Faculteit Bio-ingenieurswetenschappen

Academiejaar 2010-2011

Automatisch determineren van bomen op basis van beeldmateriaal met behulp van machine learning technieken

Melanka Brackx
Promotor: prof. dr. Bernard De Baets
Tutor: ir. Jan Verwaeren

Masterproef voorgedragen tot het behalen van de graad van
Master in de bio-ingenieurswetenschappen: land- en waterbeheer

Gent, juni 2011

De promotor	De begeleider	De auteur
Prof. dr. Bernard De Baets	Ir. Jan Verwaeren	Melanka Brackx

Woord vooraf

Een herbarium van (on)kruiden in eerste bachelor, een herbarium van bomen in derde bachelor, ... Talrijke uren hebben we voor de opleiding tot bio-ingenieur gespendeerd aan het determineren van planten. Mijn enthousiasme was dan ook groot toen ik het thesisvoorstel zag over geautomatiseerde determinatie van planten. Na een jaar van hard werken volgde daaruit deze afgewerkte masterproef!

In dit woord vooraf wil ik een aantal mensen bedanken. Mijn promotor prof. dr. Bernard De Baets wil ik bedanken om mij de kans te geven aan deze masterproef te werken en om mij op te volgen tijdens het jaar. Mijn begeleider ir. Jan Verwaeren ben ik bijzondere dank verschuldigd voor de duidelijke richtlijnen en al de uitleg. Dankzij de goede begeleiding heb ik veel bijgeleerd en weinig problemen gehad. Nog een persoon die ik wil bedanken is dr. Willem Waegeman voor de cursus *machine learning*. Aangezien *machine learning* een belangrijk deel van mijn thesis uitmaakt heb ik daar veel aan gehad. Tot slot wil ik nog mijn medestudenten bedanken voor de steun en de leuke momenten.

Melanka Brackx
Gent, Juni 2011

Samenvatting

Deze thesis beoogt de ontwikkeling van een algoritme dat in staat is bomen automatisch te determineren op basis van foto's van de bladeren, waarbij gebruik gemaakt wordt van machine learning technieken. Omdat bladeren binnen boomsoorten variatie vertonen, is het aangewezen bij de classificatie meerdere bladeren als prototype voor elke soort te gebruiken. Er werd een dataset aangelegd bestaande uit 1250 bladeren (kleurenscans) van 41 soorten die in Vlaanderen courant zijn. Ten behoeve van de classificatie werden uit de beelden features of bladkenmerken geëxtraheerd. Alle features werden bepaald op basis van het silhouet van het blad (kleur en textuur worden niet benut). Er werd geopteerd voor 98 uiteenlopende features om alle aspecten van het silhouet samen te vatten. De gebruikte features zijn geometrische features van bladschijf en steel, statistische features, histogrammen van de kromming van de contour, effeningscoëfficiënten en contourbeschrijving op basis van Fouriertransformaties. Bepaalde features vertoonden echter sterke onderlinge correlaties. Eveneens werden de beelden paarsgewijs vergeleken a.d.h.v. een afstandsmaat: Baddeley's Δ-metriek. Voorafgaand aan de feature-extractie werden de beelden verwerkt en gesegmenteerd.

Voor de classificatie werden volgende classificatiemethoden getest: (kernel) logistische regressie, (kernel) support vector machines, random forests en boosting. Als beste methoden werden random forests en support vector machines bevonden met een accuraatheid van 90% (10-voudige kruisvalidatie). Deze resultaten zijn competitief tegenover deze van gelijkaardige studies uit de literatuur. Het random forest model werd verwerkt in een standalone applicatie met GUI op basis van matlab (MCR). Daarin kan de gebruiker foto's inladen en laten determineren.

Er werden een aantal pogingen ondernomen om de accuraatheid nog te verhogen door toevoegen van kennis over hierarchie (hiërarchische classificatie) en morfologie (multilabelclassificatie). Dit bleek echter geen meerwaarde te bieden tegenover de eenvoudige modellen. Tevens werd getracht een toets uit te voeren om na te gaan of instanties niet tot een soort behoren die voor het algoritme onbekend is. Daartoe wordt de minimale Euclidische afstand van de onbekende instantie tot de instanties uit het dataset bepaald en getest tegen een bepaalde drempelwaarde.

Ten slotte kon via een aantal tests worden aangetoond dat het foutenpercentage een lineaire trend vertoont in functie van het aantal soorten: een halvering van het aantal soorten komt overeen met een halvering van het foutenpercentage. Uit analyse van de performantie per soort bleek dat de fouten zich voornamelijk binnen een aantal probleemgroepen situeren terwijl andere soorten met 100% accuraatheid worden voorspeld. Daarnaast werd gevonden dat voor de trainingsdataset een twintigtal instanties per soort volstaat.

Lijst van afkortingen

CCDC	center-contour distance curve
DFT	discrete Fouriertransformatie
EFT	elliptische Fouriertransformatie
FFT	fast Fourier transform
FN	false negative
FP	false positive
GUI	graphical user interface
HOG	histogram of oriented gradients
k-NN	k-nearest-neighbor
KLR	kernel logistische regressie
LD	lineaire discriminant
LDA	lineaire discriminantanalyse
LoG	Laplacian of Gaussian
LR	logistische regressie
MCR	Matlab Compiler Runtime
OOB	out-of-bag
PC	principale component
PCA	principale componenten analyse
PPI	pixels per inch
RBF	radiale basis functie
RF	random forest
RGB	rode, groene en blauwe band van kleurenfoto
ROI	region of interest
SVM	support vector machine

Voor de verklaring van symbolen wordt verwezen naar de notatiesectie op p. 4.
Afkortingen van soortnamen zijn terug te vinden in Tabel 4.1 op p. 32.

Inhoudsopgave

II Praktische uitwerking van de boomdeterminatie

Lijst van figuren

Lijst van tabellen

Hoofdstuk 1

Inleiding

1.1 Situering en motivatie

Het correct kunnen identificeren en classificeren van bomen en planten wordt vaak beschouwd als een belangrijke competentie van een bioloog. Omwille van de immense diversiteit die aanwezig is in het plantenrijk, kunnen zelfs de meest doorwinterde botanisten slechts een minieme fractie van het totale aanbod aan soorten correct identificeren. Deze beperkingen bemoeilijken de ontdekking van nieuwe soorten en het in kaart brengen van bestaande soorten. Voornamelijk bij zeldzame en bedreigde soorten manifesteert dit probleem zich. Volgens Reeb (1997) zijn er wereldwijd bijna 10000 boomsoorten geïdentificeerd. Door de achteruitgang van het milieu zou ongeveer 22-47% van alle planten met uitsterven bedreigd zijn (Du *et al.*, 2007 en Wu *et al.*, 2007b). Een vlotte determinatietechniek kan bijdragen tot de efficiënte conservering van deze soorten. Een mogelijke oplossing bestaat erin bomen trachten te herkennen op basis van foto's (voornamelijk van de bladeren). Het beeldmateriaal wordt dan gebruikt als input voor een algoritme dat de plant identificeert. In de literatuur zijn verschillende *machine learning* technieken beschikbaar die kunnen gebruikt worden om voorwerpen te herkennen op basis van beeldmateriaal. Het onderzoek omtrent het determineren van planten met dergelijke technieken staat echter nog in zijn kinderschoenen. Het doel van deze thesis is een automatische herkenning van bomen uit te voeren aan de hand van beelden van bladeren.

Een dergelijk boomclassificatiesysteem zou nuttig zijn in het kader van biodiversiteitsonderzoek, maar er zijn nog toepassingen mogelijk. Een voorbeeld daarvan is het informeren van natuurliefhebbers tijdens excursies, wat zou kunnen door de software te installeren op een *smartphone* met fotocamera. Op die manier kan de gebruiker ter plaatse en in *real time* de boomsoort laten bepalen en kan deze eventueel extra informatie opvragen over de boom. Dergelijke software kan eveneens interessant zijn voor bosbeheerders of personen die interesse hebben voor de bomen in hun omgeving. Het systeem zou ook als educatief hulpmiddel kunnen dienen in het (lager) onderwijs.

Momenteel gebeurt het determineren van bomen doorgaans nog met behulp van een plantengids of flora waarbij de gebruiker een sleutel volgt (van der Meijden, 2005). Determinatiesleutels zijn praktisch indien men een plant ter plaatse wil determineren, maar het risico op fouten is groot en het determineren neemt veel tijd in beslag. Alternatieve plantclassificatiemethoden zijn gebaseerd op moleculaire technieken (Wu *et al.*, 2007b). Deze bevinden zich evenwel ook nog in de ontwikkelingsfase. Brunner *et al.* (2001) en Duminil *et al.* (2010) maken gebruik van chloroplast-DNA om soorten te identificeren. De moeilijkheid in DNA-onderzoek is om de juiste markers te vinden. Zij slaagden er wel al in om soorten minstens tot op het niveau van genus te classificeren. Door de genetica van bomen te bestuderen, kunnen eveneens verwantschappen tussen soorten onderzocht worden. Minpunten van de technieken zijn dat

ze in een laboratorium dienen te worden uitgevoerd en relatief duur zijn. Bovendien is een real-time classificatie hiermee niet mogelijk.

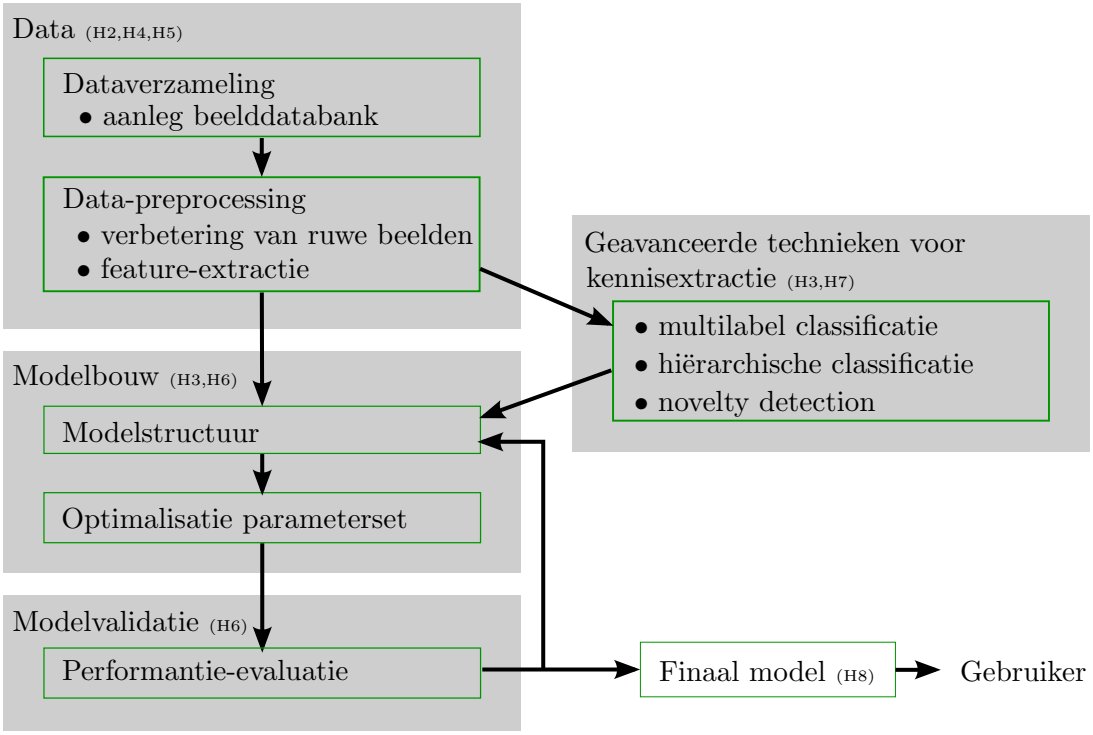
Een eerste poging om plantdeterminatie te automatiseren dateert reeds uit 1980. Dallwitz (1980) ontwierp een systeem om de beschrijvingen die gebruikt worden door botanisten te coderen. Verder bestaan er digitale plantengidsen zoal Heukels' Interactive Flora van Nederland van ETI BioInformatics (www.soortenbank.nl), die ook beschikbaar is als iPhone applicatie. Deze systemen vergen echter nog steeds veel moeite van de gebruiker, er is dus verdere automatisering mogelijk. Automatische classificatie van bomen is een onderwerp dat zich situeert in verschillende onderzoeksdomeinen, waaronder *computer vision, pattern recognition* en *machine learning*. Het probleem werd reeds door verschillende onderzoekers bestudeerd (bv. Lee en Chen, 2006; Du *et al.*, 2007; Caballero en Aranda, 2010) en er zijn al een aantal (semi-) geautomatiseerde systemen operationeel (White *et al.*, 2006a,b; Belhumeur *et al.*, 2008; Knight *et al.*, 2010). Zeer recent (in mei 2011) werd een applicatie uitgebracht voor iPhones die in staat is boomsoorten uit New York City en Washington D.C te herkennen (Belhumeur *et al.*, 2011).

Voorgaand werk kan echter nog verbeterd worden aangezien de meeste systemen nog steeds menselijke interactie vragen bij het extraheren van *features* en slechts werken op een beperkt aantal soorten. De applicatie van Knight *et al.* (2010) herkent bijvoorbeeld slechts 5 soorten. Sommige onderzoeken hebben zich bewust toegespitst op een beperkt aantal species, bijvoorbeeld *Castanea* species (Gouveia *et al.*, 2002) of *Quercus* species (Viscosi *et al.*, 2009). In de applicatie van (Belhumeur *et al.*, 2011) worden wel 191 soorten gebruikt en zijn er plannen voor uitbreiding naar alle 750+ soorten uit Noord-Amerika. Er is ook nog veel onduidelijkheid over welke blad-eigenschappen een goede voorspellende kwaliteit hebben (Wu *et al.*, 2007b).

1.2 Opbouw van de scriptie

Deze scriptie bestaat uit twee delen, in het eerste deel 'Literatuurstudie en theoretische aspecten' wordt nagegaan welke technieken in de literatuur gebruikt worden en worden ook een aantal methoden uit de machine learning toegelicht. In het tweede deel 'Praktische uitwerking van de boomdeterminatie' worden de methoden toegepast en de resultaten besproken.

De ontwikkeling van een boomclassificatiesysteem kan opgesplitst worden in verschillende deeltaken, de componenten die aan bod komen in de masterproef zijn weergegeven in Figuur 1.1. In het eerste blok, 'Data', wordt een eigen dataset met gescande bladeren van bomen uit Vlaanderen aangelegd en worden de beelden gepreprocesst. De preprocessingsfase bestaat uit enkele ingrepen die de interpreteerbaarheid van de ruwe beelden verhogen (*image enhancement*) gevolgd door een uitgebreide extractie van *features*. Extractie van features of bladkarakteristieken uit het beeldmateriaal is essentieel aangezien het niet mogelijk is om bladeren direct te classificeren op basis van intensiteitswaarden van de beelden. Pixelintensiteiten kunnen namelijk ook binnen éénzelfde soort zeer variabel zijn. Een betere benadering is dan het gebruik van features. Features zijn kenmerken of trekken die het blad karakteriseren en worden in getalwaarden uitgedrukt. Ze kunnen dus gebruikt worden als variabelen in een classificatiemodel. In het blok 'Modelbouw' worden modellen gefit aan de data. In *machine learning* zijn verschillende modellen beschikbaar voor classificatievraagstukken en enkele daarvan zullen uitgeprobeerd worden. Het model kan uitgebreid worden door toepassing van een aantal 'Geavanceerde technieken voor kennisextractie'. Er zal geprobeerd worden het model te verbeteren door middel van hiërarchische classificatie en multilabel classificatie (uitleg volgt later) en er zal een uitbreiding voorzien worden om nieuwe, ongekende soorten te detecteren. In het blok 'Modelvalidatie' worden de modellen getest en geëvalueerd. Hierbij



Figuur 1.1: Flow-chart met aanduiding van de hoofdstukken waarin de verschillende componenten aan bod komen

zal een kruisvalidatiestrategie toegepast worden om betrouwbare resultaten te bekomen en de data toch optimaal te benutten. Ten slotte wordt het beste model als finaal model geselecteerd en uitgebreid geëvalueerd. Dit model werd eveneens verwerkt tot een GUI (Bijlage A).

1.3 Doelstellingen

Het hoofddoel van deze thesis bestaat erin een geautomatiseerd boomdeterminatiesysteem te ontwikkelen dat in staat is een boom te determineren tot op soortniveau op basis van fotomateriaal van de bladeren. Meer specifiek zal getracht worden een antwoord te formuleren op onderstaande onderzoeksvragen.

Vraag 1: Is het mogelijk bomen te classificeren op basis van beeldmateriaal van de bladeren?
Bij het klassiek determineren van bomen met behulp van een flora worden behalve de bladvorm nog verschillende aspecten van de plant gebruikt: vruchten, bloeiwijze, schors, geur, etc. Het model dat opgesteld wordt zal enkel gebruik maken van informatie afkomstig uit een beeld van het blad. Er zal dus onderzocht worden in hoeverre bladeren karakteristiek zijn voor een boomsoort en of het dus mogelijk is soorten te herkennen zonder extra informatie over de plant.

Vraag 2: Welke eigenschappen zijn geschikt om bladeren te beschrijven en soorten van elkaar te onderscheiden?
Uit de literatuurstudie zal blijken dat er een heel scala aan features ter beschikking staat om beelden te karakteriseren. Wegens de hoge onderlinge correlaties is het niet nuttig deze allemaal te gebruiken. Er zal daarom gezocht worden naar een set van features dat de ver-

schillende kenmerken van de bladeren voldoende omvat. De voorspellende kwaliteit en de onderlinge correlaties tussen de geselecteerde features zullen onderzocht worden.

Vraag 3: Kunnen de features voldoende nauwkeurig bepaald worden zonder tussenkomst van een persoon?
Er wordt getracht een algoritme te ontwikkelen waarin geen tussenkomst van de gebruiker vereist is. Voor het extraheren van de features zal de computer bepaalde structuren in het beeld moeten herkennen: onder meer de steel, de oriëntatie van het blad en de bladrand zullen een rol spelen. Voor de mens is dit een eenvoudige taak, maar voor een computer is dit minder evident.

Vraag 4: Wat is de performantie van een dergelijk classificatiesysteem?
Er zal nagegaan worden welke accuraatheden behaald kunnen worden. Daarnaast zal de performantie voor elke boomsoort geëvalueerd worden.

Vraag 5: Kan de performantie verbeterd worden door extra kennis toe te voegen aan het model?
Twee pistes die zullen gevolgd worden om de performantie te proberen verbeteren zijn hiërarchische classificatie en multilabel classificatie. Bij hiërarchische classificatie worden verschillende groepen uit de taxonomische indeling van het plantenrijk voorspeld door het model, dit kan interessant zijn voor de gebruiker en mogelijks kan dit ook de performantie verbeteren. De hiërarchie kan gezien worden als een vorm van expertkennis die in het model verwerkt wordt. Bij multilabel classificatie zullen in eerste instantie een aantal kenmerkende eigenschappen van het blad voorspeld worden, bijvoorbeeld samengesteld/enkelvoudig en gelobd/gaaf. Deze kenmerken zullen dan als bijkomende input gebruikt worden bij het voorspellen van de soort.

Vraag 6: Is het mogelijk het systeem aan te passen om nieuwe soorten als dusdanig te classificeren?
Het aantal soorten in de databank is beperkt, waardoor het mogelijk is dat de gebruiker een blad probeert te identificeren dat door het model niet gekend is. Bij een classificatieprobleem worden alle data doorgaans ingedeeld in één van de gekende klassen. Het blad wordt dus met zekerheid foutief geclassificeerd. Recent zijn er technieken beschikbaar die dergelijke afwijkende data herkennen.

1.4 Basisterminologie en notatie

In deze sectie worden een aantal notaties en manieren om een blad voor te stellen geïntroduceerd. Gezien het wiskundige karakter van dit werk werd geopteerd voor een afzonderlijke sectie waarin de notatie die gebruikt zal worden formeel wordt geïntroduceerd om zodoende de leesbaarheid te verhogen. In het verdere verloop van de thesis wordt dan ook meermaals verwezen naar deze sectie.

1.4.1 Wiskundige formulering van het classificatieprobleem

Een belangrijke component van deze thesis gaat over het ontwikkelen en toepassen van datagedreven classificatiealgoritmen. Daarom wordt vooreerst het classificatieprobleem formeel gedefiniëerd. Beschouw hiertoe een verzameling objecten $O = \{o_1, ..., o_N\} \subset \mathbb{O}$. Elk van deze objecten kan voorgesteld worden aan de hand van een element $\mathbf{x}$ uit de voorstellings- of feature-ruimte $\mathbb{X}$. Deze objecten kunnen tevens onderverdeeld worden in verschillende groepen of klassen, zodat elk object O een klassenlabel y uit de labelruimte $\mathbb{Y}$ krijgt toegewezen. Het classificatieprobleem bestaat dan uit het zoeken naar een functie $f : \mathbb{X} \to \mathbb{Y}$ die toelaat

het label van een object te voorspellen op basis van de feature-representatie van dit object. Voorspellingen worden aangeduid met een hoedje, $\hat{y}$ staat zo bijvoorbeeld voor het voorspelde klassenlabel.

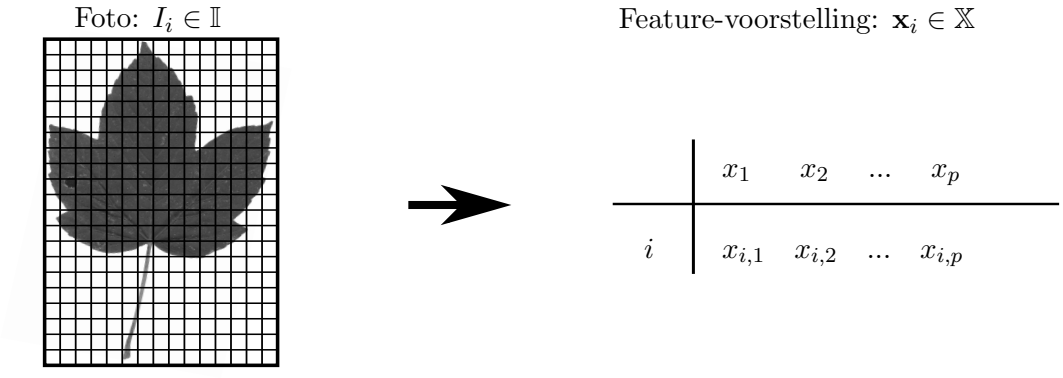
Daar deze thesis tot doel heeft bomen te determineren op basis van foto's van de bladeren, zal gewerkt worden met een databank bestaande uit digitale beelden. Deze foto's zijn rasterbeelden waarvan de resolutie wordt uitgedrukt in pixels per inch (PPI). De rasterbeelden zullen aangewend worden onder de vorm van een intensiteitsmatrix. Een digitaal beeld bestaande uit r rijen en s kolommen wordt voorgesteld als een $r \times s$ matrix $I \in \mathbb{I} = \{0, 1, ..., 255\}^{r \times s}$, waarbij de intensiteit van de pixel op de k-de rij en op de l-de kolom voorgesteld wordt door $I(k, l)$. Dezelfde rij- en kolomnummers kunnen gebruikt worden als coördinaten om de positie van een pixel aan te geven in een cartesisch assenstelsel.

Uit het volgende hoofdstuk zal blijken dat meestal niet rechtstreeks gewerkt wordt met de intensiteitsmatrix. In plaats daarvan wordt uit elk beeld een reeks features geëxtraheerd. De matrixvoorstelling kan bijgevolg getransformeerd worden in een vectoriële voorstelling. Beschouw daartoe een dataset bestaande uit N beelden. Elk beeld I_i, $i = 1, ..., N$, leidt dan tot een feature-vector $\mathbf{x}_i = (x_{i,1}, ..., x_{i,P})^T \in \mathbb{X}$ (waarbij $\mathbb{X} = \mathbb{R}^P$), met P het aantal features en $\mathbb{X}$ de feature-ruimte. Deze transformatie wordt schematisch voorgesteld in Figuur 1.2. Daarnaast wordt met elk beeld een klassenlabel $y_i \in \{c_1, c_2, ..., c_Q\} = \mathbb{Y}$ geassocieerd dat uitdrukt tot welke boomsoort I_i behoort, hierbij is Q het aantal soorten aanwezig in de dataset. Merk op dat het determeninatieprobleem dus kan gezien worden als een typisch classificatieprobleem. De feature-dataset kan voorgesteld worden als $T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_N, y_N)\}$.

1.4.2 Binair beeld van het blad

In de binaire weergave van een blad hebben alle pixels die tot het blad behoren waarde 1, deze die tot de achtergrond behoren hebben waarde 0. Het binaire beeld $B(k, l)$ is een $r \times s$ matrix $B_i \in \mathbb{B} = \{0, 1\}^{r \times s}$ en wordt als volgt afgeleid uit de intensiteitsmatrix I:

$$B(k,l) = \begin{cases} 1 & , I(k,l) \in \text{blad}, \\ 0 & , I(k,l) \notin \text{blad}. \end{cases}$$



Figuur 1.2: Overgang van het beeld uitgedrukt als matrix naar een reeks features uitgedrukt als vector.

1.4.3 Bladcontour

Bij de berekening van een aantal features zal met de contour gewerkt worden. De contour wordt bepaald door de punten van het binaire beeld die tot het blad behoren (waarde 1) en die grenzen aan minstens 1 punt dat niet tot het blad behoort (waarde 0). Men kan kiezen om punten die schuin grenzen aan een pixel met waarde 0 wel of niet bij de contour te rekenen door gebruikt te maken van 4- of 8-connectiviteit (Figuur 1.3).

(a) 4-connectiviteit (b) 8-connectiviteit

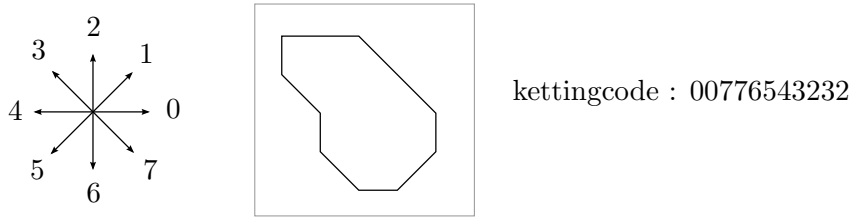
Figuur 1.3: Ligging van de 4- en 8-geconnecteerde buurcellen.

De meest voor de hand liggende manier om de contour weer te geven is als een binair beeld (met hetzelfde formaat als het initiële beeld) waarin alle contourpixels waarde 1 hebben. De contour kan echter ook uitgedrukt worden als een reeks van paren rij- en kolomcoördinaten (k, l) door de contour in wijzerzin (of tegenwijzerzin) te doorlopen. Een algoritme dat voor een gegeven binair beeld een dergelijke reeks genereert wordt beschreven in Burger en Burge (2008). De contour wordt dan voorgesteld door de $K \times 2$ matrix C:

$$C = \begin{bmatrix} k_1 & l_1 \\ k_2 & l_2 \\ \vdots & \vdots \\ k_K & l_K \end{bmatrix},$$

met K het aantal pixels dat tot de contour behoort.

Een nog compactere manier om de contour uit te drukken is de kettingcode (*chain code*), het principe van de kettingcode is weergegeven in Figuur 1.4, het contour wordt beschreven als de sequentie van richtingen waarin de pixels aan elkaar grenzen (Burger en Burge, 2008). De kettingcode kan voorgesteld worden als $\mathbf{v} = (a_1, a_2, a_3, ..., a_{K+1})$ waarbij a_i een getal van 0 tot 7 is voor elke stap $i \in \{1, ..., K + 1\}$.



(a) (b) (c)

Figuur 1.4: Kettingcode notatie voor contouren: (a) Codes voor de 8 richtingen, (b) lijnvoorstelling van een contour en (c) kettingcode voor de lijnen (figuur gebaseerd op Neto *et al.* (2006)).

Deel I

Literatuurstudie en theoretische aspecten

Hoofdstuk 2

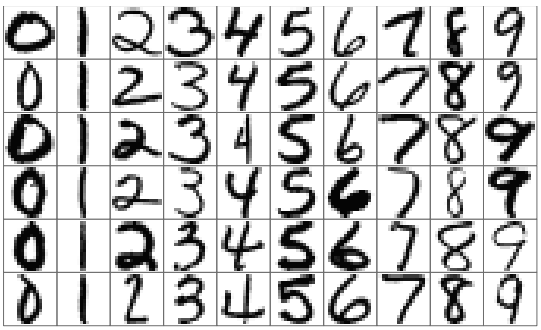
Feature-extractie

In dit eerste deel van de literatuurstudie wordt de extractie van features, en de preprocessing van de ruwe beelden die daarvoor noodzakelijk is, besproken. Er zal onderzocht worden hoe dit in de literatuur gebeurt, in de context van bladherkenning, maar ook in andere toepassingen.

2.1 Waarom feature-extractie?

In essentie is het mogelijk de beelden te classificeren op basis van de intensiteitsmatrix. Het beeld I_i dat bestaat uit $r \times s$ pixels met elk een bepaalde intensiteit wordt dan omgezet naar een vector $\mathbf{x}_i$ van $r \cdot s$ intensiteiten. In Hastie *et al.* (2008) wordt dit geïllustreerd aan de hand van de U.S. postal dataset, waarbij men herkenning van handgeschreven cijfers uitvoert. In Figuur 2.1 is ter illustratie een deel van de U.S. postal dataset weergegeven. Dit is een klassiek classificatieprobleem aangezien men op zoek gaat naar een functie $f : \mathbb{X} \to \mathbb{Y}$ waarbij $\mathbb{Y} = \{0, 1, ..., 9\}$. Wanneer de grootte en de hoek van de cijfers gestandaardiseerd wordt, kan op basis van de intensiteiten met een neuraal netwerk nog een hoge accuraatheid behaald worden.

Wanneer men echter grotere beelden (met meer pixels) wil classificeren, zullen modellen die de intensiteitswaarden als input gebruiken belangrijke patronen in de data niet ontdekken. Voor deze modellen staan de pixels los van elkaar, er wordt dus geen rekening gehouden met het ruimtelijke karakter van een foto. Er zou bijgevolg een zeer grote dataset nodig zijn om een aanvaardbare performantie te halen (Hastie *et al.*, 2008). In het geval van bladeren is het zeer waarschijnlijk dat de meeste informatie over de aard van de bladrand en de bladvorm ongebruikt zal blijven. Een ander probleem dat zich stelt bij de classificatie van bladeren is de grote variatie in bladvorm binnen eenzelfde soort. Ook na normalisatie van grootte en oriëntatie kunnen grote verschillen voorkomen. Daarnaast is het normaliseren van grootte en



Figuur 2.1: Voorbeelden uit de U.S. postal databank. Elk beeld is een 16 x 16 8-bit grijswaarden voorstelling van een handgeschreven cijfer (Hastie *et al.*, 2008).

oriëntatie al een vraagstuk op zich. Het directe gebruik van de sequentie van pixelwaarden voor classificatie is in het geval van bladherkenning dus niet aangewezen.

Het extraheren van features uit de beelden is een frequent toegepast alternatief (Sinha, 2004 en Lee en Chen, 2006). Features zijn kenmerken of trekken die het blad karakteriseren, ze drukken de vorm- en andere eigenschappen van het blad uit in getalwaarden. Aldus wordt een nieuwe feature-dataset bekomen die gebruikt wordt voor de classificatie.

Om tot een goede classificatie te komen dienen de features te voldoen aan een aantal criteria (Wang *et al.*, 2003 en Du *et al.*, 2007):

- de feature moet voldoende variabiliteit te vertonen (i.e. niet constant zijn),
- de variabiliteit dient gecorreleerd zijn met de klassen, indien dit niet het geval is gaat het om niet-gecorreleerde ruis,
- translatie- en rotatie-invariantie: de positie en oriëntatie van het blad in het beeld zijn niet relevant en mogen geen invloed hebben op de waarde van de feature,
- schaalinvariantie: de grootte van het beeld mag geen invloed hebben op de waarde van de feature. Echter kan het, omdat de grootte op zich een mogelijke feature is (van der Meijden, 2005), bij bladherkenning wel nuttig zijn om de grootte van de bladeren in rekening te brengen. Voor dergelijke features is dan wel resolutie-invariantie vereist.

Merk op dat de laatste twee criteria omzeild kunnen worden door het standaardiseren van positie, oriëntatie en grootte (of resolutie) van het blad.

2.2 Digitaliseren van bladeren

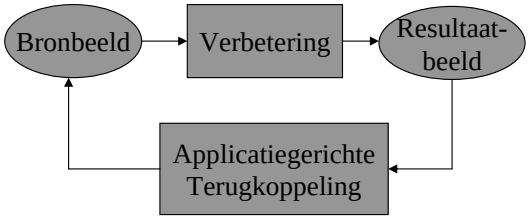
De wijze waarop de bladeren gedigitaliseerd worden zal een invloed hebben op de feature-extractie. Scherpe beelden zijn noodzakelijk voor een nauwkeurige berekening van de features. Om de noodzaak aan complexe *computer vision* technieken te vermijden kan bij het scannen of fotograferen een egale achtergrond gebruikt worden.

In de literatuur worden verschillende technieken gebruikt om de bladeren te digitaliseren. Veelal worden de bladeren gedigitaliseerd met een egale achtergrond (Lee en Chen, 2006 en Du *et al.*, 2007). De gebruikte opstellingen gaan van zeer eenvoudig – foto's van het blad op een wit vel papier (bv. Karrels, 2006)– tot meer professioneel, zo fotografeerde Lee en Chen (2006) bijvoorbeeld de bladeren op een lichtbak. Andere onderzoeken hebben zich meer toegespitst op het extraheren van bladeren uit complexe achtergrond of bij overlappende bladeren (Du *et al.*, 2006 en Wang *et al.*, 2008).

2.3 Beeldverbetering

In deze sectie worden een aantal methoden beschreven die gebruikt worden om de ruwe beelden te verbeteren. Om een bevredigende beeldverbetering te bekomen is het doorgaans nodig om te experimenteren met methoden en parameters. Beeldverbetering kan daardoor gezien worden als een proces met terugkoppeling zoals geïllustreerd in Figuur 2.2. De verschillende doelen van beeldverbetering kunnen als volgt samengevat worden (Baeten, 2006):

- ongewenste ruis en distorties onderdrukken,
- kenmerken benadrukken of verscherpen voor weergave of analyse,
- probleemgerichte operaties.



Figuur 2.2: Schematische weergave van het principe van beeldverbetering (Baeten, 2006).

Bij beeldverbetering wordt veelvuldig gebruik gemaakt van filters (Sinha, 2004), daarom wordt eerst een algemene introductie van filters gegeven. Daarna volgt een uiteenzetting van beeldverbetering in het kader van bladherkenning.

2.3.1 Gebruik van filters in beeldverwerking

In de beeldverwerking zijn twee klassen van filters van belang: ruimtelijke filters (bv. afvlakfilter of uitmiddelen) en frequentiedomeinfilters (bv. Fouriertransformatie) (Baeten, 2006).

Een ruimtelijke filter kan gezien worden als een functie $f : \mathbb{I} \to \mathbb{I}'$ die het beeld I omzet naar het beeld I' door in elke pixel een bepaalde operatie uit te voeren. Indien de resultaatwaarde voor elke pixel enkel afhankelijk is van de waarde op gelijke positie in het originele beeld spreekt men van een puntoperatie: $I'(k,l) = f(I(k,l))$. Enkele voorbeelden van puntoperaties zijn inversies, contrastaanpassingen, kleurenfilters en drempels. Er zijn echter ook filters waarin de waarde in een pixel (mede) bepaald wordt door de waarde van de omliggende pixels in het initiële beeld, dergelijke operaties zijn maskeroperaties. $I'(k,l)$ is dan $f(I(m))$, waarin m een matrix met de rij- en kolom-coördinaten van het masker in functie van k en l voorstelt. Maskeroperaties kunnen lineair (bv. afvlakfilter en laplaciaan) of niet lineair (bv. maximum, mediaan) zijn (Burger en Burge, 2008).

Meer uitleg over filters in het frequentiedomein volgt in Sectie 2.5.2, waar dergelijke methoden gebruikt worden voor feature-extractie.

2.3.2 Conversie van RGB naar grijstinten

De kleur van bladeren is moeilijk te interpreteren aangezien bladeren bijna altijd groen zijn en verkleuringen gedurende het jaar optreden (lichtgroen in de lente, donkerder in de zomer en daarna herfstverkleuringen). De kleur is dus maar gedeeltelijk gerelateerd aan de boomsoort en hangt vooral af van omgevingsfactoren en ander factoren, waaronder ook de gevoeligheid van het toestel waarmee het beeld gemaakt is. De RGB kleuren worden daarom bij de berekening van de features zelden gebruikt. Men werkt doorgaans met enkelvoudige intensiteitsbeelden (Solé-Casals *et al.*, 2009 en Knight *et al.*, 2010). Indien de databank uit kleurenfoto's bestaat, kunnen de RGB waarden worden omgezet naar grijswaarden aan de hand van een puntoperatie. Elke pixelwaarde wordt dan vervangen door een gewogen som van de drie kleurencomponenten:

$$I = 0.2989 * R + 0.5870 * G + 0.1140 * B, \tag{2.1}$$

met $I \in \mathbb{I}$ de intensiteitsmatrix en R, G en B de rode, de groene en de blauwe component van het kleurenbeeld. De omzetting van kleur naar grijswaarden is niet uniek, verschillende combinaties van de drie kanalen kunnen een goed resultaat geven. Bovenstaande formule behoudt de helderheid (*luminance*) van het beeld. Deze formule wordt algemeen gebruikt en is ook standaard in Matlab (Wu *et al.*, 2007b en The MathWorks, Inc., 2010a).

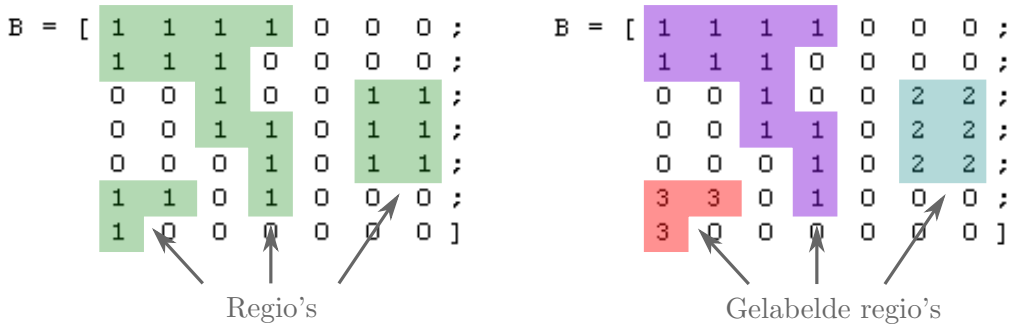
2.3.3 Beeldsegmentatie

In Sectie 2.1 werd gesteld dat het af te raden is om classificatie uit te voeren op basis van de intensiteitsmatrix en dat men beter features berekent. Bijgevolg is het nodig om de *regions of interest* (ROI's) te bepalen op basis waarvan de features berekend worden. Beeldsegmentatie is een klassieke stap in beeldherkenning. In eerste instantie is het nodig om de positie van het blad in de foto te bepalen. Dit is nodig aangezien men niet geïnteresseerd is in de relatieve positie of oriëntatie van het blad op de foto.

Voor deze thesis zal bij het scannen een witte achtergrond gebruikt worden. De literatuurstudie wordt daarom beperkt tot de segmentatie van beelden met een egale achtergrond. De moeilijkheden die zich stellen zijn de aanwezigheid van achtergrondruis door verschillende belichting en het voorkomen van onzuiverheden op de achtergrond. Er zal gewerkt worden met beelden in matrixformaat (rechthoekig) dus elk blad moet uitgesneden worden en op een neutrale achtergrond gezet worden. Bij het uitsnijden van bladeren wordt het intensiteitsbeeld omgezet in een binair beeld waarin de pixels die tot het bladpixel behoren waarde 1 hebben en de achtergrondpixels waarde 0 (Burger en Burge, 2008). Indien men de intensiteitsmatrix verder wil gebruiken kan het binaire beeld als masker gebruikt worden om het blad uit te snijden (Park *et al.*, 2008), maar het binaire beeld zelf kan ook gebruikt worden om features uit te extraheren (Wu *et al.*, 2007b).

In de literatuur wordt het uitsnijden doorgaans gedaan aan de hand van een drempelwaarde (*threshold*) op de grijswaarden (Zhang *et al.*, 2004 en Karrels, 2006). De optimale drempel kan voor elk blad worden afgeleid uit het grijswaarden histogram via Otsu's methode (Otsu, 1979). Doordat een blad een zekere dikte en reliëf heeft komen aan de bladrand en aan de steel geregeld schaduwen voor. Een drempel werkt dan niet correct als er bleke delen in het blad zitten, zoals bijvoorbeeld bij de (witte) abeel en de Amerikaanse eik, waar de steel en de nerven bleker zijn dan deze schaduwen. Dit probleem werd ook beschreven door Karrels (2006), op een dataset dat ook herfstbladeren omvat, werd vastgesteld dat ongeveer 80% van de bladeren foutief uitgesneden werd. Karrels haalde de foutief uitgesneden bladeren handmatig uit de dataset. Een mogelijk alternatief is om naast de drempelwaarde ook een kleurcriterium te gebruiken (Sinha, 2004).

Drempelwaarden en kleurcriteria zijn niet in staat om het blad te onderscheiden van andere (ongewenste) structuren in het beeld. Om het blad van de andere objecten te onderscheiden wordt gebruik gemaakt van *connected-component labeling* (Burger en Burge, 2008). Een *connected component* of regio in een binair beeld is een verzameling van pixels die een samenhangende groep vormen. In een beeld kunnen dus verschillend regio's voorkomen. Het opzet van *connected-component labeling* is om elke regio een bepaalde waarde te geven. Dit principe wordt geïllustreerd in Figuur 2.3. Met betrekking tot bladherkenning zal het blad doorgaans gevonden worden als de regio met de grootste oppervlakte. Dan is de ligging van het blad gekend, maar weet men nog niets over de oriëntatie van het blad. Eerder werd aangehaald dat rotatie-invariantie een belangrijke vereiste is voor goede features en dat deze vereiste omzeild kan worden door het standaardiseren van de oriëntatie van de bladeren. In de literatuur werd echter weinig informatie gevonden over de manier waarop de lengterichting bepaald kan worden daar veelal rotatie-invariate features gebruikt worden (Zhang *et al.*, 2004). Wanneer oriëntatie wel van belang is, worden de bladeren reeds gelijk gepositioneerd bij het fotograferen (Söderkvist, 2001), dienen een aantal referentiepunten (einde en begin van de hoofdnerf) handmatig te worden ingegeven (Wu *et al.*, 2007a) of wordt de gebruiker gevraagd het blad te oriënteren op een touch-screen (Knight *et al.*, 2010).



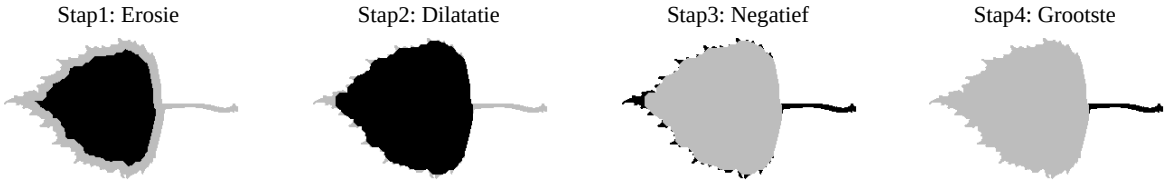
Figuur 2.3: Het principe van *connected-components labeling*. Binair beeld met drie regio's voor labeling (links) en na labeling (rechts).

De volledige bladregio's die bekomen worden door *connected-components labeling* kunnen opgesplitst worden in twee ROI's: de bladsteel en het blad zonder steel (de bladschijf). Het is aangewezen om de bladsteel te verwijderen uit het beeld bij de berekening van een aantal features, ook kan de bladsteel gebruikt worden om de oriëntatie te bepalen en kunnen een aantal kenmerken van de steel zelf berekend worden.

De steel kan gevonden worden door het uitvoeren van een morfologische opening op het binaire beeld (Zhang *et al.*, 2004). Bij morfologische opening worden enkel de zones waar een gekozen structuurelement volledig in kan passen behouden. Het proces is stapsgewijs geïllustreerd in Figuur 2.4. Morfologische opening is het resultaat van een erosie en een dilatatie met hetzelfde structuurelement (Gonzalez en Woods, 2008). Wanneer dit beeld wordt afgetrokken van het originele binaire beeld bekomt men een figuur met enkel de steel en eventueel een aantal tanden van de bladrand. De bladsteel wordt dan bepaald als de grootste regio van aan elkaar grenzende pixels. Door het verwijderen van de steel kan de bladschijf als aparte figuur bewaard worden.

Een andere techniek die kan gebruikt worden om de steel te vinden is *ridge detection* (Staal en Viergever, 2004). *Ridges* zijn lijnen (bestaande uit punten) in het beeld waar de eerste afgeleide van de intensiteit in de richting van de sterkste kromming van teken wisselt. De term *ridge* (Nederlands: nok, kam) komt voort uit het feit dat een intensiteitsbeeld gezien kan worden als een topografisch oppervlak, de *ridges* zijn dan de bergkammen. *Ridge detection* wordt frequent toegepast in medische beeldvorming, onder meer om aders te benadrukken (Staal en Viergever, 2004). Om de *ridges* te vinden kan een *Laplacian of Gaussian* (LoG) filter gebruikt worden. De kernel van deze filter heeft de vorm van een Mexicaanse hoed (Burger en Burge, 2008).

Eens bovenstaande stappen doorlopen zijn, kunnen de features berekend worden. De features die in de literatuur gebruikt worden kunnen ingedeeld worden in drie types op basis van op welk aspect van het blad ze karakteriseren: regio-, contour- en textuurfeatures.



Figuur 2.4: Extractie van de steel met behulp van morfologische opening

2.4 Regiofeatures

In de groep van regiofeatures worden deze ingedeeld die de bladvorm beschrijven. In Figuur 2.5 worden een aantal uitgesproken bladvormen geïllustreerd.

De meest gebruikte features zijn digitale morfologische features zoals aspect ratio, rechthoekigheid, oppervlakte ratio, cirkelvormigheid en excentriciteit. De berekening zal toegelicht worden in Deel II. Deze features zijn echter over het algemeen eenvoudig te berekenen. Digitale morfologische features worden onder meer gebruikt door Lee en Chen (2006), Du *et al.* (2007), Wu *et al.* (2007b) en Caballero en Aranda (2010). In *computer vision*, schriftherkenning en medische beeldvorming worden gelijkaardige methoden gebruikt.

Er kan ook gebruik gemaakt worden van invariante momentenfeatures om vormen te beschrijven (Du *et al.*, 2007 en Wang *et al.*, 2008). Momenten zijn statistische vormeigenschappen. Het blad wordt beschouwd als een verzameling punten die verspreid liggen in een 2-dimensionale ruimte (Burger en Burge, 2008). Voor een binair beeld B worden het (i, j)-de orde moment als volgt berekend:

$$\eta_{ij} = \sum_k \sum_l k^i \; l^j \; B(k, l)$$

Deze η_{ij} kunnen translatie-invariant gemaakt worden door de coördinaten relatief ten opzichte van het zwaartepunt (k_Z, l_Z) van B uit te drukken:

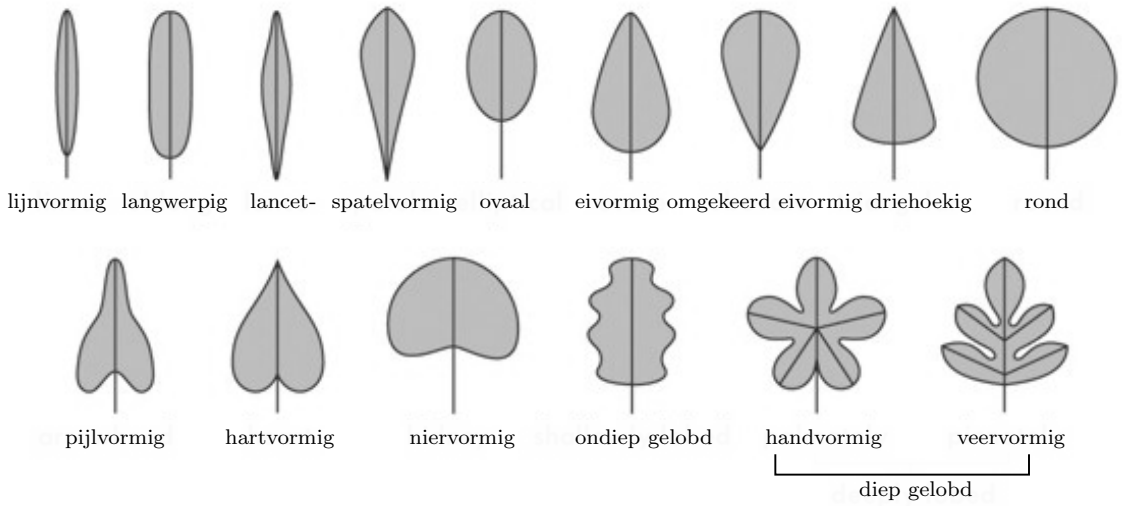
$$\mu_{ij} = \sum_k \sum_l (k - k_Z)^i \; (l - l_Z)^j \; B(k, l)$$

Schaalinvariantie wordt bereikt door volgende normalisatie:

$$\bar{\mu}_{ij} = \frac{\mu_{ij}}{\mu_{00}^{\frac{i+j+2}{2}}}$$

Door een gepaste combinatie van genormaliseerde gecentraliseerde momenten $\bar{\mu}_{ij}$ kunnen rotatie-invariante features berekend worden. Dergelijke features worden *moment invariants* genoemd. De bekendste zijn de 7 momenten van Hu (Hu, 1962):

$$\begin{aligned}
H_1 &= \bar{\mu}_{20} + \bar{\mu}_{02}, \\
H_2 &= (\bar{\mu}_{20} - \bar{\mu}_{02})^2 + 4\bar{\mu}_{11}^2, \\
H_3 &= (\bar{\mu}_{30} - 3\bar{\mu}_{12})^2 + (3\bar{\mu}_{21} - \bar{\mu}_{03})^2, \\
H_4 &= (\bar{\mu}_{30} + \bar{\mu}_{12})^2 + (\bar{\mu}_{21} + \bar{\mu}_{03})^2, \\
H_5 &= (\bar{\mu}_{30} - 3\bar{\mu}_{12})(\bar{\mu}_{30} + \bar{\mu}_{12})[(\bar{\mu}_{30} + \bar{\mu}_{12})^2 - 3(\bar{\mu}_{21} + \bar{\mu}_{03})^2] \\
&\quad + (3\bar{\mu}_{21} - \bar{\mu}_{03})(\bar{\mu}_{21} + \bar{\mu}_{03})[3(\bar{\mu}_{30} + \bar{\mu}_{12})^2 - (\bar{\mu}_{21} + \bar{\mu}_{03})^2], \\
H_6 &= (\bar{\mu}_{20} - \bar{\mu}_{02})[(\bar{\mu}_{30} + \bar{\mu}_{12})^2 - (\bar{\mu}_{21} + \bar{\mu}_{03})^2 + 4\bar{\mu}_{11}(\bar{\mu}_{30} + \bar{\mu}_{12})(\bar{\mu}_{21} + \bar{\mu}_{03})], \\
H_7 &= (3\bar{\mu}_{21} - \bar{\mu}_{03})(\bar{\mu}_{30} + \bar{\mu}_{12})[(\bar{\mu}_{30} + \bar{\mu}_{12})^2 - 3(\bar{\mu}_{21} + \bar{\mu}_{03})^2] \\
&\quad + (\bar{\mu}_{30} - 3\bar{\mu}_{12})(\bar{\mu}_{21} + \bar{\mu}_{03})[3(\bar{\mu}_{30} + \bar{\mu}_{12})^2 - (\bar{\mu}_{21} + \bar{\mu}_{03})^2].
\end{aligned}$$



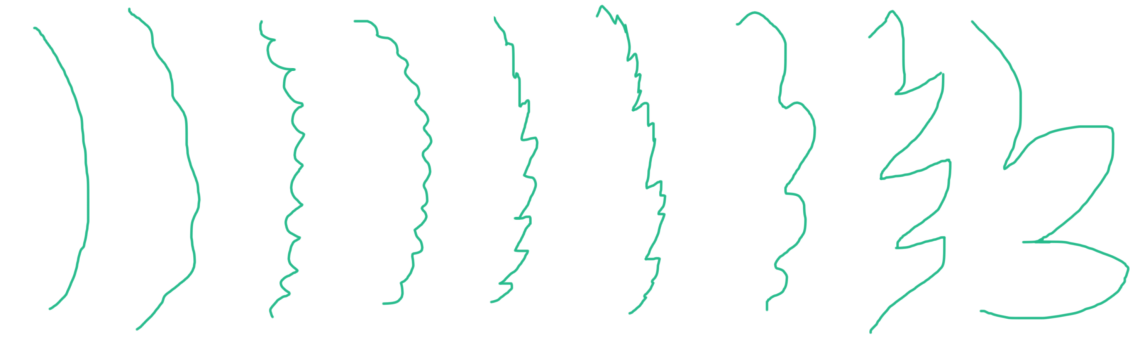
Figuur 2.5: Illustratie van een aantal bladvormen (Dickinson *et al.*, 2004).

Een algemene theorie over de afleiding van rotatie-invariante momenten werd voorgesteld door Flusser en Suk (Flusser, 2000 en Flusser en Suk, 2006). Bovendien voegden zij aan de originele Hu's set nog een achtste element toe:

$$H_8 = \bar{\mu}_{11}[(\bar{\mu}_{30} + \bar{\mu}_{12})^2 - (\bar{\mu}_{03} + \bar{\mu}_{21})^2] - (\bar{\mu}_{20} - \bar{\mu}_{02})(\bar{\mu}_{30} + \bar{\mu}_{12})(\bar{\mu}_{03} + \bar{\mu}_{21}).$$

2.5 Contourfeatures

Er komen in de plantenwereld menige types bladranden voor, in Figuur 2.6 zijn er een aantal weergegeven. Bladranden zijn vaak typerend voor een soort. Gegevens over de bladrand kunnen dus een belangrijke bijdrage leveren aan een correcte classificatie. De contourfeatures die hier besproken worden zullen voor een deel ook informatie bevatten over de algemene bladvorm, maar ze zijn in het bijzonder bedoeld om de bladrand te karakteriseren aangezien deze informatie nog niet vervat zit in de regiofeatures.



Figuur 2.6: Illustratie van een aantal bladranden

2.5.1 Centrum-contourafstandscurve

In de notatiesectie (Sectie 1.4), werden reeds een aantal manieren beschreven waarop de contour kan worden voorgesteld:

- als binair beeld (contour 1, achtergrond 0),
- als de $K \times 2$ matrix C die bestaat uit paren van rij- en kolomcoördinaten (k, l) en
- als kettingcode $\mathbf{v}$, een sequentie van cijfers.

Er worden in de literatuur echter nog verschillende *contour descriptors* voorgesteld, vaak wordt gebruik gemaakt van de *Center-contour distance curve* (CCDC) (o.a. Fu *et al.*, 2005 en Park *et al.*, 2008). Door gebruik te maken van de centrum-contourafstand is het mogelijk om de contour voor te stellen als een één-dimensionale curve: een sequentie met de afstand van het middelpunt tot de contour terwijl deze laatste doorlopen wordt. Als centrum kan het zwaartepunt van het blad gebruikt worden. Dit wordt geïllustreerd in Figuur 2.7.

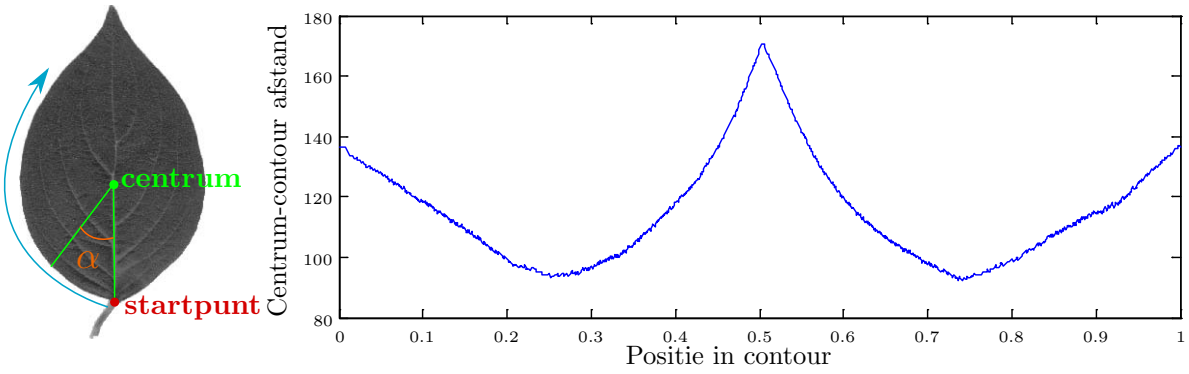
Mits normalisatie van de grootte en vastleggen van een beginpunt is de CCDC invariant voor translatie, schaal en rotatie. Dit kan eenvoudig aangetoond worden (Wang *et al.*, 2003):

- Translatie-invariantie: de curve wordt volledig bepaald door de zwaartepunt-contourafstand, deze is onafhankelijk van de positie in een coördinatenstelsel.
- Schaalinvariantie: Bij grotere bladeren zijn de zwaartepunt-contourafstanden langer, dit kan gecorrigeerd worden door de grootte vooraf te standaardiseren of door de curve te normaliseren met een schaalfactor. Grote bladeren hebben tevens meer contourpixels, dit kan opgelost worden door *down sampling* van het aantal punten in de CCDC.
- Rotatie-invariant: Door rotatie zal de CCD-curve een faseverschuiving ondergaan ter grootte van de hoek waarover geroteerd werd. Dit kan tegengegaan worden door het beginpunt op de curve vast te leggen zodat het als het ware mee roteert. Als beginpunt kan bijvoorbeeld de plaats waar de steel aan het blad zit genomen worden.

Deze eigenschappen zijn noodzakelijk om de curves van verschillende bladeren te kunnen vergelijken. In Wang *et al.* (2003) worden de curves paarsgewijs vergeleken met volgende afstandsmaat:

$$D(I_A, I_B) = \sqrt{\frac{\sum_{j=1}^{K} |CCDC_A(j) - CCDC_B(j)|}{K}} \tag{2.2}$$

waarin $f_A(j)$ en $f_B(j)$ de centrum-contourafstanden zijn van het j-de punt van de twee contouren en K het totaal aantal punten op elke curve is. Het gebruik van een dergelijke af-



Figuur 2.7: Voorbeeld van een centrum-contourafstandscurve (CCDC).

standsmaat is echter zeer gevoelig voor kleine faseverschuivingen en kan bijgevolg falen om gelijkenissen tussen twee bladeren op te merken. Een andere methode om curves te vergelijken die minder gevoelig is voor deze verschuivingen maakt gebruik van Fourieranalyse.

2.5.2 Fouriertransformatie van contourfuncties

Fourieranalyse behoort tot de spectrale technieken, er wordt gewerkt in het frequentiedomein. Met behulp van de Fouriertransformatie kunnen signalen gedecomposeerd worden in sinus- en cosinusfuncties, ook harmonische functies genoemd (Burger en Burge, 2008). Door de contour van een blad te beschouwen als een periodieke functie (een volledige omwenteling is één periode) wordt het mogelijk hiervan de Fouriercoëfficiënten te bepalen. De lage-frequentie Fouriercoëfficiënten modelleren de globale vormeigenschappen zoals lengte en breedte, de hoge-frequentie coëfficiënten zijn gerelateerd aan de details van de contour (Sahbi, 2007).

Contourfuncties zoals de CCDC kunnen gebruikt worden voor de berekening van Fourier coëfficiënten. Karrels (2006) toonde aan dat Fouriertransformatie van de CCDC gebruikt kan worden voor bladherkenning. Aangezien de CCDC een discreet signaal is, wordt de discrete Fouriertransformatie (DFT) gebruikt.

De DFT kan eenvoudig en elegant genoteerd worden door gebruik te maken van complexe getallen. De reeks van K getallen $g_0, ..., g_{K-1}$ wordt door de DFT getransformeerd naar de reeks van K complexe getallen $z_0, ..., z_{K-1}$ volgens volgende formule (Smith, 1998):

$$z_h = \sum_{j=0}^{K-1} g_j e^{-\frac{2\pi \mathrm{i}}{N} hn} \qquad , h = 0, \ldots, K-1$$

waarbij i de imaginaire eenheid voorstelt ($\mathrm{i}^2 = -1$). Deze transformatie kan ook genoteerd worden met het symbool $\mathcal{F}$, zodat $\mathbf{z} = \mathbf{g}(x)$.

Uit de complexe getallen z_h kunnen de amplitude en fase van de sinusoidale componenten van het inputsignaal berekend worden als respectievelijk de modulus en het argument:

$$\begin{array}{rcccl} A_h & = & |z_h| & = & \sqrt{\mathrm{Re}(z_h)^2 + \mathrm{Im}(z_h)^2}, \\ \varphi_h & = & \arg(z_h) & = & \mathrm{atan2}\left(\mathrm{Im}(z_h), \mathrm{Re}(z_h)\right) \end{array}$$

met atan2 de 2-argumenten notatie van de boogtangens. Rotatie en beginpunt hebben enkel invloed op de fase, de amplitudes zijn rotatie- en beginpuntinvariant.

Een andere *contour descriptor* die toelaat de contour te beschrijven aan de hand van de FFT maakt gebruik van de geordende contour voorgesteld in complexe coördinaten (Berlin University of Technology, 2009). De contourvoorstelling C kan eenvoudig omgezet worden naar complexe coördinaten, de rijcoördinaat wordt gebruikt als het reële deel en de kolomcoördinaat als het imaginaire deel:

$$\mathbf{c} = \begin{bmatrix} k_1 & + & \mathrm{i}\ l_1 \\ k_2 & + & \mathrm{i}\ l_2 \\ & \vdots & \\ k_K & + & \mathrm{i}\ l_K \end{bmatrix}.$$

De reeks $\mathbf{c}$ van complexe getallen kan net als de CCDC rechtstreeks als input voor DFT gebruikt worden.

2.5.3 Elliptische Fouriertransformatie

De elliptische Fouriertransformatie (EFT) is eveneens een manier om een gesloten contour uit te drukken in Fouriercoëfficiënten en werd voorgesteld door Kuhl en Giardina (1982). De functies zijn niet alleen nuttig om vormen te beschrijven, maar worden ook vaak gebruikt om randen te effenen. Kuhl en Giardina gebruikten de kettingcode van de contour (Figuur 1.4) als input, maar mits kleine aanpassingen kunnen rij- en kolomcoördinaten van de contourpunten gebruikt worden. De resulterende Fourierdescriptoren zijn na normalisatie onafhankelijk van rotatie, schaal, translatie en beginpunt op de contour (Kuhl en Giardina, 1982).

De kettingcode kan voorgesteld worden als $\mathbf{v} = a_1, a_2, a_3, ..., a_{K+1}$ met $a_i \in \{0, ..., 7\}$ voor elke stap $i \in \{1, ..., K+1\}$ met K de totale lengte (Figuur 1.4). De verandering van de l- en k-coördinaat terwijl men van een pixel naar de volgende gaat is dan:

$$\Delta l_i = \text{sign}(6 - a_i)\ \text{sign}(2 - a_i),$$
$$\Delta k_i = \text{sign}(4 - a_i)\ \text{sign}(a_i),$$

waarbij

$$\text{sign}(g) = \begin{cases} 1 & , g > 0, \\ 0 & , g = 0, \\ -1 & , g < 0. \end{cases}$$

Met g een getal. Als nu het begin van de kettingcode als startpunt wordt gekozen, dan is de projectie op de x- en y-as van de t-de link in de ketting respectievelijk

$$l(t) = \sum_{i=1}^{t} \Delta l_i,$$
$$k(t) = \sum_{i=1}^{t} \Delta k_i.$$

Aldus kan de benaderende Fourierreeks berekend worden voor de l- en de k-projectie van de kettingcode in functie van de tijdstap t:

$$l_H(t) = A_0 + \sum_{h=1}^{N} a_n \cos\left(\frac{2h\pi t}{t_K}\right) + b_h \sin\left(\frac{2h\pi t}{t_K}\right),$$
$$k_H(t) = C_0 + \sum_{h=1}^{N} c_n \cos\left(\frac{2h\pi t}{t_K}\right) + d_h \sin\left(\frac{2h\pi t}{t_K}\right),$$

met H het aantal Fourier harmonischen dat in rekening gebracht wordt. t_K is de basisperiode van de kettingcode, ofwel de tijd om de volledige omtrek te doorlopen. Elke harmonische h heeft 4 coëfficiënten a_h, b_h, c_h en d_h. A_0 en C_0 zijn de bias-coëfficiënten die overeenkomen met een frequentie van 0. De waarde ervan is te wijten aan translatie en is hier dus van geen betekenis. Men kan aantonen dat de overige coëfficiënten als volgt kunnen berekend worden

(Kuhl en Giardina, 1982):

$$a_h = \frac{1}{2h^2\pi^2}\sum_{i=1}^{K}\frac{\Delta l_i}{\Delta t_i}\left[\cos\left(\frac{2h\pi t_i}{t_K}\right)-\cos\left(\frac{2h\pi t_{i-1}}{t_K}\right)\right],$$

$$b_h = \frac{1}{2h^2\pi^2}\sum_{i=1}^{K}\frac{\Delta l_i}{\Delta t_i}\left[\sin\left(\frac{2h\pi t_i}{t_K}\right)-\sin\left(\frac{2h\pi t_{i-1}}{t_K}\right)\right],$$

$$c_h = \frac{1}{2h^2\pi^2}\sum_{i=1}^{K}\frac{\Delta k_i}{\Delta t_i}\left[\cos\left(\frac{2h\pi t_i}{t_K}\right)-\cos\left(\frac{2h\pi t_{i-1}}{t_K}\right)\right],$$

$$d_h = \frac{1}{2h^2\pi^2}\sum_{i=1}^{K}\frac{\Delta k_i}{\Delta t_i}\left[\sin\left(\frac{2h\pi t_i}{t_K}\right)-\sin\left(\frac{2h\pi t_{i-1}}{t_K}\right)\right].$$

Δt_i stelt de afstand tussen het $(i-1)$-de en het i-de punt voor ($\Delta t_i = \sqrt{(\Delta k_i)^2 + (\Delta l_i)^2}$). De gevonden coëfficiënten kunnen genormaliseerd worden voor startpunt, rotatie en grootte door een aantal bewerkingen zoals beschreven door Neto *et al.* (2006). Een gevolg van de normalisatie is dat de eerste 3 coëfficiënten (a_1, b_1 en c_1) steeds gelijk zijn aan 1, 0 en 0.

De amplitude van een harmonische wordt als volgt berekend (Tort, 2003):

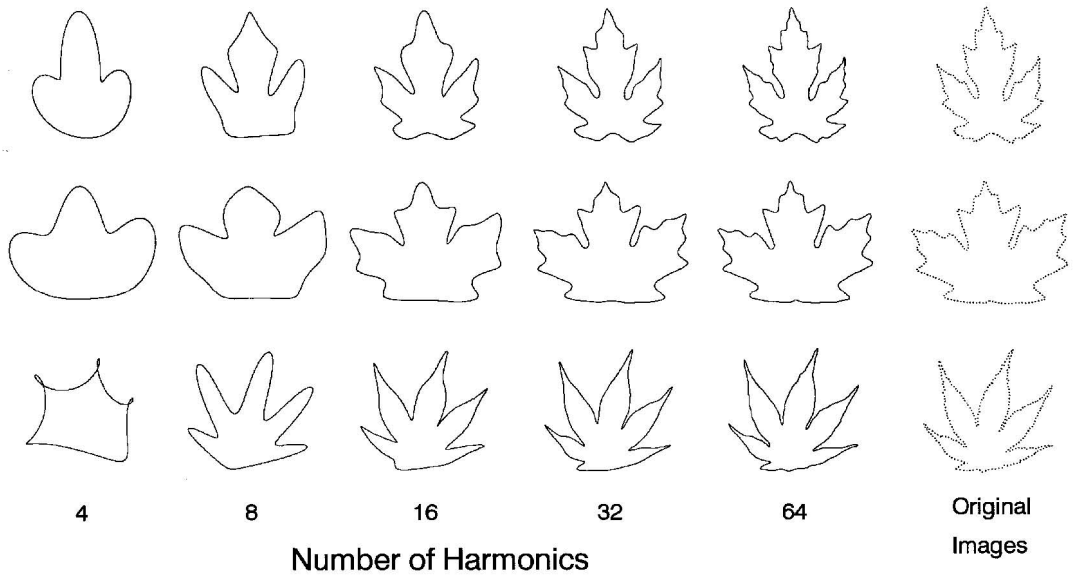
$$amp_h = \frac{1}{2}\sqrt{a_h^2 + b_h^2 + c_h^2 + d_h^2}.$$

In Figuur 2.8 is te zien in welke mate de originele contour gereconstrueerd kan worden met de eerste 4, 8, 16, 32 of 64 harmonischen. Volgens McLellan en Endler (1998) worden goede overeenkomsten bekomen vanaf 20 harmonischen en wordt de verbetering in fit tussen reconstructie en origineel bij toevoegen van meer harmonischen daarna erg laag, dit kan men ook nagaan in de figuur.

2.6 Textuurfeatures

Bovenstaande regio- en contourfeatures worden afgeleid van het silhouet (binaire beeld), echter zijn er ook enkele auteurs die gebruik maken van kleur-, textuur- en nervatuurfeatures, bv. Fu *et al.* (2005) en Wu *et al.* (2007b) en Man *et al.* (2008) en Nam *et al.* (2008). Man *et al.* (2008) gebruikten features die afgeleid worden uit de distributies van de intensiteit van de verschillende kleurenbanden. In Casanova *et al.* (2009) wordt de textuur beschouwd aan de hand van *Gabor wavelets*. *Gabor wavelets* worden vaker gebruikt voor textuuranalyse. De techniek is in zekere zin vergelijkbaar met 2-dimensionale Fourieranalyse (Nixon en Aguado, 2008). De nervatuur kan teruggevonden worden via filters voor *ridge*- (zie Sectie 2.3.3) en *edge*-detectie (bv. de Canny-filter) (Burger en Burge, 2008).

Uit de *computer vision* is er ook een techniek beschikbaar die de textuur van een beeld beschrijft aan de hand van de lokale oriëntatie van de intensiteitsgradiënt, namelijk *histogram of oriented gradients* (HOG) descriptors. Daarvoor wordt een raster over het beeld gelegd en wordt in elke cel van dat raster de richting van eventuele randen en textuur geëvalueerd in een histogram. Deze techniek wordt bijvoorbeeld gebruikt om mensenlichamen te herkennen op foto's (Dalal en Triggs, 2005). Xiao *et al.* (2010) gebruiken HOG-descriptor voor het classificeren van bladeren, daarbij worden de bladeren gestandaardiseerd en er wordt een witte achtergrond gebruikt. Hoewel oorspronkelijk ontwikkeld om textuur te beschrijven zullen de HOG-descriptors dan voornamelijk de bladrand karakteriseren aangezien de overgang op de achtergrond sterke gradiënten in het beeld veroorzaakt.



Figuur 2.8: Reconstructie van beelden op basis van verschillende aantallen van elliptische Fourier harmonischen. (Boven) *Acer saccharinum*, (midden) *Acer saccharum*, (onder) *Acer palmatum* (McLellan en Endler, 1998).

Op de textuurfeatures wordt echter niet dieper ingegaan omdat ze in deze thesis niet gebruikt worden. Deze keuze is te verantwoorden omdat de waargenomen textuur afhangt van verschillende factoren zoals het jaargetijde en het type fototoestel of scanner. Bijgevolg zijn zeer robuuste technieken nodig om de textuur te kunnen beschrijven en dit schept de nood aan doorgedreven beeldverwerking (denk aan combinaties van verschillende filters en perfecte afstelling van parameters). Dit behoort echter niet meer tot het bereik van deze thesis.

Hoofdstuk 3

Machine learning modellen

In dit hoofdstuk worden de principes van de classificatiemodellen beschreven die in Deel II gebruikt worden. Eerst volgt een algemene inleiding over de verschillende types van multiklasse classificatie.

3.1 Multiklasse classificatiestrategieën

Een typisch classificatieprobleem is een binair probleem waarbij de data in twee klassen ingedeeld wordt. Het determineren van bomen is echter geen binair, maar een multiklasse classificatieprobleem. Dit wil zeggen dat de voorspelde waarden steeds komen uit een eindige ongeordende verzameling $\{c_1, ..., c_Q\}$. Bepaalde technieken zoals de gekende classificatiebomen en *nearest-neighbor* classificatie zijn van nature in staat meerdere klassen te onderscheiden, maar de meeste classificatiemethoden zijn ontwikkeld voor binaire classificatie. Om met deze modellen toch meerdere klassen te voorspellen bestaan verschillende oplossingen (Witten en Frank, 2005 en Hastie *et al.*, 2008):

1. de modellen uitbreiden zodat ze meerdere klassen toelaten,
2. het multiklasse probleem opsplitsen in binaire deelproblemen, dit kan op twee manieren:
   (a) één tegen allen: één model per klasse dat 1 voorspelt indien de instantie tot de klasse behoort, 0 indien dit niet zo is, de klasse van het model dat de hoogste waarde voorspelt is dan de output,
   (b) één tegen één: een model voor elke combinatie van twee verschillende klassen, voorspellingen volgen dan uit een stemming, de klasse met die het vaakst voorspeld wordt is de output.

In de literatuur omtrent classificatie van planten worden in meerdere artikels *k-nearest neighbor classifiers* of hiermee verwante methoden gebruikt (Du *et al.*, 2006 en Nam *et al.*, 2008). Soms wordt de gelijkenis met alle bladeren in een dataset berekend en wordt dan *nearest neighbor* classificatie uitgevoerd (Wang *et al.*, 2003 en Lee en Chen, 2006). Andere gebruikte technieken zijn *moving (median) centers hypersphere classifiers* (MMCH) (Du *et al.*, 2007 en Wang *et al.*, 2008), (probabilistische) neurale netwerken (Fu *et al.*, 2005 en Wu *et al.*, 2007b) en *support vector machines* (Man *et al.*, 2008 en Liu *et al.*, 2009 en Solé-Casals *et al.*, 2009).

3.2 Lineaire methoden

Beschouw opnieuw het classificatieprobleem. Aanzien de afhankelijke variabele y categorisch is ($y_i \in \{c_1, c_2, ..., c_Q\}$) kan de feature-ruimte opgedeeld worden in Q regio's volgens het klassenlabel van de instanties die zich er bevinden. De grenzen tussen de regio's kunnen verschillende vormen aannemen. Bij de lineaire methoden wordt gewerkt met grenzen die lineair zijn in $\mathbf{x}$. Echter kunnen ook niet lineaire grenzen berekend worden door de feature-set uit te breiden met onder meer machts- en interactietermen. Dit kan gezien worden als een featuretransformatie $\Phi : \mathbb{X} \to \mathbb{X}'$, waarmee een vector $\mathbf{x}' = \Phi(\mathbf{x})$ berekend wordt. Grenzen die lineair zijn in $\mathbf{x}'$ hoeven dan niet lineair te zijn in $\mathbf{x}$. Kwadratische grenzen in $\mathbf{x}$ kunnen bijvoorbeeld berekend worden door kwadraten en kruisproducten ($x_1^2, x_2^2, ..., x_1x_2, ...$) toe te voegen aan de originele ruimte.

3.2.1 Logistische regressie

De meest eenvoudige classificatiemethode behandelt het classificatieprobleem als een regressieprobleem. Voor elke klasse q wordt een regressiemodel opgesteld in functie van de features $x_1, ..., x_P$. Deze modellen zijn van de vorm $\hat{f}_q(\mathbf{x}) = \beta_{q,0} + \boldsymbol{\beta}_q^{\mathrm{T}}\mathbf{x}$, met $\beta_{q,0}$ het intercept en $\boldsymbol{\beta}_q$ een vector van P coëfficiënten. Bij het opstellen van de modellen is de responsvariabele 1 voor instanties die tot klasse q behoren en 0 voor alle andere klassen. Aldus wordt voor elke klasse een waarde bekomen die de waarschijnlijkheid uitdrukt dat de instantie tot de klasse behoort. Deze met de hoogste waarde is dan de verwachte klasse (Witten en Frank, 2005). De beslissingsgrens (*decision boundary*) tussen klasse q_1 en q_2 wordt gevormd door de verzameling van punten waarvoor $\hat{f}_{q_1}(\mathbf{x}) = \hat{f}_{q_2}(\mathbf{x})$. Dit is een hypervlak in $P+1$ dimensies. Hetzelfde geldt voor elk ander paar van twee klassen, elke klasseregio in de featureruimte wordt dus gescheiden door een aantal hypervlakken. Om nieuwe instanties te voorspellen dient enkel te worden nagegaan in welk regio die zich bevinden (Hastie *et al.*, 2008). Nadelen van het model zijn dat de berekende waarschijnlijkheidswaarden geen echte kansen zijn, ze kunnen namelijk buiten het interval [0,1] liggen, ook is de som over de verschillende klassen niet gelijk aan één. Daarnaast wordt, indien kleinste kwadratenschatting gebruikt wordt, verondersteld dat de afwijkingen onafhankelijk, normaal verdeeld zijn met constante standaardafwijking. Dit is echter duidelijk niet het geval aangezien er enkel data beschikbaar zijn bij de waarden 0 en 1 (Witten en Frank, 2005).

Bovenstaande regressie kan gezien worden als een methode die een discriminantfunctie $\delta_q(\mathbf{x})$ modelleert voor elke klasse en $\mathbf{x}$ indeelt bij de klasse met de hoogste waarde voor $\delta_q(\mathbf{x})$ (Hastie *et al.*, 2008). Dit geldt ook voor methoden die de probabiliteit van een bepaalde klasse gegeven een featurevector $\mathbf{x}$ modelleren. Deze *a posteriori* kans wordt genoteerd als $\Pr(y_i = c_q \mid \mathbf{x} = \mathbf{x}_i)$. Wanneer $\Pr(y_i = c_q \mid \mathbf{x} = \mathbf{x}_i)$ lineair is zullen de beslissingsgrenzen ook lineair zijn. Het is zelfs voldoende dat een monotone transformatie van δ_k of $\Pr(y_i = c_q \mid \mathbf{x} = \mathbf{x}_i)$ lineair is om lineaire grenzen te bekomen.

Voor een 2-klassenprobleem met $y \in \{0, 1\}$ kunnen de *a posteriori* kansen als volgt gemodelleerd worden:

$$\Pr(y_i = 1 \mid \mathbf{x} = \mathbf{x}_i) = \frac{e^{(\beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x})}}{1 + e^{(\beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x})}} \tag{3.1}$$

$$\Pr(y_i = 0 \mid \mathbf{x} = \mathbf{x}_i) = \frac{1}{1 + e^{(\beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x})}} \tag{3.2}$$

De monotone transformatie die dan kan gebruikt worden is de logit transformatie $\text{logit}(a) = \log \frac{a}{1-a}$. Deze leidt namelijk tot een lineaire functie:

$$\log \frac{\Pr(y_i = 1 \mid \mathbf{x} = \mathbf{x}_i)}{\Pr(y_i = 0 \mid \mathbf{x} = \mathbf{x}_i)} = \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}.$$

De beslissingsgrens is dan het hypervlak waarvoor de kans op beide klassen gelijk is. Dit vlak wordt gedefinieerd door $\{x \mid \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x} = 0\}$. Twee populaire technieken die gebruik maken van de logit functie zijn lineaire discriminant analyse en lineaire logistische regressie. Het verschil tussen beide methoden is de manier waarop de lineaire functies gefit worden aan de data (Hastie *et al.*, 2008).

Bij logistische regressie wordt getracht volgende *likelihood* te optimaliseren, gebruik makende van Vgln. (3.1) en (3.2) (Zhu en Hastie, 2005):

$$\mathcal{L}(T, \boldsymbol{\beta}) = \prod_{i=1}^{N} \Pr(y_i = 1 \mid \mathbf{x} = \mathbf{x}_i)^{y_i} \Pr(y_i = 0 \mid \mathbf{x} = \mathbf{x}_i)^{1-y_i} - \lambda \sum_{p=1}^{P} \|\beta_p\|, \tag{3.3}$$

met λ een regularisatieparameter. In bovenstaande formule wordt regularisatie van de L_1-norm ($L_1 = \sum_{p=1}^{P} \|\beta_p\|$) toegepast. Alternatief kan de L_2-norm geregulariseerd worden ($L_2 = \sum_{p=1}^{P} \|\beta_p\|^2$). Merk op dat het intercept β_0 niet afgestraft wordt.

De doelfunctie voor een gegeven dataset T is dan:

$$\boldsymbol{\beta}^* = \arg\max_{\beta_0, \boldsymbol{\beta}} \mathcal{L}(T, \boldsymbol{\beta}). \tag{3.4}$$

Multinomiale logistische regressie is een uitbreiding van het 2-klassenprobleem uit Vgln. (3.1) tot (3.4) voor meerdere klassen ($y \in \{c_1, ..., c_Q\}$) en heeft als voordeel ten opzichte van eenvoudige logistische regressie voor elke klasse afzonderlijk dat de som van de kansen gelijk is aan 1 (Witten en Frank, 2005). Het model is van de vorm (Hastie *et al.*, 2008):

$$\begin{cases} \log \dfrac{\Pr(y_i = c_1 \mid \mathbf{x} = \mathbf{x}_i)}{\Pr(y_i = c_Q \mid \mathbf{x} = \mathbf{x}_i)} &= \beta_{1,0} + \boldsymbol{\beta}_1^{\mathrm{T}}\mathbf{x} \\ \log \dfrac{\Pr(y_i = c_2 \mid \mathbf{x} = \mathbf{x}_i)}{\Pr(y_i = c_Q \mid \mathbf{x} = \mathbf{x}_i)} &= \beta_{2,0} + \boldsymbol{\beta}_2^{\mathrm{T}}\mathbf{x} \\ & \vdots \\ \log \dfrac{\Pr(y_i = c_{Q-1} \mid \mathbf{x} = \mathbf{x}_i)}{\Pr(y_i = c_Q \mid \mathbf{x} = \mathbf{x}_i)} &= \beta_{Q-1,0} + \boldsymbol{\beta}_{Q-1}^{\mathrm{T}}\mathbf{x}. \end{cases}$$

Het model bestaat dus uit $Q-1$ logit transformaties. In bovenstaande formules werd steeds de laatste klasse, Q, in de noemer gebruikt, deze klasse is echter vrij te kiezen. De *a posteriori* kans van elke klasse is dan (Hastie *et al.*, 2008):

$$\Pr(y_i = c_q \mid \mathbf{x} = \mathbf{x}_i) = \frac{e^{\beta_{q,0} + \boldsymbol{\beta}_q^{\mathrm{T}}\mathbf{x}}}{1 + \sum_{r=1}^{Q-1} e^{\beta_{r,0} + \boldsymbol{\beta}_r^{\mathrm{T}}\mathbf{x}}}, \qquad \text{voor } q = 1, ..., Q-1$$

$$\Pr(y_i = c_Q \mid \mathbf{x} = \mathbf{x}_i) = \frac{1}{1 + \sum_{r=1}^{Q-1} e^{\beta_{r,0} + \boldsymbol{\beta}_r^{\mathrm{T}}\mathbf{x}}}.$$

Uit deze vergelijkingen kan eenvoudig gezien worden dat de som van de kansen één is.

3.2.2 Support vector machines

In de vorige sectie werd gezocht naar hypervlakken die twee klassen optimaal van elkaar scheiden. *Support vector machines* (SVM) zoeken eveneens optimaal scheidende hypervlakken door de vrije marge rond de beslissingsgrens tussen twee klassen zo breed mogelijk te maken. Dit wordt geïllustreerd in Figuur 3.1(a). Deze techniek heeft ook een uitbreiding voor niet lineair scheidbare klassen. Deze laat toe dat een aantal punten zich aan de verkeerde kant van de marges bevinden. In Figuur 3.1(b) zijn deze aangeduid met de notatie ζ^*.

Het berekenen van een *support vector machines* model is een kwadratisch optimalisatieprobleem met enkele lineaire ongelijkheden als beperkingen (Hastie *et al.*, 2008):

$$\min_{\boldsymbol{\beta},\beta_0} \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\sum_{i=1}^{N}\zeta_i \tag{3.5}$$

$$\text{onderhevig aan : } \zeta_i \leq 0,\ y_i(\mathbf{x}_i^{\mathrm{T}}\boldsymbol{\beta} + \beta_0) \leq 1 - \zeta_i\ \forall i, \tag{3.6}$$

met C een kostparameter ($C = \infty$ in het lineair scheidbare geval).

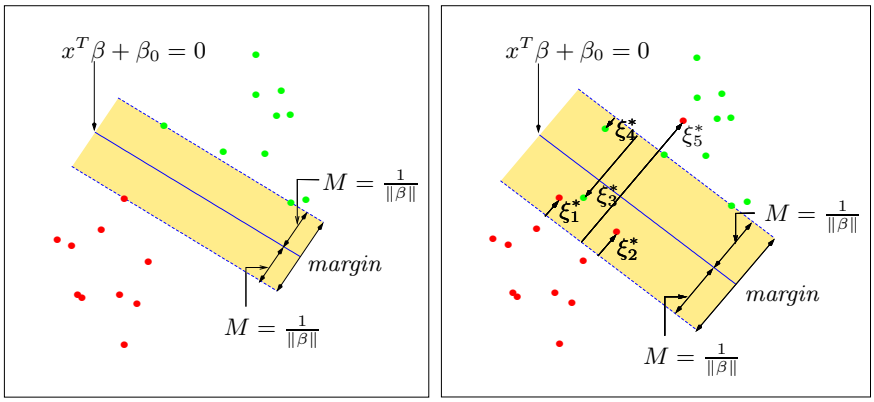
De optimalisatie kan uitgevoerd worden met behulp van *Lagrange multipliers*. Beschouw een 2-klassenprobleem met $y \in \{-1, 1\}$. Volgens de afleiding in Hastie *et al.* (2008) kan het fitten van een SVM herleid worden tot de optimalisatie van de volgende Langrange duale objectief functie:

$$L_D = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^{\mathrm{T}}\mathbf{x}_j, \tag{3.7}$$

er moet gelden dat $0 \leq \alpha_i \leq C$ en $\sum_{i=1}^{N}\alpha_i y_i = 0$.

3.3 Ensemble methoden met classifictiebomen

Ensemble methoden zijn gebaseerd op het idee dat door combinatie van een aantal eenvoudige basismodellen een beter voorspellend model kan bekomen worden. Door de resultaten van de verschillende basismodellen te combineren daalt de variantie zonder dat de bias toeneemt. Als basismodel wordt voornamelijk de algemeen bekende classificatieboom gebruikt. Classifica-



(a) (b)

Figuur 3.1: *Support vector classifier.* Voor (a) lineair scheidbare klassen, (b) niet lineair scheidbare klassen met $\zeta_i^* = M\zeta_i$ (Hastie *et al.*, 2008).

tiebomen zijn in staat om complexe interacties tussen variabelen te modelleren en hebben een relatief lage bias. Bovendien is het mogelijk om op basis van een dataset meerdere bomen op te bouwen die onderling sterk van elkaar verschillen, maar toch gelijkaardige accuraatheden behalen. De gemaakte fouten zullen echter per boom verschillen. Deze eigenschappen maken dat classificatiebomen baat hebben bij uitmiddeling en dus de ideale kandidaten zijn voor ensemble methoden (Hastie *et al.*, 2008).

3.3.1 Bagging, boosting en random forests

De eenvoudigste techniek om meerdere classificatiebomen te creëren en te combineren is *bootstrap aggregation* of *bagging*. Denk terug aan het classificatieprobleem dat geïntroduceerd werd in de notatiesectie. Uit de dataset $T = \{(y_1, \mathbf{x}_1), ..., (y_N, \mathbf{x}_N)\}$ worden voor bagging b_{max} (bv. $b_{max} = 100$) *bootstrap* datasets T_b afgeleid (met $b = 1, 2, ..., b_{max}$). Dit gebeurt door telkens N willekeurige instanties te selecteren uit T, de randomselectie gebeurt met teruglegging. De *bootstrap* datasets zijn dus even groot als de volledig trainingsdataset, maar ze bevatten een aantal elementen niet en andere meerdere keren. Voor elke *bootstrap* dataset wordt dan een classificatieboom opgebouwd en elke boom geeft een voorspelde klasse $\hat{y}_b(\mathbf{x})$. De voorspelling door *bagging* is dan het gemiddelde van de voorspellingen van de b_{max} *bootstrap* modellen (Hastie *et al.*, 2008). In (Hastie *et al.*, 2008) zijn de formules gegeven voor regressie, echter voor classificatie met meerderheidsstemming van de submodellen wordt dit:

$$\hat{y}_{bagging}(\mathbf{x}) = \underset{c_q \in \{c_1,...,c_Q\}}{\arg\max} \sum_{b=1}^{b_{max}} \mathrm{I}[\hat{y}_b(\mathbf{x}) = c_q],$$

hierin is I de indicatorfunctie en c_q het soortlabel.

Meer geavanceerde technieken zijn *boosting* en *random forests*. *Boosting* is een iteratief proces waarbij gewerkt wordt met gewogen versies van de dataset: instanties die in eerdere iteraties foutief geclassificeerd werden krijgen een hoger gewicht toegekend. Bij het combineren van de modellen wordt bij *boosting* ook rekening gehouden met de performantie van het submodel. Goede submodellen krijgen meer belang in de finale voorspelling (Hastie *et al.*, 2008). *Boosting* is wijdverspreid met het Adaboost algoritme voor twee klassen (Freund en Schapire, 1996, 1997), maar recent zijn ook theorieën beschikbaar over *boosting* voor multiklasseproblemen. De finale voorspelling wordt dan als volgt berekend (Mukherjee en Schapire, 2010):

$$\hat{y}_{boosting}(\mathbf{x}) = \underset{c_q \in \{c_1,...,c_Q\}}{\arg\max} \sum_{b=1}^{b_{max}} \mathrm{I}[\hat{y}_b(\mathbf{x}) = c_q)]\alpha_b, \tag{3.8}$$

met α_b het gewicht van het b-de submodel. Veelal worden voor *boosting stumps* gebruikt als submodellen, dit zijn classificatibomen die slechts uit één spliting bestaan. De resultaten van *boosting* zijn over het algemeen beter dan deze bekomen met *bagging* (Hastie *et al.*, 2008).

Random forests (Breiman, 2001) zijn een gemodificeerde vorm van *bagging*. Er wordt namelijk uitgemiddeld over een collectie van gedecorreleerde bomen. Elke boom in een *bagging*-algoritme komt uit dezelfde distributie van bomen, de verwachtingswaarde van de voorspelling van een bepaalde instantie door b_{max} dergelijke bomen is gelijk aan de verwachtingswaarde van elke boom afzonderlijk. Bijgevolg is de bias van de gecombineerde bomen gelijk aan de de bias van de individuele bomen en is de enige manier om de performantie te verbeteren door het verlagen van de variantie. Dit is in contrast met *boosting*, waar de bias verminderd wordt en waar de bomen geen identieke distributie hebben.

De gemiddelde variantie van b_{max} identiek en onafhankelijk verdeelde variabelen, elk met variantie σ^2 bedraagt $\frac{1}{b_{max}}\sigma^2$. Voor identiek maar niet onafhankelijk verdeelde variabelen wordt dit $\rho\sigma^2 + \frac{1-\rho}{b_{max}}\sigma^2$, met ρ de paarsgewijze correlatie. Bij grote b_{max} wordt de tweede term verwaarloosbaar en wordt de variantie dus bepaald door de correlatie tussen paren van bomen. Het opzet van *random forests* is dan om de variantie te reduceren door de correlaties te verminderen. Daartoe wordt bij de opbouw van de boom in elke splitsing een aantal $m \leq P$ random gekozen variabelen geselecteerd uit $\{x_1, x_2, ..., x_P\}$ waarop gesplitst kan worden (Breiman, 2001 en Hastie *et al.*, 2008). Het *random forests* algoritme is weergegeven in Algoritme 3.1.

Over het algemeen is de performantie van *random forests* vergelijkbaar met die van *boosting* algoritmes. *Random forests* zijn echter eenvoudiger te trainen en te tunen (Hastie *et al.*, 2008) en volgens Breiman (2001) zijn ze ook minder gevoelig voor ruis.

3.3.2 De out-of-bag (OOB) foutenschatting

Met *random forests* is het mogelijk een soort interne kruisvalidatie uit te voeren die de fout op testdata schat zonder dat er nood is aan afzonderlijke testdata.

Elke boom wordt opgebouwd op basis van een verschillende random geselecteerde *bootstrap sample* T_b. Aangezien sommige instanties meerdere keren voorkomen zijn er ook instanties die niet in T_b voorkomen, deze worden *out-of-bag* (OOB) genoemd. De kans dat een bepaalde instantie niet in een bepaalde T_b zit wordt gegeven door $(1 - 1/N)^N \approx e^{-1} = 0,368$ (Efron en Tibshirani, 1993). Ongeveer een derde van de instanties wordt dus niet gebruikt bij het opstellen van de b-de boom. Deze kunnen als beschikbare testdata voor die boom gezien worden. Elke instantie kan bijgevolg getest worden in ongeveer één derde van de bomen. Uit de voorspellingen van de bomen waarvoor een instantie *out-of-bag* (OOB) was wordt de populairste klasse genomen als uiteindelijke voorspelling. Op basis van deze voorspellingen wordt dan het foutenpercentage van het random forest berekend: de OOB-foutenschatting. Verschillende tests hebben aangetoond dat de OOB-foutenschatting een onafhankelijke (*unbiased*) schatting is van de generalisatiefout (Hastie *et al.*, 2008).

3.4 Kernelmethoden

In voorgaande modellen werd steeds het featuredataset als input verondersteld, er zijn echter ook modellen die een kernelmatrix met similariteiten als input gebruiken. Dergelijke kernelmatrices worden al dan niet bepaald op basis van de features.

3.4.1 Kernel support vector machines

Aan de hand van een voorbeeld met support vector machines zal aangetoond worden hoe het in verschillende modellen mogelijk is om de features vervangen door een kernelfunctie.

Uit de Lagrange duale objectieffunctie voor het oplossen van SVM's (Vergelijking 3.7) blijkt dat de features enkel voorkomen onder de vorm van inwendige product termen. Daardoor is het niet nodig om de features te kennen indien men de inwendige producttermen kent. In Sectie 3.2 werd aangehaald dat features getransformeerd mogen worden naar een andere featureruimte via een functie $\Phi : \mathbb{X} \to \mathbb{X}'$. Het is dus voldoende om het inproduct van de getransformeerde features $\langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2) \rangle$ te kennen en de functie Φ zelf hoeft niet gekend te zijn. Het inwendig product kan dus vervangen worden door een functie $\mathcal{K} = \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2) \rangle$.

Algoritme 3.1 Random forest algortime voor classificatie (naar Hastie *et al.*, 2008)

Stel: $b_{max} \leftarrow$ aantal bomen

Bereken:

for $b = 1 : b_{max}$

1. Trek een *bootstrap* dataset T_b van grootte N uit de dataset T
2. Stel een boom op voor T_b, herhaal daarvoor volgende stappen tot in elke tak de minimale nodegrootte bereikt wordt:
   (a) selecteer random m variabelen uit de P beschikbare variabelen
   (b) kies uit deze m variabelen de beste split (variabele en splitsingswaarde)
   (c) splits de node in twee dochternodes

end for

Output: $\{boom\}^{b_{max}} \leftarrow$ het *random forest* ensemble van bomen

Voorspellingen:

$$\hat{y}_b(\mathbf{x}) \leftarrow \text{voorspelde klasse door de } b\text{-de boom}$$

$$\hat{y}_{RF}(\mathbf{x}) \leftarrow \underset{c_q \in \{c_1,\ldots,c_Q\}}{\arg\max} \sum_{b=1}^{b_{max}} \mathrm{I}[\hat{y}_b(\mathbf{x}) = c_q], \text{ voorspelde klasse van het } \textit{random forest}$$

Deze functie wordt een kernelfunctie genoemd. $\mathcal{K}$ moet gekend zijn voor elke combinatie van 2 instanties. Voor een gegeven dataset T met N instanties kan $\mathcal{K}$ dus ook gezien worden als een $N \times N$ matrix (de kernelmatrix). In plaats van features te berekenen kan men dus rechtstreeks proberen om een geschikte kernelmatrix te vinden. Kernels kunnen echter ook berekend worden op basis van features om zo bepaalde featuretransformaties uit te voeren. Voorbeelden daarvan zijn de n-de orde polynomiale kernels en radiale basis functies (formules zijn opgenomen in Deel II).

Deze 'kerneltruc' is toepasbaar op alle modellen waarin de features enkel als inwendige producten gebruikt worden. Dit is bijvoorbeeld ook het geval bij logistische regressie en *nearest-neighbor* (Hastie *et al.*, 2008).

3.4.2 Kernel logistische regressie

In de formules van logistische regressie (Vgln. (3.1) tot (3.4)) werd gebruik gemaakt van een functie $g(\mathbf{x}) = \beta_0 + \boldsymbol{\beta}^{\mathrm{T}}\mathbf{x}$. Bij de kernelversie van logistische regressie (KLR) wordt $g(\mathbf{x})$ vervangen door (Zhu en Hastie, 2005):

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \ \mathcal{K}(\mathbf{x}, \mathbf{x}_i) + \alpha_0,$$

met α_0 en α_i de te bepalen coëfficiënten. De kernelvarianten van de *a posteriori* kansen uit Vgln. (3.1) en (3.2) zijn dus:

$$\Pr(y_i = 1 \mid \mathbf{x} = \mathbf{x}_i) = \frac{e^{\sum_{i=1}^{N} \alpha_i \ \mathcal{K}(\mathbf{x},\mathbf{x}_i)+\alpha_0}}{1 + e^{\sum_{i=1}^{N} \alpha_i \ \mathcal{K}(\mathbf{x},\mathbf{x}_i)+\alpha_0}} \tag{3.9}$$

$$\Pr(y_i = 0 \mid \mathbf{x} = \mathbf{x}_i) = \frac{1}{1 + e^{\sum_{i=1}^{N} \alpha_i \ \mathcal{K}(\mathbf{x},\mathbf{x}_i)+\alpha_0}} \tag{3.10}$$

De regularisatie term uit Vgl. (3.3) heeft eveneens een kernelvariant: $\boldsymbol{\alpha}\mathcal{K}\boldsymbol{\alpha}$ hierin stelt $\boldsymbol{\alpha}$ de vector van coëfficiënten . De te optimaliseren *likelihood* bij KLR is dus (Zhu en Hastie, 2005):

$$\mathcal{L}(T, \boldsymbol{\beta}) = \prod_{i=1}^{N} \Pr(y_i = 1 \mid \mathbf{x} = \mathbf{x}_i)^{y_i} \Pr(y_i = 0 \mid \mathbf{x} = \mathbf{x}_i)^{1-y_i} - \lambda\boldsymbol{\alpha}\mathcal{K}\boldsymbol{\alpha},$$

met gebruik van de *a posteriori* kansen uit Vgln. (3.9) en (3.9).

3.4.3 Kernelmatrices op basis van afstandsmaten

Een meer directe manier om kernels te berekenen is op basis van een afstandsmaten. De discrepantie tussen twee binaire beelden B_A en B_B zou eenvoudig kunnen berekend worden als het percentage pixels die een verschillende waarde hebben in elk beeld (*misclassification error rate* (Burger en Burge, 2008). Deze techniek heeft als nadeel dat op het zicht onbelangrijke verschillen zoals kromming en ligging van verschillende lobben van het blad zwaar zullen doorwegen. Daarom is het beter om een maat te gebruiken die de afstand tussen de contouren van de beide bladeren berekent. Een voorbeeld van een dergelijk afstandsmaat werd reeds gegeven in Vgl. (2.2), waar men CCD-curves vergelijkt. Het probleem met de meeste afstandsmaten is dat ze zeer gevoelig zijn voor structuren die in het ene blad aanwezig zijn en in het andere niet, bijvoorbeeld ten gevolge van een beschadiging, waardoor de meting onstabiel wordt (Baddeley, 1992). In de literatuur werd een afstandsmaat gevonden die ontwikkeld is voor binaire beelden: de Δ-metriek van Baddeley (1992). Stel $\mathcal{A}$ respectievelijk $\mathcal{B}$ de verzameling van pixels van het binaire beeld B_A respectievelijk B_B die waarde 1 hebben (B_A en B_B hebben dezelfde dimensies). De afstandsmaat wordt dan als volgt bepaald door het n-de orde gemiddelde verschil tussen geregulariseerde afstandstransformaties van de twee beelden:

$$\Delta(B_A, B_B) = \left[\frac{1}{k_{max} * l_{max}} \sum_{z \in Z} |w(d(z, \mathcal{A})) - w(d(x, \mathcal{B}))|^n \right]^{1/n} \tag{3.11}$$

met $d(z, \mathcal{A})$ en $d(z, \mathcal{B})$ de kortste Euclidische afstand van pixel z tot een pixel die tot $\mathcal{A}$ respectievelijk $\mathcal{B}$ behoort, Z de verzameling van alle pixels z en n een parameter, standaard $n = 2$. w is een functie die instaat voor het beperken van de gevoeligheid voor uitschieters: $w(x) = \min(x, c)$, met c een te kiezen parameter.

De Δ metriek voldoet aan de eigenschappen van een wiskundige metriek (Baddeley, 1992):

- $\Delta(B_A, B_B) = 0 \Leftrightarrow B_A = B_B$
- $\Delta(B_A, B_B) = \Delta(B_B, B_A)$
- $\Delta(B_A, B_B) \leq \Delta(B_A, B_C) + \Delta(B_C, B_B)$

De tweede eigenschap, symmetrie, is noodzakelijk indien de afstandsmaat voor een kernel zal gebruikt worden, de overige twee eigenschappen zijn voornamelijk van theoretisch belang.

De Δ metriek kan berekend worden voor elk paar van beelden uit de dataset. Op die manier wordt een symmetrische $N \times N$-matrix bekomen. De Δ metriek kan omgezet worden naar een kernel die de similariteiten weergeeft door de afstandsmaat in te voeren in een RBF-kernel. Deze RBF-kernel is dan van de vorm:

$$\mathcal{K}(\mathbf{x}_A, \mathbf{x}_B) = e^{-\frac{\Delta(A,B)^2}{2\sigma^2}} \tag{3.12}$$

waarbij $\sigma > 0$ een parameter is (een 'grote' waarde voor σ leidt tot een min of meer 'lineaire' classifier, een kleinere σ laat meer flexibiliteit toe). Deze functie kan gebruikt worden om een matrix op te bouwen, de kernelmatrix. Klassiek wordt in plaats van $\Delta(A, B)$ de Euclidische afstand $\|\mathbf{x}_A - \mathbf{x}_B\|$ genomen, maar het is mogelijk deze te vervangen door een andere afstandsmaat (zoals hier de Baddeley-metriek).

De functie $\mathcal{K}$ die aldus bekomen wordt, stelt een inproduct voor in een getransformeerde feature-ruimte $\langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2) \rangle$, en voldoet aan volgende voorwaarden die vereist zijn voor een kernel (Schölkopf en Smola, 2002):

$$\text{Symmetrisch: } \mathcal{K}(\mathbf{x}_A, \mathbf{x}_B) = \mathcal{K}(\mathbf{x}_B, \mathbf{x}_A), \quad \forall \mathbf{x}_A, \mathbf{x}_B \in \mathbb{R}^P$$

$$\text{Positief definiet: } \sum_{i=1}^{N} \sum_{j=1}^{N} c_i c_j \mathcal{K}(\mathbf{x}_A, \mathbf{x}_B) > 0, \quad \forall c_i, c_j \in \mathbb{R}.$$

Of de kernelfunctie $\mathcal{K}$ positief definiet is, kan nagegaan worden door de eigenwaarden van de kernelmatrix te berekenen. Deze moeten daarvoor groter of gelijk aan nul zijn. Indien de matrix niet positief definiet is, kan gezocht worden naar een positief definiete matrix die zeer gelijkaardig is aan de originele matrix, een algoritme dat hiervoor gebruikt kan worden is het algoritme van Higham (Higham, 1988).

3.5 Geavanceerde modellen voor kennisextractie

Door de hiervoor besproken modellen worden de data kortweg ingedeeld in klassen. In deze sectie worden een aantal geavanceerde technieken toegelicht waarmee extra kennis aan het model kan worden toegevoegd om de accuraatheid van de predicties te verbeteren en bijkomende informatie te voorspellen.

3.5.1 Multilabelclassificatie

Bij conventionele single-labelclassificatie, zoals hiervoor besproken, wordt elke instantie $\mathbf{x}_i$ geassocieerd met een klasselabel y_i. Bij multilabelclassificatie kan elke instantie daarentegen tot verschillende klassen behoren of er wordt als het ware aan elke instantie een reeks van verschillende labels $\mathbf{y}_i$ toegekend (Tsoumakas en Katakis, 2007). Multilabelclassificatie werd voornamelijk bestudeerd in het kader van documentclassificatie volgens thema (André Elisseeff, 2001), een document kan dan behoren tot verschillende thema's. Voor elk thema kan men daartoe een binair label voorzien dat aangeeft of een document tot het thema behoort of niet (Tsoumakas en Katakis, 2007). Meer algemeen kan men stellen dat instanties op basis van verschillende normen in verschillende groepen gedeeld worden.

In het geval van boomclassificatie kan elke instantie slechts tot één enkele soort behoren, toch kan men nog steeds multilabelclassificatie toepassen door toevoegen van een aantal op de morfologie van bladeren gebaseerde labels. Denk bijvoorbeeld aan het al dan niet samengesteld zijn van een blad.

De multilabels kunnen elk afzonderlijk voorspeld worden met behulp van een apart model per label. Echter is dit niet optimaal wanneer men afhankelijkheid verwacht tussen verschillende labels, dergelijke informatie gaat immers verloren (Ghamrawi en McCallum, 2005). Technieken als *stacking* (Cheng en Hüllermeier, 2009) werden ontwikkeld om correlaties tussen labels uit te buiten en zo de accuraatheid van de voorspellingen te verhogen. Bij *stacking* worden initiëel voorspelde labels toegevoegd aan het model als variabelen om zo bij te dragen aan een betere voorspelling van de overige labels.

3.5.2 Hiërarchische classificatie

De mens maakt veelvuldig gebruik van hiërarchische structuren om zaken overzichtelijk te houden. Het is ook mogelijk voor computermodellen om gebruik te maken van hiërarchische indelingen om gerichter te kunnen classificeren (Silla en Freitas, 2011). In veel datasets die voor *machine learning* gebruikt worden is een gekende hiërarchie aanwezig. Voorbeelden hiervan zijn de taxonomische indeling van bacteriële stammen (op basis van afstamming en verwantschappen) en de hiërarchie van overkoepelende termen in document- of websiteclassificatie volgens onderwerp (Hofmann *et al.*, 2003). Algoritmes kunnen ook zelf hiërarchische structuren uit data extraheren via hiërarchische clustering. Door het gebruik van hiërarchische classificatie kan de performantie van modellen verbeteren omdat de modellen zich kunnen toespitsen op eenvoudigere deelproblemen. Ook laat het opdelen van het probleem toe dat in elk deelmodel andere variabelen een belangrijke rol spelen (Koller en Sahami, 1997). Een bezwaar tegen hiërarchische classificatie is echter dat om het onderste niveau juist te voorspellen alle voorgaande classificaties ook correct moeten zijn, gemaakte fouten zijn dus onherstelbaar (Koller en Sahami, 1997).

Hiërarchische classificatie kan gezien worden als een vorm van multilabelclassificatie waarbij de labels een hiërarchie vertonen. De eenvoudigste hiërarchische modellen worden opgebouwd door voor elke splitsing in de hiërarchische boomstructuur een apart model op te stellen (Silla en Freitas, 2011). Dit kan de performantie verhogen doordat de verschillende eenvoudigere deelproblemen meer gericht aangepakt worden, door onder meer verschillende feature-selectie in elk model (Koller en Sahami, 1997). Door de vele splitsingen worden deelproblemen bekomen die slechts een beperkt aantal instanties bevatten. wat nadelig kan zijn voor parameterschattingen. Er bestaat een statistische techniek, *shrinkage*, die de parameterschattingen van een deelmodel met weinig data afstemt op de parameters van de modellen die volgens de hiërarchie aan het deelmodel voorafgaan. Op die manier worden meer robuuste parameterschattingen bekomen (McCallum *et al.*, 1998).

Bij hiërarchische classificatie kan een hiërarchische verliesfunctie gedefinieerd worden. Deze verliesfunctie houdt niet enkel rekening met het voorspelde klasselabel, maar ook met de voorspellingen op elk niveau: hoe meer correcte niveaus, hoe minder belangrijk de fout. Een voorbeeld van een dergelijke verliesfunctie is de H-loss (Cesa-Bianchi *et al.*, 2006):

$$H(\mathbf{y}, \hat{\mathbf{y}}(\mathbf{x})) = \sum_{i=1}^{d} C_i \; \mathrm{I}[\; \hat{\mathbf{y}}_i \neq \mathbf{y}_i \;\wedge\; \hat{\mathbf{y}}_j = \mathbf{y}_j,\; j \in \mathrm{anc}(i) \;], \tag{3.13}$$

met $\mathbf{y}$ een multilabelvector met een element voor elk niveau $i \in \{1, ..., d\}$ en $C_1, ..., C_d > 0$ vaste kostencoëfficiënten, $\mathrm{anc}(i)$ zijn de nodes die voorafgaan aan i (*ancestors*).

Voor bepaalde doeleinden is het wenselijk de hiërarchische verliesfunctie te optimaliseren. Cesa-Bianchi *et al.* (2006) vonden in verdere literatuur dat een eenvoudig hiërarchisch model zoals hierboven gebruikt in verschillende onderzoeken reeds een verbetering van de H-loss geeft. Er zijn echter ook modellen die het hiërarchische verlies expliciet optimaliseren. Het *Online Hieron* van Dekel *et al.* (2004) is zo'n methode. Het *Online Hieron* is een globaal model, er wordt dus geen afzonderlijk model opgesteld voor elke node (Silla en Freitas, 2011). De techniek is gebaseerd op het brede marge principe en Bayesiaanse methoden. Met elk label v (op alle niveaus) wordt een prototype W^v geassocieerd dat tijdens training wordt aangepast. De training van het model is een optimalisatieprobleem met variërende randvoorwaarden. Het model veronderstelt gelijkenissen tussen de prototypes van opeenvolgende labels in de hiërarchie. Het algoritme is weergegeven in Algoritme 3.2. In het algoritme stelt $\mathbf{w}^v$ het

verschil voor tussen een prototype en dat van de voorganger. Er geldt dat $W^v = \sum_{u \in \mathcal{P}(v)} \mathbf{w}^u$. De notatie $\mathcal{P}(v)$ staat voor de vector met het label v zelf en de labels die daaraan voorafgaan.

Algoritme 3.2 Online Hieron (naar Dekel *et al.*, 2004)

Stel: $\forall v \in \mathbb{Y} : \mathbf{w}_1^v = \{0\}^P$

for i in $1 : N$

1. Ontvang een instantie $\{\mathbf{x}_i, y_i\}$
2. Voorspel het klasselabel:

$$\hat{y}_i = \underset{v \in \{c_1, \ldots, c_Q\}}{\arg\max} \sum_{u \in \mathcal{P}(v)} \mathbf{w}_i^u \mathbf{x}_i$$

3. Bereken het verlies:

$$L(\{\mathbf{w}^v\}, y_i, \hat{y_i}) = \sum_{v \in \hat{\mathbf{y}}_i} \mathbf{w}_i^v \mathbf{x}_i - \sum_{v \in \mathbf{y}_i} \mathbf{w}_i^v \mathbf{x}_i + \sqrt{H(\mathcal{P}(y_i), \mathcal{P}(\hat{y}_i))}$$

4. Update:

$$\begin{aligned} \mathbf{w}_{i+1}^v &= \mathbf{w}_i^v + \alpha \mathbf{x}_i, & \forall v \in \mathcal{P}(y_i) \backslash \mathcal{P}(\hat{y}_i) \\ \mathbf{w}_{i+1}^v &= \mathbf{w}_i^v - \alpha \mathbf{x}_i, & \forall v \in \mathcal{P}(\hat{y}_i) \backslash \mathcal{P}(y_i) \end{aligned}$$

met:

$$\alpha_i = \frac{L(\{\mathbf{w}^v\}, y_i, \hat{y}_i)}{H(y_i, \hat{y}_i) \|\mathbf{x}_i\|^2}$$

end for

Deel II

Praktische uitwerking van de boomdeterminatie

Hoofdstuk 4

Aanmaak van een databank

4.1 Samenstelling en hiërarchie

Er werd een dataset aangelegd van in totaal 1250 bladeren. De dataset bevat bladeren van 41 in Vlaanderen algemeen voorkomende bomen en struiken. Van elke soort is een afbeelding weergegeven in Figuur 4.1. Per soort werden ongeveer 20 tot 25 bladeren verzameld (voor exacte aantallen zie Tabel 4.1). Daarnaast werden van een aantal vermoedelijk moeilijk te onderscheiden soorten extra bladeren ingescand.

De bladeren werden verzameld van juli tot oktober. Er werd geprobeerd om de natuurlijke variabiliteit van de bladeren zo goed mogelijk te weerspiegelen door per soort bladeren van zo veel mogelijk verschillende planten te verzamelen en zowel grote als kleine bladeren te nemen.

De taxonomie van planten bestaat volgens van der Meijden (2005) uit Rijk (bv. Plantae), Klasse (bv. Magnoliopsida), Onderklasse (bv. Magnoliidae), Superorde (bv. Hamameliflorae), Orde (bv. Fagales), Familie (bv. Fagaceae), Genus (bv. Quercus) en Soort (bv. Quercus rubra). In Tabel 4.1 zijn de meest relevante groepen weergegeven voor de soorten in de databank (superorde, familie en genus). De overige groepen bevatten weinig extra informatie omdat ze voor alle soorten gelijk zijn of omdat de verdeling overeenkomt met een andere groep.

4.2 Digitaliseren

De bladeren werden gedigitaliseerd met behulp van een A4 kleurenscanner (Canon® MP520 series) en opgeslagen in jpg-formaat. Een eerste reeks bladeren werd gescand op een resolutie van 600 PPI en opgeslagen onder de standaard jpg-compressie. Omdat de jpg-compressie de bladranden vervaagt en omdat scannen op 600 PPI veel tijd in beslag neemt, werd overgeschakeld op scans met resolutie 300 PPI die opgeslagen werden zonder compressie. Deze scans zijn van gelijkaardige kwaliteit als de eerste scans, maar databank bestaat dus uit 2 soorten van beelden.

Elke afbeelding bevat één of meerdere bladeren van dezelfde soort. In Figuur 4.2 is een voorbeeld weergegeven van een scan van bladeren van de zomereik.

4.3 Preprocessing van de beelden

Voor de beeldverwerking wordt gebruik gemaakt van Matlab (The MathWorks, Inc., 2010b)® en de daaraan gekoppelde *Image Processing Toolbox*.

De eerste behandeling die de beelden ondergaan is een herschaling, meerbepaald een verkleining. De bladeren werden ingescand op hoge resolutie, maar voor de uiteindelijke toepassing is deze hoge kwaliteit niet noodzakelijk. Er moet een afweging gemaakt worden tussen kwaliteitsverlies en snelheid.

Figuur 4.1: Afbeelding van de 41 soorten in de databank

Tabel 4.1: Samenstelling en vereenvoudigde taxonomische indeling van de soorten in de databank

Naam nl	**Superorde**	**Familie**	**Genus**	**Soort**	**Instanties**
vlier(Vli)	Asteriflorae	Caprifoliaceae	*Sambucus*	*Sambucus nigra*	26
gewone es(GEs)	Asteriflorae	Oleaceae	*Fraxinus*	*Fraxinus excelsior*	24
liguster(Lig)	Asteriflorae	Oleaceae	*Ligustrum*	*Ligustrum vulgare / ovalifolium*	22
linde(Lin)	Dilleniiflorae	Tiliaceae	*Tilia*	*Tilia sp.*	27
balsempopulier(BPo)	Dilleniiflorae	Salicaceae	*Populus*	*Populus balsamifera*	20
abeel(Abe)	Dilleniiflorae	Salicaceae	*Populus*	*Populus canescens(x) / alba*	24
populier(Pop)	Dilleniiflorae	Salicaceae	*Populus*	*Populus nigra / canadensis(x)*	27
schietwilg(ScW)	Dilleniiflorae	Salicaceae	*Salix*	*Salix alba*	24
boswilg(BWi)	Dilleniiflorae	Salicaceae	*Salix*	*Salix caprea*	24
smalbladige wilg(SmW)	Dilleniiflorae	Salicaceae	*Salix*	*Salix exigua*	20
zwarte els(ZEl)	Hamamelidiflorae	Betulaceae	*Alnus*	*Alnus glutinosa*	20
witte els(WEl)	Hamamelidiflorae	Betulaceae	*Alnus*	*Alnus incana*	30
papierberk(PBe)	Hamamelidiflorae	Betulaceae	*Betula*	*Betula papyrifera*	20
berk(Ber)	Hamamelidiflorae	Betulaceae	*Betula*	*Betula pendula / pubescens*	20
haagbeuk(HBe)	Hamamelidiflorae	Betulaceae	*Carpinus*	*Carpinus betulus*	58
hazelaar(Haz)	Hamamelidiflorae	Betulaceae	*Corylus*	*Corylus avellana*	22
tamme kastanje(TKa)	Hamamelidiflorae	Fagaceae	*Castanea*	*Castanea sativa*	20
beuk(Beu)	Hamamelidiflorae	Fagaceae	*Fagus*	*Fagus sylvatica*	40
wintereik(WEi)	Hamamelidiflorae	Fagaceae	*Quercus*	*Quercus petraea*	51
zomereik(ZEi)	Hamamelidiflorae	Fagaceae	*Quercus*	*Quercus robur*	53
Amerikaanse eik(AEi)	Hamamelidiflorae	Fagaceae	*Quercus*	*Quercus rubra*	30
walnoot(Wal)	Hamamelidiflorae	Juglandaceae	*Juglans*	*Juglans regia*	16
ruwe iep(RIe)	Hamamelidiflorae	Ulmaceae	*Ulmus*	*Ulmus glabra*	23
klimop(Kli)	Rosiflorae	Araliaceae	*Hedera*	*Hedera helix*	60
wilde kardinaalsmuts(WKa)	Rosiflorae	Celastraceae	*Euonymus*	*Euonymus europaeus*	60
gele kornoelje(GKo)	Rosiflorae	Cornaceae	*Cornus*	*Cornus mas*	21
rode kornoelje(RKo)	Rosiflorae	Cornaceae	*Cornus*	*Cornus sanguinea*	23
robinia pseudoacacia(Rob)	Rosiflorae	Fabaceae	*Robinia*	*Robinia pseudoacacia*	21
meidoorn(Mei)	Rosiflorae	Rosaceae	*Crataegus*	*Crataegus sp.*	77
appel(App)	Rosiflorae	Rosaceae	*Malus*	*Malus sylvestris*	31
zoete kers(ZKe)	Rosiflorae	Rosaceae	*Prunus*	*Prunus avium*	21
zure kers(Kri)	Rosiflorae	Rosaceae	*Prunus*	*Prunus cerasus*	49
laurier(Lau)	Rosiflorae	Rosaceae	*Prunus*	*Prunus laurocerasus*	22
Amerikaanse vogelkers(AVo)	Rosiflorae	Rosaceae	*Prunus*	*Prunus serotina*	50
sleedoorn(Sle)	Rosiflorae	Rosaceae	*Prunus*	*Prunus spinosa*	22
bosroos(BRo)	Rosiflorae	Rosaceae	*Rosa*	*Rosa arvensis*	21
rimpelroos(RRo)	Rosiflorae	Rosaceae	*Rosa*	*Rosa rugosa*	15
lijsterbes(Lij)	Rosiflorae	Rosaceae	*Sorbus*	*Sorbus aucuparia*	21
veldesdoorn(VEs)	Rosiflorae	Aceraceae	*Acer*	*Acer campestre*	51
Noorse esdoorn(NEs)	Rosiflorae	Aceraceae	*Acer*	*Acer platanoides*	21
gewone esdoorn(Esd)	Rosiflorae	Aceraceae	*Acer*	*Acer pseudoplatanus*	24

Figuur 4.2: Voorbeeld van een scan van *Quercus robur* (Zomereik)

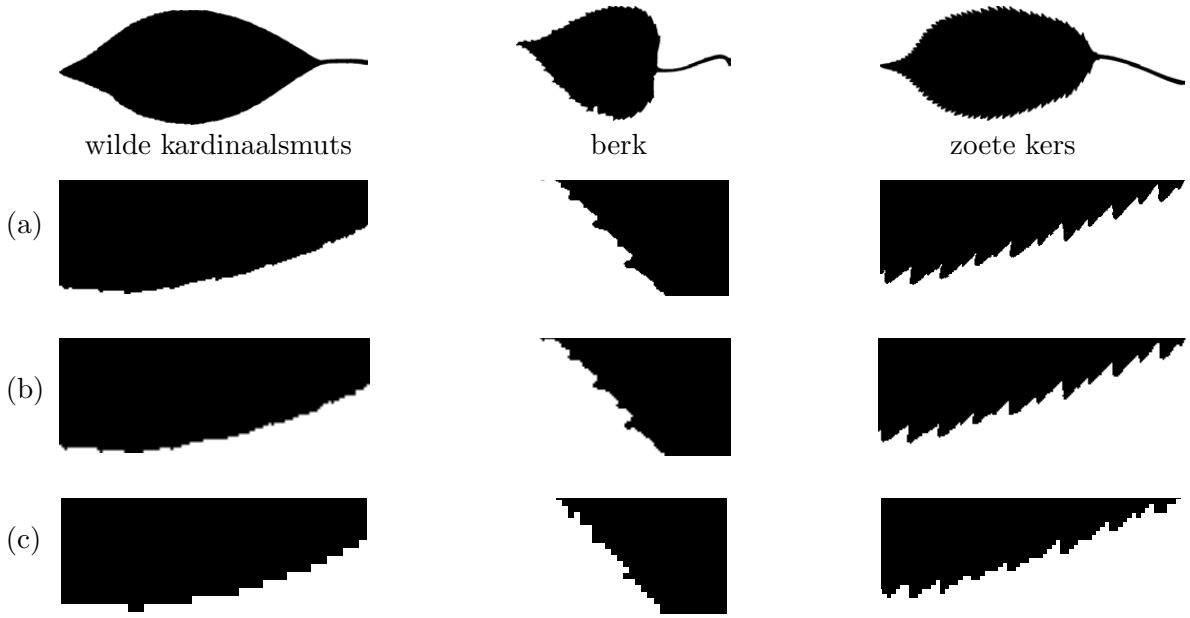
In Figuur 4.3 is te zien dat wanneer er te weinig pixels zijn de bladrand niet meer duidelijk is, maar dat het nodige aantal pixels om de bladrand weer te geven ook afhangt van het type bladrand. Bij 176 PPI is bij de 3 soorten nog reliëf zichtbaar. Aangezien de wilde kardinaalsmuts uiterst ondiep gezaagd is zal deze resolutie voor andere planten ook voldoende zijn. De scans (A4-formaat) worden daarom omgezet naar 1500 x 2060 pixels, wat overeenkomt met 176 PPI.

Daarna worden de kleurenbeelden omgezet naar grijstinten. Dit wordt gedaan aan de hand van Vergelijking (2.1) uit de literatuurstudie.

Zoals te zien in Figuur 4.2, kunnen de scans meedere bladeren bevatten. Deze worden blad per blad uitgesneden en vervolgens wordt elk blad op een egaal witte achtergrond gezet. In de literatuurstudie werd gesteld dat de bladeren gevonden kunnen worden met behulp van een grijsdrempel wanneer de achtergrond wit is. Deze techniek werkt echter niet bij bladeren met bleke delen zoals de stelen van de Amerikaanse eik (*Quercus rubra*) en de witte en grauwe abeel (*Populus sp.*). Daarnaast kunnen er ten gevolge van het reliëf van de bladeren schaduwen voorkomen langs de bladrand. Omdat het een doelstelling is om geen menselijke interactie te hebben, werd naast de grijsdrempel een kleurcriterium toegevoegd. Er wordt dus gebruik gemaakt van het kleurenbeeld. De schaduwen aan de bladrand en steel zijn grijs (hebben gelijke componenten van rood, groen en blauw), terwijl de delen van het blad meestal wel een kleur hebben zijn. Een aantal verschillende criteria werden uitgetest en een criterium dat goede resultaten geeft met de beelden uit de dataset is $|I_R - I_B| > 12$. Het totale criterium wordt dan voor elke pixel:

$$\begin{cases} I_{RGB} < t \quad \text{of} \quad |I_R - I_B| > 12 & \text{, mogelijke blad pixel,} \\ \text{anders} & \text{, niet-blad pixel,} \end{cases}$$

met t een drempelwaarde bepaald via Otsu's methode (Otsu, 1979).



Figuur 4.3: Detail van 3 verschillende resoluties: (a) 294 PPI, (b) 176 PPI en (c) 59 PPI.

Op het resulterende beeld wordt een filter toegepast die gaten kleiner dan 50 pixels in het blad opvult. Deze gaten zijn meestal ontstaan door gallen en insectenvraat. Dan wordt een segmentatie uitgevoerd met behulp van *connected-components labeling* waardoor alle mogelijke bladpixels ingedeeld worden in regio's van aaneengrenzende pixels. Een algoritme dat dit uitvoert zit geïmplementeerd in Matlab. Elke regio krijgt een verschillend cijfer dus zal elk blad en elk artefact (zwarte randen en vuil) een regio vormen. Van elke regio wordt dan de oppervlakte berekend. Indien de oppervlakte groter is dan 10000 pixels wordt de regio als blad beschouwd (tenzij de vorm zeer langwerpig is, dan wordt het genegeerd omdat het om een schaduwlijn gaat zoals te zien links en rechtsboven in Figuur 4.2). Elk blad wordt vervolgens uitgesneden volgens de regio en op een witte achtergrond geplaatst.

Voordat er features kunnen berekend worden moeten de *regions of interest* (ROI's) worden bepaald op basis waarvan de features berekend zullen worden. Voor de berekening van de features wordt het blad opgesplitst in steel en bladschijf. Deze scheiding wordt uitgevoerd via morfologische opening (Sectie 2.3.3). Als structuurelement wordt een cirkel met een straal van 10 pixels gebruikt.

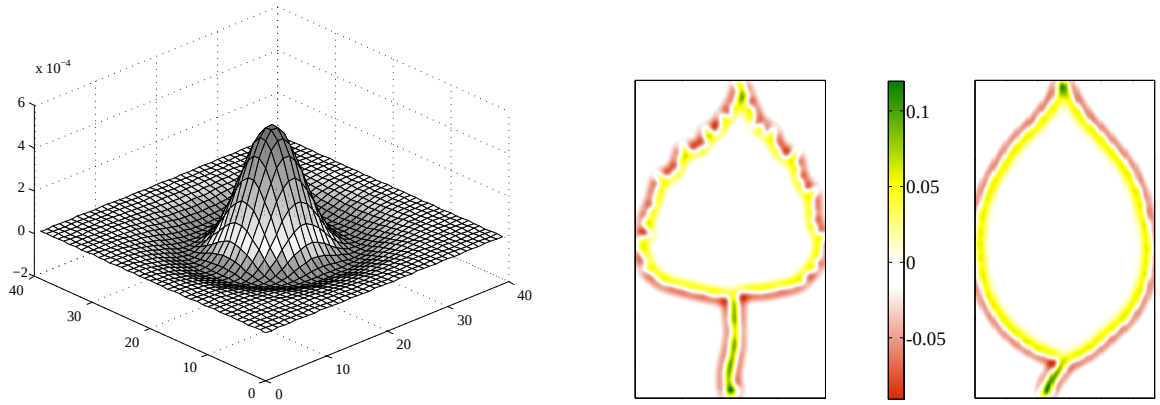
Er werd ook op een andere manier geprobeerd om de steel te verwijderen, namelijk via *ridge detection* (Sectie 2.3.3). In Figuur 4.4(a) is de kernel weergegeven die gebruikt werd om de beelden in Figuur 4.4(b) te bekomen: een *Laplacian of Gaussian* (LoG) kernel met variantie σ gelijk aan 7. Uit Figuur 4.4(b) blijkt dat zowel de steel als de bladtop en enkele tanden van het berkenblad hoge waarden vertonen. Net als wanneer morfologische opening gebruikt wordt kan dan de grootste regio uit *connected components labeling* als de steel gekozen worden. Er is dan wel eerst een drempelfilter en nodig om de mogelijke steelpixels (hoge waarden) te vinden.

Uit een vergelijkende studie van beide methoden om de steel te extraheren blijkt dat ze dezelfde tekortkoming hebben: bij sommige bladeren wordt de bladtop geselecteerd in plaats van de steel. Het is dus niet nuttig om beide methoden te combineren. De techniek via

morfologische opening is iets eenvoudiger (er is geen drempel nodig) en laat onmiddellijk toe de volledige steel te extraheren. Daarom wordt volgens de eerste methode gewerkt.

Vervolgens wordt de oriëntatie gestandaardiseerd. De features worden berekend op een recht georiënteerd blad, waarbij de bladtop naar boven wijst en de steel zich onderaan bevindt (zoals in Figuur 4.4(b)). Het gebruik van gelijk georiënteerde beelden vereenvoudigt een aantal berekeningen zoals de bepaling van de lengte en de breedte. Door het beeld van bij het begin te roteren worden latere normalisaties en rotaties van het assenstelsel overbodig. De juiste oriëntatie wordt bereikt door het blad te roteren over de hoek die de verticale as met de lengterichting van het blad maakt. Het komt regelmatig voor dat de bladsteel krom is of niet in de lengterichting van de bladschijf ligt. In dergelijke gevallen komt de bladsteel schuin te liggen. De hoek waarover geroteerd wordt is de hoek tussen de x-as en de lijn die de bladtop en de plaats waar de steel aan het blad zit verbindt. Afhankelijk van de oorspronkelijke ligging van de steel wordt deze hoek vermeerderd of verminderd met 90° zodat de steel steeds onderaan komt te liggen. Om het beeld terug in een matrixformaat te zetten met de pixels geordend in rijen en kolommen wordt *nearest-neighbor* interpolatie gebruikt.

Het extraheren van de steel (en bijgevolg oriënteren van de bladeren) loopt echter fout bij een aantal bladeren. De verschillende problemen die optreden worden geïllustreerd in Figuur 4.5. Om de invloed van de mislukte oriëntaties te minimaliseren is het dus toch nuttig om features te gebruiken die intrinsiek rotatie-invariant zijn.



(a) (b)

Figuur 4.4: Detectie van de steel met een LoG-filter, (a) LoG-kernel met $\sigma = 7$ en (b) 2 voorbeelden van bladeren waarop de LoG-filter werd toegepast.

(a) (b) (c) (d) (e)

Figuur 4.5: Typische problemen die kunnen resulteren in verkeerde oriëntatie: (a,b) zeer korte (of afwezige) stelen, (c) problematische bepaling van het hechtpunt van de steel, (d) samengestelde bladeren en (a,e) bladtop lang-smaller dan steel.

Hoofdstuk 5

Feature-extractie

In het vorige hoofdstuk werden de ROI's bepaald, op basis daarvan worden nu features geëxtraheerd. De features bestaan uit twee groepen: regiofeatures en contourfeatures.

5.1 Regiofeatures

Regio features worden berekend op basis van het binaire beeld (0/1-beeld) van het blad. Het binaire beeld van een blad kan eenvoudig bekomen worden uit de gepreprocesste dataset omdat daar de achtergrond al wit is en dus met behulp van een lage *treshold* al de witte achtergrondpixels in nul en de grijstinten in één omgezet worden. De meeste regiofeatures zijn morfologische features die worden berekend uit geometrische eigenschappen van het blad of uit de momenten.

5.1.1 Geometrische features

Omdat de bladsteel vaak voor ongewenste interferenties zorgt worden de meeste geometrische features berekend op basis van de bladschijf. De **oppervlakte** kan eenvoudig bepaald worden als het totaal aantal zwarte pixels van het zwart-wit beeld. In principe wordt gestreefd naar schaalinvariante features, maar de oppervlakte van de bladschijf kan toch worden meegenomen als feature omdat in de gebruikte dataset alle foto's dezelfde resolutie hebben. De oppervlakte kan dus gebruikt worden om na te gaan of de grootte een belangrijke bijdrage kan leveren in de classificatie, maar het is niet de aangewezen om deze feature in een finaal model te gebruiken dat met andere resoluties moet om kunnen.

Daarnaast wordt de **lengte-breedteverhouding** bepaald. Lengte en breedte worden als volgt berekend uit de recht georiënteerde bladschijf:

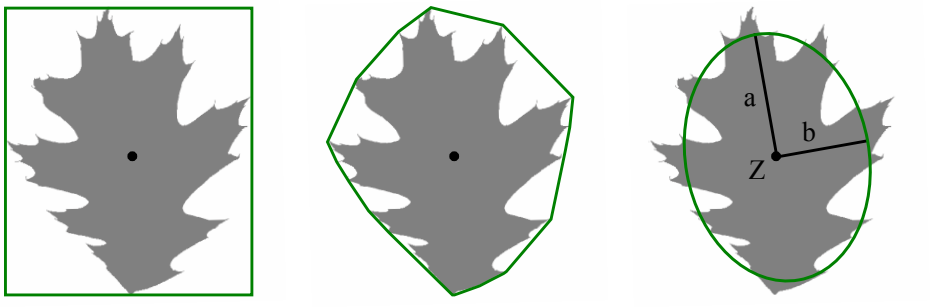
$$\text{lengte} = \max\{k \mid \exists l : B(k,l) = 1\} - \min\{k \mid \exists l : B(k,l) = 1\}$$
$$\text{breedte} = \max\{l \mid \exists k : B(k,l) = 1\} - \min\{l \mid \exists k : B(k,l) = 1\}$$

Met behulp van deze lengte en breedte wordt ook de **rechthoekigheid** berekend, dit is de verhouding van de oppervlakte van het blad en de oppervlakte van de omhullende rechthoek (Figuur 5.1(a)):

$$\text{rechthoekigheid} = \frac{O}{LB}$$

met O, L en B hier respectievelijk de oppervlakte, lengte en breedte van de bladschijf.

Een andere eigenschap die typerend is voor bepaalde soorten is of de grootste breedte boven of onder het midden ligt (van der Meijden, 2005). De **afstand tussen de grootste breedte**



(a) (b) (c)

Figuur 5.1: Illustratie van (a) de omhullende rechthoek, (b) het convex omhulsel en (c) de ellips met hetzelfde 2de orde moment als de bladschijf

en het midden wordt uitgedrukt als ratio door deze te delen door de halve lengte van de bladschijf. De grootste breedte wordt bepaald door de punten die het verst van de lengte-as door het zwaartepunt liggen. Omdat dit links en rechts kan verschillen wordt de gemiddelde y-coördinaat van het verste punt links en rechts genomen. De waarde ligt tussen 1 en -1, groter dan 0 betekent dat de grootste breedte boven het midden ligt, kleiner dan 0 betekent onder het midden.

Cirkelvormigheid kan op verschillende manieren uitgedrukt worden. Hier wordt het isoperimetrisch quotiënt gebruikt:

$$\text{cirkelvormigheid} = \frac{4\pi O}{P^2} \tag{5.1}$$

Met O de oppervlakte en P de omtrek van de bladschijf. Deze maat wordt ook wel vormfactor genoemd. Dit getal drukt de oppervlakte van het blad uit in verhouding tot de oppervlakte van een cirkel met dezelfde omtrek. Voor een cirkel is deze waarde 1, voor een lijn convergeert de waarde naar 0. Dus hoe langer en smaller het blad, hoe lager de waarde. Deze cirkelvormigheid is dimensieloos en schaalonafhankelijk.

De **soliditeit** geeft aan welke fractie van de pixels in het convex omhulsel (*convex hull*, Figuur 5.1(b)) ook tot het blad behoren. Deze wordt berekend door de oppervlakte van de bladschijf te delen door de oppervlakte van het convex omhulsel. Het convex omhulsel is de vorm die men zou bekomen wanneer een touw rond het blad wordt gespannen.

Om de **asymmetrie** van het blad uit te drukken worden de linker en rechter helft met elkaar vergeleken. Er worden 3 ratio's berekend: voor oppervlakte, omtrek en breedte. Het grootste getal wordt telkens door het kleinste gedeeld zodat de ratio's steeds groter dan 1 zijn.

Ook op basis van de bladsteel worden een aantal features berekend. Ten eerste wordt de **verhouding tussen de oppervlakte van de steel en het blad** berekend :

$$\text{percentSteel} = \frac{O_{\text{Steel}}}{O_{\text{Blad}}} * 100,$$

met O_{Steel} en O_{Blad} de oppervlakte van respectievelijk de steel en het blad met steel.

Ten tweede wordt de **cirkelvormigheid** van de steel bepaald via Vgl. (5.1). Door de schaalinvariantie kan deze maat geen onderscheid maken tussen brede, lange stelen en smalle, korte stelen. Daarom is het waarschijnlijk nuttig om de cirkelvormigheid met het percentage steel te vermenigvuldigen.

5.1.2 Statistische features

Er worden 8 schaal-, rotatie- en translatie-invariante momentenfeatures berekend, de 7 van Hu en een 8ste extra zoals beschreven in Sectie 2.4.

Een andere feature waarbij momenten gebruikt worden is de **excentriciteit**. Het gaat hier om de excentriciteit van de ellips met hetzelfde tweede orde moment als de bladschijf, zie Figuur 5.1(c). De excentriciteit is de ratio van de afstand tussen de brandpunten van de ellips en de lengte van de hoofdas (Burger en Burge, 2008).

$$\text{excentriciteit} = \sqrt{1 - \frac{b^2}{a^2}},$$

met a de lange straal en b korte straal van de ellips. De waarde ligt tussen 0 en 1, waarbij 0 de waarde is van een cirkel en 1 de waarde voor een lijn.

5.2 Contourfeatures

In deze sectie wordt op een aantal manieren getracht de bladrand of de contour te beschrijven.

5.2.1 Histogram van de kromming

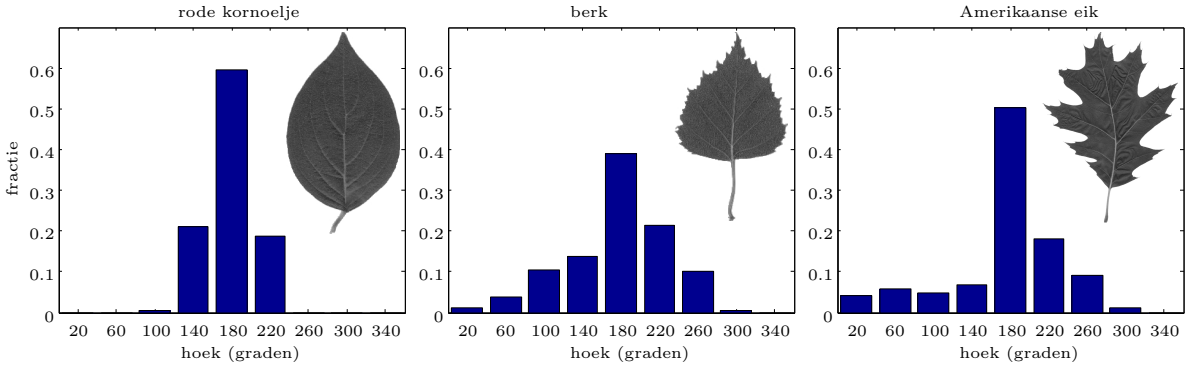
Deze feature is gebaseerd op het idee dat bij gave bladeren de kromming van de contour vrij constant is terwijl bij gekartelde, getande of gezaagde bladeren de richting zeer variabel is wanneer men de contour doorloopt. Voor de berekening wordt gebruik gemaakt van de contour van de bladschijf voorgesteld als een $K \times 2$ matrix C (zie Sectie 1.4.3).

De kromming kan geëvalueerd worden als de richtingscoëfficiënt van de raaklijn aan de contour in een bepaald punt ervan. De richtingscoëfficiënten zullen benaderd worden door de richtingscoëfficiënten te van de lijnstukken die twee contourpixels die op een bepaalde afstand van elkaar liggen verbinden. De verandering van de kromming kan uitgedrukt worden worden door de hoek tussen de raaklijnen in twee opeenvolgende punten. Deze wordt afgeleid uit de richtingscoëfficiënten. Bijgevolg moet het interval gekozen worden waarover de richtingscoëfficiënten berekend worden. Een klein interval zal de kromming op kleine schaal uitdrukken. Een groter interval zal daarentegen eerder de lobben en insnijdingen in een blad modelleren. Er zal gewerkt worden met een kleine intervallengte (a_1) en een grotere intervallengte (a_2). In de veronderstelling dat grote bladeren grotere tanden hebben dan kleine bladeren worden de intervalgroottes bepaald in functie van de oppervlakte van het blad:

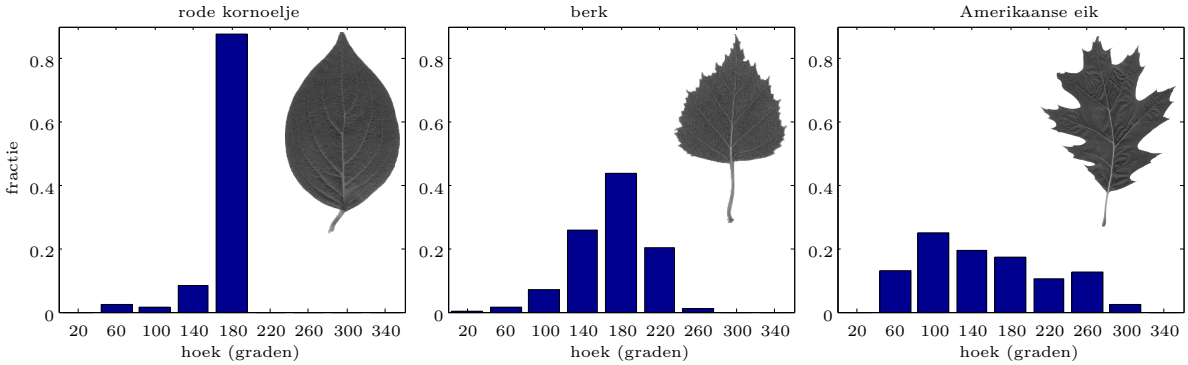
$$a_1 = O/10000,$$
$$a_2 = O/1000,$$

met O de oppervlakte uitgedrukt in aantal pixels (bij 176 PPI).

Er wordt gekozen om steeds de hoek langs de binnenkant van het blad te berekenen. Deze hoeken worden dan ingedeeld volgens grootte in 9 intervallen van elk 40°, aldus worden histogrammen bekomen zoals te zien in Figuren 5.2 en 5.3. In de figuren is een voorbeeld van een



Figuur 5.2: Voorbeelden van histogrammen van de kromming van de bladrand met kromming op kleine schaal (a_1, $\pm$ over 4 pixels).



Figuur 5.3: Voorbeelden van histogrammen van de kromming van de bladrand met kromming op grote schaal (a_2, $\pm$ over 40 pixels)

gaaf, een getand en een gelobd blad weergegeven, de histogrammen zijn duidelijk verschillend. De 9 waarden van het histogram worden gestandaardiseerd door ze te delen door de som van alle balken en kunnen zo als features gebruikt worden in een model.

5.2.2 Fouriertransformatie van CCDC

De Fouriertransformatie van de CCDC wordt uitgevoerd zoals beschreven in de literatuurstudie (Sectie 2.5.2). De contour die hierbij gebruikt wordt is die van het blad zonder steel.

Er zijn twee manieren waarop een CCDC bepaald kan worden, met enigszins verschillend resultaat. Een eerste manier maakt gebruik van een hoek α (zie Figuur 2.7) die een volledige cirkel doorloopt, $\alpha \in [0, 2\pi[$, en waarbij op vaste intervallen de afstand tot de contour berekend wordt. Deze methode zal eigenaardige resultaten geven als ze wordt toegepast op gedeelde of samengestelde bladeren omdat er dan bij bepaalde hoeken meerdere snijpunten met de contour zijn. Een tweede methode is om achtereenvolgens de afstand tot elke pixel van de contour te berekenen. Hierbij zal een kleine fout ontstaan doordat de afstand tussen pixels niet constant is, voor 4-geconnecteerde pixels is die 1, voor schuine 8-geconnectreede buurpixels is de afstand $\sqrt{2}$. Dit kan in rekening gebracht worden door de CCDC te definiëren in functie van t, dus $CCDC(t)$, waarbij Δt de waarde 1 of $\sqrt{2}$ aanneemt afhankelijk van de connectiviteit.

De CCDC wordt hier bepaald door de centrum-contourafstand voor alle contourpixels te berekenen aan de hand van de contourmatrix $C = [(k_1, l_1); (k_2, l_2); ...; (k_K, l_K)]$ die beschreven werd in de notatiesectie (Sectie 1.4.3). De DFT-coëfficiënten van een vector x worden

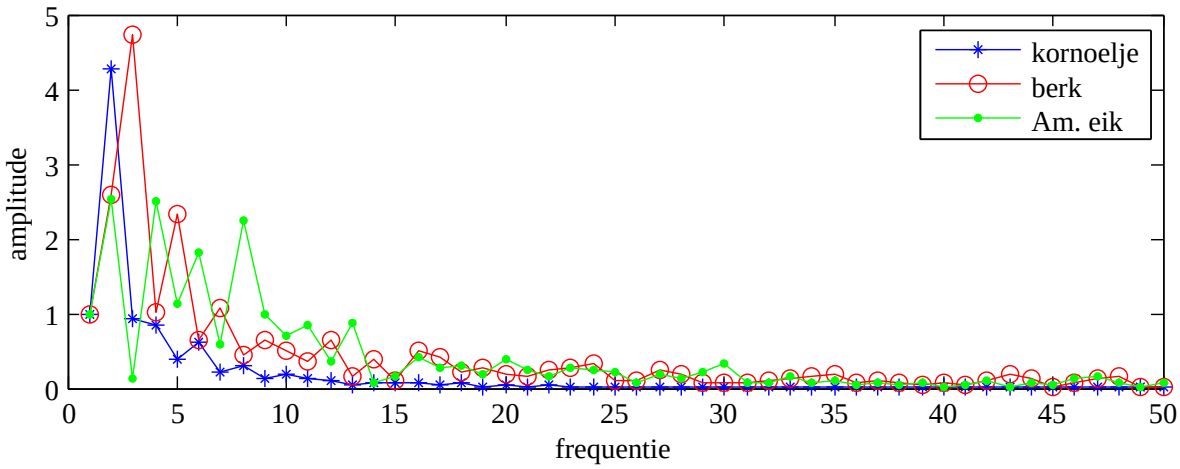
berekend met het *fast Fourier transform* (FFT) algoritme. Om invariantie van schaal en beginpunt te garanderen wordt enkel gebruik gemaakt van de amplitudes. De amplitudes worden vermenigvuldigd met een schaalfactor s. Typisch wordt als schaalfactor de lengte van de curve gebruikt, maar voor bladeren is dit niet ideaal omdat herschalingen de grilligheid van de contour en dus de relatieve lengte ervan beïnvloeden. Daarom werden kort een nog een aantal schaalfactoren getest die wel dezelfde amplitudes genereren voor herschaalde versies van eenzelfde blad:

1. De amplitudes worden genormaliseerd op de amplitude van de harmonische met frequentie 1 zodat deze steeds waarde 1 heeft dus $s = 1/apm_1$.
2. De amplitude van de harmonische met frequentie 0 drukt translatie uit en wordt in principe achterwege gelaten, maar in het geval van de CCDC is de translatie eerder maat voor de grootte, aldus kan deze als schaalfactor gebruikt worden.
3. De amplitudes worden gestandaardiseerd op de lengte van de bladschijf.

De eerste optie gaf iets betere resultaten bij een classificatietest, maar de verschillen waren klein. Volgende illustraties zijn op basis van optie 1.

In Figuur 5.4 zijn de amplitudes van de eerste 50 harmonischen weergegeven voor een blad van kornoelje, berk en Amerikaanse eik. Het gave blad van de kornoelje is duidelijk te herkennen aan de lage amplitudes bij hogere frequenties, het verschil tussen berk en Amerikaanse eik is minder goed te zien.

Uit scatterplots van de frequenties onderling werd het belang van de verschillende frequenties niet onmiddellijk duidelijk. Om een beter beeld te krijgen werd een principale componenten analyse uitgevoerd. De scatterplots van de eerste twee en de eerste en de derde principale componenten zijn weergegeven in Figuur 5.5. De eerste principale component $PC1$ verklaart 62,4% van de variantie, $PC2$ en $PC3$ verklaren respectievelijk 6,5% en 3,1%. Uit de figuur blijkt echter dat hoewel $PC1$ veel variantie verklaart, deze weinig informatie geeft over het verschil tussen de 3 klassen. $PC2$ is duidelijk veel nuttiger voor classificatie, ook $PC3$ kan gebruikt worden om Amerikaanse eik van de twee andere klassen te onderscheiden. Uit analyse van de samenstelling van de principale componenten blijkt dat alle frequenties ongeveer evenveel bijdragen aan $PC1$, deze component is dus vergelijkbaar met de som over alle frequenties.



Figuur 5.4: Discrete Fouriertransformatie van CCDC: Gestandaardiseerde amplitudes van de eerste 50 harmonischen voor een blad van kornoelje, berk en Amerikaanse eik

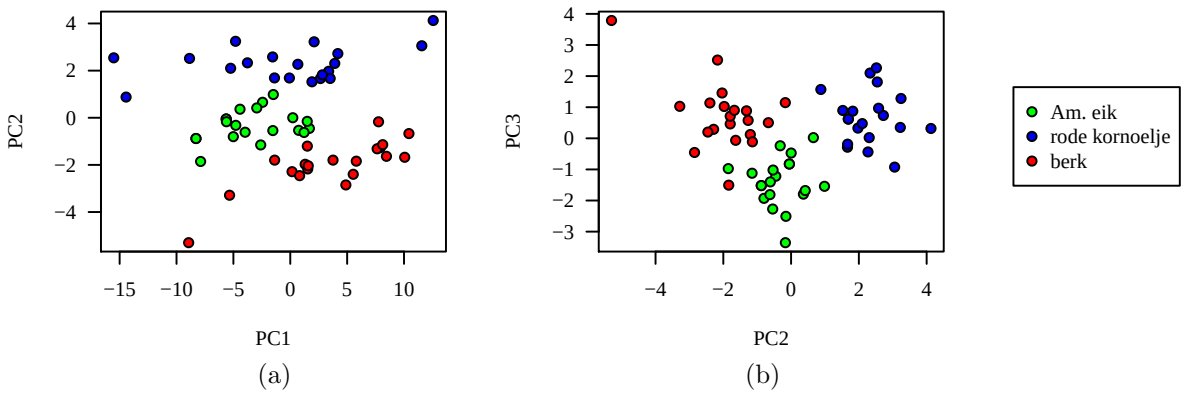
$PC2$ bestaat voornamelijk uit de amplitude van frequenties 1 tot 11, met uitzondering van frequentie 3, die de voornaamste component van $PC3$ is.

Een alternatief voor PCA is lineaire discriminantanalyse (LDA). Beide methoden zoeken naar lineaire combinaties van variabelen die de data zo goed mogelijk beschrijven. Daar waar PCA als doel heeft zo veel mogelijk variantie te verklaren (klassen worden niet in rekening gebracht) wordt in LDA gezocht naar combinaties die in staat zijn klassen van elkaar te onderscheiden. LDA modelleert dus de verschillen tussen klassen. Daarbij wordt aangenomen dat de onafhankelijke variabelen normaal verdeeld zijn (Hastie *et al.*, 2008).

Links in Figuur 5.6 is een scatterplot weergegeven met op de assen de eerste twee lineaire discriminanten $LD1$ en $LD2$. Uit deze figuur blijkt dat de drie klassen zeer goed onderscheiden kunnen worden op basis van de amplitude van de eerste 60 frequenties, maar door het hoge aantal variabelen is er *overfitting* gebeurd. Dit wordt bevestigd door de figuur rechts (Figuur 5.6(b)) waarin de helft van de data gebruikt werden om de LD's te bepalen en de andere helft om de figuur te maken. Hieruit blijkt dat het resultaat van LDA weinig beter is dan bv. een scatterplot van freqentie 2 t.o.v. frequentie 3.

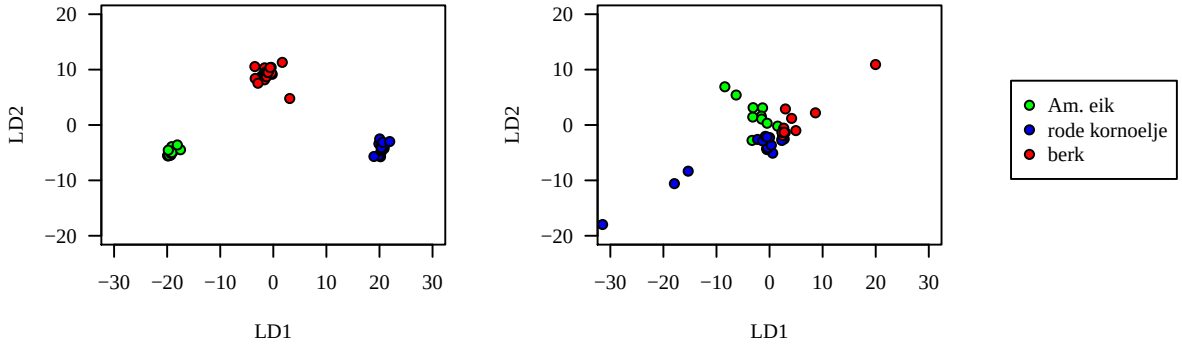
Het gebruik van lineaire combinaties zou later in de classificatiefase voor extra moeilijkheid zorgen bij het uitvoeren van kruisvalidatie. Dit komt doordat de lineaire combinaties steeds opnieuw moeten bepaald worden om onafhankelijkheid tussen test en trainingsdata te garanderen. Daarom wordt verkozen om met amplitudes en aantal sommen van amplitudes te werken. Uit de figuren werd immers afgeleid dat deze ongeveer evenveel informatie bevatten. Een ander argument om de te gebruiken frequenties zelf te bepalen, is dat de informatie die vervat zit in de lage freqenties reeds gevonden werd in de regiofeatures aangezien deze voornamelijk de algemene bladvorm beschrijven. De hogere frequenties bevatten weliswaar minder informatie, maar aangezien ze de details van de bladrand kenmerken kunnen ze een nuttige aanvulling zijn op de overige features.

De amplitudes die meegenomen worden naar de classificatiefase zijn die van frequentie 2 t.e.m. 11, deze keuze wordt gemaakt op basis van het resultaat van de PCA. De sommen die berekend worden zijn $\sum_{i=2}^{15} \text{amp}_i$, $\sum_{i=15}^{120} \text{amp}_i$, $\sum_{i=120}^{240} \text{amp}_i$ en $\sum_{i=240}^{480} \text{amp}_i$. De keuze van de sommen werd visueel gemaakt op basis van scatterplots. Er werden een aantal sommen geselecteerd die onderling weinig correlatie vertoonden.



(a) (b)

Figuur 5.5: Scatterplots van de eerste 3 principale componenten van de log(amplitudes) van de eerste 60 harmonischen uit de Fouriertransformatie van de CCDC. (a) PC1 vs. PC2, (b) PC3 vs. PC2.



(a) train (b) test

Figuur 5.6: Scatterplot van de eerste 2 lineaire discriminanten van de log(amplitudes) van de eerste 60 harmonischen uit de Fouriertransformatie van de CCDC: (a) voor traindata, (b) voor onafhankelijke testdata.

5.2.3 Fouriertransformatie van de complexe voorstelling van de contour

De berekeningen verlopen volledig analoog als bij de Fouriertransformatie van de CCDC. Dezelfde frequenties en sommen als in de vorige sectie worden weerhouden aangezien de grote gelijkenis tussen beide methoden doet verwachten dat dezelfde frequenties een belangrijke rol spelen. Een verschil is wel dat de amplitude bij frequentie 0 de translatie als de afstand tussen het centrum van het blad en de oorsprong uitdrukt en niet als de afstand tussen zwaartepunt en contour.

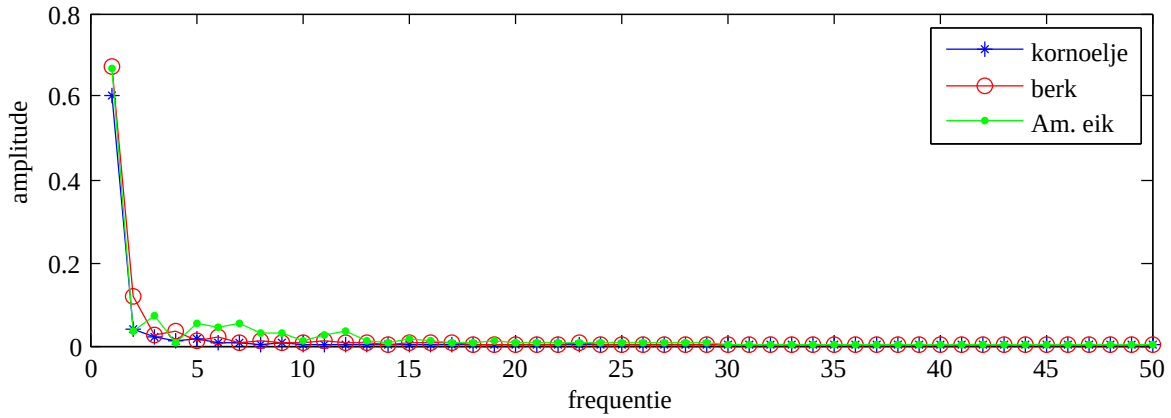
5.2.4 Elliptische Fouriertransformatie

Ook de elliptische Fouriertransformatie wordt uitgevoerd zoals beschreven in de literatuurstudie (Sectie 2.5.3). Een Matlab implementatie van de Elliptische Fouriertransformatie met normalisatie kan gedownload worden (Thomas, 2006). Net als in de vorige sectie wordt de contour gebruikt van het blad zonder steel en zal enkel met de amplitudes gewerkt worden.

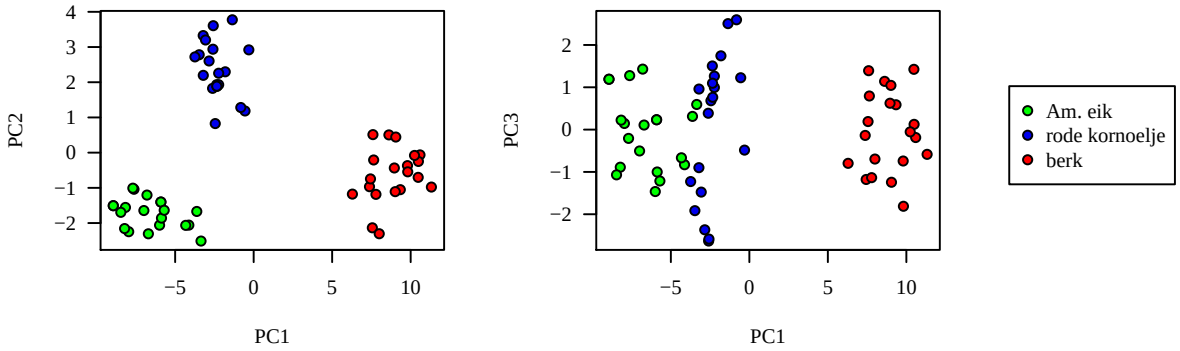
Figuren 5.7 en 5.8 zijn gelijkaardig aan Figuren 5.4 en 5.5 uit de vorige sectie. Figuur 5.7 toont de amplitudes van de eerste 50 harmonischen voor een blad van kornoelje, berk en Amerikaanse eik. Er valt opnieuw op dat de kornoelje lage amplitudes heeft en dat de Amerikaanse eik de hoogste amplitudes heeft bij lage frequenties. Voor Figuur 5.8 werd ook hier de logaritme van de amplitudes gebruikt omdat dit betere figuren oplevert. Uit de PCA blijkt dat de $PC1$, $PC2$ en $PC3$ respectievelijk 74,9%, 5,8% en 2,4% van de variantie verklaren. Uit de opbouw van de principale componenten blijkt net als bij de Fouriertransformatie van de CCDC dat $PC1$ bij benadering de som is van alle amplitudes, $PC1$ is hier echter wel nuttig om klassen te onderscheiden. Ook $PC2$ en $PC3$ bestaan opnieuw voornamelijk uit freqenties 1 t.e.m. 11. Uit scatterplots bleek dat het gebruik van sommen ook bij de elliptische Fouriertransformatie goede resultaten oplevert. Daarom worden dezelfde frequenties en sommen als in de vorige sectie meegenomen naar de classificatiefase.

5.2.5 Bladrand effenen

De invloed van effening is anders dan de vorige features in die zin dat opnieuw gebruik gemaakt wordt van het binaire beeld en dus niet enkel van de contour. Er bestaan verschillende filters waarmee de bladrand geëffend kan worden. Een voorbeeld hiervan werd reeds gezien in Sectie 2.3.3, namelijk morfologische opening. Hier wordt gekozen om het binaire beeld te effenen met een uitmiddelingsfilter. Deze filter vervangt elke pixel door het gemiddelde van de



Figuur 5.7: Elliptische Fouriertransformatie van contour: Gestandaardiseerde amplitudes van de eerste 50 harmonischen voor een blad van kornoelje, berk en Amerikaanse eik



Figuur 5.8: Scatterplot van de eerste 3 principale componenten van de log(amplitudes) van de eerste 60 harmonischen uit de elliptische Fouriertransformatie

omringende pixels, er ontstaan dus grijswaarden. De grijswaarden worden dan weer als wit en zwart geklasseerd met een drempel bij 0,5. Als masker worden vierkanten 10x10 en 100x100 pixels gebruikt. In Figuur 5.9 is te zien hoe na het filteren de bladrand gaver geworden is. Uit de figuur blijkt dat er enerzijds bladpunten uit de contour steken en anderzijds stukken achtergrond zich binnen het contour bevinden. De oppervlakte van beide wordt apart berekend. Deze worden als twee verschillende features gebruikt na normalisatie door te delen door de totale oppervlakte van de originele bladschijf.

5.3 Evaluatie van de features

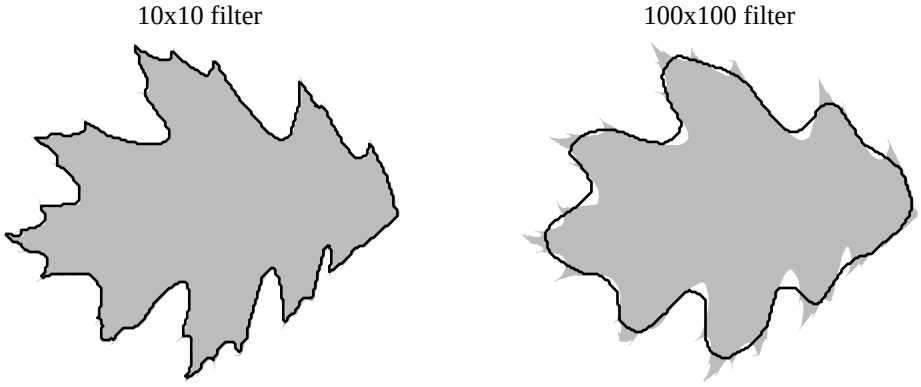
De geschiktheid van de verschillende features wordt getest met behulp van *random forest* modellen. Deze modellen worden verkozen omdat het trainen slechts enkele tellen duurt waardoor ze gemakkelijk in gebruik zijn. Bovendien kan bij *random forests* de performantie onmiddellijk geëvalueerd worden op basis van de OOB-foutenschatting (Sectie 3.3.2).

Bij *random forests* modellen kan het belang van de variabelen beoordeeld worden op basis van het aantal keren dat de variabele gekozen werd als beste uit de m random geselecteerde mogelijkheden. Voor meer precieze resultaten kan voor elke node in elke boom de daling van de Gini-onzuiverheidsindex berekend worden als het verschil in Gini-index tussen moedercel en dochtercellen. De Gini-index wordt gedefinieerd als (Breiman en Cutler, 2004):

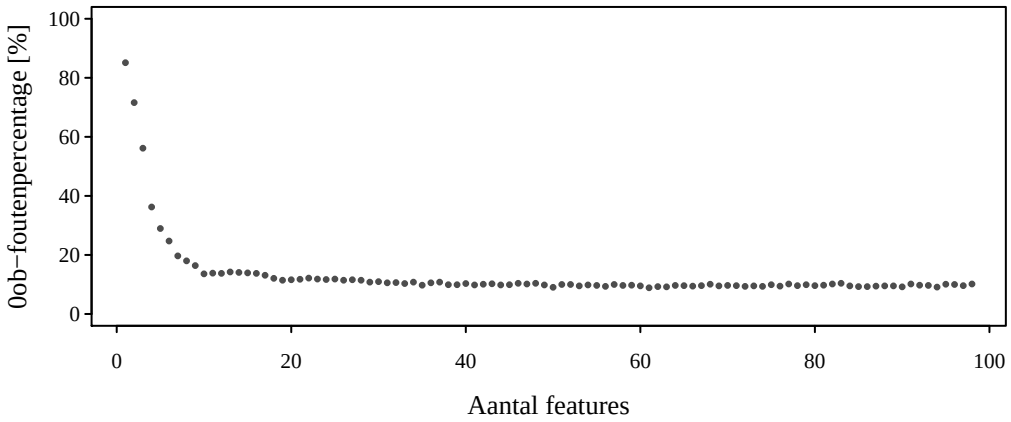
$$G = 1 - \sum_{q=1}^{Q} f_q^2,$$

met f_q de fractie instanties van de q-de klasse. Voor elke variabele kan aldus de gemiddelde daling van de Gini-index bepaald worden op basis van alle splitsingen waarvoor deze variabele gebruikt wordt. Deze waarde werd berekend voor alle features in de dataset. De volgende features gaven de hoogste waarden: het percentage steel, de lengte-breedteverhouding, de excentriciteit, de soliditeit, de eerste twee momenten van Hu, de eerste drie somtermen van de verschillende op Fouriertransformatie gebaseerde technieken en de '100x100'-effeningscoëfficiënten. Deze variabelen bevatten dus de belangrijkste informatie om soorten te onderscheiden.

In Figuur 5.10 is het OOB-foutenpercentage weergegeven in functie van het aantal features. De features werden voor deze figuur gerangschikt volgens de gemiddelde daling van de Gini-



Figuur 5.9: Effect van een 10x10 en een 100x100 uitmiddelingsfilter. Grijs is het originele blad, de zwarte lijn is de contour na het filteren.



Figuur 5.10: OOB-foutenpercentage in functie van het aantal features. De features werden van goed naar slecht gerangschikt volgens de gemiddelde daling van de Gini-index. OOB's werden bepaald via RF-modellen met 2000 bomen.

index. In de figuur is te zien dat het foutenpercentage sterk daalt (tot 13,60%) op basis van de eerste 10 features, daarna vlakt de OOB af naar 10%. De laatste 50 features hebben praktisch geen invloed meer.

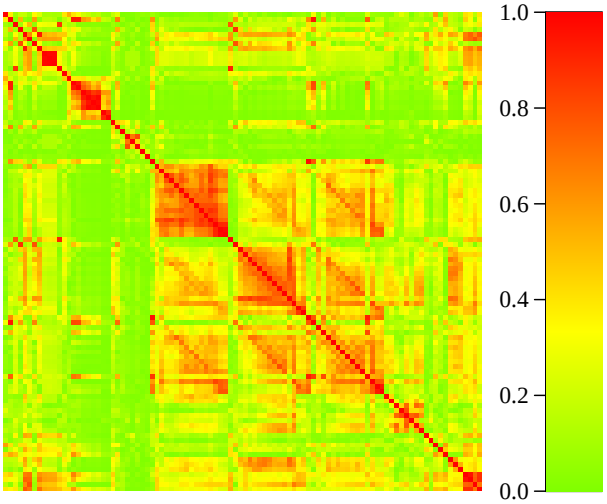
De features die berekend werden, kunnen ingedeeld worden in een aantal groepen zoals weergegeven in Tabel 5.1. Om het nut te evalueren van elke groep werden telkens twee RF-modellen opgebouwd: één waarbij enkel een subset van de features (deze uit de groep) gebruikt werden en één op basis van een subset met alle andere features. De OOB van het eerste model geeft een beeld van de hoeveelheid informatie die de features bevatten. Uit de tabel blijkt dat alle features een zekere hoeveelheid informatie geven aangezien alle OOB-waarden beduidend lager zijn dan de fout die men bekomt met willekeurig gokken: $(1 - 1/41) * 100\% = 97{,}56\%$. Uit de waarden blijkt dat voornamelijk de geometrische features en de Fourierfeatures (in het bijzonder deze van de EFT) alleen in staat zijn om tot een accurate classificatie te leiden. De informatie die verschillende features bevatten, kan echter dezelfde zijn waardoor een deel van de features overbodig zijn. Een voorbeeld hiervan is de correlatie tussen de amplitude van de eerste harmonische uit de elliptische Fouriertransformatie en de lengte-breedteverhouding.

Uit de OOB van het tweede model kan afgeleid worden hoeveel de features bijdragen aan de classificatie, of in hoeverre ze vervangbaar zijn. Weglaten van belangrijke features zal namelijk een stijging van de OOB veroorzaken, terwijl als features geen unieke informatie bevatten de OOB gelijk blijft. De lage waarden in de kolom 'RF op basis van complement subset' wijzen erop dat er een overvloed aan features berekend werd. Slechts de karakteristieken van de bladsteel bevatten een noemenswaardige hoeveelheid unieke informatie. Dit is te verklaren doordat de andere features op basis van de bladschijf berekend zijn. Een belangrijke opmerking is dat het classificeren bijna net zo goed gaat met als zonder de oppervlakte, het zou dus mogelijk zijn om de schaal volledig te elimineren zodat beelden ongeacht de resolutie geclassificeerd kunnen worden.

Om een beter beeld te krijgen van welke features op elkaar gelijken werd de correlatie berekend tussen alle paren van features, de absolute waarde van de correlaties zijn weergegeven in Figuur 5.11, de volgorde is zoals in Tabel 5.1. Voor bepaalde modellen of omwille van computationele beperkingen kan het wenselijk zijn om het aantal features te reduceren, maar voor de modellen die in deze thesis gebruikt worden is dit niet nodig. Tenzij anders vermeld zijn alle verdere resultaten berekend op basis van alle 98 features.

Tabel 5.1: Evaluatie van de groepen van features a.d.h.v. RF-modellen. De kolom 'RF op basis van subset' geeft het foutenpercentage wanneer slechts de features aangegeven in de eerste kolom gebruikt worden. De kolom 'RF op basis van complement subset' geeft het foutenpercentage van een model waarin de features uit de eerste kolom geëlimineerd werden.

Groep	Aantal features	OOB-foutenpercentage [%]	
		RF op basis van complement subset	RF op basis van subset
Oppervlakte	1	90,00	9,76
Geometrische features (excl. Hu)	13	20,64	**11,52**
Geometrische features steel	2	73,04	**11,12**
Hu's momentenfeatures	16	34,08	9,68
Hu's momentenfeatures: oppervlakte	8	57,92	9,92
Hu's momentenfeatures: contour	8	45,84	9,44
Features op basis van Fourieranalyse	48	**17,04**	10,88
DFT van CCDC	16	27,44	9,76
EFT	16	**20,32**	10,64
DFT van complexe voorstelling	16	23,52	9,76
Histogrammen van kromming	16	33,12	10,88
kleine schaal	8	50,72	10,32
grote schaal	8	61,44	9,76
Effening	4	51,25	9,68
Alle features	98	9,76	97,56



Figuur 5.11: *Heatmap* van de correlaties tussen de 98 features (absolute waarden).

Hoofdstuk 6

Opbouw classificatiemodellen

In dit hoofdstuk worden verschillende classificatiemodellen opgebouwd en getest. Er wordt gestart met een algemene sectie over performantie-analyse omdat dit voor de validatie van alle modellen gelijk is. De opbouw van classificatiemodellen wordt gedaan met behulp van het statistische softwarepakket R (R Development Core Team, 2010).

6.1 Performantie-analyse

Om een betrouwbare schatting van de performantie van een model te krijgen dient het model getest te worden op data die niet gebruikt werden voor het trainen van het model. Daarnaast kunnen aparte data nodig zijn voor het schatten van parameters. Bijgevolg moet de dataset worden opgesplitst in drie delen (Hastie *et al.*, 2008):

- trainingsdata: om het model te fitten,
- validatiedata: voor modelselectie en optimalisatie van parameters,
- testdata: evaluatie van de performantie van het geselecteerde model.

Aangezien het aantal instanties per klasse eerder beperkt is zal kruisvalidering uitgevoerd worden. Daarvoor wordt de dataset opgesplitst in n_F gelijke delen. Elk deel wordt dan beurtelings als testset gebruikt waarbij het model getraind wordt op de overige delen. Op die manier wordt een meer betrouwbare schatting van de generalisatiefout bepaald dan wanneer men slechts éénmalig op een kleinere testset zou testen.

Er zijn niet evenveel beelden van elke soort. De instanties in de dataset zijn dus niet gelijk verdeeld over de verschillende klassen, toch is het de bedoeling om alle soorten als even belangrijk te beschouwen. Dit kan op drie niveaus in rekening worden gebracht:

- bij de verdeling in *folds* voor kruisvalidering,
- bij de training van het model,
- bij de berekening van de accuraatheid (gewogen accuraatheid).

Voor de kruisvalidatie worden de data verdeeld in n_F onafhankelijke subsets. De instanties van elke soort worden gelijk verdeeld over de verschillende subsets (gestratificeerde kruisvalidatie). Indien het aantal instanties van een bepaalde soort geen veelvoud is van n_F worden de resterende instanties toegevoegd aan een random subset (max. één instantie extra per subset). Op deze wijze worden subsets bekomen die de verdeling van de volledige dataset reflecteren.

Bij het berekenen van de accuraatheid van de predicties kan opnieuw rekening gehouden worden met de ongelijke verdeling van de instanties in de klassen, met name door eerst de accuraatheid van elke klasse te berekenen en dan het gemiddelde te nemen:

$$Acc = 1 - \frac{1}{Q} \sum_{q=1}^{Q} \mathrm{E}[\ L(y, \hat{y}(\mathbf{x})) \mid y = c_q\ ],$$

met y de echte klasse, $\hat{y}(\mathbf{x})$ de voorspelde klasse en $L(y, \hat{y}(\mathbf{x}))$ een verliesfunctie. Om accuraateid te berekenen wordt de 0/1-verliesfunctie gebruikt (Hastie *et al.*, 2008):

$$L(y, \hat{y}(\mathbf{x})) = \begin{cases} 1 & , y \neq \hat{y}(\mathbf{x}) \\ 0 & , y = \hat{y}(\mathbf{x}). \end{cases} \qquad \text{(0/1-verliesfunctie)}$$

6.2 Logistische regressie

Er werd een multiklasse logistisch regressiemodel opgebouwd, dit werd gedaan met het algoritme *glmnet* in R (Friedman *et al.*, 2010). *Glmnet* voert L_1-regularisatie uit. De regularisatieparameter λ wordt door deze implementatie voor alle deelmodellen gelijk gesteld, de ene λ werd getuned. De accuraatheid van het model bedroeg 86,72%.

6.3 Support vector machines

Voor het opbouwen van SVM modellen wordt gebruik gemaakt van de implementatie van Dimitriadou *et al.* (2010). Een aantal kernels die typisch gebruikt worden zijn de volgende:

- Lineaire kernel: $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$
- Polynomiale kernel: $(\gamma\ \langle \mathbf{x}_1, \mathbf{x}_2 \rangle)^m$
- RBF kernel: $\exp(-\gamma\ \|\mathbf{x}_1 - \mathbf{x}_2\|^2)$
- Sigmoidale kernel: $\tanh(\gamma\ \langle \mathbf{x}_1, \mathbf{x}_2 \rangle)$

Voor elk van deze kernels werden de kernelparameters en een kostparameter die instaat voor regularisatie getuned. De trainingsfout bij de getunede parameters is weergegeven in Tabel 6.1. Het model met de lineaire kernel gaf het beste resultaat. Voor dat model werd de performantie berekend aan de hand van 10-voudige kruisvalidatie. Daarbij werden in elke *fold* de parameters opnieuw geoptimaliseerd. De resulterende accuraatheid bedroeg 89,69%.

6.4 Random forests

Voor de opbouw van een *random forest* wordt gebruikt gemaakt van het pakket *randomForest* in R (Liaw en Wiener, 2002). Dit pakket bevat een implementatie van het *random forest* algoritme zoals oorspronkelijk beschreven door Breiman (2001). Er worden 500 bomen per *forest* gebruikt. Dit blijkt ruim voldoende om stabiele resultaten te geven. De standaardwaarde voor m (aantal willekeurig geselecteerde variabelen per split) bij classificatie is $\sqrt{p} \cong 10$, met $p = 98$ het aantal variabelen. Deze standaardwaarde levert een accuraatheid op van 89,19%.

Tabel 6.1: Accuraatheid en optimale parameters van verschillende kernels bij SVM's.

Model	Trainingserror [%]	Parameters		
		kost	γ	m
lineaire kernel	**8,84**	1,5	-	-
polynomiale kernel	10,08	0,005	2	2
RBF-kernel	12,15	5	0,01	-
sigmoidale kernel	9,44	400	0,0005	-

m kan echter getuned worden. De optimale waarde voor m wordt bepaald via interne 3-voudige kruisvalidatie op de trainingsdata maar blijkt slechts een geringe invloed te hebben. De accuraatheid bekomen door 10-voudige kruisvalidatie van een random forest met tuning van m was 90,06%. Als optimale waarde van m voor de volledige dataset werd 22 gevonden. Bij de trekking van de *bootstrap* instanties waarop de bomen gefit worden, kan men ervoor zorgen dat elke *bootstrap* dataset evenveel instanties van elke soort bevat. Dit bleek evenwel de performantie niet te verhogen, want deze bedroeg 88,74% (d.i. zonder tuning van m).

6.5 Boosting

In de literatuurstudie werd een multiklasse *boosting*-algoritme aangehaald (Vgl. (3.8)). Daarvan werd echter geen implementatie gevonden. Dus wordt het multiklasse probleem opgesplitst in binaire deelproblemen. Er wordt gekozen voor één-tegen-allen modellen (zie Sectie 3.1). Dit is namelijk computationeel efficiënter dan één-tegen-één modellen, aangezien slechts Q één-tegen-allen modellen nodig zijn tegenover $\frac{Q}{2}(Q+1)$ één-tegen-één modellen (het aantal instanties in elk model is daarentegen wel groter).

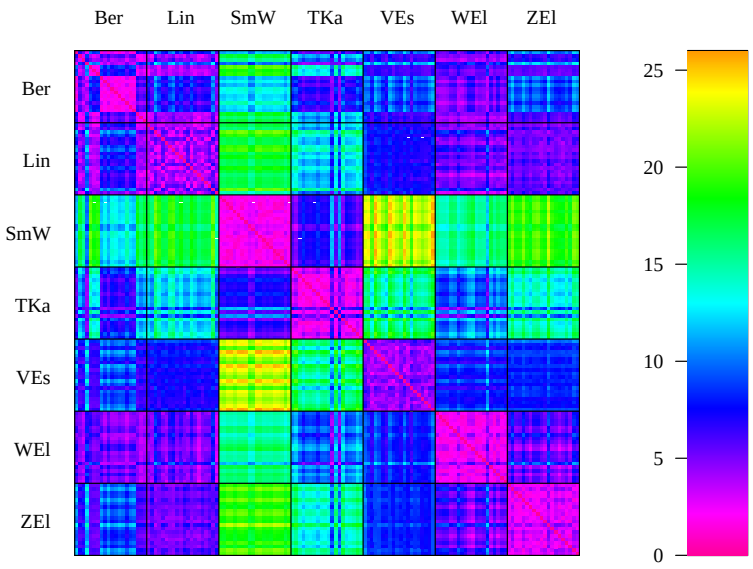
Eerst worden Q dummy responsvariabelen aangemaakt: één voor elke klasse. Deze dummy variabelen hebben waarde 1 voor instanties die tot de klasse behoren, 0 voor alle andere instanties. Voor elke dummy responsvariabele wordt dan een *boosting*-model bepaald dat bestaat uit 500 submodellen. Een algoritme uit het R-pakket *ada* wordt gebruikt (Culp *et al.*, 2010). Als deelmodellen worden *stumps* gebruikt, dit zijn classificatiebomen die slechts uit één enkele splitsing bestaan. *Boosting* met *stumps* levert over het algemeen het beste resultaat op. De verliesfunctie die gebruikt wordt is de exponentiële.

Met het binaire *boosting*-algoritme wordt de probabiliteit van elke klasse bepaald. Om een finale voorspelling te selecteren uit de Q binaire deelmodellen wordt de klasse met de hoogste probabiliteit gekozen. De accuraatheid van het boosting algoritme bedraagt 87,27%.

6.6 Kernel logistische regressie

In de literatuurstudie werd aangehaald dat het logistische regressiemodel net als SVM's gekerneliseerd kan worden. Een dergelijk KLR model kan gebruikt worden om classificatie met een kernel afgeleid van een afstandsmaat uit te voeren. Om de kernelmatrix te berekenen wordt gebruik gemaakt van Baddeley's Δ-metriek (Vgl. (3.11)). Daarbij wordt $n = 2$ genomen, de regularisatiefunctie w wordt niet gebruikt. Gilleland *et al.* (2008), die de Δ-metriek gebruikt in ruimtelijke weersvoorpellingen stelde vast dat de keuze van de parameter c (die de functie w bepaald) weinig invloed heeft op de voorspellingen. Figuur 6.1 geeft een beeld van de spreiding van de waarden voor de Δ-metriek. Uit deze figuur blijkt dat bladeren van dezelfde soort gelijkenis vertonen volgens de afstandsmaat. Let wel dat voor deze figuur 7 soorten geselecteerd werden die visueel sterk verschillen en dat enkel succesvol georiënteerde bladeren gebruikt werden. In de volledige matrix zijn er meer onregelmatigheden en is een minder uitgesproken blokkenpatroon aanwezig.

De afstandsmatrix wordt omgezet naar een RBF-kernelmatrix via Vergelijking (3.12) en positief definiet gemaakt door het algoritme van Higham (Schaefer *et al.*, 2010). De resulterende kernel wordt gebruikt in een KLR model dat bestaat uit Q één-tegen-allen deelmodellen. De parameter σ van de RBF kernel werd vrij ruw getuned: als optimale waarde werd $\sigma = 5$ gevonden. In Figuur 6.2 is de kernelmatrix met $\sigma = 5$ weergegeven. De KLR implementatie maakt gebruik van L_2-regularisatie (L_2-norm= $\sum |\beta|^2$). Daarom werd tegelijk met σ eveneens de regularisatieparameter λ getuned (wegens de lange rekentijden werd het aantal



Figuur 6.1: Baddeley's Δ-metriek voor telkens 20 bladeren van 7 soorten.

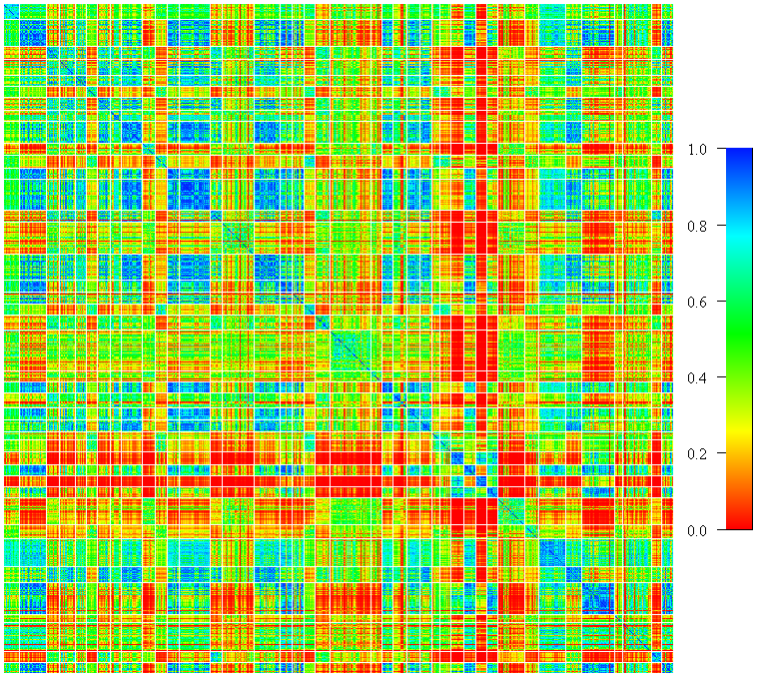
te testen parameters ook hier beperkt). λ werd op 2 manieren getuned: variabel (voor elk van de Q één-tegen-allen deelmodellen werd een aparte λ_q getuned) en constant (λ gelijk voor alle deelmodellen). Het tunen van σ en de λ's gebeurde slechts éénmalig (opnieuw om lange rekentijden te vermijden) via 3-voudige kruisvalidatie op basis van de volledige dataset. De resultaten zijn weergegeven in Tabel 6.2. Tegen de verwachtingen in bleek het tunen van een λ_q voor elk binair soortmodel afzonderlijk slechtere resultaten op te leveren dan het tunen van één constante λ. Een mogelijke verklaring is dat er te weinig data beschikbaar zijn (gemiddeld slechts 20 positieve instanties) om de λ's voor elk model afzonderlijk te tunen. Verschillende uiteenlopende waarden geven namelijk dezelfde accuraatheid, vaak 100%) op de trainingsdata, het optimum wordt dan eerder bepaald door toeval dan door de onderliggende distributies, als de optimale waarde onbeslist was werd steeds de hoogste λ-waarde (het eenvoudigste model) gekozen. Het gebruik van één constante λ kan interessant zijn wanneer men verwacht dat de optimale *decision boundaries* van de verschillende soorten een gelijkaardige complexiteit vertonen ($\mathrm{E}[\lambda_1] = \mathrm{E}[\lambda_2] = ... = \mathrm{E}[\lambda_Q]$).

Het KLR model werd eveneens toegepast op de lineaire featurekernel $\mathcal{K}_{ft}(i,j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ (die het beste resultaat gaf bij SVM's). De accuraatheid van dit model (10-voudige kruisvalidatie, $\lambda = 0{,}05$) bedraagt 86,62% en stemt overeen met de accuraatheid van het gelijksoortige LR model uit Sectie 6.2.

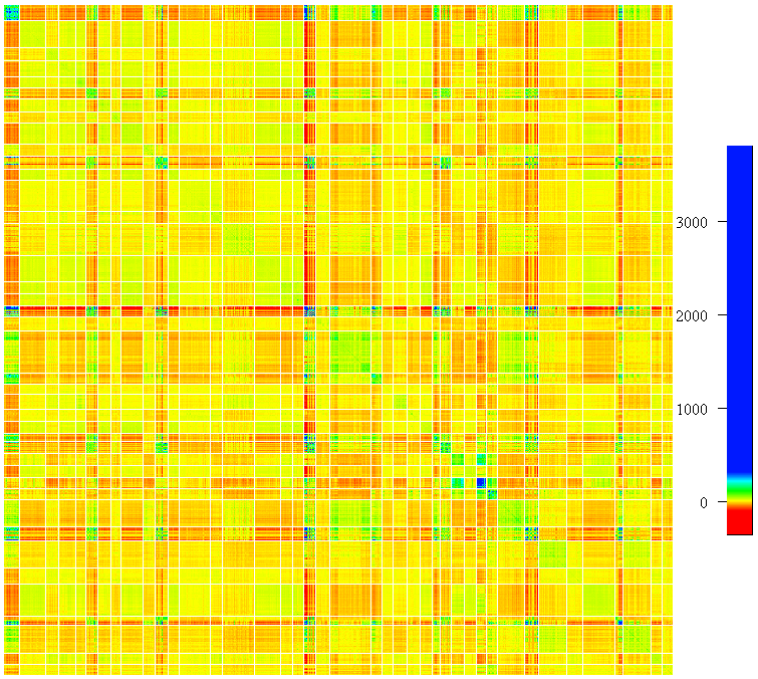
Het is mogelijk om nieuwe kernels te creëren uit bestaande kernels. Een aantal rekenregels daarvoor zijn gegeven in Bishop (2006). Het is onder meer toegestaan kernels te vermenigvuldigen met een scalair en verschillende kernels op te tellen. Met behulp van deze twee regels worden de lineaire featurekernel $\mathcal{K}_{ft}$ en de gaussiaanse Baddeleykernel $\mathcal{K}_b$ als volgt gecombineerd tot de combinatiekernel $\mathcal{K}_c$:

$$\mathcal{K}_c = \mathcal{K}_{ft} + 10\ \mathcal{K}_b,$$

$\mathcal{K}_b$ wordt eerst met 10 vermenigvuldigd omdat de waarden ervan ongeveer een factor 10 lager zijn dan die van $\mathcal{K}_b$, aldus krijgen beide kernels ongeveer evenveel gewicht. De beste accuraatheid die bereikt werd met 3-voudige kruisvalidatie met één getunede lambda is 89,16%.



Figuur 6.2: $\mathcal{K}_b$: RBF kernel met $\sigma = 5$ op basis van Baddeley's Δ-metriek. De verschillende soorten worden gescheiden door een witte lijn.



Figuur 6.3: $\mathcal{K}_{ft}$: lineaire kernel op basis van de features. De verschillende soorten worden gescheiden door een witte lijn.

Tabel 6.2: Accuraatheid van KLR met RBF-kernel op basis van Baddeley's Δ-metriek $\mathcal{K}_b$, lineaire featurekernel $\mathcal{K}_{ft}$ en combinatiekernel $\mathcal{K}_c$. Resultaten van 3-voudige of 10-voudige (*) kruisvalidatie.

Kernel	Accuraatheid [%]	
	λ variabel	λ constant
$\mathcal{K}_b$	73,18	74,35
$\mathcal{K}_{ft}$	77,02	86,62*
$\mathcal{K}_c$	88,35	**89,16**

6.7 Vergelijking van de modellen

Tabel 6.3 geeft een overzicht van de accuraatheden die bekomen werden met verschillende modellen. Het *Random forest* en de SVM met lineaire kernel geven de hoogste resultaten. Ook de accuraatheid van het KLR-model met de gecombineerde kernel is vergelijkbaar en de accuraatheid van deze methode kan waarschijnlijk nog verbeterd worden. De extra informatie die voortkomt uit de afstandsmaat is dus nuttig.

Tabel 6.3: Vergelijking van de accuraatheid van de verschillende modellen op basis van 10-voudige kruisvalidatie.

Model	Accuraatheid [%]
Logistische regressie	86,72
SVM met lineaire kernel	**89,69**
Random forests	**90,06**
Boosting	87,27
KLR met $\mathcal{K}_c$	89,16

Hoofdstuk 7

Geavanceerde modellen voor kennisextractie

In het vorige hoofdstuk werden de instanties ingedeeld in klassen. De technieken die daarbij gebruikt werden zijn eerder standaard. In dit hoofdstuk zullen een aantal meer geavanceerde technieken toegepast worden waarmee extra informatie uit de data gehaald kan worden door toevoegen van expertkennis en wordt getracht *novelty* detectie uit te voeren. Dergelijke technieken zijn over het algemeen nog in een experimentele fase.

7.1 Multilabelclassificatie

Bladeren kunnen ingedeeld worden op basis van een aantal kenmerken: samenstelling, bladvorm, bladrand. Daaruit werden een aantal binaire labels afgeleid die kunnen gebruikt worden voor multilabelclassificatie: samengesteld/enkelvoudig, gelobd/niet-niet-gelobd, langwerpig/rond en gaaf/gezaagd. Er werd als volgt te werk gegaan:

1. voor elk label wordt een binair classificatiemodel getraind,
2. voor alle instanties worden de labels (of de kans op een label) voorspeld a.d.h.v. de binaire classificatiemodellen,
3. een classificatiemodel wordt getraind op basis van de features en de voorspelde labels (of de kans op een label).

Voor de classificatiemodellen werden RF-modellen gebruikt zonder *tuning* van parameters. De accuraatheid gevonden door 10-voudige kruisvalidatie bedroeg slechts 86,13% via labels en 87,12% via kansen (Tabel 7.1). Beide zijn lager dan de accuraatheid van een enkel RF-model zonder multilabels (89,19% zonder tuning van m). De oorzaak hiervan is een overschatting van de accuraatheid van de voorspelde multilabels voor testsdata: de labels van trainingsdata worden voorspeld door een model dat op basis van de trainingsdata zelf opgebouwd werd. Ten gevolge van overfitting zijn de voorspellingen van de multilabels van de trainingsdata zeer accuraat en wordt het belang van de multilabelvariabelen in het finale model onterecht groot, dit met lagere accuraatheden tot gevolg.

Het probleem van de overfitte labels van de trainingsdata werd opgelost door deze labels te voorspellen aan de hand van 10-voudig gekruiste modellen binnen de trainingsdata. De accuraatheden die op deze manier bekomen werden zijn eveneens weergegeven in Tabel 7.1. Voornamelijk de methode waarbij de kansen op de vier multilabels als variabelen werden toegevoegd bleek de accuraatheid iets te kunnen verhogen. De meerwaarde is echter klein en er staat tegenover dat dit model veel trager is (training van 45 RF-deelmodellen).

7.2 Hiërarchische classificatie

Voor de hier gebruikte bladdataset is de taxonomische indeling van het plantenrijk beschikbaar. Deze heeft een boomstructuur en kan dus gebruikt worden als basis voor hiërarchische

Methode	Accuraatheid[%]
Zonder multilabels	89,19
Met multilabels, zonder correctie:	
op basis multilabels als factoren	86,13
op basis van kansen op multilabels	87,12
Met multilabels, met correctie voor overfitting:	
op basis multilabels als factoren	89,32
op basis van kansen op multilabels	**89,80**

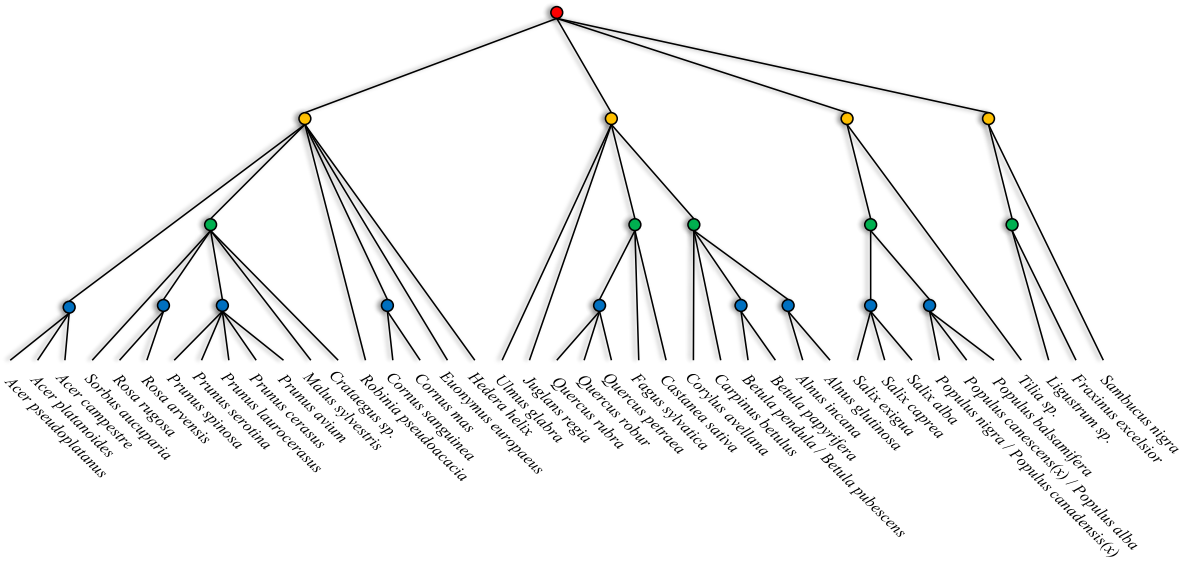
Tabel 7.1: Performantie van multilabelclassificatie op basis van RF-modellen

classificatie. De taxonomische indeling van het volledige plantenrijk is zeer complex, echter kan voor de aanwezige soorten de structuur vereenvoudigd worden tot vijf niveaus: 1 rijk, 4 superordes, 14 families, 25 genera en 41 soorten. In Figuur 7.1 is de taxonomische boomstructuur weergegeven.

In eerste instantie wordt een hiërarchisch model opgebouwd op basis van logistische regressiemodellen (met regularisatie van de L_1-norm). Dit wordt gedaan volgens de boomstructuur zoals voorgesteld in Figuur 7.1. In tabel 7.2(a), Opstelling 1 zijn de resultaten hiervan weergegeven, de kolom 'Δ acc.' geeft aan hoeveel fouten op elk niveau geïnduceerd worden. Er bleek dat reeds op het hoogste niveau, bij de classificatie in superordes 13,89% van de instanties foutief geklasseerd werden. Dit resultaat is opmerkelijk slecht, aangezien het lager is dan de accuraatheid van classificatie in 41 soorten ($\pm$90%). Vermoelijk is de samenstelling van de superordes te complex voor een lineair model zoals het logistisch regressiemodel, daarom werd eveneens een test gedaan met een random forests model voor het superordeniveau (Tabel 7.2(a), Opstelling 2). De resultaten blijven echter ontoereikend, waardoor besloten kan worden dat de superorde geen relevante indeling is met betrekking tot classificatie op basis van de beschikbare features. Deze bevinding komt niet onverwacht aangezien elke superorde bestaat uit een combinatie van zowel samengestelde als niet-samengestelde bladeren, gezaagde als gave bladeren, etc. Dit gaat in tegen de vereiste dat soorten die tot dezelfde superorde behoren meer met elkaar gemeen hebben dan met soorten uit andere superordes.

In Tabel 7.2(b) zijn de resultaten weergegeven van een gelijkaardig experiment met weglating van de superordes, dus met onmiddellijke indeling in het 2de niveau, de families. Dezelfde conclusie als voor superordes bleek echter te gelden voor de families. Een derde experiment (Tabel 7.2(c)) met enkel genera en soorten bleek wel de performantie iets te kunnen verhogen, indien voor de genera een RF-model gebruikt wordt. De stijging ie echter weinig significant.

Hiërarchische classificatie kan gebruikt worden als een techniek om performantie te verbeteren, echter indien de taxonomie gebruikt wordt laat dit ook toe om superordes, families, genera te voorspellen. Voor bepaalde doeleinden zal men belang hechten aan het juist classificeren van de voorgenoemde groepen. Ook voelt het natuurlijk aan om te zeggen dat zomereik (*Quercus robur*) voorspellen bij een blad van wintereik (*Quercus petraea*) minder ernstig is dan een andere niet nauw verwante soort voorspellen. Dit komt erop neer dat men niet de accuraatheid op soortniveau geminimaliseerd wil zien, maar een hiërarchische variant. Echter bleek uit de kolommen met Δ accuraatheid in Tabel 7.2 dat in het geval van bladclassificatie met taxonomie de fouten zich hoofdzakelijk in het eerste niveau situeren. Om de hiërarchische fout te minimaliseren is het de bedoeling dat de gemaakte fouten zich situeren in de onderste



Figuur 7.1: Dendrogram van de hiërarchie. Elke knoop stelt een classificatiemodel voor dat de data verdeelt in superordes (rood), families (oranje), genera (groen) of soorten (blauw). Groepen waarvan slecht één subgroep aanwezig is, hebben geen knoop aangezien er geen classificatie plaats vindt.

Tabel 7.2: Accuraatheid van de indeling in verschillende taxonomische niveaus op basis van 10-voudige kruisvalidatie. In Opstelling 1 worden steeds LR-modellen gebruikt, in Opstelling 2 wordt RF gebruikt voor het hoogste niveau. (a) Geeft de resulaten wanneer 4 niveaus gebruikt worden, (b) voor 3 niveaus en (c) voor 2 niveaus.

	Niveau	Opstelling 1			Opstelling 2		
		model	acc. [%]	Δ acc.[%]	model	acc. [%]	Δ acc.[%]
(a)	Superorde	LR	86,11	13,89	RF	93,06	6,94
	Familie	LR	80,56	5,55	LR	85,42	7,64
	Genus	LR	79,86	0,69	LR	84,72	0,69
	Soort	LR	77,78	2,08	LR	82,64	2,08
(b)	Familie	LR	82,64	17,36	RF	90,28	9,72
	Genus	LR	81,25	1,39	LR	88,89	1,39
	Soort	LR	78,47	2,78	LR	85,42	3,47
(c)	Genus	LR	90,90	9,10	RF	93,06	6,94
	Soort	LR	88,89	2,01	LR	**90,97**	2,09

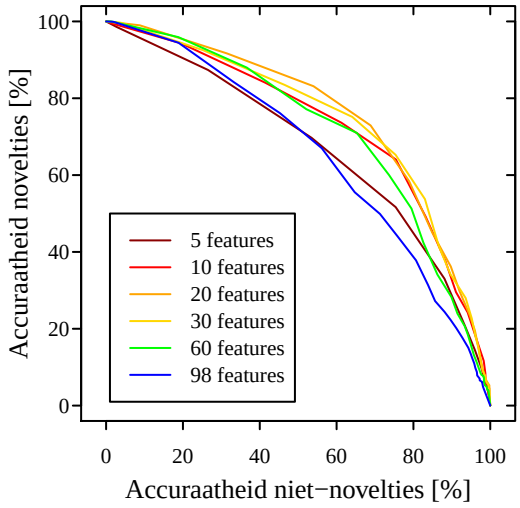
niveaus. Een berekening van de H-loss (Vergelijking 3.13 uit Sectie 3.5.2) toont aan dat voor het model (a), Opstelling 1 de gemiddelde H-loss 0,75 bedraagt, voor een enkelvoudig random forest model is de H-loss lager: 0,31. Voor deze berekeningen werden de kostencoëfficiënten uit Vergelijking 3.13 als volgt gekozen: $C_1 = 4, C_2 = 3, C_3 = 2, C_4 = 1$, per verkeerd niveau werd dus een gelijke strafmaat toegepast. Opmerkelijk is ook dat een enkelvoudig *random forests* model 95% van de genera juist voorspelt in vergelijking met een maximum van 93,06% juiste genera in Tabel 7.2.

7.3 Opsporing van nieuwe soorten

Classificatiemodellen maken gebruik van trainingsdata om soorten te herkennen, bijgevolg kunnen enkel soorten die aanwezig zijn in de databank correct gedetermineerd worden. Ongekende soorten zullen door de eerder besproken classificatiemodellen steeds bij een andere klasse ingedeeld worden. Om dit te vermijden kan men vooraf testen of een bepaalde instantie tot de gekende soorten behoort of niet (*novelty detection*). Een intuïtieve manier hiervoor bestaat erin om de Euclidische afstand in de featureruimte te berekenen tussen de nieuwe instantie en alle instanties in de dataset. Een instantie wordt als onbekend beschouwd als

$$\min_{\mathbf{x} \in T_{train}} \|\mathbf{x}_{nieuw} - \mathbf{x}\| > a,$$

met a een nader te bepalen drempelwaarde. Deze methode is gebaseerd op de redenering dat instanties van dezelfde soort dichter bij elkaar liggen dan instanties die tot een andere soort behoren. Echter is het doordat de distributies van de verschillende soorten overlappen onmogelijk om alle novelties correct te identificeren. Er moet een afweging gemaakt worden tussen niet-geïdentificeerde *novelties* (*false negative*, FN) en onterecht als *novelty* geïdentificeerde instanties (*false positive*, FP). Elke curve in Figuur 7.2 geeft de fracties FN en FP weer bij verschillende drempelwaarden. Bij het bepalen van de Euclidische afstand kunnen variabelen met veel ruis de betrouwbaarheid negatief beïnvloeden. Het is daarom aangewezen de afstand te bepalen op basis van een beperkt aantal robuuste features. De beste features werden geselecteerd op basis van de gemiddelde daling van de Gini-index bij *random forests* (Sectie 5.3). In Figuur 7.2 zijn de curves weergegeven op basis van verschillende aantallen features, een twintigtal features blijkt optimaal te zijn (hoogste oppervlakte onder de curve). De performantie van de *novelty detection* blijft echter zwak, indien men bijvoorbeeld 50% van de *novelties* wenst te vatten gaat dit gepaard met 85% van de normale instanties die verkeerdelijk als *novelty* worden beschouwden.



Figuur 7.2: Afweging van niet-geïdentificeerde novelties tegenover onterecht als novelty aangewezen instanties. De 'accuraatheid *novelties*' werd bepaald door elke soort afzonderlijk uit de dataset te verwijderen en de instanties als *novelties* te beschouwen (en vervolgens uit te middelen over de klassen). De 'accuraatheid niet-*novelties*' werd bepaald met 10-voudige kruisvalidatie op de volledige dataset.

Hoofdstuk 8

Evaluatie van het finale boomclassificatiesysteem

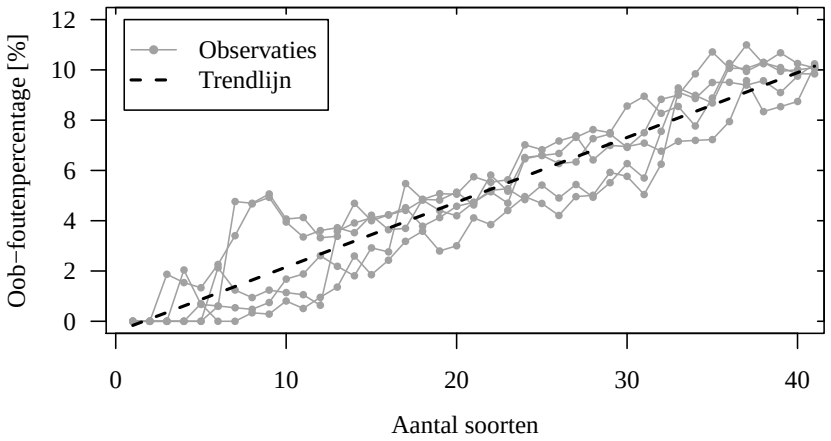
In dit laatste hoofdstuk voor het besluit wordt de performantie van een finaal model nader bestudeerd. Twee modellen bleken evenwaardig op vlak van accuraatheid: *random forests* en SVM. RF-modellen evalueren gaat echter sneller en eenvoudiger (via OOB), daarom zijn in dit hoofdstuk voornamelijk RF's gebruikt. Een volledige determinatie-algoritme op basis van een RF-model werd verwerkt tot een *standalone* applicatie met een GUI (zie Bijlage)

8.1 Performantie in functie van het aantal soorten

Om een idee te krijgen van hoe de performantie evolueert naarmate een model meer soorten kan voorspellen werd een experiment uitgevoerd aan de hand van *random forests*. In Figuur 8.1 zijn vijf reeksen van geobserveerde OOB-foutenpercentages weergegeven. Deze werden bekomen door te vertrekken vanuit één enkele willekeurige soort en telkens een nieuwe willekeurig geselecteerde soort aan het model toe te voegen. De observaties vertonen een lineair stijgende trend. Indien deze trend doorgetrokken wordt naar 100 soorten (d.i. een realistische schatting voor het totaal aantal soorten in Vlaanderen) zou men een accuraatheid in de grootte-orde van 75% verwachten. Echter zullen mogelijks naarmate het aantal soorten stijgt, nieuwe soorten steeds sterker op reeds aanwezige soorten gelijken en maakt de lineaire trend vermoedelijk een lichte overschatting van de performantie.
De gevonden accuraatheden kunnen worden vergeleken met de literatuur. Door het gebruik van verschillende datasets op vlak van aantal soorten, variatie in soorten, aantal instanties, type beelden, etc. is het moeilijk om resultaten te vergelijken. Uit Tabel 8.1 blijkt toch dat de performanties die teruggevonden werden in de literatuur in dezelfde grootte orde liggen als de eigen 90% bij 41 soorten.

Tabel 8.1: Accuraatheden uit de literatuur

Auteur (jaar)	Accuraatheid[%]	aantal soorten
Du *et al.* (2007)	93	20
Lee en Chen (2006)	82,33	60
Man *et al.* (2008)	92	24
Belhumeur *et al.* (2011)	70	191
Du *et al.* (2006)	92,3	25
Du *et al.* (2009)	92,25	20
Feng en Zhang (2009)	92,65	30
Wu *et al.* (2007b)	90	32

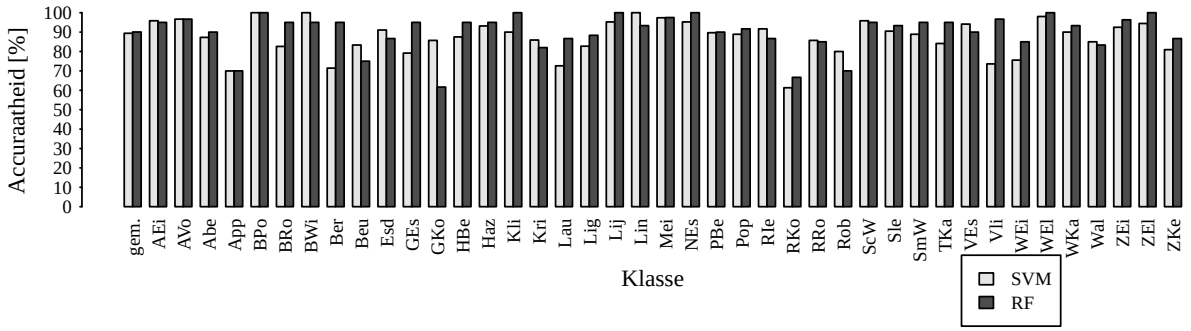


Figuur 8.1: Performantie in functie van het aantal soorten waarin geclassificeerd wordt. Er zijn 5 observaties weergegeven van de evolutie van het OOB-foutenpercentage, bepaald op basis van RF-modellen.

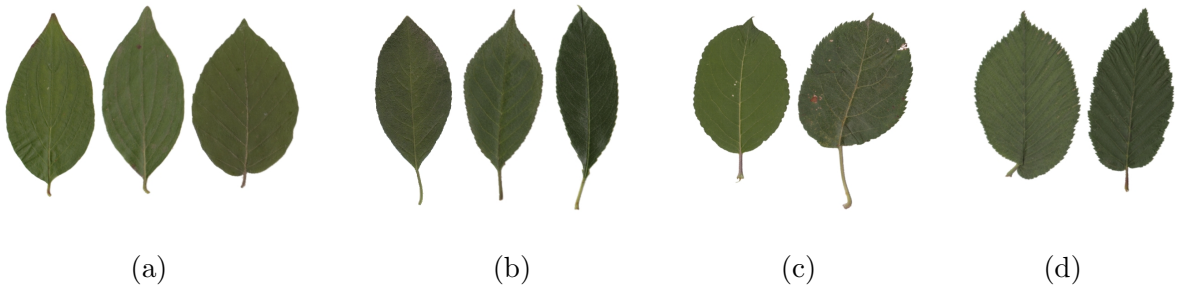
8.2 Performantie per soort

De performantie voor elke soort afzonderlijk wordt bestudeerd voor de twee modellen die de beste gemiddelde accuraatheden opleverden: SVM en RF. De accuraatheid per klasse is weergegeven in Figuur 8.2. Uit deze figuur blijkt dat de soortaccuraatheden voor beide methoden vergelijkbaar zijn. Het SVM geeft iets mindere resultaten voor soorten met samengestelde bladeren en soorten met meerdere mogelijke bladvormen zoals de berk (driehoekig of ruitvormig) en de klimop (puntig of afgerond). Het niet-lineaire RF model kan dergelijke soorten beter modelleren.

Uit analyse van de predicties bleek dat een aantal soorten regelmatig onderling verward worden, dergelijke soorten zijn weergegeven in Figuur 8.3. Vergissingen tussen de tien soorten uit Figuur 8.3 zijn verantwoordelijk voor ongeveer 40% van alle fouten.



Figuur 8.2: Accuraatheden per soort volgens een SVM met lineaire kernel en volgens een *random forest* model

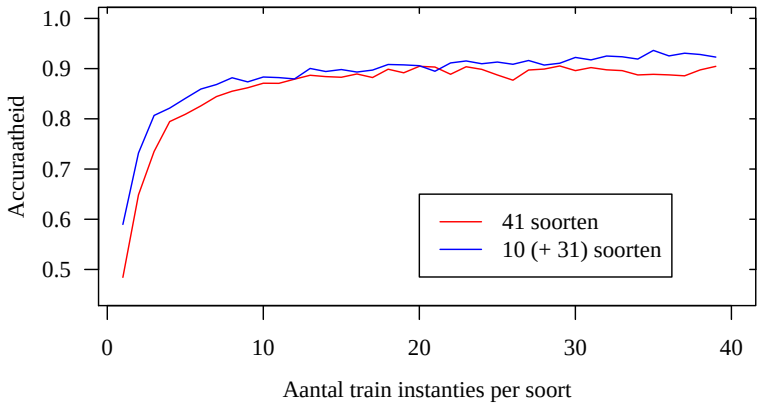


(a) (b) (c) (d)

Figuur 8.3: Moeilijk te onderscheiden soorten: (a) rode kornoelje, gele kornoelje, beuk, (b) wilde kardinaalsmuts, zure kers en Amerikaanse vogelkers, (c) boswilg en appel en (d) ruwe iep en haagbeuk.

8.3 Performantie in functie van het aantal trainingsdata

De performantie in functie van het aantal trainingsinstanties per soort is weergegeven in Figuur 8.4. De rode lijn werd berekend op basis van alle 41 soorten, bij hogere waarden (> 15) was echter niet het volledige aantal instanties beschikbaar voor elke soort waardoor dit geen goed beeld geeft. Van 10 soorten werden wel minstens 40 bladeren verzameld, daarom werd ook de gemiddelde accuraatheid voor enkel deze 10 soorten berekend (blauwe lijn). Voor de training van het tweede model werden ook alle klassen gebruikt. Uit de figuur blijkt dat een 20-tal instanties per soort voldoende is, meer instanties toevoegen heeft maar weinig positief effect.



Figuur 8.4: Accuraatheid in functie van het aantal trainingsdata.

Hoofdstuk 9

Besluit

9.1 Conclusies

In de literatuur zijn reeds tal van features voorgesteld die kunnen gebruikt worden voor de herkenning van (binaire) beelden. Voor deze scriptie werd een selectie gemaakt van features uit de literatuur en werd deze selectie aangevuld met een aantal zelf ontwikkelde features. Zo werd tot een finaal set van 98 features gekomen. Bepaalde features vertoonden sterke onderlinge correlaties of bevatten weinig informatie. Uit de experimenten bleek dat een geschikte subset van 20 features reeds tot een hoge accuraatheid kan leiden.De beste features zijn het percentage steel, de lengte-breedteverhouding, de excentriciteit, de soliditeit, de eerste twee momenten van Hu, de eerste drie somtermen van de verschillende op Fouriertransformatie gebaseerde technieken en de '100x100'-effeningscoëfficiënten. Deze uiteenlopende combinatie toont aan dat het zin heeft om verschillend technieken voor feature-extractie te combineren (iets dat in de literatuur nog niet gedaan werd). Wat de verschillende manieren om Fouriertransformatie op de contour toe te passen betreft, bleek de elliptische Fouriertransformatie tot de beste resultaten te leiden.

Voor de classificatie stond eveneens een ruim aanbod aan modellen ter beschikking. Ook hier werd een selectie gemaakt van een aantal uiteenlopende modellen. De beste methoden waren *random forests* en *support vector machines* (met lineaire kernel). Zowel een niet-lineair als een lineair model is dus geschikt.

Het KLR model op basis van de RBF-kernel met Baddeley's Δ-metriek voor het paarsgewijs vergelijken van beelden gaf gematigde resultaten, maar het is duidelijk minder efficiënt dan het gebruik van features. Voor de combinatie van de Baddeleykernel met de featurekernel werden hoopgevende resultaten bekomen. De performantie kan waarschijnlijk nog op verschillende manieren verbeterd worden: door optimalisatie van de parameters van de Baddeleykernel, door optimalisatie van de functie die de kernels combineert en door het gebruik van een beter geschikt classificatiemodel (bv. SVM). Bovendien gaf het gebruik van 3-voudige kruisvalidatie een onderschatting van de accuraatheid.

Door toevoegen van informatieve labels (multilabelclassificatie) kon de accuraatheid van de voorspellingen niet noemenswaardig verbeterd worden. Dergelijke informatie komt reeds duidelijk naar voor uit de features zelf. Dit kan verklaard worden doordat misclassificaties voornamelijk gebeuren tussen soorten die visueel, en ook op vlak van de toegevoegde labels, weinig verschillen. Toepassing van hiërarchische classificatie op basis van de taxonomische indeling had een negatief effect. Er kon worden besloten dat de taxonomische groepen hoger dan het geslacht, geen verband meer vertonen met morfologie van de soorten die ertoe behoren.

Voor het opsporen van nieuwe soorten werd enkel een eerste poging gedaan aan de hand van een eenvoudige methode op basis van de Euclidische afstand tussen instanties. Hier

bleek het nuttig om aan feature-selectie te doen: de beste resultaten werden bekomen met een afstandscriterium op basis van de 20 beste features. Vermoedelijk kan de *novelty*-test verbeterd worden door de distributie van de klassen afzonderlijk in rekening te brengen en dus de probabiliteit te berekenen dat een instantie tot elke soort behoort.

Als finaal model dat aangeboden kan worden aan een gebruiker wordt een eenvoudig RF-model verkozen. Daarvoor werd gevonden dat het foutenpercentage een lineaire trend vertoont in functie van het aantal soorten: een halvering van het aantal soorten komt overeen met een halvering van het foutenpercentage. Uit analyse van de performantie per soort bleek dat de fouten zich voornamelijk binnen een aantal probleemgroepen situeren terwijl andere soorten met 100% accuraatheid worden voorspeld. Daarnaast werd gevonden dat voor de trainingsdataset een twintigtal instanties per soort volstaat.

Het *random forest* model werd finaal geïncorporeerd in een *standalone* applicatie met een grafische interface die kan worden beschikbaar gesteld aan gebruikers (een handleiding tot deze applicatie werd toegevoegd als bijlage en een gecompileerde versie is beschikbaar op CD).

9.2 Antwoord op de onderzoeksvragen

In de inleiding werden een aantal onderzoeksvragen gesteld die in de loop van de thesis uitgewerkt werden. Hier volgt een samenvattend antwoord op elk van de vragen.

Vraag 1: Is het mogelijk bomen te classificeren op basis van beeldmateriaal van de bladeren?
Het classificeren van bomen op basis van beeldmateriaal is reëel. Dit bleek zowel uit de literatuurstudie als uit eigen experimenten. Echter werden ook beperkingen vastgesteld: het aantal soorten in de databank heeft een belangrijke invloed op de performantie. Bij benadering werd gevonden dat bij een verdubbeling van het aantal soorten ook het foutenpercentage verdubbelt. Daarnaast bestaan er soorten die op basis van het blad moeilijk of helemaal niet te onderscheiden zijn (bv. *Cornus sanguinea* en *Cornus mas*), voor deze soorten zou toevoegen van extra informatie over vruchten, bloeiwijze of schors een oplossing bieden. De onderzochte classificatietechniek is ook beperkt tot loofbomen, voor toepassing op coniferen zullen andere methoden nodig zijn.

Vraag 2: Welke eigenschappen zijn geschikt om bladeren te beschrijven en soorten van elkaar te onderscheiden?
De features die werden berekend (en waarvan dus intuïtief gedacht werd dat ze nuttig waren) bleken allen een zekere hoeveelheid informatie te bevatten. De eenvoudige geometrische features hadden de beste voorspellende kwaliteit, maar het was nuttig deze aan te vullen met de overige, meer complexe, features. Ook het paarsgewijs vergelijken van beelden aan de hand van een afstandsmaat werd nuttig bevonden.

Vraag 3: Kunnen de features voldoende nauwkeurig bepaald worden zonder tussenkomst van een persoon?
Bij de extractie van features treden voornamelijk fouten op bij het standaardiseren van de oriëntatie en het verwijderen van de bladsteel. Er werd echter in de inleiding vooropgesteld om geen interactie van de gebruiker te vragen, daardoor zijn deze fouten onvermijdelijk. Het belang van de fouten bleef gelukkig beperkt doordat bepaalde fouten systematisch voorkomen bij bladeren van eenzelfde soort. Een mogelijke oplossing zou zijn om de gebruiker te laten fungeren als supervisor en de kans te geven de oriëntatie te corrigeren indien nodig.

Vraag 4: Wat is de performantie van een dergelijk classificatiesysteem?
Voor de 41 soorten in de aangelegde dataset werd een performantie bekomen van 90%. Uit analyse van de performante per soort kwam naar voor dat de fouten zich voornamelijk situeren binnen een aantal probleemgroepen bestaande uit zeer similaire soorten, andere soorten worden daarentegen geregeld met 100% nauwkeurigheid voorspeld.

Vraag 5: Kan de performantie verbeterd worden door extra kennis toe te voegen aan het model?
Waarschijnlijk niet. Zowel het gebruik van multilabels als het toevoegen van kennis over de taxonomische indeling van het plantenrijk konden de performantie niet noemenswaardig verhogen.

Vraag 6: Is het mogelijk het systeem aan te passen om nieuwe soorten als dusdanig te classificeren?
Er werd in deze thesis getracht de *novelties* op te sporen op een relatief eenvoudige manier. Deze methode gaf slechts matige resultaten, vermoedelijk kunnen meer geavanceerde methoden iets betere resultaten opleveren. Bepaalde soorten vertonen echter zodanig grote gelijkenissen met elkaar, waardoor het onwaarschijnlijk is dat deze soorten als *novelties* voorspeld kunnen worden.

9.3 Toekomstvisie

Wereldwijd zijn verschillende teams bezig met de ontwikkeling van boomclassificatiesystemen. Recent werd een iPhone-applicatie, 'Leafsnap', op de markt gebracht die 191 boomsoorten uit New York City en Washington D.C. kan herkennen met een accuraatheid van 70% (Belhumeur *et al.*, 2011). De applicatie krijgt media-aandacht tot in België, wat erop wijst dat er wel degelijk interesse is voor dergelijke software. Op zich is het perfect mogelijk een gelijkaardige applicatie te ontwikkelen voor Vlaamse of Europese boomsoorten. Het is slechts afwachten tot iemand het initiatief neemt om een volledige databank aan te leggen. De ontwikkelaars van Leafsnap zien ook mogelijkheden om grote databanken te realiseren door alle bladeren die gebruikers van de applicatie uploaden te bewaren (*crowdsourcing*).

Wat het onderzoek naar methoden betreft zijn er reeds tal van mogelijkheden voor feature-extractie en classificatie. Voornamelijk het benutten van textuur en nervatuur kan nog verbeterd worden. Ook segmentatie van beelden op niet egale achtergrond zou nuttig zijn om de gebruiksvriendelijkheid te verhogen. Toekomstig werk zal zich dus waarschijnlijk vooral in de beeldverwerking situeren.

Tot slot zijn er nog tal van opties voor verder werk inzake het determineren van kruiden, bloemen, vogels, vissen, insecten, macro-invertebraten, ...

Literatuurlijst

J. W. André Elisseeff (2001). Kernel methods for multi-labelled classification and categorical regression problems. *Technical Report, BioWulf Technologies*.

A. Baddeley (1992). An error metric for binary images. *Proceedings of the International Workshop on Robust Computer Vision*, 59–78.

J. Baeten (2006). Beeldverwerking: Beeldverbetering, -bewerking en -analyse. *Slides Beeldverwerking Deel1*. Katholieke Hogeschool Limburg.

P. Belhumeur, D. Chen, S. Feiner, D. Jacobs, W. Kress, H. Ling, I. Lopez, R. Ramamoorthi, S. Sheorey, S. White en L. Zhang (2008). Searching the world's herbaria: A system for visual identification of plant species. *Proceedings of the 10th European Conference on Computer Vision*, 116–129.

P. Belhumeur, D. Jacobs en J. Kress (2011). Leafsnap: An electronic field guide. *Columbia University, University of Maryland, Smithsonian Institution*. URL `http://leafsnap.com/`.

Berlin University of Technology (2009). Fourier descriptors. *Slides Computer vision and remote sensing*.

C. M. Bishop (2006). Pattern Recognition and Machine Learning. *Springer, Heidelberg, Duitsland*.

L. Breiman (2001). Random forests. *Machine Learning*, 45:5–32.

L. Breiman en A. Cutler (2004). Random forests. *Department of Statistics, University of California, Berkeley*. URL `http://stat-www.berkeley.edu/users/breiman/RandomForests/cc_home.htm`.

I. Brunner, S. Brodbeck, U. Büchler en C. Sperisen (2001). Molecular identification of fine roots of trees from the Alps: Reliable and fast DNA extraction and PCR-RFLP analyses of plastid DNA. *Molecular Ecology*, 10:2079–2087.

W. Burger en M. J. Burge (2008). Digital Image Processing: an Algorithmic Introduction using Java. *Springer, Heidelberg, Duitsland*. 1ste editie.

C. Caballero en M. C. Aranda (2010). Plant species identification using leaf image retrieval. *Proceedings of the 9th Association for Computing Machinery International Conference on Image and Video Retrieval*, 327–334.

D. Casanova, J. J. de Mesquita Sá Junior en O. M. Bruno (2009). Plant leaf identification using Gabor wavelets. *International Journal Imaging System Technology*, 19:236–243.

N. Cesa-Bianchi, C. Gentile en L. Zaniboni (2006). Hierarchical classification: combining bayes with SVM. *Proceedings of the 23th International Conference on Machine Learning*, 177–184.

W. Cheng en E. Hüllermeier (2009). Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning*, 76:211–225.

M. Culp, K. Johnson, en G. Michailidis (2010). Ada: an R package for stochastic boosting. URL `http://CRAN.R-project.org/package=ada`. R package version 2.0-2.

N. Dalal en B. Triggs (2005). Histograms of oriented gradients for human detection. *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2:886–893.

M. Dallwitz (1980). A general system for coding taxonomic descriptions. *Taxon*, 29:41–46.

O. Dekel, J. Keshet en Y. Singer (2004). Large margin hierarchical classification. 209–216.

R. Dickinson, T. Dickinson, D. Metsger en T. Winterhalt (2004). The ROM field guide to wildflowers of Ontario. *McClelland & Stewart, Toronto, Canada*.

E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer en A. Weingessel (2010). e1071: Misc functions of the department of statistics (e1071), TU Wien. URL `http://CRAN.R-project.org/package=e1071`. R package version 1.5-24.

J.-X. Du, D.-S. Huang, X.-F. Wang en X. Gu (2006). Computer-aided plant species identification based on leaf shape matching technique. *Transactions of the Institute of Measurement and Control*, 28:275–284.

J.-X. Du, X.-F. Wang en G.-J. Zhang (2007). Leaf shape based plant species recognition. *Applied Mathematics and Computation*, 185:883–893.

M. Du, S. Zhang en H. Wang (2009). Supervised isomap for plant leaf image classification. *International Conference on Intelligent Computing 2009,Lecture Notes in Artificial Intelligence*, 5755:627–634.

J. Duminil, M. Heuertz, J.-L. Doucet, N. Bourland, C. Cruaud, F. Gavory, C. Doumenge, M. Navascués en O. J. Hardy (2010). CpDNA-based species identification and phylogeography: application to African tropical tree species. *Molecular Ecology*, 19:5469–5483.

B. Efron en R. Tibshirani (1993). An introduction to the bootstrap. *Chapman and Hall*.

ETI BioInformatics (2010). Heukels' interactive flora van Nederland. Heukels' iPhone app. URL `http://www.SoortenBank.nl`.

Y. Feng en S. Zhang (2009). Supervised locally linear embedding for plant leaf image feature extraction. *Lecture Notes in Computer Science*, 5754:1–7.

J. Flusser (2000). On the independence of rotation moment invariants. *Pattern Recognition*, 33:1405–1410.

J. Flusser en T. Suk (2006). Rotation moment invariants for recognition of symmetric objects. *IEEE Transactions on image processing*, 15:3784–3790.

Y. Freund en R. E. Schapire (1996). Experiments with a new boosting algorithm. *Proceedings of the 13th International Conference on Machine Learning*, 148–156.

Y. Freund en R. E. Schapire (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139.

J. Friedman, T. Hastie en R. Tibshirani (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33:1–22. URL `http://CRAN.R-project.org/package=glmnet`.

H. Fu, Z. Chi, D. Feng en J. Song; (2005). Machine learning techniques for ontology-based leaf classification. *Proceedings of the Control, Automation, Robotics and Vision Conference*, 1:681–686.

N. Ghamrawi en A. McCallum (2005). Collective multi-label classification. *Proceedings of the 14th Association for Computing Machinery international conference on Information and knowledge management*, 195–200.

E. Gilleland, T. C. M. Lee, J. H. Gotway, R. G. Bullock en B. G. Brown (2008). Computationally efficient spatial forecast verification using Baddeley's delta image metric. *Monthly Weather Review*, 136:1747–1757.

R. C. Gonzalez en R. E. Woods (2008). Digital Image Processing. *Prentice Hall, New Jersey, U.S.A.* 3de editie.

F. Gouveia, V. Filipe, M. Reis, C. Couto en J. Bulas-Cruz (2002). Biometry: the characterisation of chestnut-tree leaves using computer vision. *Proceedings of the International Symposium on Industrial Electronics*, 3:757–760.

T. Hastie, R. Tibshirani en J. Friedman (2008). *The Elements of Statistical Learning: Data Mining, Inference and Prediction*. Springer Series in Statistics. Springer, Heidelberg, Duitsland, tweede editie.

N. Higham (1988). Computing a nearest symmetric positive semidefinite matrix. *Linear Algebra and Applications*, 103:103–118.

T. Hofmann, L. Cai en M. Ciaramita (2003). Learning with taxonomies: Classifying documents and words.

M. K. Hu (1962). Visual pattern recognition by moment invariants. *IEEE Transactions on information theory*, 8:179–187.

T. Karrels (2006). Fourier descriptors: Properties and utility in leaf classification. *Electrical and Computer Engineering*, 533:1–12.

D. Knight, J. Painter en M. Potter (2010). Automatic plant leaf classification for a mobile field guide: An android application. URL `http://www.stanford.edu/~jpainter/documents/Plant%20Leaf%20Classification.pdf`.

D. Koller en M. Sahami (1997). Hierarchically classifying documents using very few words. *Proceedings of the 14th International Conference on Machine Learning*, 170–178.

F. P. Kuhl en C. R. Giardina (1982). Elliptic fourier features of a closed contour. *Computer Graphics and Image Processing*, 18:236–258.

C.-L. Lee en S. Chen (2006). Classification of leaf images. *International Journal of Imaging Systems and Technology*, 16:15–23.

A. Liaw en M. Wiener (2002). Classification and regression by random forest. *R News*, 2:18–22. URL `http://CRAN.R-project.org/package=randomForest`.

J. Liu, S. Zhang en S. Deng (2009). A method of plant classification based on wavelet transforms and support vector machines. *Lecture Notes in Computer Science*, 5754:253–260.

Q.-K. Man, C.-H. Zheng, X.-F. Wang en F.-Y. Lin (2008). Recognition of plant leaves using support vector machine. *Communications in Computer and Information Science*, 15:192–199.

A. McCallum, R. Rosenfeld en T. M. A. Y. Ng (1998). Improving text classification by shrinkage in a hierarchy of classes. *Proceedings of the 15th International Conference on Machine Learning*, 359–367.

T. McLellan en J. A. Endler (1998). The relative success of some methods for measuring and describing the shape of complex objects. *Systems biology*, 47:264–281.

I. Mukherjee en R. E. Schapire (2010). A theory of multiclass boosting. *Advances in Neural Information Processing Systems*, 23:1714–1722.

Y. Nam, E. Hwang en D. Kim (2008). A similarity-based leaf image retrieval scheme: Joining shape and venation features. *Computer Vision and Image Understanding*, 110:245–259.

J. C. Neto, G. E. Meyer, D. D. Jones en A. K. Samal (2006). Plant species identification using elliptic Fourier leaf shape analysis. *Computers and Electronics in Agriculture*, 50:121–134.

M. S. Nixon en A. S. Aguado (2008). Feature extraction and image processing. *Academic Press*. 2de editie.

N. Otsu (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9:62–66.

J. Park, E. Hwang en Y. Nam (2008). Utilizing venation features for efficient leaf image retrieval. *Journal of Systems and Software*, 81:71–82.

R Development Core Team (2010). R: A language and environment for statistical computing. *R Foundation for Statistical Computing*. URL `http://www.R-project.org`. Versie R 2.11.1.

J. E. Reeb (1997). Scientific classification of trees: An introduction for wood workers. *Agriculture home economics 4-H development*, 61:1–4.

H. Sahbi (2007). Kernel PCA for similarity invariant shape recognition. *Neurocomputing*, 70:3034–3045.

J. Schaefer, R. Opgen-Rhein en K. Strimmer (2010). Corpcor: Efficient estimation of covariance and (partial) correlation. URL `http://CRAN.R-project.org/package=corpcor`. R package version 1.5.7.

B. Schölkopf en A. J. Smola (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. *MIT Press, Massachusetts, U.S.A.*

O. J. O. Söderkvist (2001). Computer vision classification of leaves from Swedish trees. *Univerity of Linköping*. Afstudeerwerk.

C. N. Silla en A. A. Freitas (2011). A survey of hierarchical classification across different application domaines. *Data Mining and Knowledge Discovery*, 22:31–72.

S. Sinha (2004). Leaf shape recognition via support vector machines with edit distance kernels. *Oregon State University*. Afstudeerwerk.

S. W. Smith (1998). *The Scientist and Engineer's Guide to Digital Signal Processing*. California Technical Publishing, San Diego, U.S.A. URL `http://www.dspguide.com/pdfbook.htm`.

J. Solé-Casals, C. M. Travieso, J. B. Alonso en M. A. Ferrer (2009). Improving a leaves automatic recognition process using PCA. *Proceedings of the International Workshop on Practical Applications of Computational Biology and Bioinformatics*, 49:243–251.

J. Staal en M. A. Viergever (2004). Ridge-based vessel segmentation in color images of the retina. *IEEE Transacions on medical imaging*, 23:501–509.

The MathWorks, Inc. (2010a). *Image Processing Toolbox ™ User's Guide*. Matlab. Versie 7.0.

The MathWorks, Inc. (2010b). Matlab®. URL `http://www.mathworks.com/products/matlab/`. Versie 7.10.0.499 (R2010a).

D. Thomas (2006). Elliptical Fourier shape descriptors: Forward and reverse elliptical Fourier transforms of x,y data. *Matlab Central File Exchange*. URL `http://www.mathworks.com/matlabcentral/fileexchange/12746-elliptical-fourier-shape-descriptors`.

A. Tort (2003). Elliptical fourier functions as a morphological descriptor of the genus *Stenosarina (Brachiopoda, Terebratulida, New Caledonia)*. *Mathematical Geology*, 35:873–885.

G. Tsoumakas en I. Katakis (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, 3:1–13.

R. van der Meijden (2005). Heukels' flora van Nederland. *Wolters/Noordhoff*. 23e editie.

V. Viscosi, P. Fortini, D. E. Slice, A. Loy en Blasi (2009). Geometric morphometric analyses of leaf variation in four oak species of the subgenus *Quercus* (Fagaceae). *Plant Biosystems - An International Journal Dealing with all Aspects of Plant Biology*, 143:575–587.

X.-F. Wang, D.-S. Huang, J.-X. Du, H. Xu en L. Heutte (2008). Classification of plant leaf images with complicated background. *Applied Mathematics and Computation*, 205:916–926.

Z. Wang, Z. Chi en D. Feng (2003). Shape based leaf image retrieval. *Institution of Electrical Engineers Proceedings - Vision, image and signal processing*, 150:34–43.

S. White, S. Feiner en J. Kopylec (2006a). Virtual vouchers: Prototyping a mobile augmented reality user interface for botanical species identification. *Proceedings of the First IEEE Symposium on 3D User Interfaces*, 119–126.

S. White, D. Marino en S. Feiner (2006b). Leafview: A user interface for automated botanical species identification and data collection. *Proceedings of the 19th annual Association for Computing Machinery Symposium on User Interface Software and Technology*, 101–102.

I. H. Witten en E. Frank (2005). Data mining: Practical machine learning tools and techniques. *The Morgan Kaufmann Series in Data Management Systems (Elsevier), Burlington, U.S.A.* 2de editie.

S. G. Wu, F. S. Bao en E. Y. Xu (2007a). A leaf recognition algorithm for plant classification using probabilistic neural network: Flavia user manual. *Proceedings of the 7th IEEE International Symposium on Signal Processing and Information Technology*, 1–8.

S. G. Wu, F. S. Bao, E. Y. Xu, Y. Wang, Y.-F. Chang en Q.-L. Xiang (2007b). A leaf recognition algorithm for plant classification using probabilistic neural network. *Clinical Orthopaedics and Related Research*, 1–6.

X.-Y. Xiao, R. Hu, S.-W. Zhang en X.-F. Wang (2010). HOG-based approach for leaf classification. *Lecture Notes in Computer Science*, 6216:149–155.

G.-J. Zhang, X.-F. Wang, D.-S. Huang, Z. Chi, Y.-M. Cheung, J.-X. Du en Y.-Y. Wan (2004). A hypersphere method for plant leaves classification. *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing*, 165–168.

J. Zhu en T. Hastie (2005). Kernel logistic regression and the import vector machine. *Journal of Computational and Graphical Statistics*, 14:185–205.

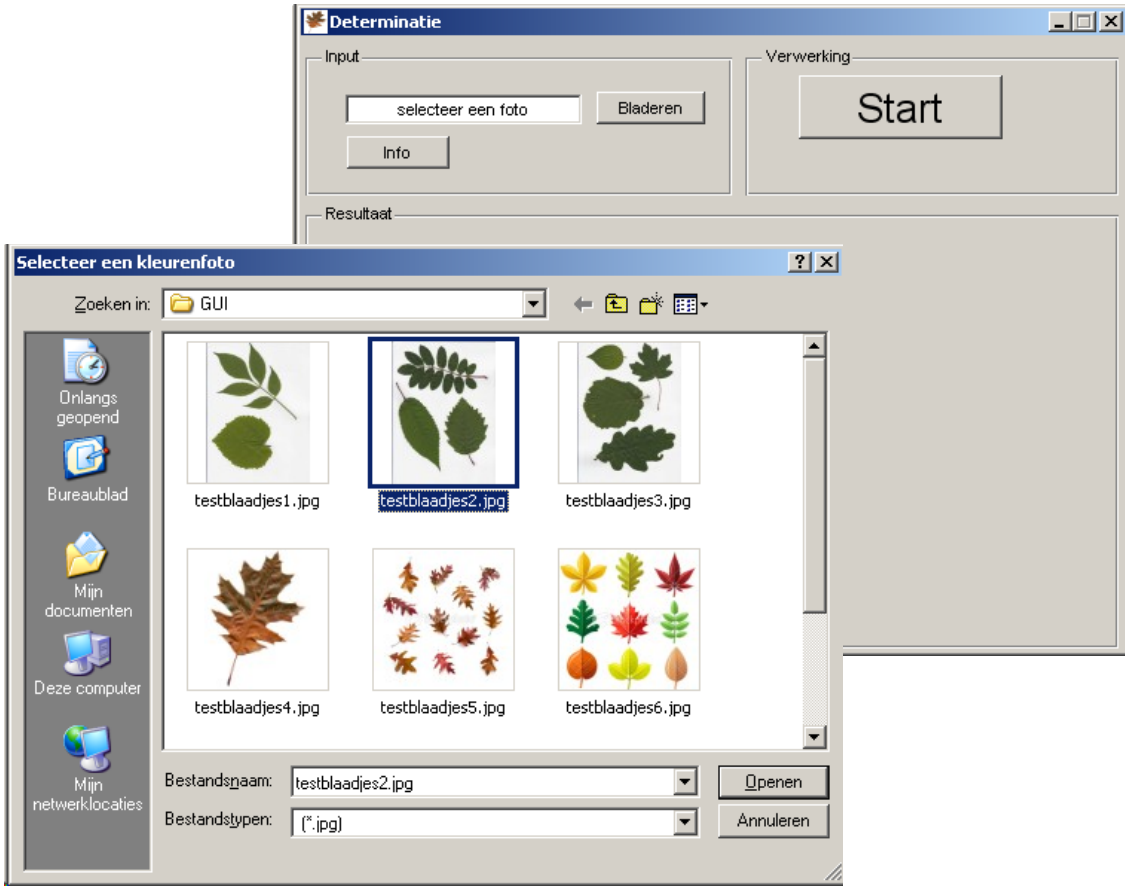
Bijlage: Handleiding bij de Determinatie app.

De feature-extractie en het finale *random forests* classificatiemodel uit de scriptie werden verwerkt tot een *standalone* applicatie met een grafische gebruikersomeving (GUI). De applicatie werkt op basis van de Matlab Compiler Runtime (MCR).

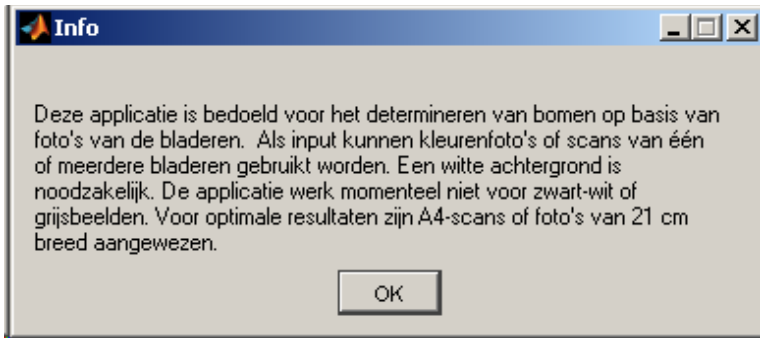
Installatie van de MCR
De applicatie maakt gebruik van MCR Versie 7.13. Indien reeds een recente versie van Matlab geïnstalleerd is, zal de applicatie onmiddellijk werken. Anders kan de MCR geïnstalleerd worden door uitvoeren van het bestand *MCRInstaller.exe* op de bijgevoegde CD.

Het determineren
De applicatie wordt geopend door uitvoeren van het bestand *Determinatie.exe* op de bijgevoegde CD.

Onderstaande figuur geeft een *screenshot* van de GUI en het venster dat verschijnt wanneer in het onderste venster op 'Bladeren' geklikt wordt.



Door op 'info' te klikken verkrijgt de gebruiker volgend informatievenster:



De determinatie begint na een klik op de startknop. Onder de startknop wordt de voortgang/status weergegeven. Als het determineren voltooid is verschijnt een menu waarmee de resultaten van de verschillende bladeren in de foto kunnen worden weergegeven. Voor elk blad worden de top 3 voorspellingen weergegeven samen met een beeld van het gepreprocesste blad (zoals te zien in onderstaand *screenshot*).