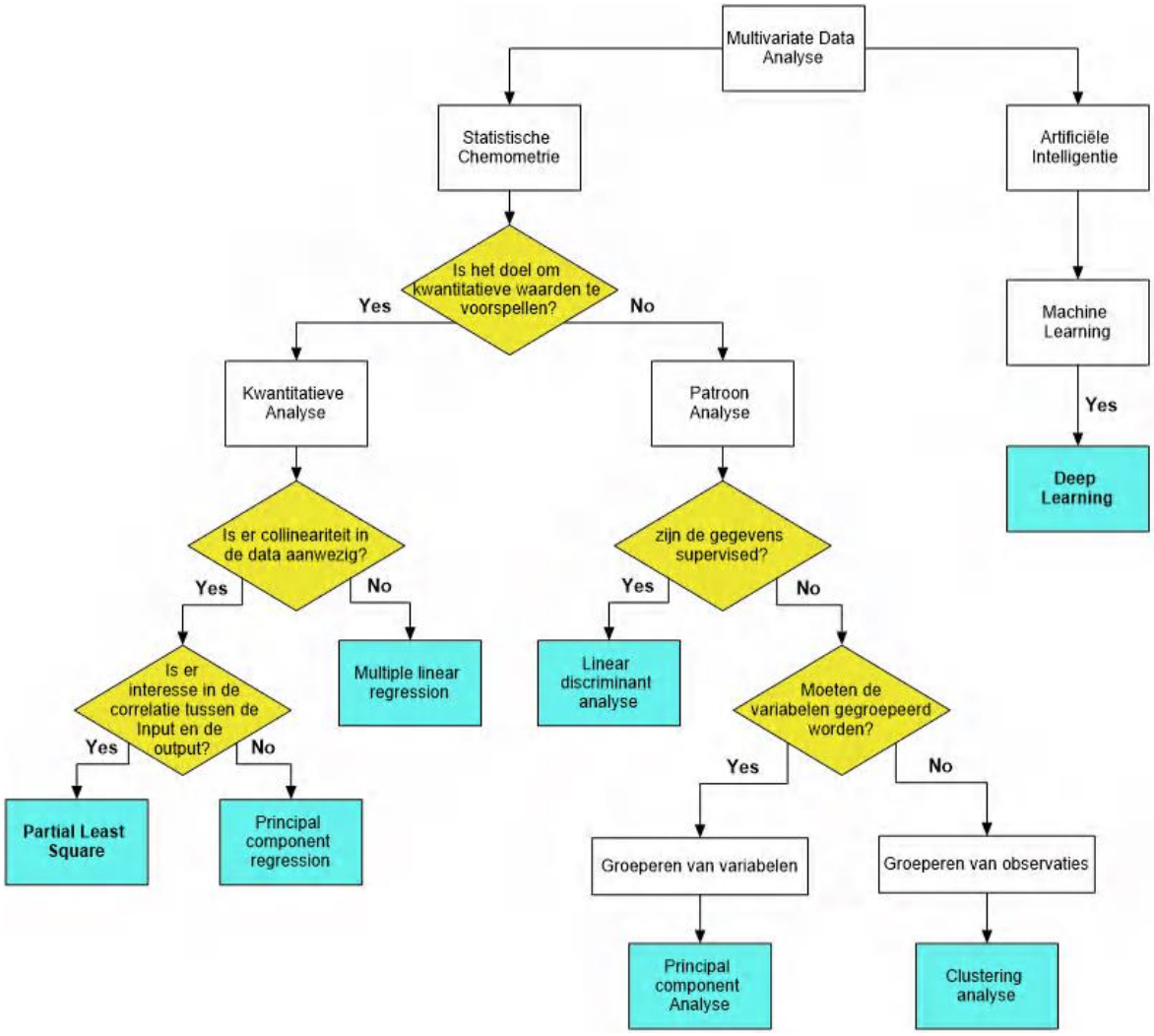
Figuur 16: Aantal ooievaars in functie van aantal inwoners. (34)

Hoewel in deze figuur duidelijk een correlatie te zien is tussen de twee parameters, gelooft niemand nog dat er werkelijk meer baby's geboren worden wanneer er veel ooievaars in de stad aanwezig zijn. Een correlatie tussen twee variabelen wordt vaak gevonden omdat ze beide geassocieerd worden met een derde factor. In dit voorbeeld neemt zowel het aantal ooievaars als het aantal inwoners toe met de tijd. De derde factor is hier dus de tijd. Het vinden van dit soort correlaties tijdens onderzoek is niet de bedoeling, maar soms is het moeilijk te vermijden. Het is daarom essentieel dat het bekomen model op een goede manier wordt gevalideerd en geïnterpreteerd. (31)(34)(35)

Er bestaan verschillende manieren voor het toepassen van Multivariate data-analyses. Figuur 17 geeft een flow diagram weer waarin wordt aangegeven in welke omstandigheden een bepaalde multivariate data-analyse techniek kan worden toegepast. De keuze van de gebruikte techniek hangt af van het type data dat beschikbaar is en de reden voor de data-analyse. Wanneer een onderzoeker geïnteresseerd is in het classificeren van zijn data in meerdere discrete categorieën, zullen er andere technieken gebruikt worden dan wanneer de onderzoeker een continue reële waarde van een bepaalde grootheid wil voorspellen. (35)

De classificatiemethoden kunnen verder worden onderverdeeld in *unsupervised*- en *supervised* classificatiemethoden. Bij de supervised classificatiemethoden wordt een model opgesteld aan de hand van een groot aantal stalen waarvan gekend is tot welke klasse ze behoren, kortom de y-data is gekend. Aan de hand van dat model kunnen nieuwe onbekende stalen aan bepaalde klassen worden toegeschreven. Bij de unsupervised classificatiemethoden daarentegen, is de y-data niet gekend. De modellen worden opgebouwd aan de hand van een groot aantal stalen waarvan niet gekend is tot welke klasse ze behoren. De data wordt dan geclassificeerd op basis van hun overeenkomsten in de x-data, zonder een specifieke klasse eraan toe te schrijven. Met dit model is het mogelijk om te kijken of bepaalde stalen gelijkaardig zijn, of net ongebruikelijk. Deze methode wordt eerder clusteren genoemd. (35)

In wat volgt wordt, aan de hand van Figuur 17, voor elke techniek bekeken of deze geschikt is voor het verwerken van de dataset afkomstig van de spectrofotometer, de FTIR en de HPLC. In een eerste deel zullen de statistische chemometrische multivariate data-analyses besproken worden. Er zal dieper worden ingegaan op die technieken die geschikt lijken voor de analyse van de gegeven dataset. Daarna zal worden overgegaan naar de nieuwere veelbelovende methoden in het gebied van artificiële intelligentie.



Figuur 17: Flow diagram voor het gebruik van verschillende methoden voor multivariate data-analyse. (36)(35)(37)

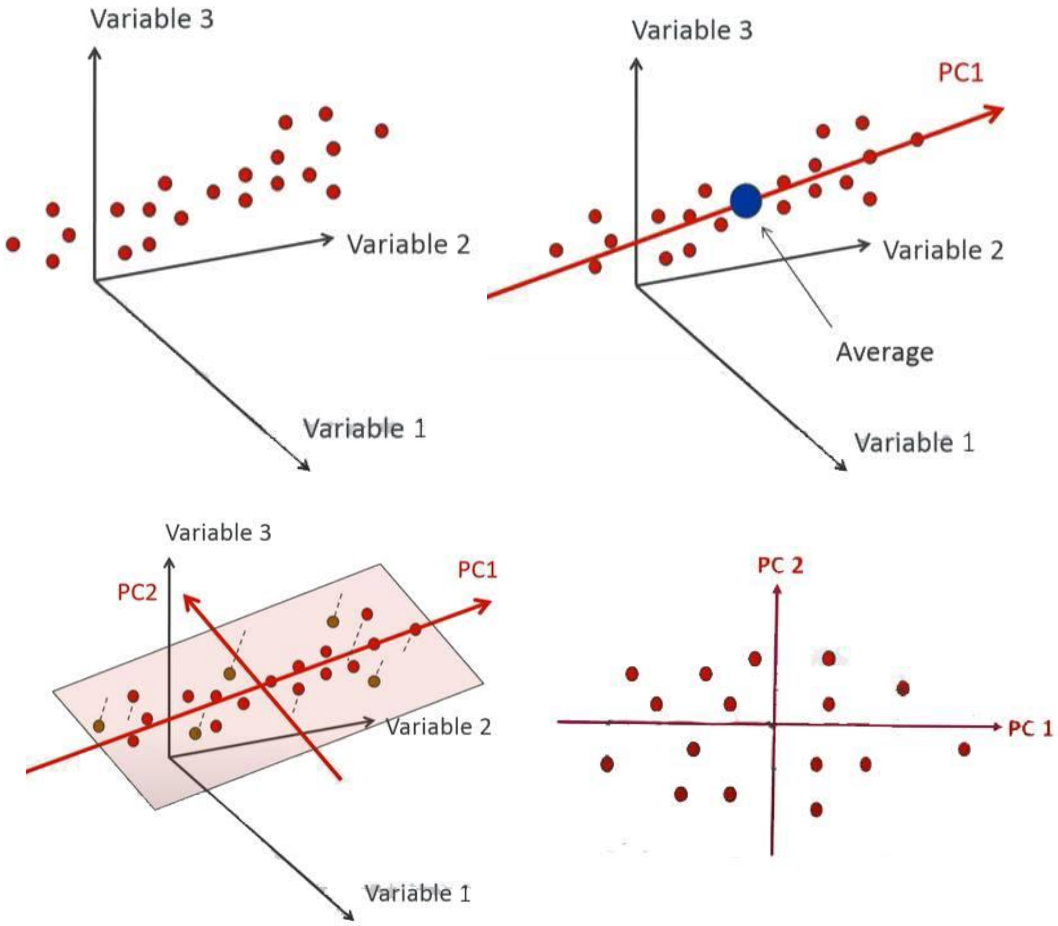
2.4.2.1 Statistische multivariate data-analyses

2.4.2.1.1 Patroon analyses

Principle component analyses (PCA) is één van de belangrijkste multivariate data-analyses. Deze techniek is vooral krachtig wanneer grote hoeveelheden gegevens worden geproduceerd. Het is een uitstekende techniek om verborgen informatie in de gegevens te onthullen. Deze techniek lijkt op het eerste zicht dan ook geschikt voor de analyse van de grote dataset met gegevens afkomstig van drie verschillende toestellen. (31)

Wanneer het aantal variabelen bij metingen heel groot is, is de kans groot dat er een aantal variabelen gecorreleerd zijn. Dit wil zeggen dat de variabelen volledig of gedeeltelijk dezelfde eigenschap van een systeem beschrijven, waardoor ze dus gelijkaardige informatie opleveren. PCA wordt gebruikt om de dimensies van zo een grote dataset te verminderen, met behoud van zo veel mogelijk van de in de dataset aanwezige variatie. Deze vermindering wordt bekomen door de dataset te transformeren naar een nieuwe set niet-gecorreleerde variabelen die principle componants (PC) worden genoemd. De PC's worden gerangschikt volgens de grootte van hun variantie in de x-variabelen. Variantie is een maat die de spreiding van een

reeks waarde. De eerste PC's geven dus de meeste variantie in de dataset weer en bevatten daarom ook de meeste informatie. De laatste PC's identificeren de richtingen waarin er zeer weinig variantie is. Dit wil zeggen dat ze een zo goed als lineaire relatie hebben met de oorspronkelijke variabelen, waardoor ze geëlimineerd kunnen worden. Dit principe wordt aangetoond met behulp van Figuur 18. (38)



Figuur 18: Werking van PCA. (38)

In de eerste figuur worden de verschillende stalen uitgezet in functie van de oorspronkelijke variabelen. De eerste PC wordt dan bepaald door de grootste variantie in de data te zoeken, dit is weergegeven in de tweede figuur van Figuur 18. De eerste PC is meestal nog niet genoeg om de variatie van data te beschrijven. De richting die de op één na meeste variantie in de data bevat wordt aangeduid met PC2. De kans bestaat dat deze twee PC's samen al meer dan 90 % van de totale variantie van de data beschrijven. Nog meer PC's gebruiken voor het beschrijven van de data zou dus nog maar weinig extra informatie opleveren. In andere gevallen kan het wel nodig zijn om nog PC's te gebruiken voor het beschrijven van de data. (38)

PCA wordt vaak gebruikt bij spectrale gegevens. Bij een dataset van een spectrum staat elke opgemeten golflengte voor een aparte x-variabele, waardoor het aantal x-variabelen al snel oploopt tot enkele duizenden. PCA kan dan gebruikt worden om de dimensies van de x-variabelen te verkleinen. Wanneer met behulp van PCA wordt vastgesteld dat een bepaalde dataset met veel variabele in werkelijkheid dicht bij een twee- of driedimensionale subruimte ligt, is het mogelijk om deze gegevens grafisch weer te geven. In deze weergaven van de data kunnen patronen, zoals clusters, trends en uitschieters uit de complexe datasets makkelijk

worden onthuld. Deze techniek zou dus geschikt kunnen zijn voor het classificeren van verschillende soorten textiel. Bovendien zou het mogelijk kunnen zijn om te onderzoeken of er al dan niet een duidelijk onderscheid in het spectrum is tussen textiel met additief of textiel zonder additief. Aangezien enkel de variantie in de x-variabelen in rekening wordt gebracht en de y-data hier dus niet voor gekend moet zijn, is deze techniek een unsupervised classificatie techniek. (39) (31)(35)(39)(40)

Een tweede manier voor unsupervised classificatie zijn de clustering analyses. Er bestaan verschillende algoritmes waarmee op zoek wordt gegaan naar clusters of groepen die gelijkaardige kenmerken bevat, zonder dat er info is over welke groepen er bestaan of zelf hoeveel groepen er moeten verdeeld worden. Deze technieken zouden ook handig kunnen zijn voor het analyseren van de FTIR-spectra en de reflectie spectra van de spectrofotometer. Met deze technieken zou het bijvoorbeeld mogelijk zijn om de verschillende FTIR-spectra te clusteren in groepen naargelang de aanwezigheid van een bepaald additief. Het gaat hier echter wel weer om een unsupervised methode en er kan dus opnieuw geen info over de y-waarde worden opgenomen in de analyses. (33)

Een van de veel gebruikte en makkelijkst uit te voeren classificatie methoden voor supervised data is Lineair discriminantanalyses (LDA). Deze methode gaat net zoals bij PCA de dimensies van de gegevens reduceren, maar tegelijk ook proberen om de scheidbaarheid tussen gekende categorieën te maximaliseren. De classificatie gebeurt door de variantie tussen de verschillende klassen te maximaliseren en de variantie binnen klassen te minimaliseren om op die manier de optimale grenzen tussen de klassen te vinden. Door deze twee criteria op hetzelfde moment te optimaliseren wordt een goede scheiding tussen de verschillende klassen bekomen. De methode wordt gebruikt om de relaties die een invloed hebben op de categorie waarin een object zich bevindt te onderzoeken of te voorspellen. Ook deze techniek lijkt interessant te zijn voor het verwerken van de aanwezige dataset, maar de techniek ondervindt al snel moeilijkheden bij het verwerken van hoog-dimensionale spectrale gegevens. (35)(37)(41)(42)(43)

2.4.2.1.2 Kwantitatieve analyses

Multiple linear regression (MLR) is de meest eenvoudige manier om een multivariate model op te bouwen voor het voorspellen van een continue reële waarde van een bepaalde grootheid. Het doel van deze voorspellende modellen is om de relatie te bepalen tussen verschillende x-variabelen (onafhankelijke variabelen) en één of meer y-variabelen (afhankelijk of respons variabelen). Het is een uitbreiding van de univariate lineaire regressie waarbij één enkele y-variabele wordt voorspeld aan de hand van een opgebouwd model dat er als volgt uitziet.

$$y = b_0 + b_1 * x + f \tag{2.10}$$

Hierin is y de y-variabele, b_0 is de intercept, b_1 de helling, x de x-variabele en f de fout. De helling wordt bepaald met behulp van de kleinste kwadraten methode. Bij deze methode wordt de som van de kwadraten van het verschil tussen het waargenomen resultaat (y) en de functie van de x-variabelen geminimaliseerd. Bij MLR moeten verschillende x-variabelen tegelijkertijd in rekening worden gebracht om de y-waarde te voorspellen. Het model wordt dan als volgt uitgebreid.

$$y = b_0 + b_1 * x_1 + b_1 * x + b_2 * x_2 + \cdots + b_n * x_n \tag{2.11}$$

Hierin staat y en b_0 opnieuw voor respectievelijk de y-variabelen en de intercept. X_1, x_2, … zijn de verschillende x-variabelen en b_1, b_2, … zijn de regressie coëfficiënten. Deze laatste worden opnieuw bepaald via de kleinste kwadraten methode. (40)

MLR wordt vaak toegepast wanneer het aantal x-variabelen vrij laag is, bovendien moeten de variabelen zo goed als lineair onafhankelijk zijn. Bij de dataset van de drie analysemethode is het aantal x-variabelen zeer groot. Zo wordt bij zowel de FTIR en de witheidsmetingen een heel spectrum opgemeten, wat dus zorgt voor een groot aantal x-variabelen. Bij zo een dataset is de kans groot dat er collineariteit bestaat tussen de variabelen. Wanneer dit het geval is, is het niet meer mogelijk om MLR toe te passen. Daardoor zal deze techniek dus niet geschikt zijn voor het verwerken van de dataset. (40)

Principal component regression (PCR) biedt wel een mogelijkheid om grote datasets met collineaire x-variabele te analyseren. Met behulp van PCA worden de dimensies van de grote dataset verminderd en blijft er een set niet-gecorreleerde variabelen over. Op deze variabelen kan vervolgens wel een MLR worden uitgevoerd. (31)(40)

Een beter veelgebruikt alternatief voor PCR is *Partial least square regression* (PLSR)**.** In tegenstelling tot PCR wordt bij PLSR gezocht naar combinaties van de originele gegevens met een maximale covariantie tussen de x-variabelen en de y-variabelen, in plaats van een maximale variantie tussen de x-variabelen. Covariantie is een maat dat de lineaire samenhang van twee variabelen aangeeft. Voor het beschouwen van de covariantie wordt zowel de correlatie met de respons en de variantie binnen de x-variabelen in rekening gebracht. Hierdoor bevat de gereduceerde data set zowel relevante info over de x-variabelen als over de y-variabelen. Dit is een groot verschil met PCR aangezien de PC's enkel bepaald worden op basis van de variantie van de x-variabelen. PLSR maakt het dus mogelijk om de x-variabelen en de y-variabelen gezamenlijk te evalueren. Dit maakt dat dit de meest geschikte techniek is voor het analyseren van de gegeven dataset. (31)(39)(41)(44)

2.4.2.2 Artificiële intelligentie

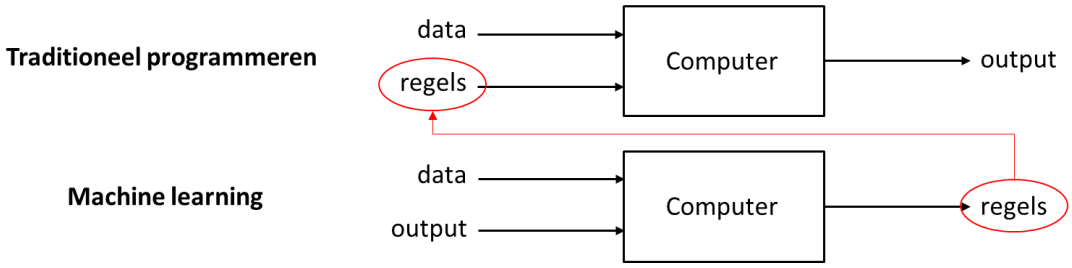
Naarmate de datasets groter worden, wordt het steeds moeilijker om ze met bovenstaande statistische technieken te verwerken. Er zal snel *overfitting* optreden, wat moeilijk te vermijden is. Om die rede worden tegenwoordig technieken binnen het veld van artificiële intelligentie ingezet voor het analyseren en categoriseren van grote ongestructureerde datasets. Deze technieken zoals bijvoorbeeld machine learning en deep learning zijn nauw verwant met de wiskundige statistiek. De meeste multivariate technieken die hierboven al werden besproken, worden ook toegepast in machine learning voor het vinden van regels. Toch zijn er belangrijke verschillen tussen deze technieken en statistiek. Om te beginnen behoren ze tot de computerwetenschap terwijl statistische modelleren een deel is van de wiskunde. Hoewel zowel machine learning als deep learning gebruik maken van statistische technieken om het doel te bereiken, gaat deze techniek veel verder dan dat. (45)(46)

Artificial intelligence (AI) is een overkoepelende term voor elke techniek waarmee computers menselijk gedrag kunnen nabootsen. Hierbij worden algoritmes opgebouwd om informatie te verwerken en toekomstige voorspellingen te maken. In de eerste plaats werd intelligentie op menselijk niveau bereikt door het opstellen van een groot aantal expliciete regels. Een voorbeeld hiervan zijn schaakprogramma's waarvoor hard gecodeerde regels worden opgesteld door ervaren programmeurs en die vervolgens in staat zijn te winnen van mensen. Deze benadering was vooral in de jaren 50 tot de jaren 80 het dominante paradigma bij AI. Al snel bleek dat deze vorm van AI enkel geschikt was voor het oplossen van goed gedefinieerde,

logische problemen, maar dat het te moeilijk was om expliciete regels te bedenken voor het oplossen van complexe en vage problemen zoals bijvoorbeeld spraakherkenning en taalvertaling. Nochtans zijn dit twee voorbeelden waar we in het dagelijks leven toch regelmatig mee in aanmerking komen of gebruik van maken. Deze problemen werden namelijk wel opgelost met behulp van een nieuwe techniek binnen dit veld namelijk machine learning. (45)(47)

2.4.2.2.1 Machine learning

Machine learning is een deel van artificiële intelligentie. Hier wordt intelligentie op niveau van de mens niet meer behaald met behulp van de door de programmeurs opgestelde regels. Wel gebeurt dit doordat de computer zelf de regels gaat leren, door input van de omgeving op te slaan, patronen erin te herkennen en deze vervolgens toe te passen met oog op succes. In 1959 definieerde Arthur Samual machine learning als “De studie dat computers de mogelijkheid geeft om te leren zonder expliciet geprogrammeerd te worden.” Het verschil tussen traditioneel programmeren en machine learning wordt uitgelegd aan de hand van Figuur 19. (45)(48)(49)



Figuur 19: Verschil tussen traditioneel programmeren en machine learning schematisch weergegeven. (45) (48)

Bij traditioneel programmeren wordt een programma geschreven dat gevoed wordt aan de computer, deze gebruikt dan de beschikbare data om een bepaald doel (output) te bekomen. Bij machine learning daarentegen wordt de output gevoed aan de computer. Het machine learning algoritme zal hiermee een programma produceren dat vervolgens gebruikt kan worden voor nieuwe output te creëren met data waarvan de output nog niet bekend is. De output die aan de computer gevoed wordt om machine learning toe te passen bestaat uit een zeer groot aantal voorbeelden die relevant zijn voor een bepaald doel. Wanneer het doel bijvoorbeeld is om verschillende vakantiefoto’s te labelen volgens hun bestemming, moet er een grote verzameling van die foto’s beschikbaar zijn als data om aan de computer te voeden. De output die erbij gevoed wordt is het gekende label voor elke foto. Het algoritme zoekt statistische structuren in deze voorbeelden, waarmee het uiteindelijk regels bedenkt voor het automatiseren van het doel. In het voorbeeld wordt dus een programma gevonden dat toe laat om nieuwe niet gelabelde foto’s automatisch te labelen. (45)(48)

Machine learning algoritmes kunnen net zoals multivariate data-analyses in twee groepen ingedeeld worden, naargelang het doel van de machine learning. Classificatie algoritmes worden gebruikt wanneer het de bedoeling is om datasets te classificeren in verschillende groepen en te kunnen voorspellen tot welke groep een bepaald staal behoort. Wanneer het doel is om gegevens kwantitatief te gaan voorspellen wordt gebruik gemaakt van regressie methoden. Daarnaast worden de machine learning algoritmes ook opgesplitst in twee groepen naargelang het soort data dat beschikbaar is. Net zoals bij de multivariate data analysis gaat het dan om supervised en unsupervised Learning. Bij supervised learning wordt een dataset

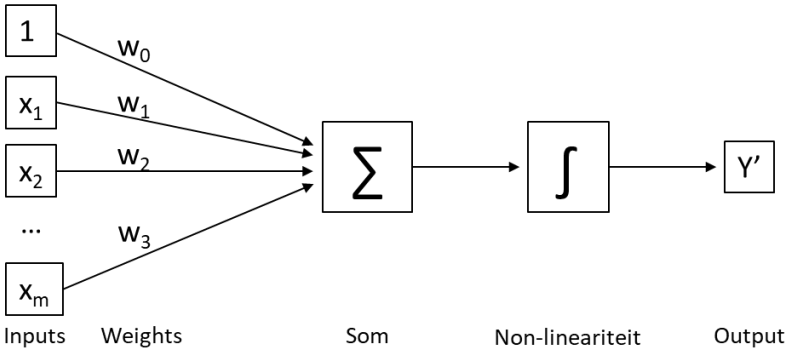
gebruikt waarbij voor elk staal het label gekend is. Hier wordt met een algoritme gezocht naar een programma die de labels voorspelt. Wanneer een dataset beschikbaar is zonder dat de labels ervan gekend zijn wordt gesproken van unsupervised learning. De labels kunnen hier dan ook niet voorspeld worden. Wel kan de dataset in clusters verdeeld worden of kan voorspeld worden of een bepaald staal gelijkt op de andere stalen. (48)

Door de toenemende beschikbaarheid van snellere hardware en grotere datasets is machine learning heel snel het meest populaire en succesvolle onderdeel van AI geworden. Toch worden langs alle kanten nieuwe technieken ontwikkeld en wordt steeds verder gegaan in het veld van artificiele intelligentie. Zo slaagde de Hilton's groep er in 2012 in om bij de jaarlijkse grootschalige uitdaging voor beeldclassificatie ImageNet een nauwkeurigheid van 83,6 % te halen. De ImageNet uitdaging bestond uit het classificeren van kleurenafbeeldingen met een hoge resolutie in 1000 verschillende categorieën, na een training met 1,4 miljoen verschillende afbeeldingen. Deze opdracht was in die tijd onvoorstelbaar moeilijk, maar met behulp van een nieuwe techniek binnen machine learning die gebruik maakt van de zogenaamde neurale netwerken werd zelfs deze classificatiestaak volledig oplosbaar. Deze techniek wordt deep learning genoemd. (45)

2.4.2.2.2 Deep Learning

Bij deep learning wordt er nog een stap verder gegaan. Hier worden patronen gezocht in ruwe data, zonder de behoefte van de mens om de regels die het systeem moet leren te expliceren. Dit gebeurt door het invoegen van lagen met een steeds meer betekenisvolle voorstelling. De gegevens worden per laag omgezet in representaties die steeds meer afwijken van hun oorspronkelijke vorm en steeds meer informatie bieden voor het eindresultaat. Al deze lagen tezamen wordt een neuraal netwerk genoemd. De term neuraal netwerk verwijst naar de neurobiologie, aangezien een groot deel van deep learning werd ontwikkeld door inspiratie te putten uit de werking van onze hersenen. (45)

Om de werking van de een neuraal netwerk uit te leggen wordt eerst gekeken naar één enkele neuron, wat ook een preceptor wordt genoemd. Vervolgens wordt dit uitgebreid naar de werking van een volledig neuraal netwerk. In Figuur 20 wordt één enkele neuron van een neuraal netwerk afgebeeld.



Figuur 20: Voorstelling van één enkele neuron. (47)

Er wordt een set van inputs voor het neuron gedefinieerd namelijk x_1 tot x_m. Elk van deze inputs bevat een bijbehorende *weight*. In een neuron wordt elke input vermenigvuldigd met deze weight, vervolgens wordt de som genomen van al deze producten. Hierdoor wordt één getal bekomen dat door een bepaalde activatie functie g getransformeerd wordt naar de output. Een neuron bevat ook altijd een bias term, w_0. Deze term geeft de mogelijkheid om de activatie

functie op te schuiven naar links of naar rechts onafhankelijk van de input waarden. De bewerkingen uit Figuur 20 kunnen mathematisch geschreven worden met volgende formule.

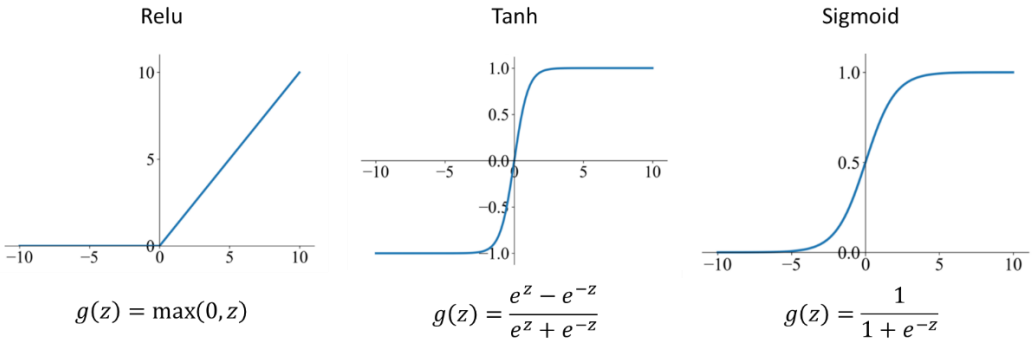
$$Y' = g(w_0 + \sum_{i=1}^{m} x_i\, w_i) \tag{2.12}$$

Door gebruik te maken van lineaire algebra kan deze formule herschreven worden met behulp van vectoren.

$$Y' = g(w_0 + X^T W) \quad met\ X = \begin{bmatrix} x_1 \\ \dots \\ x_m \end{bmatrix}\ en\ W = \begin{bmatrix} w_1 \\ \dots \\ w_m \end{bmatrix} \tag{2.13}$$

De activatie functie maakt het mogelijk om non-lineairiteit in het netwerk te verkrijgen. Dit is heel belangrijk, aangezien data in veel gevallen non-lineair is. Binnen deep learning worden verschillende activatie functies gebruikt zoals bijvoorbeeld de “sigmoid” function, “hyperbolic tangent” of een “unit step”. Elke functie heeft zijn voor en nadelen en de keuze hangt af van de aard van het op te lossen probleem. (47)

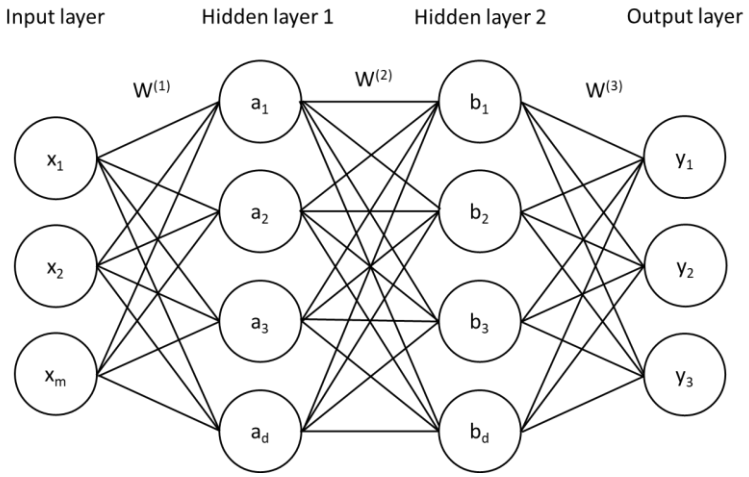
In Figuur 21 worden enkele veelgebruikte activatiefuncties weergegeven.



Figuur 21: “Relu”, “tanh” en “sigmoid” activatie functie. (50)

Bij de rectified linear activation functie (“relu”) blijft de input onveranderd wanneer deze een waarde boven 0 heeft. De waarden onder 0 worden omgezet naar een output met waarde nul. Bij de hyperbolic tangent functie (“tanh”) wordt de input omgezet naar een waarde tussen -1 en 1. De sigmoid activatie functie transformeert de input naar een waarde tussen 0 en 1. Deze laatste activatie functie wordt vaak gebruikt in de laatste laag van een neuraal netwerk voor een classificatie in twee klassen omdat een waarde tussen 0 en 1 de kans beschrijft dat het staal tot een van de twee klassen behoort. Een speciaal geval van de “sigmoid” functie is de “softmax” activatie functie. Hierbij is de output een vector met waarden tussen 0 en 1, de som van alle waarden is gelijk aan 1. De elementen van de vector zijn dus geschikt voor het aangeven van de categorische kans, waardoor deze activatie functie vooral gebruikt wordt in de laatste laag van een neuraal netwerk voor een classificatie in meerdere klassen. (47)(51)

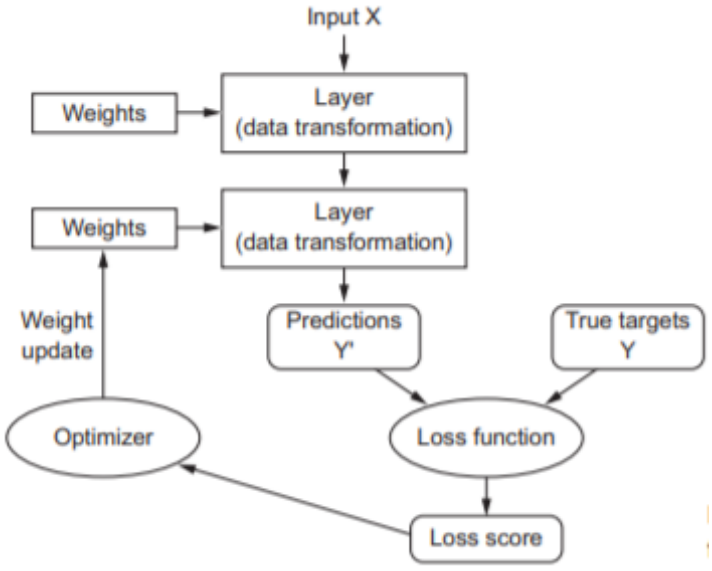
Door meerdere preceptoren samen te voegen wordt een laag gevormd tussen de inputmatrix en de output. Dit wordt een *single layer network* genoemd. De laag wordt ook wel de verborgen laag of *hidden layer* genoemd aangezien de toestanden daar niet observeerbaar zijn. Een *deep neural network* wordt gecreëerd door verschillende verborgen lagen achter elkaar te plaatsten. In Figuur 22 wordt een neuraal netwerk met twee verborgen lagen op een schematische manier afgebeeld.



Figuur 22: Schematische voorstelling van een neuraal netwerk met twee verborgen lagen. (47)

De eerste laag bestaat uit de inputs, de volgende laag is een verborgen laag die is opgebouwd uit verschillende neuronen. Omdat ze in de verborgen laag aanwezig zijn worden deze ook wel verborgen eenheden of *hidden units* genoemd. Het neuraal netwerk kan verschillende verborgen lagen bevatten. De laatste laag is de laag met de outputs. In Figuur 22 is elke verborgen eenheid verbonden met elke verborgen eenheid van de vorige laag, elk met een verschillende set van weights. Wanneer alle neuronen uit de verschillende lagen op die manier nauw geconnecteerd zijn met elkaar, wordt er gesproken van *dense layers*. Tussen elke laag wordt de data aan de hand van vergelijking 2.12 getransformeerd. Elk van deze transformaties heeft zijn eigen weight matrix W, in Figuur 22 worden deze aangeduid met $W^{(1)}$, $W^{(2)}$ en $W^{(3)}$.

Het doel van het netwerk is het vinden van een set waarden voor de weights van alle lagen in het netwerk, zodat het netwerk nieuwe inputs zo correct mogelijk in kaart kan brengen. Het vinden van deze waarden is echter niet vanzelfsprekend aangezien een deep neural network wel tientallen miljoenen parameters kan bevatten. De manier waarop zo een neuraal netwerk toch aan de optimale weights komt wordt weergegeven in Figuur 23 en vervolgens uitgelegd aan de hand van een voorbeeld. (47)



Figuur 23: Training van neurale netwerken. (45)

Er moet een neuraal netwerk worden opgesteld om te voorspellen welke leerlingen zullen slagen voor hun examen. Als inputs worden de volgende parameters gebruikt

x_1 = het aantal lessen dat de leerling bijwoonde
x_2 = het aantal uur dat de leerling studeerde voor het examen

Stel dat een leerling 12 lessen bijwoonde en 10 uur studeerde voor het examen, wat is dan de kans dat hij of zij zal slagen voor het vak? Wanneer deze inputs gevoed worden aan een niet getraind neuraal netwerk, wordt er een kans van 10 % geschat dat de leerling geslaagd zal zijn. Nochtans blijkt dat vele andere studenten die ongeveer evenveel tijd in het vak staken toch geslaagd waren voor dit vak. Het neuraal netwerk geeft dus een slecht resultaat. Dit is logisch aangezien het nog niet getraind is. De weights waarmee in dit neuraal netwerk gestart wordt zijn een willekeurige initiële gok, waardoor de kans groot is dat een fout resultaat zal bekomen worden. Het neuraal netwerk wordt getraind door telkens aan te geven dat het fout is, zodat het zich in de toekomst kan corrigeren. Dit gebeurt via de verliesfunctie (*Loss function*). De verliesfunctie kijkt naar de door het netwerk voorspelde output en de werkelijke output en berekent de afstandsscore, wat aangeeft hoe ver de voorspelde output verwijderd is van wat verwacht wordt. Zoals op Figuur 23 te zien is wordt deze score gebruikt als feedbacksignaal om de waarde van de weights geleidelijk aan te passen.

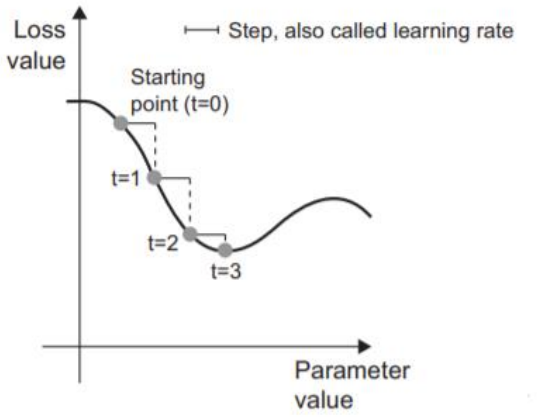
Bij een grote afstandsscore is het verschil tussen de werkelijke en de voorspelde waarde groot, dit wil zeggen dat het netwerk niet goed presteert en gecorrigeerd moet worden. Bij een kleine afstandsscore liggen de twee dicht bij elkaar en moeten de weights maar een beetje worden aangepast. Wanneer dit voldoende herhaald wordt zal de afstandsscore altijd maar kleiner worden. Op die manier leert het neuraal netwerk zelf hoe het aan het beste resultaat kan komen, zonder dat de mens expliciet moet zeggen hoe hij dit moet doen. (45) (47)

Bovenstaande *training loop* kan worden samengevat in de volgende drie stappen.

1. Het netwerk laten lopen op training monsters x, waarmee de bijbehorende y waarden voorspeld worden
2. Het berekenen van het verlies van het netwerk op de batch, dit geeft het verschil tussen de werkelijke en de door het netwerk voorspelde y waarden weer.
3. Het bijwerken van alle weights van het netwerk op die manier dat het verlies van de batch verkleind wordt.

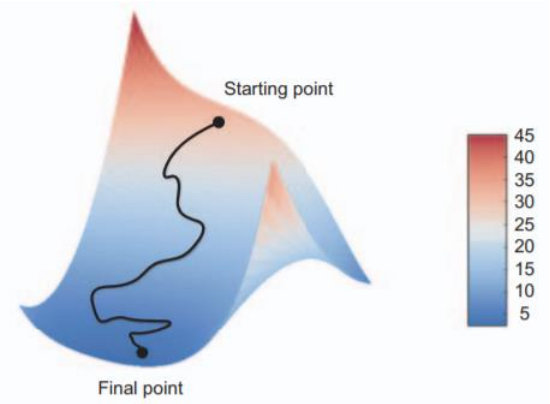
Voor elk specifiek doel wordt er een andere verlies functie gebruikt. Wanneer het doel bijvoorbeeld het classificeren van data is wordt een “Cross-Entropy” verlies functie gebruikt. Bij deze functie wordt het verschil tussen twee kansverdelingen van dezelfde dataset bepaald. Het verlies is hoger wanneer de voorspelde kansverdeling verschilt van het werkelijke label van een bepaald staal. Een veelgebruikte verliesfunctie bij regressies is de *mean squared error* (MSE), waarmee het kwadraat van het verschil tussen de werkelijke en voorspelde waarde berekend wordt. Deze methode wordt veel toegepast bij regressie en werd bij de multivariate analyses ook al aangehaald. (45)

Na het bereken van het verlies moeten de weights worden geoptimaliseerd om het verlies te laten dalen. Het is vooral deze stap die voor de meeste moeilijkheden zal zorgen. Voor de combinatie van weights die de kleinst mogelijke verliesfunctie oplevert, moet dus worden gezocht naar het minimum van de verliesfunctie. Dit gebeurt meestal aan de hand van *stochastic gradient descent* (SGD). De werking van deze techniek wordt uitgelegd aan de hand van Figuur 24.



Figuur 24: SGD bij een verliesfunctie met slechts één lerende parameter. (45)

In deze figuur is de verliescurve van een netwerk dat slechts één lerende parameter bevat weergegeven, de verliesfunctie is dus in functie van slechts één weight. Om het minimum van deze curve te bepalen wordt er gestart in een willekeurig punt, wat overeenkomt met een willekeurige weight. Vervolgens wordt de gradiënt van de functie in dat punt berekend. De gradiënt geeft de richting van de maximale verandering van een functie in een bepaald punt weer. Om het verlies te laten dalen moeten de weights dus in de tegenovergestelde richting worden aangepast. De grote van de aanpassing wordt de *step* of *learning rate* genoemd. De grootte van deze step heeft een grote invloed op het vinden van het minimale verlies. Wanneer deze te klein wordt gekozen zal het lang duren eer het minimum wordt bereikt, bovendien is het ook mogelijk dat het proces vast komt te zitten in een lokaal minimum. Bij een te grote step bestaat de kans dat er voorbij het minimum wordt gegaan of dat de updates zelf willekeurige waarden op de curve aannemen. Door deze procedure meermaals te herhalen wordt het verlies uiteindelijk minemaal. Dit algoritme is iteratief, wat betekent dat het zoekproces plaatsvindt in meerdere afzonderlijke stappen, waarbij elke stap de modelparameters enigszins verbeteren. In een echt neuraal netwerk moeten echter heel veel weights tegelijk worden aangepast bij elke stap. Het daadwerkelijke proces kan dus niet gevisualiseerd worden aangezien een ruimte met bijvoorbeeld 1 000 000 dimensies niet mogelijk is. In Figuur 25 wordt wel nog geïllustreerd hoe SGD in zijn werk gaat bij twee lerende parameters. (45)(52)

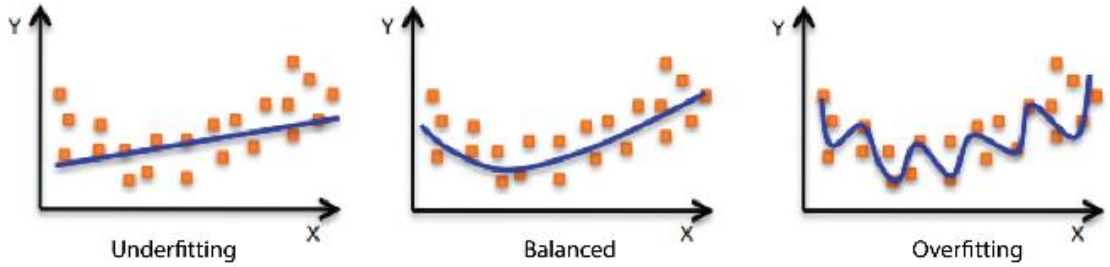


Figuur 25: SGD bij een verliesfunctie met twee lerende parameters. (45)

Bij deep learning worden verschillende varianten van SGD gebruikt als *optimizer* bij het updaten van de weights. Er wordt slechts één deze varianten besproken, namelijk de “RMSProp” optimizer, wat een afkorting is voor *root mean square propagation*. Hierbij wordt

het gemiddelde van het kwadraat van de gradiënt van alle voorgaande stappen bepaald en vervolgens wordt de gradiënt gedeeld door de vierkantswortel van dit gemiddelde. Door de voorgaande gradiënten ook in rekening te brengen, is de kans groter dat optimalisatieproces niet vast komt te zitten in een lokaal minimum. Dit is vergelijkbaar met een bal die langs de verliescurve rolt. Wanneer deze genoeg momentum bevat zal deze voorbij het lokaal minimum geraken en uiteindelijk terecht komen in het globaal minimum. Dit momentum is afhankelijk van snelheid van de bal, wat dan weer afhankelijk is van de helling voor het lokaal minimum. (45)

Uiteindelijk is het de bedoeling dat door de juiste parameter instellingen een goed model bekomen wordt, zonder dat underfitting of overfitting optreedt. In Figuur 23 wordt het verschil tussen een goed model en modellen waar overfitting en underfitting plaats vindt aangetoond.

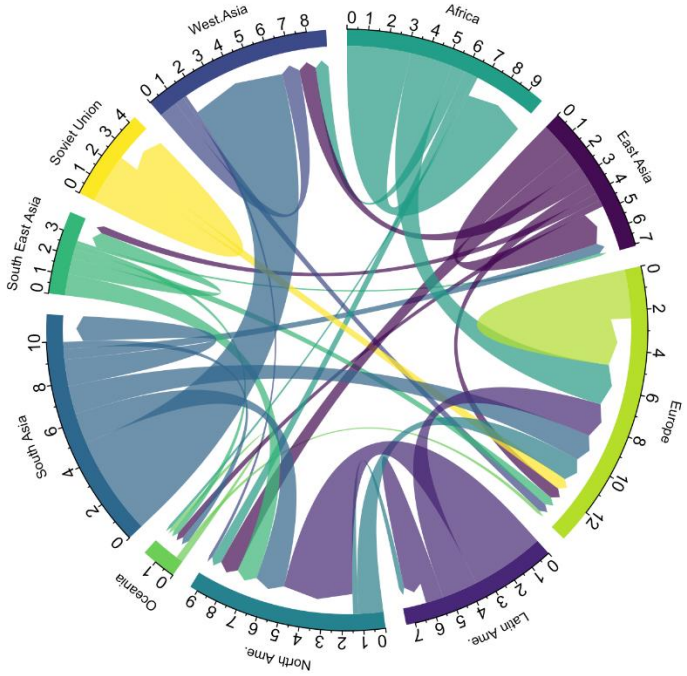


Figuur 26: Verschil tussen een goed model en model met underfitting en overfitting. (53)

Underfitting komt voor wanneer het niet goed presteert op de training data. Het model modelleert nog niet alle relevante patronen in de trainingsdata en is daardoor niet in staat om de relatie tussen de input variabelen X en de output Y te vinden. Overfitting treedt op wanneer het model goed presteert op de training data, maar niet meer goed presteert op de test data. Het model vindt hier ongewenste patronen in de training data en begint stalen ervan te memoriseren waardoor het niet lukt om ongeziene stalen te voorspellen. (53)

2.4.3 Visualisatie methoden

Er bestaan veel verschillende manieren om multivariate data visueel voor te stellen. Deze visualisaties kunnen een belangrijk hulpmiddel zijn voor het begrijpen van grootschalige gegeven. Na het uitvoeren van de multivariate analyses is het ook belangrijk om de resultaten op een goede manier visueel te kunnen weergeven. Met behulp van een goede visualisatie wordt direct duidelijk welk effect de ingewikkelde berekeningen hadden op de grote dataset en vaak is het op die manier in één oogomslag duidelijk welke informatie aanwezig is in de complexe dataset. De manier waarop er iets wordt voorgesteld hangt natuurlijk af van wat er duidelijk gemaakt moet worden. Voor dit onderzoek wordt er gezocht naar correlaties tussen verschillende x-variabelen afkomstig van verschillende meettoestellen. Een goede manier om de gevonden correlaties duidelijk te kunnen afbeelden is met behulp van een Chord diagram. In Figuur 27 wordt een voorbeeld gegeven van zo een diagram.



Figuur 27: Voorbeeld van een Chord diagram. (54)

Voor dit onderzoek zullen de drie analysemethoden op de buitenste cirkel worden afgebeeld. Binnen deze methoden worden de belangrijkste variabelen die met deze techniek gemeten worden naast elkaar in de cirkel geplaatst (op de plaats van de cijfers in Figuur 27). Wanneer twee variabelen met elkaar gecorreleerd zijn worden ze met elkaar verbonden in het chord diagram. Op die manier kan heel snel worden gezien welke variabelen met elkaar gecorreleerd zijn en welke niet. Het zou dus een geschikte methode zijn om de correlaties tussen de gegevens van de dataset afkomstig van de drie toestellen weer te geven.

3 METHODOLOGIE

Uit de literatuurstudie blijken twee technieken veelbelovend voor het zoeken van verborgen correlaties tussen verschillende meetgegevens, namelijk PLS en deep learning. In deze thesis wordt gefocust op het toepassen van deep learning op de beschikbare dataset.

3.1 Materialen

Om deep learning toe te passen op de dataset van drie verschillende analysetoestellen werd het boek *Deep Learning with Python* van François Chollet als leidraad gebruikt (44). In dit boek wordt gebruik gemaakt van het Keras deep-learning framework voor Python. Keras is een toegankelijke en productieve gebruiksomgeving waarmee bijna elk type deep learning model getraind kan worden. Keras maakt dan weer gebruik van wiskundige bibliotheken zoals TensorFlow, Theano en CNTK. In het boek wordt vooral gebruik gemaakt van TensorFlow, het is dan ook deze bibliotheek die in deze thesis gebruikt wordt om deep learing toe te passen op de gegeven dataset. (45)(55)(56)

Hoewel het niet noodzakelijk is, wordt het wel aangeraden om deep learnig uit te voeren op een NVIDIA *graphics processing unit* (GPU). De snelheid van de berekeningen zou hiermee 5 tot 10 keer sneller gaan dan wanneer gewerkt wordt met een *central processing unit* (CPU). Er was oorspronkelijk een Jetson Nano voorzien. Dit is een kleine maar krachtige computer die dankzij de sterke GPU zeer geschikt is voor het uitvoeren van deep learning. Omwille van de getroffen maatregelen naar aanleiding van het Covid-19 moesten de testen vanaf 17 maart van thuis uit worden uitgevoerd, waar geen directe toegang was tot de Jetson Nano. Dit werd opgelost door gebruik te maken van een *virtual network computing* (VNC). Dit maakt het mogelijk om een computer vanop afstand te bedienen. Het scherm van een andere computer kan op de eigen computer worden getoond en kan bediend worden met de muis en het toetsenbord. Door enkele moeilijkheden lukte het niet om de VCN te connecteren met de Jetson Nano. Dit was uiteindelijk wel mogelijk voor een Raspberry Pi, dit is ook een kleine computer, gelijkaardig aan de Jetson Nano maar met een minder sterke GPU, waardoor het minder geschikt is voor het uitvoeren van deep learning. De Raspbarry Pi zal echter voldoende goed presteren voor de nodige toepassingen van deze thesis. In het slechtste geval zullen de uit te voeren berekeningen iets meer tijd in beslag nemen met de Raspberry Pi dan wanneer ze met de Jetson Nano worden uitgevoerd. (45)(57)(58)

3.2 Methode

Het is onmogelijk om zonder veel ervaring met deze techniek direct aan de slag te gaan met zeer grote ingewikkelde datasets. Door deze techniek eerst op een kleinere dataset, die nog interpreteerbaar is voor mensen, toe te passen, kan de techniek volledig ontleed worden en wordt er een goed zicht bekomen op wat mogelijk is en wat niet. Wanneer duidelijk is wat deze techniek kan halen uit een kleine dataset, wordt de stap naar een grotere dataset gemaakt. Op die manier zal altijd met een groter deel van de dataset verder gegaan worden, waardoor er steeds meer informatie uit de dataset zal vrijkomen.

Deep learning kan zowel voor classificatie als voor regressie gebruikt worden. Voor het opbouwen van een geschikt neuraal netwerk moet voor beide gevallen hetzelfde stappenplan worden doorlopen. Hieronder wordt het stappenplan beschreven, samen met de verschillende beslissingen die bij elke stap gemaakt worden.

Vooraf

- Welke gegevens uit de dataset zijn van belang?
- Is de dataset geschikt?
- Hoe dataset optimaliseren?
- Hoe splitsen in training en test data?

Stap 1: Het collecteren en klaarmaken van de train en test data

- Gaat het om discrete of continue data?
- In welke vorm moet de data gevoed worden?

Stap 2: Het opbouwen van een netwerk van lagen

- Hoeveel verborgen lagen?
- Hoeveel verborgen eenheden?
- Welke activatie functie

Stap 3: Het configureren van het leerproces

- Welke loss functie?
- Welke optimizer?
- Welke metric?

Stap 4: Het valideren van het leerproces

- Welke validatie methode wordt er gebruikt?
- Hoeveel epochs?
- Welke Batch size?

In de volgende paragrafen worden de verschillende stappen besproken. Waar nodig wordt onderscheid gemaakt tussen een classificatie of een regressie. (45)

3.2.1 Vooraf

Om een neuraal netwerk op een goede manier op te bouwen, is het belangrijk dat de beschikbare data goed begrepen is. De dataset afkomstig van de drie verschillende analysetoestellen kan niet volledig gevoed worden aan het neuraal netwerk. Het is dus belangrijk om op voorhand te weten welke gegevens nodig zijn voor bepaalde classificaties en regressies.

Daarnaast is het belangrijk om na te gaan hoeveel stalen er beschikbaar zijn en wat de verdeling ervan is. Wanneer bij het uitvoeren van een classificatie bijvoorbeeld 90 % van alle stalen tot een bepaalde klasse behoren, kan het zijn dat een classificatie niet mogelijk is. Hoe meer data er beschikbaar is, hoe beter het neuraal netwerk zal presteren. De minimale hoeveelheid data nodig voor een goedwerkend netwerk hangt af van de complexiteit van de classificatie of de regressie. Of de dataset al dan niet geschikt is wordt dus enkel door empirisch onderzoek nagegaan.

Door op voorhand de dataset te verkennen, verloopt de interpretatie van deze resultaten vlotter. Bovendien kunnen op die manier misschien al enkele duidelijke uitschieters uit de dataset verwijderd worden. Wanneer de dataset niet geschikt blijkt te zijn, zijn er enkele oplossingen om de dataset te optimaliseren, zoals bijvoorbeeld het verzamelen van meer en gerichtere data, generen van synthetische stalen, stalen weglaten of stalen bewerken en hergebruiken.

Ook aan het opgebouwde netwerk kunnen enkele aanpassingen gedaan worden om de resultaten te verbeteren wanneer de dataset niet geschikt lijkt, zo kan gebruik van een andere metriek voor het valideren van het model helpen bij een slechte verdeling van de data of het toepassen van k-fold validatie bij een te kleine dataset. Deze laatsten komen later nog aan bod. (44)(55)

Bij het trainen van een neuraal netwerk wordt slechts een deel van de dataset (de training dataset) gebruikt. Het andere deel van de dataset is nodig voor het testen van het getrainde netwerk (de test dataset). Een laatste zaak dat nog moet beslist worden vooraleer het neuraal netwerk kan worden opgebouwd is op welke manier de dataset wordt opgesplitst in een test en een train dataset. Het is ook hiervoor belangrijk om vooraf te kijken naar de verdeling van de dataset. Wanneer de dataset slecht verdeeld is kan een slechte splitsing van train en test data voor zeer slechte resultaten zorgen. Dit is bijvoorbeeld het geval wanneer bij een classificatie geen enkel staal van een bepaalde klasse in de train dataset terecht komt. (45)

3.2.2 Het collecteren en klaarmaken van de train en test dataset

In Tabel 2 wordt een voorbeeld gegeven van hoe de train en test dataset er kan uitzien. Het gaat hier om een veelgebruikt voorbeeld voor regressie met deep learning waarbij de prijs van een huis voorspeld wordt aan de hand van enkele kenmerken zoals woonoppervlakte, bouwjaar, aanwezigheid van garage of een zwembad. In werkelijkheid zijn meer kenmerken nodig voor de prijs van een huis te voorspellen, In Tabel 2 wordt ook slechts een deel van de kenmerken van de oorspronkelijke dataset voor het voorspellen van de prijs van huizen weergegeven. (59)

Tabel 2: *Targets en Features* voor het voorspellen van huis prijzen.

Prijs	Oppervlakte	Bouwjaar	Garage oppervlakte	Zwembad oppervlakte
208500	8450	2003	548	0
181500	9600	1976	460	0
223500	11250	2001	608	0

In Tabel 2 blijkt dat de dataset uit twee verschillende delen bestaat. De in het groen aangeduide kolom bevat het kenmerk dat voorspeld moet worden, in dit geval de prijs van het huis. Dit wordt ook wel de *target* genoemd. De in het rood aangeduide kolommen bevatten de kenmerken waarmee de voorspelling gemaakt wordt, ook wel *features* genoemd.

Hetzelfde voorbeeld zou ook voor classificatie kunnen gebruikt worden. Hetgeen voorspeld moet worden is in dat geval niet meer de prijs van het huis, maar wel of de prijs bijvoorbeeld veel, gemiddeld of weinig is. In dit geval wordt de groene kolom vervangen door *labels,* bijvoorbeeld 1 voor een goedkoop huis, 2 voor een gemiddelde prijs en 3 voor een duur huis.

In een eerste stap moet de train en test data uit de tekstfiles met Python worden opgehaald. De training features, training labels of targets, test features en test labels of targets worden elk apart opgeslagen in een lijst. Vooraleer deze lijsten kunnen gebruikt worden om te modelleren moeten ze worden opgeslagen in zogenaamde multidimensionale *Numpy arrays* of *tensors*. Voor de stalen gebeurt dit via Python, gebruikmakend van het Numpy pakket met de functie “np.array”.

De schaal van de features en targets is een belangrijke factor. Het voeden van hoge waarden aan een neuraal netwerk kan tot gevolg hebben dat het model met grote weights leert. Een model met grote weights resulteert vaak in een traag of onstabiel leerproces en zorgt daardoor ook voor slechte prestaties en een hoge gevoeligheid voor invoerwaarden. Het schalen van de data kan de stabiliteit en de prestaties van het neuraal netwerk verbeteren. Twee veel gebruikte technieken voor het schalen van de data zijn normaliseren en standaardiseren. (60)

Bij normaliseren wordt de data met behulp van vergelijking 3.1 omgezet naar getallen tussen 0 en 1.

$$y_{norm} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3.1}$$

Bij standaardiseren wordt met behulp van vergelijking 3.2 een dataset bekomen met een gemiddelde van 0 en een standaarddeviatie van 1. (57)

$$y_{stand} = \frac{x - x_{gem}}{x_{stdev}} \tag{3.2}$$

3.2.3 Opbouwen van een neuraal netwerk

De bouwstenen van een neuraal netwerk zijn de verschillende op elkaar gestapelde lagen. Zo een laag is een gegevensverwerkingsmodule die als input één of meer tensoren heeft en na een bepaalde bewerking een output levert die ook kan bestaan uit één of meer tensoren. Naargelang de gebuikte tensorformaten en de verschillende soorten gegevens worden andere lagen gebruikt. Eenvoudige vectorgegevens die zoals het voorbeeld uit Tabel 2 worden opgeslagen in 2D-tensoren met een vorm (stalen, kenmerken) worden in de meeste gevallen verwerkt met behulp van een eenvoudige stapel volledig verbonden lagen of dense layers genoemd.

Zoals in het stappenplan wordt aangegeven moeten bij het creëren van een geschikt netwerk drie belangrijke beslissingen gemaakt worden.

1. Hoeveel lagen moeten gebruikt worden?
2. Welke activatie functie moet bij elke laag gebruikt worden?
3. Hoeveel verborgen eenheden (hidden units) zijn er nodig per laag?

In Figuur 28 wordt een voorbeeld getoond hoe zo een netwerk in Python wordt opgebouwd. De antwoorden op de drie bovenstaande vragen hangen af van het doel en de complexiteit ervan. De activatie functie van de laatste laag wordt gekozen naargelang het doel dat voor ogen is. Bij een classificatie zal bijvoorbeeld een andere activatie functie in de laatste laag nodig zijn dan bij regressie. De activatie functie van de andere lagen, het aantal lagen en het aantal verborgen eenheden (afzonderlijke neuronen binnen een laag) zijn parameters die afhangen van de complexiteit en zijn enkel via trial and error te bepalen. (45)

```
#------------------Building Network------------------
model = models.Sequential()
model.add(layers.Dense(64, activation = 'relu', input_shape =(1,)))
model.add(layers.Dense(64, activation = 'relu'))
model.add(layers.Dense(1, activation = 'sigmoid'))
```

Figuur 28: Voorbeeld van hoe een neuraal netwerk wordt opgebouwd met behulp van Python.

Het netwerk bestaat hier uit drie volledig verbonden lagen (*dense layers*). Bij de eerste laag wordt gebruik gemaakt van de activatiefunctie “relu”. Dit is een van de meest gebruikte

activatiefuncties bij deep learning. Het getal 64, in het groen afgebeeld op Figuur 28, wijst op het aantal verborgen eenheden. Hoe meer verborgen eenheden hoe beter het netwerk complexe problemen kan oplossen, maar ook hoe groter de kans dat het netwerk ongewenste patronen zal vinden en daarmee dus overfitting zal optreden.

Elke laag accepteert enkel input-tensors van een bepaalde vorm en retourneert output-tensoren van een bepaalde vorm. Door een input_shape (1,) aan te geven wordt een laag gecreëerd die een 2D tensor als input accepteert waarvan de eerste dimensie 1 is. Deze laag kan enkel aan een tweede laag gekoppeld worden die een 64-dimensionale vector verwacht, aangezien de 64 verborgen eenheden zorgen voor een 64-dimensionale output. De tweede laag is exact hetzelfde als de eerste laag. De input_shape moet hier niet meer gespecifieerd worden, de input vorm wordt automatisch afgeleid van de output vorm van de vorige laag. De tweede laag bevat opnieuw 64 verborgen eenheden, wat zorgt voor een 64 dimensionele output. De derde laag heeft maar één verborgen eenheid. Deze laag zal dus als output een scalair geven. De activatiefunctie die hier wordt gebruikt is de sigmoïd functie. De output van deze laag is een getal tussen 0 en 1 dat aangeeft hoe waarschijnlijk het staal een label 1 heeft.

Classificatie

Het netwerk uit Figuur 28 is dus geschikt voor een classificatie in slechts twee verschillende klassen. Aangezien door de sigmoïd activatie functie een scalair als output bekomen wordt dat de kans aangeeft dat het staal tot klasse 1 behoort, is ook direct bekend wat de kans is dat het staal tot de andere klasse behoort. Bij een classificatie in meerdere klassen geeft deze output niet genoeg informatie over de verschillende klassen. Er wordt in dat geval met een andere activatie functie gewerkt in de laatste laag namelijk “softmax”. Deze activatiefunctie geeft de kansverdeling over de verschillende klassen. De tweede aanpassing is het aantal verborgen eenheden van de laatste laag, dit moet evenveel zijn als het aantal klassen. In Figuur 29 wordt een voorbeeld gegeven van de opbouw van een neuraal netwerk voor de classificatie in drie verschillende klassen. (45)

```
#-------------------Building Network-------------------

model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(1,)))
model.add(layers.Dense(16, activation='tanh'))
model.add(layers.Dense(3, activation='softmax'))
```

Figuur 29: Opbouw van neuraal netwerk voor multinomiale classificatie in drie klassen.

Aangezien de drie verborgen eenheden zorgen voor een driedimensionale output zal de “softmax” activatie functie een kansverdeling over de drie verschillende klassen genereren. De eerste waarde in de driedimensionale outputvector geeft de kans aan dat het staal tot klasse 1 behoort, de tweede waarde geeft de kans dat het tot klasse 2 behoort en de derde waarde geeft aan in hoeverre het staal tot klasse 3 behoort. De som van deze drie getallen is steeds 1, aangezien de kans dat het staal tot één van de drie klassen behoort 100 % is. (45)

Regressie

Een neuraal netwerk opgebouwd voor regressie wordt afgebeeld in Figuur 30.

```
#-------------------Building Network-------------------

model = models.Sequential()
model.add(layers.Dense(32, activation='relu', input_shape=(39,)))
model.add(layers.Dense(32, activation='relu'))
model.add(layers.Dense(1))
```

Figuur 30: Opbouw van neuraal netwerk voor regressie.

Hier eindigt het netwerk met één enkele verbogen eenheid en zonder een activatie functie. Dit wil zeggen dat de laatste laag volledig lineair is, waardoor het netwerk een waarde kan voorspellen zonder dat het beperkt wordt in een bepaald bereik.

In Tabel 3 wordt nog eens een overzicht gegeven van de geschikte activatie functie van de laatste laag voor elk doel apart.

Tabel 3: Overzicht van activatie functie van laatste laag per doel.

Doel	Activatie functie laatste laag
Classificatie in twee klassen	Sigmoid
Classificatie in meerdere klassen	Softmax
Regressie	Geen

3.2.4 Opstellen van een training model

In een volgende stap moet het neuraal netwerk getraind worden. Zoals in het stappenplan wordt aangegeven, moet hiervoor een geschikte verlies functie (*loss function*), optimalisator (*optimizer)* en metriek gekozen worden. De verlies functie is net zoals de activatie functie van de laatste laag afhankelijk van het vooropgestelde doel en zijn weergegeven in Tabel 4.

Tabel 4: Overzicht van verlies functie per doel.

Doel	Verlies functie
Classificatie in twee klassen	Binary Cross Entropy
Classificatie in meerdere klassen	Categorical Cross Entropy
Regressie	Mean Squared Error (MSE)

In het boek *Deep Learning met Python* van F. Chollet (45), dat als richtlijn wordt gebruikt, wordt vermeld dat het gebruik van de optimalisator “RMSprop” met de standaard ingestelde learning rate over het algemeen de beste keuze is, ongeacht het doel of het model. Er bestaan nog andere optimalisators zoals “Adam” of “Adagrad”. Ook is het mogelijk om de leersnelheid bij elke optimalisator aan te passen.

Er bestaan ook verschillende metrieken om het model te valideren. Voor classificatie problemen wordt vaak *classification accuracy* gebruikt. Deze metriek geeft aan hoeveel van alle stalen juist werden geclassificeerd. Een hoge *accuracy* wijst niet in alle gevallen op een goed model. Wanneer bijvoorbeeld 80 % van de stalen in een bepaalde klassen behoort, dan zou een model dat alle aanwezige stalen in die klasse indeelt een accuraatheid van 80 %

hebben. Deze accuraatheid geeft daarom een verkeerd beeld over de juistheid van dit model. In die gevallen moet extra kritisch naar de accuraatheid gekeken worden of kan gekozen worden voor een andere metriek. (45)

Aangezien er bij regressie niet meer in klassen wordt verdeeld moet er een andere metriek gebruikt worden. Een veelgebruikte metriek hierbij is de *Mean Absolute Error* (MAE). Dit geeft de absolute waarde van het verschil tussen de door het netwerk voorspelde waarde en de werkelijke waarde weer. (45)

In Figuur 31 wordt een voorbeeld gegeven van hoe een model met Python getraind wordt.

```
#-------------------Training Model-------------------
model.compile(optimizer = 'rmsprop', loss = 'binary_crossentropy', metrics= ['acc'])
```

Figuur 31: Script voor het trainen van een netwerk.

3.2.5 Het valideren van het training model

Er bestaan meerdere manieren om het training model te evalueren. De eenvoudigste manier is de *simple hold-out validation*, Bij deze methode wordt de trainingsdata opnieuw opgesplitst in een deel voor training en een deel voor de validatie. Het eerste deel van de dataset wordt dus gebruikt om het neuraal netwerk te trainen en een ander deel van de dataset wordt gebruikt om deze training te valideren.

Het model kan gevalideerd worden met behulp van de functie '*model.fit'* zoals weergegeven is Figuur 32.

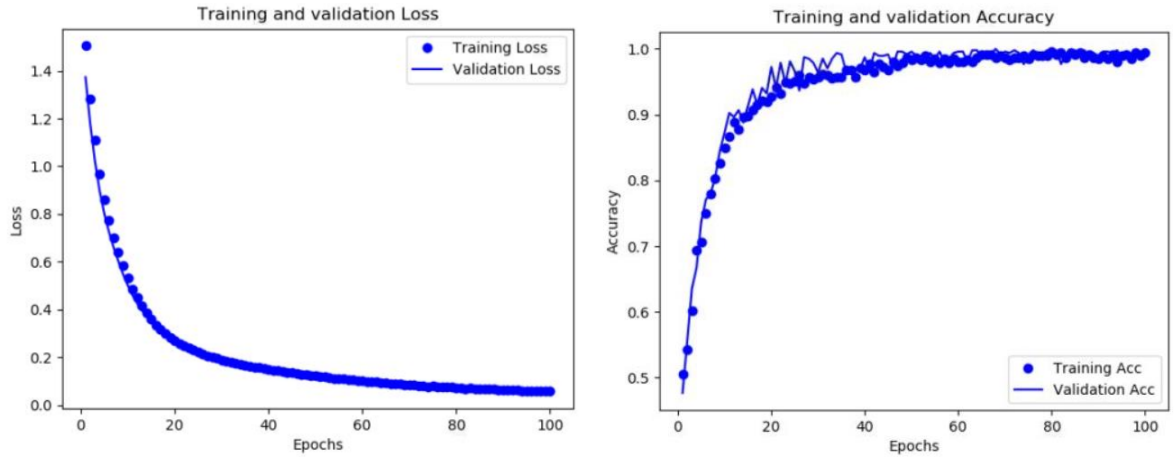
```
history = model.fit(partial_x_train, partial_y_train, epochs=50, batch_size=20, validation_data = (x_val, y_val))
```

Figuur 32: Script voor het valideren van het training model.

Bij het valideren van het model moeten ook enkele beslissingen gemaakt worden, namelijk het aantal *epoch* en de grootte van de batch. De grootte van de batch is het aantal stalen die het model doorloopt vooraleer met behulp van de verlies functie het verschil tussen de werkelijke waarden en de voorspelde waarden berekend wordt, waarna de weights van het model worden geüpdatet.

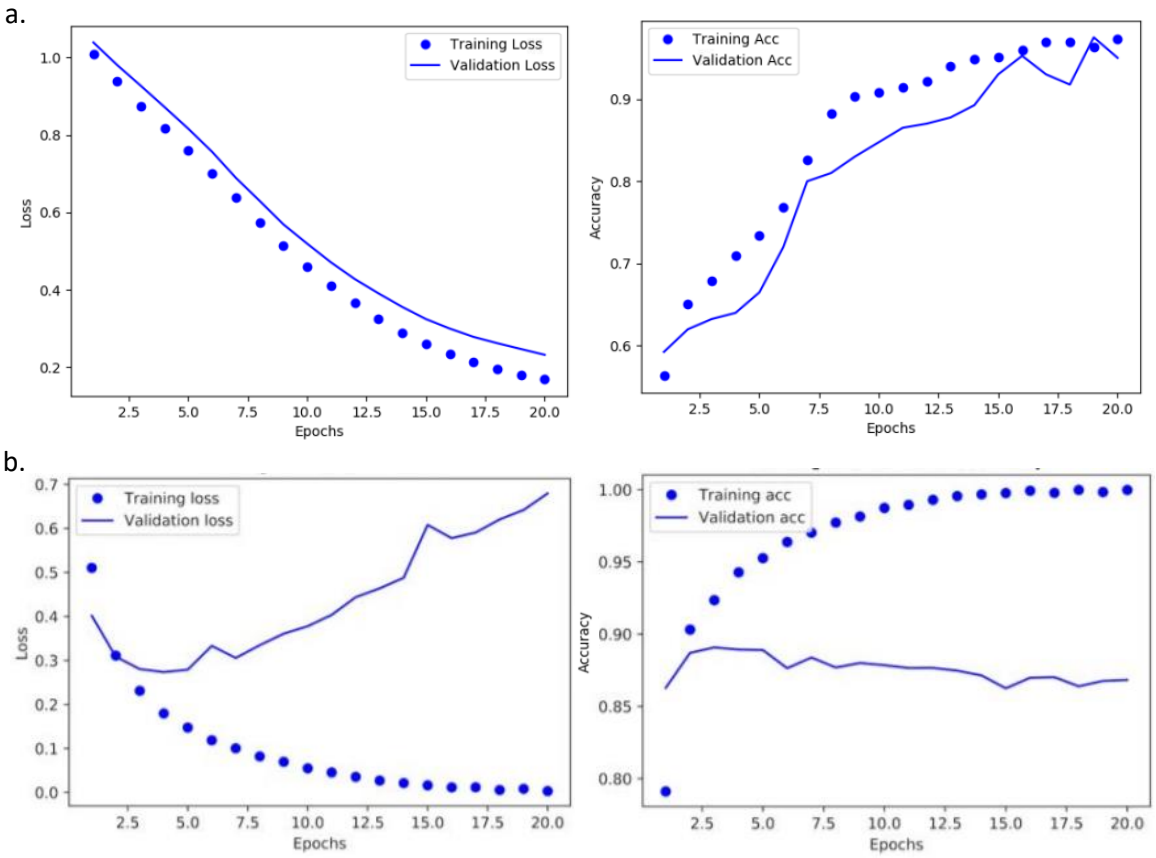
De dataset kan in één of meer batches verdeeld worden. Het aantal epochs geeft weer hoeveel keer het neuraal netwerk de volledige dataset doorloopt. Bij het doorlopen van één epoch heeft elk staal uit de trainingsdata dus eenmaal invloed gehad op de update van de weights. De grote van de batch en het aantal epochs zijn ook twee parameters die via trial and error bepaald worden en kunnen aangepast worden om het model te verbeteren. (52)

De training en de validatie ervan wordt geëvalueerd om te weten op welke manier de verschillende parameters moeten aangepast worden zodat de prestaties van het model verbeteren. Bij classificatie gebeurt deze evaluatie aan de hand van het plotten van de *training* en *validation loss* samen met het plotten van de *training* en *validation accuracy*. De *training* en validation loss moet tijdens het trainen zo laag mogelijk zijn, terwijl de training en validation accuracy daarbij zo hoog mogelijk moet zijn. Vaak worden deze parameters in een grafiek uitgezet in functie van het aantal epochs. In Figuur 33 wordt een duidelijk voorbeeld gegeven van hoe de training en validation loss en accuracy gedurende het trainen van een model voor classificatie eruitzien.



Figuur 33: Voorbeeld van het verloop van de (a) *training* en *validation loss* en *(b) training* en *validation accuracy.*

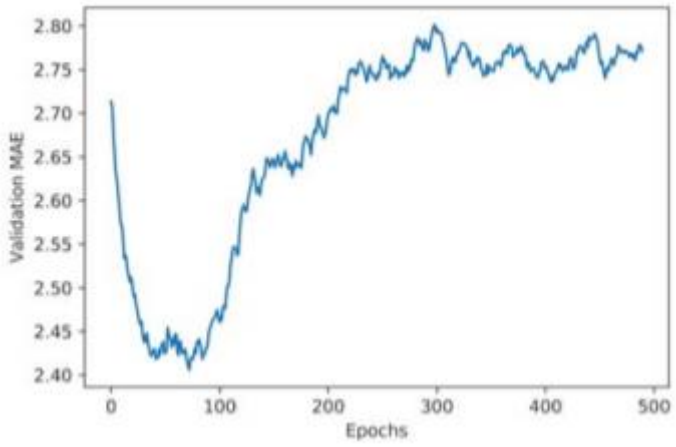
In deze figuur is duidelijk te zien dat zowel de training en validatie loss daalt bij elke epoch. Eerder werd al beschreven dat de loss function aangeeft hoe ver de voorspelde output verwijderd is van wat verwacht wordt. Een daling van de loss geeft dus aan dat het model de trainingsdata steeds beter kan voorspellen. Dit is ook te zien aan de curve van de accuracy. Deze wordt steeds hoger, wat ook aangeeft dat er steeds meer stalen in de juiste klasse worden ingedeeld. De daling van de loss per epoch is logisch, aangezien het net deze loss function is die geminimaliseerd wordt om het model te trainen. De curves van de training geven dus weer hoe goed het model getraind is, terwijl de curves van de validatie aangeven hoe goed het model presteert met nieuwe data. Figuur 33 toont aan dat validatie van het model telkens verbetert tot de loss curves een waarde van ongeveer 0,1 bereiken. Er wordt een accuraatheid van bijna 1 bereikt. De curves van de training validatie lopen bijna gelijk met die van de training data, wat aangeeft dat het model ook goed presteert met nieuwe, nooit eerder geziene data. In Figuur 34 worden enkele voorbeelden gegeven van hoe de curves eruitzien wanneer het model niet goed zal presteren.



Figuur 34: Voorbeelden van het verloop van de *training* en *validation loss* en *accuracy* (a) *onderfitting*. (b) *overfitting.* (45)

In de eerste grafiek uit Figuur 34 (a) is te zien dat zowel de training als de validatie loss niet stabiliseert naar een bepaalde waarde. Tot de laatste epoch blijkt de loss te dalen. Dit geeft aan dat het model nog niet alle relevante patronen in de trainingsdata modelleert en dat er nog verbetering mogelijk is. Er treedt in dit geval underfitting op. In de eerste grafiek uit Figuur 34 (b) is te zien dat de training loss daalt naarmate het aantal epochs en stabiliseert ongeveer bij een waarde van 0,0. De validatie loss daarentegen begint vanaf ongeveer de vierde epoch terug te stijgen. Hierdoor daalt ook de accuracy wat te zien is op de tweede grafiek van Figuur 34 (b). Het model gaat vanaf dat punt wel steeds beter presteren op de trainingsdata, maar de prestaties voor de validatie data worden altijd maar slechter. Dit wijst erop dat overfitting optreedt. (45)

Bij regressie wordt enkel gekeken naar de MAE tijdens de validatie per epoch. Deze wordt weergegeven in Figuur 35.

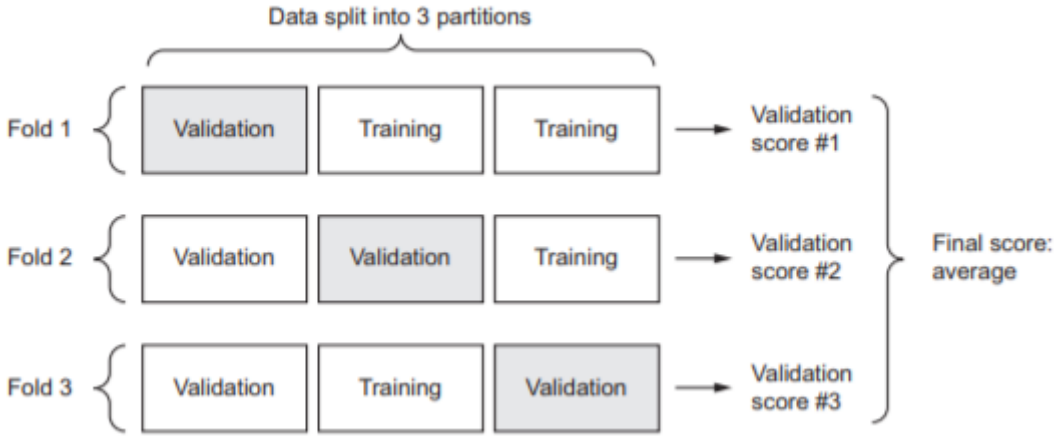


Figuur 35: Verloop van MAE van de validatie per epoch. (45)

Hierop is te zien dat de MAE daalt tot ongeveer 80 epochs, hierna stijgt deze terug wat aangeeft dat het model vanaf dan minder presteert.

Na elke validatie worden bepaalde parameters zoals aantal lagen, aantal verborgen eenheden, aantal epoch en de grootte van de batch aangepast om de prestaties ervan te verbeteren. Door het nieuwe model te valideren en opnieuw aan te passen en dit telkens te herhalen wordt uiteindelijk een optimaal model bekomen, waarbij geen underfitting of overfitting plaatsvindt. De reden waarom het model gevalideerd wordt met een aparte validatie set en niet rechtstreeks met de test set is dat per validatie een deel van de informatie van de validatie set “lekt” in het model. Hoe meer het model gevalideerd wordt, hoe meer het informatie krijgt over de validatie set. Dit heeft als gevolg dat het model steeds beter zal presteren voor de validatie data, aangezien het model hiervoor werd geoptimaliseerd. De bedoeling is om een model te creëren dat goed presteert op compleet nieuwe data. Het optimale model voor de validatiedata moet dus nog eenmaal getest worden op data die nog nooit eerder gebruikt werd namelijk de testdata.

Naast de *simple hold-out validation* bestaan er ook nog ander manieren om het model te valideren zoals *K-fold validation* en *iterated K-fold validation with shuffling*, bij deze twee validatiemethoden wordt de dataset in een K aantal delen verdeeld, de training en validatie gebeurt dan K keer achter elkaar telkens met een ander deel van de dataset als validatie data, zoals te zien in Figuur 36. Na de K validaties wordt de gemiddelde MAE bepaald. (45)



Figuur 36: Werking van 3-cross fold validatie. (44)

3.2.6 Getraind netwerk gebruiken om nieuwe data te voorspellen

Het getrainde neuraal netwerk moet vervolgens nog éénmaal getest worden met data die tot nog toe nog niet aan het neuraal netwerk gevoed werd, namelijk de test dataset, om te controleren of het netwerk ook voor compleet nieuwe data goed presteert. Wanneer dit het geval is kan het neuraal netwerk gebruikt worden voor het voorspellen van nieuwe data. (45)

Het evalueren van het getrainde model en het voorspellen van nieuwe waarden gebeurt met Python via de functies weergegeven in Figuur 37.

```
#-------------------Generate predictions on new data-------------------
results = model.evaluate(x_test,y_test)
predictions = model.predict(x_test)
```

Figuur 37: Voorspellen van nieuwe waarden met getraind model.

Met de functie “model.evaluate” worden de prestaties van het getrainde model op nieuwe data geëvalueerd aan de hand van de vooraf gekozen metriek. Met de functie “model.predict” worden de waarden van onbekende stalen voorspeld.

3.2.7 Classificaties en regressies in dit onderzoek

In dit onderzoek wordt nagegaan of verschillende classificaties en regressies met betrekking tot de dataset kunnen uitgevoerd worden met een neuraal netwerk. In Tabel 5 wordt een overzicht gegeven van de verschillende classificaties die worden uitgevoerd.

Tabel 5: Verschillende classificaties die in het onderzoek worden uitgevoerd.

Classificatie	Features	Aantal klassen
Naargelang de Y-waarden (basiswitheid)	Y-waarden	2
Naargelang de Y-waarden (basiswitheid)	Y-waarden	3
Naargelang de Y-waarden (basiswitheid)	Y-waarden	5
Naargelang de Y-waarden (basiswitheid)	X-, Y- en Z-waarden	2
Naargelang de Y-waarden (basiswitheid)	Emissie-spectrum	2
Naargelang WI_{Ganz} (Witheidindex)	X-, Y- en Z-waarden	3
Naargelang WI_{Ganz} (Witheidindex)	Emissie-spectra	3
Naargelang hoeveelheid optische witmaker (FWA)	X-, Y- en Z-waarden	3
Naargelang hoeveelheid optische witmaker (FWA)	Emissie-spectra	3

De eerste classificatie is op basis van de Y-waarden. Deze waarde geeft de basiswitheid van het textiel weer en wordt gemeten met een spectrofotometer. De indeling die door Christeyns werd opgelegd is de volgende, alle stalen met een Y-waarde hoger dan 80 worden aanzien als een staal met een goede basiswitheid, stalen met een Y-waarde lager dan 80 hebben een slechte basiswitheid.

Na het uitvoeren van deze classificatie wordt er overgegaan naar classificaties in meerdere klassen. Eerst in drie verschillende klassen waarbij stalen met een Y-waarde onder 75 een slechte basiswitheid hebben, stalen met een Y-waarde boven 85 een goede en de stalen met een Y-waarde ertussen hebben een gemiddelde basiswitheid. De verschillende klassen bij de classificatie in vijf klassen zijn een Y-waarde van 0 tot 50, tussen 50 en 70, tussen 70 en 80, tussen 80 en 85 en tenslotte alles boven 85. Ook dit zijn indelingen die opgelegd werden door Christeyns.

De classificatie met behulp van één enkele feature heeft vooral als doel om vertrouwd te geraken met deep learning en neurale netwerken. Bij het classificeren van stalen in een groep met een hoge en een groep met een lage Y-waarde is niet per se een computer nodig. Wanneer het gaat om een te groot aantal Y-waarde om handmatig in de klassen te verdelen kan dit ook gebeuren door op een traditionele manier te programmeren en dus zelf de regels in de computer in te geven. Het is zeker niet noodzakelijk om dit classificatieprobleem met behulp van deep learning op te lossen. De techniek wordt wel interessant wanneer meerdere features tegelijkertijd een invloed zouden hebben op de klasse waartoe het staal behoort. Het wordt al veel moeilijker om de classificatie dan uit te voeren door traditioneel programmeren en het wordt onmogelijk wanneer de invloed van de verschillende features nog niet gekend is.

In een volgende stap wordt de classificatie dus uitgebreid naar meerdere features. In het eerste geval naar X-, Y- en Z-waarden, vervolgens wordt overgestapt naar het volledige reflectie-spectrum wat bestaat uit 49 golflengtes.

Dezelfde features worden gebruikt voor de classificatie op basis van WI_{Ganz}. WI_{Ganz} is een witheidindex waarbij ook de aanwezigheid van optische witmakers en bluing agents in rekening wordt gebracht. Er wordt hier onderzocht of het mogelijk is om stalen te classificeren op basis van de WI_{Ganz}, zonder dat hiervoor de WI_{Ganz} zelf aan het netwerk wordt gevoed.

Voorlopig zijn alle gegevens afkomstig van één enkel analysetoestel namelijk de Konica Minolta spectrofotometer. In de volgende stap wordt nagegaan of het ook mogelijk is om gegevens van HPLC te voorspellen aan de hand van data van de spectrofotometer. De classificatie gebeurt naargelang de hoeveelheid optische witmaker (FWA) op het staal dat bepaald wordt met HPLC. De gebruikte features zijn ook hier weer de X-, Y- en Z-waarden en de volledige spectra.

In Bijlage 1 en Bijlage 2 wordt een voorbeeld gegeven van een script geschreven in Python van respectievelijk een classificatie in twee klassen en in meerdere klassen

Tot nu toe werd enkel gekeken naar classificatieproblemen. Is een staal goed of slecht, of bevat het veel of weinig van een bepaald product. Wanneer deep learning wordt toegepast voor regressie zal het neuraal netwerk voorspellen hoe goed een staal is of hoeveel van een bepaald product op het staal aanwezig is.

In totaal zullen er op de witheidsdata vier regressies getest worden. Er zal worden nagegaan of het mogelijk is om de exacte waarde van WI_{Ganz} te voorspellen en dit met zowel de X-, Y-, Z-waarden als met de volledige spectra. Vervolgens wordt onderzocht of de hoeveelheid van

de aanwezige optische witmakers kan voorspeld worden aan de hand van de X-, Y-, Z-waarden en de spectra. In Tabel 6 wordt een overzicht gegeven van de verschillende regressies.

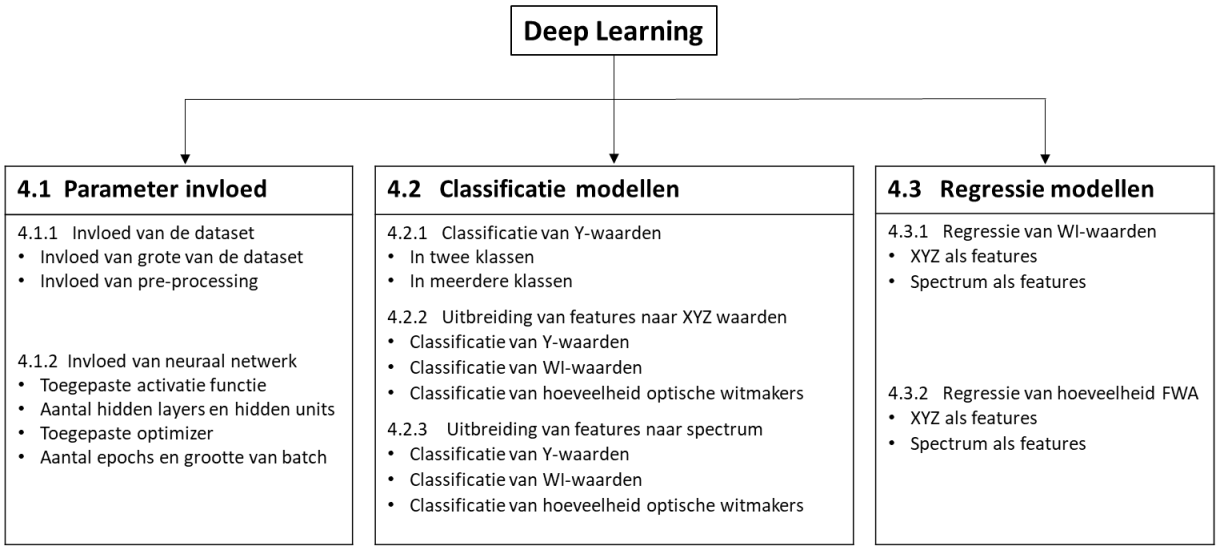
Tabel 6: Verschillende regressies die in het onderzoek worden uitgevoerd.

Regressie	Features
Naargelang WI_{Ganz} (Witheidindex)	X-, Y- en Z-waarden
Naargelang WI_{Ganz} (Witheidindex)	Reflectie-spectra
Naargelang hoeveelheid optische witmaker (FWA)	X-, Y- en Z-waarden
Naargelang hoeveelheid optische witmaker (FWA)	Reflectie-spectra

In Bijlage 3 wordt een voorbeeld gegeven van een script voor het uitvoeren van regressie en door gebruik te maken van de K-fold validatie methode.

4 RESULTATEN EN DISCUSSIE

In deze thesis wordt de techniek deep learning gebruikt voor de verwerking van de beschikbare dataset van de textielverzorgingsafdeling bij Christeyns. De techniek wordt gebruikt bij het toepassen van zowel classificaties als regressies op de dataset. In Figuur 38 wordt een overzicht gegeven van de verschillende zaken die tijdens het onderzoek onderzocht worden.



Figuur 38: Overzicht van verschillende delen van het onderzoek met betrekking op deep learning.

Het onderzoek wordt opgesplitst in 3 delen. In het eerste deel wordt aandacht besteed aan de invloed van verschillende parameter. Hierbij wordt gekeken naar de invloed van de dataset zelf en de invloed van de verschillende parameters van het neuraal netwerk.

Vervolgens worden de resultaten van de uitgevoerde classificaties en regressies besproken. Bij de classificaties wordt er gestart met classificatie van de dataset in klassen met een hoge en een lage Y-waarde en dit met enkel de Y-waarden als feature. Vervolgens wordt het aantal features uitgebreid naar de X-, Y- en Z-waarden en naar het volledige reflectie-spectrum. Met deze features worden ook classificaties op basis van WI_{Ganz}-waarden en hoeveelheid optische witmakers uitgevoerd.

Deze twee laatste classificaties worden uitgebreid naar regressies waarbij de WI_{Ganz}-waarde en de hoeveelheid optische witmakers wordt voorspeld en dit met zowel de X-, Y- en Z-waarden als het volledig spectrum. Hierbij wordt ook nog extra aandacht besteed aan het belang van pre-processing methoden.

4.1 Parameter invloed

4.1.1 Invloed van dataset

4.1.1.1 Invloed van de grootte van de dataset

Een veel gestelde vraag bij het toepassen van deep learning op een bepaalde dataset is hoeveel data er effectief nodig is om het vooropgestelde doel te bereiken. Deze vraag kan niet direct beantwoord worden, het hangt namelijk sterk af van de complexiteit van het doel en van de complexiteit van het gebruikte algoritme. Hier zal de invloed van de grootte van de dataset worden nagegaan voor de classificatie van stalen in twee klassen, namelijk stalen met een goede basiswitheid en stalen met een slechte basiswitheid.

Niet alleen de grote van de dataset, maar ook de verdeling ervan kan een belangrijke rol spelen voor het al dan niet bekomen van een goed resultaat bij het leerproces. Wanneer bijvoorbeeld slechts 10 % van een bepaalde dataset een slechte basiswitheid heeft, zou het leerproces moeilijker kunnen verlopen dan wanneer de dataset evenveel stalen met een goede en een slechte basiswitheid heeft.

Een ander probleem komt voor wanneer er niet voldoende stalen in de train data aanwezig zijn met een Y-waarde rond 80. Deze Y-waarde is namelijk de scheiding tussen slechte en goede basiswitheid. Wanneer er bijvoorbeeld geen stalen aanwezig zijn met een basiswitheid tussen 75 en 85, zal het getrainde neuraal netwerk niet in staat zijn om onbekende stalen met een basiswitheid tussen deze waarden juist in te delen.

Om een beeld te krijgen van de invloed van de grote en de verdeling van de dataset worden acht verschillende datasets gebruikt voor het trainen van een bepaald neuraal netwerk en dit voor het classificeren van de dataset in stalen met een Y-waarde hoger of lager dan 80 (goede of slechte basiswitheid). In Tabel 7 wordt de informatie van deze datasets weergegeven. Het doel is om na te gaan wat er nog aan de dataset verbeterd kan worden om betere resultaten te krijgen bij het trainen van een neuraal netwerk.

Tabel 7: Gebruikte datasets om de invloed ervan na te gaan op het neuraal netwerk.

Dataset	Totaal aantal stalen	Bereik Y-waarde	Gemiddelde Y-waarde	Standaard-deviatie	Procent slechte basiswitheid	Verdeling
1	945	3,4 - 91,7	80,7	10,4	0,26	Slecht
2	460	3,4 - 90,8	81,3	10,2	0,24	Slecht
3	460	3,4 - 91,5	76,8	13,6	0,49	Goed
4	230	72,5 - 90,8	84,4	2,8	0,06	Slecht
5	230	3,4 -91,5	77,0	12,4	0,50	Goed
6	50	89,5 - 46,5	78,4	7,8	0,44	Goed
7	658	3,4 - 91,7	81,1	12,4	0,17	Slecht
8	287	75,0 - 82,5	79,7	2,1	0,47	Goed

Met deze acht datasets worden drie verschillende zaken onderzocht.

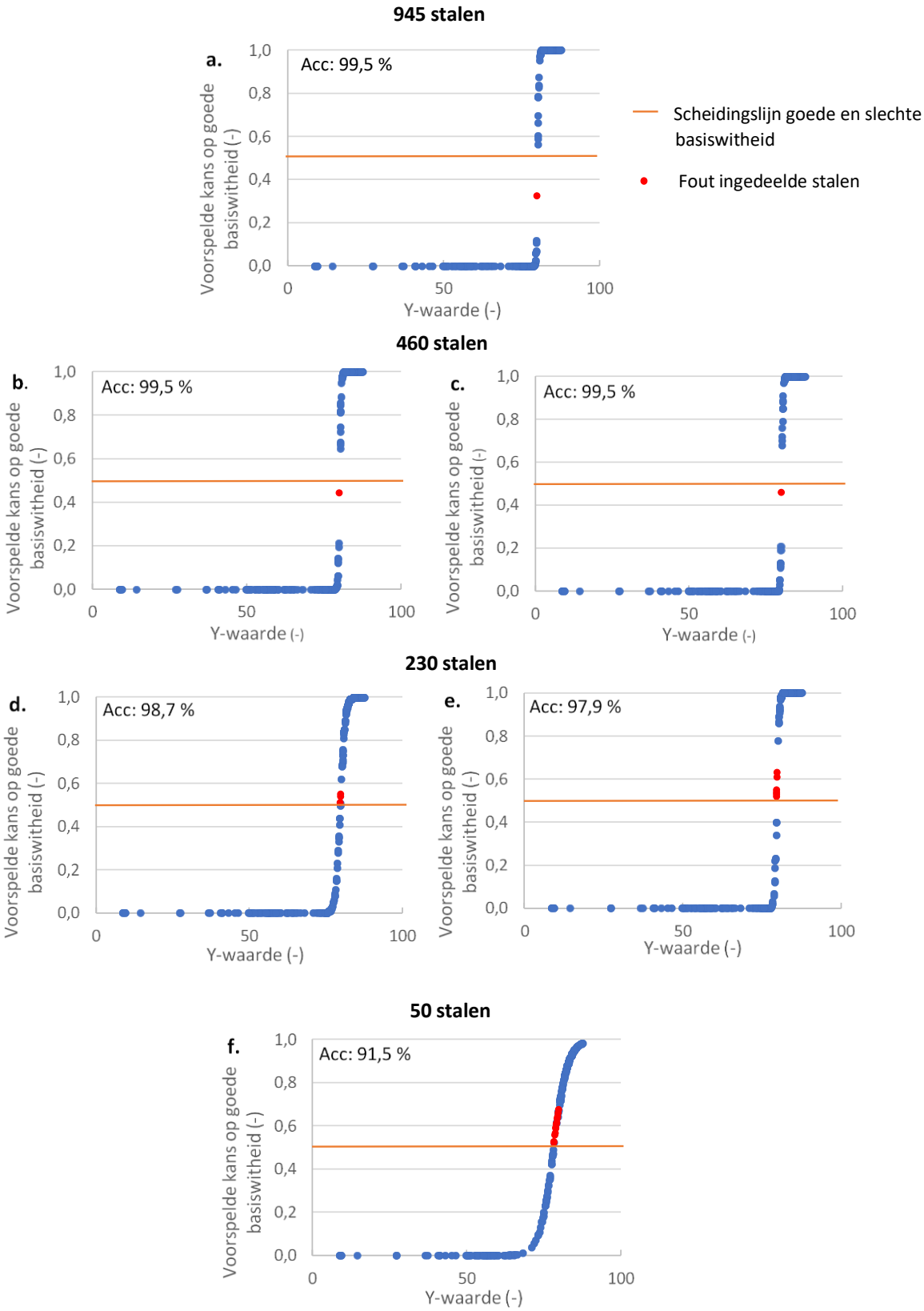
1. De invloed van een steeds kleinere dataset. De oorspronkelijke dataset (dataset 1) bevat 945 stalen. De invloed van de grootte van de dataset wordt nagegaan door te testen met steeds kleinere datasets van 460, 230 en 50 stalen.

2. De invloed van een goede verdeling of slechte verdeling van de dataset. Hierbij wordt nagegaan of de hoeveelheid stalen per klasse een belangrijke invloed heeft op de resultaten van het neuraal netwerk. Een dataset met een goede verdeling bevat ongeveer evenveel stalen in alle klassen. Dataset waarbij een bepaalde categorie slechts weinig stalen bevat wordt aanzien als een dataset met een slechte verdeling.

3. De invloed van de stalen dicht bij het overgangspunt tussen een goede en een slechte basiswitheid. Dit wordt nagegaan aan de hand van de twee laatste datasets 7 en 8. Bij dataset 7 zijn alle stalen met een basiswitheid tussen 75 en 82,5 weg gelaten. Dataset 8 bevat enkel de stalen tussen 75 en 82,5.

De resultaten worden geëvalueerd aan de hand van de accuraatheid van het model en aan de hand van een grafiek waarin de voorspelde kans dat het staal een goede basiswitheid heeft wordt uitgezet in functie van de werkelijke Y-waarde.

De accuraatheid van het model geeft aan hoeveel van alle stalen van de testdata in de juiste categorie worden ingedeeld. Een staal wordt tot de categorie ingedeeld waarvan de voorspelde kans dat het ertoe behoort het grootst is. Een staal dat bijvoorbeeld 40 % kans heeft om een goede basiswitheid te hebben en daardoor automatisch 60 % kans heeft dat het een slechte basiswitheid heeft, zal in de categorie van slechte basiswitheid worden ingedeeld. In Figuur 39 worden de grafieken met de voorspelde kans in functie van de werkelijke Y-waarden van de eerste zes datasets uit Tabel 7 weergegeven. De scheidingslijn tussen de twee verschillende klassen wordt aangegeven met een oranje lijn. Alle stalen boven die lijn worden ingedeeld als stalen met een goede basiswitheid, de stalen onder deze lijn worden ingedeeld als stalen met een slechte basiswitheid. Bij elke dataset worden de stalen die door het neuraal netwerk in de verkeerde klasse worden ingedeeld aangeduid in het rood. Daarnaast wordt ook de accuraatheid van het model en het aantal fout ingedeelde stalen bij elke grafiek meegegeven.



Figuur 39: Voorspellingen van het model voor de test data met behulp van zes verschillende train datasets. (a) Dataset 1. (b) Dataset 2. (c) Dataset 3. (d) Dataset 4. (e) Dataset 5. (f) Dataset 6.

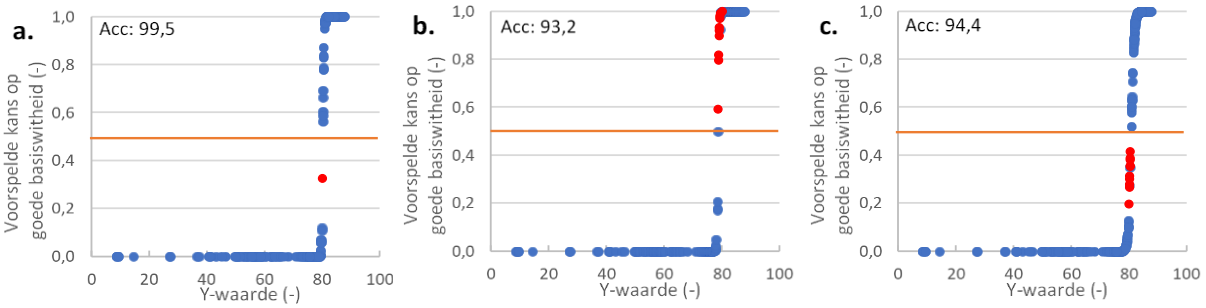
Figuur 39 (a), (b) en (c) zien er zeer gelijkaardig uit. Bij alle drie de datasets wordt een accuraatheid van 99,5 gehaald, wat wil zeggen dat slechts één staal van de test data in de verkeerde klasse wordt ingedeeld. In de drie gevallen gaat het over het staal met een basiswitheid van 80,01 dat in de categorie met een slechte basiswitheid wordt ingedeeld. Bij alle drie de datasets is de kans dat een staal met een Y-waarde onder 79 een goede basiswitheid heeft 0 %. Vanaf een Y-waarde van 79 stijgt de kans op een goede basiswitheid snel. Wanneer een staal een Y-waarde hoger dan 81 heeft is de kans dat ze tot de klasse met een goede basiswitheid wordt ingedeeld 100 %. Hieruit blijkt dat het neuraal netwerk in dit geval evengoed presteert wanneer het getraind wordt met een dataset van 460 stalen dan met een dataset van 945 stalen. De resultaten zijn ook niet significant beter wanneer een dataset met een goede verdeling wordt gebruikt in vergelijking met een dataset met een slechtere verdeling.

Wanneer nog eens de helft van de dataset gebruikt wordt (dataset 4 en 5 uit Tabel 7), wordt wel een lagere accuraatheid vastgesteld, namelijk 98,7 % voor dataset 4 en 97,9 % voor dataset 5. Bij de dataset met een slechte verdeling (6 % van de stalen heeft een slechte basiswitheid) worden 3 stalen met een Y-waarde boven 80 ingedeeld als stalen met een slechte basiswitheid. Bij de dataset met een goede verdeling zijn dat er 6. Daarnaast wordt het aantal stalen dat niet met 100 % zekerheid in de juiste klasse wordt ingedeeld groter. Dit is meer het geval bij de dataset die slecht verdeeld is, de kans dat een staal een goede basiswitheid heeft stijgt hier al vanaf een Y-waarde van 75 en is 100 % vanaf een Y-waarde van 84. Bij de dataset waarbij elke categorie evenveel stalen bevat, liggen deze grenzen bij Y-waarden van 78 en 82.

Bij het gebruik van een nog kleinere dataset met slechts 50 stalen zijn dezelfde trends te zien. De accuraatheid daalt naar 91,5 %, wat wil zeggen dat nog meer stalen in de verkeerde categorie worden ingedeeld. Ook de grenzen waartussen de stalen niet met 100 % zekerheid in de juiste categorie worden ingedeeld verbreden. Enkel voor de stalen met een Y-waarde minder dan 65 is de kans op een goede basiswitheid 0 %. Er worden zelfs geen stalen meer met 100 % zekerheid met een goede basiswitheid ingedeeld.

In dit voorbeeld lijkt een dataset van 460 stalen dus voldoende. Bij meer complexere classificaties of regressies zal meer data nodig zijn voor dezelfde accuraatheid. De hoeveelheid data nodig voor het bereiken van een bepaald doel moet dus voor elk doel apart worden nagegaan.

In Figuur 40 worden de resultaten van de classificatie met training dataset 7 en 8 vergeleken met de resultaten van de classificatie met training dataset 1. Ook hier geeft de oranje lijn de scheiding van de twee klassen weer en worden de fout ingedeelde stalen in het rood aangeduid.



Figuur 40: Voorspellingen van het model voor de test data met behulp van drie verschillende train datasets. (a) Dataset 1. (b) Dataset 7. (c) Dataset 8.

Dataset 7 bevat 658 stalen, maar bevat geen info over de stalen met een basiswitheid tussen 75 en 82,5. Bij dataset 8 is het net omgekeerd, deze bevat enkel de stalen met een basiswitheid tussen 75 en 82,5 en bestaat slechts uit 287 stalen.

Op het eerste zicht lijken de grafieken uit Figuur 40(a) en Figuur 40(b) zeer gelijkaardig. Toch blijkt uit de accuraatheid dat het neuraal netwerk minder presteert bij het trainen met dataset 7. 16 stalen worden hier verkeerd ingedeeld, terwijl dat bij de eerste dataset slechts 1 staal is. Bovendien wordt voor een staal met een Y-waarde van 79,8 zelfs een kans van 100 % gegeven dat het een goede basiswitheid heeft, terwijl dit staal eigenlijk tot de klasse van slechte basiswitheid behoort. Dit is een logisch resultaat aangezien het neuraal netwerk geen informatie gekregen heeft over de stalen tussen 75 en 82,5. Het is dan ook niet in staat om te leren hoe deze stalen moeten in gedeeld worden. Voor het staal met een Y-waarde van 78,7 wordt een kans voorspelt van 50 %. Deze Y-waarde ligt precies in het midden van 75 en 82,5. Het model kiest hier dus zelf waar de scheiding tussen de twee klassen ligt.

Dataset 8 bestaat maar uit 287 stalen, dit lager aantal stalen zou de minder goede accuraatheid kunnen verklaren. Wat hier wel opvalt is dat het model ook zelf beslist dat stalen met een Y-waarde onder 75 automatisch geen kans hebben op een goede basiswitheid en stalen met een Y-waarde hoger dan 82,5 direct een kans van 100 % hebben om een goede basiswitheid te hebben, ook al heeft het geen informatie over deze stalen gekregen.

4.1.1.2 Invloed van pre-processing

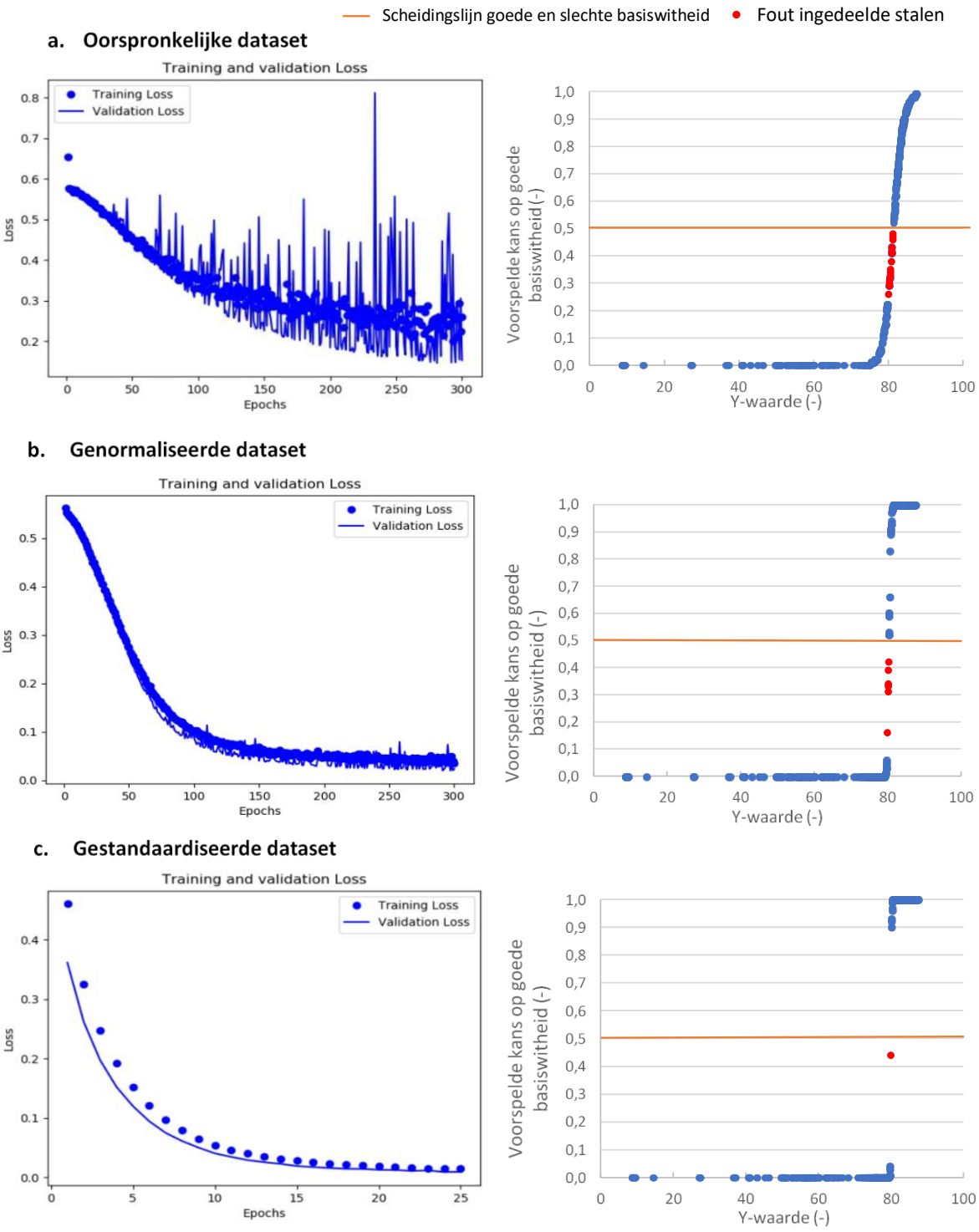
Tijdens het onderzoek werd snel duidelijk dat de prestaties van het netwerk verbeteren wanneer de gegevens van de dataset vooraf worden getransformeerd. Dit wordt pre-processing genoemd. Er bestaan verschillende methoden om de dataset te verbeteren voor optimale prestaties van een neuraal netwerk. De meest geschikte methode hangt sterk af van de gebruikte dataset. In veel gevallen volstaat het om de data te schalen naar kleinere waarden. Bij hoge input waarden werkt het neuraal netwerk namelijk met hoge weights wat het leerproces bemoeilijkt en onstabiel maakt. Het schalen van de data kan via normalisatie, waarbij elke feature wordt getransformeerd naar waarden tussen 0 en 1 of via standaardisatie waar de features worden getransformeerd naar een gemiddelde van 0 en een standaarddeviatie van 1.

In grote datasets treedt vaak collineariteit op, dit is bijvoorbeeld altijd het geval bij spectrale data. Met behulp van een aantal multivariate technieken zoals bijvoorbeeld PCA en PLS kunnen de datasets sterk vereenvoudigd worden met behoud van zo veel mogelijk variantie in de gegevens. De vereenvoudigde dataset kan voor betere prestaties zorgen bij het uitvoeren van deep learning. (61)(43)

Pre-processing van de data zal niet in elk geval betere resultaten geven, ook hier moet voor elk geval apart via trial and error worden nagegaan welke pre-processing methoden al dan niet tot betere resultaten leiden.

In de volgende paragrafen wordt opnieuw de classificatie van stalen met een goede basiswitheid (Y-waarde hoger dan 80) en stalen met een slechte basiswitheid (Y-waarde lager dan 80) uitgevoerd. De invloed van het schalen van de features van zowel de training als de test data wordt nagegaan. In Figuur 41 worden de resultaten van de classificatie met de oorspronkelijke dataset (dataset 1 uit Tabel 7) vergeleken met de resultaten van classificaties met genormaliseerde datasets en gestandaardiseerde dataset.

Het leerproces van het neuraal netwerk wordt geëvalueerd aan de hand van de training en validation loss die wordt uitgezet in functie van het aantal epochs. De resultaten van het neuraal netwerk worden opnieuw geëvalueerd aan de hand van de accuraatheid en de voorspelde kans dat het staal een goede basiswitheid heeft in functie van de werkelijke Y-waarden. Om de stabiliteit van de prestaties van het neuraal netwerk na te gaan wordt elke dataset telkens vier keer gevoed aan eenzelfde model. De gemiddelde accuraatheid en standaarddeviatie worden weergegeven in Tabel 8.



Figuur 41: Verlies curve en voorspellingen van model met (a) Oorspronkelijke dataset. (b) Genormaliseerde dataset. (c) Gestandaardiseerde dataset.

Tabel 8: Gemiddelde accuraatheid en standaarddeviatie op de accuraatheid van drie verschillende pre-processing methoden.

Pre-processing	Gemiddelde accuraatheid (%)	Standaarddeviatie	Bereik stalen met kans > 0 en < 1
Geen	86,3	6,1	75,0 – 87,0
Normalisatie	97,2	2,1	78,4 – 81,6
Standaardisatie	99,5	0	79,6 – 80,4

Bij de verliescurven van het leerproces met de oorspronkelijke dataset, afgebeeld in Figuur 41(a), zijn veel fluctuaties te zien. De fluctuaties zijn veel minder uitgesproken bij de verliescurven van de genormaliseerde data en bij de gestandaardiseerde data zijn er zelfs geen fluctuaties meer aanwezig.

Bij de oorspronkelijke dataset daalt de verliescurve naar een waarde van 0,25, terwijl dit bij de genormaliseerde en gestandaardiseerde dataset verder daalt tot 0,05. Het valt op dat bij de gestandaardiseerde data al na slechts 20 epochs een waarde van 0,05 wordt bereikt, terwijl dit bij de genormaliseerde data pas na 200 epochs is. Het leerproces gebeurt dus sneller wanneer de dataset op voorhand gestandaardiseerd wordt.

Een probleem dat werd ondervonden bij het toepassen van deep learning op de oorspronkelijke dataset was dat de resultaten van eenzelfde model met eenzelfde dataset sterk varieerde. Neurale netwerken zijn stochastisch, dit wil zeggen dat ze willekeurige getallen gebruiken, bijvoorbeeld bij het kiezen van de initiële weights. Wanneer eenzelfde model gebruikt wordt op eenzelfde dataset zullen de initiële weights telkens anders gekozen worden, waardoor het mogelijk is dat andere resultaten bekomen worden. Dit heeft als gevolg dat de resultaten van een bepaald neuraal netwerk niet echt geëvalueerd kunnen worden. Om die rede werd hetzelfde model telkens vier keer met dezelfde dataset uitgetest. De gemiddelde accuraatheid en de standaarddeviatie worden dan als maat gebruikt om aan te geven of een neuraal netwerk al dan niet goed presteert. Een goed presterend neuraal netwerk moet een hoge accuraatheid vertonen en een kleine standaarddeviatie.

Uit de standaarddeviatie blijkt dat het neuraal netwerk stabielere resultaten bekomt wanneer de data genormaliseerd wordt. Bij de gestandaardiseerde data werd vier keer eenzelfde accuraatheid bekomen van 99,5 %. Dit wil zeggen dat er telkens maar één staal in de foute categorie werd ingedeeld.

Aan deze resultaten is duidelijk te zien dat data pre-processing een grote invloed kan hebben op de prestaties van het netwerk. Uit dit voorbeeld blijkt dat het neuraal netwerk beter presteert wanneer de dataset vooraf genormaliseerd wordt dan wanneer de dataset niet getransformeerd wordt. De beste prestaties worden bekomen bij de gestandaardiseerde dataset. Er mag echter niet van uitgegaan worden dat deze invloed altijd positief is. Voor elke classificatie of regressie die in dit onderzoek zijn uitgevoerd werd getest met welke pre-processing methode de beste resultaten werden behaald, ook wanneer het niet specifiek vermeld wordt.

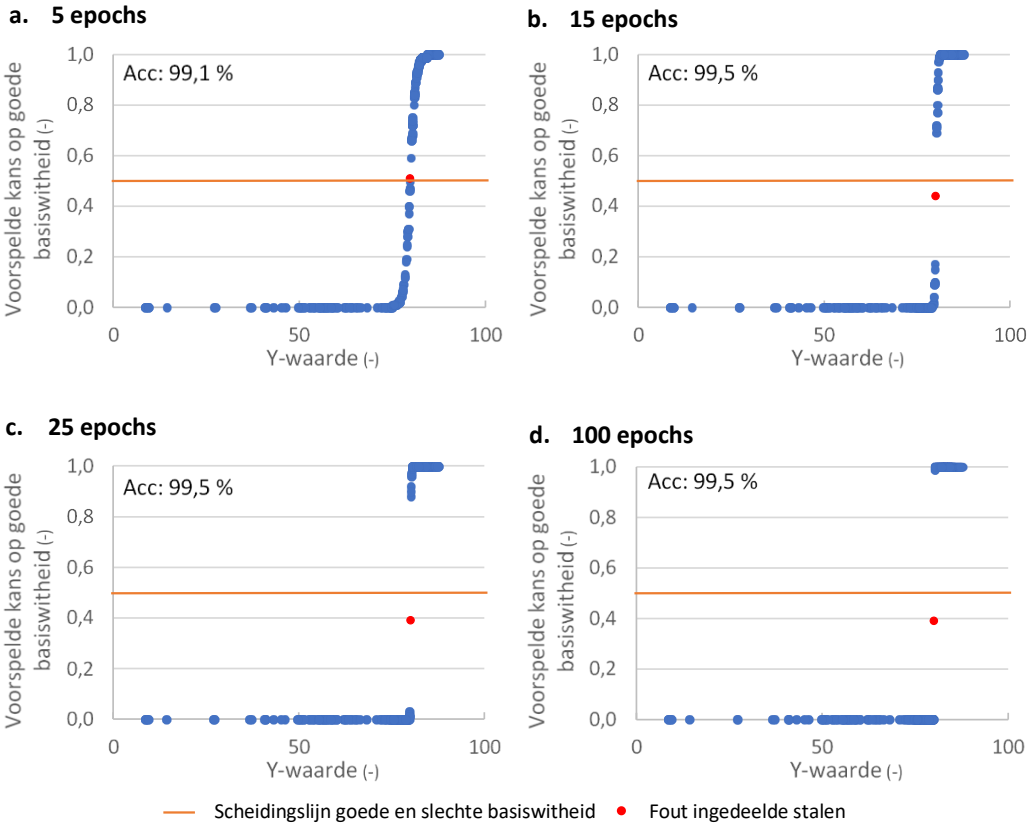
4.1.2 Invloed van parameters van neuraal netwerk

In paragraaf 3.2 werd een stappenplan besproken dat doorlopen moet worden bij het opstellen van een neuraal netwerk. Bij elke stap moeten verschillende keuzes gemaakt worden. Sommige keuzen zoals de activatie functie in de laatste laag, de gebruikte metriek of de verliesfunctie hangen af van het vooropgestelde doel. Andere keuzes zoals bijvoorbeeld de grote van de batch, het aantal epochs, het aantal verborgen eenheden of het aantal verborgen lagen hangen af van de complexiteit van de opdracht en moeten voor elke opdracht apart via trial end error worden nagegaan.

De invloed van de verschillende parameters van het neuraal netwerk worden eenmaal aangetoond. Dit gebeurt opnieuw aan de hand van een classificatie op basis van de Y-waarde in twee klassen, de stalen met een Y-waarde boven 80 heeft een goede basiswitheid, de stalen met een Y-waarde onder 80 hebben een slechte basiswitheid. De gestandaardiseerde dataset gaf de meest stabiele resultaten, waardoor de invloed van verschillende parameters het best wordt aangetoond aan de hand van gestandaardiseerde data. De dataset die hiervoor gebruikt wordt is dataset 1 uit Tabel 7.

In de volgende paragrafen wordt achtereenvolgens de invloed van het aantal epochs, de grootte van de batch, het aantal verborgen eenheden, het aantal verborgen lagen, de activatie functie in de verborgen lagen en de gebruikte optimalisator samen met de learning rate ervan nagegaan.

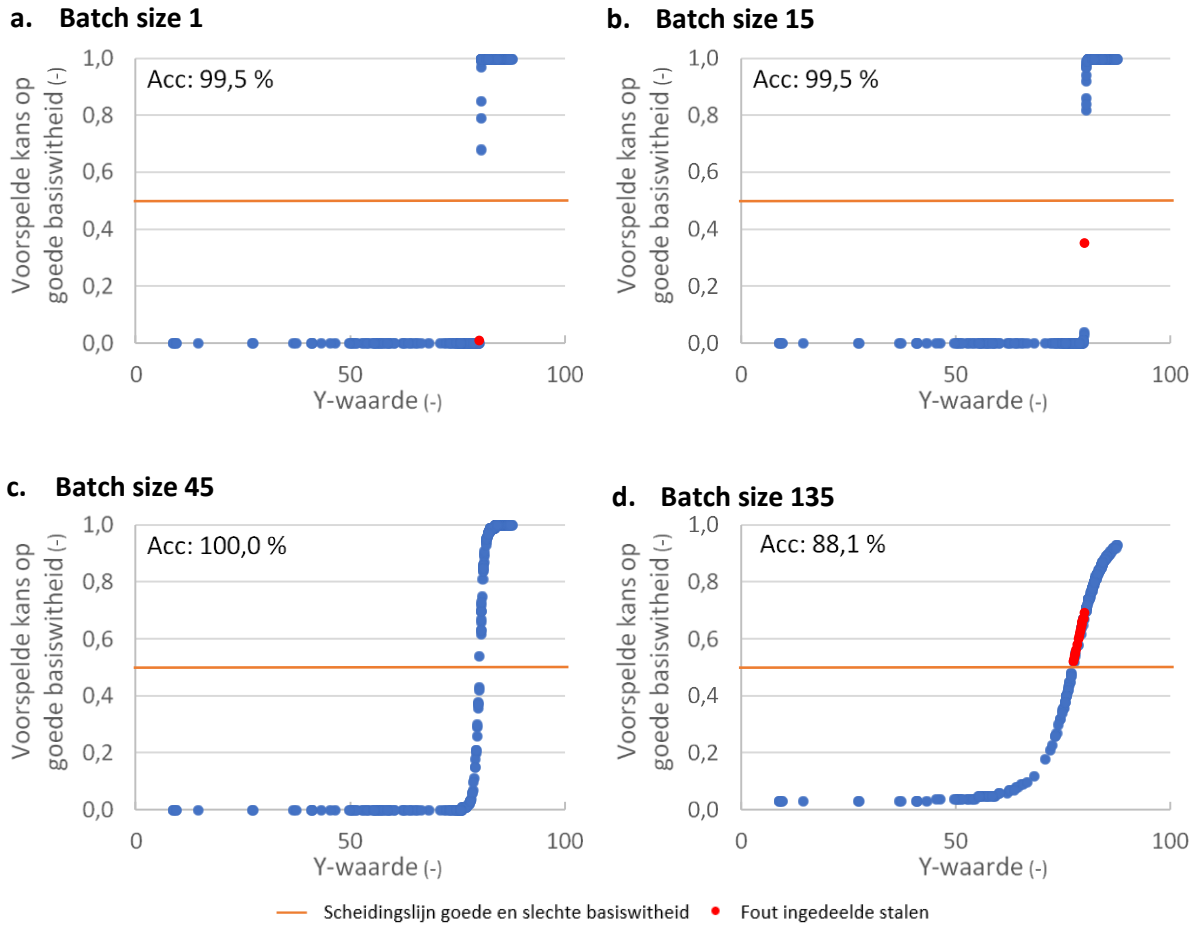
Om de invloed van het aantal epochs na te gaan werd eenzelfde model telkens met een verschillend aantal epochs getraind met eenzelfde dataset. De resultaten zijn te zien in Figuur 42.



Figuur 42: Invloed van aantal epochs op prestaties van neuraal netwerk.
(a) 5 epochs. (b) 15 epochs. (c) 25 epochs. (d) 100 epochs.

Hoewel er zelfs bij 5 epochs al een hoge accuraatheid van 99,1 wordt behaald, blijkt uit Figuur 42 dat de prestaties steeds beter worden naarmate meer epochs worden gebruikt. Steeds meer stalen worden met 100 % zekerheid in de juiste categorie ingedeeld. Na 100 epochs is er zelfs nog maar 1 staal dat niet met zekerheid tot een bepaalde categorie behoort. Het gaat hier niet om een complexe opdracht, waardoor meer epochs steeds betere resultaten opleveren. Bij complexere opdrachten is de kans groter dat er overfitting optreedt bij een groot aantal epochs, waardoor de prestaties op ongeziene data verminderen.

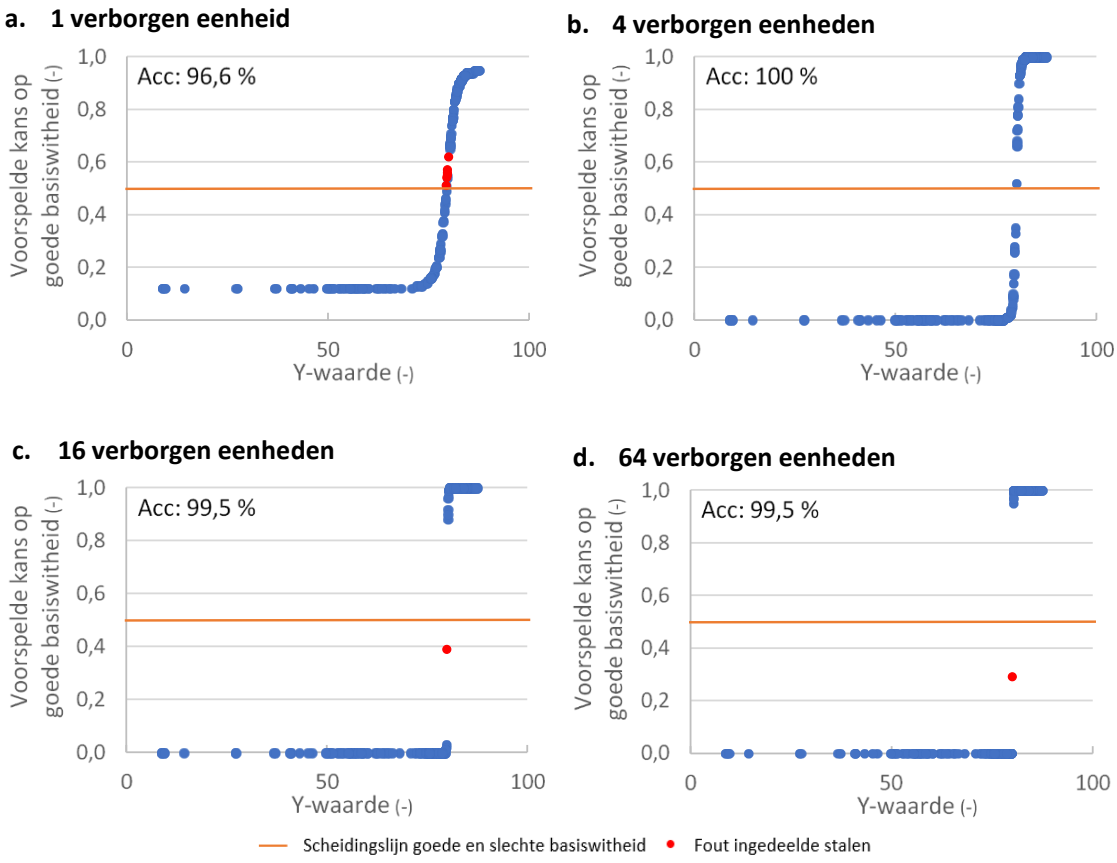
Op dezelfde manier wordt de invloed van de grootte van de batch na gegaan, de resultaten zijn weergegeven in Figuur 43.



Figuur 43: Invloed van batch size op prestaties van neuraal netwerk.
(a) Batch size 1. (b) Batch size 15. (c) Batch size 45. (d) Batch size 135.

Naarmate de batch size groter wordt verslechteren de prestaties van het neuraal netwerk. Hoewel bij een batch size van 45 alle stalen in de juiste categorie worden ingedeeld, zijn er hier toch aanzienlijk meer stalen die niet met 100 % zekerheid in de juiste klasse worden ingedeeld dan bij een kleinere batch size. Bij een batch size van 135 wordt zelf geen enkel staal met 100 % zekerheid in de juiste categorie ingedeeld.

Bij het opbouwen van het neuraal netwerk moeten drie keuzes gemaakt worden, namelijk het aantal verborgen lagen, het aantal verborgen eenheden per laag en de gebruikte activatie functie per laag. In Figuur 44 wordt de invloed van het aantal verborgen eenheden in elke laag van het neuraal netwerk nagegaan.

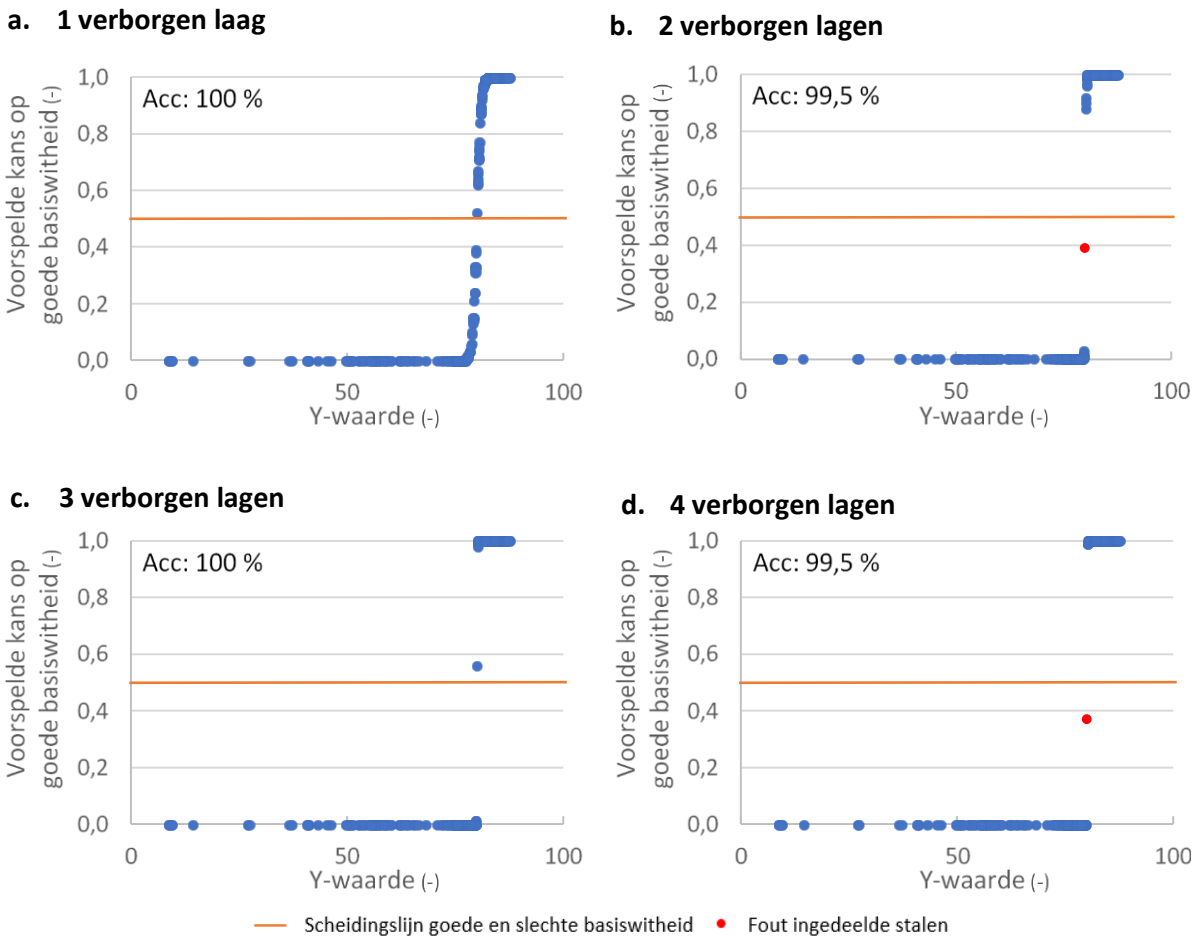


Figuur 44: Invloed van aantal verborgen eenheden op prestaties van neuraal netwerk. (a) 1 verborgen eenheid. (b) 4 verborgen eenheden. (c) 16 verborgen eenheden. (d) 64 verborgen eenheden.

Hoe meer verborgen eenheden hoe beter het netwerk complexe problemen kan oplossen, maar ook hoe groter de kans dat het netwerk ongewenste patronen zal vinden en daarmee dus overfitting zal optreden.

De opdracht van het neuraal netwerk is in dit geval niet zo complex waardoor de kans op overfitting klein is. Om die reden blijft het neuraal netwerk beter presteren naarmate de verschillende lagen meer verborgen eenheden bevatten. De laagste accuraatheid wordt behaald bij het neuraal netwerk met slechts één verborgen eenheid in elke laag. Bovendien wordt hier geen enkel staal met 100 % zekerheid tot de juiste categorie ingedeeld. Het neuraal netwerk met vier verborgen eenheden in elke laag behaalde een accuraatheid van 100%, dit wil zeggen dat alle stalen in de juiste categorie werden ingedeeld. Deze accuraatheid is hoger dan die van de neurale netwerken met 16 en 64 vergborgen eenheden, waarbij telkens toch één staal fout wordt ingedeeld. Toch geeft dit niet direct aan dat het neuraal netwerk met 4 verborgen eenheden beter presteert dan de anderen. Hier worden namelijk 71 stalen van de 235 niet met 100 % zekerheid in de juiste klasse ingedeeld, bij het neuraal netwerk met 16 verborgen eenheden zijn dat er maar 15 en bij 64 verborgen eenheden zelfs maar 6.

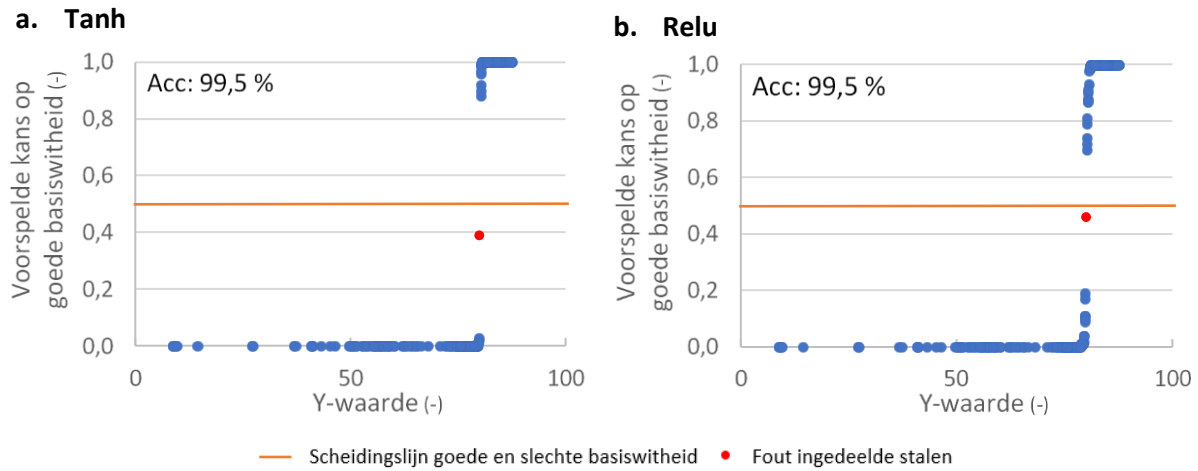
De resultaten van de invloed van het aantal verborgen lagen worden weergegeven in Figuur 45. Hier geldt hetzelfde als bij de verborgen eenheden binnen de verschillende lagen. Hoe meer verborgen lagen hoe beter het netwerk complexe problemen kan oplossen, maar ook hoe groter de kans dat overfitting zal optreden.



Figuur 45: Invloed van aantal verborgen lagen op prestaties van neuraal netwerk.
(a) 1 verborgen laag. (b) 4 verborgen lagen. (c) 16 verborgen lagen. (d) 64 verborgen lagen.

Ook hier daalt het aantal stalen dat niet met 100 % zekerheid tot de juiste categorie wordt ingedeeld naarmate het aantal lagen stijgt.

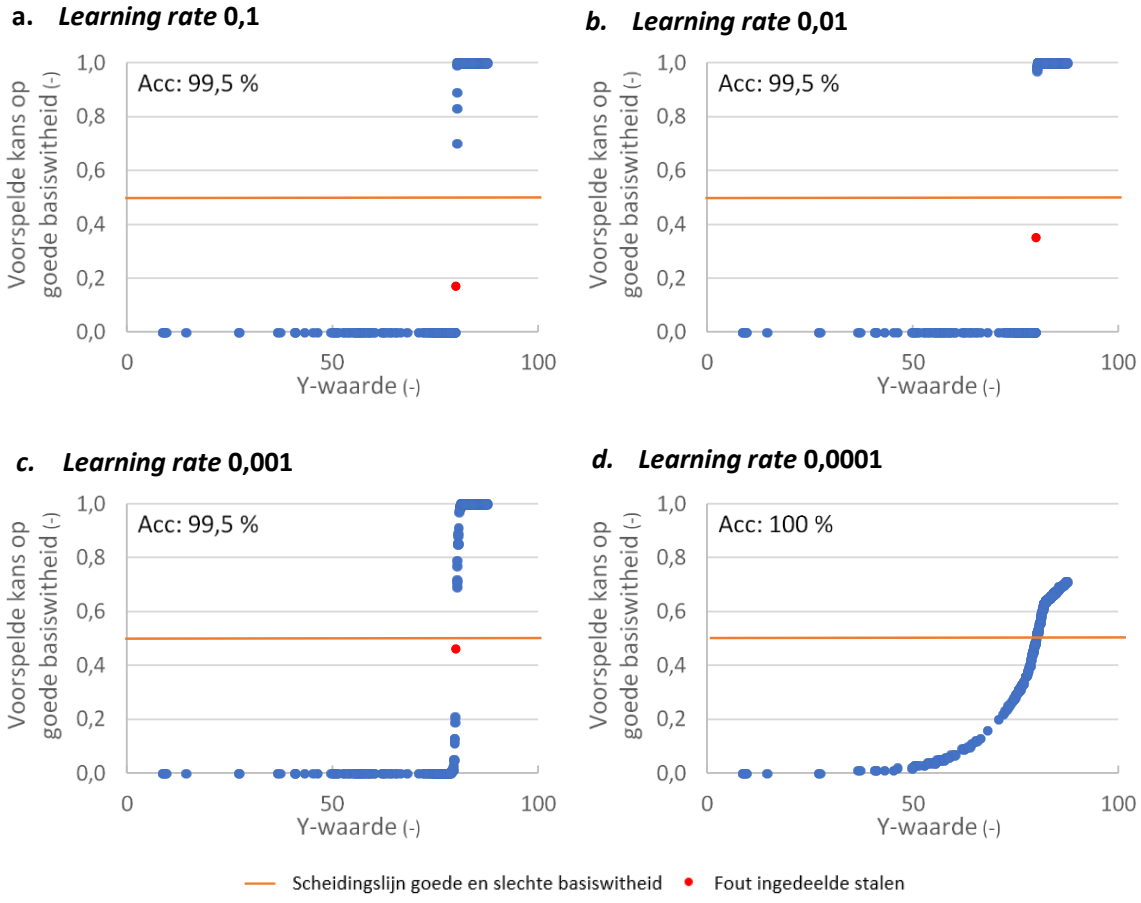
In Figuur 46 worden de resultaten van de neurale netwerken met activatie functie “relu” en “tanh” met elkaar vergeleken.



Figuur 46: Invloed van activatie functie op prestaties van neuraal netwerk. (a) Tanh. (b) Relu.

Het verschil in prestaties tussen neurale netwerken met als activatie functie “relu” of “tanh” was niet altijd duidelijk zichtbaar. In het voorbeeld in Figuur 46 blijkt “tanh” voor betere prestaties te zorgen. Hoewel in beide gevallen een accuraatheid van 99,5 % wordt gehaald, gebeurt de scheiding van de stalen in de twee klassen beter met “tanh”. Hier worden slechts 15 stalen niet met 100 % zekerheid in de juiste categorie ingedeeld terwijl dit er bij “relu” 35 zijn.

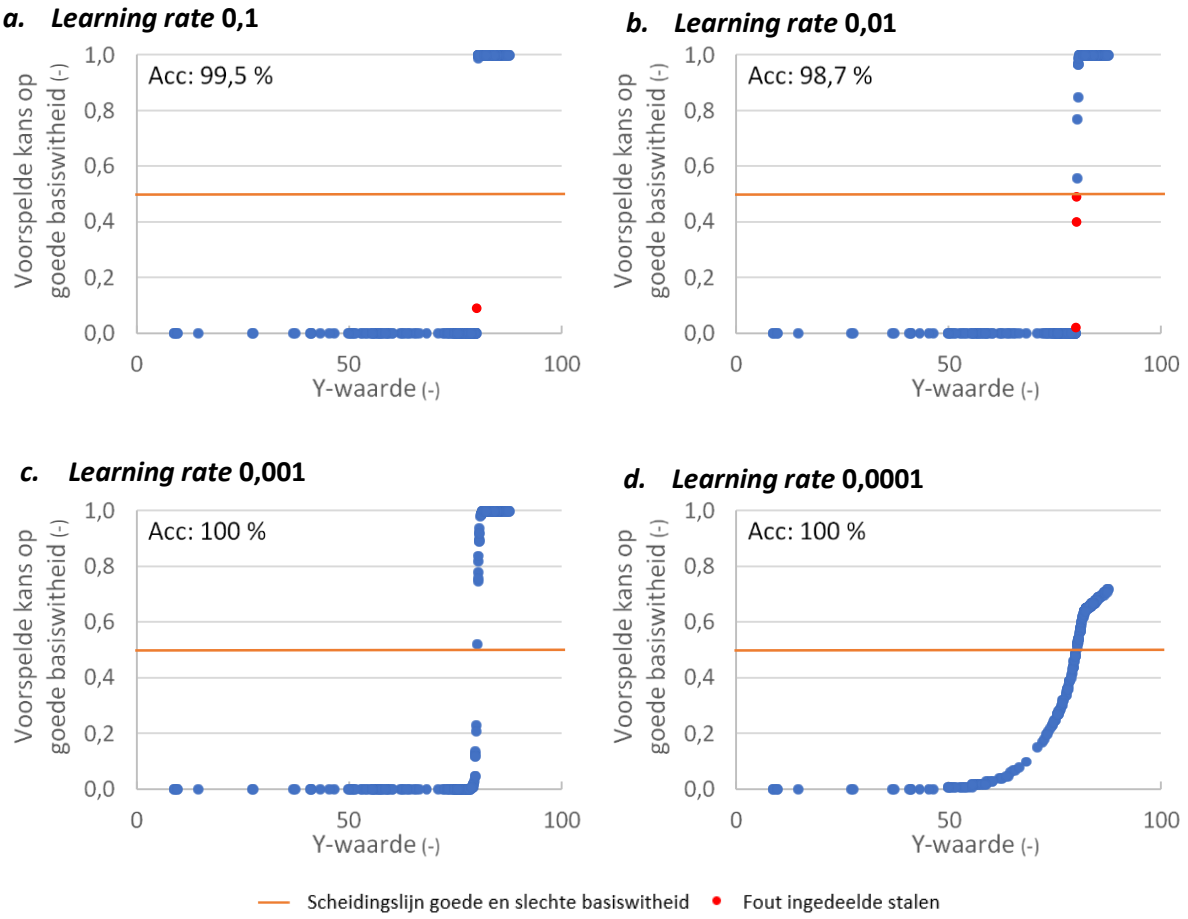
Als laatste wordt de invloed van de gebruikte optimalisator nagegaan samen met de ingestelde learning rate ervan. De learning rate geeft de grootte van de aanpassing van alle weights weer per iteratie. Een te kleine learning rate zorgt voor een traag leerproces en de mogelijkheid dat het proces vast komt te zitten in een lokaal minimum. Bij een te grote learning rate bestaat de kans dat er voorbij het minimum wordt gegaan of dat de updates zelf willekeurige waarden zullen aannemen. In Figuur 47 zijn de resultaten van het neuraal netwerk dat geoptimaliseerd wordt met de “RMSProp” optimalisator weergegeven. Hierbij worden vier verschillende learning rates ingesteld.



Figuur 47: Invloed van learning rate van RMSProp op prestaties van neuraal netwerk.
(a) Learning rate 0,1. (b) Learning rate 0,01. (c) Learning rate 0,001. (c) Learning rate 0,0001.

Hoewel de hoogste accuraatheid behaald wordt bij een learning rate van 0,0001, blijkt uit Figuur 47 dat dit toch niet de beste resultaten voor de classificatie oplevert. De kans dat een staal een goede basiswitheid heeft begint al te stijgen vanaf een Y-waarde van 40. De learning rate is hier te klein waardoor de optimalisatie te traag verloopt.

Hoewel in het boek *Deep learning met Python* van F. Chollet (44) wordt aangegeven dat de “RMSProp” optimalisator voldoende goede resultaten zal opleveren voor eender welk doel, wordt hier toch nog een andere optimalisator getest namelijk de “Adam” optimalisator. De resultaten ervan worden weergegeven in Figuur 48



Figuur 48: Invloed van *learning rate* van Adam op prestaties van neuraal netwerk.
(a) *Learning rate* 0,1. (b) *Learning rate* 0,01. (c) *Learning rate* 0,001. (c) *Learning rate* 0,0001.

Ook hier blijkt een learning rate van 0,0001 niet voldoende voor een goede optimalisatie, ondanks de accuraatheid van 100 %. Ook bij een learning rate van 0,001 worden alle stalen wel in de juiste categorie ingedeeld maar nog steeds bevinden veel stalen zich in het gebied tussen de twee categorieën. Bij grotere learning rates wordt een lagere accuraatheid bekomen, maar steeds meer stalen worden met 100 % zekerheid in de juiste categorie ingedeeld.

Wanneer Figuur 47 vergeleken wordt met Figuur 48 blijken de resultaten zeer gelijkaardig. Daarom wordt bij alle classificaties en regressies enkel gebruik gemaakt van de optimalisator “RMSProp” zoals in het boek van F. Chollet beschreven wordt.

De invloed van al deze parameters hangen sterk af van de complexiteit van de opdracht en moet voor elke opdracht apart worden nagegaan. Voor elke uitgevoerde classificatie en regressie in dit onderzoek werden de geschikte parameterinstellingen nagegaan, ook als dit niet specifiek vermeld wordt.

4.2 Classificaties

Bij elke uitgevoerde classificatie wordt het stappenplan dat hieronder nog eens wordt weergegeven doorlopen.

Vooraf

- Welke gegevens uit de dataset zijn van belang?
- Is de dataset geschikt?
- Hoe de dataset optimaliseren?
- Hoe splitsen in training en test data?

Stap 1: Het collecteren en klaarmaken van de train en test data

- Gaat het om discrete of continue data?
- In welke vorm moet de data gevoed worden?

Stap 2: Het opbouwen van een netwerk van lagen

- Hoeveel verborgen lagen?
- Hoeveel verborgen eenheden?
- Welke activatie functie?

Stap 3: Het configureren van het leerproces

- Welke loss functie?
- Welke optimizer?
- Welke metric?

Stap 4: Het valideren van het leerproces

- Welke validatie methode wordt er gebruikt?
- Hoeveel epochs?
- Welke Batch size?

In de eerste twee stappen wordt de dataset voorbereid om gevoed te worden aan het neuraal netwerk. Deze stappen moeten voor elke dataset en voor elke classificatie opnieuw doorlopen worden. Bij het opbouwen en trainen van het neuraal netwerk moet een antwoord worden gegeven op de negen vragen van stap 2, stap 3 en stap 4. De in het groen aangeduide vragen liggen vast voor elke classificatie. De in het rood aangeduide vragen daarentegen kunnen enkel via trial end error worden nagegaan en moeten dus voor elke classificatie apart empirisch onderzocht worden.

4.2.1 Classificatie op basis van Y-waarden

Bij een eerste classificatie worden de stalen verdeeld in twee klassen. Stalen met een Y-waarde boven 80 hebben een goede basiswitheid en stalen met een Y-waarde onder 80 hebben een slechte basiswitheid. Als features worden de Y-waarden zelf gebruikt. Van alle gegevens uit de dataset van de drie analysetoestellen is voor deze classificatie dus enkel de Y-waarde nodig. Aan elk staal wordt daarnaast nog een label toegekend, stalen met een Y-waarde lager dan 80 krijgen een label 0, stalen met een Y-waarde hoger dan 80 krijgen een label 1. Tabel 9 toont een deel van de train data voor deze classificatie. Dit toont aan hoe de

train en test data er precies uitzien. De labels worden in het groen aangeduid en de features in het rood.

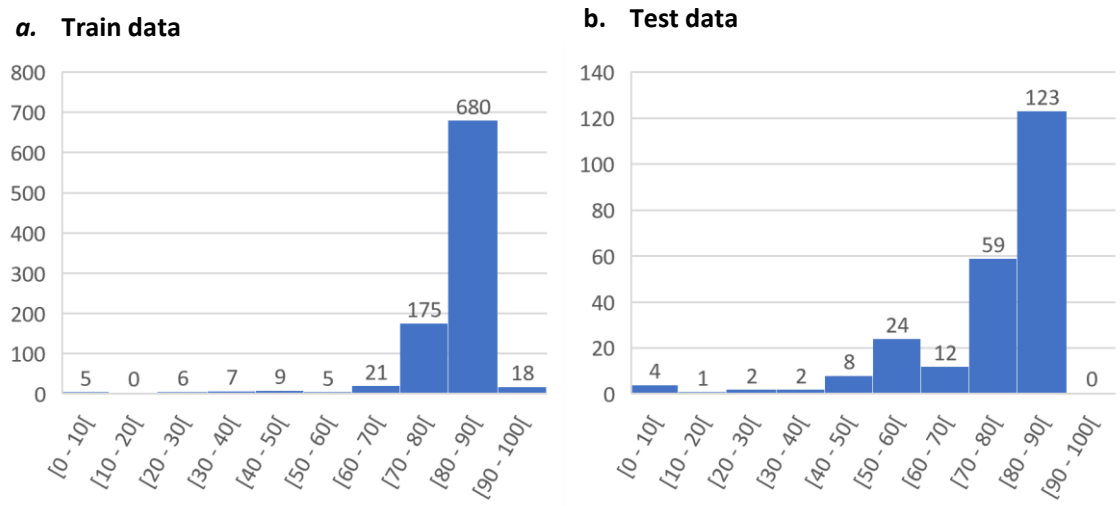
Tabel 9: Voorbeeld van training data voor classificatie van stalen in goede en slechte basiswitheid.

Label	Y-waarde
1	84,12
0	78,52
0	78,56
0	78,69
0	77,73
1	83,61

In Tabel 10 wordt meer informatie gegeven over de train en test dataset die voor deze classificatie gebruikt wordt. De verdeling van de train en test data wordt in de histogrammen in Figuur 49 afgebeeld.

Tabel 10: Gebruikte train en test data voor classificatie van stalen in goede en slechte basiswitheid.

Dataset	Totaal aantal stalen	Bereik Y-waarde	Gemiddelde Y-waarde	Standaarddeviatie Y-waarde
Train	945	3,4 - 91,7	80,7	10,4
Test	235	8,7 – 87,5	74,0	15,5



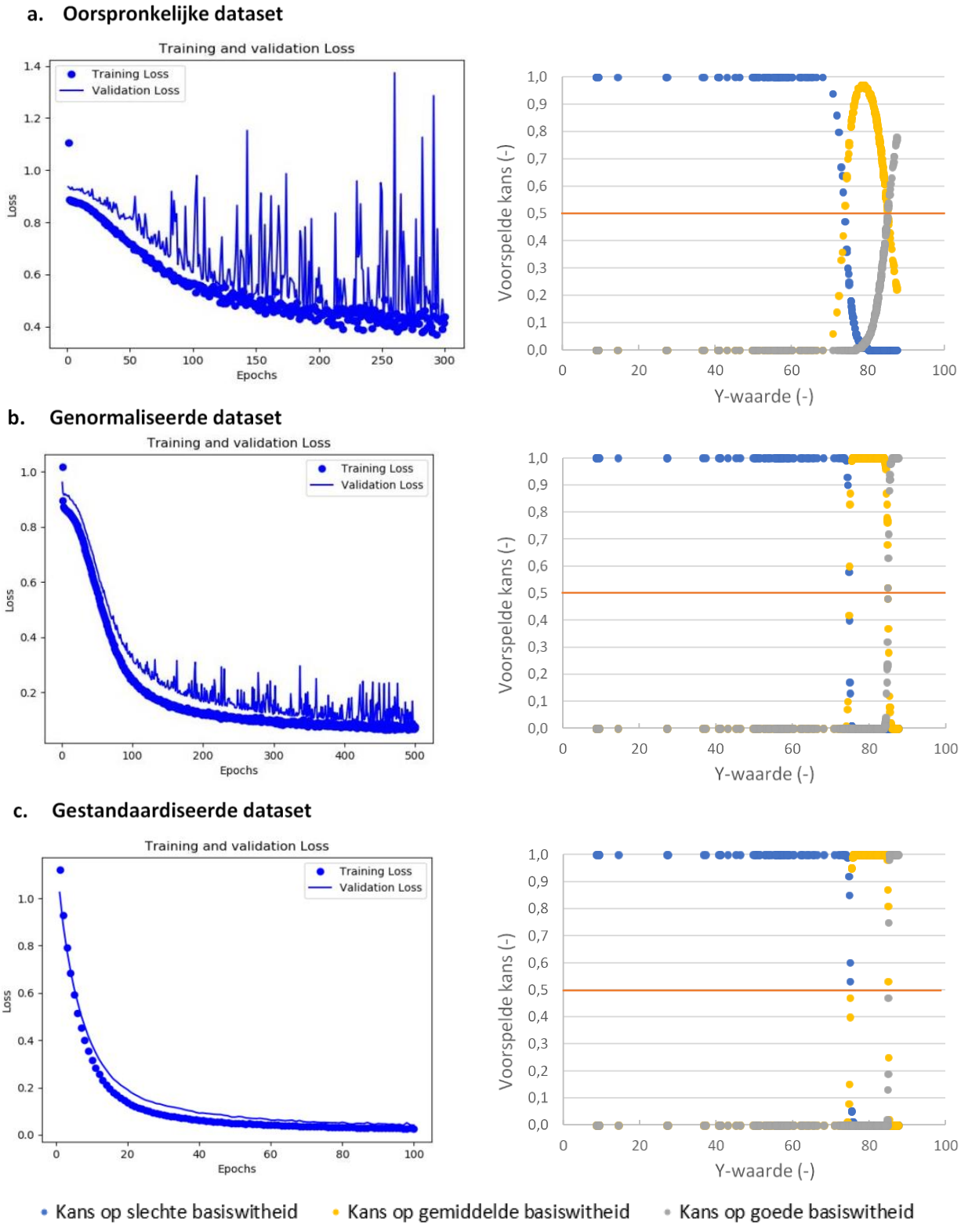
Figuur 49: Histogram van (a) train data en (b) test data.

De resultaten van deze classificatie werden in paragraaf 4.1.1 en 4.2.1 al uitgebreid besproken, waarbij de invloed van de dataset en de invloed van de parameters van het neuraal netwerk worden nagegaan. De prestaties van het neuraal netwerk verbeteren aanzienlijk wanneer de data vooraf gestandaardiseerd wordt. Daarnaast blijkt deze classificatie niet gevoelig voor overfitting, waardoor de prestaties steeds verbeteren bij een hoger aantal epochs, verborgen eenheden en verborgen lagen.

In een volgende stap wordt dezelfde dataset geclassificeerd in meerdere klassen. De resultaten van de classificatie in drie klassen worden weergegeven in Figuur 50. De indeling in drie klassen gebeurt als volgt:

- Y-waarde < 75 = Slechte basiswitheid (label 1)
- 75 < Y-waarde > 85 = Gemiddelde basiswitheid (label 2)
- Y – waarde > 85 = Goede basiswitheid (label 3)

De Y-waarden in de train en test data krijgen respectievelijk een label 0, 1 en 2 toegekend. Slechts 8 % van test data heeft een Y-waarde hoger dan 85 en behoort tot de hoogste categorie. Om die rede moet er extra kritisch naar de accuraatheid gekeken worden. Een accuraatheid van bijvoorbeeld 92 % is op zich geen slecht resultaat, maar aangezien slechts 8 % van de stalen in de testdata een goede basiswitheid hebben, zou een accuraatheid van 92 % eveneens kunnen betekenen dat geen enkel staal met een Y-waarde hoger dan 85 juist wordt ingedeeld, wat dus een slecht model zou zijn.



Figuur 50: Verlies curve en voorspellingen van model met (a) Oorspronkelijke dataset. (b) Genormaliseerde dataset. (c) Gestandaardiseerde dataset.

In deze figuur wordt nogmaals de invloed van de pre-processing methoden normaliseren en standaardiseren aangetoond. De gemiddelde accuraatheid en standaarddeviatie worden weergegeven in Tabel 11.

Tabel 11: Gemiddelde accuraatheid en standaarddeviatie op de accuraatheid van drie verschillende pre-processing methoden.

Pre-processing	Gemiddelde accuraatheid (%)	Standaard-deviatie	Bereik stalen met kans > 0 en < 1
Geen	95,9	1,0	< 68
Normalisatie	98,1	0,8	74 – 76 en 84 – 86
Standaardisatie	99,6	0,4	74,6 – 75,4 en 84,8 – 85,6

Wanneer de dataset niet genormaliseerd of gestandaardiseerd wordt, zijn er weer zeer veel fluctuaties te zien in de verliescurves, wat een effect zal hebben op de stabiliteit van het model. In Figuur 50(a) is te zien dat de verliescurve in dit geval geleidelijk aan daalt tot een waarde van ongeveer 0,4. In Figuur 50(b) is de verliescurve van het neuraal netwerk dat getraind wordt met de genormaliseerde data afgebeeld. De curve daalt sneller en bereikt een waarde van 0,05. Dit is lager dan bij de niet-genormaliseerde dataset, het model wordt dus beter getraind. Er zijn nog steeds fluctuaties waarneembaar in de validatie verlies, maar dit is aanzienlijk minder dan bij de niet-genormaliseerde data. De verliescurve bij de gestandaardiseerde dataset ziet er zeer stabiel uit. Het daalt in een vloeiende lijn naar 0 in minder dan 100 epochs. Er zijn ook geen fluctuaties meer te zien bij de validatie.

De drie datasets werden telkens driemaal getest om de stabiliteit van het model na te gaan. Hier is dezelfde trend waarneembaar als bij de classificatie in twee klassen. De accuraatheid wordt hoger bij genormaliseerde data en nog hoger bij gestandaardiseerde data. Tegelijk daalt de standaarddeviatie, wat aangeeft dat de resultaten van eenzelfde neuraal netwerk met eenzelfde model stabieler zijn.

De geschikte parameter instellingen van het neuraal netwerk worden ook voor deze classificatie uitvoerig nagegaan. Naast het aantal epochs werden de beste resultaten voor zowel de niet getransformeerde, de genormaliseerde en de gestandaardiseerde dataset bij dezelfde parameter instellingen bekomen. Deze parameter instellingen worden weergegeven in Tabel 12. Het zijn ook deze parameters die gebruikt werden bij de neurale netwerken van Figuur 50.

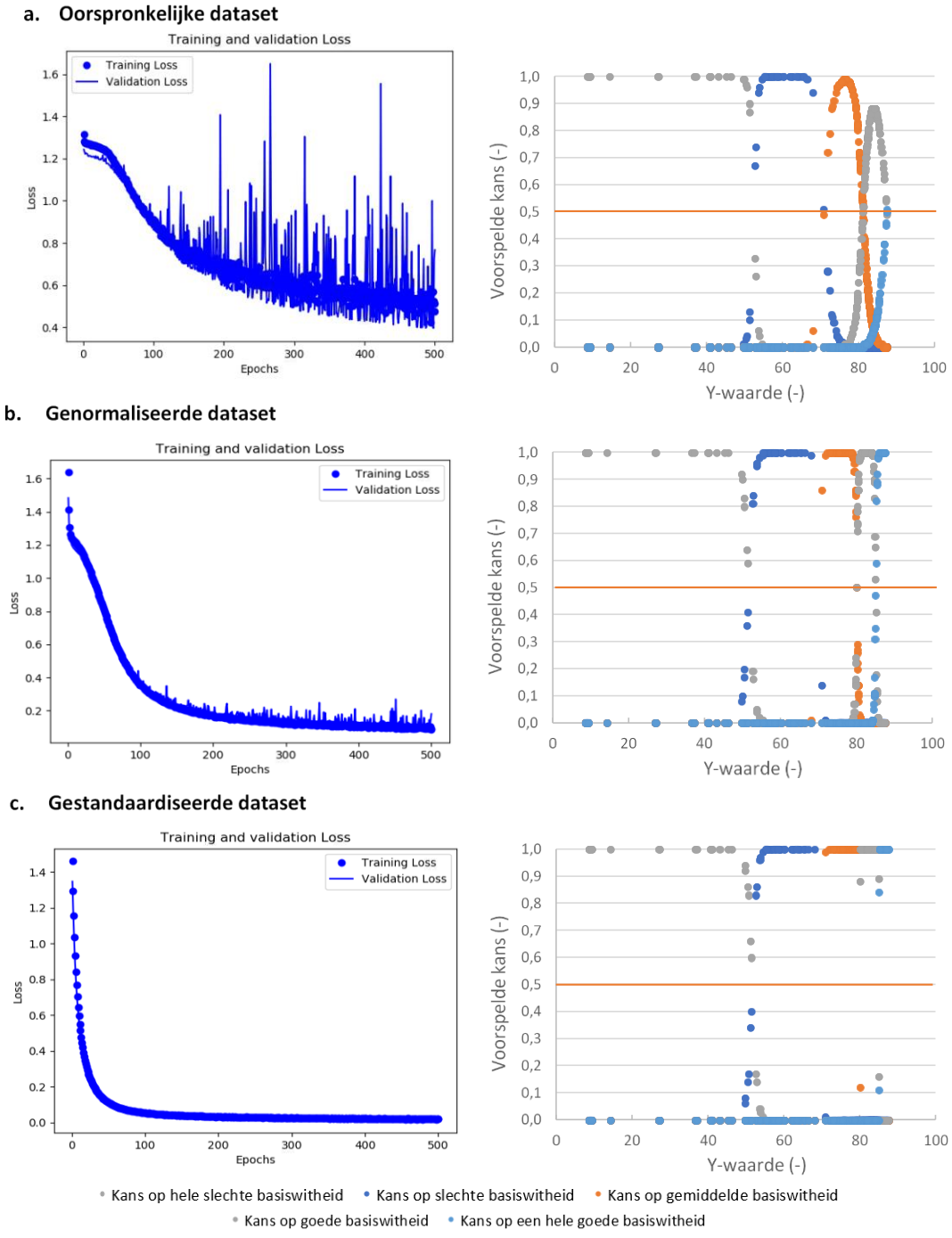
Tabel 12: Parameter instellingen van neuraal netwerk voor classificatie naargelang Y-waarden.

Parameter	Instelling
Intermediate Layers	2
Hidden Units	16
Activation	Tanh
loss function	categorical_crossentropy
optimizer	RMSProp
metric	accuracy
Validation method	simple hold-out validation
Batch size	20

Vanuit de classificatie in drie verschillende categorieën is de stap naar een classificatie naar meerdere categorieën niet meer zo groot. De volgende stap in dit onderzoek is dus een classificatie van dezelfde dataset in vijf klassen. De indeling in vijf klassen gebeurt als volgt:

- Y – waarde < 50 = Zeer slechte basiswitheid (label 1)
- 50 < Y-waarde > 70 = Slechte basiswitheid (label 2)
- 70 < Y-waarde < 80 = Gemiddelde basiswitheid (label 3)
- 80 < Y-waarde < 85 = Goede basiswitheid (label 4)
- Y-waarde > 85 = Zeer goede basiswitheid (label 5)

De parameter instellingen die voor de beste resultaten zorgde waren dezelfde als bij de classificatie in drie klassen, weergegeven in Tabel 12. De resultaten worden opnieuw weergegeven voor zowel de niet getransformeerde, de genormaliseerde en de gestandaardiseerde dataset in Figuur 51.



Figuur 51: Verlies curve en voorspellingen van model met (a) Oorspronkelijke dataset. (b) Genormaliseerde dataset. (c) Gestandaardiseerde dataset.

De gemiddelde accuraatheid en standaarddeviatie worden weergegeven in Tabel 13.

Tabel 13: Gemiddelde accuraatheid en standaarddeviatie op de accuraatheid van drie verschillende pre-processing methoden.

Pre-processing	Gemiddelde accuraatheid (%)	Standaarddeviatie
Geen	76,6	15,0
Normalisatie	93,9	3,8
Standaardisatie	98,4	0,2

Het neuraal netwerk behaalt de hoogste accuraatheid en de laagste standaarddeviatie bij de gestandaardiseerde dataset. Ook wordt de scheiding tussen de verschillende klassen beter bij de gestandaardiseerde data, bij de gestandaardiseerde dataset worden slechts 15 van de 235 stalen niet met 100 % zekerheid in de juiste klasse ingedeeld, terwijl dit bij de genormaliseerde dataset 74 stalen zijn en bij de niet-getransformeerde dataset zelf 195.

4.2.2 Uitbreiding van features naar X-, Y- en Z-waarden

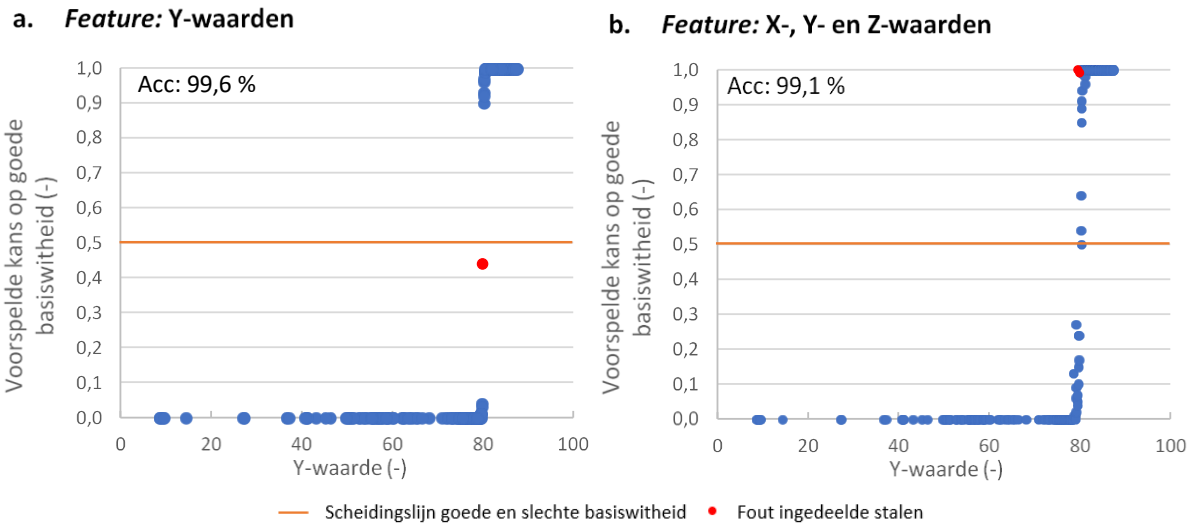
4.2.2.1 Classificatie op basis van Y-waarden

In een volgende stap worden de features uitgebreid naar X-, Y- en Z-waarden. Als eerste wordt dezelfde classificatie tussen stalen met een slechte basiswitheid (Y-waarden boven 80) en stalen met een goede basiswitheid (Y-waarden onder 80) uitgevoerd. Aangezien het verschil tussen een staal met een goede basiswitheid en een staal met een slechte basiswitheid enkel bepaald wordt door de Y-waarden, zou het neuraal netwerk dus zelf moeten leren om de X- en Z-waarde als irrelevant te beschouwen. In Tabel 14 wordt een deel van de trainingsdata weergegeven, de features zijn in het rood afgebeeld, de labels in het groen.

Tabel 14: Voorbeeld van training data voor classificatie van stalen in goede en slechte basiswitheid.

Label	X-waarde	Y-waarde	Z-waarde
1	81,60	85,02	97,93
0	78,94	80,94	110,43
0	79,03	80,97	111,73
0	79,31	81,24	112,60
0	78,21	80,17	110,55
1	82,94	85,62	109,84

Ook hier weer wordt de dataset op 3 verschillende manieren getest. Het trainen van het neuraal netwerk met de gestandaardiseerde data leverde de beste resultaten. Enkel deze resultaten zullen besproken worden. De optimale parameter instellingen van het neuraal netwerk waren ook hier weer dezelfde als weergegeven in Tabel 12. In Figuur 52 worden de resultaten van de classificatie met enkel de Y-waarde uit de vorige paragraaf vergeleken met de classificatie met de X-, Y- en Z-waarden. Voor beide classificaties werd dezelfde dataset hiervoor gebruikt, de gegevens worden weergegeven in Tabel 10.



Figuur 52: Vergelijking resultaten van classificatie met als features Y-waarden en X-, Y- en Z-waarden.

Bij de classificatie met enkel de Y-waarden wordt telkens maar één staal in de verkeerde klasse ingedeeld, bij de classificatie met de X-, Y- en Z-waarden zijn dat er altijd twee. Bovendien worden deze twee stalen door het neuraal netwerk aanzien als stalen waarbij de kans 100 % is dat ze een goede basiswitheid hebben, terwijl ze eigenlijk tot de categorie van slechte basiswitheid toebehoren. Daarnaast zijn er bij de classificatie met slechts één feature maar 12 stalen die niet met 100 % zekerheid in de juiste categorie worden ingedeeld terwijl dit er bij de classificatie met drie features 34 zijn. De reden van deze minder goede resultaten bij de classificatie met behulp van drie features, kan zijn dat de classificatie nog steeds gebeurt naargelang de Y-waarde (UV0), maar dat als features de X-, Y- en Z-waarden (UVadj) wordt gevoed aan het neuraal netwerk.

4.2.2.2 Classificatie naargelang WI_{Ganz}

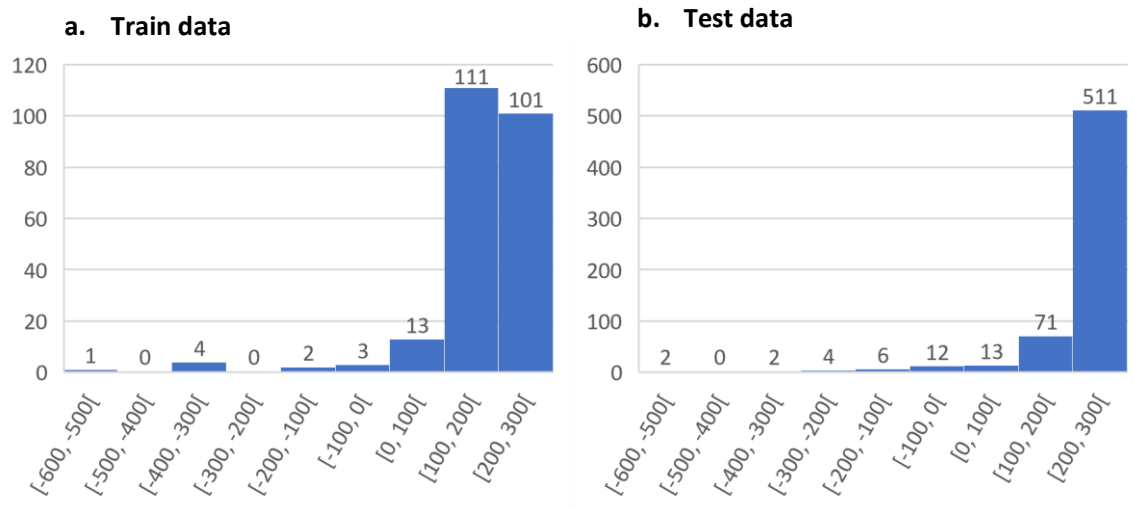
WI_{Ganz} is een witheidindex die wordt berekend aan de hand van de X-, Y- en Z-waarden. In een volgende classificatie wordt daarom nagegaan of het mogelijk is om stalen in klassen met een hoge, een gemiddelde en een lage WI_{Ganz} in te delen, zonder dat deze waarde zelf aan het neuraal netwerk wordt gevoed. De classificatie zou mogelijk moeten zijn door enkel informatie over de X-, Y- en Z-waarden voor elk staal mee te geven. De classificatie die hier gemaakt wordt is de volgende:

- $WI_{Ganz} < 200$ = Slechte witheid
- $200 < WI_{Ganz} < 220$ = Gemiddelde witheid
- $WI_{Ganz} > 220$ = Goede witheid

Informatie over de train en test data worden weergegeven in Tabel 15 en Figuur 53.

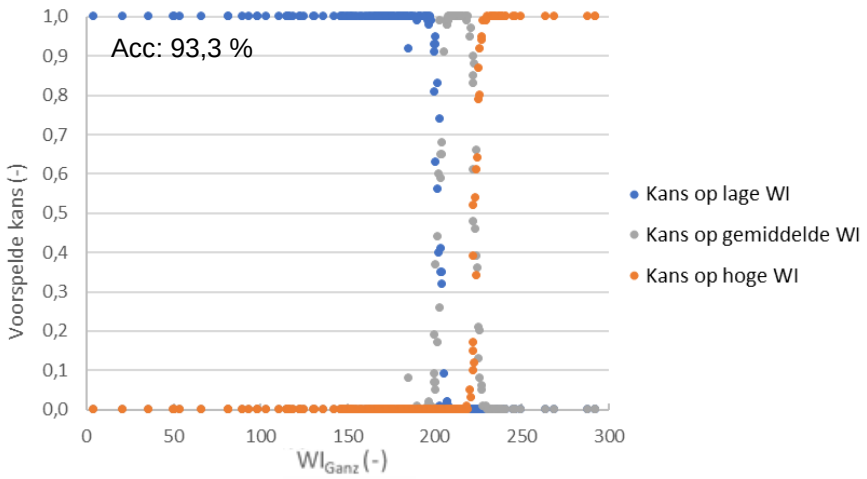
Tabel 15: Gebruikte train en test data voor classificatie van stalen met hoge, gemiddelde en lage WI_{Ganz}.

Dataset	Totaal aantal stalen	Bereik WI_{Ganz}	Gemiddelde WI_{Ganz}	Standaarddeviatie WI_{Ganz}
Train	945	-609 - 390	176,5	79,9
Test	235	-547 - 292	168,0	101,2



Figuur 53: Histogram van (a) train data en (b) test data.

Ook hier wordt de dataset in de drie verschillende vormen getest en worden de beste parameterinstellingen van het neuraal netwerk nagegaan. De beste resultaten werden bekomen na standaardisatie van de dataset, de beste parameterinstellingen waren dezelfde als in Tabel 12. De resultaten zijn weergegeven in Figuur 54.



Figuur 54: Resultaten van classificatie op basis van WI_{Ganz}.

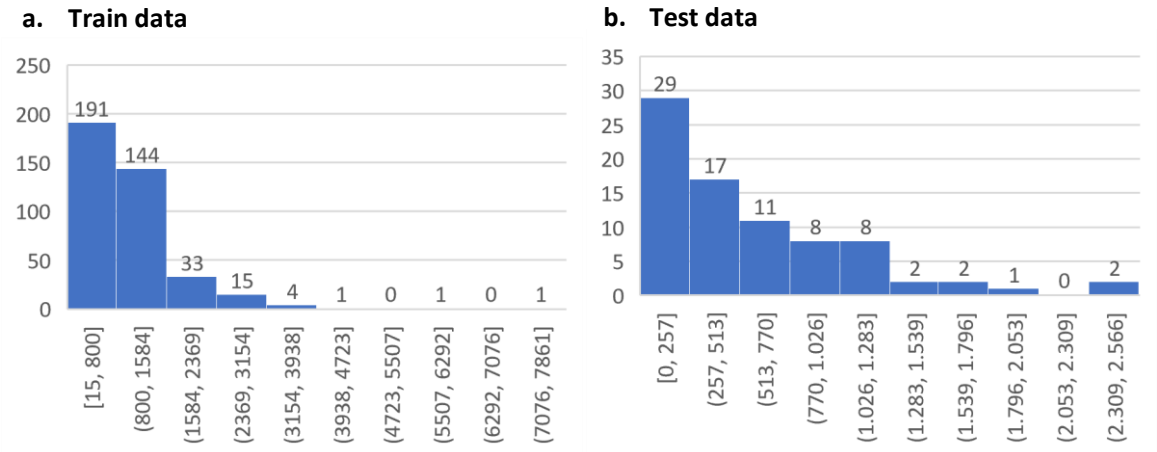
Er werd een accuraatheid van 93,3 gehaald. Dit wil zeggen dat 18 van de 235 stalen van de test set fout worden ingeschat. Deze resultaten zijn al iets minder goed dan bij de classificatie op basis van de Y-waarden, maar in Figuur 54 blijkt dat er wel degelijk een classificatie van stalen met een lage, gemiddelde en hoge WI_{Ganz} mogelijk is zonder deze zelf aan het netwerk te voeden. Enkel rond de grenzen van de verschillende klassen worden enkele stalen fout ingedeeld.

4.2.2.3 Classificatie naargelang hoeveelheid optische witmaker

De hoeveelheid optische witmaker (FWA) aanwezig op het textiel wordt bepaald met behulp van HPLC. De hoeveelheid DSBP en DAS op het textiel worden apart waargenomen met de HPLC, de som hiervan geeft de totale hoeveelheid optische witmaker op het textiel weer. Bij deze classificatie wordt nagegaan of de stalen geclassificeerd kunnen worden in stalen met weinig, gemiddeld en veel FWA. De informatie over de train en test dataset die hiervoor beschikbaar zijn wordt weergegeven in Tabel 16 en Figuur 55.

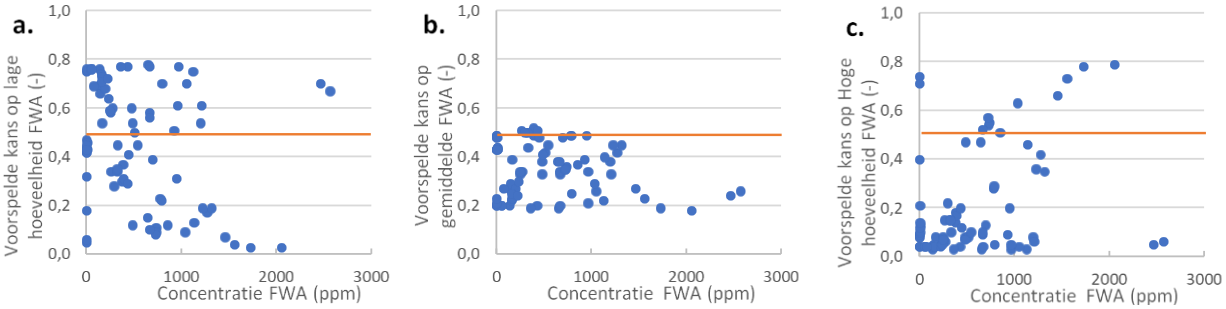
Tabel 16: Gebruikte train en test data voor classificatie van stalen met hoge, gemiddelde en lage Hoeveelheid FWA.

Dataset	Totaal aantal stalen	Bereik Conc. FWA (ppm)	Gemiddelde Conc. FWA (ppm)	Standaarddeviatie
Train	390	15 - 7861	980,9	807,9
Test	80	0 - 2566	571,5	561,7



Figuur 55: Histogram van (a) train data en (b) test data.

De train dataset bevat slechts 390 stalen. Dit is minder dan bij de vorige classificaties. De resultaten van de classificatie in drie verschillende klassen worden weergegeven in Figuur 56. Om de grafieken overzichtelijk te houden wordt de kans dat een staal tot een bepaalde categorie behoort telkens in een aparte grafiek afgebeeld. De stalen worden ingedeeld in stalen met minder dan 500 ppm FWA, stalen met een concentratie FWA tussen 500 ppm en 1000 ppm en stalen met meer dan 1000 ppm FWA.



Figuur 56: Resultaten van classificatie naargelang hoeveelheid FWA met X-, Y- en Z-waarden als features. (a) Kans op lage hoeveelheid FWA. (b) Kans op gemiddelde hoeveelheid FWA. (c) Kans op hoge hoeveelheid FWA.

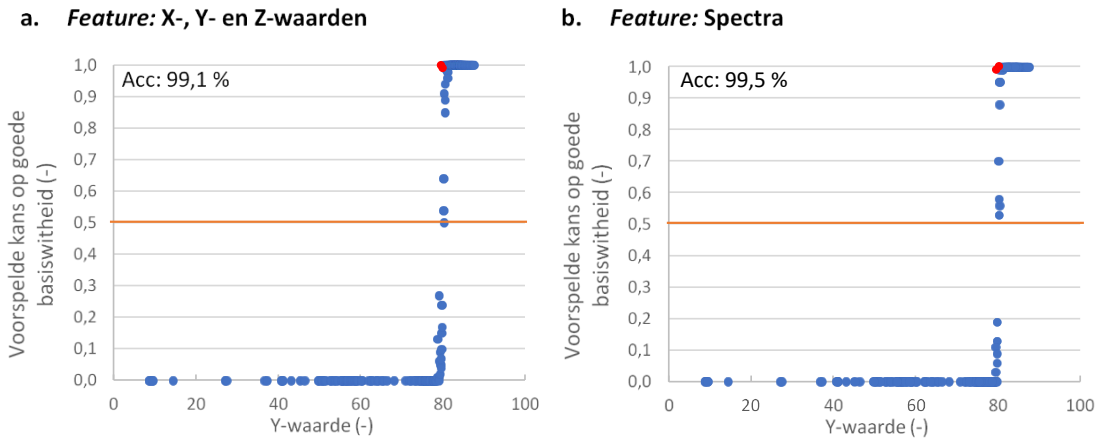
Bij deze classificatie wordt een accuraatheid van 49,9 % behaald. Dit is zeer laag. Uit de drie grafieken van Figuur 56 blijkt dat er niet echt een classificatie gebeurt. Hieruit kan besloten worden dat het met deze beschikbare dataset niet mogelijk is om de stalen in klassen volgens de hoeveelheid FWA te kunnen indelen. Verder onderzoek is nodig om na te gaan of meer data of het optimaliseren van de dataset voor betere resultaten kan zorgen.

4.2.3 Uitbreiding van features naar volledige reflectie-spectra

De X-, Y- en Z-waarden worden bepaald aan de hand van de verschillende delen van het emissie-spectra. Het is daarom ook interessant om de features uit te breiden naar het volledig spectrum. Het spectrum bestaat uit 39 golflengten van 360 nm tot 740 nm. De drie verschillende classificaties die in de vorige paragraaf werden uitgevoerd met de X-, Y- en Z-waarden als features worden nu uitgevoerd met de volledige spectra als features. Bij elke classificatie worden dezelfde dataset en dezelfde parameter instellingen gebruikt als bij de classificaties met de X-, Y- en Z-waarden, weergegeven in respectievelijk Tabel 15 en Tabel12 en wordt de dataset vooraf gestandaardiseerd.

4.2.3.1 Classificatie naargelang de Y-waarden

In Figuur 57 worden de resultaten van de classificaties naargelang de Y-waarden met de X-, Y- en Z-waarden en met de volledige spectra met elkaar vergeleken.

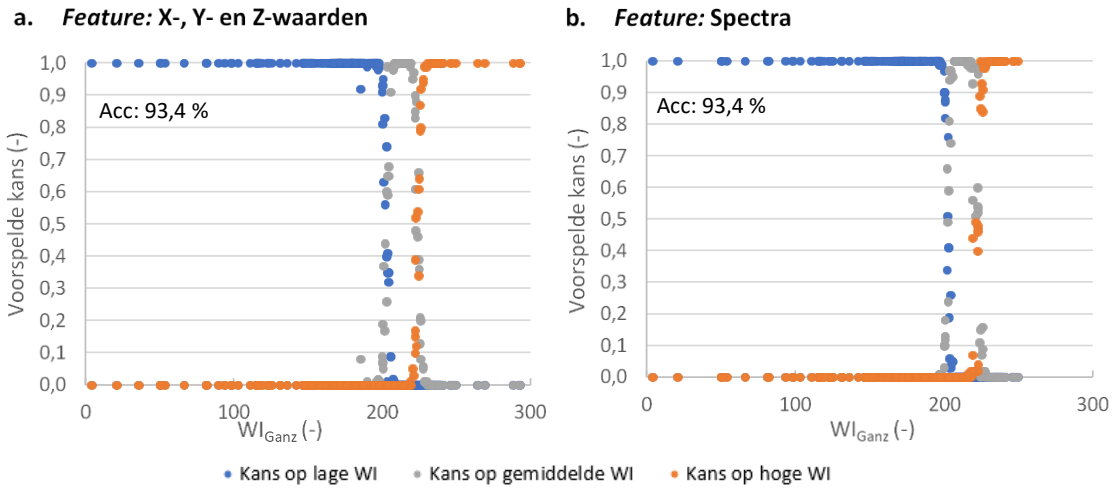


Figuur 57: Vergelijking resultaten van classificatie met als features X-, Y- en Z-waarden en spectra.

De resultaten die bekomen worden met het volledige spectra zijn zeer gelijkaardig aan die met de X-, Y- en Z-waarden. De accuraatheid is zelfs net iets hoger en er worden 20 stalen minder in de zone tussen 0 en 1 voorspeld. Het is dus mogelijk om aan de hand van het emissie-spectra te voorspellen of een staal een goede of een slechte basiswitheid heeft. Enkel de stalen met een Y-waarde tussen 79 en 81 zouden nog verkeerd kunnen worden ingeschat.

4.2.3.2 Classificatie naargelang WI_{Ganz}

In Figuur 57 worden de resultaten van de classificaties naargelang de WI_{Ganz} met de X-, Y- en Z-waarden en met de volledige spectra met elkaar vergeleken.

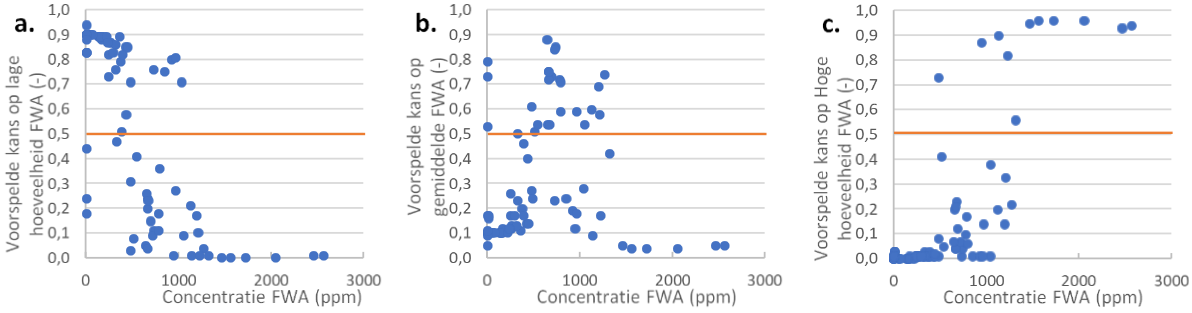


Figuur 58: Vergelijking resultaten van classificatie met als features X-, Y- en Z-waarden en spectra.

De resultaten zijn ook hier zeer gelijkaardig. Met behulp van neurale netwerken kan uit het spectrum worden afgeleid of het gaat om een staal met een hoge, een gemiddelde of een lage WI_{Ganz}.

4.2.3.3 Classificatie naargelang hoeveelheid optische witmaker

In Figuur 57 worden de resultaten van de classificaties naargelang de hoeveelheid FWA met behulp van de volledige spectra getoond. Net zoals bij de classificatie met de X-, Y- en Z-waarden als features, wordt de kans dat een staal tot een bepaalde categorie behoort telkens in een aparte grafiek afgebeeld.



Figuur 59: Resultaten van classificatie naargelang hoeveelheid FWA met volledige spectra als features. (a) Kans op lage hoeveelheid FWA. (b) Kans op gemiddelde hoeveelheid FWA. (c) Kans op hoge hoeveelheid FWA

De accuraatheid van dit model bedraagt 79,8 %. Dit is al hoger dan de accuraatheid van hetzelfde model wanneer er enkel de X-, Y- en Z-waarden als features gebruikt worden. Dit betekent dat het spectrum meer relevante informatie bevat over de hoeveelheid optische witmaker dan de X-, Y- en Z-waarden. Ook is deze keer wel een lichte vorm van classificatie zichtbaar in Figuur 59. In de eerste grafiek daalt over het algemeen de kans dat het staal een lage hoeveelheid FWA bevat naarmate de hoeveelheid FWA groter wordt. Bij de middelste grafiek lijkt de kans dat het een gemiddelde hoeveelheid FWA heeft over het algemeen eerst te stijgen en dan weer te dalen en bij de laatste grafiek stijgt de kans op een grote hoeveelheid FWA naarmate de hoeveelheid FWA stijgt. Toch zijn de grenzen van de categorieën hier niet duidelijk te zien en blijkt de classificatie daardoor niet echt nauwkeurig. Ook hier zal verder onderzoek noodzakelijk zijn om de prestaties te verbeteren.

4.3 Regressies

Het is ook mogelijk om met deep learning een model op te bouwen dat continue waarden kan voorspellen. In dit onderzoek worden twee regressies uitgewerkt. Er wordt gekeken of het mogelijk is om de WI_{Ganz} te voorspellen aan de hand van de X-, Y- en Z-waarden en aan de hand van de volledige reflectie-spectra.

Vervolgens wordt de regressie van hoeveelheid FWA uitgevoerd met behulp van de reflectie-spectra. Ook hier moet voor elke uitgevoerde regressie het stappenplan doorlopen worden. Bij elke regressie worden de gepaste pre-processing methoden en parameter instellingen van het neuraal netwerk apart nagegaan.

De bekomen resultaten worden geëvalueerd aan de hand van de fout op de voorspelling in functie van de werkelijke waarde en de gemiddelde absolute procentuele fout (MAPE) op de voorspellingen. Deze laatste wordt berekend met volgende formule:

$$MAPE = \frac{100\ \%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right| \quad (4.1)$$

Hierin staat n voor het aantal stalen in de test data, A_t voor de werkelijke waarde en F_t voor de voorspelde waarde. Ook de voorspelde waarde in functie van de werkelijke waarde geeft een mooi beeld van de prestaties van het model.

4.3.1 Regressie van WI_{Ganz}

De regressie van de WI_{Ganz} gebeurt met dezelfde dataset als die voor de classificatie van stalen naargelang hun WI_{Ganz}. De informatie over deze dataset werd weergegeven in paragraaf 4.2.1 in Tabel 15. Bij regressie wordt niet meer gewerkt met labels, maar wel met targets. Deze targets zijn de werkelijke WI_{Ganz} waarden van elk staal. In Tabel 17 wordt een voorbeeld gegeven hoe de train en test data eruitziet.

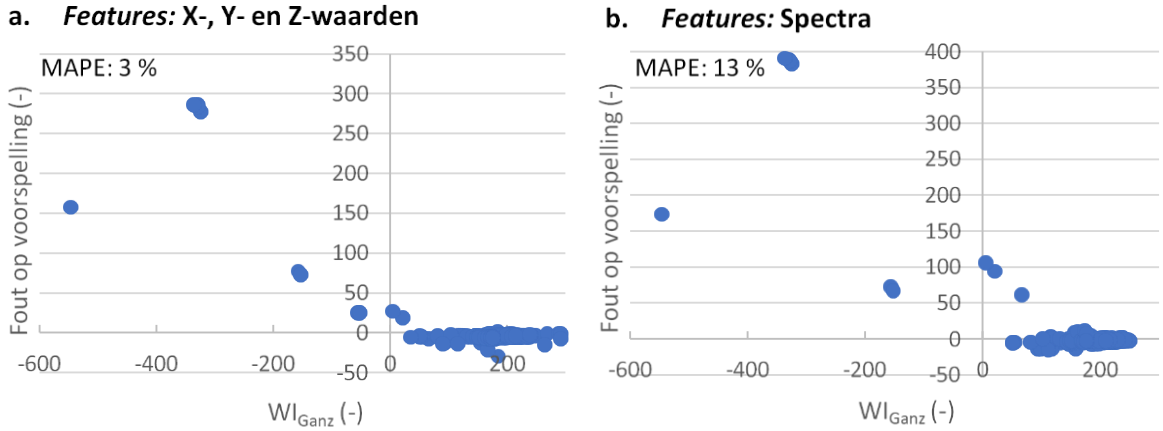
Tabel 17: Voorbeeld van training data voor regressie van WI_{Ganz}.

WI_{Ganz}	X-waarde	Y-waarde	Z-waarde
128,6484	81,602	85,01	97,93
236,35	78,94	80,93	110,43
243,9312	79,03	80,97	111,73
247,173	79,31	81,24	112,60
242,7237	78,21	80,17	110,55
202,1967	82,94	85,62	109,84

In Figuur 60 worden de resultaten van de regressie van de WI aan de hand van de X-, Y- en Z-waarden en aan de hand van de spectra met elkaar vergeleken. De parameter instellingen van het neuraal netwerk die voor beide regressie werden toegepast wordt weergegeven in Tabel 18.

Tabel 18: Parameter instellingen van neuraal netwerk voor regressie van WI_{Ganz}.

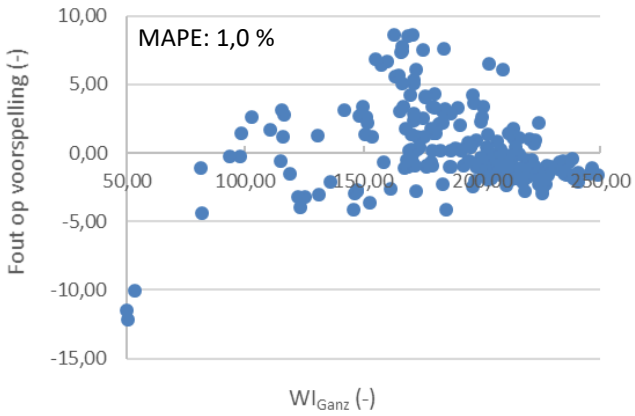
Parameter	Instelling
Intermediate Layers	2
Hidden Units	16
Activation	relu
loss function	MSE
optimizer	RMSProp
metric	MAE
Validation method	simple hold-out validation
Batch size	1
Epoch	100



Figuur 60: Vergelijking resultaten van regressie met als features X-, Y- en Z-waarden en spectra.

De resultaten van de regressie met de X-, Y- en Z-waarden en met de spectra zijn ook hier zeer gelijkaardig. Het valt op dat er bij de stalen met een negatieve WI een zeer grote fout op de voorspelling zit. De fouten bij stalen met een hogere WI lijken op het eerste zicht vrij klein te zijn maar door de hoge fouten van de negatieve WI's is dit moeilijk uit Figuur 60 af te leiden. De hoge fouten bij de stalen met een negatieve WI waarde zorgen voor een MAPE van 3 % voor X-, Y- en Z-waarden en 13 % voor de spectra. Hieruit blijkt dat de WI beter voorspeld wordt met behulp van de X-, Y- en Z-waarden dan met de volledige specta.

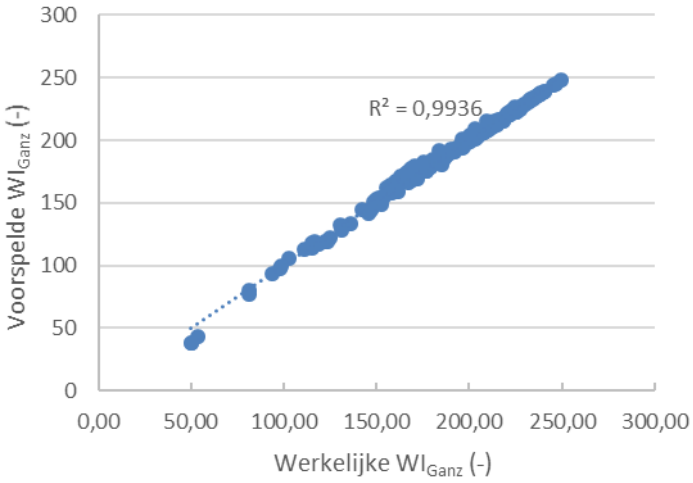
Wanneer met de spectrofotometer een negatieve WI_{Ganz} waarde wordt bekomen, wil dat zeggen dat het staal al zeer geel is en dat er eigenlijk niet meer van een wit staal kan gesproken worden. Het is niet logisch om niet witte stalen te beschrijven met een witheidindex. De spectra van deze stalen wijken bovendien sterk af van de spectra van witte stalen en zouden daardoor het leerproces van het neuraal netwerk verstoren. De regressie wordt daarom nog eens herhaald nadat alle stalen met een negatieve WI_{Ganz} uit de dataset verwijderd zijn. De resultaten voor de regressie met de volledige spectra worden weergegeven in Figuur 61.



Figuur 61: Fout op voorspelling in functie van WI_{Ganz} na aanpassing dataset.

Slechts 10 % van stalen heeft een absolute fout groter dan 5. De MAPE is in dit geval slechts 1,0 %. Met behulp van deep learning kan de WI_{Ganz} dus voldoende goed voorspeld worden aan de hand van het volledige spectra. Uit Figuur 61 blijkt dat vooral de stalen met een WI_{Ganz} rond 50 minder goed voorspeld worden. Wanneer alle spectra uit de testdata bekeken worden blijken deze stalen de enige stalen te zijn waarbij geen emissie piek afkomstig van de FWA aanwezig is. De drie emissie-spectra wijken dus af van die van de andere stalen.

In Figuur 62 worden de werkelijke WI uitgezet in functie van de voorspelde WI. Er wordt een R^2 van 0,994 behaald wat aangeeft dat de waarden nauwkeurig voorspeld worden.



Figuur 62: Voorspelde in functie van werkelijke WI_{Ganz}.

Hieruit blijkt dat het belangrijk is om stalen die niet representatief zijn voor het vooropgestelde doel uit de dataset te verwijderen. Door de dataset op voorhand goed te bestuderen en door het toepassen van bepaalde multivariate analysetechnieken kunnen deze stalen op voorhand al worden opgespoord en verwijderd uit de dataset, waardoor tijd bespaard wordt.

Bij het uitvoeren van de classificaties van de stalen naargelang de WI_{Ganz} in paragrafen 4.2.1.2 en 4.2.2.2 werden de stalen met negatieve WI_{Ganz} nog niet verwijderd. De resultaten zouden hier dus ook verbeterd kunnen worden na aanpassing van de gebruikte test en train data.

4.3.2 Regressie van hoeveelheid optische witmaker

De dataset die beschikbaar is voor regressie van de hoeveelheid FWA met behulp van de volledige spectra bestaat uit 491 stalen. Deze dataset wordt gesplitst in een train en test set, de informatie is weergegeven in Tabel 19. De parameter instellingen van het neuraal netwerk die voor beide regressie werden toegepast wordt weergegeven in Tabel 20.

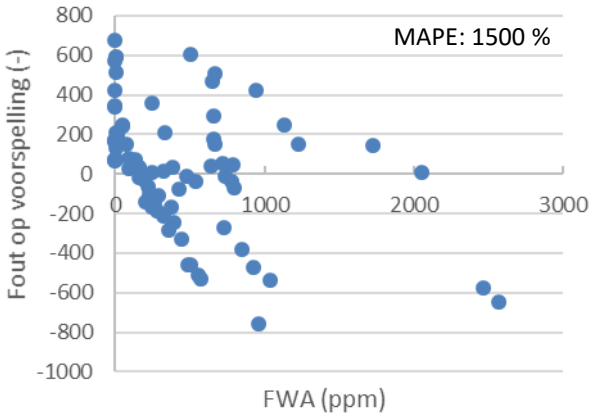
Tabel 19: Gebruikte train en test data voor regressie van hoeveelheid FWA.

Dataset	Totaal aantal stalen	Bereik Conc. FWA (ppm)	Gemiddelde Conc. Conc. FWA (ppm)	Standaarddeviatie Conc. FWA (ppm)
Train	411	0 – 7861	969,2	796,4
Test	80	0 - 2565	444,2	520,5

Tabel 20: Parameter instellingen van neuraal netwerk voor regressie van WI_{Ganz}.

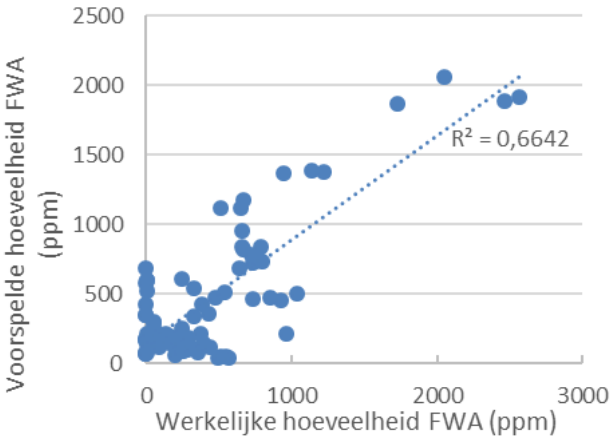
Parameter	Instelling
Intermediate Layers	2
Hidden Units	16
Activation	relu
loss function	MSE
optimizer	RMSProp
metric	MAE
Validation method	Simple hold-out validation
Batch size	1
Epoch	100

De resultaten worden weergegeven in Figuren 63 en 64.



Figuur 63: Fout op voorspelling in functie van hoeveelheid FWA.

Hier wordt de fout tussen de werkelijke en voorspelde waarde uitgezet voor elk staal van de testdata. Voor heel veel stalen wordt de hoeveelheid FWA aanwezig op het staal onnauwkeurig ingeschat. Bij stalen die zeer weinig FWA bevatten wordt er in sommige gevallen zelf voorspeld dat het staal 300 keer meer FWA bevat dan dat er werkelijk aanwezig is. De voorspellingen van het opgebouwde model zijn dus niet betrouwbaar. De MAPE bedraagt 1500 %, dit wil zeggen dat de hoeveelheid FWA gemiddeld 15 keer groter of kleiner wordt ingeschat dan de werkelijke waarden.



Figuur 64: Voorspelde in functie van werkelijke hoeveelheid FWA.

In de grafiek waar de voorspelde waarden worden uitgezet in functie van de werkelijke waarden is wel enigszins een rechte waarneembaar. De R^2 bedraagt 0,664, wat aangeeft dat de voorspellingen niet nauwkeurig zijn.

Hier wordt opnieuw nagegaan of de dataset geen irrelevante stalen bevat. Deze moeten verwijderd worden om dataset te optimaliseren.

Net zoals bij de regressie van de WI_{Ganz} worden alle stalen die een negatieve WI_{Ganz} hebben uit de dataset gehaald.

Daarnaast worden enkel de stalen met een Y-waarde hoger dan 70 bijgehouden. Bij een Y-waarde lager dan 70 gaat het om gekleurde stalen. De emissie-spectra ervan zijn anders dan de witte stalen wat de resultaten negatief kan beïnvloeden.

Er wordt ook enkel gekeken naar katoen stalen. De dataset bevat ook enkele polyester stalen. De affiniteit van de optische witmakers voor polyester is zeer laag, waardoor de concentratie FWA op polyester altijd zeer laag zal zijn. Daarnaast zit FWA 'gevangen' in de polyestervezel, waardoor het niet geëxtraheerd kan worden voor het toepassen van HPLC en er dus een onderschatting gemaakt wordt. In Figuren 63 en 64 is te zien dat vooral stalen met een zeer lage hoeveelheid FWA zeer slecht worden voorspeld. Om de prestaties van het model te verbeteren worden alle stalen met een concentratie FWA onder 100 ppm verwijderd uit de dataset.

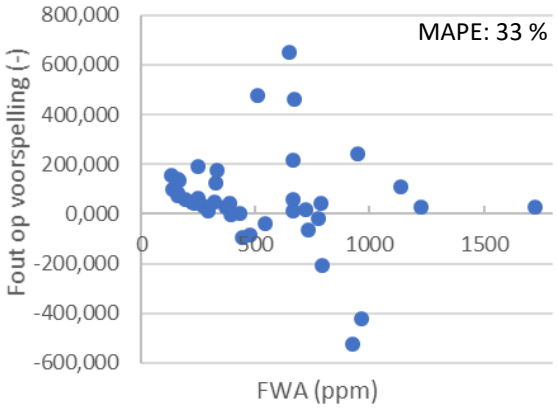
Ook de stalen met een concentratie FWA hoger dan 2000 ppm worden uit de dataset verwijderd. Bij verzadiging van FWA zal het textiel er namelijk niet meer witter uitzien, maar zal er een gele tint ontstaan waardoor de witheidindex zal dalen. Dit omgekeerde effect bij hoge hoeveelheden FWA zal het leerproces negatief beïnvloeden.

Uiteindelijk wordt er een dataset van slechts 323 stalen bekomen, die wordt gesplitst in train en test data. De informatie ervan wordt weergegeven in Tabel 21.

Tabel 21: Aangepaste train en test data voor regressie van hoeveelheid FWA.

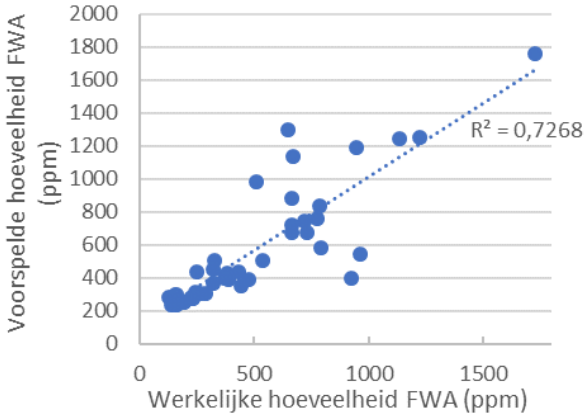
Dataset	Totaal aantal stalen	Bereik (ppm)	Gemiddelde (ppm)	Standaarddeviatie (ppm)
Train	282	101 - 1987	905,9	431,2
Test	41	129 - 1725	520,3	351,8

De resultaten worden weergegeven in Figuren 66 en 67.



Figuur 65: Fout op voorspelling in functie van FWA na aanpassing dataset.

In deze figuur is te zien dat de fout op de voorspelde waarde voor veel stalen nog altijd zeer hoog is. De concentratie FWA wordt maximaal dubbel zo groot voorspeld dan de werkelijke waarde. Gemiddeld worden de hoeveelheid FWA 33 % groter of kleiner ingeschat dan de werkelijke waarden.



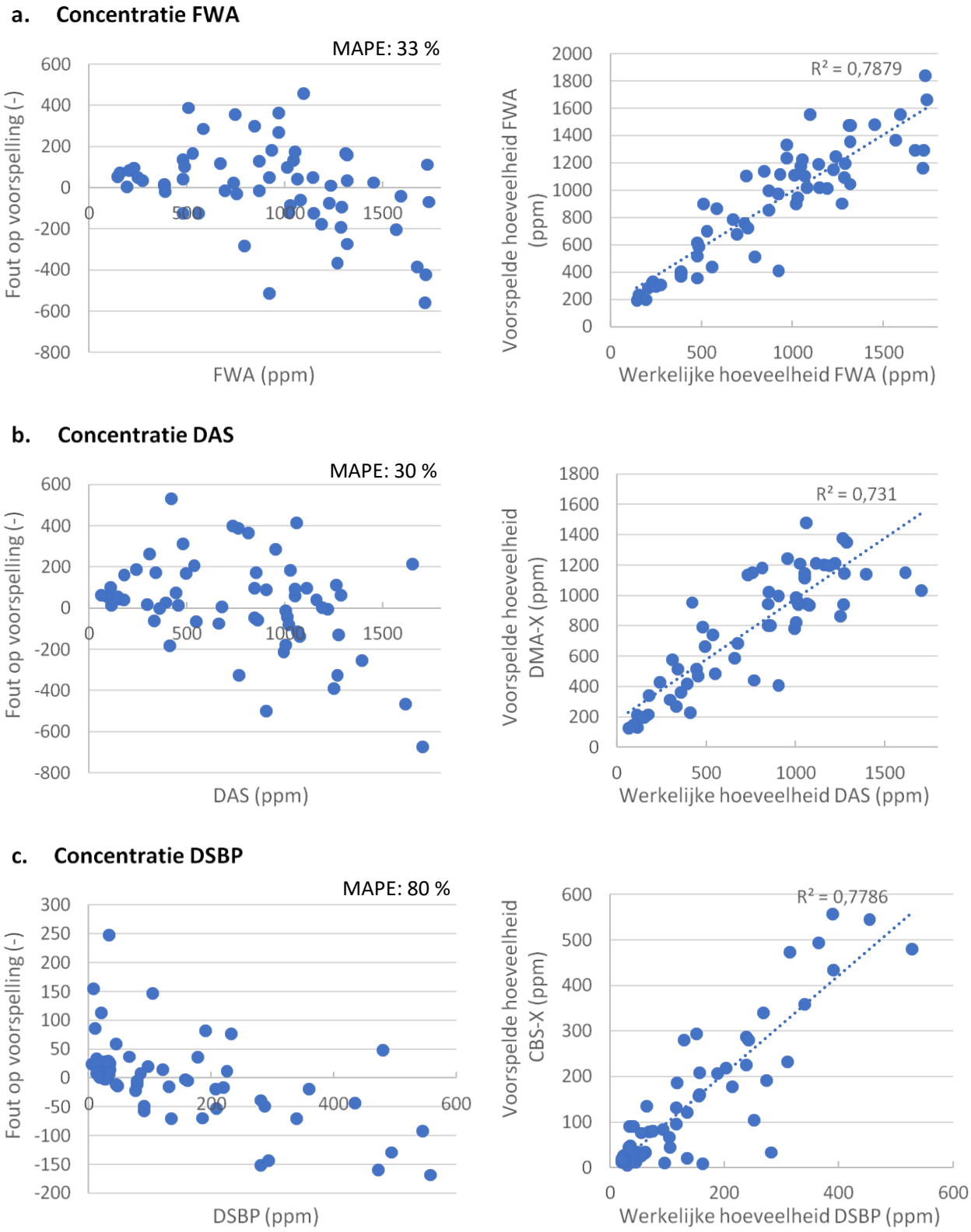
Figuur 66: Voorspelde in functie van werkelijke hoeveelheid FWA.

In de grafiek waar de voorspelde waarden worden uitgezet in functie van de werkelijke waarden is de rechte nog iets meer uitgesproken. De R^2 bedraagt 0,727. Wanneer deze resultaten worden vergeleken met de resultaten van de volledige dataset is dit wel al een grote verbetering. Toch zijn de voorspellingen nog niet betrouwbaar.

Er kunnen verschillende redenen zijn waarom de hoeveelheid FWA bepaald met HPLC niet nauwkeurig kan voorspeld worden aan de hand van de emissie-spectra. Een eerste reden kan zijn dat 282 stalen in de train dataset niet voldoende is om neuraal netwerk te trainen. Hoe meer relevante data er beschikbaar is hoe beter het model in staat zal zijn om nauwkeurig te voorspellen.

Een andere reden kan zijn dat de hoeveelheid FWA op de manier dat het bepaald wordt met de HPLC niet genoeg correleert met de informatie van het spectrum. Bij de HPLC-analyse wordt namelijk de concentratie DAS en DSBP apart bepaald. Voor de totale concentratie FWA wordt dan de som genomen. Het zou kunnen zijn dat de twee verschillende optische witmakers het spectrum op een andere manier beïnvloeden. De som van de DSBP en de DAS geeft dan geen juiste representatie van de informatie in het spectrum.

Om hier dieper op in te gaan wordt nagegaan of het neuraal netwerk beter in staat zou zijn om de hoeveelheid DSBP en DAS apart te voorspellen in plaats van de som ervan. In Figuur 67 worden de resultaten van de regressie van de totale concentratie FWA, de concentratie DSBP en de concentratie DAS met elkaar vergeleken.

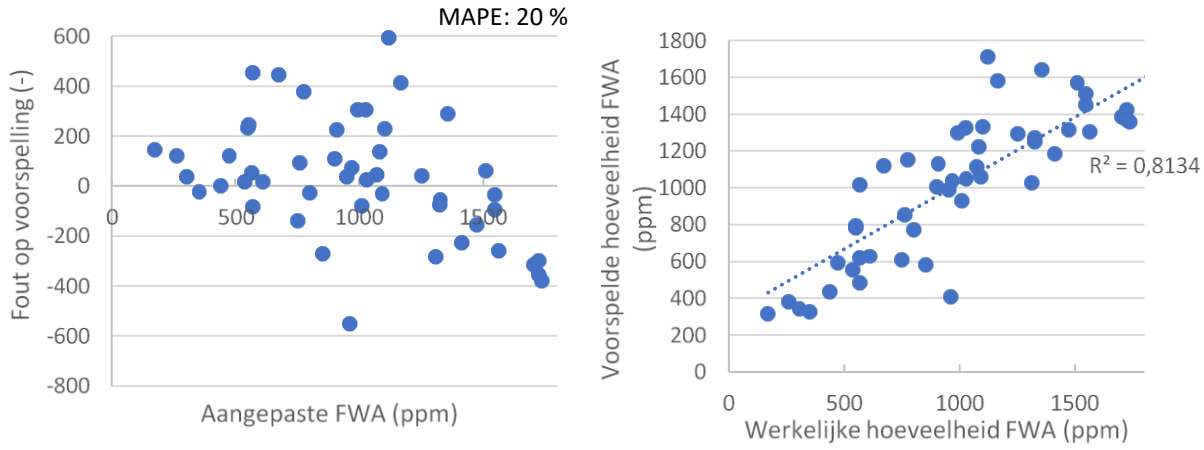


Figuur 67: Vergelijking resultaten van regressie van (a) conc. FWA, (b) conc. DAS en (c) conc DSBP.

Hieruit blijkt dat resultaten van de regressie voor de drie concentraties zeer gelijkaardig zijn. De absolute fouten op de voorspellingen zijn lager bij de voorspellingen van DSBP. Dit komt omdat de concentraties van DSBP op het textiel lager liggen dan de concentraties van de DAS. Wanneer gekeken wordt naar de MAPE blijkt echter dat deze bij DSBP 80 % is, terwijl dit bij DAS slechts 30 % is. Het is dus moeilijk te zeggen of één van de twee optische witmakers aanzienlijk beter voorspeld wordt dan de andere.

Er is nog een reden waarom het optellen van de concentratie van de twee optische witmakers niet representatief zou zijn voor de volledige concentratie aan optische witmakers. De absorptie coëfficiënt van DSBP zou drie keer groter zijn dan die van DAS. Dit wil zeggen dat DSPB types drie keer hogere absorptie en emissie gaan vertonen (op gewichtsbasis) in verhouding met DAS type optische witmakers. Het effect van eenzelfde concentratie van DSBP zou dus drie keer groter zijn dan bij DAS.

Om die reden wordt de regressie ook eens uitgevoerd met de som van de concentratie DAS en driemaal de concentratie DSBP. Op die manier wordt het driedubbele effect van de DSBP in rekening gebracht. De resultaten van deze regressie worden weergegeven in Figuur 68.



Figuur 68: Resultaten van regressie van aangepaste hoeveelheid FWA.

Nog steeds is er bij vele stalen een aanzienlijk verschil tussen de werkelijke en de voorspelde concentratie FWA. Gemiddeld wijkt de voorspelde waarde 20 % af van de werkelijke waarde. Dit is het beste resultaat dat werd bekomen voor de regressie van hoeveelheid FWA.

De werkelijke invloed van de DSBP en de DAS op het reflectie-spectrum zou kunnen nagegaan worden door stalen met enkel DSBP en DAS te onderzoeken. Multivariate data-analyses zoals PCA en PLS zouden gebruikt kunnen worden om aan te tonen welke delen van het spectra belangrijk zijn voor het voorspellen van de hoeveelheid de DSBP en de DAS.

Nog een mogelijke reden waarom de regressie van hoeveelheid FWA niet nauwkeurig kan voorspeld worden aan de hand van de reflectie-spectra, is omdat de dataset nog te veel afwijkende stalen bevat, die het leerproces verstoren. Sommige stalen bevatten namelijk bluing agents, wat ook een invloed zal hebben op het spectrum en bij sommige stalen is er een te hoge concentratie surfactans of andere component aanwezig op het textiel wat als *bocking agent* dient, waardoor de uitstraling van de FWA vermindert. Al deze stalen gaan een negatieve invloed hebben op de prestaties van het model. Het is niet mogelijk om alle stalen op voorhand één voor één te bestuderen om na te gaan of er al dan niet bluing agents of

blocking agents aanwezig zijn. Ook hier kunnen de multivariate analysetechnieken PCA en PLS helpen om afwijkende patronen in de dataset op te sporen, om op die manier niet relevante stalen uit de dataset te verwijderen.

4.4 Advies voor verder onderzoek

Uit het onderzoek blijk dat dataoptimalisatie een belangrijke stap is bij het toepassen van deep learning op een bepaalde dataset. Stalen die foute of irrelevante informatie bevatten voor de te voorspellen waarde hebben een negatieve invloed op het leerproces, met slechtere resultaten als gevolg. De stalen die afwijken zijn vaak niet op het eerste zicht in de dataset te zien. In dit onderzoek werden zo veel mogelijk afwijkende stalen handmatig uit de dataset verwijderd. Het is niet mogelijk om voor alle stalen apart na te gaan of ze representatieve informatie bevatten. Om alle afwijkende stalen in de dataset op te sporen zouden verschillende andere multivariate data-analyse methoden gebruikt kunnen worden. De technieken PCA en PLS lijken veelbelovend om meer inzicht in de dataset te verkrijgen en daarmee zou het ook mogelijk kunnen zijn de afwijkende stalen op te sporen. Eerste experimenten met deze technieken werden al uitgevoerd en de resultaten leken veelbelovend. Deze technieken zijn echter complex en er was niet genoeg tijd om ze voldoende te beheersen, waardoor de resultaten ervan ook niet in deze thesis werden opgenomen. Verder onderzoek zal laten blijken of PLS en PCA geschikt zouden zijn voor het optimaliseren van de dataset.

Het gebruik van PCA en PLS als pre-processing methoden zal in verder onderzoek steeds meer van belang worden. De volgende stap in het onderzoek is echter om de features uit te breiden naar FTIR-spectra. Deze spectra bestaan uit 2179 golflengten. Wanneer al deze 2179 golflengten als aparte variabelen gebruikt worden bij het trainen van een neuraal netwerk zal het leerproces zeer veel tijd in beslag nemen, bovendien zullen binnen het spectrum veel golflengten irrelevant zijn voor de bepaalde classificatie of regressie, wat een slechte nauwkeurigheid en de stabiliteit van het model kan veroorzaken. Technieken zoals PCA en PLS zijn in dit geval noodzakelijk om de dimensies van de data te verkleinen en enkel de interessante informatie over te houden. (60)(61)

5 BESLUIT

Deze thesis is de start van een volledig nieuw onderzoek naar de verwerking van de dataset die bij Christeyns verworven is door de analyse van textielstalen. Binnenkomende textielstalen worden geanalyseerd aan de hand van drie verschillende analysetechnieken, namelijk witheidsmetingen met een spectrofotometer, FTIR-spectroscopie en HPLC. Per staal wordt dus een aanzienlijke hoeveelheid gegevens gegenereerd. De gegevens van alle geanalyseerde stalen door de jaren heen worden opgeslagen in een steeds groter wordende database. In deze thesis werd onderzoek gedaan naar verschillende methoden voor de verwerking van data afkomstig van de analyse van textielstalen.

Uit de theoretische studie over hoe moet omgegaan worden met zo een dataset blijken verschillende multivariate dataverwerkingsmethoden zoals PCA, LDA en PLS geschikt voor het onthullen van verborgen informatie in de dataset en het opzoeken van complexe relaties tussen verschillende variabelen. In deze thesis werd echter gefocust op artificiële intelligentie en in het bijzonder deep learning methoden voor de verwerking van de dataset.

Deep learning is een techniek dat gebruik maakt van zogenaamde neurale netwerken waarbij de best mogelijke verbinding wordt gedetecteerd tussen input- en outputdata. Een neuraal netwerk wordt geoptimaliseerd tot een minimaal verschil tussen reële waarden en voorspelde waarde bereikt wordt. Neurale netwerken kunnen patronen in de dataset zelf 'leren' en hun eigen model bouwen voor interpretatie en voorspellingen van onbekende monsters. Neurale netwerken zijn geschikt voor het uitvoeren van classificaties van monsters in afzonderlijke groepen en voor regressies, waarbij continue waarden worden voorspeld.

De invloed van verschillende parameters op de prestaties van het neuraal netwerk werden nagegaan. Veel van deze parameters, zoals de grote van de dataset, de uitgevoerde pre-processing methode of de parameterinstellingen van het neuraal netwerk zijn sterk afhankelijk van de complexiteit van het vooropgestelde doel en kunnen alleen via trial and error worden nagegaan. Dit maakt het opbouwen van een geschikt neuraal netwerkmodel voor het classificeren of voorspellen van onbekende data een zeer iteratief proces.

In totaal waren er 945 stalen van de witheidsmetingen aanwezig. Dit blijkt voldoende voor de classificatie en regressie van de stalen naargelang hun basiswitheid en witheid index. Na de optimalisatie van de dataset met informatie over de concentratie FWA, blijven hier nog maar 323 stalen over. Hoe meer data beschikbaar is, hoe accurater het neuraal netwerk voorspellingen kan doen. Meer relevante data om het neuraal netwerk te trainen zou hier dus voor betere resultaten kunnen zorgen.

Uit het onderzoek kwam ook het belang van preprocessing methoden naar boven. Bij hoge input waarden werkt het neuraal netwerk namelijk met hoge weights wat het leerproces bemoeilijkt en onstabiel maakt. Met preprocessing methoden zoals normaliseren en standaardiseren worden de input waarden naar kleinere waarden getransformeerd, wat in veel gevallen de prestaties van het netwerk verbetert. Uit dit onderzoek bleek dat de beste resultaten van alle uitgevoerde classificaties en regressies bekomen werden wanneer de input data vooraf gestandaardiseerd werd. Uit literatuur blijkt dat dit niet altijd het geval zal zijn.

In het onderzoek werden verschillende classificaties en regressies uitgevoerd met verschillende gegevens uit de beschikbare dataset. Er wordt hierbij getracht om bepaalde eigenschappen van het textiel zoals de basiswitheid, de witheid index en de hoeveelheid optische witmaker te voorspellen, aan de hand van X-, Y- en Z-waarden enerzijds en de volledige reflectie-spectra anderzijds.

Zodra een geschikt neuraal netwerkmodel werd opgebouwd, was het mogelijk om de stalen in verschillende klassen in te delen naargelang hun basiswitheid. In deze classificatie werd 99% van de monsters in de juiste klasse geclassificeerd. De accuraatheid lijkt iets lager te zijn in de classificatie volgens de witheidindex. Bij het meest geschikte model werd hier slechts 93 % van de monsters in de juiste klasse ingedeeld. Deze resultaten zouden wel nog verbeterd kunnen worden door de afwijkende stalen uit de dataset te verwijderen.

Bij het uitvoeren van de regressie van de witheidindex werden de afwijkende stalen met een negatieve witheidindex wel uit de dataset verwijderd. De gemiddelde absolute procentuele fout tussen de voorspelde en de werkelijke witheidindex is hier 1,0 %. Er kan dus besloten worden dat het neuraal netwerk in staat is om aan de hand van zowel de X-, Y- en Z-waarden als de spectra de witheidindex van onbekende stalen voldoende nauwkeurig te voorspellen.

In een volgende stap wordt getracht om de gegevens afkomstig van de witheidsmetingen namelijk de X-, Y- en Z-waarden en reflectie-spectra te koppelen aan gegevens afkomstig van HPLC. Er werd onderzocht of de concentratie optische witmaker, die bepaald wordt met HPLC, kan voorspeld worden aan de hand van de reflectie-spectra van de witheidsmetingen. Uit de resultaten blijkt dat het niet mogelijk is om de stalen te classificeren naargelang hun concentratie FWA met behulp van de X-, Y- en Z-waarden. De resultaten verbeterden wanneer de features werden uitgebreid naar de emissie-spectra. Er werd een accuraatheid van 80 % behaald. Bij het uitvoeren van de regressie werd in de eerste instantie een model verkregen waarmee de hoeveelheid optische witmaker voorspeld kon worden met een MAPE van 1500%. Dit betekent dat de voorspelde concentratie van optische witmakers gemiddeld 15 keer meer of 15 keer minder is dan de werkelijke concentratie, wat dus geen goede voorspellingen zijn. Uit de resultaten bleek dat ook hier irrelevante stalen in de dataset aanwezig waren. Na optimalisatie van de dataset en nadat in rekening werd gebracht dat het effect van een bepaalde hoeveelheid DSBP drie keer groter zou zijn dan het effect van DAS werd uiteindelijk een model bekomen met een MAPE van 20 %, wat toch al een grote verbetering is.

Dit onderzoek toont aan dat deep learning een veelbelovende techniek kan zijn voor de verwerking van de dataset en het vinden van correlaties tussen de gegevens van de drie verschillende textiel analysemethoden. Deze thesis is echter nog maar het begin van het onderzoek en het is duidelijk dat met deze techniek nog veel verder kan gegaan worden. Zo zou deep learning in een volgende stap kunnen toegepast worden op de FTIR-spectra, waarmee mogelijks de concentratie van andere additieven zoals wasverzachters en andere oppervlakte-actieve stoffen zou voorspeld kunnen worden.

Uit het onderzoek in deze thesis blijkt ten slotte ook nog dat de combinatie van verschillende multivariate technieken zoals PCA en PLS bij verder onderzoek steeds belangrijker zal worden als pre-processing methoden voor het optimaliseren van de dataset en daarmee het verbeteren van de prestaties van de neurale netwerken.

BIBLIOGRAFIE

1. Christeyns NV [Internet]. [cited 2020 Apr 22]. Available from: https://www.christeyns.com/nl
2. Rosen MJ. Surfactants and interfacial phenomena. Sons. JW&, editor. Hoboken, New Jersey; 1989. 444 p.
3. Schmitt TM. Analysis of Surfactants. 2nd ed. New York: Taylor & Francis; 2001.
4. Showell MS. Handbook of Detergents Part D: Formulation. New york: Taylor & Francis Group; 2006.
5. Gerlache M, Kauffmann JM, Quarin G, Vire JC, Bryant GA, Talbot JM. Electrochemical analysis of surfactants: An overview. Talanta. 1996;43(4):507–19.
6. Kosswig K. Surfactans. In: KGaA. W-VVG& C, editor. Ullmann's encexlopedia of industrial chemistry. Weilheim, Germany; 2000. p. 432–501.
7. Levinson MI. Rinse-added fabric softener technology at the close of the twentieth century. J Surfactants Deterg. 1999;2(2):223–35.
8. Saraiva1 SA, Abdelnur P V., Catharino RR, Nunes G, Eberlin1 MN. Fabric softeners: nearly instantaneous characterization and quality control of cationic surfactants by easy ambient sonic-spray ionization mass spectrometry. Rapid Commun Mass Spectrom. 2009;23:357–62.
9. Davey HM, Kell DB. Fluorescent brighteners: Novel stains for the flow cytometric analysis of microorganisms. Cytometry. 1997;28(4):311–5.
10. Bruneel D. Cursus toepassingen in de chemische industrie: Deel: Toepassingen. In: Cursus toepassingen in de chemische industrie: Deel: Toepassingen. 2019.
11. BASF. Technical information Tinopal® DMA-X [Internet]. Tinopal® DMA-X. 2011 [cited 2020 Mar 12]. p. 1–4. Available from: file:///C:/Users/marjo/Downloads/Tinopal_DMA-X_TI_en.pdf
12. VestaChemicals. Viobrite DMS-X. 2020.
13. BASF. Technical Information TINOPAL CBS-X [Internet]. 2011. p. 1–4. Available from: http://www.hss.gov.yk.ca/homecare.php
14. BASF. Safety data sheet Tinopal CBS-X [Internet]. 2018. Available from: https://www.sdsinventory.com/substances/accessSDS/SDS-5638-5c333a86eab727.42749969
15. Pupchem. Disodium 4,4'-bis(2-sulfostyryl)biphenyl, (Z,Z)- _ C28H20Na2O6S2 [Internet]. [cited 2020 Mar 14]. Available from: https://pubchem.ncbi.nlm.nih.gov/compound/6434006
16. POWER POINT.
17. Shoemaker ML, Hughes DN, Kuchta steven L. METHOD FOR CORRELATING COLOR MEASURING SCALES. United States; 5,150,199, 1992.
18. De Vrindt L. Relatie tussen CIELAB en visuele kleurwaarneming. Ku leuven; 2019.
19. Precise Color Communication. Konica Minolta; 2007.
20. Števek J, Katuščák S, Dubinyová L, Fikar M. An automatic identification of wood materials from color images. 2016 Cybern Informatics, K I 2016 - Proc 28th Int Conf. 2016;200280301(2003).

21. Ma S, Wei M, Liang J, Wang B, Chen Y, Pointer M, et al. Evaluation of whiteness metrics. Light Res Technol. 2018;50(3):429–45.

22. Dietz C. Whiteness indices and UV standards. Konica Minolta. 2011;(2).

23. Peets P, Leito I, Pelt J, Vahur S. Identification and classification of textile fibres using ATR-FT-IR spectroscopy with chemometric methods. Spectrochim Acta - Part A Mol Biomol Spectrosc. 2017;173:175–81.

24. Van de Voorde I. Spectroscopische technieken: Infrarood. KU Leuven; 2018.

25. Banwell CN. Fundamentals of Molecular Spectroscopy. 3rd ed. London: McGraw-Hill International; 1983.

26. Roex H. Spectroscopie. Acco, editor. Leuven; 2017.

27. Bunker Optics Inc. Attenuated Total Reflection (ATR) – a versatile tool for FT-IR spectroscopy. Appl Note AN # 79. 2011;4.

28. Busca G. The use of vibrational spectroscopies in studies of heterogeneous catalysis by metal oxides: An introduction. Catal Today. 1996;27(3–4):323–52.

29. Manyika J, Chui Brown M, B. J. B, Dobbs R, Roxburgh C, Hung Byers A. Big data: The next frontier for innovation, competition and productivity. McKinsey Glob Inst [Internet]. 2011;(June):156. Available from: https://bigdatawg.nist.gov/pdf/MGI_big_data_full_report.pdf

30. Dearing T. Fundamentals of Chemometrics and Modeling. CPAC. 2010.

31. Rajalahti T, Kvalheim OM. Multivariate data analysis in pharmaceutics: A tutorial review. Int J Pharm [Internet]. 2011;417:280–90. Available from: http://dx.doi.org/10.1016/j.ijpharm.2011.02.019

32. Jacyna J, Kordalewska M, Markuszewski MJ. Design of Experiments in metabolomics-related studies : An overview. J Pharm Biomed Anal [Internet]. 2019;164:598–606. Available from: https://doi.org/10.1016/j.jpba.2018.11.027

33. Filzmoser P, Varmuza K. Introduction to multivatiate statistical analysis in chemometrics. Boca Raton: Taylor & Francis Group; 2008.

34. Box GEP, Hunter JS, Hunter WG. Statistics for Experilllenters. 2005;623.

35. ElMasry G, Kamruzzaman M, Sun DW, Allen P. Principles and Applications of Hyperspectral Imaging in Quality Evaluation of Agro-Food Products: A Review. Crit Rev Food Sci Nutr. 2012;52(11):999–1023.

36. Li X, Nsofor GC, Song L. A comparative analysis of predictive data mining techniques. Int J Rapid Manuf. 2009;1(2):150.

37. Pearce TC, Schhiffman SS, Nagle HT, Gardner JW. Handbook of Machine Olfaction: Electronic Nose Technology. Weinheim: Wiley-VCH Verlag GmbH &Co; 2003. 3–527 p.

38. Heater B. PCA basics. 2014;

39. Jolliffe IT. Principal Component Analysis. New York: Springer; 1986.

40. Kumar S, Kumar S, Mishra P. Multivariate analysis: An overview. .journalofdentofacialsciences. 2013;2(3):19–26.

41. Steinfath M, Groth D, Lisec J, Selbig J. Metabolite profile analysis: From raw data to regression and classification. Physiol Plant. 2008;132(2):150–61.

42. Ouyang M, Zhang Z, Chen C, Liu X, Liang Y. Application of sparse linear discriminant analysis for metabolomics data. Anal Methods. 2014;6(22):9037–44.

43. Yang J, Xu J, Zhang X, Wu C, Lin T, Ying Y. Deep learning for vibrational spectral

analysis: Recent progress and a practical guide. Anal Chim Acta [Internet]. 2019;1081:6–17. Available from: https://doi.org/10.1016/j.aca.2019.06.012

44. Filzmoser P, Gschwandtner M, Todorov V. Review of sparse methods in regression and classification with application to chemometrics. J Chemom. 2012;26(3–4):42–51.

45. Chollet F. Deep Learning mit Python und Keras: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek [Internet]. Manning Publications Co.; 2018. p. 447. Available from: https://books.google.de/books?id=ouVcDwAAQBAJ

46. Srivastava T. Difference between Machine Learning & Statistical Modeling. Analytics Vidhya [Internet]. 2015; Available from: https://www.analyticsvidhya.com/blog/2015/07/difference-machine-learning-statistical-modeling/

47. Amini A. MIT Introduction to Deep Learning | 6.S191 [Internet]. 2020. Available from: https://www.youtube.com/watch?v=njKP3FqW3Sk&t=1269s

48. Grimson E. Introduction to Machine Learning [Internet]. MIT OpenCourseWare; 2016. Available from: https://www.youtube.com/watch?v=h0e2HAPTGF4&t=670s

49. Eynikel J. Robot aan het stuur. 4th ed. Tielt: Uitgeverij Lannoo nv; 2018.

50. Feng J, He X, Teng Q, Ren C, Chen H, Li Y. Reconstruction of porous media from extremely limited information using conditional generative adversarial networks. Phys Rev E [Internet]. 2019;100(3):33308. Available from: https://doi.org/10.1103/PhysRevE.100.033308

51. Keras. Layer activation functions [Internet]. [cited 2020 Jun 1]. Available from: https://keras.io/api/layers/activations/

52. Brownlee J. What is the Difference Between a Batch and an Epoch in a Neural Network? [Internet]. Machine Learning Mastery. 2018 [cited 2020 Jun 1]. p. 3–4. Available from: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

53. Mishra A. Amazon Machine Learning. Mach Learn AWS Cloud. 2019;317–51.

54. Chord diagram [Internet]. www.fromdatatoviz.be. [cited 2020 Mar 30]. Available from: https://www.data-to-viz.com/graph/chord.html

55. Keras. About Keras [Internet]. 2020 [cited 2020 Jun 1]. Available from: https://keras.io/about/

56. Brownlee J. TensorFlow 2 Tutorial: Get Started in Deep Learning With tf.keras [Internet]. Machinelearningmastery.com. 2020. Available from: https://machinelearningmastery.com/tensorflow-tutorial-deep-learning-with-tf-keras/

57. Developer N. Jetson Nano [Internet]. 2020 [cited 2020 May 25]. Available from: https://developer.nvidia.com/embedded/jetson-nano

58. Raspberry PI 4 [Internet]. [cited 2020 May 25]. Available from: raspberrypi.org

59. Kaggle. House Prices: Advanced Regression Techniques [Internet]. 2020 [cited 2020 May 25]. Available from: https://www.kaggle.com/c/house-prices-advanced-regression-techniques

60. Brownlee J. How to use Data Scaling Improve Deep Learning Model Stability and Performance [Internet]. Machine Learning Matter. 2019 [cited 2020 May 26]. p. 1–39. Available from: https://machinelearningmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with-data-scaling/

61. Rinnan Å, Berg F van den, Engelsen SB. Review of the most common pre-processing techniques for near-infrared spectra. TrAC - Trends Anal Chem. 2009;28(10):1201–22.

Bijlagen

Bijlage A	Script in Python voor het classificeren van data in twee verschillende klassen.
Bijlage B	Script in Python voor het classificeren van data in meerdere klassen.
Bijlage C	Script in Python voor regressie van data.

Bijlage A SCRIPT IN PYTHON VOOR HET CLASSIFICEREN VAN DATA IN TWEE VERSCHILLENDE KLASSEN

```
import os
import sys
import xlrd
import keras
import numpy as np
import matplotlib.pyplot as plt
from keras import models
from keras import layers
from keras.utils import to_categorical
from keras import optimizers
from keras.utils.np_utils import to_categorical
#-------------------Collect data-------------------
# Directory to text files containing train and test data + labels
path_to_train = "/home/pi/Desktop/Train_Data_Yvalue/Train_Data_Yvalue.txt"
path_to_test = "/home/pi/Desktop/Test_Data_Yvalue/Test_Data_Yvalue.txt"

# Create some empty lists to store our train_data, train_label, test_data, test_label
train_data = []
train_label = []

test_data = []
test_label = []
int_labels = []

# Import data from text files and store them in the lists defined above
with open(path_to_train) as opentrainfile:
    for line in opentrainfile:
        train_data_line =[]
        train_label_line =[]
        line = line.split()
        Y_value_train = line[0].replace(",",".")
        Label_train = line[1]
        train_data_line.append(Y_value_train)
        train_data.append(train_data_line)
        train_label_line.append(Label_train)
        train_label.append(train_label_line)

with open(path_to_test) as opentestfile:
    for line in opentestfile:
        test_data_line =[]
        test_label_line =[]
        line = line.split()
        Y_value_test = line[0].replace(",",".")

#-------------------Prepare data-------------------

x_train = np.array(train_data).astype('float32')
x_test = np.array(test_data).astype('float32')
y_train = np.asarray(train_label).astype('int')
y_test = np.asarray(test_label).astype('int')

#--------------------scale data-------------------

#MAX = x_train.max(axis=0)
#MIN = x_train.min(axis=0)
#x_train -= MIN
#x_train /= (MAX-MIN)
#x_test -= MIN
#x_test /= (MAX-MIN)

#mean = x_train.mean(axis=0)
#x_train -= mean
#std = x_train.std(axis=0)
#x_train /= std
#x_test -= mean
#x_test /= std

#-------------------Building Network-------------------

model = models.Sequential()
model.add(layers.Dense(1, activation = 'tanh', input_shape =(1,)))
model.add(layers.Dense(1, activation = 'tanh'))
model.add(layers.Dense(1, activation = 'sigmoid'))

#-------------------Training Model-------------------

model.compile(optimizer = 'rmsprop', loss = 'binary_crossentropy', metrics= ['acc'])

#-------------------Validation Model-------------------

# split our training set in 2 groups with a similar size
x_val = x_train[:461]
partial_x_train = x_train[461:]

y_val = y_train[:461]
partial_y_train = y_train[461:]
```

```
92
93  # Number of epochs = number of iterations over mini-batches of 20 samples (batch_size)
94  history = model.fit(partial_x_train, partial_y_train, epochs=20, batch_size=20, validation_data = (x_val, y_val))
95
96  history_dict = history.history
97  history_dict.keys()
98  print("Keys :", history_dict.keys())
99
100 acc_values = history_dict['acc']
101 val_acc_values = history_dict['val_acc']
102
103 loss_values = history_dict['loss']
104 val_loss_values = history_dict['val_loss']
105
106 epochs = range(1, len(acc_values)+1)
107
108 # plotting the training and validation loss
109 plt.plot(epochs, loss_values, 'bo', label='Training Loss')
110 plt.plot(epochs, val_loss_values, 'b', label='Validation Loss')
111 plt.title('Training and validation Loss')
112 plt.xlabel('Epochs')
113 plt.ylabel('Loss')
114 plt.legend()
115
116 plt.show()
117
118 # plotting the training and validation accuracy
119 plt.plot(epochs, acc_values, 'bo', label='Training Acc')
120 plt.plot(epochs, val_acc_values, 'b', label='Validation Acc')
121 plt.title('Training and validation Accuracy')
122 plt.xlabel('Epochs')
123 plt.ylabel('Accuracy')
124 plt.legend()
125
126 plt.show()
127
128 #-------------------Generate predictions on new data-------------------
129 predictions = model.predict(x_test)
130 print("predictions on new data (test_data): ", predictions)
131
132 # write predictions to text file 'Results_Model.txt'
133 with open(path_to_Results, 'w+') as openresults:
134     for line in predictions:
135         np.savetxt(openresults, line, fmt='%.2f')
136
```

Bijlage B SCRIPT IN PYTHON VOOR HET CLASSIFICEREN VAN DATA IN MEERDERE KLASSEN

```python
import os
import sys
import xlrd
import keras
import numpy as np
import matplotlib.pyplot as plt
from keras import models
from keras import layers
from keras.utils import to_categorical
from keras import optimizers
from keras.utils.np_utils import to_categorical

#-------------------Collect data-------------------
# Directory to text files containing train and test data + labels
path_to_train = "/home/pi/Desktop/multiclass 3 train_data_Yvalue.txt"
path_to_test = "/home/pi/Desktop/Multiclass 3 Test_Data_Yvalue.txt"

# Create some empty lists to store our train_data, train_label, test_data, test_label
train_data = []
train_label = []

test_data = []
test_label = []
int_labels = []

# Import data from text files and store them in the lists defined above
with open(path_to_train) as opentrainfile:
    for line in opentrainfile:
        train_data_line =[]
        train_label_line =[]
        line = line.split()
        Y_value_train = line[0].replace(",",".")
        Label_train = line[1]
        train_data_line.append(Y_value_train)
        train_data.append(train_data_line)
        train_label_line.append(Label_train)
        train_label.append(train_label_line)

with open(path_to_test) as opentestfile:
    for line in opentestfile:
        test_data_line =[]
        test_label_line =[]
        line = line.split()
        Y_value_test = line[0].replace(",",".")
        Label_test = line[1]
        test_data_line.append(Y_value_test)
        test_data.append(test_data_line)
        test_label_line.append(Label_test)
        test_label.append(test_label_line)

one_hot_train_labels = to_categorical(train_label)
one_hot_test_labels = to_categorical(test_label)

#-------------------Prepare data-------------------

x_train = np.array(train_data).astype('float32')
x_test = np.array(test_data).astype('float32')
int_label = np.unique(train_label)

y_train = np.asarray(one_hot_train_labels).astype('int')
y_test = np.asarray(one_hot_test_labels).astype('int')

#--------------------scale data-------------------

#MAX = x_train.max(axis=0)
#MIN = x_train.min(axis=0)
#x_train -= MIN
#x_train /= (MAX-MIN)
#x_test -= MIN
#x_test /= (MAX-MIN)

#mean = x_train.mean(axis=0)
#x_train -= mean
#std = x_train.std(axis=0)
#x_train /= std
#x_test -= mean
#x_test /= std

#-------------------Building Network-------------------

model = models.Sequential()
model.add(layers.Dense(16, activation='relu', input_shape=(1,)))
model.add(layers.Dense(16, activation='relu'))
model.add(layers.Dense(3, activation='softmax'))

#-------------------Training Model-------------------

model.compile(optimizer = 'rmsprop', loss = 'categorical_crossentropy', metrics= ['acc'])

```

```
#-------------------Validation Model-------------------

# split our training set in 2 groups with a similar size
x_val = x_train[:400]
partial_x_train = x_train[400:]

y_val = y_train[:400]
partial_y_train = y_train[400:]

# Number of epochs = number of iterations over mini-batches of 20 samples (batch_size)
history = model.fit(partial_x_train, partial_y_train, epochs=20, batch_size=20, validation_data = (x_val, y_val))

history_dict = history.history
history_dict.keys()
print("Keys :", history_dict.keys())

acc_values = history_dict['acc']
val_acc_values = history_dict['val_acc']

loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']

epochs = range(1, len(acc_values)+1)

#plt.plot(x_train,y_train,'bo')
#plt.show()


# plotting the training and validation loss
plt.plot(epochs, loss_values, 'bo', label='Training Loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation Loss')
plt.title('Training and validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.show()

#-------------------Generate predictions on new data-------------------
predictions = model.predict(x_test)
print("predictions on new data (test_data): ", predictions)

# write predictions to text file 'Results_Model.txt'
with open(path_to_Results, 'w+') as openresults:
    for line in predictions:
        np.savetxt(openresults, line, fmt='%.2f')
```

Bijlage C SCRIPT IN PYTHON VOOR REGRESSIE VAN DATA.

```python
import os
import sys
import xlrd
import keras
import numpy as np
import matplotlib.pyplot as plt
from keras import models
from keras import layers
from keras.utils import to_categorical
from keras import optimizers
from keras.utils.np_utils import to_categorical

#-------------------Collect data-------------------
# Directory to text files containing train and test data + labels
path_to_train = "/home/pi/Desktop/Train_Data_Regressie/Regressie13"
path_to_test = "/home/pi/Desktop/Test_Data_Regressie/Regressie13"

# Create some empty lists to store our train_data, train_label, test_data, test_label
train_data = []
train_target = []

test_data = []
test_target = []

# Import data from text files and store them in the lists defined above

with open(path_to_train) as opentrainfile:
    for line in opentrainfile:
        line = line.split()

        train_target_line = []
        target_train = line[0].replace(",", ".")
        train_target_line.append(target_train)
        train_target.append(train_target_line)

        empty_list =[]
        for i in range(1, len(line)):
            Value = line[i].replace(",", ".")
            empty_list.append(Value)
        train_data.append(empty_list)

with open(path_to_test) as opentestfile:
    for line in opentestfile:
        line = line.split()

        test_target_line = []
        target_test = line[0].replace(",", ".")
        test_target_line.append(target_test)
        test_target.append(test_target_line)

        empty_list =[]
        for i in range(1, len(line)):
            Value = line[i].replace(",", ".")
            empty_list.append(Value)
        test_data.append(empty_list)

#-------------------Prepare data-------------------

train_data = np.array(train_data).astype('float32')
test_data= np.array(test_data).astype('float32')
train_targets = np.array(train_target).astype('float32')
test_targets = np.array(test_target).astype('float32')

#--------------------scale data-------------------

#mean = train_data.mean(axis=0)
#train_data -= mean
#std = train_data.std(axis=0)
#train_data /= std
#test_data -= mean
#test_data /= std

#MAX = train_data.max(axis=0)
#MIN = train_data.min(axis=0)
#train_data -= MIN
#train_data /= (MAX-MIN)
#test_data -= MIN
#test_data /= (MAX-MIN)

```

```
#------------------Building Network------------------

def build_model():
    model = models.Sequential()
    model.add(layers.Dense(16,activation='relu', input_shape=(39,)))
    model.add(layers.Dense(16, activation='relu'))
    model.add(layers.Dense(1))
    model.compile(optimizer = RMSpropk, loss='mse', metrics=['mae'])
    return model

#------------------k-fold valiation-----------------
k=4
num_val_samples = len(train_data) // k
num_epochs = 50
all_scores = []
all_mae_histories = []
for i in range(k):
    print('processing fold #', i)
    val_data = train_data[i * num_val_samples: (i + 1) * num_val_samples]
    val_targets = train_targets[i * num_val_samples: (i + 1) * num_val_samples]

    partial_train_data = np.concatenate([train_data[:i * num_val_samples],train_data[(i + 1) * num_val_samples:]],axis=0)
    partial_train_targets = np.concatenate([train_targets[:i * num_val_samples],train_targets[(i + 1) * num_val_samples:]],axis=0)

    model = build_model()
    history=model.fit(partial_train_data, partial_train_targets,validation_data=(val_data, val_targets),epochs=num_epochs, batch_size=1, verbose=0)
    mae_history = history.history['val_mae']
    all_mae_histories.append(mae_history)
    val_mse, val_mae = model.evaluate(val_data, val_targets, verbose=0)
    all_scores.append(val_mae)

print("all_scores:",all_scores)
print("maen:",np.mean(all_scores))

average_mae_history = [np.mean([x[i] for x in all_mae_histories]) for i in range(num_epochs)]

plt.plot(range(1, len(average_mae_history) + 1), average_mae_history)
plt.xlabel('Epochs')
plt.ylabel('Validation MAE')
plt.show()

def smooth_curve(points, factor=0.9):
    smoothed_points = []
    for point in points:
        if smoothed_points:
            previous = smoothed_points[-1]
            smoothed_points.append(previous * factor + point * (1 - factor))
        else:
            smoothed_points.append(point)
    return smoothed_points

smooth_mae_history = smooth_curve(average_mae_history[10:])

plt.plot(range(1, len(smooth_mae_history) + 1), smooth_mae_history)
plt.xlabel('Epochs')
plt.ylabel('Validation MAE')
plt.show()

#------------------Generate predictions on new data------------------

test_mse_score, test_mae_score = model.evaluate(test_data, test_targets)
print("results",test_mae_score)

predictions = model.predict(test_data)
print("predictions on new data (test_data): ", predictions)

# write predictions to text file 'Results_Model.txt'

with open(path_to_Results, 'w+') as openresults:
    np.savetxt(openresults,predictions,fmt="%.2f")
```

FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN
TECHNOLOGIECAMPUS GENT
Gebroeders De Smetstraat 1
9000 GENT, België
tel. + 32 92 65 86 10
iiw.gent@kuleuven.be