Norges miljø- og biovitenskapelige universitet

Masteroppgave 2018 30 stp

Fakultet for realfag og teknologi
Ivar Maalen-Johansen

Klassifisering av trær i urbane områder ved analyser på hyperspektrale flybilder

Airborne hyperspectral mapping of trees in an urban area

Erik Røstad
Geomatikk
Fakultet for realfag og teknologi

Sammendrag

Kartlegging av vegetasjon i urbane områder har vært en utfordring for fjernmåling på grunn av kompleks variasjon i romlig innhold og spektralinformasjon. Hyperspektrale bilder kan være løsningen. I denne oppgaven ble to hyperspektrale datasett fra HySpex brukt til å klassifisere trær i en urban sone i Oslo.

HySpex-sensorene har en oppløsning som er unik innenfor fjernmåling. Datasettet VNIR bestod av 186 bånd med en romlig oppløsning på 0,3 m. SWIR bestod av 288 bånd med en romlig oppløsning på 0,7 m. Av totalt 474 bånd ble 370 bånd brukt til klassifiseringen. Målet med klassifiseringen var å se hvilke resultater man kunne oppnå på trær i et urbant område med så gode data. I tillegg var det ønskelig å finne ut om SWIR kunne gjøre klassifiseringen bedre.

For å teste dette ble spektralinformasjon fra 805 trær hentet ut. For å ta høyde for variasjoner i belysning ble et normalisert datasett basert på radiansverdier brukt. For å ta høyde for overlapp med andre materialer ble middelverdien for hvert tre benyttet. Et utvalg bestående av 585 trær og ni arter dannet utgangspunktet for klassifiseringen. Klassifisering ble først kjørt på VNIR og deretter på VNIR+SWIR. To ulike algoritmer for styrt klassifisering ble testet, logistisk regresjon og SVM. Dessuten ble to ulike måter for vekting testet, lik vekting og balansert vekting. Balansert vekting tar hensyn til ubalanse i datasettet.

Balansert vekting ga best resultat. Med VNIR oppnådde SVM en total nøyaktighet på 88,6 % og en kappa-koeffisient på 0,85. Med data VNIR og SWIR kombinert oppnådde SVM en total nøyaktighet på 91,5 % og en kappa–koeffisient på 0,89.

Abstract

Remote sensing of urban areas is challenging due to complex spatial and spectral variations. In this thesis, two hyperspectral datasets from HySpex were used to classify trees in an urban zone in Oslo. The HySpex sensors have a resolution that is unique in the field of remote sensing. The VNIR dataset consisted of 186 bands with a spatial resolution of 0,3 m. The other data set consisted of 288 bands in the SWIR region, with a spatial resolution of 0,7 m. A total of 370 bands were used in the classification.

The primary object of this thesis was to see what results such good data would give in an urban area. In addition, it was desirable to determine, whether the combined information from VNIR and SWIR would give better results than VNIR alone. To test the ability of the hyperspectral data to discriminate between tree species, spectral features from 805 trees were extracted. A normalised dataset was used to account for variations in illumination. To account for complex overlap with other materials, the mean spectrum from each tree was used.

A selection of 585 trees and nine species formed the basis for the classification. The classification was first done only using the VNIR dataset and then using both VNIR and SWIR. Two different supervised algorithms were used, logistic regression and Support Vector Machine (SVM). In addition, two different ways of assigning weights to the data were tested, equal and balanced weighting.

Balanced weighting accounted for class imbalance and gave the best results. SVM achieved an overall accuracy of 88,6 % and a kappa-coefficient of 0,82 when the VNIR dataset was used. The overall accuracy increased to 91,5 % when both VNIR and SWIR were used. The kappa-coefficient now increased to 0,89. The results shows that hyperspectral data can be used to map urban trees with a high degree of accuracy.

Forord

Denne masteroppgaven markerer slutten på mine fire år ved NMBU. Oppgaven har blitt til ved et samarbeid med Oslo kommune. Det har vært en lærerik og spennende prosess. Samtidig har det også vært utfordrende til tider. Siden det var et nytt datasett var det mye nytt å sette seg inn i av teori og programvarer.

Først og fremst vil jeg takke Oslo kommune for samarbeidet og muligheten til å jobbe med et så spennende datasett. Deretter vil jeg rette en stor takk til biveilederen min Ingunn Burud som har bidratt til økt forståelse og kunnskap omring hyperspektrale data, og gitt gode råd og innspill underveis.

En stor takk rettes til hovedveileder Ivar Maalen-Johansen for gode råd og ideer, samt god veiledning ved behov for det.

Videre vil jeg takke Floris Jan Groesz ved Blom for gode innspill og Vetle Odin Jonassen ved TerraTec AS for behjelpelighet omkring de hyperspektrale dataene. Takk til Oliver Tomic og Kristian Hovde Liland ved NMBU for hjelp med programmering.

Jeg ønsker også å takke de studentene som har vært med i prosessen på disse dataene. Deretter vil jeg takke mine venner og familie. En spesiell takk til Lars og Karin for korrekturlesing.

Til slutt vil jeg takke mine foreldre for å ha støttet meg og hjulpet til med å holde motet oppe.

Norges miljø- og biovitenskapelige universitet

Ås, 12.mai 2018

Erik Røstad

Forkortelser

nm nanometer ($=10^{-9}$)

CSV Comma Separated Value

DBH Diameter at Breast Height (diameter ved brysthøyde)

FN False Negatives

FP False Positives

NDVI Normalised Difference Vegetation Index

NIR Near InfraRed

PC Prinsipal Component (prinsipalkomponent)

PCA Principal Component Analysis (Prinsipalkomponentanalyse)

PRE Presisjon

ROI Region Of Interest (Region av interesse)

SAM Spectral Angle Mapper

SVM Support Vector Machine

SWIR Shortwave Infrared

TB TeraByte

TF True False

TIFF Tagged Image File Format

TP True Positives

VNIR Visible Near Infrared

Innholdsfortegnelse

1 Innledning XIII
1.1 Bakgrunn XIII
1.2 Tema og problemstilling XIV
1.3 Tidligere forskning XV
1.4 Oppsett og struktur XVII
2 Teori 1
2.1 Lys og elektromagnetisk stråling 1
2.2 Radians og reflektans 2
2.3 Vegetasjon og stråling 3
2.3.1 NDVI 4
2.4 Definisjon av fjernmåling 5
2.5 Hyperspektrale sensorer 6
2.6 Pushbroom-skanner 7
2.6.1 Hyperkube 7
2.7 PCA (Prinsipalkomponentanalyse) 8
2.8 Klassifisering 10
2.8.1 SVM (Support Vector Machine) 10
2.8.2 SAM (Spectral Angle Mapper) 12
2.8.3 Logistisk regresjon 13
2.8.4 Kryssvalidering 14
2.8.5 Over- og undertilpasning 15
2.8.6 Lærings- og valideringskurver 15
2.8.7 GridSearchCV til finjustering av modellen 17
2.8.8 Evaluering av klassifiseringen 18
3 Materialer og metode 22
3.1 Materialer 22
3.1.1 Spesifikasjoner HySpex 22
3.1.2 Prosjektområdet 23
3.1.3 Gjennomgang av hyperspektrale datasett 24
3.1.4 Valg av hyperspektrale datasett 26
3.1.5 Testområder feltarbeid 29

3.1.6 Stort testområde 30
3.1.7 Programvarer og tilleggsmoduler 31
3.1.8 Filformater 32
3.2 Metode 34
3.2.1 Preprosessering og bearbeiding av hyperspektrale data 36
3.2.2 Identifisering og valg av trær 37
3.2.3 Uthenting av spektralinformasjon fra trærne 38
3.2.4 Vurdering av spektral separasjon 39
3.2.5 Veivalg klassifisering 41
3.2.6 Beskrivelse av klassifiseringen 41
4 Resultater og diskusjon 43
4.1 Resultater eksplorativ analyse 43
4.1.1 PCA 44
4.1.2 Evaluering av spektralsignaturer 47
4.1.3 Vurdering av spektral separasjon 51
4.1.4 Test av ulike algoritmer for styrt klassifisering 51
4.1.5 Test av forskjellige størrelser på ROI-ene 55
4.1.6 Hovedfunn eksplorativ analyse 56
4.2 Resultater stort testområde 57
4.2.1 PCA 57
4.2.2 Evaluering av spektralsignaturer 60
4.2.3 Vurdering av spektral separasjon 63
4.2.4 Klassifiseringen 66
4.2.5 Valideringskurver 68
4.2.6 Læringskurver 71
4.2.7 Forvirringsmatriser 74
4.2.8 Evaluering av modellenes prediksjonsevne 77
4.2.9 Sammenligning av modellene 82
4.2.10 En nærmere kikk på klasser som skilte seg ut 83
4.3 Utfordringer og svakheter 86
5 Konklusjon 89
5.2 Forslag til videre arbeid ..... **Feil! Bokmerke er ikke definert.**
Litteraturliste 92

Vedlegg 96

Figur 1: Det elektromagnetiske spekteret. 1
Figur 2: Reflektans (rød) og normalisert radians (grønn) for samme pikselkoordinat plottet mot bølgelengde. 2
Figur 3: Spektralsignaturer for vegetasjon og vann (Campbell & Wynne, 2011). 3
Figur 4: Reflektansspektra for ulike typer grønn vegetasjon. Tørr vegetasjon har en karakteristisk flatere kurve (Smith, 2006). 4
Figur 5**:** Multispektrale versus hyperspektrale bånd (Borengasser et al., 2007). 6
Figur 6**:** Prinsippet ved pushbroom-skanning (Richards, 2013). 7
Figur 7: Illustrasjon av en hyperkube. 8
Figur 8: Prinsippet med PCA. 9
Figur 9: Scores-plott for ulike trær. 10
Figur 10: Prinsippet ved SVM (OpenCV, 2016). 11
Figur 11: Lav C-verdi (til venstre) gir store marginer, mens høy C-verdi (høyre) gir lave marginer. 12
Figur 12: Prinsippet ved SAM (Borengasser et al., 2007). 12
Figur 13: Prinsippet med kryssvalidering ((Pedregosa et al., 2011). 14
Figur 14: over- og undertilpasning (quora, 2017). 15
Figur 15: Prinsippet med læringskurver (Rascha, 2016). 16
Figur 16: Eksempel på valideringskurve (Pedregosa et al., 2011). 17
Tabell 1: Prinsippet med forvirringsmatrisa. 18
Tabell 2: inndeling for kappa-koeffisienten (Richards, 2013). 19
Tabell 3: HySpex-spesifikasjoner (Norsk Elektro Optikk AS, u.å.). 22
Tabell 4: Spesifikasjoner for flyvingen (TerraTec AS, 2017). 23
Figur 17: Oversikt over prosjektområdet. 23
Figur 18: Noen deformasjoner på hus. Husene på bildet er i virkeligheten rette. Bildet er fra det normaliserte datasettet. 24
Figur 19:Pikselen på dette bildet har nullverdier fra omtrent bånd 30 til bånd 1.Hvite områder er områder hvor pikselinformasjonen er fjernet og pikselen har verdien null. 27
Figur 20: Spektralinformasjonen for en piksel i et skyggeområde vist i bånd. For denne pikselen er informasjonen fram til rundt bånd 90 fjernet og satt til null. Som man kan se er områdene med nullverdier forskjellig fra figur 3.19. 27
Figur 21: Bånd 31 i reflektansssettet. 27
Figur 22:Samme område før og etter normalisering. Originalbilde til venstre, normalisert bilde til høyre. Skygger er minimert på bildet til høyre, samtidig kan det se ut som om skyggekorreksjonen har gjort skyggeområdene lysere enn omgivelsene. Begge bildene er vist med båndkombinasjonen R, G, B = 55,41,21. 28
Figur 23: Oversikt over de ni små testområdene sammen med prosjektområdet (gult). 29
Figur 24: Oversikt over det store testområdet. 30
Figur 25: Flytdiagram for den eksplorative analysen. 34
Figur 26: Flytdiagrammer for prosesseringen. 35

Figur 27: Spektralprofil før og etter fjerning av bånd. Brudd i kurven tilsvarer bånd som er fjernet. 36
Figur 28: Eksempel på område med klynger av trær i forskjellige klasser. Her er det kjørt en PCA for å forsøke å se forskjell på klassene. Båndkombinasjon: R=PC1, G=PC2, B=PC3. .. 37
Figur 29:Prosessen for å hente ut spektralinformasjon. sirkel på 1m radius(lyseblått) ble brukt som utgangspunkt til å definere en ROI (rødt). 38
Figur 30: Jeffries-Matusita-avstanden (Richards, 2013). 40
Tabell 5:Inndeling av separasjon. 40
Figur 31: Flytdiagram for klassifiseringen. 42
..... 43
Figur 32: Oversiktsbilde av testområde 1 med nummererte trær i henhold til tabell 6. 43
. 44
Figur 33: Forklart varians per prinsipalkomponent sammen med akkumulert varians. Detaljer for hver komponent er gitt i tabell 7. 44
Tabell 7:Detaljer for hver prinsipalkomponent. 44
Figur 34: score-bilde av PC1 med tilhørende ladningsplott. Størst variasjon er det mellom vegetasjon og veier. Dette gjenspeiles i grafen til høyre som viser bidraget fra hver bølgelengde. Det største bidraget kommer fra bølgelengder fra ca.700 nm og oppover, i det nærinfrarøde området. Det er nettopp i dette området at vegetasjon reflekterer mest stråling. 45
Figur 35: Score-bilde av PC2 med tilhørende ladningsplott. PC2 har størst bidrag fra bølgelengder mellom ca. 600 nm til ca.750 nm. Her er den største variasjonen mellom ulike tak, fra taket på blokkene (sort) til taket på bodene (hvitt). 45
Figur 36: score-bilde av PC3 med tilhørende ladningsplott. Mesteparten av bidraget til PC3 kommer fra bølgelengder under 600 nm. På score-bildet er det størst forskjell mellom tak på boder (sort) og asfalt (hvitt). Selv om PC3 forklarer bare 0,9% av variasjonen i datasettet er det fortsatt en del detaljer igjen. 45
Figur 37: score-bilde av PC4 med tilhørende ladningsplott. Den fjerde prinsipalkomponenten fanger opp en enda mindre del av variasjonen i datasettet. Nå begynner det å bli merkbart mye støy i bildet. Her er det størst variasjon mellom busker og trær (hvitt), og gress (sort). Det er interessant å observere at enkelte trær skiller seg ut fra de andre, spesielt treet nede til høyre som er Lind. 46
Figur 38: Score-bilde av PC5 med tilhørende ladningsplott. På PC5 er det størst bidrag fra området mellom 700nm og til ca. 750nm. På bildet tilsvarer det lyse områder og er representert ved løvtre og busker. 46
Figur 39: PC6 inneholder bare støy og ikke noe brukelig informasjon. 46
Figur 40: Fargebilde basert på de tre første prinsipalkomponentene for vegetasjon. Her er PC1 representert ved rød farge, PC2 ved grønn farge og PC3 ved blå farge. Største bidrag til trær og busker kommer fra PC3 (blått). 47
Figur 41: ROI på løvtre, bartre og gress. 47
Figur 42: Spektralsignaturer for løvtre, bartre og gress. 48
Tabell 8: Inndeling etter art, familie og treslag med tilhørende statistikk for hver klasse. 48
Figur 43: Plott av spektralsignatur for trær inndelt etter familie. 48
Figur 44: spektralsignaturer i VNIR for arter. 49

Figur 45: Spektralsignaturer i SWIR for løvtre, bartre og gress. 50
Figur 46: Spektralsignaturer i SWIR for familier av trær. 50
Figur 47: spektralsignaturer i SWIR for arter. SWIR har større verdier enn VNIR. Størst forskjell er det fra 1000 nm til ca.1150 nm. Sonene for vannabsorpsjon er tydelig synlige, i form av områder med lave verdier (rundt 1400 nm og 1900 nm) 50
Tabell 9: Separasjon, fra minst til størst. 51
Tabell 10: Innstillinger for SVM. 52
Figur 48: Resultat av SVM på vegetasjonsbildet. 52
Figur 49: Visuell sammenligning av SVM på VNIR (venstre) og på VNIR+SWIR (høyre). Spesielt gress, edelgran og syrin er mer tydelig definert i bildet til høyre. Samtidig har hengebjørk og gress blitt forvekslet enkelte steder. 52
Figur 50: Maximum likelihood basert på de fem første prinsipalkomponentene. 53
Figur 51: Resultat med en lik spektralvinkel på 0,1 radianer for alle klassene, basert på VNIR. 53
Figur 52: Test av ulike spektralvinkler for SAM. Bildet til høyre viser resultatet med en lik vinkel på 0,1 radianer for klassene. Bildet til venstre viser resultatet med ulike vinklet på klassene. Verdiene for vinklene er gitt i tabell 11 54
Figur 53: Resultat basert på en piksel i hver klasse. SVM har bare klart å predikere edelgran og hengebjørk. 55
Figur 54: Resultat med ROI på 1m radius for hver klasse. 55
Figur 55: Forklart varians per prinsipalkomponent. Detaljer for hver komponent er gitt i tabell 12. 57
Tabell 12: Detaljer for hver prinsipalkomponent. 57
..... 57
..... 58
Figur 56: Score-bilde for PC1 med tilhørende ladningsplott. PC1 forklarer hele 92,59% av variasjonen i datasettet. Det største bidraget til denne komponenten kommer fra det nærinfrarøde området (700 nm til 900 nm). Størst variasjon er det mellom vegetasjon og områder uten vegetasjon. 58
Figur 57: Scores-bilde for PC2 med tilhørende ladningsplott. Det største bidraget fra PC2 kommer fra tak og bygninger (hvit). Her er det størst bidrag fra bølgelengdene 600 til 750 nm, og under 500 nm. 58
Figur 58: Scores-bilde for PC3 med tilhørende ladningsplott. Her er det størst variasjon mellom urbane materialer som tak og bygninger, og andre områder som ikke er vegetasjon. Størst bidrag til denne komponenten kommer fra bølgelengder mellom 500 til 600 nm. 58
Figur 59: RGB-kompositтbilde av de tre første prinsipalkomponentene for VNIR. R = PC1, G = PC2, B = PC3. PC1 er representert ved det røde båndet og viser vegetasjon. 59
Figur 60: RGB-kompositтbilde av PC1, PC2 og PC3 for SWIR. R=PC1, G=PC2, B=PC3. Her er det store variasjoner i fargenyanser både mellom tak og bygninger. 59
Figur 61: Spektralsignaturer for de ni mest frekvente artene. 60
Figur 62: Spektralsignaturer for lind, hestekastanje og hengebjørk. Disse artene har veldig like spektralsignaturer, bortsett fra i området rundt 1000 nm. Dette området stammer fra datasettet SWIR. 61

Figur 63: Spektralsignaturer for spisslønn, eik og alm. 61
Figur 64: Spektralsignaturer for spisslønn og eik. Eik og spisslønn har så og si identiske spektralsignaturer i datasettet VNIR. Fra figuren kan det se ut som det er størst forskjell i øvre del av spekteret. Bare ekstremalverdiene er forskjellige ellers. 62
Figur 65: Spektralsignaturene til eik og spisslønn gjennom hele spekteret. Det er et lite område rundt 1000nm hvor det er litt forskjell i spektralsignaturene. Akkurat her kan informasjonen fra SWIR være verdifull for å skille disse artene fra hverandre. 62
Tabell 13: Kombinasjonene som ga lavest separasjon. 64
Tabell 14: Kombinasjonene som ga best separasjon. 64
Tabell 15: Kombinasjonene som ga dårligst resultat med VNIR+SWIR. 65
Tabell 16: Kombinasjonene som ga best resultat med VNIR+SWIR. 65
Tabell 17: Oversikt over klassene. 66
Tabell 18: Oversikt over parameterverdier som ble testet. 67
. 67
Tabell 19: Parameterinnstillinger. 67
Figur 66: Valideringskurver for parameteren C for logistisk regresjon og SVM (høyre) med balansert vekting og scoring = accuracy. 68
Figur 67: Valideringskurver for logistisk regresjon og SVM med balansert vekting og scoring lik f1macro. 69
Figur 68: Valideringskurver for logistisk regresjon og SVM med lik vekting (standardinnstillinger). 70
Figur 69: Læringskurver for logistisk regresjon og SVM med balansert vekting 71
Figur 70: Læringskurver for logistisk regresjon og SVM med lik vekting på klassene. 73
Tabell 20: Oversikt over testdatasettet. 74
Figur 71: Forvirringsmatriser med balansert vekting på klassene. Første kolonne er VNIR, andre kolonne er VNIR+SWIR. Tallene angir antall trær. 75
Figur 72: Forvirringsmatriser med lik vekting på klassene. Første kolonne er VNIR, andre kolonne er VNIR+SWIR. Tallene angir antallet trær. 76
Tabell 21: Precision, recall og f1 for logistisk regresjon med balansert vekting. Tall i parentes (grønn) indikerer endring fra VNIR. 77
Tabell 22: Precsion, recall og f1for SVM med balansert vekting. Tall i parentes (grønn) indikerer endring fra VNIR. 78
Tabell 23: Precsion, recall og f1for logistisk regresjon med lik vekting. Tall i parentes indikerer endring fra VNIR. 80
Tabell 24: Presisjon, recall, og f1 for SVM med lik vekting. Tall i parentes indikerer endring fra VNIR. 81
Tabell 25: kvalitetsmål for balansert vekting. 82
Tabell 26: Kvalitetsmål for lik vekting. 82
Figur 73: Spektralinformasjon for de klassene alm ble forvekslet med. Enheten er nm for bølgelengdene. 83
Figur 74: Spektralsignaturer til alm og eik med variasjonen i hver klasse. Enheten er nm for bølgelengdene. 84

Figur 75: Spektralsignaturene til alm og eik i VNIR og SWIR. Enheten er nm for bølgelengdene..... 84
Tabell 27: Tabellen viser ROI-separasjon for alm. 85
Figur 76:Spektralplott av klasse 1 spisslønn mot klasse 16 hestekastanje med variasjonen i hver klasse. Enheten er nm for bølgelengdene. 85

1 Innledning

1.1 Bakgrunn

Bakgrunnen for oppgaven er at byrådet i Oslo ønsker å vite status til grøntareal i Oslo kommune. Grøntareal er viktig for byområdene og kartlegging er blant annet viktig i forhold til planlegging. Oslo kommune har derfor anskaffet hyperspektrale flybilder for å undersøke mulighetene med slike data, og om de kan brukes til kartlegging av vegetasjon i urbane områder.

Kartlegging av vegetasjon i urbane områder blir tradisjonelt utført ved feltarbeid. Ulempen med feltarbeid er at det kan være både arbeidsomt og tidkrevende. I tillegg kan tilgjengelighet være et problem. Fjernmåling kan i så måte være et godt alternativ. Med fjernmåling kan man dekke store områder på kort tid og gi mulighet for data med høy romlig oppløsning. Urbane områder har tidligere vært en utfordring innen fjernmåling med tanke på kompleksiteten (Herold et al., 2004).

Med hyperspektrale bilder får man en informasjonsmengde og detaljgrad som er unik innenfor fjernmåling. Kombinasjonen mange smale nærliggende bånd over et bredt spekter og høy romlig oppløsning gir muligheten til å fange opp svært detaljerte nyanser i spektralinformasjonen til et materiale. Disse nyansene kan da brukes til å lettere skille ulike materialer fra hverandre. I tillegg til identifisering kan spektralsignaturen også fortelle noe om egenskapene eller tilstanden til et materiale. Ved å studere spektralsignaturen til et tre kan man eksempelvis få ut informasjon om helse, stress, vanninnhold, mm. (Harris Geospatial Solutions, u.å.-c).

1.2 Tema og problemstilling

Målet med denne oppgaven er å se hvor bra hyperspektrale bilder egner seg til å klassifisere trær i urbane områder. En pikselbasert fremgangsmåte skal benyttes. Det betyr at klassifiseringen baserer seg på kun på informasjonen fra pikslene og ikke objektenes form. De hyperspektrale bildene som brukes i denne oppgaven er tatt med to HySpex-sensorer som dekker hvert sitt spektrale område, VNIR og SWIR. VNIR dekker området fra synlig lys til nærinfrarødt (400 nm til 1000 nm), mens SWIR dekker området i kortbølget infrarødt (1000 nm til 2500 nm). Hyperspektrale data dekker vanligvis gjerne bare området VNIR. Få studier har blitt gjort på klassifisering av trær i urbane områder med data fra området 400 nm til 2500 nm. I forbindelse med klassifiseringen er det derfor ønskelig å svare på følgende problemstillinger:

- *Hvilke resultater for klassifisering på trær får man med kun VNIR?*
- *Hvor mye bedre blir klassifiseringen ved å også inkludere SWIR?*

Hyperspektrale bilder har tidligere vært mest brukt på skog og innen jordbruk, på homogene områder. Urbane områder består av et helt annet landskapsbilde, med en helt annen kompleksitet. Trær i urbane miljø er utsatt for store variasjoner i forhold til vekstforhold og påvirkning fra omgivelsene. Her er ikke lenger påvirkningen lik. På klassifisering av urbane områder står man derfor ovenfor helt andre utfordringer enn på et homogent område. Her kan den rike informasjonsmengden hyperspektrale bilder gir være avgjørende for resultatet.

1.3 Tidligere forskning

Hyperspektrale data har vært brukt i en rekke ulike studier på vegetasjon. Fra kartlegging av invasive (invaderende) planter i et våtmarksområde (Hestir et al., 2008), studier på avlinger (Bannari et al., 2006; Thenkabail, 2001), til overvåking av grønne tak i middelhavsklima (Piro et al., 2017). De fleste studiene som har vært gjort på vegetasjon har tatt for seg uniforme områder som avlinger, bestemte planter eller arter. Færre studier har vært utført på urbane områder. Likevel viser tidligere studier at hyperspektrale bilder kan brukes til klassifisering av urbane områder med høy nøyaktighet (Heiden et al., 2012; Stein et al., 2009). Stein et al. Oppnådde eksempelvis en nøyaktighet på 85,1% og en kappa-koeffisient på 0,82 på klassifisering av urbane overflater i München.

Hyperspektrale bilder har vært brukt til å klassifisere trær i urbane områder tidligere. En studie fra 2004 brukte hyperspektrale bilder i området 400 til 2500 nm til å klassifisere trær basert på spektralsignaturer (Xiao et al., 2004). Siden testområdet var lite ble ikke de hyperspektrale dataene atmosfærekorrigert. I stedet ble radians brukt. I denne studien oppnådde de en nøyaktighet på 94% på treslag og en gjennomsnittlig nøyaktighet på 70% på artsnivå, basert på 16 ulike arter. De oppnådde også en høyere nøyaktighet for trær med større trekroner og tett løvverk.

I 2011 ble en klassifisering på et mye større urbant område testet med hyperspektrale data (Jensen et al., 2012). Her ble det fokusert på et 150 km^2 urbant område med 500 trær. Det hyperspektrale datasettet dekket her et spektralt område fra 400 nm til 970 nm. Datamengden ble redusert ved å bruke prinsipalkomponentanalyse (PCA). Dette er en vanlig metode å bruke på hyperspektrale data.

Ved å bare bruke de seks første prinsipalkomponentene klarte de å skille mellom 10 ulike arter med en nøyaktighet på 82%. Ved å kombinere PCA med vegetasjonsindekser, middelverdi fra båndene og band forhold oppnådde de en nøyaktighet på 91,4%. I denne studien pekte de på to hovedårsaker til at enkelte trær ble feilklassifisert: (1) atypisk spektralsignatur og (2) at informasjon fra bakgrunnen ble med.

For klassifisering av trær i urbane områder er det flere hensyn å ta. I følge (Launeau et al., 2017) må man ta hensyn til (1) et stort antall ulike arter, (2) en stor variasjon i romlig fordeling, og (3) kompleks overlapp med andre materialer.

En annen faktor som også må tas hensyn til er at trekronene kan ha store variasjoner på grunn av lys- og skyggeforhold. En måte å ta hensyn til overlapp med andre materialer og varierende lysforhold er å bruke middelverdien til spektralinformasjonen på hvert tre.

En studie fra 2017 brukte hyperspektrale bilder og middelverdien for hvert tre (Launeau et al., 2017). I denne studien oppnådde de en nøyaktighet på 91% på klassifiseringen. Beste resultat ble oppnådd for isolerte trekroner og trær med tett løvverk. Her tok de hensyn til både kjemiske og fysiske egenskaper til trærne og bygget et lagvis bilde bestående av ulike spektralindekser. I denne studien brukte de dessuten den originale informasjonen til klassifiseringen, ikke reduksjon med PCA. I samme studie kom de også fram til at arter med like spektralsignaturer kan forveksles. Da er det heller bedre å gruppere dataene i vegetasjonsgrupper.

1.4 Oppsett og struktur

Masteroppgaven har følgende struktur:

Kapittel 1: Introduksjon

Beskriver bakgrunnen for oppgaven, tema, problemstillinger og tidligere forskning på feltet.

Kapittel 2: Teori

Beskriver teorien bak hyperspektrale bilder, om vegetasjon og stråling og hvilke algoritmer som har vært brukt.

Kapittel 3: Materialer og metode

Materialdelen presenterer datasett, instrumenter, samt hvilke programvarer og filformater har vært brukt. Metodedelen beskriver metoden som er brukt til å besvare problemstillingene.

Kapittel 4: Resultater og diskusjon

Presenterer resultatene fra analysene og klassifiseringen. Først presenteres resultatene fra en eksplorativ analyse på et lite område, deretter resultatene fra området som ble klassifisert. Funnene blir deretter diskutert og oppsummert. Til slutt presenteres utfordringer og svakheter med dataene.

Kapittel 5: Konklusjon

Konklusjonen oppsummerer funnene og svarer på problemstillingene. Til slutt presenteres forslag til videre arbeid.

Vedlegg

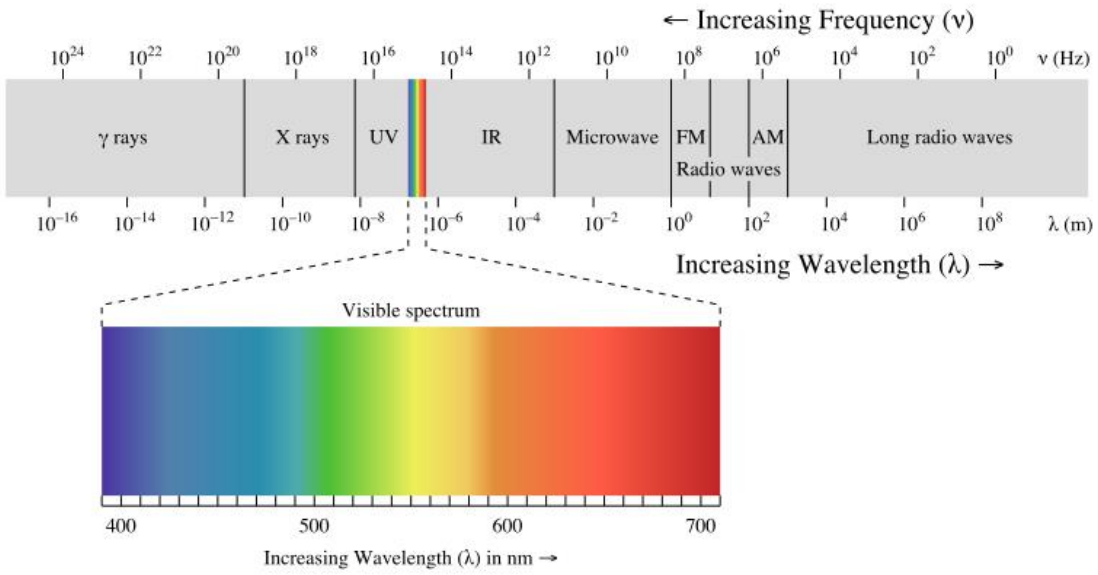
Spektralsignaturer til trær og python-script.

2 Teori

For å forstå hva hyperspektrale bilder er og hvordan en hyperspektral sensor fungerer, er det nødvendig med noe grunnleggende forståelse for lys og hvordan det reagerer med ulike materialer.

2.1 Lys og elektromagnetisk stråling

Lys kan ses på som partikler eller bølger. Elektromagnetisk stråling kan beskrives som en strøm av masseløse partikler som beveger seg i lysets hastighet (NASA, 2013a). Disse partiklene kalles for fotoner. Hvert foton har en bestemt bølgelengde ut i fra energinivået sitt. Vanligvis brukes bølgelengde til å beskrive elektromagnetisk stråling. Bølgelengde uttrykkes vanligvis i micrometer (10^{-6} m) eller nanometer (nm), der 1nm = 10^{-9} m.



Figur 1: Det elektromagnetiske spekteret.

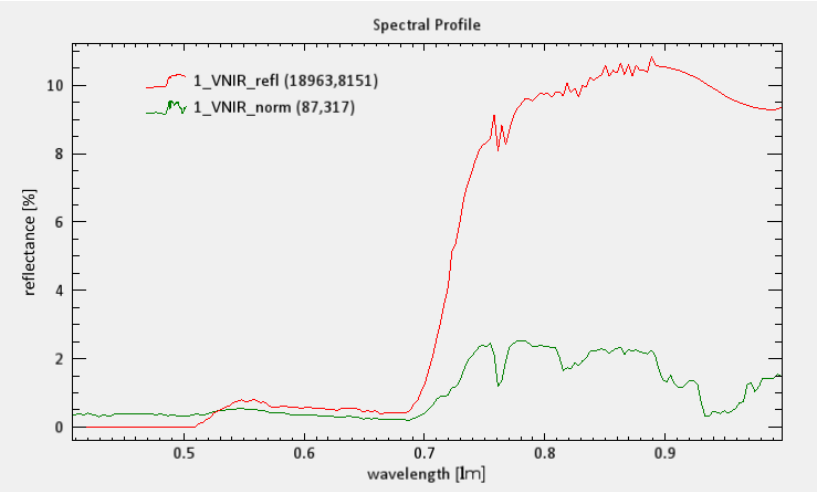
Det elektromagnetiske spekteret viser elektromagnetisk stråling inndelt i klasser etter bølgelengder. Synlig lys ligger i området 400 til 700nm (Lowe et al., 2017) og utgjør bare en liten del av det elektromagnetiske spekteret (se figur 1).

Stråling fra sola kan bli reflektert, absorbert, eller transmittert (Borengasser et al., 2007) Når stråling treffer et objekt vil vanligvis noe av strålingen bli absorbert og noe bli reflektert. Transmittert betyr at strålingen passerer gjennom et materiale og får en endring i hastighet (NASA, 2013b). Andel reflektert stråling fra et objekt i forhold til innkommende stråling er

gitt ved reflektansen. Reflektansen til et materiale kommer an på hva materialet består av. Ulike materialer reflekterer og absorberer stråling ulikt. Ved å plotte reflektans mot bølgelengde får man spektralsignaturen til et materiale. Områder med lav reflektans kalles for absorbsjonsbånd. Posisjonen og styrken på slike absorbsjonsbånd, samt formen på selve spektralkurven kan brukes til å identifisere og skille mellom ulike materialer (Smith, 2006).

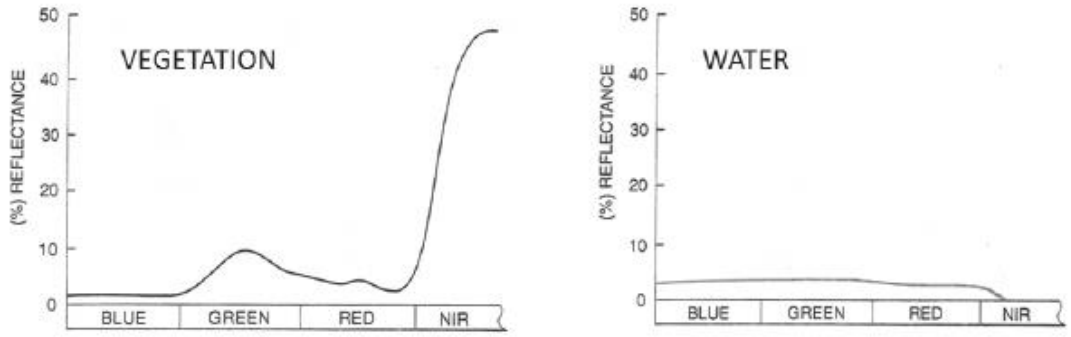
2.2 Radians og reflektans

Radians og reflektans er to begrep som brukes mye om hverandre men som ikke må forveksles. Radians er det sensoren måler direkte og kan defineres som "mengden reflektert lys instrumentet detekterer i hver bølgelengde" (Smith, 2006). På vei gjennom atmosfæren vil noe av strålingen fra sola bli absorbert. Det gjør at observert radians er mindre enn utgangspunktet ved toppen av atmosfæren. I tillegg vil sensoren fange opp lys som er spredt i atmosfæren. Radians avhenger altså av påvirkning gjennom atmosfæren, av belysningen (både intensitet og retning), samt orienteringen og posisjonen til objektet (Borengasser et al., 2007).



Figur 2: Reflektans (rød) og normalisert radians (grønn) for samme pikselkoordinat plottet mot bølgelengde.

2.3 Vegetasjon og stråling



Figur 3: Spektralsignaturer for vegetasjon og vann (Campbell & Wynne, 2011).

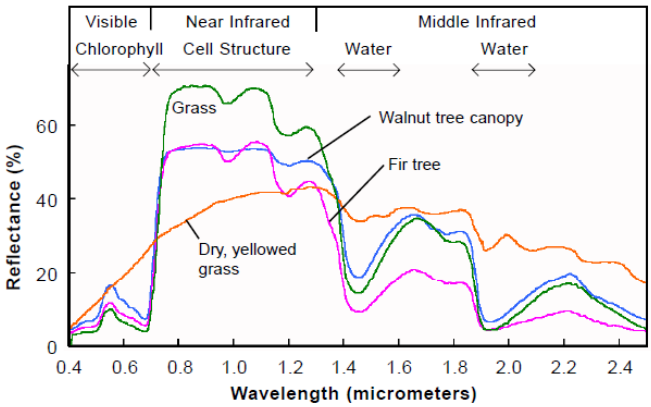
Figur 3 viser forholdet mellom reflektans og bølgelengde for to vanlige overflatetyper, vegetasjon og vann. Områder med lave verdier representer bølgelengder hvor det foregår absorpsjon. Ulike vegetasjonstyper kan skilles på bakgrunn av spektralsignaturene. Vegetasjon har en særegen spektralsignatur som skiller seg ut fra andre typer overflater. I det synlige området fra 400 til 700 nm finner man en karakteristisk reflektanstopp i området for grønt lys. Dette skyldes effekten fra klorofyll i fotosyntesen (CRISP, 2001). Klorofyll absorberer mer rødt og blått lys enn grønt. Dermed ser en sunn grønn plante "grønn" ut for øyet vårt.

I området mellom synlig lys og nærinfrarødt lys, ca. 680 nm til 730 nm stiger reflektansen brått (Harris Geospatial Solutions, u.å.-a). Dette området omtales som «red edge», eller den «røde kanten» på norsk. Mengden klorofyll i en plante påvirker formen på spekteret direkte. Jo høyere innhold av klorofyll, jo sterkere absorpsjon i området for rødt lys og jo bredere blir dette absorpsjonsområdet. Dette medfører også at den røde kanten forskyves mot høyere bølgelengder (SEOS, u.å.).

Vegetasjon har høyest reflektans i det nærinfrarøde området (NIR) mellom 700 og 1300 nm. Reflektansen i dette området skyldes først og fremst indre cellestruktur i bladene (Campbell & Wynne, 2011). Planter har forskjellig bladstruktur. I følge (Smith, 2006) blir reflektansen i det nærinfrarøde området påvirket av faktorer som type art, stress og tilstand.

Reflektansen avtar med økende bølgelengde ovenfor det nærinfrarøde området, bortsett fra to karakteristiske vannabsorpsjonsbånd nær 1400nm og 1900nm (Smith, 2006). Når et blad visner avtar produksjonen av klorofyll og dermed minker absorpsjonen i det blå og særlig det røde området. I tillegg synker reflektansen i det nærinfrarøde området, slik at spektralsignaturen får en noe flatere form. Dette medfører at bladet får en rød eller gulaktig farge (Richards, 2013). Slik kan spektralsignaturen også fortelle noe om tilstanden til vegetasjonen.

Vegetasjon har generelt lav reflektans i det synlige området og mye høyere reflektans i det nærinfrarøde området. Dette er derfor de mest nyttige bølgelengdeområdene å studere for å se på vegetasjon. For å si noe om vanninnhold bør også områder der vannabsorpsjon foregår tas med.



Figur 4: Reflektansspektra for ulike typer grønn vegetasjon. Tørr vegetasjon har en karakteristisk flatere kurve (Smith, 2006).

2.3.1 NDVI

NDVI står for Normalised Difference Vegetation Index og er en vegetasjonsindeks som sier noe om frodigheten til vegetasjonen.Vegetasjonsindekser baserer seg på to eller flere bølgelengdeområder i det elektromagnetiske spekteret, for å analysere bestemte egenskaper til vegetasjon. Verdiene for NDVI går fra -1 til 1, der verdier rundt null indikerer områder som ikke er vegetasjon. Sunn grønn vegetasjon befinner seg i øvre halvdel av denne skalaen med verdier rundt 1. Formelen for NDVI er gitt ved (NASA Earth Observatory):

$$NDVI = \frac{NIR - RØD}{NIR + RØD} \tag{2.1}$$

2.4 Definisjon av fjernmåling

Det finnes en rekke ulike definisjoner på fjernmåling. Generelt kan fjernmåling defineres som vitenskapen og teknologien som karakteriserer objekt uten direkte kontakt mellom sensor og objekt (NASA Earth Observatory, 1999). Fjernmåling går ut på å måle reflektert stråling fra jordas overflate med en sensor montert på en plattform som fly eller satellitt. Sensoren på en slik plattform måler elektromagnetisk stråling ved bestemte bølgelengdeområder kalt bånd (Congedo, 2017). Målingene brukes deretter til å danne et bilde av landskapet. Sensorer på slike plattformer dekker ofte områder utenfor området for synlig lys (400 -700 nm) (Richards, 2013). Dermed kan en slik sensor fange opp detaljer øyet ikke kan oppfatte. De mest vanlige optiske fjernmålingssystemene registrerer data fra det synlige området til det infrarøde området (400 – 2500 nm), i det såkalte optiske spekteret. Det optiske spekteret kan deles inn i fire deler (Harris Geospatial Solutions, u.å.-d):

- Synlig område: 400nm til 700 nm.
- Nær-infrarødt (NIR) 700nm til 1300 nm.
- Kortbølget nærinfrarødt 1 (SWIR 1) 1300 nm til 1900 nm.
- Kortbølget nærinfrarødt 2 (SWIR 2) 1900 nm til 2500 nm.

Innenfor fjernmåling skiller man mellom to hovedtyper sensorer: aktive og passive. En aktiv sensor utstråler sin egen energikilde og er dermed ikke avhengig av sollys for å fungere. Laser og radar er eksempler på slike sensorer. En passiv sensor genererer ingen stråling selv og er dermed avhengig av ekstern stråling som sollys for å gjøre opptak (Borengasser et al., 2007). Passive sensorer er derfor sensitive for variasjoner i solinnstråling, samt vær og vind. Oppløsningen til en sensor påvirker resultatet direkte. Begrepet oppløsning brukes i flere sammenhenger:

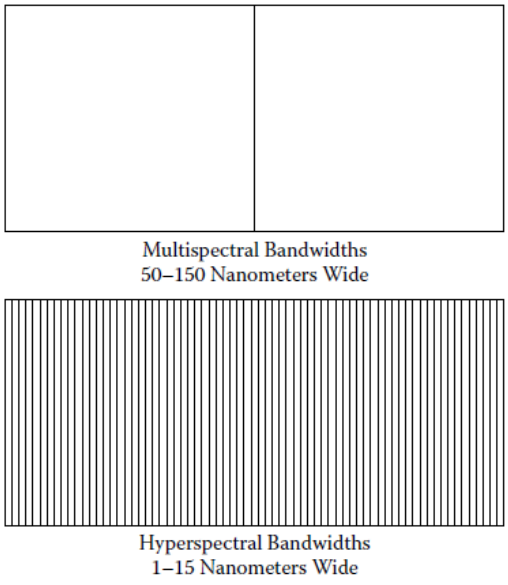
- Romlig oppløsning defineres vanligvis som arealet hvert pikselelement representerer på bakken (Dick, 2015)
- Spektral oppløsning er bredden på et spektralbånd definert av to bølgelengder.

2.5 Hyperspektrale sensorer

Sensorer kan også deles inn måten innsamlingen av dataene er gjort på. De vanligste sensorene er multispektrale og hyperspektrale sensorer. Richards (2013) beskriver en sensor som hyperspektral når sensoren har flere enn 10 bånd, og multispektral når den har færre. Antall bånd alene avgjør dog ikke om en sensor er hyperspektral. Det er måten målingene er gjort på, med mange smale kontinuerlige målinger som kjennetegner en hyperspektral sensor. Dersom en sensor måler stråling i 20 brede separerte bånd er ikke sensoren hyperspektral, men multispektral. Et bilde fra en hyperspektral sensor kan altså kjennetegnes ved at det har (Grahn & Geladi, 2007):

- Full spektralinformasjon for hver piksel.
- Svært mange smale nærliggende bånd, ofte mer enn 100.

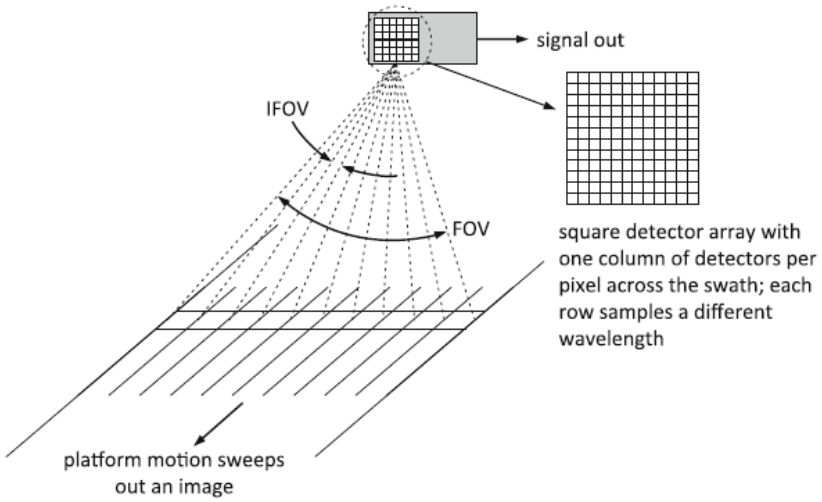
Typisk båndbredde på hyperspektrale bilder er 1 til 15 nm, mens multispektrale bilder har båndbredde på 50-120 nm (Borengasser et al. 2007, s.17). Et hyperspektralt bilde kan derfor gi mye mer info om overflaten enn et multispektralt bilde (Shippert, 2003).



Figur 5**:** Multispektrale versus hyperspektrale bånd (Borengasser et al., 2007).

2.6 Pushbroom-skanner

HySpex-sensoren i er en pushbroom-skanner. En pushbroom-skanner kjennetegnes ved at den skanner et område linje for linje på tvers av flyretningen (se figur 6). Skanneren består av en rekke detektorer som tar opp en smal stripe bildedata om gangen. Detektorene utgjør sammen en bildebrikke, også kalt CCD. Bredden på stripen tilsvarer synsvinkelen til sensoren, ofte betegnet som FOV (Field Of View). Stripebredden hver enkelt detektor dekker omtales som IFOV (Instantaneous Field Of View) og tilsvarer pikselstørrelsen. En hyperspektral sensor bruker en to-dimensjonal bildebrikke. Mens den første bildebrikken registrer bildedata på tvers, registrerer den andre dimensjonen bølgelengder for hver piksel (Richards, 2013). Resultatet er at hver piksel inneholder full spektralinformasjon.

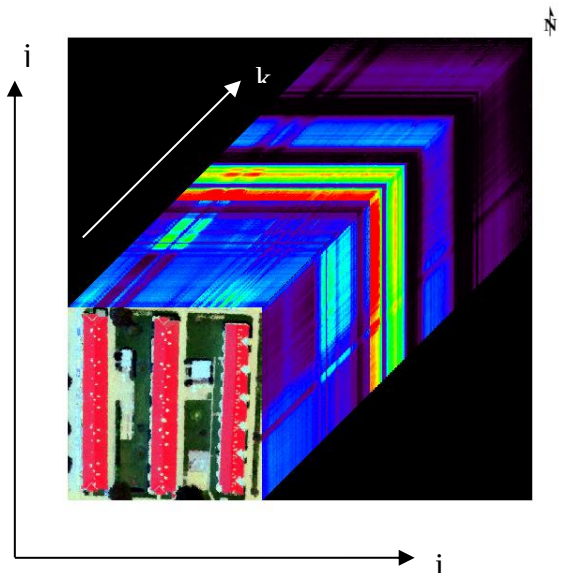


Figur 6**:** Prinsippet ved pushbroom-skanning (Richards, 2013).

2.6.1 Hyperkube

Siden hver piksel i et hyperspektralt kamera har full spektraloppløsning definerer bildene fra et hyperspektralt kamera en hyperkube. Hyperkuben har to romlige dimensjoner i og j, og en spektral dimensjon k (spektral oppløsning).

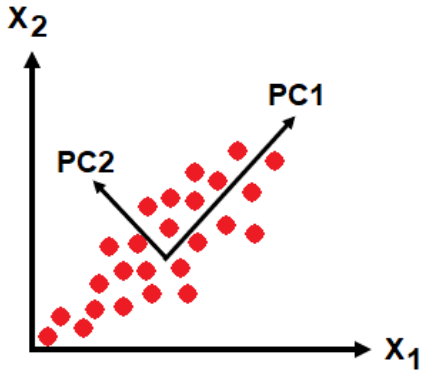
Figur 7 viser en hyperkube. Jo varmere farger i den spektrale dimensjonen k, jo høyere verdi.



Figur 7: Illustrasjon av en hyperkube.

2.7 PCA (Prinsipalkomponentanalyse)

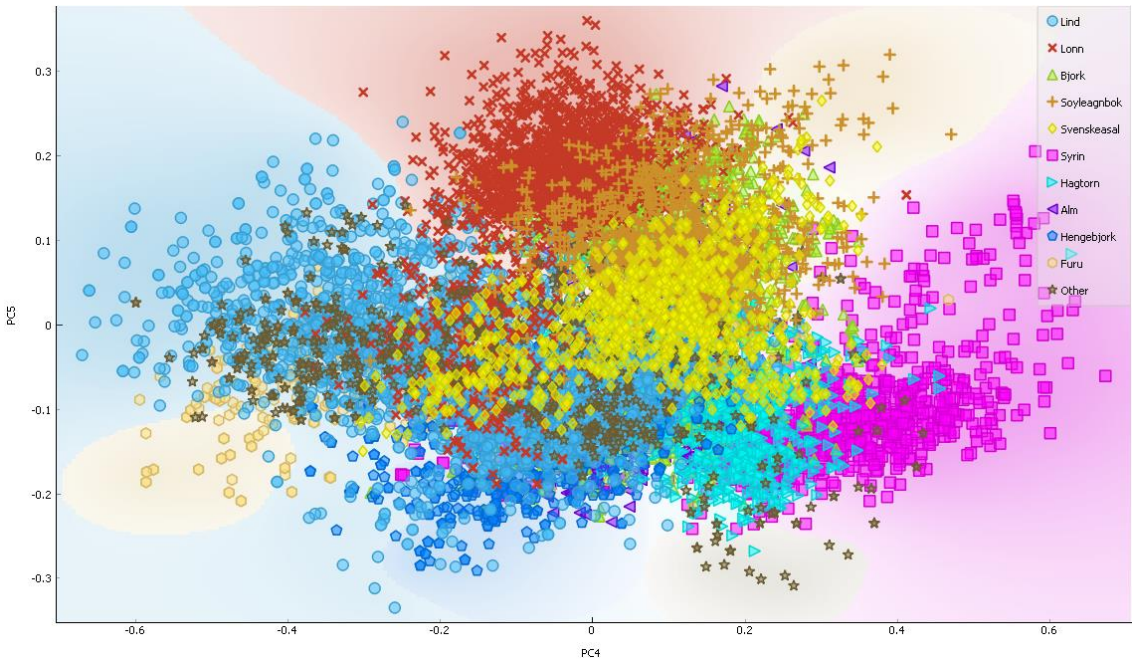
Hyperspektrale bilder kan bestå av flere hundre bånd, i tillegg kan det være høy korrelasjon mellom nærliggende bånd. Høy korrelasjon mellom bånd betyr at verdiene i båndene ligner mye på hverandre. Prosessering av slike data kan være både tidkrevende og utfordrende på grunn av mengden data. En vanlig metode for å redusere datamengden til et datasett og samtidig beholde signifikant informasjon er å bruke PCA (prinsipalkomponentanalyse). Flere studier har utforsket klassifisering på bakgrunn av PCA eller mulighetene med PCA på hyperspektrale data (Jensen et al., 2012; Pervez & Khan, 2015; Rodarmel & Shan, 2002; Torbick & Becker, 2009; Wang & Chang, 2006). PCA går ut på å danne et minimum antall ukorrelerte variabler til å forklare størsteparten av variasjonen i et datasett. Metoden ser på de originale båndene som står for størst variasjon i pikselverdier og finner en optimal lineær kombinasjon av disse (Campbell & Wynne, 2011).



Figur 8: Prinsippet med PCA.

Ved PCA blir det opprinnelige koordinatsystemet overført til et koordinatsystem der aksene står ortogonalt på hverandre (se figur 8). Aksene i det nye koordinatsystemet kalles for prinsipalkomponenter. Prinsipalkomponentene (egenvektorene) tilsvarer altså retningene med størst variasjon i datasettet. Hver egenvektor har en tilhørende eigenverdi som indikerer størrelsen på variasjonen (Hamilton, 2014). Prinsipalkomponentene blir ofte forkortet til PC1 for prinsipalkomponent 1, PC2 for prinsipalkomponent 2, osv. PC1 forklarer størsteparten av variasjonen i datasettet, PC2 forklarer den nest største, osv. (Lillesand et al.). Siden prinsipalkomponentene står ortogonalt på hverandre er de ukorrelerte.

Sammenhengen mellom de opprinnelige variablene og prinsipalkomponentene kalles for ladninger (loadings). Ladninger er elementene i en egenvektor (Holland, 2008). Jo høyere verdi, jo høyere bidrag til prinsipalkomponenten (Bruker Daltonics, u.å.). Verdiene i det nye koordinatsystemet kalles for scores. Disse kan illustreres i et scores-plott:



Figur 9: Scores-plott for ulike trær.

2.8 Klassifisering

Hensikten med klassifisering på et bilde er å kategorisere alle pikslene til bestemte klasser (Harris Geospatial Solutions, 2018a). Det finnes to hovedtyper av klassifisering, styrt klassifisering og ikke-styrt klassifisering. Dersom klassifiseringen foregår uten bruk av treningsdata brukes betegnelsen ikke-styrt klassifisering. Klassifiseringen blir da beregnet kun på bakgrunn av statistikk. Brukeren angir kun antall klasser det skal deles inn i.

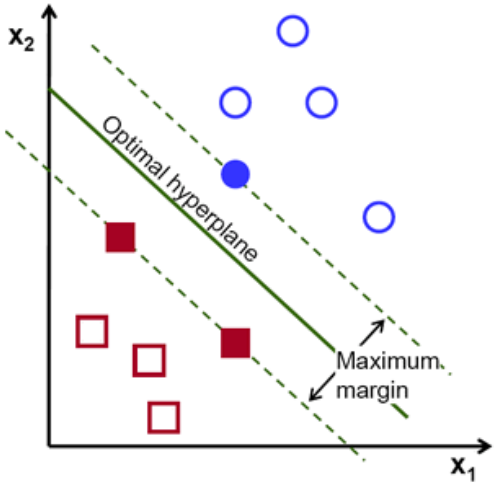
Styrt klassifisering går ut på at brukeren gir treningsdata i form av representative områder til algoritmen som utfører klassifiseringen. Dataene blir brukt til å trene opp algoritmen, derav navnet treningsdata. En piksel blir da plassert i klassen med høyest sannsynlighet. Treningsdataene kan komme fra områder i bildet eller fra en importert fil. Det finnes en rekke ulike algoritmer, hver med sin egen måte å tildele pikselverdiene inn i klasser på. En gjennomgang av de algoritmene som er brukt i denne oppgaven følger.

2.8.1 SVM (Support Vector Machine)

SVM er en algoritme som finner en optimal lineær overflate for å separere treningsdataene, et såkalt hyperplan. Det optimale hyperplanet er det som maksimerer avstanden til

treningsdataene, slik at det størst mulig separasjon mellom dem ((Harris Geospatial Solutions, u.å.-b). I 2D er hyperplanet en linje som deler et plan i to deler. Marginen er den vinkelrette avstanden mellom linjen og de nærmeste datapunktene. Bare de nærmeste punktene er relevant for å definere linjen. Disse kalles «support vectors», de støtter eller definerer hyperplanet (OpenCV, 2016).

SVM har en rekke parametre for tilpasning av algoritmen, blant annet kernel, C-verdi, gamma og terskelverdi.



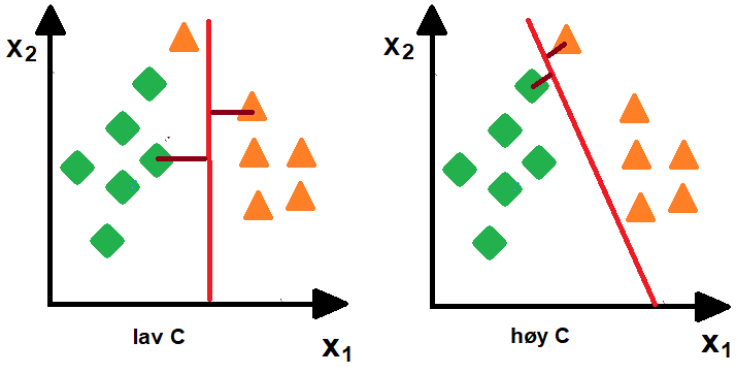
Figur 10: Prinsippet ved SVM (OpenCV, 2016).

Kernelen

Kernelen er en funksjon som tar datapunkt i lavere dimensjoner og transformerer de til en høyere ordens dimensjon for å lettere separere dataene. Det er en likhetsfunksjon. Flere typer kan brukes, for eksempel lineær, polynom og radiell (RBF). Hvilken kernel som bør brukes kommer helt an på datasettet.

C-parameteren (penalty-parameter)

C-parameteren kan ses på som kostnaden ved feilklassifisering. En stor verdi for C gir mindre marginer til hyperplanet, mens en liten verdi gir hyperplanet større marginer. Større verdier for C øker kostnaden ved feilklassifiserte punkt og fører til en mer nøyaktig modell (Harris Geospatial Solutions, u.å.-b).



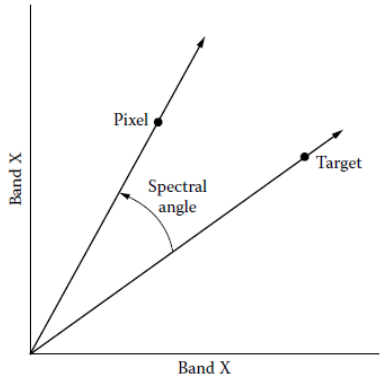
Figur 11: Lav C-verdi (til venstre) gir store marginer, mens høy C-verdi (høyre) gir lave marginer.

Gamma

Gammaverdien brukes for enkelte kerneltyper som RBF, polynom og sigmoid.
Den styrer hvor langt fra separasjonslinjen påvirkningen fra treningsdataene går. En høy gamma betyr at bare de nærmeste punktene blir brukt i beregningen, mens en lav gamma betyr at også punkt lengre unna en tenkt separasjonslinje blir med (Patel, 2007).

2.8.2 SAM (Spectral Angle Mapper)

Spectral Angle Mapper (SAM) sammenligner et spekter mot et referansespekter ved å se på vinkelen mellom spektrene. Algoritmen betrakter spektrene som vektorer i rommet med felles origo, og beregner spektralvinkelen mellom dem. Størrelsen på vinkelen indikerer graden av likhet mellom materialene som sammenlignes. Jo mindre vinkel, jo høyere korrelasjon og jo bedre match. Siden en endring i lysstyrke påvirker størrelsen men ikke retningen på vektoren, blir SAM relativt lite påvirket av endringer i lysstyrke for et material (Borengasser et al., 2007).



Figur 12: Prinsippet ved SAM (Borengasser et al., 2007).

2.8.3 Logistisk regresjon

Logistisk regresjon er en maskinlæringsalgoritme som ser på sammenhengen mellom en binært avhengig variabel og en eller flere uavhengige variabler. At variabelen er binær betyr at den kun kan ha to mulige verdier, for eksempel 0 eller 1. Målet med logistisk regresjon er å finne den modellen som passer best til å beskrive forholdet mellom den avhengige variabelen og de uavhengige variablene (MedCalc Software, 2017). Modellen bygger på en sannsynlighetsmodell hvor sannsynligheten for at en positiv hendelse skal inntreffe kan betegnes ved p. Oddsen for en bestemt hendelse kan skrives som (Rascha, 2016):

$$\frac{p}{1-p} \tag{2.2}$$

Sannsynligheten for at en prøve tilhører en bestemt klasse blir bestemt med en sigmoid-funksjon:

$$\phi(z) = \frac{1}{1+e^{-z}} \tag{2.3}$$

Hvor z er den lineære kombinasjonen av vekter og forklaringsvariabler x_0, …, x_m.

$$z = w_0 X_0 + w_1 x_1 + \cdots + w_m X_m = \sum_{j=0}^{m} x_j w_j = w^T x \tag{2.4}$$

Sigmoid-funksjonen tar reelle verdier og transformerer de til verdier mellom 0 og 1.

C-verdien

C-verdien regulerer modellen. En liten C-verdi begrenser modellen, mens en større verdi gir mer frihet til modellen (Pedregosa et al., 2011). Høy c-verdi gir en så kompleks modell som algoritmen tillater. Standardverdi for C er 1.

L1 og L2

L1 og L2 er straffeparametre som prøver å redusere kompleksiteten til modellen ved å straffe store individuelle vekter og minimere feilen mellom predikerte og faktiske verdier. De straffer uønsket oppførsel i vektene. Forskjellen mellom de er måten de straffer på. L1 regulering straffer ved å se på absoluttverdien av vektene, mens L2 legger til straff tilsvarende kvadratsummen av vektene (Rascha, 2016). Formlene for L1 og L2 kan skrives:

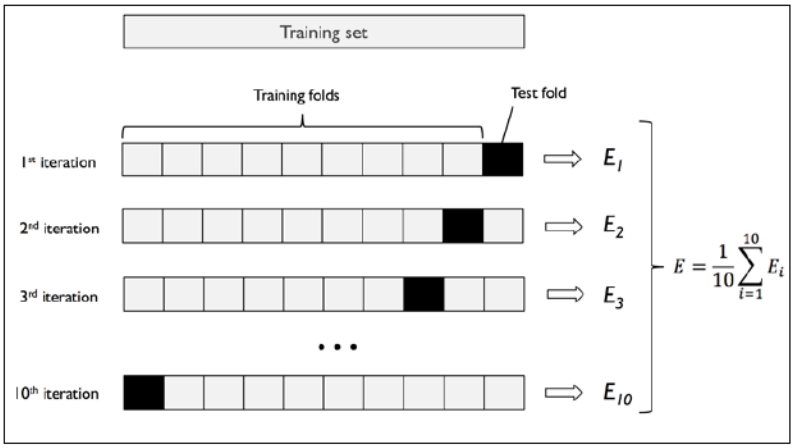
$$L1: \quad |w| = \sum_{j=1}^{m} |w_j| \tag{2.5}$$

$$L2: \quad |w|^2 = \sum_{j=1}^{m} w_j^2 \tag{2.6}$$

der w er vekt og j indikerer vektnummer.

2.8.4 Kryssvalidering

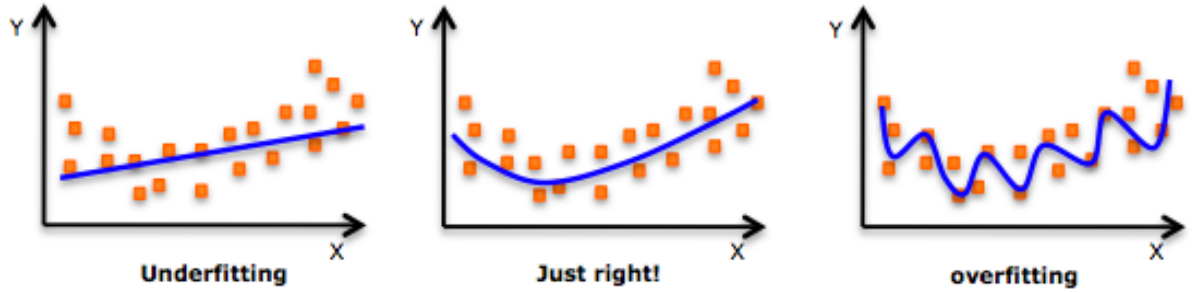
Kryssvalidering er en mye brukt metode for å evaluere ytelsen til en klassifiseringsmodell. Kryssvalidering går ut på å dele datasettet i k deler med lik størrelse. Modellen blir deretter trent på alle delene bortsett fra en. Prosessen repeteres k-1 ganger slik at hver del er testsett en gang og treningssett alle andre ganger (Pedregosa et al., 2011). Poenget med kryssvalidering er å gi modellen flere treningsprøver. Det fører vanligvis til en mer nøyaktig og robust modell. Figur 13 viser prinsippet med kryssvalidering. Her er datasettet delt i 10 deler slik at en del brukes til testing mens de ni andre delene brukes til trening. Prosessen repeteres 10 ganger.



Figur 13: Prinsippet med kryssvalidering ((Pedregosa et al., 2011).

2.8.5 Over- og undertilpasning

Et vanlig problem med en klassifiseringsmodell er at den kan lide av underfitting (undertilpasning) eller overfitting (overtilpasning). Undertilpasning betyr at modellen er for simpel, mens overtilpasning betyr at modellen er for kompleks for treningsdataene (Pedregosa et al., 2011). En modell overtilpasser dersom den yter bedre på treningsdataene enn testdataene. En slik modell vil ikke generalisere bra på usette data. Undertilpasning er tilfellet dersom en modell yter dårlig på treningsdataene. Modellen kan da være for simpel til å fange opp mønsteret i datasettet. Det er viktig å ta hånd om denne problematikken siden det fører feilaktige prediksjoner. Flere metoder kan brukes for å håndtere og identifisere over- og undertilpasning i et datasett. Disse blir gjennomgått i de neste delkapitlene.

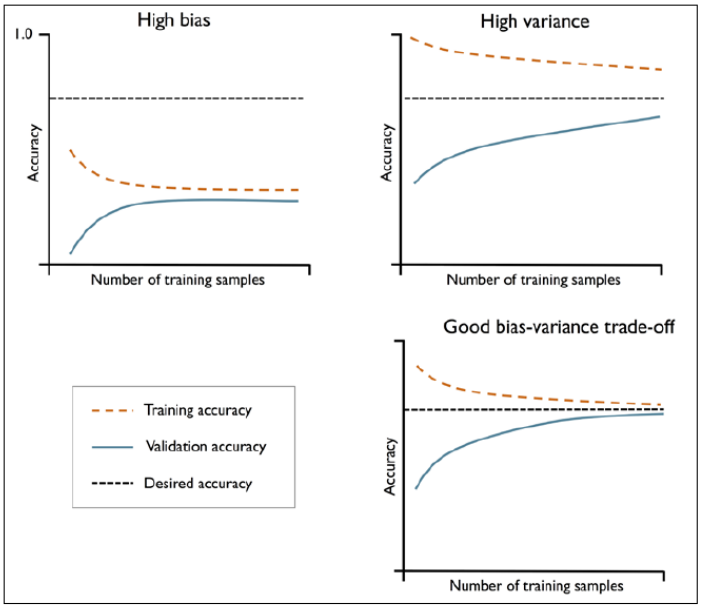


Figur 14: over- og undertilpasning (quora, 2017).

Figur 14 illustrerer problemet med over- og undertilpasning. Grafen til venstre viser et tilfelle av underfitting. Modellen er her for simpel. I midten ses en modell som gjør en god jobb med å generalisere dataene. Grafen til høyre viser et tilfelle av overtilpasning. Modellen er her veldig kompleks og tilpasser seg dataene i for stor grad.

2.8.6 Lærings- og valideringskurver

To verktøy for å evaluere tilfeller av overtilpasning og undertilpasning er lærings- og valideringskurver. Både lærings- og valideringskurver bruker treningsdatene som utgangspunkt og utfører kryssvalidering på disse for å sammenligne nøyaktigheten. En læringskurve kan brukes for å identifisere om modellen lider av høy bias eller høy variasjon, og om modellen kan ha nytte av større datasett (Rascha, 2016). Høy bias er forbundet med undertilpasning, mens høy varians er forbundet med overtilpasning.

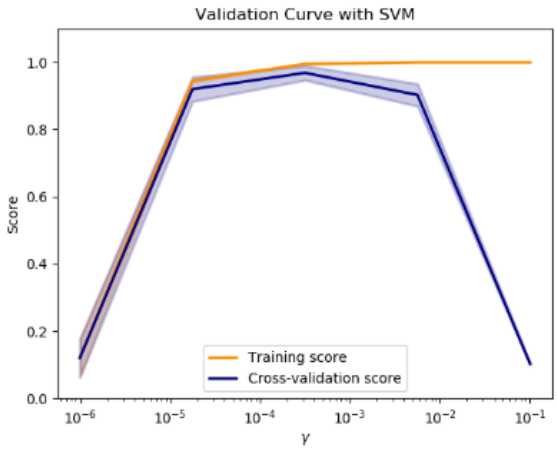


Figur 15: Prinsippet med læringskurver (Rascha, 2016).

Figur 15 viser prinsippet med læringskurver. På grafen øverst til høyre er det et gap mellom kurvene. Det indikerer at modellen lider av høy varians og er et tegn på overtilpasning. Tre måter å håndtere dette tilfellet er å redusere kompleksiteten til modellen, øke reguleringsparameterne eller samle inn mere treningsdata (Rascha, 2016).

Grafen øverst til venstre viser et tilfelle av undertilpasning. Modellen har lav nøyaktighet for både trening og validering. Undertilpasning kan håndteres ved å justere reguleringsparameterne slik at modellen får mer frihet til å tilpasse seg treningsdataene.

Til å vurdere over- eller underfitting kan i tillegg valideringskurver benyttes. En valideringskurve viser hvordan forskjellige parameterverdier til påvirker nøyaktigheten. Dersom både verdiene for treningsscore og valideringsscore er lav er modellen undertilpasset. Dersom det er et gap mellom treningsscoren og valideringsscoren er det en indikasjon på at modellen overtilpasser. En ideell verdi for parameteren er der hvor både trening og validering scorer høyt på nøyaktighet.



Figur 16: Eksempel på valideringskurve (Pedregosa et al., 2011).

2.8.7 GridSearchCV til finjustering av modellen

Valg av parameterverdier kan ha stor påvirkning på resultatet av klassifiseringen. Det er derfor hensiktsmessig å teste ulike kombinasjoner av parametere for å se hvilke som passer best til datasettet. Parameterverdier på et datasett trenger nødvendigvis ikke passe like bra til et annet datasett (Pedregosa et al., 2011). Funksjonen GridSearchCV tester automatisk ut gitte parameterverdier og finner den optimale kombinasjonen som gir best resultat. Den optimale kombinasjonen blir funnet ved kryssvalidering over gitte parameterverdier. I tillegg til parameterverdier kan funksjonen også håndtere antall inndelinger i kryssvalideringen og type score.

2.8.8 Evaluering av klassifiseringen

Flere metoder kan brukes til å evaluere klassifiseringsresultatet, blant annet nøyaktighet, forvirringsmatrisa, kappa-koeffisienten, mm. Evaluering av klassifiseringen er viktig for å vurdere resultatet og kvaliteten på modellen.

Forvirringsmatrisa (confusion matrix)

En måte å vurdere resultatet av klassifiseringen er å se på forvirringsmatrisa. En forvirringsmatrise viser sammenhengen mellom predikerte og sanne verdier. Den består av en kvadratisk matrise lik antallet klasser. Diagonalen viser antall korrekt klassifiserte element i hver klasse. Verdier utenfor diagonalen er element som er feilklassifisert. Feilklassifiserte element representerer utelatelsesfeil (omision error) eller følgefeil (comission error). Utelatelsesfeil tilsvarer kolonneelement utenfor diagonalen ((Lillesand et al., 2004). Ved å unngå å ta med elementet fra den sanne klassen har det blitt begått en utelatelsesfeil. Radelement utenfor diagonalen tilsvarer følgefeil. Det er element som har blitt plassert i en klasse de ikke hører hjemme. Dermed har det blitt begått en følgefeil. Tabell 1 viser prinsippet med forvirringsmatrisa.

Tabell 1: Prinsippet med forvirringsmatrisa.

Predikert / Faktisk	K1	K2	K3	Rad total
K1	**60**		0	60
K2	6	**50**	11	67
K3	0	7	**40**	47
Kolonne total	66	57	50	

Den totale nøyaktigheten fra forvirringsmatrisa kan fås ved å dividere summen av elementene i diagonalen på totalt antall element i hver klasse (rad total helt til høyre).

For å ta høyde for tilfeldigheter kan Kappa-koeffisienten benyttes. Kappa-koeffisienten sammenligner observert nøyaktighet med forventet nøyaktighet (tilfeldighet). Formelen for kappa kan i enkelthet settes opp som (Campbell & Wynne, 2011):

$$\kappa = \frac{observert - forventet}{1 - forventet} \tag{2.7}$$

der observert er verdien total nøyaktighet fra forvirringsmatrisa, og forventet er et estimat på bidraget fra ren tilfeldighet på den totale nøyaktigheten. Kappa-koeffisienten går fra -1 til 1 der verdier over 0,8 indikerer generell god overenstemmelse (Pedregosa et al., 2011). Verdier på 0 eller lavere betyr ingen overenstemmelse (ren tilfeldighet). En vanlig inndeling av kappa-verdiene kan ses i tabell 2.

Tabell 2: inndeling for kappa-koeffisienten (Richards, 2013).

Kappa-koeffisient	Klassifiseringen kan ses på som:
under 0,4	Dårlig
0,41-0,6	Moderat
0,61-0,75	God
0,76-0,8	Utmerket
0,8 og over	Nesten perfekt

Accuracy (nøyaktighet)

Accuracy er definert som andelen korrekt klassifiserte tilfeller (Rascha, 2016) og brukes til å evaluere kvaliteten på en modell. På ubalanserte datasett kan accuracy være et misvisende mål på treffsikkerheten til en modell (Rascha, 2016). For å ta høyde for ubalanserte datasett kan andre mål enn accuracy benyttes. Disse blir gjennomgått i de påfølgende avsnittene.

Precision (presisjon)

Presisjon er evnen en algoritme har til å ikke klassifisere en positiv klasse som negativ. Med positiv og negativ menes i denne sammenhengen algoritmens prediksjonsevne. Formelen for presisjon er gitt ved ((Pedregosa et al., 2011):

$$Precision = \frac{t_p}{t_p + f_P} \tag{2.8}$$

hvor t_p står for "true positive" og f_p står for "false positive". Med t_p menes antall klassifisert som positiv og som faktisk tilhører klassen positiv (sann positiv). Antallet som feilaktig er klassifisert som positiv er f_p. Precision er dermed gitt som antallet sanne positive dividert på totalt antall element i den positive klassen.

Recall (Gjenkalling)

Recall er evnen algoritmen har til å finne alle positive prøver. Formelen for recall er (Pedregosa et al., 2011):

$$recall = \frac{t_p}{t_p + f_n} \tag{2.9}$$

der t_p står for "true positive" og f_n står for "false negativ". Med f_n menes antall element klassifisert som negativ men som faktisk tilhører klassen positiv.

F1-verdien

F1-verdien kalles også for balansert f-score og kan ses på som et vektet harmonisk middel av presisjon og recall, med lik vekt på disse. F1 er et mål på nøyaktigheten til en test. Verdiene for f1 går fra 0 til 1, der 1 er beste verdi og 0 er dårligste. Formelen for f1 er (Pedregosa et al., 2011):

$$F1 = 2 \times \frac{precision \times recall}{precision + recall} \tag{2.10}$$

Vekting av parametre

Ved klassifisering med flere enn to mulige utfall kan presisjon, recall og F1 beregnes som et vektet middel på klassene med argumentet «average». Dette argumentet spesifiserer ulike måter å vekte midlingen på. Midlingen kan utføres blant annet som «micro» eller «macro». Ved macro blir verdien beregnet uavhengig for hver klasse og deretter midlet, slik at klassene blir vektet likt. Ved micro blir midlingen utført med lik vekt på klassene. Her blir bidragene fra hver klasse aggregert og deretter midlet. Midling ved micro gjør dermed at den dominerende klassen vil ha stor innflytelse på resultatet. Midlet presisjon (PRE) med macro og micro kan beregnes som følger (Rascha, 2016):

$$PRE_{micro} = \frac{TP_1 + \cdots + TP_k}{TP_1 + \cdots TP_k + FP_1 + \cdots + FP_k} \tag{2.11}$$

$$PRE_{macro} = \frac{PRE_1 + \cdots + PRE_k}{k} \tag{2.12}$$

der TP står for sanne positive, FP står for falske positive, $PRE_1 + \ldots + PRE_k$ er gjennomsnittlig score i hver klasse, og k er antall klasser. Dersom et datasett er ubalansert (ulikt antall i hver klasse) kan vekting benyttes. En annen måte å håndtere ubalanserte datasett er å tildele en større straff på feilprediksjoner i minoritetsklasser (Pedregosa et al., 2011). Hvilken metode som passer best kommer helt an på datasettet. Det finnes ingen universal metode som passer alle typer datasett.

3 Materialer og metode

Dette kapitlet tar for seg materialer og metode og er inndelt i to delkapitler med tilhørende navn. Materialer omhandler datasettet, instrumenter, programvare og filformater som er benyttet for å komme frem til metoden. For å få en forståelse for datasettet og hvilke instrumenter som er brukt, blir metoden presentert etter materialdelen. I metodedelen blir selve metoden presentert og drøftet. Veien fram til en endelig metode blir gjennomgått og veivalg blir begrunnet. Metoden beskriver fremgangsmåten for å besvare problemstillingen på en best mulig måte.

3.1 Materialer

3.1.1 Spesifikasjoner HySpex

To HySpex-sensorer er brukt til innsamling av data. Disse er produsert av Norsk Elektro Optikk AS. De to sensorene dekker hvert sitt spektrale område, VNIR og SWIR. VNIR står for Visible and Near Infrared, SWIR er forkortelse for ShortWave Infrared. Spesifikasjonene for sensorene er listet opp i tabell 3.

Tabell 3: HySpex-spesifikasjoner (Norsk Elektro Optikk AS, u.å.).

	VNIR-1800	SWIR-384
Spektral rekkevidde	400-1000 nm	1000-2500 nm
Antall romlige piksler	1800	384
Blendertall	F2.5	F2.0
Maks åpningsvinkel (FOV)	17°	16°
Piksel FOV across track/ along track	0,16 / 0,32 mrad	0,73 / 0,73 mrad
Spektral oppløsning	3,26 nm	5,45 nm
Antall bånd	186	288
Radiometrisk oppløsning	16 bit	16 bit
Dynamisk rekkevidde	20000	7500
Maks bildefrekvens	260 fps	400 fps

3.1.2 Prosjektområdet

Innsamling av hyperspektrale flybilder er utført av TerraTec AS på oppdrag for Oslo Kommune. Flyvingen er gjort den 19.07.2017. Det er flydd ni flystriper innenfor prosjektområdet ved en høyde på 1300 meter over terrenget. Med denne flyhøyden er den romlige oppløsningen for sensorene på 0,3 m for VNIR og 0,7 m for SWIR. Prosjektområdet dekker et område på 12,95 km^2 over sentrale deler av Oslo.

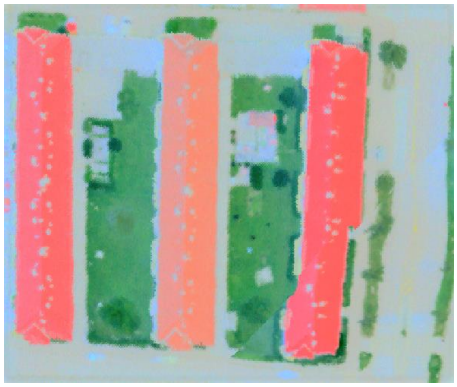
Figur 17: Oversikt over prosjektområdet.

Tabell 4: Spesifikasjoner for flyvingen (TerraTec AS, 2017).

Spesifikasjoner	
Leverandør	TerraTec AS
Lokasjon	Oslo
Områdedekning	12,95 km^2
Sensor	HySpex SWIR-384 og HySpex VNIR-1800
IMU	Micro IRS IE-IPAS-uIRS
GNSS-mottaker	Topcon Legacy E
Plattform	FW/ RW (type og reg-nr)
Høyde over terreng	1300
Flyhastighet	130 kt
Åpningsvinkel (FOV)	8 (SWIR), 8,5 (VNIR)
Linjeavstand	300 m
Brutto Stripebredde	506 m
Datum	WGS84 UTM sone 32

I sin rapport (TerraTec AS, 2017) for hyperspektral datainnsamling nevner TerraTec AS flere utfordringer knyttet til innsamlingen av de hyperspektrale flybildene. Selve innsamlingsområdet lå i innflygningssonen til Gardermoen flyplass, dermed ble det noen minutters ventetid mellom enkelte flystriper. Det førte til at flylinjene har noe forskjellig solinnstråling.

Enkelte deler av innsamlingen har blitt gjennomført i situasjoner med utfordrende turbulensforhold. Dette har medført at graden av roll og/eller pitch til flyet har oversteget det som gyrorammen klarte å kompensere. Resultatet er at noen områder har blitt forvrengte på grunn av roll- og/eller pitch-feil.



Figur 18: Noen deformasjoner på hus. Husene på bildet
er i virkeligheten rette. Bildet er fra det normaliserte datasettet.

3.1.3 Gjennomgang av hyperspektrale datasett

De hyperspektrale flybildene er levert i tre ulike versjoner fra TerraTec AS; radians, reflektans og normaliserte verdier. Alle dataene er ortorektifisert og georeferert. Radiansdatasettet er levert som en mosaikk for hvert bølgelengdeområde (VNIR og SWIR). Mosaikken er satt sammen på bakgrunn av flystripene.

Den andre versjonen som er levert inneholder reflektansverdier. Disse dataene er atmosfærekorrigert ved modellen ATCOR-4 og er inndelt i flystriper for hvert bølgelengdeområde. Atmosfærekorreksjonen er utført av TerraTec As. Absorpsjonseffekter fra atmosfæren er da fjernet og gjør at dataverdiene vises som reflektans i stedet for radians.

Det medfører at piksler som kun har bidrag fra atmosfæren har verdien 0 etter at korreksjonen er utført.

Den tredje varianten som TerraTec AS har levert er et normalisert datasett. Dette datasettet er delt inn i en mosaikk for hvert bølgelengdeområde (VNIR og SWIR). Normaliseringen er utført på en mosaikk bestående av radiansverdier. Hensikten med normaliseringen var å minimere skyggeforskjeller og slik skape et godt utgangspunkt for klassifiseringen (TerraTec AS, 2018). Normaliseringen går ut på at hver pikselverdi har blitt dividert på summen av verdiene i alle bånd for samme piksel. Formelen som er brukt som utgangspunkt for normaliseringen er gitt ved (Yu et al., 1999):

$$\frac{X_{ij}}{\frac{1}{K} + \sum X_{ij}} \tag{3.1}$$

Hvor K er båndnummer og j er radiansverdi for piksel X_i.

Dersom radiansen er lav (som i skyggeområder) vil formelen gi et lavere tall i nevneren. Det normaliserte datasettet vil derfor ha noe høyere verdier i skyggeområder.

3.1.4 Valg av hyperspektrale datasett

Hver versjon av datasettene har sine styrker og svakheter. Det atmosfærekorrigerte datasettet inneholder reflektansverdier. Fordelen med det er all påvirkning fra atmosfæren er korrigert for, slik at verdiene viser de faktiske strålingsegenskapene til materialer på overflaten. I utgangspunktet var det derfor ønskelig å bruke dette datasettet. Derimot hadde datasettet også en del ulemper slik det var levert. For det første var selve datamengden problematisk å håndtere. Totalt hadde reflektanssettet en størrelse på over 4,7 TB. Selv om bare deler av datasettet skulle brukes var fortsatt filstørrelsen så stor at det var en utfordring å håndtere, både for programvare og med tanke på lagringsplass. Med slike datamengder gikk også analysering og bearbeiding tregt.

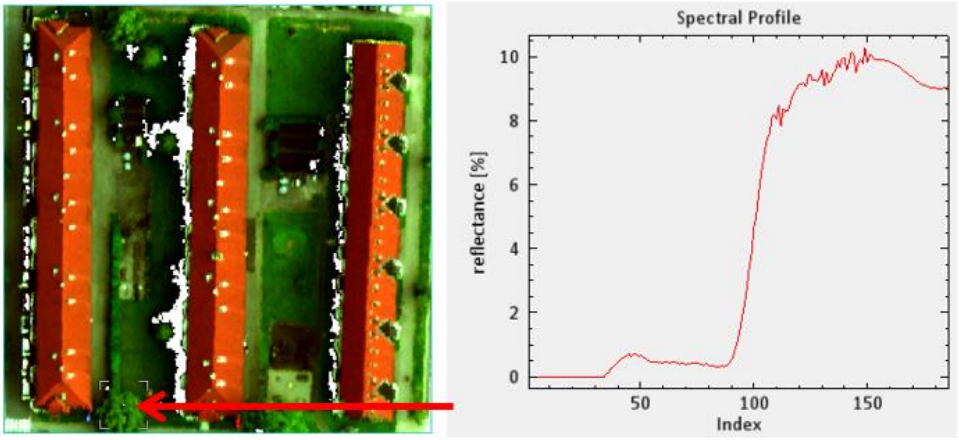
Skygger var også en faktor. Skygger kan redusere mengden lys som treffer et område. Trær, bygninger eller andre topografiske hindringer kan kaste skygge og påvirke lysmengden som treffer en sensor. Effekten av skygge gjør at lysmengden til en berørt piksel blir redusert over alle bølgelengder ((Smith, 2006). Skygger kan også påvirke egenskapene til vegetasjon. I løvverk som befinner seg i skygge kan mengden klorofyll øke for å kompensere for mangelen på lys, mens i områder med mye lys kan andre pigmenter øke (Lichtenthaler et al., 2013).

I data hvor det ikke er korrigert for skygger bør derfor dette tas hensyn til. En måte å gjøre dette på er å skille mellom piksler fra skygge og ikke skygge.

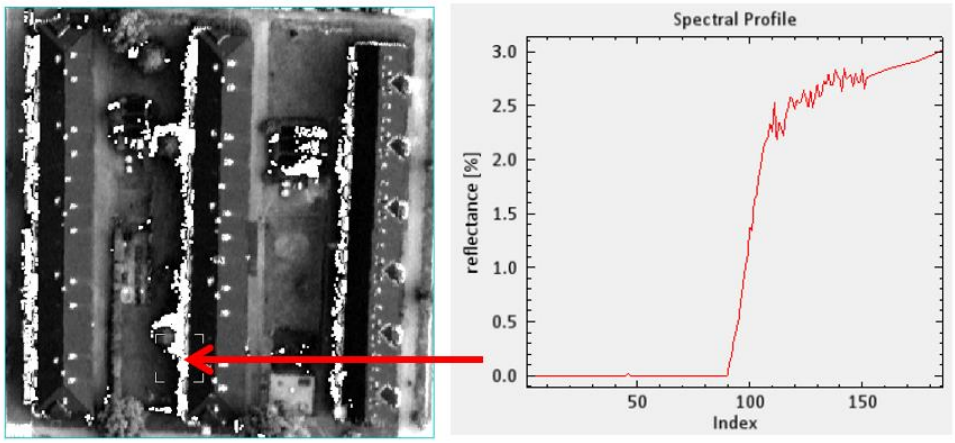
Den neste ulempen med reflektansdataene var at de var levert som flystriper, totalt ni i alt. Siden testområdene strakk seg over flere flystriper måtte reflektansdataene først settes sammen til en mosaikk dersom de skulle brukes.

Den siste ulempen med reflektansdataene var at de var redigert ved at de var atmosfærekorrigerte. Piksler som kun hadde bidrag fra atmosfæren var satt til verdien null. Problemet var at hvilke piksler som var satt til null varierte fra bånd til bånd. Det ville gjøre det vanskelig å hente ut spektralinformasjon uten at piksler med nullverdi ble med.

Hadde nullverdiene vært konsekvente over alle båndene ville håndteringen av de blitt lettere. Figur 19, 20 og 21 illustrerer problemet.



Figur 19:Pikselen på dette bildet har nullverdier fra omtrent bånd 30 til bånd 1.Hvite områder er områder hvor pikselinformasjonen er fjernet og pikselen har verdien null.



Figur 20: Spektralinformasjonen for en piksel i et skyggeområde vist i bånd. For denne pikselen er informasjonen fram til rundt bånd 90 fjernet og satt til null. Som man kan se er områdene med nullverdier forskjellig fra figur 3.19.

Figur 21: Bånd 31 i reflektansssettet.

Enhver bearbeiding av et datasett vil være et usikkerhetsmoment i seg selv. Ved å manipulere dataene får man innført en ny potensiell feilkilde. På grunn av denne oppgavens begrensede tidsspenn og de nevnte utfordringene, ble reflektanssettet derfor ikke brukt.

Både det originale og det normaliserte datasettet var levert ferdig sammensatt som mosaikk. Det gjorde at disse settene var lettere å håndtere og gjøre analyser på. Filstørrelsene var også mindre enn det atmosfærekorrigerte datasettet. Ulempen med det originale settet var først og fremst skyggeproblematikken, som måtte tas hensyn til. I det normaliserte datasettet var skyggeproblematikken tatt hånd om. Dette datasettet var altså klargjort til å gjøre analyser på.

Siden skygger var tatt hånd og det var ferdig sammensatt som en mosaikk ble det normaliserte datasettet brukt i denne oppgaven.

Figur 22:Samme område før og etter normalisering. Originalbilde til venstre, normalisert bilde til høyre. Skygger er minimert på bildet til høyre, samtidig kan det se ut som om skyggekorreksjonen har gjort skyggeområdene lysere enn omgivelsene. Begge bildene er vist med båndkombinasjonen R, G, B = 55,41,21.

3.1.5 Testområder feltarbeid

I denne oppgaven ble det brukt to ulike feltdatasett. Det ene ble samlet inn på feltarbeid i forbindelse med masteroppgaven. Dette settet består av data fra ni små testområder på 100x100 m fordelt innenfor prosjektområdet. Kriteriene for valg av områder var flere. Det var viktig at områdene var representative for bybildet i Oslo. Fra Oslo kommune sin side var det ønskelig at områdene skulle dekke typiske områder som finnes i et bymiljø som boligområder, industriområder og bygårder/bakgårder.

Figur 23: Oversikt over de ni små testområdene sammen med prosjektområdet (gult).

Beskrivelse av feltarbeidet

Feltarbeidet ble utført i flere omganger i september og oktober 2017 mens trærne fortsatt hadde bladverk på. Hensikten med feltarbeidet var å samle inn bakkeobservasjoner av trær for validering mot de hyperspektrale bildene. På et flybilde kan mye informasjon være skjult av tett vegetasjon. I tillegg til validering gir bakkeobservasjoner derfor et godt bilde av hvordan et område ser ut på bakkenivå, spesielt under trekroner.

Registreringen av trær var basert på URBAN EEA sin prosedyre for kartlegging av trær (Skarpaas, 2017). Følgende ble registrert på feltarbeidet:

1. *Alle trær over 5 meter i høyde.*

2. *Diameter i brysthøyde (DBH) ved 1,35m.* Dersom et tre hadde flere stammer i denne høyden ble hver stamme registrert som split. Dersom split var på bakkenivå ble stammene registrert som individuelle trær.

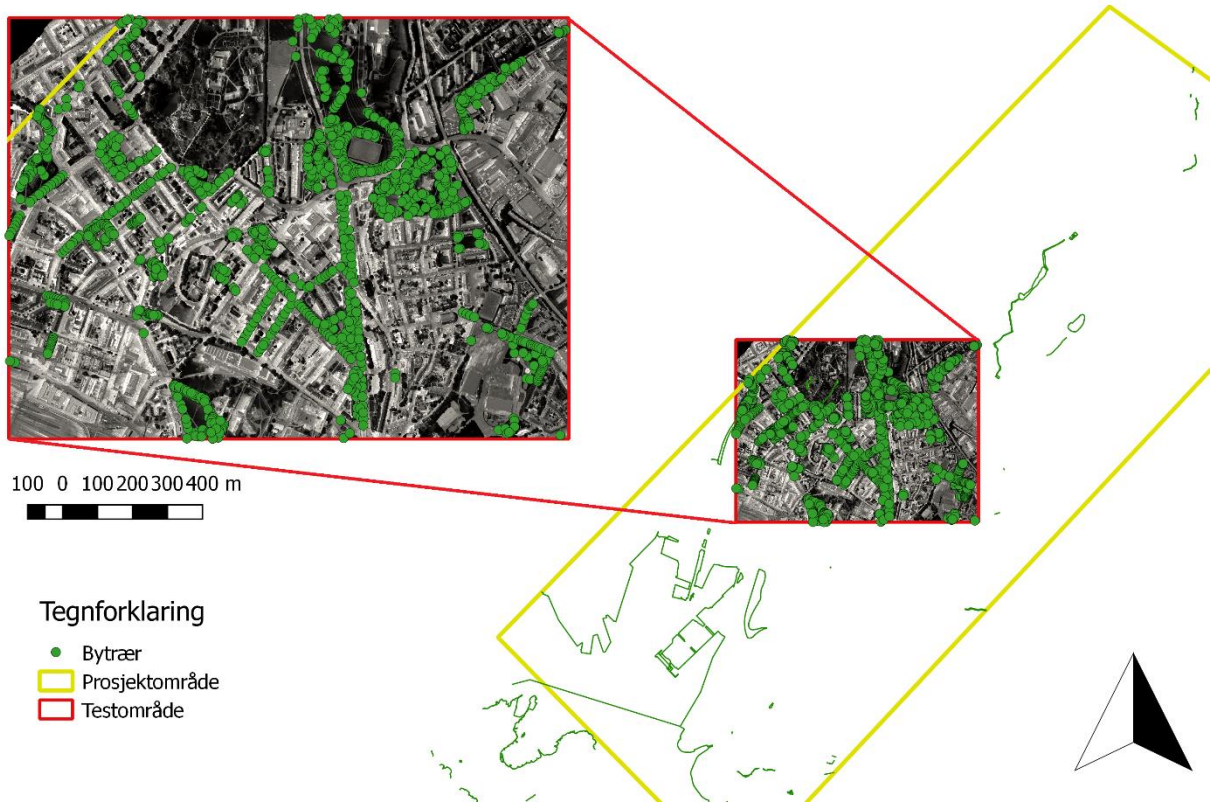
3. *Trehøyde.* Total trehøyde fra bakken til toppen av treet.

4. *Kronediameter.* Kronebredde i to retninger.

Basert på disse feltmålingene ble en database opprettet. Totalt ble det gjort 159 registreringer fra 25 ulike arter.

3.1.6 Stort testområde

I tillegg til de 9 testområdene ble det valgt ut et stort testområde på 1600 m x 1200 m til klassifiseringen. Feltdataene for dette området ble hentet fra Oslo kommune sitt register over bytrær. I utgangspunktet var det 1318 bytrær innenfor testområdet, 805 av disse ble brukt.



Figur 24: Oversikt over det store testområdet.

3.1.7 Programvarer og tilleggsmoduler

Denne delen presenterer ulike programvarer, tilleggsmoduler og filformater som er brukt i oppgaven. Først blir programvarene beskrevet.

ENVI

ENVI er et program beregnet på analyse, prosessering og visualisering av enhver type fjernmålingsdata, også bildedata. Programmet har en rekke innebygde verktøy som gjør det mulig å trekke ut informasjon fra bildedata på en enkel og rask måte (Harris Geospatial Solutions, 2018b). I denne oppgaven ble ENVI 5.2 til å bearbeide datasettene, generere PCA-bilder, hente ut spektralinformasjon, samt teste ulike algoritmer for styrt klassifisering.

QGIS

QGIS står for Quantum GIS og er en gratis og åpen GIS-klient til å analysere, redigere, visualisere og prosessere geografisk informasjon (Quantum GIS, 2018). Både raster- og vektordata er støttet i QGIS**.** I programmet er det også mulighet for å legge til programtillegg som gir mer funksjonalitet. I denne oppgaven ble QGIS versjon 2.18.3 benyttet til å legge inn data fra feltarbeidet som vektorpunkt, bearbeide feltdata, genere polygoner, utføre romlig join med datasett, samt eksportere data til CSV-filer for videre analyser.

Python

Python er et objektorientert, interaktivt programmeringsspråk med en åpen kildekode (Python Software Foundation, 2018a). I denne oppgaven blir Python 3.6 brukt til å bearbeide spektraldataene, visualisere informasjon fra datasettene, samt utføre klassifiseringen av trær. Flere ulike moduler i Python ble brukt. En kort gjennomgang av disse følger.

Scikit-learn

Scikit-learn er en åpen og gratis maskinlæringsmodul til Python. Denne modulen inkluderer verktøy for analyse, klassifisering, regresjon, og preprosessering av data (Pedregosa et al., 2011).

Pandas

Pandas er et åpent bibliotek for analyse og manipulering av data i Python (McKinney, 2010). Biblioteket inkluderer blant annet verktøy for å lese, strukturere og skrive data i forskjellige formater som CSV, Excel, tekstfiler, mm.

Seaborn

Seaborn er et åpent bibliotek for visualisering av statistikk i Python. Det bygger på matplotlib og inkluderer støtte for pandas og NumPy (Waskom et al., 2017).

Matplotlib

Matplotlib er et bibliotek for plotting av data i Python. Det inkluderer blant annet funksjoner for å lage histogram, spredningsplott, mm (Hunter, 2007).

NumPy

NumPy er et grunnleggende bibliotek for vitenskapelige beregninger i Python. Det inkluderer blant annet verktøy for å jobbe med multidimensjonale objekter (Oliphant, 2006).

3.1.8 Filformater

CSV

CSV (Comma Separated Value) er et filformat som brukes til å dele og konvertere tabelldata mellom ulike regnearkprogrammer. Dataene er strukturert i rader og kolonner der kolonnene er separert med skilletegn. Skillegnet kan for eksempel være komma, mellomrom eller semikolon (Shafranovich, 2005).

Py

Py er et format for script i programmeringsspråket Python. En fil i formatet py kan lages i en vanlig teksteditor, men er avhengig av Python for å leses (Python Software Foundation, 2018b).

Shape

Shape er et åpent vektorbasert filformat utviklet av ESRI. Shape-filer lagrer ikke-topologiske vektordata sammen med tilhørende attributtdata. Geografiske trekk kan representeres ved enten punkt, linje eller polygon (ESRI, 1998).

TIFF

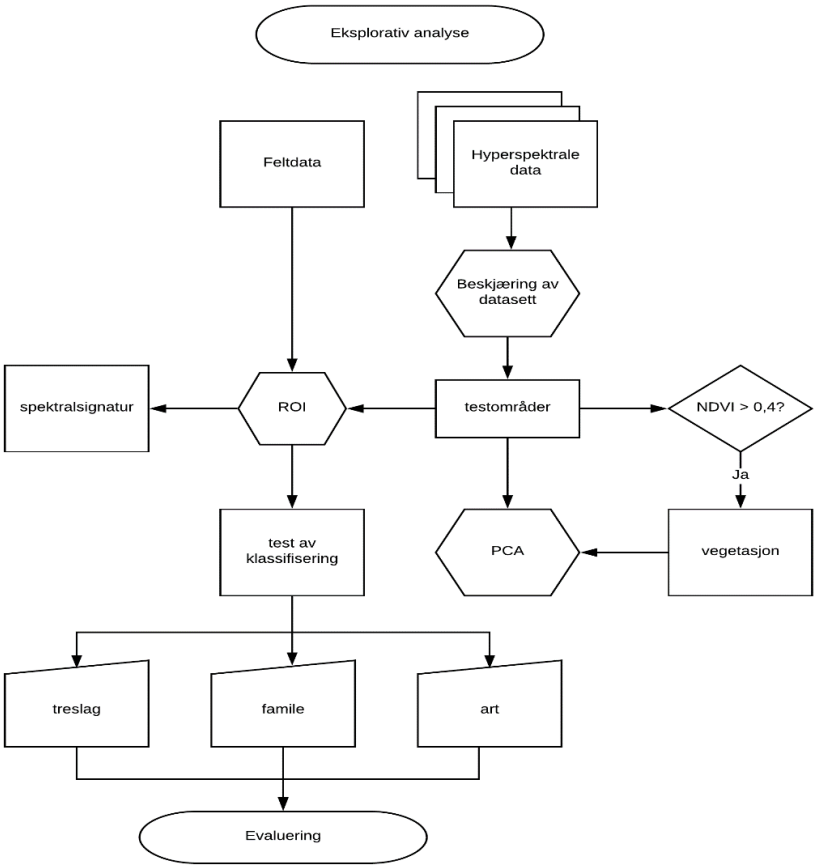
TIFF (Tagged Image File Format) er et filformat for lagring og utveksling av rasterbilder. Formatet støtter både ukomprimerte og komprimerte filer. En TIFF-fil er vanligvis organisert i en header-fil og en bildedel. Header-filen består av flere tags som peker til informasjonen i filen (FileFormat).

3.2 Metode

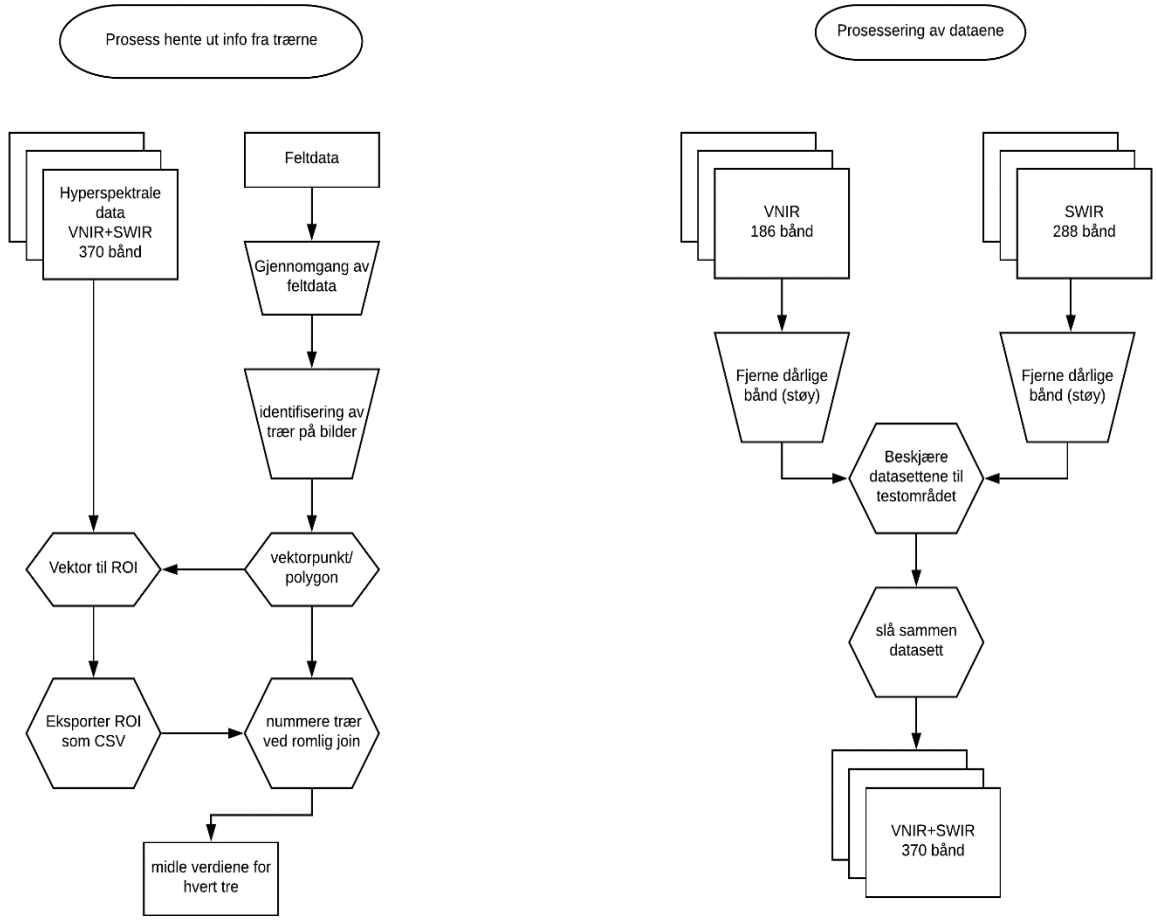
Metodedelen beskriver metodene som er brukt, fra prosessering til klassifisering. Formålet med metoden var å finne ut hvor bra klassifisering med hyperspektrale data kan bli, og besvare problemstillingene som var:

- *Hvilke resultater for klassifisering på trær får man med kun VNIR?*
- *Hvor mye bedre blir klassifiseringen ved å også inkludere SWIR?*

Til å besvare denne problemstillingen ble to vinklinger benyttet. Først en eksplorativ analyse som utforsket datasettene og en hovedmetode som beskriver fremgangsmåten for klassifiseringen. Hensikten med den eksplorative analysen var å gjøre seg kjent med de hyperspektrale dataene, og teste ulike analysemetoder for å se hvilke som kunne egne seg til videre bruk på det store testområdet. Fremgangsmåten for den eksplorative analysen er vist i figur 25.



Figur 25: Flytdiagram for den eksplorative analysen.



Figur 26: Flytdiagrammer for prosesseringen.

Figur 26 illustrerer prosesseringen på dataene for å klargjøre de til analyser. Venstre flytdiagram viser prosesseringen som er gjort på feltdataene, mens høyre flytdiagram viser prosesseringen på de hyperspektrale dataene. En nærmere gjennomgang av stegene i prosesseringen følger.

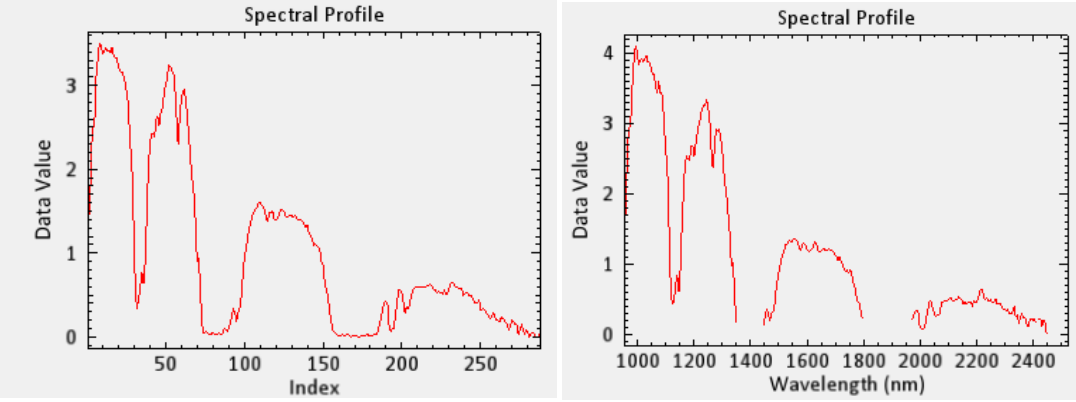
3.2.1 Preprosessering og bearbeiding av hyperspektrale data

Før noe som helst av analyser kunne utføres på de hyperspektrale dataene måtte de gjennom flere steg med preprosessering og bearbeiding. Preprosesseringen er utført av TerraTec AS og innebærer ortorektifisering og georeferering. Ortorektifiseringen og georefereringen er basert på en 20 cm digital overflatemodell innhentet 19.07.2017 fra en ALS570 Laserskanner. Offset på heading, roll og pitch er korrigert for ved å bruke manuelle kontrollpunkter på et 10 cm ortofoto fra 2013 som referanse (TerraTec AS, 2017). Alle dataene som er brukt i denne oppgaven har vært gjennom denne preprosesseringen.

Fjerning av dårlige bånd (støy)

Første steg etter preprosessering var å gjennomgå datasettet bånd for bånd for å identifisere mulige bånd uten informasjon eller bånd med mye støy. Bånd i bølgelengdeområdene rundt 1400 og 1900nm er særlig utsatt på grunn av vannabsorpsjon fra atmosfæren. Det var derfor hensiktsmessig å utelatte disse områdene for videre analyser.

Etter en visuell gjennomgang av båndene ble totalt 104 bånd fra SWIR utelatt. Bånd 73-93 og bånd 155-189 ble utelatt siden de befant seg i sonene for vannabsorpsjon og hadde mye støy. Bånd 192-196 samt bånd 254-288 ble også droppet på grunn av mye støy og lite verdifull informasjon. Av totalt 474 bånd ble 370 bånd brukt videre til analysen og klassifiseringen, 186 bånd fra VNIR og 184 bånd fra SWIR.



Figur 27: Spektralprofil før og etter fjerning av bånd. Brudd i kurven tilsvarer bånd som er fjernet.

Reskalering og sammenslåing av data

Ved en klassifisering av datasettet var det ønskelig å bruke både informasjon fra VNIR og SWIR. Disse datasettene hadde derimot ulik romlig oppløsning. VNIR hadde en romlig oppløsning på 0,3 m mens SWIR hadde en romlig oppløsning på 0,7 m. For å kunne se på hele spekteret fra VNIR til SWIR, måtte dataområdet ha den samme romlige oppløsningen. SWIR ble derfor reskalert til samme romlige oppløsning som VNIR (0,3m). Da det var utført ble VNIR slått sammen med SWIR. De åtte første båndene i SWIR overlappet med VNIR og ble derfor utelatt.

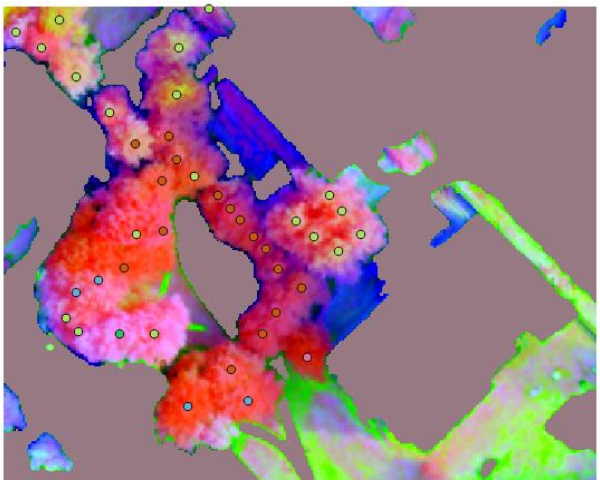
3.2.2 Identifisering og valg av trær

Samlet sett fantes det feltdata for trær fra ni små testområder samt et stort testområde. De registrerte trærne fra feltarbeidet ble identifisert på de hyperspektrale bildene ved å bruke notater fra feltarbeidet. Ukjente trær ble forsøkt identifisert i ettertid i samråd med eksperter på fagområdet.

For både det store og de små testområdene ble valg av trær basert på følgende hovedkriterier:

1. *Mulig å identifisere.* Trær som ikke var mulig å identifisere på bildene ble valgt bort.

2. *Enkelttrær.* Det ble forsøkt å begrense utvalget til enkelttrær for å minimere mulighetene for at informasjon fra andre trær/materialer ble med. Men nærliggende trær i samme klasse ble tatt med dersom ingen andre klasser var i nærheten.



Figur 28: Eksempel på område med klynger av trær i forskjellige klasser. Her er det kjørt en PCA for å forsøke å se forskjell på klassene. Båndkombinasjon: R=PC1, G=PC2, B=PC3.

På det store testområdet var utvelgelsen av trær i tillegg basert på et annet kriteria. En annen masterstudent skulle se på klassifisering av trær over det samme området ved å bruke laserdata fra Optech Titan (Lofthus, 2018). For å sammenligne resultater ble det forsøkt å bruke det samme datagrunnlaget fra bytrærne.

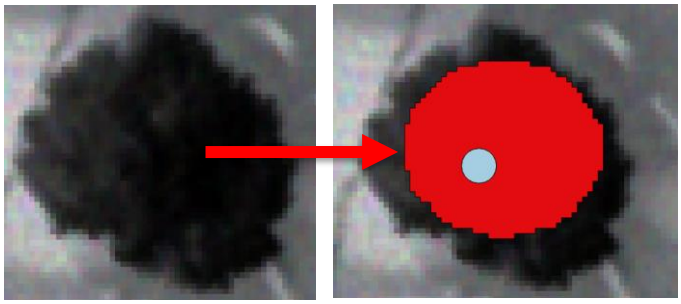
Noen trær hadde flere registreringer, men for klassifiseringen sin del var det tilstrekkelig med en id per tre. Derfor ble hvert tre representert ved et vektorpunkt. Punktene ble deretter omgjort til sirkler ved å definere en buffer på hvert punkt. Dette ble gjort for at identifiseringen av trærne skulle bli lettere, når spektralinformasjonen senere skulle hentes ut. I beste fall kunne sirklene brukes direkte når spektralinformasjonen skulle hentes ut.

Størrelsen på bufferen ble basert på målingene av trekronene fra feltarbeidet. Minste målte kronestørrelse var 1,4 x 1,6 m. En buffer på 1m radius ble derfor valgt.

3.2.3 Uthenting av spektralinformasjon fra trærne

For å hente ut spektralinformasjon fra trærne ble sirklene konvertert til ROI og inndelt etter art. ROI står for Region Of Interest og er kort fortalt en region definert av brukeren, et interesseområde. Ved å definere en ROI får man hentet ut alle pikselverdiene innenfor området, i tillegg til enkel statistikk.

Det var ønskelig å få med mest mulig av hver trekrone for å fange opp variasjonen. For å ta høyde for blanding med andre materialer ble likevel hver ROI definert godt innenfor hver trekrone. Ved en klassifisering var det viktig at grunnlaget var så bra som mulig, at spektralinformasjonen var så ren som mulig. Det betyr at så mye som mulig av spektralinformasjonen kom fra materialet som skulle klassifiseres og ikke andre materialer. Hvert tre ble nøye gjennomgått for å minimere risikoen for at elementer fra andre materialer ble med. Klynger av trær i forskjellige arter ble ikke tatt med i utvelgelsen, da det var en viss fare for at spektralinformasjonen ville bli blandet.



Figur 29:Prosessen for å hente ut spektralinformasjon. sirkel på 1m radius(lyseblått) ble brukt som utgangspunkt til å definere en ROI (rødt).

Spektralinformasjonen ble deretter eksportert til en CSV-fil og strukturert. Scriptet for strukturering av en CSV-fil ligger i vedlegg B. Et problem var at kun navnet på klassen og x- og y-koordinater for hver piksel ble lagret ved eksportering. Uten id på trærne ble det vanskelig å gruppere dataene etter hvert tre. For å løse dette problemet ble dataene koblet mot polygonlaget ved å bruke en romlig join. Dermed var det mulig å koble informasjonen til hvert tre. Pikslene ble strukturert med tilhørende X- og Y- koordinater, Klasse ID, art, treslag og båndnummer slik at det ville være lett å lokalisere tilhørigheten til pikslene.

3.2.4 Vurdering av spektral separasjon

Før klassifiseringen ble spektralsignaturene til hver klasse sammenlignet og vurdert opp mot hverandre for å se på likheter og forskjeller. Ved å studere spektralsignaturene får man dannet seg et bilde av hvor separable klassene er. Like spektralsignaturer for to klasser kan tyde på at klassene kan bli vanskelig å skille. I tillegg ble separasjonen til hver ROI vurdert ved å bruke Jeffries-Matusita-avstanden og transformert divergensmatrise. En kort gjennomgang av disse følger

Divergens

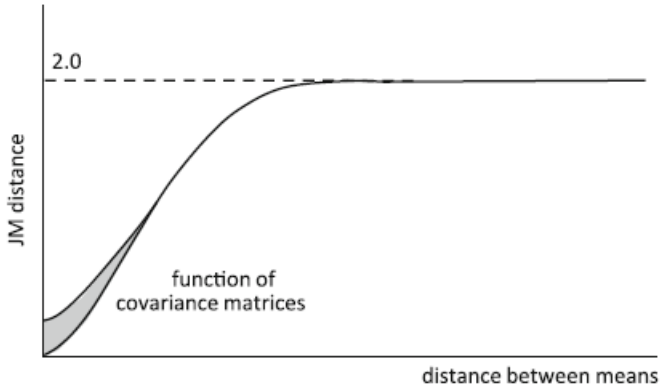
Divergens er et mål på spredning og kan defineres som graden av overlapp mellom et par sannsynlighetsfordelinger (Richards, 2013).

Jeffries-Matusita avstanden og transformert divergens

Jeffries-Matusita er et mål på gjennomsnittlig avstand mellom to fordelinger. Formelen for Jeffries-Matusita-avstanden er definert som (Richards, 2013):

$$J_{ij} = 2(1 - e^{-B_{ij}}) \tag{3.2}$$

der B er Battacharaya-avstanden. Verdiene for Jeffries-Matusita går fra 0 til 2. For avstander på 2 er Jeffries-Matusita asymptotisk. En verdi på 2 for to spektralklasser betyr derfor at en piksel kan plasseres i en av klassene med 100% nøyaktighet (Richards, 2013).



Figur 30: Jeffries-Matusita-avstanden (Richards, 2013).

Transformert divergens måler graden av separasjon mellom to spektralklasser. Verdiene for transformert divergens går fra 0 til 2, der 2 indikerer komplett separasjon mellom klassene og 0 indikerer komplett overlapp. For separasjon er følgende verdier foreslått (Landmap, u.å.):

Tabell 5:Inndeling av separasjon.

Intervall	Grad av separasjon
0 - 1	Dårlig
1 - 1,9	Moderat
1,9 - 2	God

3.2.5 Veivalg klassifisering

En rekke hensyn og valg måtte tas før en klassifisering. En del er allerede nevnt. For det første ble det valgt å bruke det normaliserte datasettet. Ved at det er normalisert er noe av skyggeproblematikken tatt hånd om

Det neste valget som måtte tas var om datamengden skulle reduseres med for eksempel PCA. Siden det var ønskelig å se på hvilke resultater som var mulig å oppnå med hvert datasett i sin helhet, ble ikke PCA brukt til klassifiseringen.

Informasjonen fra trekronene kan ha store variasjoner på grunn av varierende lysforhold eller blanding med bakgrunnen der det er sparsomt bladverk. En måte å ta høyde for variasjoner og redusere datamengden er å bruke middelspekteret for hvert tre (Launeau et al., 2017). Ved å benytte middelspekteret fra hvert tre blir påvirkningen fra andre materialer minimert, i de tilfellene spektralinformasjonen er blandet. Derfor ble middelverdi for hvert tre også brukt som utgangspunkt i denne oppgaven. Pikselverdiene ble da midlet på hvert tre. Scriptet for middelverdi finnes i vedlegg C.

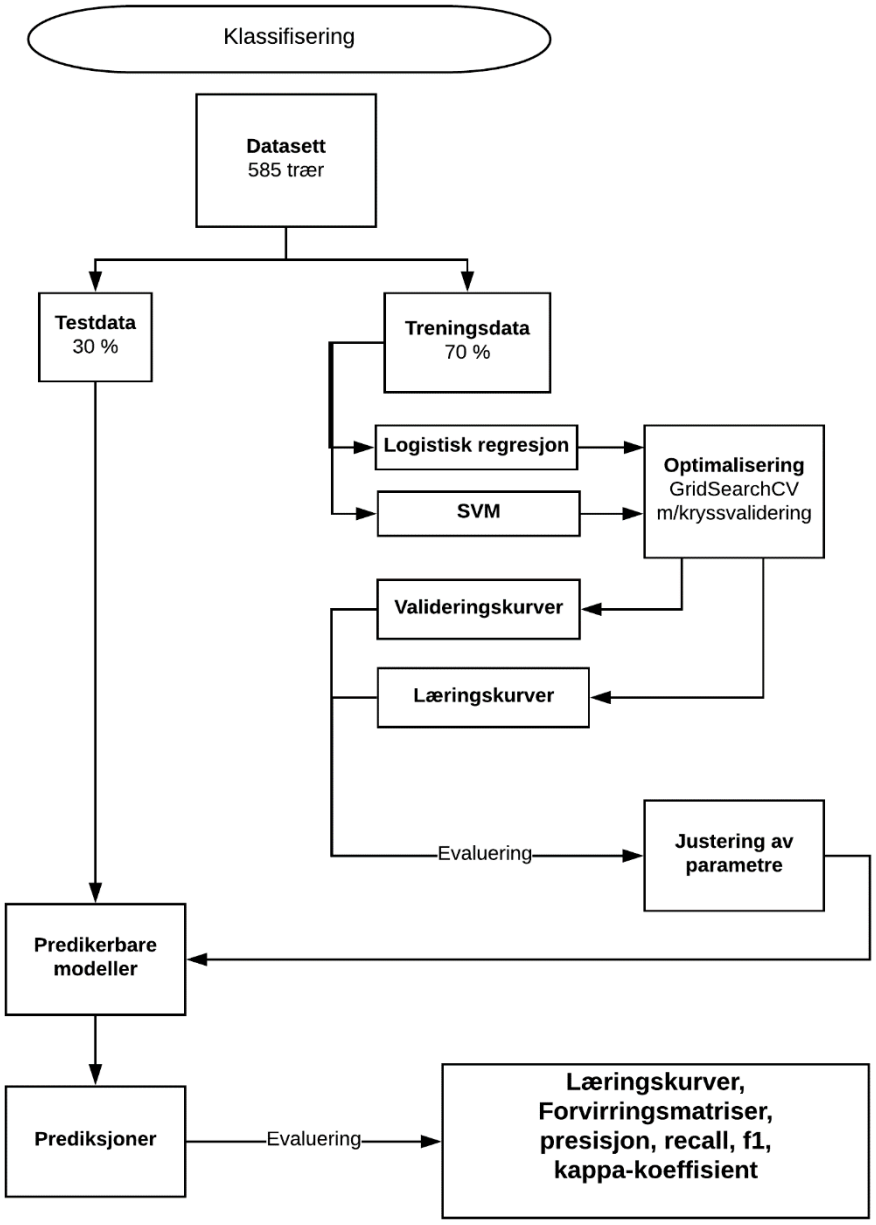
3.2.6 Beskrivelse av klassifiseringen

Den eksplorative analysen utforsket de hyperspektrale dataene på et lite testområde fra feltarbeidet. Der var målet bare å bli kjent med datasettene og mulighetene med dem. Metoden for klassifiseringen bygget videre på erfaringene i den eksplorative analysen. Selve klassifiseringen ble utført på det store testområdet. Styrt-klassifisering ble foretrukket siden det fantes tilgang på treningsdata fra bytrærne. For å virkelig se på verdien i de hyperspektrale dataene ble trærne inndelt etter art når klassifiseringen skulle utføres.

Hvert tre var nummerert slik at det var mulig å benytte ulike trær som trenings – og valideringsdata. Ulike modeller og innstillinger ble testet ut for å finne de optimale parameterne. Klassifiseringen ble evaluert ved hjelp av læringskurver, forvirringsmatriser, presisjon, recall, f1 og kappa-koeffisienten.

Bare klasser med 25 trær eller mer ble valgt ut for å ha nok datagrunnlag. Datasettet ble da redusert fra 805 til 585 trær. Datasettet ble deretter splittet til treningssett/ valideringssett i forholdet 70/30 %, med likt antall klasser i hver del. Det betyr at treningssettet utgjorde 409 trær og valideringssettet 176 trær.

Klassifiseringen ble utført på datasettet VNIR, deretter VNIR+SWIR. Totalt ble klassifiseringen kjørt fem ganger på hver modell, for å ta høyde for variasjoner i utvalget. Ved optimaliseringen av parameterne ble dataene kryssvalidert med en inndeling på 10 blokker. Dermed ble et større datagrunnlag brukt til trening av modellen hver gang. En oversikt over fremgangsmåten for klassifiseringen følger:



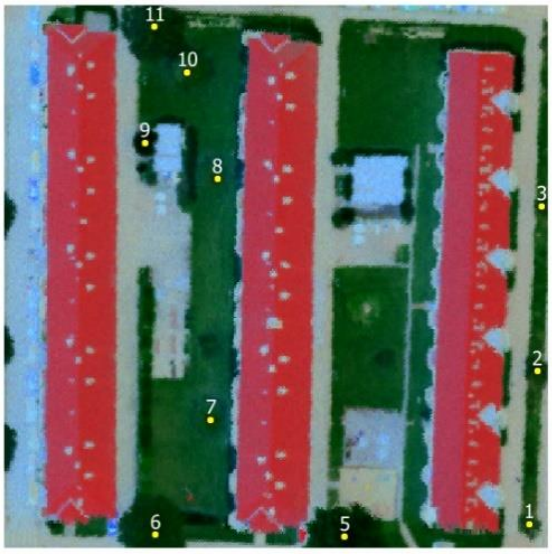
Figur 31: Flytdiagram for klassifiseringen.

4 Resultater og diskusjon

I denne delen blir resultatene fra analysene presentert og diskutert. Kapitlet er inndelt i to deler. Første del omhandler resultater fra den eksplorative analysen. Den andre delen av kapitlet tar for seg resultatene fra det store testområdet. Selve klassifiseringen ble utført på dette området.

4.1 Resultater eksplorativ analyse

Denne delen tar for seg resultatene fra den eksplorative analysen. Først presenteres resultatene fra hver analyse, deretter hovedfunnene. Den eksplorative analysen ble utført på testområde 1. På dette området var det lite skygger og kun enkelttrær slik at det var et ideelt område å utforske.



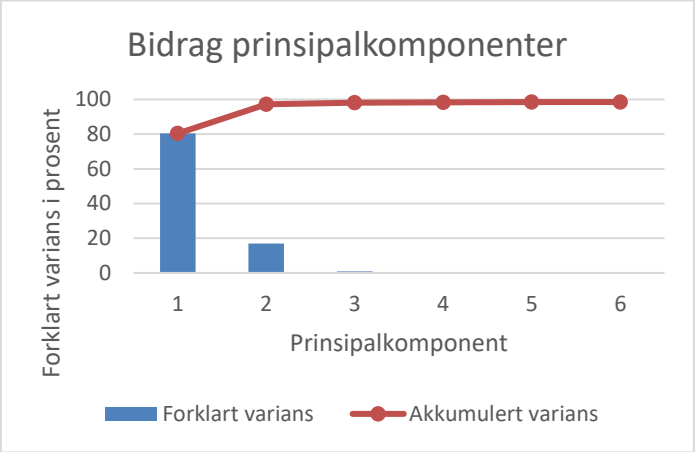
Figur 32: Oversiktsbilde av testområde 1 med nummererte trær i henhold til tabell 6.

Tabell 6: oversikt over klasser på testområde 1.

Id	Art	Familie	Treslag
1	Hengebjørk	Bjørkefamilien	Løvtre
2	Hengebjørk	Bjørkefamilien	Løvtre
3	Hengebjørk	Bjørkefamilien	Løvtre
5	Lind	Kattostfamilien	
6	Bjørk	Bjørkefamilien	Løvtre
7	Edelgran	Furufamilien	Bartre
8	Edelgran	Furufamilien	Bartre
9	Syrin	Oliventrefamilien	
10	Edelgran	Furufamilien	Bartre
11	Hengebjørk	Bjørkefamilien	Løvtre

4.1.1 PCA

PCA ble testet på VNIR for all informasjonen på testområde 1. Figur 33 viser forklart varians per prinsipalkomponent for dette området. Jo høyere verdi jo større bidrag til variasjonen i datasettet. Fra figur 33 kan man se at bidragene avtar betraktelig etter to komponenter. Av tabell 7 kan man se at de to første prinsipalkomponentene forklarer hele 97,26 % av variasjonen i datasettet. Den tredje komponenten bidrar kun med 0,9 % til variasjonen. Vanligvis ville man tatt med kun de to første komponentene på bakgrunn av akkumulert varians. Her har jeg derimot valgt å se nærmere på de fem første komponentene siden score-bildene for disse viste at de også inneholdt noe informasjon.



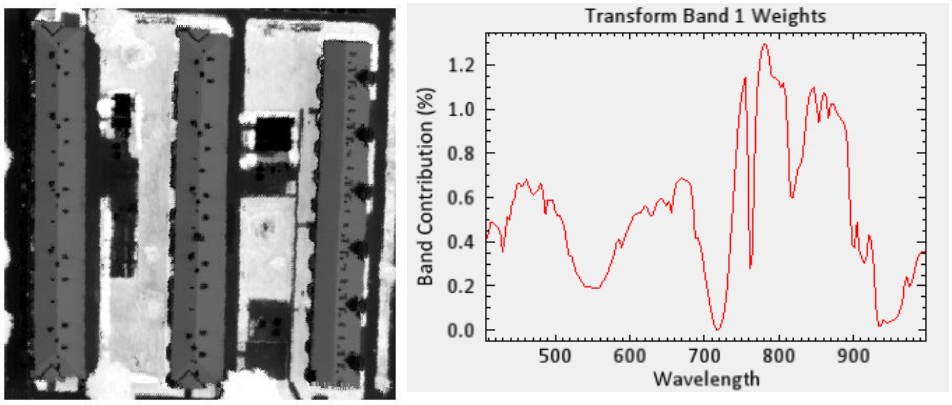
Figur 33: Forklart varians per prinsipalkomponent sammen med akkumulert varians. Detaljer for hver komponent er gitt i tabell 7.

.

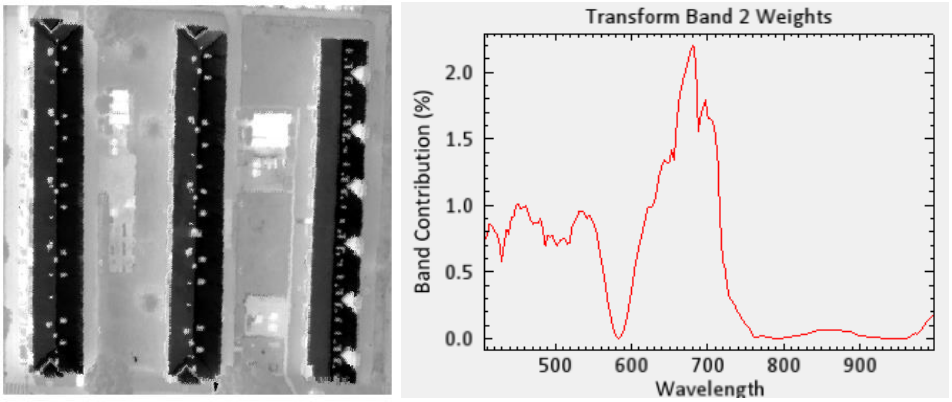
Tabell 7:Detaljer for hver prinsipalkomponent.

PC	Forklart Varians (%)	Akkumulert varians (%)
1	80,36	80,36
2	16,9	97,26
3	0,9	98,16
4	0,17	98,33
5	0,14	98,47
6	0,06	98,53

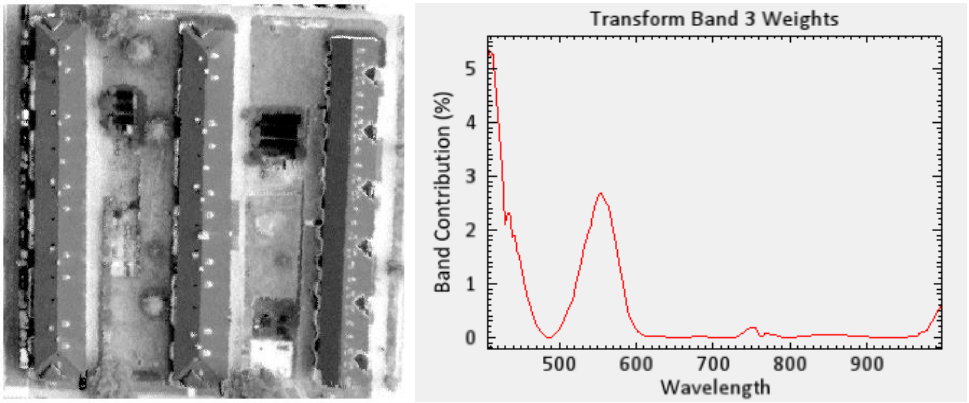
Hver komponent er her vist som et score-bilde, der hvit illustrerer størst bidrag og sort minst bidrag. Størst variasjon vil det da være mellom hvite og sorte områder. Grafen ved siden av hvert bilde er et ladningsplott som viser prosentvist bidrag fra hver bølgelengde. Enheten på bølgelengdene er nm.



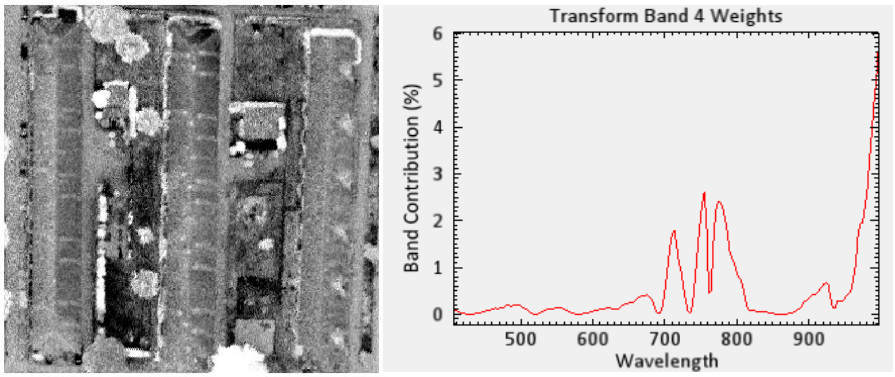
Figur 34: score-bilde av PC1 med tilhørende ladningsplott. Størst variasjon er det mellom vegetasjon og veier. Dette gjenspeiles i grafen til høyre som viser bidraget fra hver bølgelengde. Det største bidraget kommer fra bølgelengder fra ca.700 nm og oppover, i det nærinfrarøde området. Det er nettopp i dette området at vegetasjon reflekterer mest stråling.



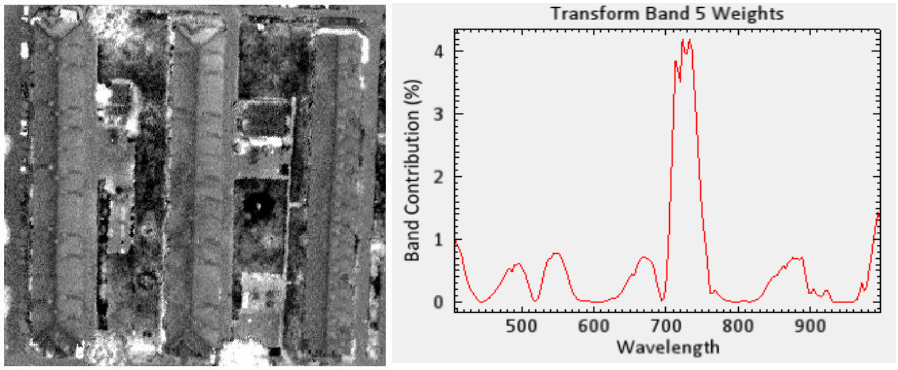
Figur 35: Score-bilde av PC2 med tilhørende ladningsplott. PC2 har størst bidrag fra bølgelengder mellom ca. 600 nm til ca.750 nm. Her er den største variasjonen mellom ulike tak, fra taket på blokkene (sort) til taket på bodene (hvitt).



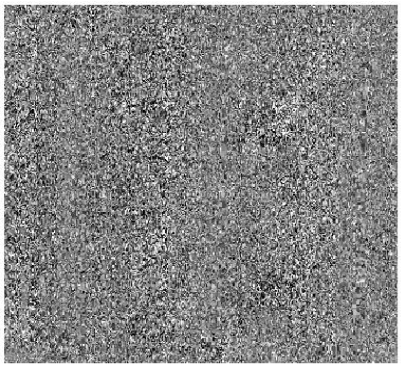
Figur 36: score-bilde av PC3 med tilhørende ladningsplott. Mesteparten av bidraget til PC3 kommer fra bølgelengder under 600 nm. På score-bildet er det størst forskjell mellom tak på boder (sort) og asfalt (hvitt). Selv om PC3 forklarer bare 0,9% av variasjonen i datasettet er det fortsatt en del detaljer igjen.



Figur 37: score-bilde av PC4 med tilhørende ladningsplott. Den fjerde prinsipalkomponenten fanger opp en enda mindre del av variasjonen i datasettet. Nå begynner det å bli merkbart mye støy i bildet. Her er det størst variasjon mellom busker og trær (hvitt), og gress (sort). Det er interessant å observere at enkelte trær skiller seg ut fra de andre, spesielt treet nede til høyre som er Lind.

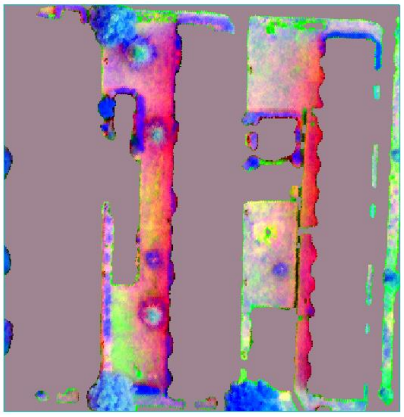


Figur 38: Score-bilde av PC5 med tilhørende ladningsplott. På PC5 er det størst bidrag fra området mellom 700nm og til ca. 750nm. På bildet tilsvarer det lyse områder og er representert ved løvtre og busker.



Figur 39: PC6 inneholder bare støy og ikke noe brukelig informasjon.

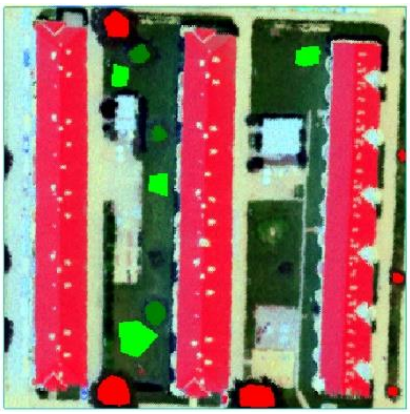
Prinsipalkomponentene kan settes sammen til et fargebilde der hver komponent har en egen farge. Et slikt bilde er fint for å illustrere hvilke komponenter som bidrar hvor. Ved å bruke de tre første prinsipalkomponentene for vegetasjon får man et bilde lik figur 40. Områder som ikke er vegetasjon er maskert bort og vises som grått.



Figur 40: Fargebilde basert på de tre første prinsipalkomponentene for vegetasjon. Her er PC1 representert ved rød farge, PC2 ved grønn farge og PC3 ved blå farge. Største bidrag til trær og busker kommer fra PC3 (blått).

4.1.2 Evaluering av spektralsignaturer

Spektralsignaturene for utvalgte klasser ble deretter evaluert for å se på likheter og forskjeller, og se om noen arter skilte seg ut. Aller først ble klassen løvtre evaluert mot klassen bartre. For sammenligning var gress også tatt med.



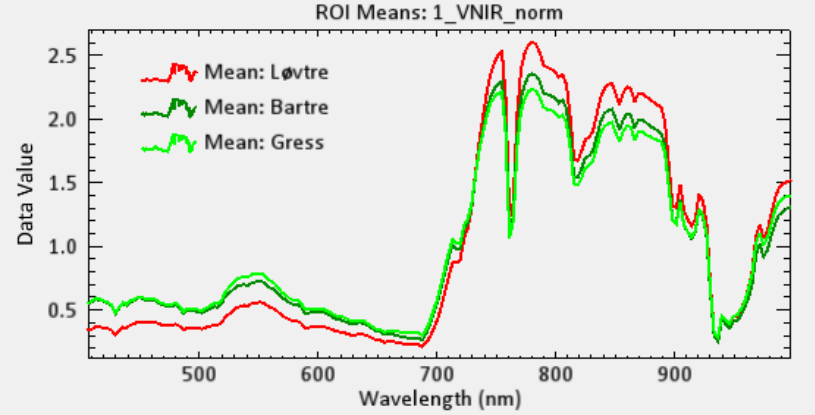
ROI	Antall piksler
Bartre	1536
Gress	571
Løvtre	1453

Figur 41: ROI på løvtre, bartre og gress.

Spektralinformasjonen ble hentet ut fra hver ROI og plottet, både for VNIR og SWIR.

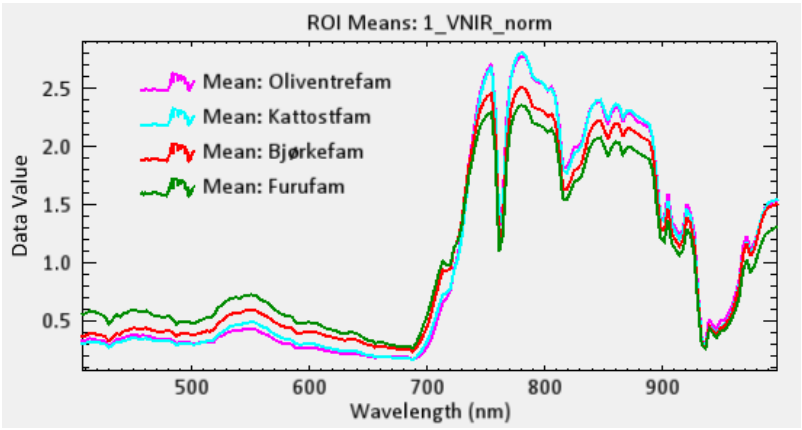
VNIR

Ved å plotte spektralinformasjonen får man spektralsignaturer som vist i figur 42.



Figur 42: Spektralsignaturer for løvtre, bartre og gress.

Figur 42 viser spektralsignaturene for gress, løvtre og bartre. Spektralsignaturene er basert på middelverdien for hver klasse. Typiske trekk ved spektralsignaturen til vegetasjon er lett synlig, med størst verdier i det nærinfrarøde området (over 700 nm) og lokalt maksimum i bølgelengdene for grønne farger.



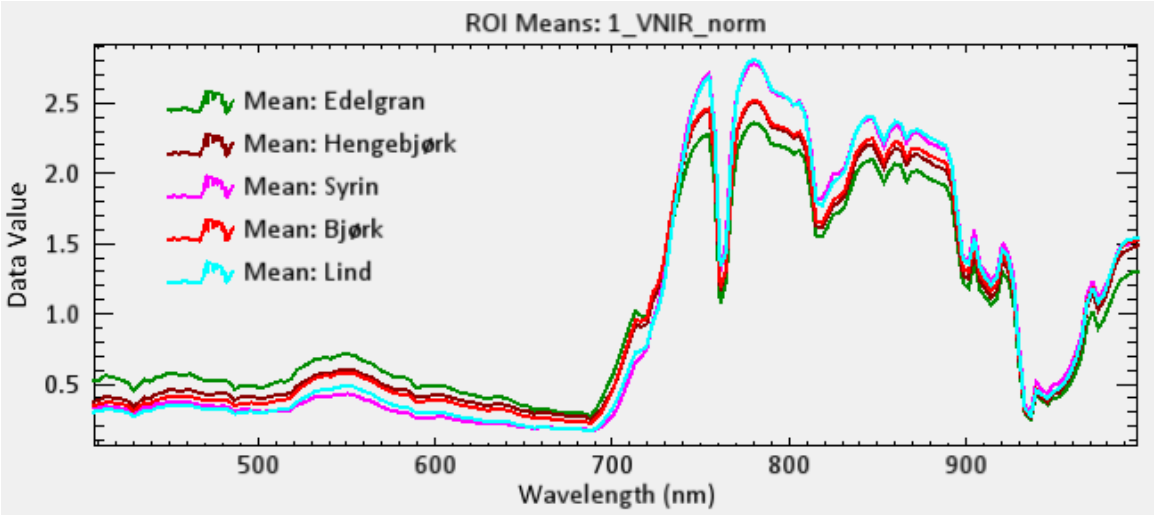
Figur 43: Plott av spektralsignatur for trær inndelt etter familie.

Tabell 8: Inndeling etter art, familie og treslag med tilhørende statistikk for hver klasse.

Art	Familie
Bjørk	Bjørkefam.
Hengebjørk	Bjørkefam.
Syrin	Oliventrefam.
Lind	Kattostfam.
Edelgran	Furufam.

Trærne ble deretter gruppert i familier. Med inndeling etter familie ble det straks tydelig at noen klasser har veldig like spektralsignaturer. Kattost- og Oliventrefamilien har så å si identiske signaturer, kun litt forskjell rundt 550nm. Oliventre- og Kattostfamilien skiller seg også ut fra bjørkefamilien og furufamilien.

I neste steg ble familiene delt opp etter art, bjørk ble for eksempel skilt fra hengebjørk.

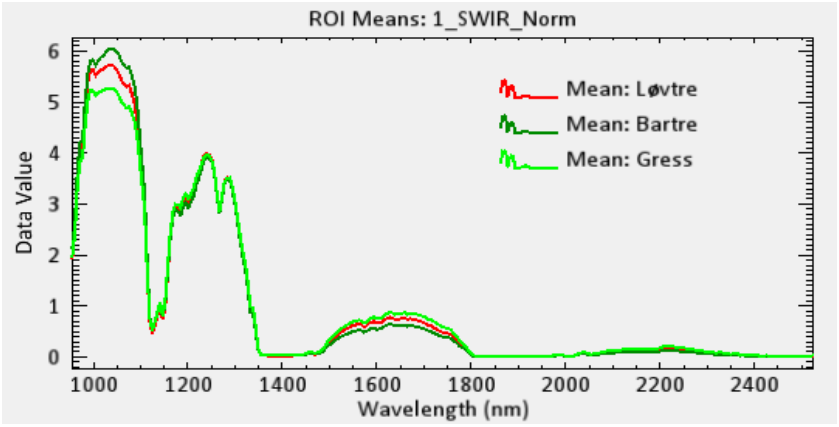


Figur 44: spektralsignaturer i VNIR for arter.

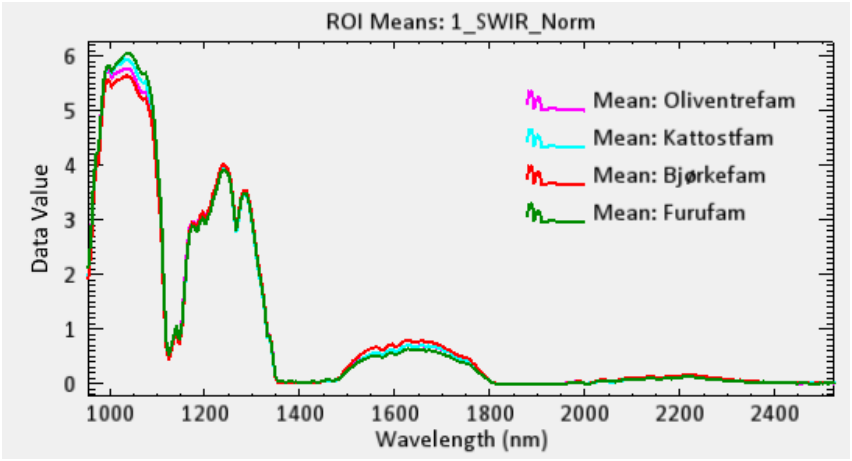
Lind og Syrin har veldig like spektralsignaturer. Det kan tyde på at disse artene kan være vanskelig å skille. Bjørk og hengebjørk har også veldig like spektralsignaturer. Det tyder på at det kan bli vanskelig å skille disse. Lind og bjørk/hengebjørk har ulike spektralsignaturer, noe som tyder på at disse lar seg skille. Edelgran skiller seg mest ut fra de andre klassene. Størst forskjell er det mellom lind/syrin og edelgran.

SWIR

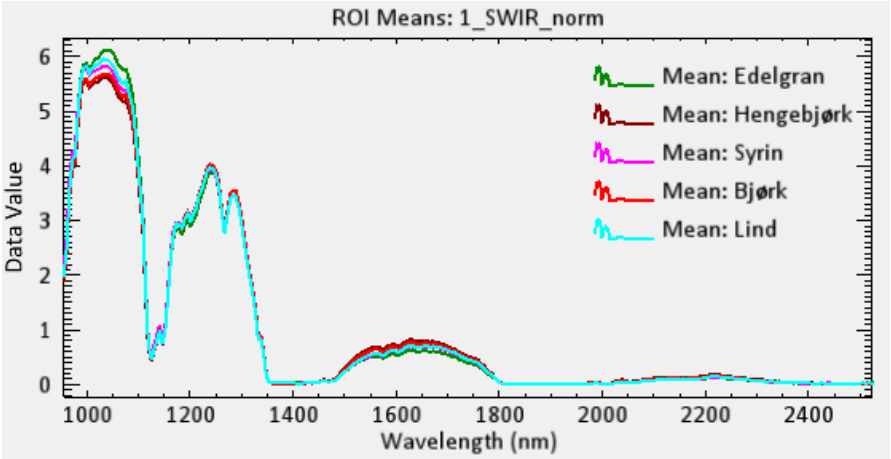
Samme inndeling ble benyttet for SWIR. Først treslag, deretter familier og til slutt arter.



Figur 45: Spektralsignaturer i SWIR for løvtre, bartre og gress.



Figur 46: Spektralsignaturer i SWIR for familier av trær.



Figur 47: spektralsignaturer i SWIR for arter. SWIR har større verdier enn VNIR. Størst forskjell er det fra 1000 nm til ca.1150 nm. Sonene for vannabsorpsjon er tydelig synlige, i form av områder med lave verdier (rundt 1400 nm og 1900 nm)

4.1.3 Vurdering av spektral separasjon

Etter en visuell inspeksjon av spektralsignaturene, ble separasjonen til klassene beregnet ved hjelp av Jeffries-Matusita avstanden. Denne er beskrevet nærmere i kapittel 3.2.4, s.41. Beregning av separasjon baserer seg på informasjonen fra hver ROI/klasse.

For å beregne separasjonen må antall piksler i hver klasse være flere enn antall bånd i bildet. Syrin var derimot bare representert ved 109 piksler, mens bildet hadde 186 bånd i VNIR. Det var to måter å løse dette på, enten ved å legge til flere piksler i klassen, eller redusere antall bånd.

Siden det ikke var flere piksler å legge til, ble separasjonen beregnet på bakgrunn av PCA fra VNIR. De fem første komponentene inneholdt informasjon og ble derfor benyttet. Ideelt sett skulle separasjonen vært beregnet på all informasjonen fra bildet, men når det var så få piksler gikk ikke det. PCA ville likevel gi en indikasjon på separasjonen for trærne.

Tabell 9: Separasjon, fra minst til størst.

Par-separasjon (minst til størst)	
Hengebjørk og Bjørk	1,74
Syrin og Lind	1,86
Edelgran og Hengebjørk	1,88
Edelgran og Bjørk	1,91
Hengebjørk og Lind	1,98
Hengebjørk og Syrin	2,00
Bjørk og Lind	2,00
Edelgran og Lind	2,00
Edelgran og Syrin	2,00
Syrin og Bjørk	2,00

Resultatet indikerer at bjørk og hengebjørk er minst separable, fulgt av syrin og lind. Dette stemmer overens med det spektralsignaturene viste. Verdier over 1,9 indikerer at separasjonen er god. En verdi på 1,74 for bjørk og hengebjørk indikerer moderat separasjon. Ingen klasser har dårlig separasjon (verdier under 1). Syrin og bjørk har størst separasjon.

4.1.4 Test av ulike algoritmer for styrt klassifisering

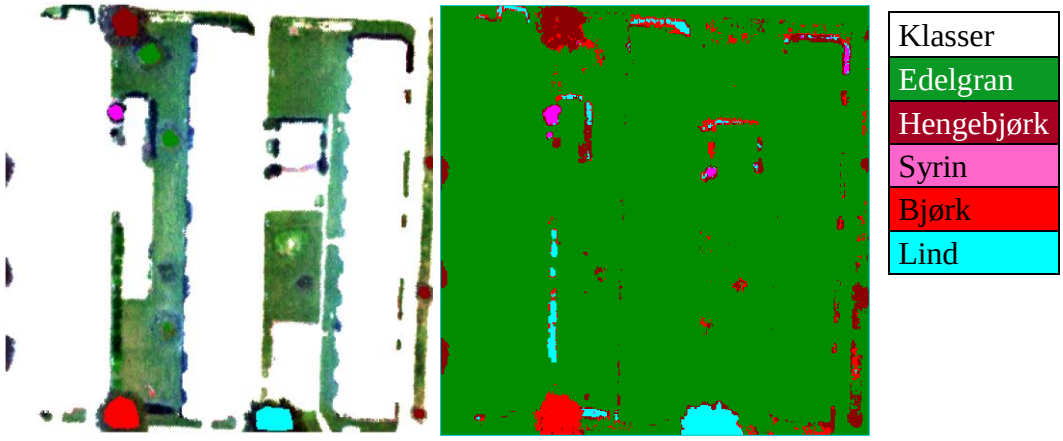
Ulike algoritmer og innstillinger ble utforsket for å finne ut hvilke som kunne egne seg til videre analyser på det store testområdet. Vurderingen her er basert på en visuell

sammenligning av resultatet. Testen er utført i ENVI. ENVI har flere innebygde algoritmer for styrt-klassifisering. De som ble testet ut her er SVM, SAM og Maximum likelihood. Algoritmene ble testet på et vegetasjonsbilde for å begrense antall klasser. Alt som ikke er vegetasjon er maskert bort og vises som hvite områder.

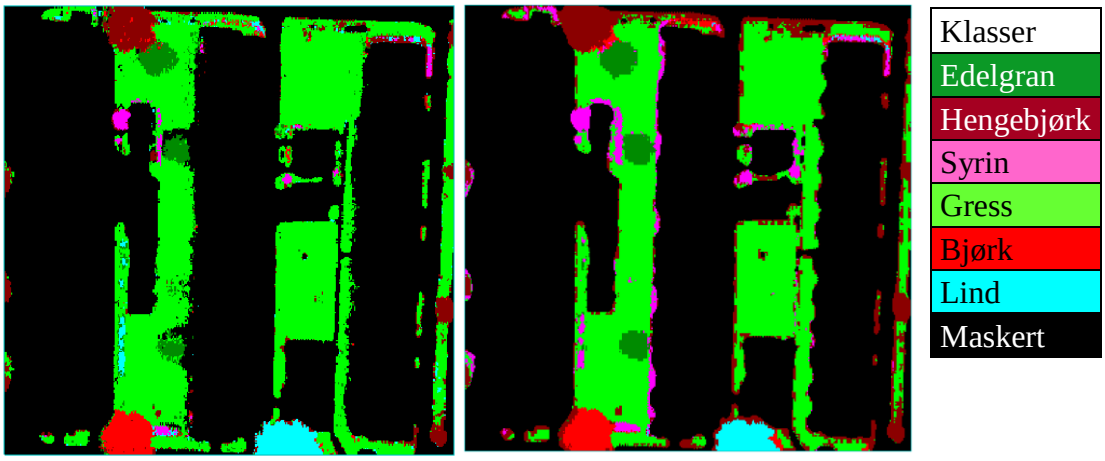
SVM (Support Vector Machine)

Tabell 10: Innstillinger for SVM.

SVM	
Penalty parameter	100
Kernel	RBF
Gamma	0,005



Figur 48: Resultat av SVM på vegetasjonsbildet.

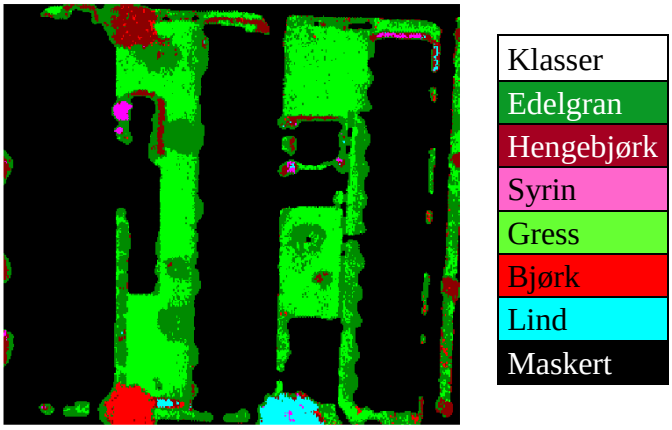


Figur 49: Visuell sammenligning av SVM på VNIR (venstre) og på VNIR+SWIR (høyre). Spesielt gress, edelgran og syrin er mer tydelig definert i bildet til høyre. Samtidig har hengebjørk og gress blitt forvekslet enkelte steder.

Figur 48 viser resultatet av SMV med treningsdata fra kun trær. Klasser for gress og maskerte områder (ikke-vegetasjon) var ikke tatt med for å se hvordan resultatet ble. Disse ble da klassifisert som Edelgran. For videre analyser ble det lagt inn klasser for gress og maskerte områder. Med definerte områder for gress og maskerte områder har SVM gjort en bedre jobb med klassifiseringen (se figur 49).

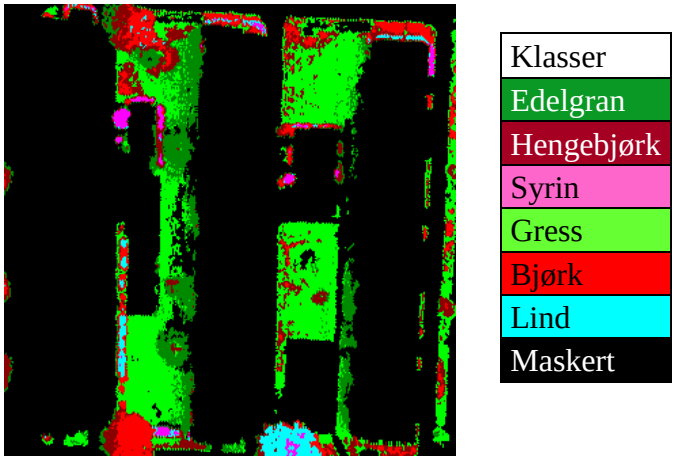
Maximum likelihood

Denne algoritmen fungerer bare dersom minste ROI er på størrelse med antall bånd + 1. Minste ROI hadde bare 109 piksler mens datasettet VNIR hadde 186 bånd. Antall bånd måtte derfor reduseres. Igjen ble PCA løsningen. De fem første prinsipalkomponentene inneholdt informasjon. Ved å bruke PCA ble antall bånd redusert fra 186 til 5. Dermed kunne Maximum likelihood testes ut.



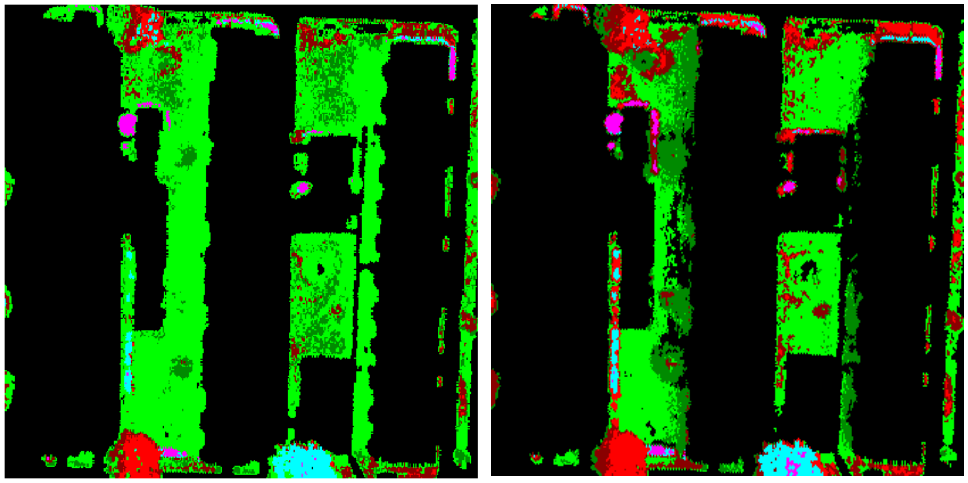
Figur 50: Maximum likelihood basert på de fem første prinsipalkomponentene.

SAM (Spectral Angle mapper)



Figur 51: Resultat med en lik spektralvinkel på 0,1 radianer for alle klassene, basert på VNIR.

Maximum likelihood har predikert klassene bra selv om datagrunnlaget kun var fem prinsipalkomponenter. Men algoritmen har blandet edelgran og gress. En visuell sammenligning viser at SAM har predikert klassene dårligere enn maximum likelihood og SVM. Figur 52 viser resultatet fra klassifiseringen med SAM. SAM har slitt med å skille hengebjørk fra bjørk. Deler av gresset har blitt predikert som edelgran. Resultatet tyder på at en lik spektralvinkel for klassene ikke er det beste alternativet. Ulike spektralvinkler ble derfor testet ut.



Figur 52: Test av ulike spektralvinkler for SAM. Bildet til høyre viser resultatet med en lik vinkel på 0,1 radianer for klassene. Bildet til venstre viser resultatet med ulike vinklet på klassene. Verdiene for vinklene er gitt i tabell 11

Tabell 11: Beste spektralvinkler for klassene.

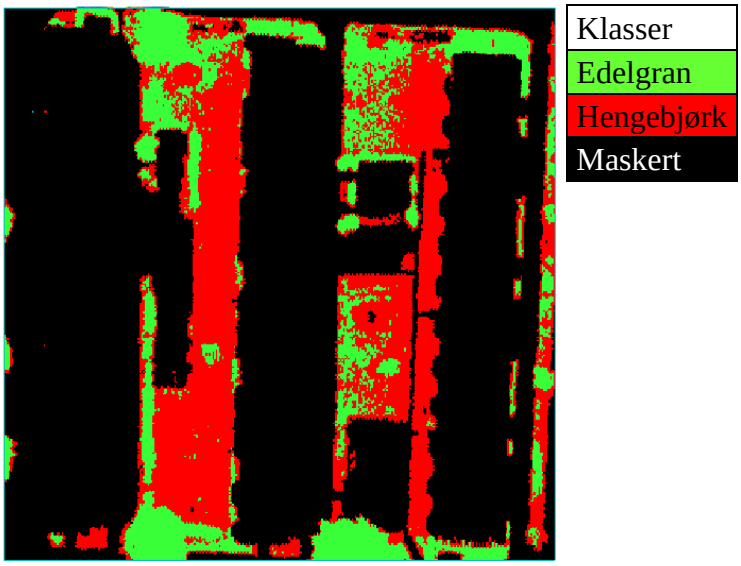
klasse	Vinkel (radianer)
Bjørk	0,04
Syrin	0,05
Hengebjørk	0,04
Lind	0,06
Gress	0,1
Edelgran	0,06

Med ulike vinkler for hver klasse ga SAM et resultat som visuelt sett stemte mye bedre de andre algoritmene. Gress og edelgran ble fortsatt forvekslet.

4.1.5 Test av forskjellige størrelser på ROI-ene

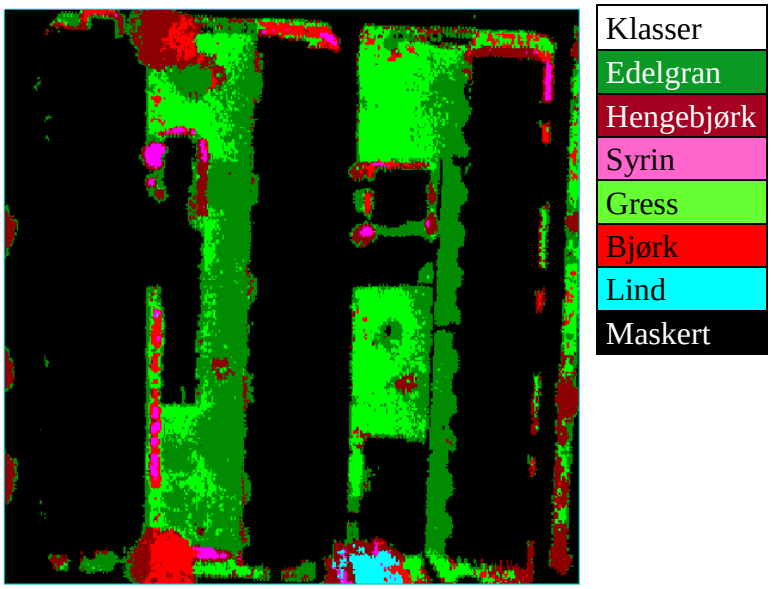
For å se hvordan treningsgrunnlaget påvirket resultatet ble førskjellige størrelser på ROI-ene testet ut. Blant de ulike algoritmene ga SVM best resultat. Denne algoritmen ble derfor brukt som utgangspunkt, med samme innstillinger som tidligere. Testen ble utført på VNIR-datasettet og med lik klasseinndeling som tidligere.

ROI basert på en piksel



Figur 53: Resultat basert på en piksel i hver klasse. SVM har bare klart å predikere edelgran og hengebjørk.

ROI basert på sirkel med 1m radius



Figur 54: Resultat med ROI på 1m radius for hver klasse.

4.1.6 Hovedfunn eksplorativ analyse

Hensikten med den eksplorative analysen var å gjøre seg kjent med de hyperspektrale dataene og teste ulike analysemetoder, for å se hvilke som kan egne seg til videre bruk på det store testområdet. Her presenteres hovedfunnene:

- ROI på en piksel er ikke tilstrekkelig for en god klassifisering.
- De fem første komponentene til PCA inneholdt informasjon.
- Prinsipalkomponentene inneholdt også informasjon selv om forklart varians var liten.
- PCA var en fin måte å visualisere variasjonen i datasettet på.
- Stor forskjell mellom spektralsignaturene til løvtre og bartre.
- Lind og syrin har like spektralsignaturer.
- Bjørk og hengebjørk har like spektralsignaturer og scoret moderat på separasjon.
- SWIR: størst variasjon mellom artene i området 1000 -1150 nm.
- Generelt størst forskjell mellom artene i det nærinfrarøde området.
- Beregning av spektral separasjon ga bra resultat med PCA som utgangspunkt.
- SVM ga best visuelt resultat sammenlignet med de andre algoritmene.
- Resultatene tyder på at SWIR gir bedre prediksjon av klassene.

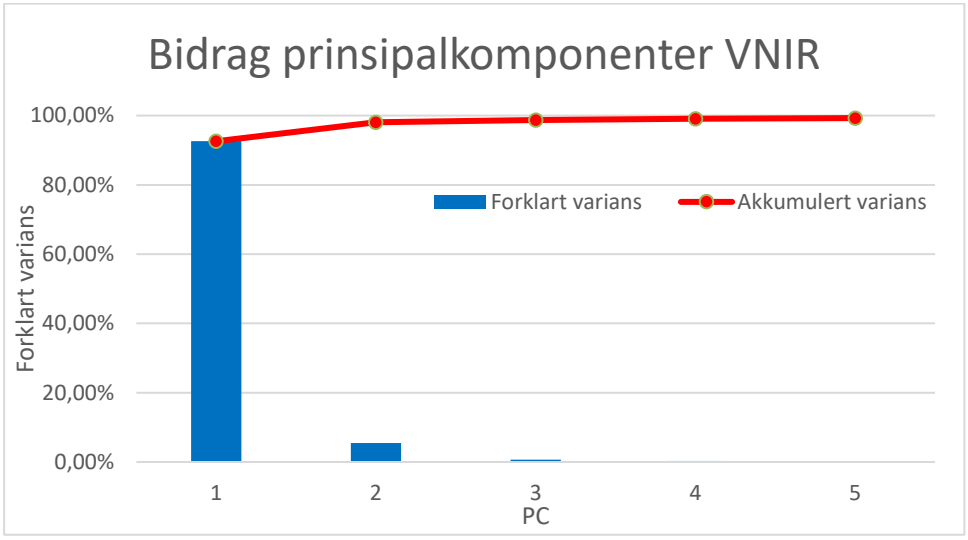
PCA, vurdering av separasjon, spektralsignaturer og SVM blir med til hovedanalysen.

4.2 Resultater stort testområde

Her presenteres resultatene fra analysene på det store testområdet. Erfaringene fra den eksplorative analysen ble tatt med i analysene. Resultatene leder fram til besvaring av problemstillingene. Først ble spektralinformasjonen analysert. Deretter ble trærne klassifisert.

4.2.1 PCA

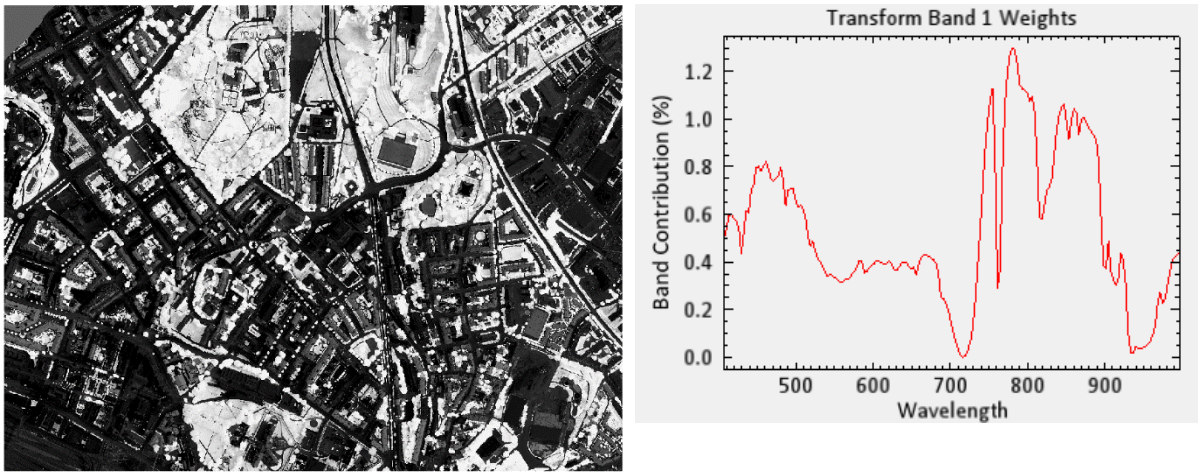
PCA ble kjørt både på både VNIR og SWIR for å få et inntrykk av variasjonen i datasettene og hvilke bølgelengdeområder som bidrar hvor. Den eksplorative analysen viste at kun de fem førte komponentene inneholdt informasjon. Derfor ble grensen satt ved fem komponenter på det store testområdet også.



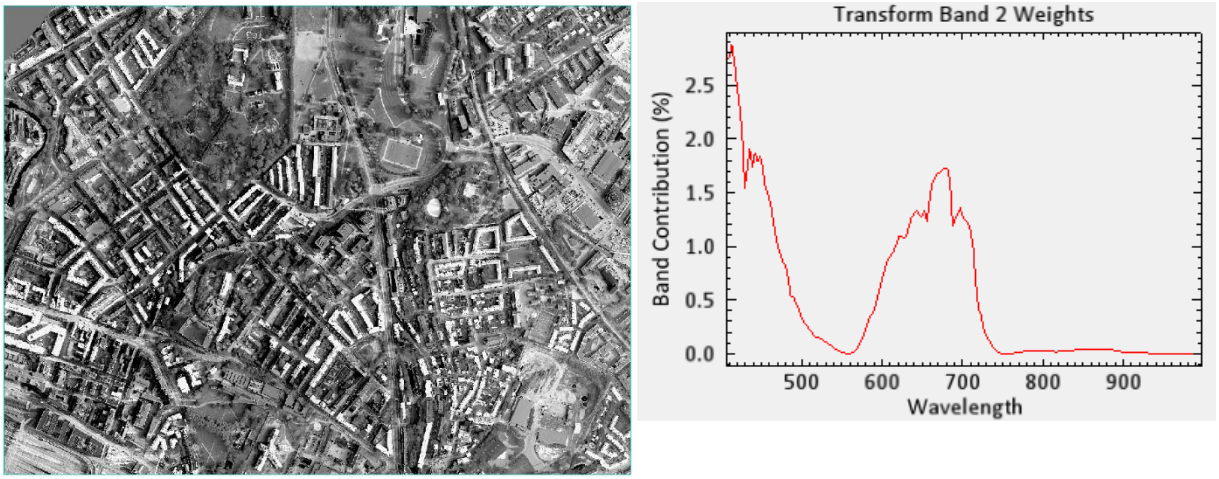
Figur 55: Forklart varians per prinsipalkomponent. Detaljer for hver komponent er gitt i tabell 12.

Tabell 12: Detaljer for hver prinsipalkomponent.

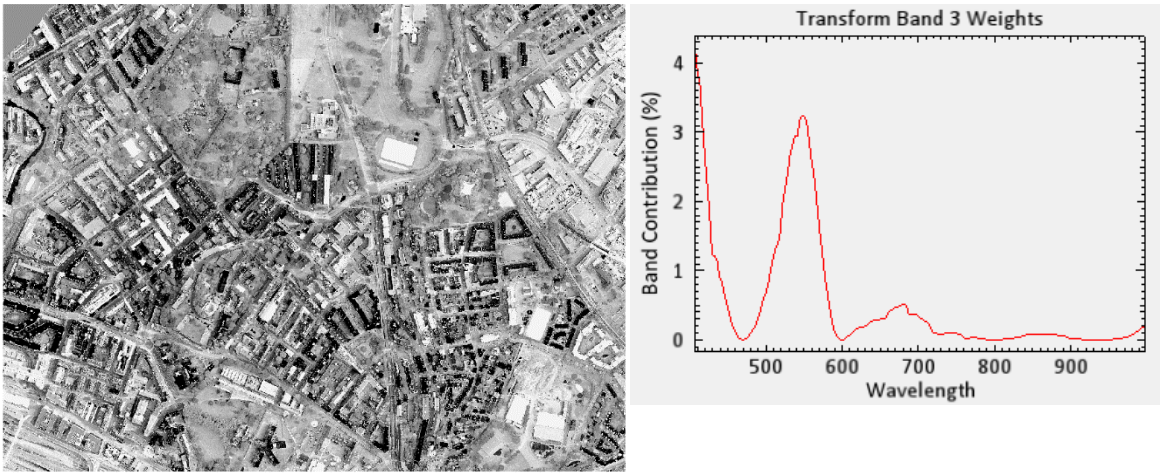
PC	Forklart varians	Akkumulert varians
1	92,59 %	92,59 %
2	5,49 %	98,08 %
3	0,67 %	98,75 %
4	0,31 %	99,06 %
5	0,21 %	99,27 %



Figur 56: Score-bilde for PC1 med tilhørende ladningsplott. PC1 forklarer hele 92,59% av variasjonen i datasettet. Det største bidraget til denne komponenten kommer fra det nærinfrarøde området (700 nm til 900 nm). Størst variasjon er det mellom vegetasjon og områder uten vegetasjon.

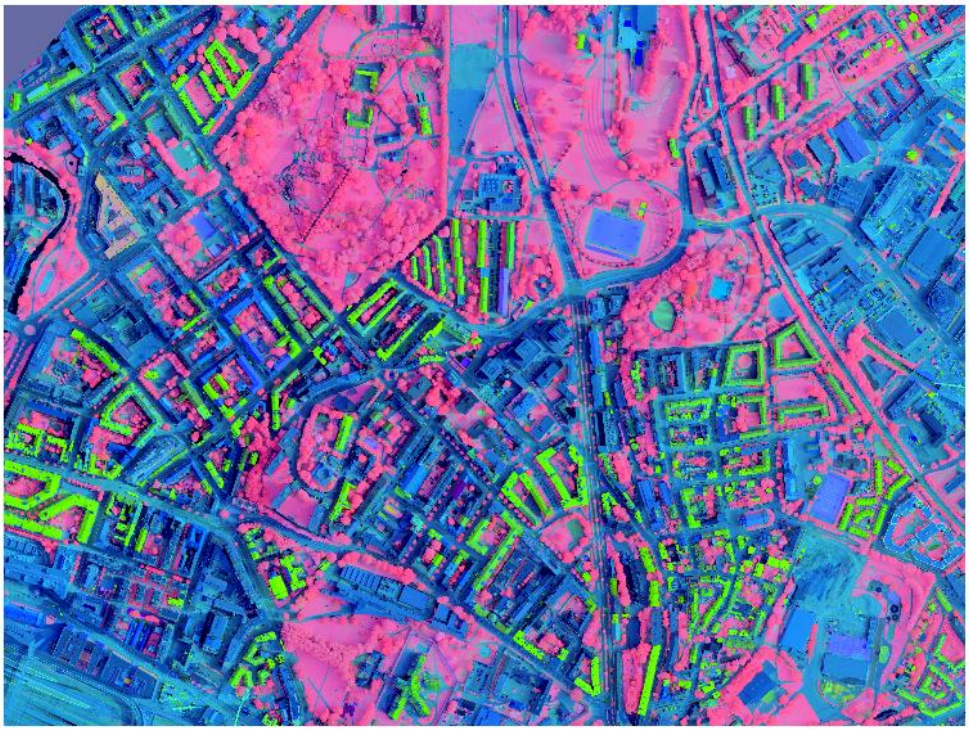


Figur 57: Scores-bilde for PC2 med tilhørende ladningsplott. Det største bidraget fra PC2 kommer fra tak og bygninger (hvit). Her er det størst bidrag fra bølgelengdene 600 til 750 nm, og under 500 nm.

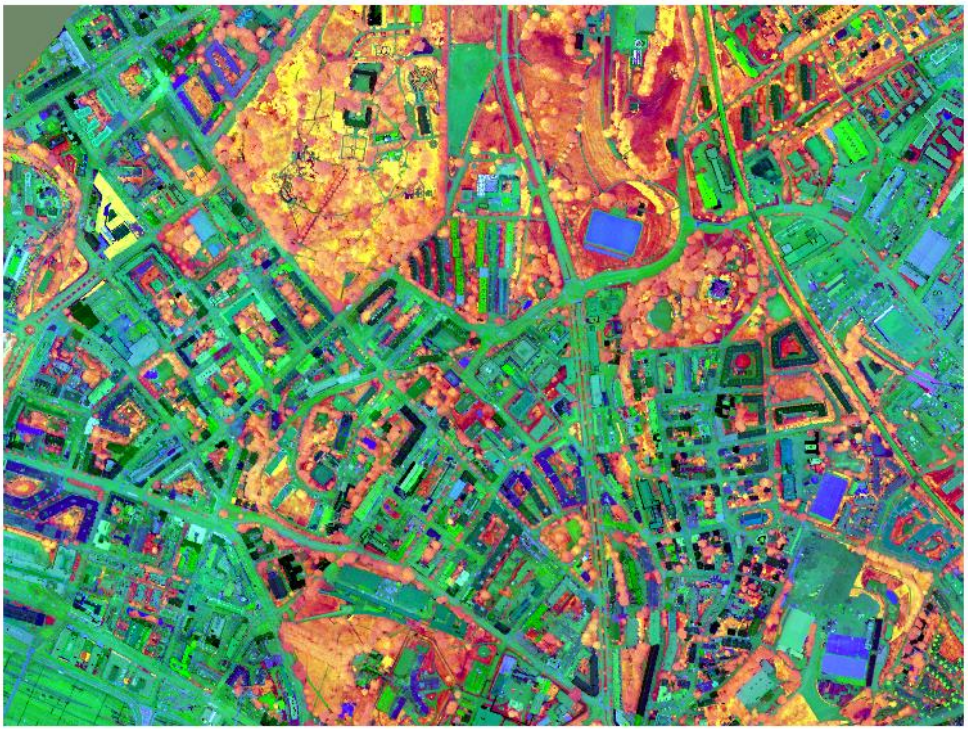


Figur 58: Scores-bilde for PC3 med tilhørende ladningsplott. Her er det størst variasjon mellom urbane materialer som tak og bygninger, og andre områder som ikke er vegetasjon. Størst bidrag til denne komponenten kommer fra bølgelengder mellom 500 til 600 nm.

Ved å sette sammen de tre første prinsipalkomponentene får man et bedre inntrykk av hvor komponentene bidrar.



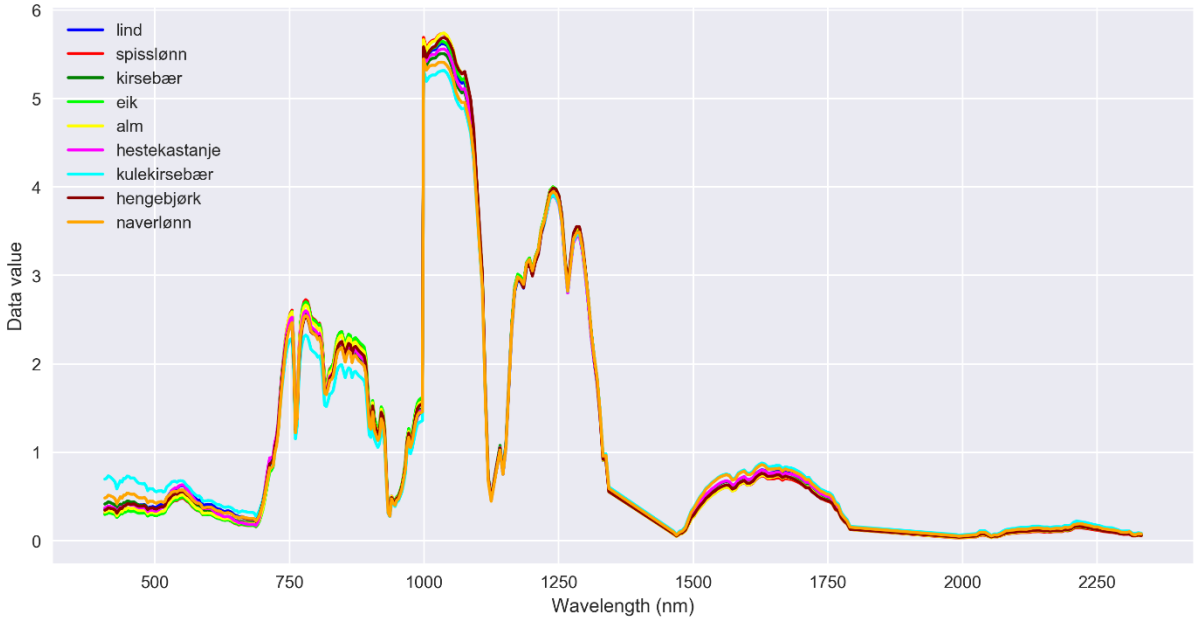
Figur 59: RGB-kompositttbilde av de tre første prinsipalkomponentene for VNIR.
R = PC1, G = PC2, B = PC3. PC1 er representert ved det røde båndet og viser vegetasjon.



Figur 60: RGB-kompositttbilde av PC1, PC2 og PC3 for SWIR. R=PC1, G=PC2,
B=PC3. Her er det store variasjoner i fargenyanser både mellom tak og bygninger.

4.2.2 Evaluering av spektralsignaturer

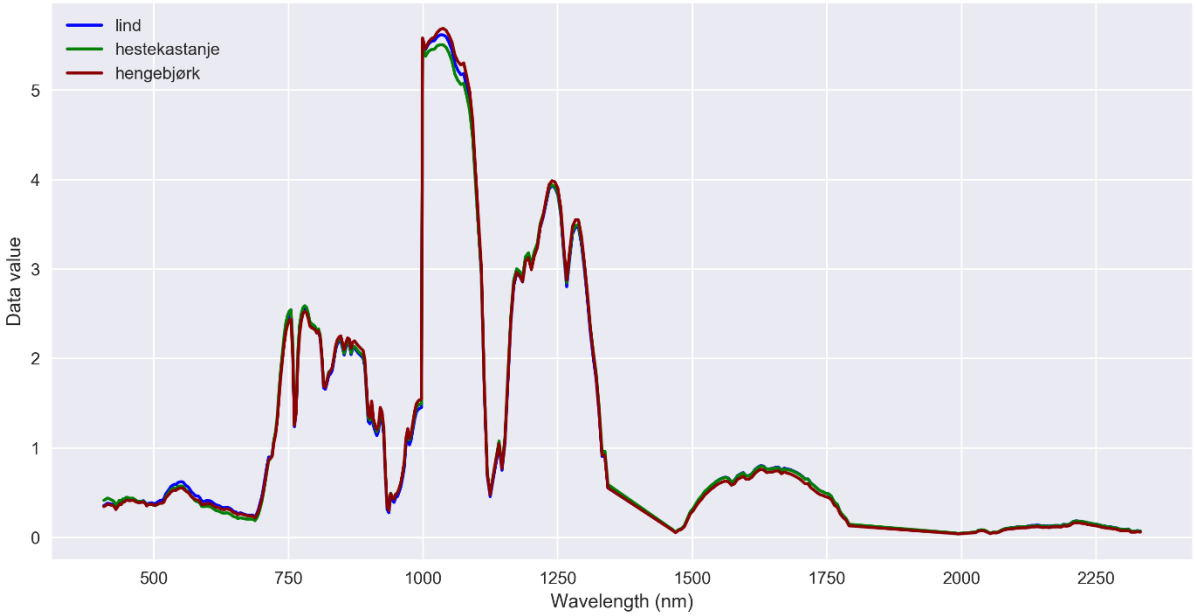
Her presenteres resultatene fra evalueringen av spektralsignaturene. Evaluering ble utført i henhold til beskrivelsen i metoden. Det ble fokusert på spektralsignaturene til de samme klassene som skulle klassifiseres. Evalueringen baserer seg informasjon fra både datasettene VNIR og SWIR.



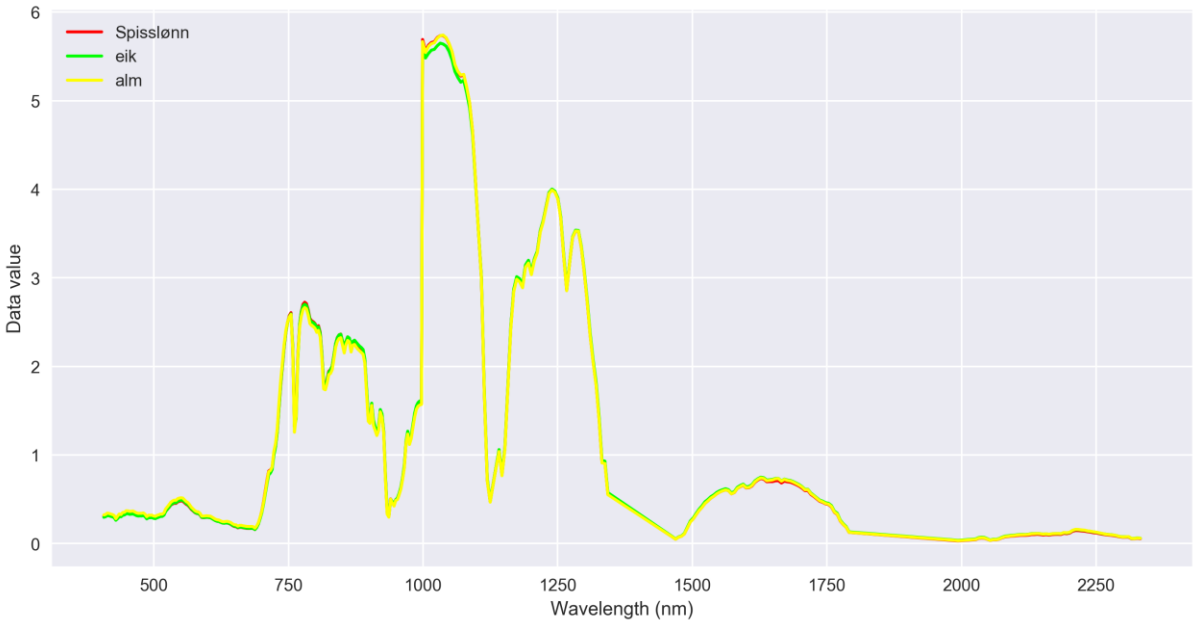
Figur 61: Spektralsignaturer for de ni mest frekvente artene.

Figur 61 viser spektralsignaturene for de ni mest frekvente artene på det store testområdet. Spektralsignaturene er basert på middelverdien for hver klasse. Områder med støy var fjernet i prosesseringen, det tilsvarte de absolutte bunnpunktene i sonene for vannabsorpsjon. Selv om disse områdene var fjernet er fortsatt den karakteristiske formen på sonene for vannabsorpsjon synlige.

Det er tre områder hvor det er størst spredning på spektralsignaturene. Det første er det synlige området (400 til 700 nm). Her er det lokale maksimum i den grønne regionen tydelig, rundt 550nm. Det andre området som skiller seg ut er området mellom 700 og 900 nm. Dette området befinner seg i deler av det nærinfrarøde området. Det tredje området er 1000 nm til ca.1150 nm. Det er spesielt en spektralsignatur som skiller seg ut i disse tre områdene og det er kulekirsebær. Kulekirsebær har høyere verdier i det synlige området, i tillegg til en noe lavere absoluttverdi enn de andre artene. Siden kulekirsebær har en så forskjellig spektralsignatur tyder det på at denne klassen lar seg separere fra de andre.

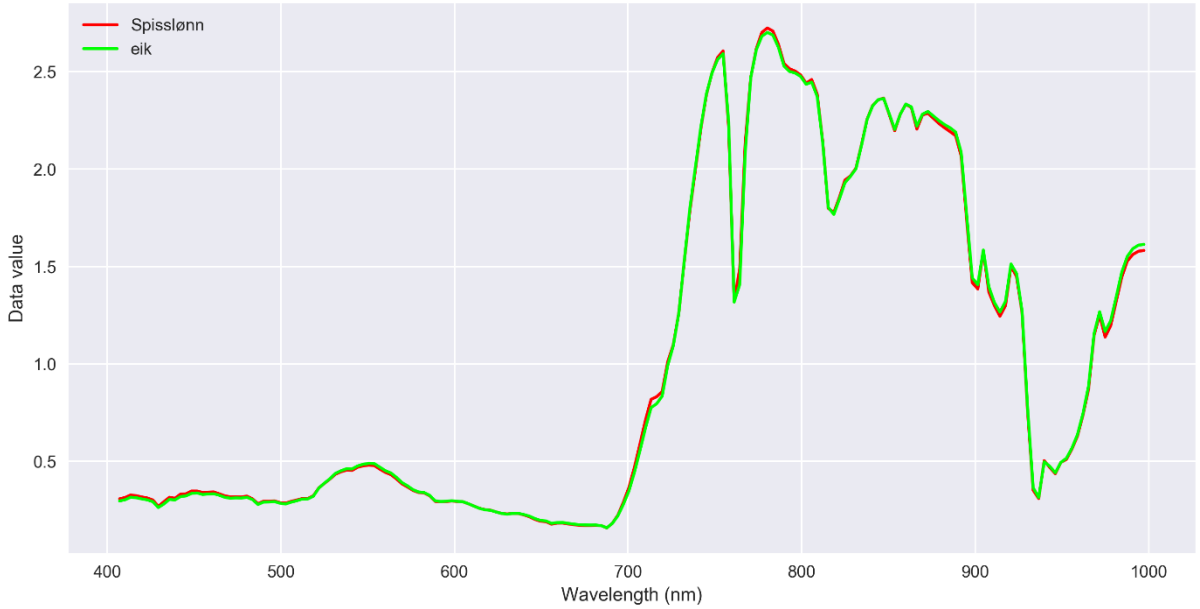


Figur 62: Spektralsignaturer for lind, hestekastanje og hengebjørk. Disse artene har veldig like spektralsignaturer, bortsett fra i området rundt 1000 nm. Dette området stammer fra datasettet SWIR.

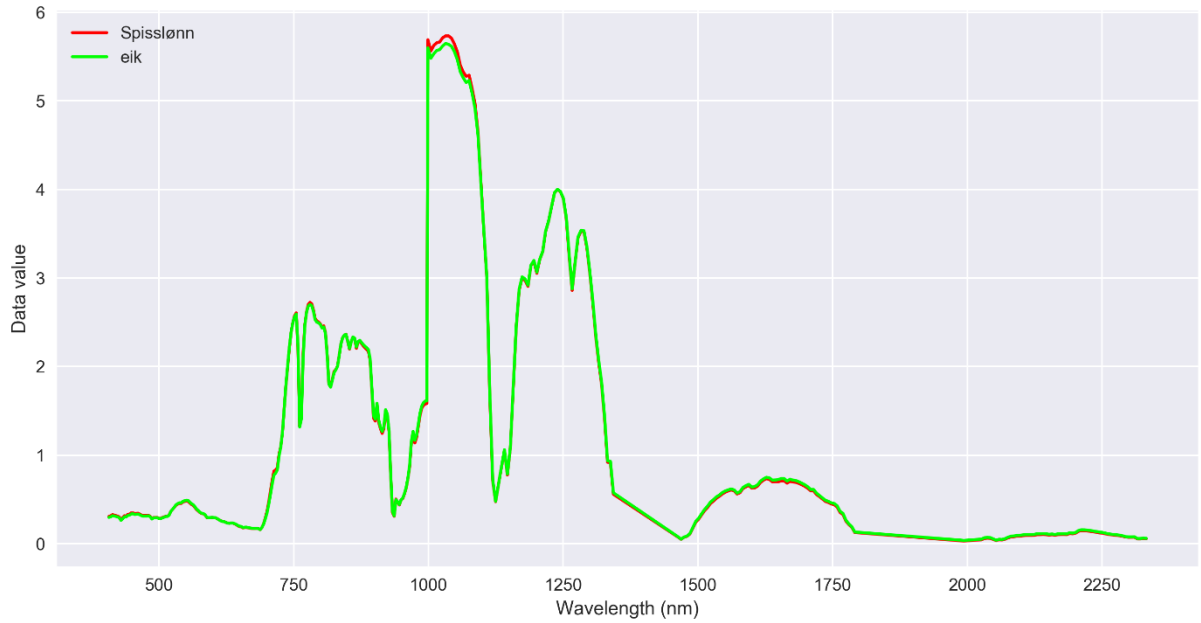


Figur 63: Spektralsignaturer for spisslønn, eik og alm.

Figur 63 viser spektralsignaturene til spisslønn, eik og alm. Spektralsignaturen til Alm er synlig gjennom hele spekteret. Spektralsignaturen til spisslønn er derimot knapt synlig, noe som tyder på at den overlapper med de andre to. For å undersøke hvilke som overlapper ble hver spektralsignatur plottet mot hverandre.



Figur 64: Spektralsignaturer for spisslønn og eik. Eik og spisslønn har så og si identiske spektralsignaturer i datasettet VNIR. Fra figuren kan det se ut som det er størst forskjell i øvre del av spekteret. Bare ekstremalverdiene er forskjellige ellers.



Figur 65: Spektralsignaturene til eik og spisslønn gjennom hele spekteret. Det er et lite område rundt 1000nm hvor det er litt forskjell i spektralsignaturene. Akkurat her kan informasjonen fra SWIR være verdifull for å skille disse artene fra hverandre.

Den visuelle gjennomgangen av spektralsignaturene tyder på at spisslønn, eik og alm kan bli vanskelig å skille siden spektralsignaturene er svært like. Spektralsignaturen til kulekirsebær skiller seg mest ut blant artene. En gjennomgående trend er at spektralsignaturene er

forskjellige rundt 1000 nm, i området fra SWIR. Det tyder på at informasjonen fra SWIR kan være nyttig for å skille artene bedre i en klassifisering. Flere spektralsignaturer ligger i vedlegg A.

4.2.3 Vurdering av spektral separasjon

Etter en visuell gjennomgang av spektralsignaturene ble separasjonen til artene vurdert på bakgrunn av informasjonen fra hver ROI. Det lot seg ikke gjøre å bruke de originale datasettene til å beregne separasjonen, selv om antall piksler i hver klasse var høyere enn antall bånd. Som i den eksplorative analysen ble derfor separasjonen beregnet på bakgrunn av prinsipalkomponentene.

Separasjonen i VNIR ble basert på de fem første prinsipalkomponentene. Disse forklarte 99,27% av variasjonen i datasettet. Separasjonen ble deretter beregnet på bakgrunn av PCA kjørt på både VNIR og SWIR. Her ble de 10 første komponentene benyttet. De forklarte 98,96% av variasjonen i datasettet.

Tabell 14 viser artene som kom dårligst ut av beregningen av separasjon fra PCA på VNIR. Spisslønn og Eik fikk lavest score med en verdi på bare 0,73. Fra den visuelle gjennomgangen av spektralsignaturene kom det frem at disse overlappet hverandre i omtrent hele spekteret. Verdien her gjenspeiler det spektralsignaturene viste. Verdier fra 0 til 1 indikerer dårlig separasjon, mens verdier over 1,9 indikerer god separasjon.

Tabell 15 viser de kombinasjonene som ga best separasjon fra PCA på VNIR. Verdier over 1,9 indikerer god separasjon. Hestekastanje og kulekirsebær ga best resultat. Fra evalueringen av spektralsignaturene kom det frem at kulekirsebær skilte seg ut fra de andre artene.

Tabell 13: Kombinasjonene som ga lavest separasjon.

PCA (VNIR)	
Lavest separasjon	**Verdi**
Spisslønn og Eik	0,73
Eik og Hestekastanje	0,88
Eik og Alm	0,95
Hengebjørk og Alm	1,00
Spisslønn og Hestekastanje	1,01
Spisslønn og Kirsebær	1,03
Spisslønn og Alm	1,17
Hestekastanje og Alm	1,17
Naverlønn og Kulekirsebær	1,21

Tabell 14: Kombinasjonene som ga best separasjon.

PCA (VNIR)	
Best separasjon	
Spisslønn og Hengebjørk	1,58
Hengebjørk og Lind	1,59
Hestekastanje og Lind	1,59
Hestekastanje og Naverlønn	1,72
Lind og Kulekirsebær	1,73
Hengebjørk og Kulekirsebær	1,74
Alm og Kulekirsebær	1,78
Spisslønn og Kulekirsebær	1,80
Eik og Kulekirsebær	1,91
Hestekastanje og Kulekirsebær	1,93

Tabell 15: Kombinasjonene som ga dårligst resultat med VNIR+SWIR.

PCA (VNIR+SWIR)		
Dårligst separasjon	**Verdi**	**% Endring**
Spisslønn og Alm	1,61	▲ 27,31%
Alm og Kirsebær	1,62	▲ 25,89%
Hengebjørk og Alm	1,63	▲ 38,88%
Spisslønn og Hestekastanje	1,66	▲ 39,35%
Spisslønn og Kirsebær	1,68	▲ 38,68%
Spisslønn og Eik	1,71	▲ 57,12%
Hestekastanje og Kirsebær	1,71	▲ 17,84%
Lind og Kirsebær	1,71	▲ 20,51%
Eik og Hestekastanje	1,71	▲ 48,81%
Eik og Kirsebær	1,75	▲ 27,67%

Med både VNIR og SWIR kombinert forbedret resultatet seg med opp til 57% sammenlignet med VNIR for de med dårligst separasjon. Mange kombinasjoner gikk nå fra å ha dårlig separasjon til moderat separasjon (verdier mellom 1 og 1,9).

Tabell 16: Kombinasjonene som ga best resultat med VNIR+SWIR.

PCA (VNIR+SWIR)		
Best separasjon	**Verdi**	**% Endring**
Lind og Naverlønn	1,93	▲ 25,33%
Eik og Naverlønn	1,93	▲ 19,74%
Naverlønn og Kulekirsebær	1,94	▲ 37,62%
Hengebjørk og Naverlønn	1,95	▲ 22,65%
Eik og Kulekirsebær	1,97	▲ 3,07%
Naverlønn og Alm	1,97	▲ 21,63%
Hestekastanje og Kulekirsebær	1,98	▲ 2,39%
Alm og Kulekirsebær	1,98	▲ 10,20%
Spisslønn og Kulekirsebær	1,99	▲ 9,25%
Hengebjørk og Kulekirsebær	1,99	▲ 12,88%

For de kombinasjonene som gå best separasjon var det også en generell forbedring. Jevnt over var ikke forbedringen prosentvis like stor, men mange arter gikk nå fra moderat separasjon til å få god separasjon.

4.2.4 Klassifiseringen

I de neste delkapitlene blir resultatene fra klassifiseringen presentert. Først presenteres parameterverdiene som ble valgt til modellene. Ulike parameterinnstillinger blir deretter gjennomgått og diskutert med tanke på resultatet. Evaluering av resultatet er utført ved hjelp av forvirringsmatriser, presisjon, recall, f1 samt kappa-koeffisienten. Resultatene fra hver evaluering er presentert i egne delkapitler. Til slutt blir resultatene sammenlignet og oppsummert.

Til klassifiseringen ble flere ulike program. For å ha større muligheter til finjustering av parametere falt valget til slutt på Python. Scriptet for klassifiseringen ligger i vedlegg D og E. Tabell 17 viser oversikten over artene med antall trær og antall piksler hver klasse var basert på. Pikselverdiene ble midlet på hvert tre slik det er beskrevet i metoden. SVM presterte best på den eksplorative analysen og ble derfor brukt her også. I tillegg ble logistisk regresjon brukt. Siden datasettet var veldig ubalansert var balansert vekting naturlig å teste ut. Balansert vekting gjør at verdiene blir vektet inverst proporsjonalt med frekvensen i hver klasse.

Tabell 17: Oversikt over klassene.

Artsnavn	Klasse ID	Antall trær	Basert på antall piksler
Lind	19	247	44811
Spisslønn	1	89	41280
Kirsebær	39	61	13077
Eik	7	43	5197
Alm	40	33	17664
Hestekastanje	16	31	9181
Kulekirsebær	41	31	2626
Hengebjørk	14	25	11333
Naverlønn	22	25	3791
Totalt		585	148960

Optimalisering ved GridSearchCV

Funksjonen GridSearchCV ble benyttet til å finjustere parametrene. Scriptet for GridSearchCV ligger i vedlegg D. Tabell 18 viser parameterverdiene som ble testet ut med GridSearchCV på datasettene VNIR og SWIR. For hver algoritme Fant GridSearchCV den kombinasjonen av parameterverdier som ga best score.

Tabell 18: Oversikt over parameterverdier som ble testet

Algoritme	Parameter	Verdier
Logistisk regresjon	**C**	0.001, 0.01, 0.1, 1, 10, 100, 1000
	C	100, 200,300,400,500,600,700,800,900,1000
SVM	**C**	0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0
	Kernel	lineær, RBF
	gamma	0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0

Valgte parameterverdier

Tabell 19: Parameterinnstillinger.

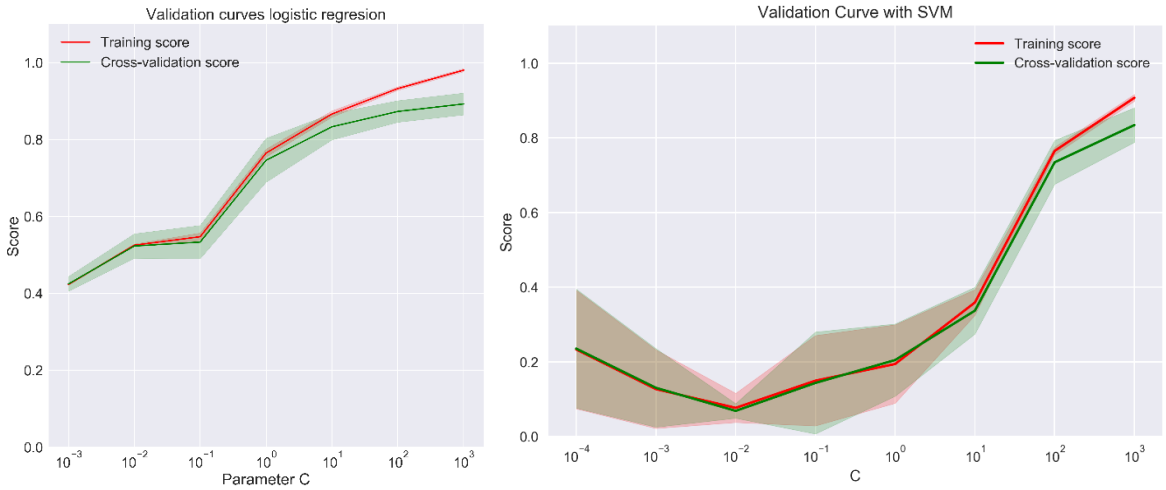
Vekting	Område	modell	GridSearchCV forslag	Valgte verdier
Balansert vekting	VNIR	Logistisk regresjon	C=600	C=10
	VNIR	SVM	C=1000, kernel = linear	C=100
Lik vekting	VNIR	Logistisk regresjon	C=600	C=10
	VNIR	SVM	C=1000, kernel = linear	C=200

Tabell 19 viser parameterinnstillingene for klassifiseringen. For logistisk regresjon ble standardverdien for reguleringen valgt (L2). GridSearchCV valgte relativt høye verdier for C for logistisk regresjon og SVM. Parameteren C i logistisk regresjon regulerer hvor mye frihet modellen får. En høy verdi betyr at modellen får stor frihet til å tilpasse seg dataene. Det kan fort gjøre at modellen blir overtilpasset.

4.2.5 Valideringskurver

Kurvene er basert på data fra treningssettet og er kryssvalidert med 10 like inndelinger. Hver kurve viser gjennomsnittlig score fra kryssvalideringen sammen med standardavvik. Hensikten med valideringskurvene var å finne de parameterverdiene som egnet seg best til klassifiseringen uten at modellen overtilpasser dataene.

Balansert vekting og scoring = accuracy



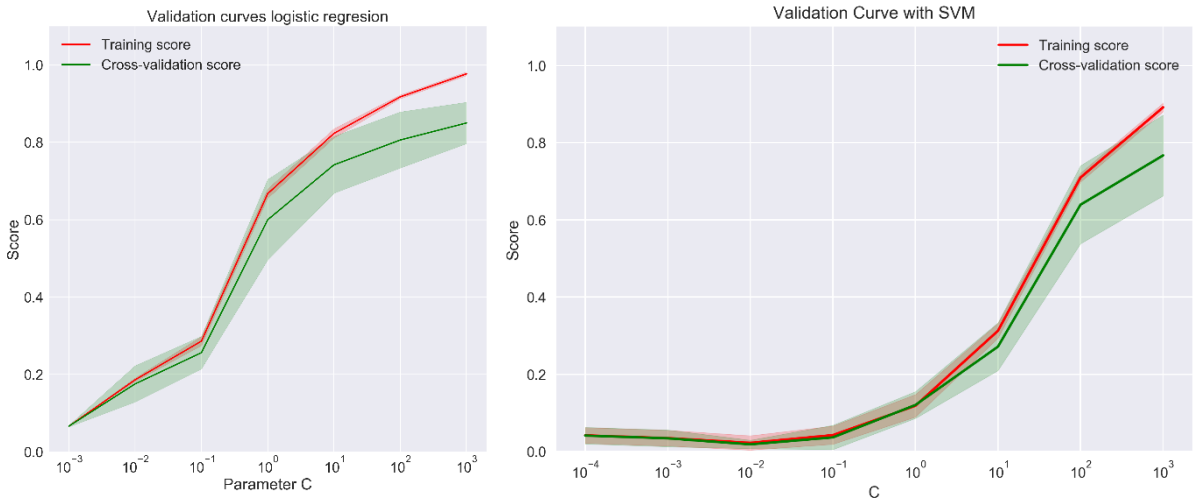
Figur 66: Valideringskurver for parameteren C for logistisk regresjon og SVM (høyre) med balansert vekting og scoring = accuracy.

Til å evaluere parameterverdiene for overtilpasning ble valideringskurver benyttet. Figur 66 viser valideringskurver for logistisk regresjon og SVM med balansert vekting og scoring = accuracy.

For lave verdier av C er både treningsscorene og valideringsscorene lave, begge modellene er undertilpasset her. Fra verdier på C over 10 er treningsscoren høyere enn valideringsscoren for logistisk regresjon. Det indikerer at modellen blir overtilpasset. En verdi på C=10 ble derfor valgt videre for logistisk regresjon med balansert vekting.

For SVM ligger grafene nære hverandre helt til en verdi på C=100. Herfra stiger treningsscoren raskere enn valideringsscoren. Mellom C=100 og C=1000 går treningsscoren utenfor området for valideringsscoren. Treningsscoren er her mer enn ett standardavvik fra valideringsscoren. En konservativ verdi for C ble valgt for å ta høyde for overtilpasning. C ble satt til verdi 100 og kernel til lineær videre.

Balansert vekting og scoring = f1 macro



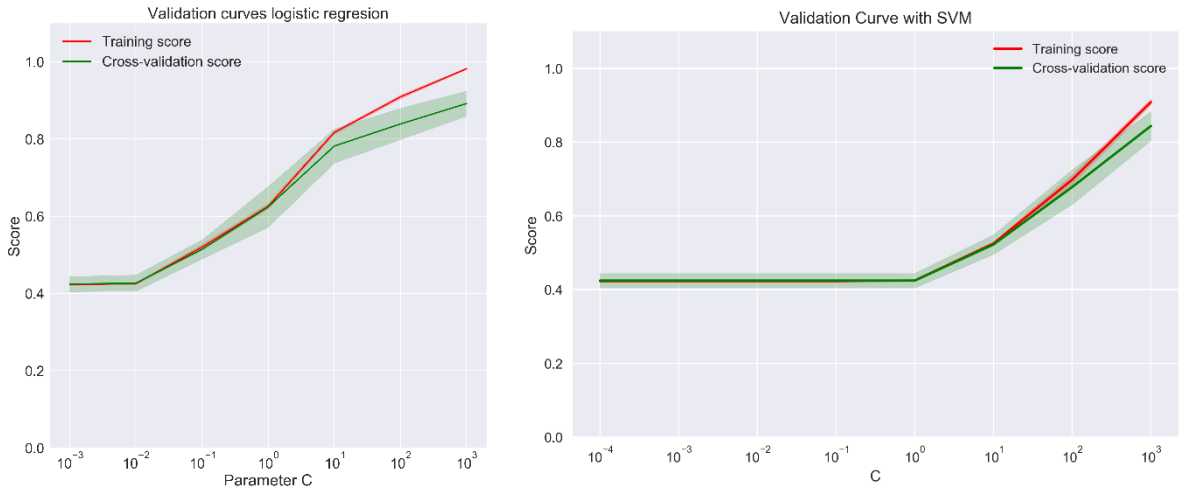
Figur 67: Valideringskurver for logistisk regresjon og SVM med balansert vekting og scoring lik f1macro.

Figur 67 viser valideringskurver med innstillingen balansert vekting og midling ved f1 macro. Midling med f1 macro gjør at hver klasse blir vektet likt. Macro tar ikke hensyn til ubalanserte datasett. Alternativet var vekting med micro, men da får de største klassene mest vekt. En vekting med macro sørget for at hver klasse teller like mye selv om bidraget var lite.

For verdier på C over 10 er treningsscoren høyere enn valideringsscoren for logistisk regresjon. Det indikerer at modellen blir overtilpasset. En verdi på C=10 ble derfor valgt til klassifiseringen med disse innstillingene.

En lineær kernel ble valgt for SVM. Treningsscoren ligger her innenfor et standardavvik fra valideringsscoren omtrent hele veien. Kun ved C=1000 ligger den så vidt utenfor. Treningsscoren stigere raskere enn valideringsscoren fra C=100. Med dette i bakhånd ble C satt til verdien 100. Også her ble en konservativ verdi for C valgt. Kernelen ble beholdt som lineær.

Lik vekting (standardinnstillinger)



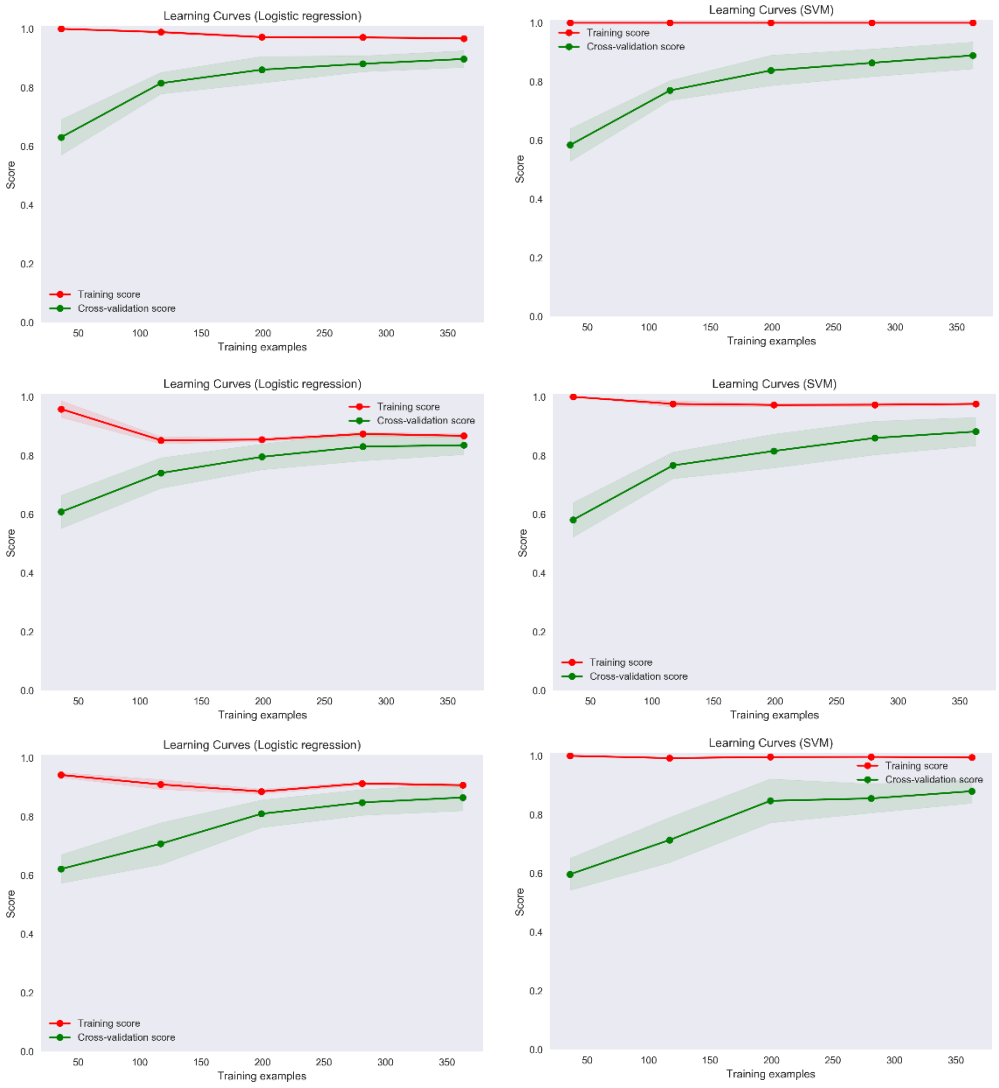
Figur 68: Valideringskurver for logistisk regresjon og SVM med lik vekting (standardinnstillinger).

Med lik vekting får klassene lik vekt. En C på 10 ble valgt for logistisk regresjon. Igjen ble en lineær kernel valgt for SVM. Fra figur 68 kan man se at mellom C = 100 og C = 1000 skiller kurvene lag for SVM. For SVM ble C=200 valgt for videre analyser med lik vekting.

4.2.6 Læringskurver

Læringskurver ble plottet for å vurdere om modellene lider av høy bias eller høy variasjon, og om modellen kan ha nytte av større datasett. Hver figur viser situasjonen med foreslåtte verdier av GridSearchCV, samt situasjonen med valgte verdier.

Balansert vekting



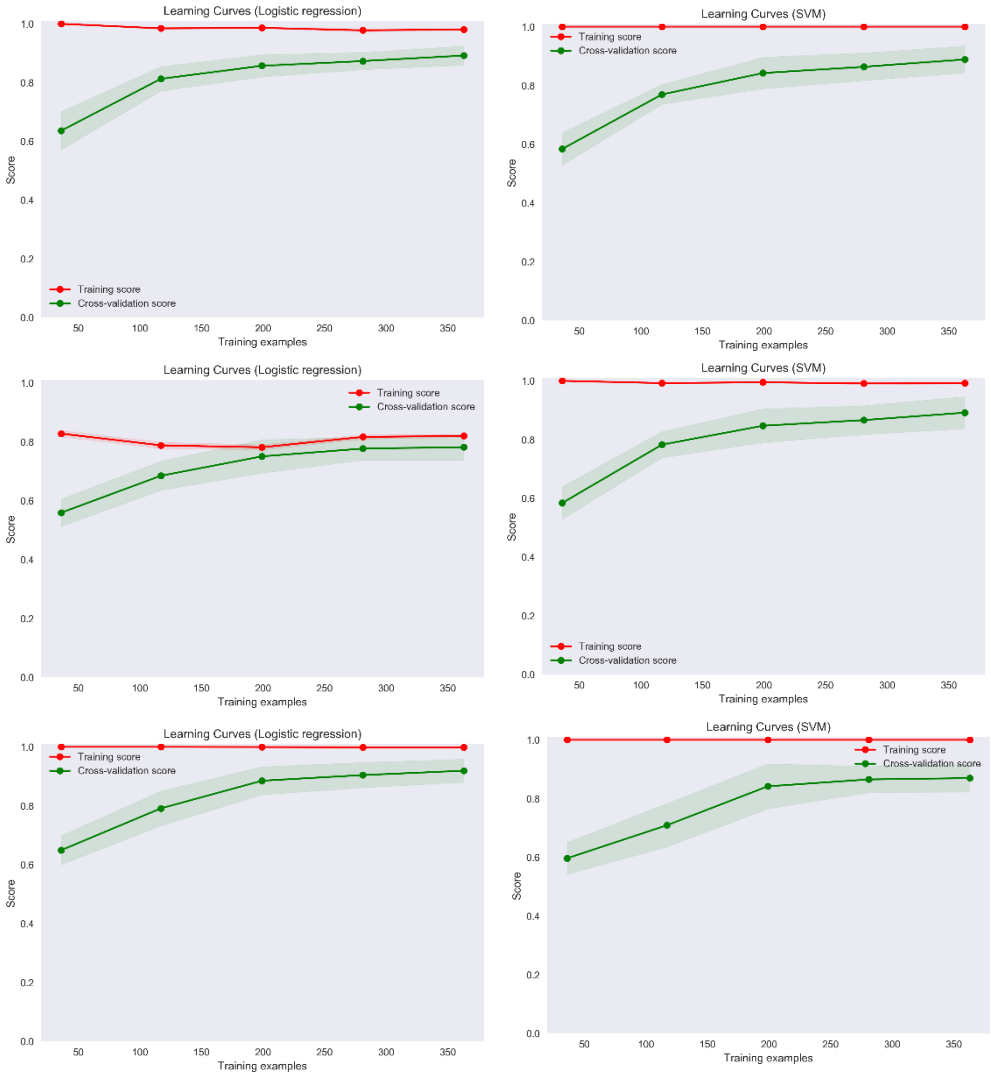
Figur 69: Læringskurver for logistisk regresjon og SVM med balansert vekting

Figur 69 viser læringskurvene for logistisk regresjon og SVM. Øverste rad viser kurvene med innstillinger foreslått av GridSearchCV. Andre rad viser læringskurver for VNIR med valgte verdier på C=10 for logistisk regresjon, og lineær kernel med C=100 for SVM. Nederste rad viser læringskurvene for VNIR+SWIR med samme parameterverdier som VNIR.

På øverste rad holder treningsscoren seg jevnt høyt, mens valideringsscoren øker med større datagrunnlag for både logistisk regresjon og SVM. Fra figur 69 kan man også se at modellene lider av høy varians i starten (overtilpasning), noe som er indikert ved et gap mellom treningsscorene og valideringsscorene. Gapet avtar med større datagrunnlag, noe som tyder på at modellene får en lettere jobb med å definere grenseflaten med større datagrunnlag.

Figuren viser at treningsscorene ligger rundt maksimum for parameterverdiene GridSearchCV foreslår. Læringskurven for logistisk regresjon VNIR (i midten) viser at treningsscoren er veldig høy i begynnelsen, men avtar noe og flater ut senere. Valideringsscoren for samme modell er lav i starten og øker senere. Dette indikerer at det er en modell med noe bias. Algoritmen får en lett jobb med å definere en grenseflate på treningssettet men treffer dårlig på valideringssettet. Kurvene til logistisk regresjon begynner å nærme seg hverandre på verdier rundt 350. Treningskurven for logistisk regresjon og VNIR+SWIR avtar saktere enn for samme modell i VNIR. Det kan tyde på at algoritmene drar nytte av den ekstra informasjonen SWIR gir. Felles for modellene er at de kunne hatt nytte av større treningsgrunnlag.

Lik vekting



Figur 70: Læringskurver for logistisk regresjon og SVM med lik vekting på klassene.

Figur 70 viser læringskurver for logistisk regresjon og SVM med lik vekt på klassene. Tendensen er den samme at valideringssettet kunne hatt nytte større datagrunnlag. Treningsscoren for SVM holder seg jevnt høyt for C1000 og C=200 så det kan tyde på at det ikke er så stor forskjell på disse parameterverdiene. Algoritmen får en lett jobb med å lage en grenseflate for treningssettet. For logistisk regresjon gir VNIR+SWIR høyere treningsnøyaktighet enn sammenlignet med VNIR. Verdiene konverterer også mot en lavere score i VNIR for logistisk regresjon.

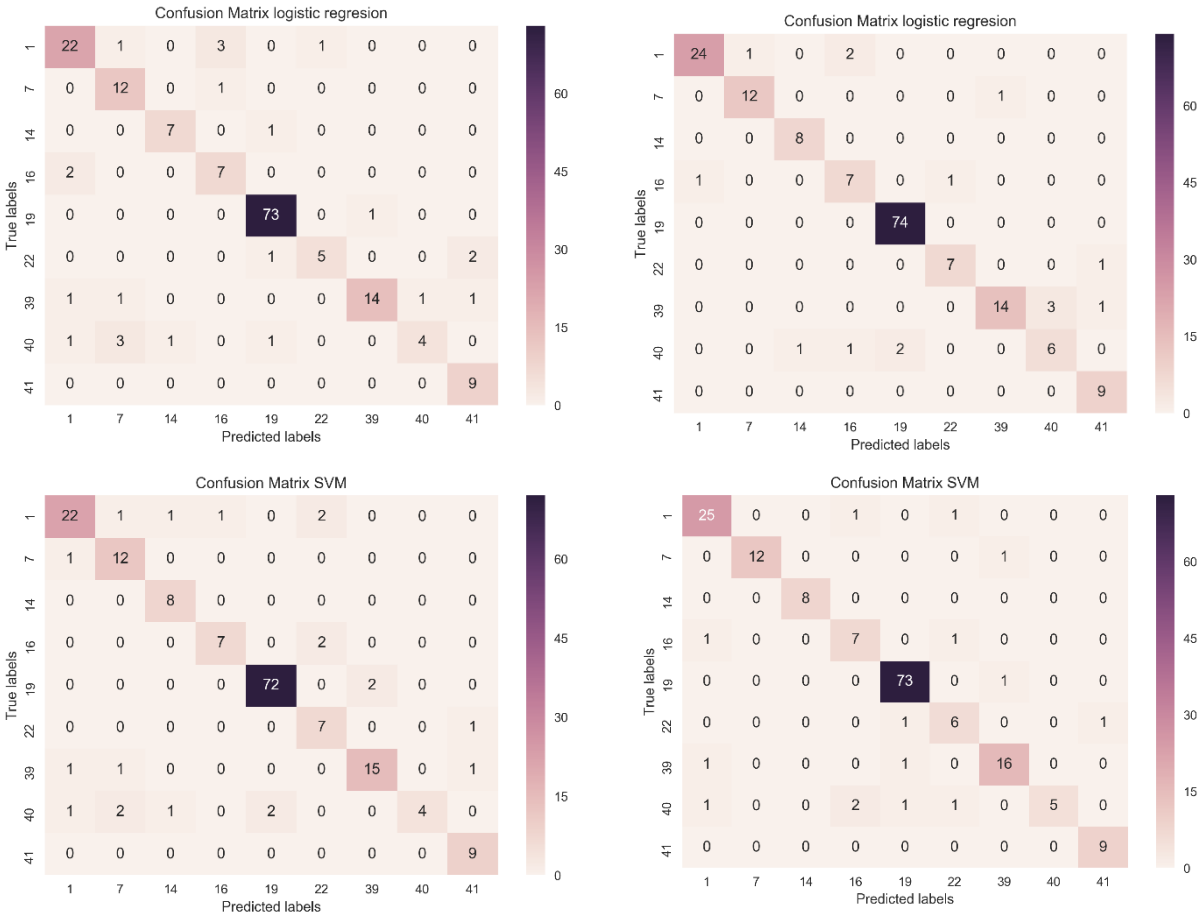
4.2.7 Forvirringsmatriser

Forvirringsmatriser basert på testdatasettet ble brukt til å evaluere og sammenligne resultatene fra klassifiseringen. Forvirringsmatrisene for lik vekting og balansert vekting blir gjennomgått og diskutert. Tabell 21 viser fordeling blant klassene i testdatasettet.

Tabell 20: Oversikt over testdatasettet.

Art	klasse_ID	Antall
Lind	19	74
Spisslønn	1	27
Kirsebær	39	18
Eik	7	13
Alm	40	10
Hestekastanje	16	9
Kulekirsebær	41	9
Hengebjørk	14	8
Naverlønn	22	8
Totalt		176

Balansert vekting

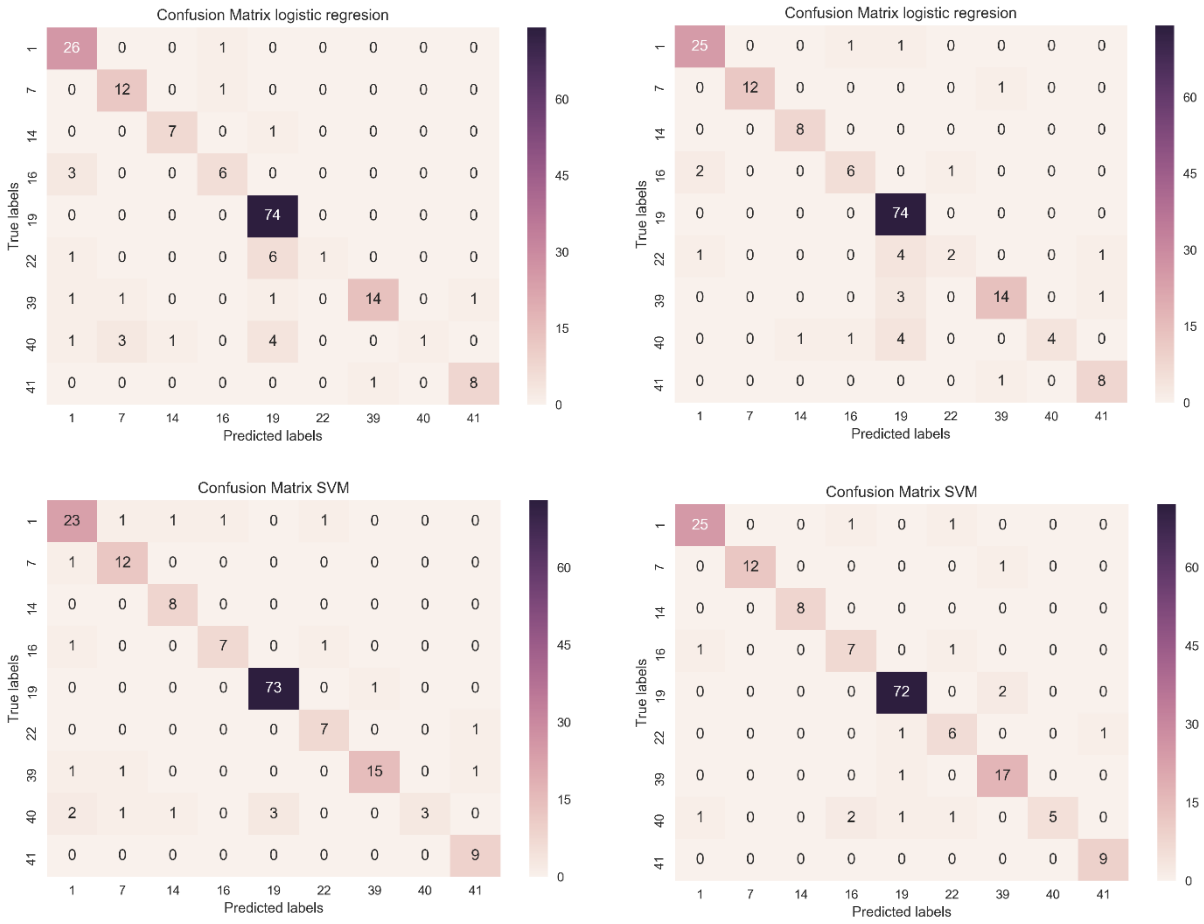


Figur 71: Forvirringsmatriser med balansert vekting på klassene. Første kolonne er VNIR, andre kolonne er VNIR+SWIR. Tallene angir antall trær.

Med balansert vekting er vektene til hver klasse justert inverst proporsjonalt med frekvensen i klassen (antall trær). Det betyr at mindre klasser får større vekt enn store klasser. Slik tar denne vektingen høyde for ubalansen i datasettet. Figur 71 viser forvirringsmatrisene med balansert vekting. Diagonalen viser antall korrekt klassifiserte verdier. Hver rad viser hva trær tilhørende en klasse har blitt klassifisert som. Fra figuren kan man se at de fleste verdiene ligger langs diagonalen, det er få feilklassifiserte trær. Klasser som kom dårligst ut med tanke på feilklassifisering var blant annet klasse 1, klasse 39 og klasse 40.

Med SWIR blir generelt flere klasser korrekt klassifisert. Total nøyaktighet på datasettet VNIR er 86,9% for logistisk regresjon og 88,6 % for SVM. Med VNIR+SWIR er total nøyaktighet 91,5% for logistisk regresjon og 91,5% for SVM.

Lik vekting



Figur 72: Forvirringsmatriser med lik vekting på klassene. Første kolonne er VNIR, andre kolonne er VNIR+SWIR. Tallene angir antallet trær.

Figur 72 viser forvirringsmatriser med lik vekt på klassene. Det første man legger merke til her er at de største klassene ser ut til å ha fått flest korrekt klassifiserte trær i forhold til fordelingen, særlig i VNIR. Det er antagelig en sideeffekt av at klassene er såpass ubalanserte. Algoritmene har fått en lettere jobb med å klassifisere de største klassene rett og slett fordi datagrunnlaget er større. Se for eksempel på klasse 19 og klasse 1. Disse representerer de to største klassene med henholdsvis 74 og 27 trær i testsettet. Med logistisk regresjon har 74 av 74 lind blitt klassifisert. Med SVM har 72 og 73 av 74 lind blitt klassifisert. Klasser som har blitt forvekslet her er blant annet klasse 22 mot klasse 19, klasse 39 mot klasse 19, og klasse 40 mot klasse 1,7 og 19.

Med SWIR har generelt flere klasser blitt korrekt klassifisert. Total nøyaktighet på datasettet VNIR er 84,1% for logistisk regresjon og 89,2 % for SVM. Med VNIR+SWIR er total

nøyaktighet 85,2% for logistisk regresjon og 91,5% for SVM. SVM har altså gjort en bedre jobb med klassifiseringen enn logistisk regresjon.

4.2.8 Evaluering av modellenes prediksjonsevne

Til å bedømme kvaliteten på modellene ble i tillegg metrikkene presisjon (precision), recall og f1 brukt. Disse er også nyttige for å sammenligne ulike modeller og se hvilke som presterer best. kappa-koeffisienten ble også beregnet for å evaluere prediksjonsevnen til modellene. Denne koeffisienten tar høyde for tilfeldigheter.

Balansert vekting

Tabell 21: Precision, recall og f1 for logistisk regresjon med balansert vekting. Tall i parentes (grønn) indikerer endring fra VNIR.

Logistisk regresjon				
VNIR				
class	**precision**	**recall**	**f1-score**	**support**
1	0,85	0,81	0,83	27
7	0,71	0,92	0,8	13
14	0,88	0,88	0,88	8
16	0,64	0,78	0,7	9
19	0,96	0,99	0,97	74
22	0,83	0,62	0,71	8
39	0,93	0,78	0,85	18
40	0,8	0,4	0,53	10
41	0,75	1	0,86	9
avg / total	0,88	0,87	0,86	176
VNIR+SWIR				
class				
1	0,96	0,89	0,92	27
7	0,92	0,92	0,92	13
14	0,89	1	0,94	8
16	0,7	0,78	0,74	9
19	0,97	1	0,99	74
22	0,88	0,88	0,88	8
39	0,93	0,78	0,85	18
40	0,67	0,6	0,63	10
41	0,82	1	0,9	9
avg / total	0,92 (+0,04)	0,91 (+0,04)	0,91 (+0,05)	

Tabell 22: Precsion, recall og f1for SVM med balansert vekting. Tall i parentes (grønn) indikerer endring fra VNIR.

SVM				
VNIR				
class	precision	recall	f1-score	support
1	0,88	0,81	0,85	27
7	0,75	0,92	0,83	13
14	0,8	1	0,89	8
16	0,88	0,78	0,82	9
19	0,97	0,97	0,97	74
22	0,64	0,88	0,74	8
39	0,88	0,83	0,86	18
40	1	0,4	0,57	10
41	0,82	1	0,9	9
avg/total	0,9	0,89	0,88	176
VNIR+SWIR				
class	precision	recall	f1-score	support
1	0,89	0,93	0,91	27
7	1	0,92	0,96	13
14	1	1	1	8
16	0,7	0,78	0,74	9
19	0,96	0,99	0,97	74
22	0,67	0,75	0,71	8
39	0,89	0,89	0,89	18
40	1	0,5	0,67	10
41	0,9	1	0,95	9
avg/total	0,92 (+0,02)	0,91 (+0,02)	0,91 (+0,03)	

Tabell 21 viser en oversikt over presisjon, recall og f1-verdien for logistisk regresjon med balansert vekting. De fleste klassene har høye verdier for både recall, presisjon og f1. Det indikerer at modellen er god og predikerer riktig i de fleste tilfellene. Noen få klasser har lavere verdier. Klasse 40 har eksempelvis en recall på 0,4 og en presisjon på 0,8. Det betyr at modellen gjenkjenner 40% av verdiene i klasse 40, og blant de 40% tilhører 80% faktisk klasse 40. Support forteller antallet verdier i hver klasse.

De største klassene har jevnt over en noe høyere score enn de små klassene. Fra tabellen kan man likevel se at både enkelte små klasser har høyere score enn større klasser. Klasse 14 har eksempelvis bare 8 trær i testdatasettet, men likevel høyere score i VNIR enn klasse 1 med 27 trær. Med VNIR+SWIR er forskjellene mellom klassene mindre enn for bare VNIR.

Verdiene er jevnere i VNIR+SWIR og det er lite forskjell mellom recall og presisjon. Både recall, presisjon og f1-verdien har høye verdier på datasettet VNIR+SWIR. Det viser at modellen presterer bedre i VNIR+SWIR enn i VNIR. Det kan man også lese ut i fra nederste rad i tabellen. Nederste rad gir vektet middelverdi til presisjon, recall og f1-verdien, der vektene er verdiene i support. Middelverdiene er høyere for VNIR+SWIR enn bare VNIR. Modellen gjenkjenner en større andel av verdiene her og andelen som faktisk tilhører en klasse er høyere. Klasse 40 har eksempelvis en jevnere fordeling mellom recall og presisjon i VNIR+SWIR og høyere f1-verdi enn i VNIR.

Tabell 22 viser presisjon, recall og f1 for SVM med balansert vekting. I gjennomsnitt har SVM predikert 90% av klassene korrekt på VNIR og 92% korrekt på VNIR+SWIR. Også SVM gir jevnt over høye verdier på presisjon, recall og f1. Det tyder på at modellen har gjort en god jobb. Unntaket er her også klasse 40 som har lav recall men høy presisjon. Med begge datasettene er det faktisk en klasse som har blitt predikert 100% riktig, nemlig klasse 14.

Totalt sett har SVM i gjennomsnitt gjort en noe bedre jobb med å predikere verdiene i VNIR enn logistisk regresjon med balansert vekting. Med VNIR+SWIR har modellene predikert like bra, middelverdiene for presisjon, recall og f1 er like store.

Lik vekting

Tabell 23 viser presisjon, recall og f1 for logistisk regresjon med lik vekting. Logistisk regresjon har i gjennomsnitt en presisjon på 86% på datasettet VNIR og 87% på VNIR+SWIR. De fleste verdiene er høye både for recall, presisjon og f1. To klasser skiller seg derimot ut på VNIR. Både klasse 22 og 40 har veldig lav recall (0,12 og 0,1) men veldig høy presisjon (1). Det betyr at modellen gjenkjenner bare 12% og 10% av verdiene i disse klassene men alle er korrekte.

Tabell 24 viser presisjon, recall og f1 for SVM med lik vekting. SVM har gjort en noe bedre jobb med å predikere verdiene enn logistisk regresjon, med en gjennomsnittlig presisjon på 90% på VNIR og 92% på VNIR+SWIR. I motsetning til logistisk regresjon har SVM gjort en mye bedre jobb med å predikere klasse 22. Her er recall, presisjon og f1 noenlunde like. Klasse 40 har også SVM problemer med å predikere.

Tabell 23: Precsion, recall og f1for logistisk regresjon med lik vekting. Tall i parentes indikerer endring fra VNIR.

Logistisk regresjon				
VNIR				
class	precision	recall	f1-score	support
1	0,81	0,96	0,88	27
7	0,75	0,92	0,83	13
14	0,88	0,88	0,88	8
16	0,75	0,67	0,71	9
19	0,86	1	0,92	74
22	1	0,12	0,22	8
39	0,93	0,78	0,85	18
40	1	0,1	0,18	10
41	0,89	0,89	0,89	9
avg/total	0,86	0,85	0,81	176
VNIR+SWIR				
class	precision	recall	f1-score	support
1	0,89	0,93	0,91	27
7	1	0,92	0,96	13
14	0,89	1	0,94	8
16	0,75	0,67	0,71	9
19	0,86	1	0,92	74
22	0,67	0,25	0,36	8
39	0,88	0,78	0,82	18
40	1	0,4	0,57	10
41	0,8	0,89	0,84	9
avg/total	0,87 (+0,01)	0,87 (+0,02)	0,85 (+0,04)	176

Tabell 24: Presisjon, recall, og f1 for SVM med lik vekting. Tall i parentes indikerer endring fra VNIR.

SVM				
VNIR				
class	precision	recall	f1-score	support
1	0,82	0,85	0,84	27
7	0,8	0,92	0,86	13
14	0,8	1	0,89	8
16	0,88	0,78	0,82	9
19	0,96	0,99	0,97	74
22	0,78	0,88	0,82	8
39	0,94	0,83	0,88	18
40	1	0,3	0,46	10
41	0,82	1	0,9	9
avg/total	0,9	0,89	0,88	176
VNIR+SWIR				
class	**precision**	**recall**	**f1-score**	**support**
1	0,93	0,93	0,93	27
7	1	0,92	0,96	13
14	1	1	1	8
16	0,7	0,78	0,74	9
19	0,96	0,97	0,97	74
22	0,67	0,75	0,71	8
39	0,85	0,94	0,89	18
40	1	0,5	0,67	10
41	0,9	1	0,95	9
avg/total	0,92 (+0,02)	0,91 (+0,02)	0,91 (+0,03)	176

4.2.9 Sammenligning av modellene

Tabell 25: kvalitetsmål for balansert vekting.

	Logistisk regresjon			SVM	
	VNIR	VNIR+SWIR		VNIR	VNIR+SWIR
f1 micro	0,87	0,91		0,87	0,91
f1 macro	0,79	0,86		0,79	0,86
accuracy	0,87	0,91		0,87	0,91
kappa	0,83	0,89		0,85	0,89

Tabell 26: Kvalitetsmål for lik vekting.

	Logistisk regresjon			SVM	
	VNIR	VNIR+SWIR		VNIR	VNIR+SWIR
f1 micro	0,85	0,87		0,85	0,87
f1 macro	0,71	0,78		0,71	0,78
accuracy	0,85	0,87		0,85	0,87
kappa	0,79	0,82		0,86	0,89

Tabell 25 viser blant annet accuracy og kappa-koeffisienten for balansert vekting. Som man kan se av tabellen gir accuracy et veldig høyt tall. Det kommer av et fenomen kalt «accuracy-paradokset» (Tryolabs, 2013): Når accuracy viser et utmerket resultat, men egentlig bare reflekterer den underliggende klassefordelingen. Spesielt ved ubalanserte datasett er accuracy misvisende å bruke. Da er presisjon og recall bedre mål for å evaluere klassifiseringen (Pedregosa et al., 2011).

Med balansert vekting har SVM en noe høyere kappa-koeffisient enn logistisk regresjon, med 0,85 mot 0,83 i VNIR. På VNIR+SWIR har modellene en lik kappa-koeffisient på 0,89. Kappa-koeffisienten måler som nevnt tidligere graden av overenstemmelse og prøver å ta høyde for tilfeldigheter. Modellene har altså større grad av overenstemmelse for VNIR+SWIR enn for VNIR.

Logistisk regresjon har lavere kappa-koeffisient med lik vekting enn med balansert vekting, noe som tyder på at modellen har prestert dårligere med lik vekting. Det indikerte også

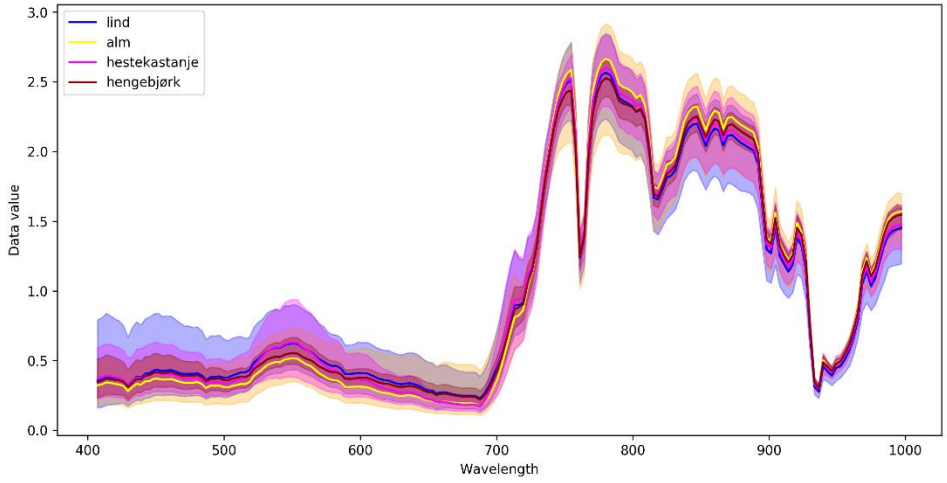
verdiene for presisjon, recall og f1. SVM har omtrent like kappa-koeffisienter med lik vekting som med balansert vekting. Igjen er verdiene høyre med VNIR+SWIR enn bare VNIR.

4.2.10 En nærmere kikk på klasser som skilte seg ut

Evalueringen av klassifiseringen viste at det var enkelte klasser som skilte seg ut. En gjennomgang av presisjon, recall og f1 viste at modellene hadde problemer med å predikere klasse 40. Fra forvirringsmatrisene kan man se at klasse 40 blant annet ble forvekslet med klasse 1,7,14, 16 og 19. Klasse 1 ble blant annet forvekslet med klasse 16. Klasse 14 ble predikert 100% korrekt med lik vekting av SVM, selv om denne klassen bare hadde 8 trær.

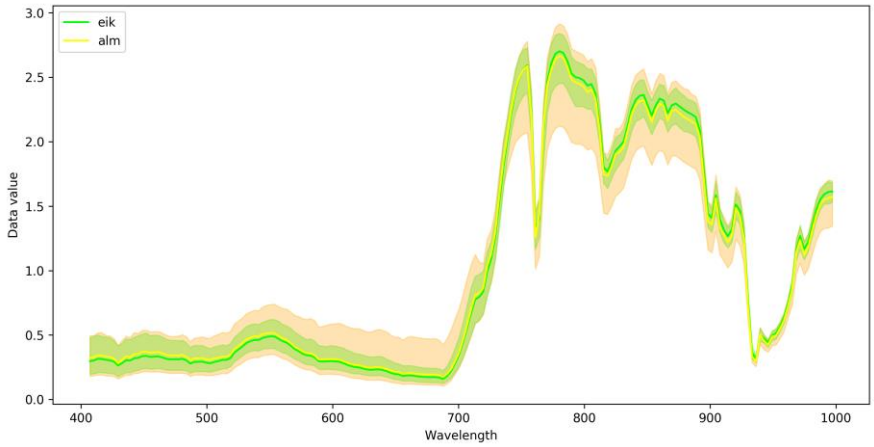
For å se om det var mulig å finne en forklaring på hvorfor disse klassene skilte seg ut ble de undersøkt nærmere.

Klasse 40: alm



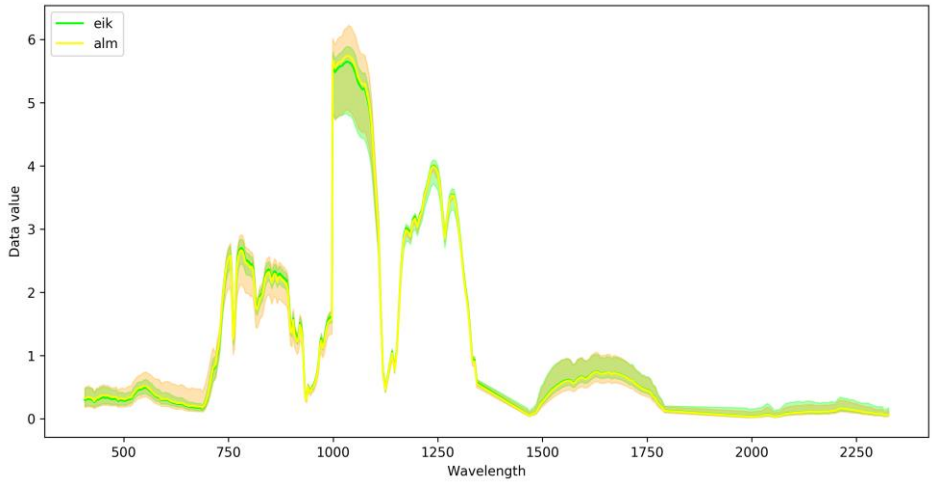
Figur 73: Spektralinformasjon for de klassene alm ble forvekslet med. Enheten er nm for bølgelengdene.

Fra figur 61 kan man se at spektralsignaturen til Alm ligger mellom de andre artene. Figur 73 viser noen av klassene alm ble forvekslet med. Figuren viser også at de artene som ble feilklassifisert som alm har spektralinformasjon som overlapper med området til alm. Dermed er det ikke så rart at modellene har predikert alm dårlig. Her har minimum og maksimumsverdiene til artene blitt lagt på for å se variasjonen innad i hver klasse. Klassifiseringen var basert på middelverdi for hvert tre men hver klasse kan likevel ha stor variasjon fra tre til tre, noe figur 73 viser.



Figur 74: Spektralsignaturer til alm og eik med variasjonen i hver klasse. Enheten er nm for bølgelengdene.

Figur 74 viser spektralinformasjonen til alm (klasse 40) og eik (klasse 7). Eik var en av de artene som hadde størst forveksling med alm på klassifiseringen. Grafen gir et visst innblikk hvorfor. Spektralinformasjonen til eik ligger innenfor spredningen til alm gjennom hele spekteret i VNIR. Dermed ble det vanskelig for modellene å skille disse.



Figur 75: Spektralsignaturene til alm og eik i VNIR og SWIR. Enheten er nm for bølgelengdene.

Med VNIR+SWIR ble ingen alm feilklassifisert som eik. Figur 75 viser at det er størst variasjon mellom eik og alm i bølgelengdeområdet omkring 1000 nm. Informasjonen fra 1000 nm og oppover er hentet fra datasettet SWIR. Fra figuren kan man se at det kun er i dette området at eik ikke ligger innenfor spredningen til alm.

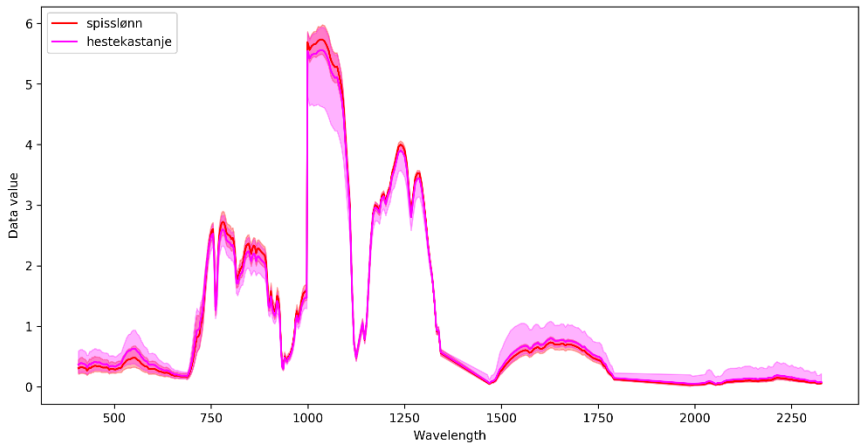
Tabell 27: Tabellen viser ROI-separasjon for alm.

Alm	Jeffries-Matusita	Transformed Divergence
Spisslønn	1,61	1,94
Eik	1,77	1,95
Hengebjørk	1,63	1,79
Hestekastanje	1,80	2,00
Lind	1,76	1,96
Naverlønn	1,97	2,00
Kulekirsebær	1,98	2,00
Kirsebær	1,62	1,91
Kirsebær	1,62	1,91

Tabell 27 viser separasjonen til alm mot de andre klassene basert på Jeffries-Matusita og transformert divergensmatrise. Av tabellen kan man lese at eik har moderat separasjon med klassene spisslønn, eik, hengebjørk, lind, kirsebær og kulekirsebær basert på Jeffries-Matusita. Spektralsignaturene illustrerer dette visuelt.

Klasse 1: spisslønn

Klasse 1 spisslønn ble blant annet predikert som klasse 16 hestekastanje. Disse to klassene ble derfor undersøkt nærmere.



Figur 76:Spektralplott av klasse 1 spisslønn mot klasse 16 hestekastanje med variasjonen i hver klasse. Enheten er nm for bølgelengdene.

Fra figur 76 kan man se at spektralinformasjonen til spisslønn og hestekastanje er like. Men hestekastanje har større variasjon innad i klassen Størst forskjell er det fra 1000nm og

oppover. Forvirringsmatrisene viser da også at modellene har skilt klassene bedre med både VNIR og SWIR med.

4.3 Utfordringer og svakheter

Denne delen tar for seg utfordringer og svakheter med dataene og bygger på egne erfaringer som er gjort underveis.

Håndtering av datamengden

Siden de hyperspektrale bildene hadde så stor romlig og spektral oppløsning var filstørrelsene veldig store. Det gjorde håndteringen av filene problematisk. Reflektansdatasettene utgjorde eksempelvis over 4,7 terabyte med data totalt. Selv om reflektansdataene ble beskjært til mindre områder var fortsatt filstørrelsen betydelig. Nå ble ikke reflektansdatasettet bruk til selve klassifiseringen, bare til utforsking.

De normaliserte radiansdataene var noe mindre med størrelser på rundt 600 mb hver. For å håndtere dataene ble de derfor først beskjært til mindre områder tilpasset testområdene. Likevel utgjorde datamengden totalt flere gigabyte siden det ble kjørt flere analyser på hvert testområde, og det var ni små testområder i tillegg til et stort testområde. Erfaringen er at det er best å begrense datamengden til mindre områder og ikke lagre flere filer enn nødvendig.

Prosessering

Store filer var ikke bare en utfordring i forhold til lagringsplass. Det gjorde også at prosessering av dataene tok tid. Mye bearbeiding og prosessering måtte til for å få resultater fra de hyperspektrale dataene. Formatet på CSV-filen fra ENVI var heller ikke særlig lett å jobbe med. For framtidige analyser anbefales det å undersøke mulighetene med andre programvarer for eksportering av dataene. Når først dataene var strukturert og prosessert ga hyperspektrale data veldig gode resultat. Akkurat nå er metoden for klassifiseringen med hyperspektrale data langt i fra en automatisert prosess. Den har mye utviklingspotensiale.

Prosessering og bearbeiding endrer datamaterialet

Faren med å bearbeide et datasett er at man kan miste informasjon som kan være verdifull. Flere studier har brukt PCA til å redusere datamengden. Selv om de første komponentene

forklarer den største variasjon i datasettet, inneholder også de mindre komponentene noe informasjon, som den eksplorative analysen i denne oppgaven viser. Ved redusering av datamengden bør man være obs på dette.

Skyggeområder

Skyggeområder er en stor utfordring med fjernmålingsdata. Et materiale kan ha store variasjoner i lysforhold på grunn av områder i direkte sollys og områder i skygge. Mengden sollys påvirker egenskapen til et tre og formen på spektralsignaturen til bladene. Det er derfor viktig å ta hensyn til lys og skyggeforhold på en trekrone ved klassifisering. En vanlig måte å gjøre det på er å skille mellom lys og skyggeområder ved uthenting av spektralinformasjonen. I denne oppgaven ble et normalisert datasett brukt for å minimere variasjonene i lysforhold.

Overlapp med andre materialer

Overlapp med andre materialer er en stor utfordring ved klassifisering av trær. En trekrone består av mange blader som varierer i både størrelse og form. De øvre bladene dekker over andre blader under slik at den målte strålingen er en blanding fra bladverk i lys og skygge. Trær varierer dessuten både i størrelse og form. En trekrone kan ha områder hvor bakgrunnen blir blandet med bladverket eller åpne områder hvor bakgrunnen vises gjennom.

Med en romlig oppløsning på henholdsvis 0,3m og 0,7m for VNIR og SWIR består hver trekrone av hundrevis eller tusenvis av piksler i HySpex-datasettene. Med et godt definert utvalgsområde vil bidragene fra andre materialer være små. Ved å være selektiv på utvelgelsen av trær kan risikoen gjøres mindre. Man bør passe på at spektralinformasjonen er hentet godt innenfor materialet som skal undersøkes for å unngå blanding med andre materialer.

Middelverdi

I denne oppgaven ble middelverdien fra hvert tre brukt som utgangspunkt for klassifiseringen. Dette ble gjort for å minske påvirkningen fra andre materialer i tilfelle overlapp. Denne metoden har flere svakheter. For det første risikerer man å miste nyanser i variasjonen som kan være viktig. For det andre er denne metoden sårbar for trær med åpne trekroner hvor bakgrunnen vises. Trær med åpne trekroner eller trær som ikke lot seg identifisere ble ikke

tatt med i klassifiseringen i denne oppgaven. Det ble fokusert på å velge veldefinerte trær og helst enkelttrær for å minske risikoen for at andre materialer ble med.

Ubalansert datasett

Utvalget av bytrær innenfor testområdet var veldig ubalansert. Totalt var 585 trær brukt der den største klassen inneholdt hele 247 trær. Utfordringen med et slikt datasett er at accuracy ikke gir et representativt bilde av ytelsen til modellen men bare representerer den underliggende klassefordelingen. Får å ta høyde for dette ble andre mål en nøyaktighet også brukt, som presisjon, recall og f1.

Lite datagrunnlag

Siden datasettet var såpass ubalansert ble datagrunnlaget veldig lite for enkelte klasser. Treningskurvene viste at modellene kunne trengt mye større datagrunnlag for å få en enda bedre klassifisering. Ideelt sett skulle man også hatt et balansert datasett. Her ble det forsøkt å ta hensyn til ubalanse i datasettet ved å bruke balansert vekting.

Unøyaktige feltdata

Feltarbeid på ni testområder ble utført i forbindelse med oppgaven, i samarbeid med flere studenter som jobbet med de samme dataene. Ved en gjennomgang av målingene fra feltarbeidet viste det seg at kvaliteten på de ikke var spesielt god.

For det første burde trekronene vært målt konsekvent i nord-sør og øst-vest-retning. Dette var ikke gjort. Målingene av trekronene var utført i forskjellige akseretninger på hvert tre. Noen steder var også målingene basert på størst mulige diameter, slik at kronestørrelsen ble kunstig stor. I tillegg var det stor usikkerhet knyttet til artsbestemmelsen av trærne. Mange trær var vanskelig å identifisere i felt på grunn av begrenset adkomst eller sikt. I etterkant viste det seg at en del trær derfor var feilklassifisert.

I denne oppgaven endte jeg opp med å bare bruke ett av de ni testområdene i den eksplorative analysen. Dette området var åpent, godt belyst og bestod bare av enkelttrær. Kronestørrelsen ble ikke brukt til klassifiseringen. Dermed fikk ikke målingene så stor betydning. Likevel, det er viktig at feltarbeidet gjøres grundig siden god kvalitet er avgjørende for resultatet.

5 Konklusjon

Fra de foregående resultatene kommer det frem at hyperspektrale bilder kan brukes til å klassifisere trær i urbane områder og samtidig gi gode resultater. To ulike algoritmer ble benyttet til klassifiseringen, SVM og logistisk regresjon. I tillegg ble to forskjellige måter å vekte dataene på testet, for å se hvilken type vekting som passet best. Balansert vekting tok hensyn til ubalanse i datasettet, mens lik vekting vektet klassene likt.

Resultatene viser at med lik vekting er det større forskjell mellom de små og store klassene enn med balansert vekting, spesielt for logistisk regresjon. Totalt sett har modellene predikert klassene bedre med balansert vekting enn lik vekting, med høyere gjennomsnittlig score på presisjon, recall og f1. Unntaket er SVM som har predikert omtrent like bra med begge typer vekting. I tillegg er verdiene for kappa-koeffisientene høyere med balansert vekting enn lik vekting. SVM har generelt gjort en noe bedre jobb med å skille klassene enn logistisk regresjon, både i VNIR og for VNIR+SWIR.

Problemstillingene gikk ut på å teste hvilke resultater man kunne oppnå for klassifisering av trær med VNIR, og om VNIR+SWIR ville gi et bedre resultat. For å teste dette ble klassifisering først kjørt på kun området VNIR og deretter på VNIR+SWIR. Balansert vekting ga bedre resultat og ble derfor lagt til grunn for å svare på problemstillingene. Følgende resultat ble oppnådd med klassifisering på VNIR:

- Logistisk regresjon ga en kappa-koeffisient på 0,83.
- SVM ga en kappa-koeffisient på 0,85.
- Logistisk regresjon predikerte i gjennomsnitt 88 % av verdiene riktig, mens SVM predikerte 90 % av verdiene riktig (basert på presisjon).
- Total nøyaktighet var 86,9 % for logistisk regresjon og 88,6 % for SVM.
- SVM ga altså noe bedre resultat enn logistisk regresjon og jevnt over høye verdier på presisjon, recall og f1.
- Klasse 40 var en av artene som kom dårligst ut med lav recall men høy presisjon.

Følgende resultat ble oppnådd med klassifisering på VNIR+SWIR:

- Logistisk regresjon ga en kappa-koeffisient på 0,89. En forbedring på 0,06 fra VNIR.
- SVM ga en kappa-koeffisient på 0,89. En forbedring på 0,04 fra VNIR.
- Logistisk regresjon predikerte i gjennomsnitt 92 % av verdiene riktig, mens SVM predikerte 92 % av verdiene riktig (basert på presisjon).
- Total nøyaktighet var 91,5 % for logistisk regresjon og 91,5 % for SVM.

Med SWIR i tillegg til VNIR oppnådde modellene generelt noe bedre resultat enn med VNIR. Med VNIR+SWIR oppnådde SVM 2 % forbedring på presisjon. Logistisk regresjon oppnådde 4 % forbedring på presisjon sammenlignet mot VNIR. Verdiene for presisjon, recall og f1 viser at med data fra SWIR oppnår klassene en jevnere score. Særlig klasser med lav eller middel score i VNIR kommer bedre ut med SWIR i tillegg.

Resultatene er noe overraskende siden kun middelverdien fra hvert tre ble brukt i klassifiseringen. De detaljerte datasettene er antagelig noe av årsaken. De fanger opp skarpe spektraltrekk i spektralsignaturene. Dessuten ble det fokusert på å velge ut godt definerte trær til klassifiseringen og enkelttrær ble prioritert. Det ble gjort for å minske risikoen for blanding med andre materialer.

Konklusjonen er at hyperspektrale flybilder basert på VNIR kan gi gode resultater for klassifisering av trær i urbane områder. Med VNIR+SWIR kan man potensielt oppnå enda bedre resultat. Analysene viste at informasjonen fra SWIR i enkelte tilfeller var avgjørende for å skille problemklasser. Klasser med lav score i VNIR hadde altså mest å hente på informasjonen fra SWIR.

5.1 Forslag til videre arbeid

Hyperspektrale flybilder har stort potensiale. Her er noen forslag til videre arbeid:

- Klassifisering av trær i urbane områder ved å bruke reflektansdata.
- Sammenlign klassifisering med reflektans mot radians.
- Klassifisering av urbane materialer.
- Sammenlign klassifisering med PCA mot originale verdier.
- Klassifisering av trær med utgangspunkt i PCA.
- Hvor mye bedre blir klassifisering av hyperspektrale data sammenlignet med en vanlig sensor?
- Kartlegging av stress eller sykdom i vegetasjon.

Litteraturliste

Bannari, A., Pacheco, A., Staenz, K., McNairn, H. & Omari, K. (2006). Estimating and mapping crop residues cover on agricultural lands using hyperspectral and IKONOS data. *Remote sensing of environment*, 104 (4): 447-459.

Borengasser, M., Hungate, W. S. & Watkins, R. (2007). *Hyperspectral remote sensing: principles and applications*: CRC press.

Bruker Daltonics. (u.å.). Principal Component Analysis

(PCA)

Basics. Tilgjengelig fra: http://research.med.helsinki.fi/corefacilities/proteinchem/pca_introduction_basics.pdf (lest 15.02.2018).

Campbell, J. B. & Wynne, R. H. (2011). *Introduction to remote sensing*: Guilford Press.

Congedo, L. (2017). *Part IV Brief Introduction to Remote Sensing*. Semi-Automatic Classification Plugin Documentation. Tilgjengelig fra: https://www.researchgate.net/profile/Luca_Congedo/publication/307593091_Semi-Automatic_Classification_Plugin_Documentation_Release_5361/links/58a5fae492851cf0e3a5b3d5/Semi-Automatic-Classification-Plugin-Documentation-Release-5361.pdf (lest 24.01.2018).

CRISP. (2001). *Optical Remote Sensing*. Tilgjengelig fra: https://crisp.nus.edu.sg/~research/tutorial/optical.htm (lest 09.02.2018).

Dick, Ø. (2015). *GMBB100 - Bildebruk i geomatikk*. Ås: Norges miljø- og biovitenskapelige universitet (forelesning jan. 2015).

ESRI. (1998). *ESRI Shapefile Technical Description*: Environmental Systems Research Institute, Inc.

FileFormat. *The TIFF File Format*. Tilgjengelig fra: http://www.fileformat.info/format/tiff/corion.htm (lest 09.05.2018).

Grahn, H. & Geladi, P. (2007). *Techniques and applications of hyperspectral image analysis*: John Wiley & Sons.

Hamilton, L. (2014). *Introduction to Principal Component Analysis (PCA)* Tilgjengelig fra: http://www.lauradhamilton.com/introduction-to-principal-component-analysis-pca (lest 16.02.2016).

Harris Geospatial Solutions. (2018a). *Classification*. Using ENVI. Tilgjengelig fra: http://www.harrisgeospatial.com/docs/Classification.html (lest 27.03.2018).

Harris Geospatial Solutions. (2018b). *ENVI*. Tilgjengelig fra: https://www.harris.com/solution/envi (lest 02.04.2018).

Harris Geospatial Solutions. (u.å.-a). *EO-1 Hyperion Vegetation Analysis Tutorial*. Using ENVI. Tilgjengelig fra: https://www.harrisgeospatial.com/docs/hyperionvegetationanalysistutorial.html (lest 25.01.2018).

Harris Geospatial Solutions. (u.å.-b). *Support Vector Machine Background*. Using ENVI. Tilgjengelig fra: http://www.harrisgeospatial.com/docs/BackgroundSVM.html (lest 28.03.2018).

Harris Geospatial Solutions. (u.å.-c). *Vegetation Analysis: Using Vegetation Indices in ENVI*. Tilgjengelig fra: http://www.harrisgeospatial.com/Learn/Whitepapers/TabId/2359/ArtMID/10212/Artic

leID/16162/Vegetation-Analysis-Using-Vegetation-Indices-in-ENVI.aspx (lest 22.01.2018).
Harris Geospatial Solutions. (u.å.-d). *Vegetation and Its Reflectance Properties*. Using ENVI. Tilgjengelig fra: http://www.harrisgeospatial.com/docs/understandingvegetation.html (lest 20.01.2018).
Heiden, U., Heldens, W., Roessner, S., Segl, K., Esch, T. & Mueller, A. (2012). Urban structure type characterization using hyperspectral remote sensing and height information. *Landscape and Urban Planning*, 105 (4): 361-375.
Herold, M., Roberts, D. A., Gardner, M. E. & Dennison, P. E. (2004). Spectrometry for urban area remote sensing—Development and analysis of a spectral library from 350 to 2400 nm. *Remote Sensing of Environment*, 91 (3-4): 304-319.
Hestir, E. L., Khanna, S., Andrew, M. E., Santos, M. J., Viers, J. H., Greenberg, J. A., Rajapakse, S. S. & Ustin, S. L. (2008). Identification of invasive vegetation using hyperspectral remote sensing in the California Delta ecosystem. *Remote Sensing of Environment*, 112 (11): 4034-4047.
Holland, S. M. (2008). Principal Components Analysis (PCA).
Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in science & engineering*, 9 (3): 90-95.
Jensen, R. R., Hardin, P. J. & Hardin, A. J. (2012). Classification of urban tree species using hyperspectral imagery. *Geocarto International*, 27 (5): 443-458.
Landmap. (u.å.). *6.4 Assessing Training Data Quality*. Tilgjengelig fra: http://learningzone.rspsoc.org.uk/index.php/Learning-Materials/Image-Processing-for-ENVI/6.4.-Assessing-Training-Data-Quality (lest 30.04.2018).
Launeau, P., Kassouk, Z., Debaine, F., Roy, R., Mestayer, P. G., Boulet, C., Rouaud, J.-M. & Giraud, M. (2017). Airborne hyperspectral mapping of trees in an urban area. *International journal of remote sensing*, 38 (5): 1277-1311.
Lichtenthaler, H. K., Babani, F., Navrátil, M. & Buschmann, C. (2013). Chlorophyll fluorescence kinetics, photosynthetic activity, and pigment composition of blue-shade and half-shade leaves as compared to sun and shade leaves of different trees. *Photosynthesis research*, 117 (1-3): 355-366.
Lillesand, T. M., Kiefer, R. W. & Chipman, J. W. (2004). *Remote sensing andimage interpretation*. New York: John Wiley&Sons Inc.
Lofthus, H. B. (2018). *Klassifisering av treslag i urbane områder med multispektral laserskanning*. Masteroppgave. Ås: Norges miljø- og biovitenskapelige universitet.
Lowe, A., Harrison, N. & French, A. P. (2017). Hyperspectral image analysis techniques for the detection and classification of the early onset of plant disease and stress. *Plant methods*, 13: 80. doi: 10.1186/s13007-017-0233-z.
McKinney, W. (2010). *Data structures for statistical computing in python*. Proceedings of the 9th Python in Science Conference, Austin, Texas: AQR Capital Management.
MedCalc Software. (2017). Logistic regression.
NASA. (2013a). *The Electromagnetic Spectrum*. Imagine the universe! Tilgjengelig fra: https://imagine.gsfc.nasa.gov/science/toolbox/emspectrum1.html (lest 31.01.2018).
NASA. (2013b). *Landsat 7 Science Data Users Handbook*: NASA's Goddard Space Flight Center.
NASA Earth Observatory. *Normalized Difference Vegetation Index (NDVI)*. I: EVI), M. v. N. (red.). Tilgjengelig fra: https://earthobservatory.nasa.gov/Features/MeasuringVegetation/measuring_vegetation_2.php (lest 09.01.2018).
NASA Earth Observatory. (1999). *Remote Sensing -Introduction and History*. Tilgjengelig fra: https://earthobservatory.nasa.gov/Features/RemoteSensing/ (lest 23.02.2018).

Norsk Elektro Optikk AS. (u.å.). *HySpex Main Spesifications*: Hyspex. Tilgjengelig fra: https://www.hyspex.no/products/all_specs.php (lest 17.01.2018).
Oliphant, T. E. (2006). *A guide to NumPy*, b. 1: Trelgol Publishing USA.
OpenCV. (2016). *Introduction to Support Vector Machines*. Tilgjengelig fra: https://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html (lest 28.03.2018).
Patel, S. (2007). *Chapter 2 : SVM (Support Vector Machine)—Theory*. Machine Learning 101. Tilgjengelig fra: https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72 (lest 28.03.2018).
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825-2830.
Pervez, W. & Khan, S. (2015). Hyperspectral hyperion imagery analysis and its application using spectral analysis. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40 (3): 169.
Piro, P., Porti, M., Veltri, S., Lupo, E. & Moroni, M. (2017). Hyperspectral Monitoring of Green Roof Vegetation Health State in Sub-Mediterranean Climate: Preliminary Results. *Sensors*, 17 (4): 662.
Python Software Foundation. (2018a). *General Python FAQ*. Tilgjengelig fra: https://docs.python.org/3/faq/general.html (lest 02.04.2018).
Python Software Foundation. (2018b). *Python Language Reference, version 2.7.15*. Tilgjengelig fra: http://www.python.org (lest 09.05.2018).
Quantum GIS. (2018). *QGIS User Guide*. Documentation QGIS 2.18. Tilgjengelig fra: https://docs.qgis.org/2.18/en/docs/user_manual/preamble/foreword.html (lest 02.04.2018).
quora. (2017). *overfitting vs underfitting*. quora.
Rascha, S. (2016). *Python Machine Learning*. Birmingham: Packt Publishing.
Richards, J. A. (2013). *Remote sensing digital image analysis*. New York: Springer.
Rodarmel, C. & Shan, J. (2002). Principal component analysis for hyperspectral image classification. *Surveying and Land Information Science*, 62 (2): 115.
SEOS. (u.å.). *Vegetation Indices*. Remote Sensing and GIS in Agriculture. Tilgjengelig fra: http://www.seos-project.eu/modules/agriculture/agriculture-c01-s02.html (lest 09.02.2018).
Shafranovich, Y. (2005). *Common format and MIME type for comma-separated values (CSV) files*. Tilgjengelig fra: https://tools.ietf.org/html/rfc4180 (lest 09.05.2018).
Shippert, P. (2003). Introduction to hyperspectral image analysis. *Online Journal of Space Communication*, 3.
Skarpaas, O. N. N. (2017). *URBAN EEA VEGETATION SURVEY SAMPLE Summer 2017*.
Smith, R. B. (2006). *Introduction to Hyperspectral Imaging*: MicroImages. Tilgjengelig fra: http://www.microimages.com/documentation/Tutorials/hyprspec.pdf (lest 06.10.2017).
Stein, E., Bachmann, M., Heldens, W., Müller, A. & Schmullius, C. (2009). A spectral feature based classification algorithm for characterization of urban surfaces in Munich.
TerraTec AS. (2017). *Rapport for hyperspektral datainnsamling*. Oslo.
TerraTec AS. (2018). *Rapport for hyperspektral tilleggsleveranse*. Oslo.
Thenkabail, P. S. (2001). Optimal hyperspectral narrowbands for discriminating agricultural crops. *Remote Sensing Reviews*, 20 (4): 257-291.
Torbick, N. & Becker, B. (2009). Evaluating principal components analysis for identifying optimal bands using wetland hyperspectral measurements from the Great Lakes, USA. *Remote Sensing*, 1 (3): 408-417.

Tryolabs. (2013). *Why accuracy alone is a bad measure for classification tasks, and what we can do about it*. Tilgjengelig fra: https://tryolabs.com/blog/2013/03/25/why-accuracy-alone-bad-measure-classification-tasks-and-what-we-can-do-about-it/ (lest 05.05.2018).

Wang, J. & Chang, C. (2006). Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis. *IEEE transactions on geoscience and remote sensing*, 44 (6): 1586-1600.

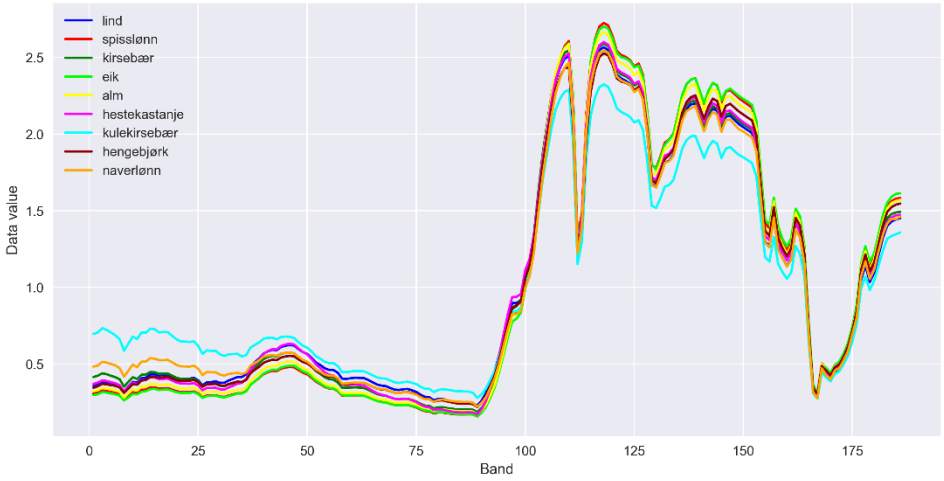
Waskom, M., Botvinnik, O., O'Kane, D., Hobson, P., Lukauskas, S., Gemperline, D. C., Augspurger, T., Halchenko, Y., Cole, J. B., Warmenhoven, J., et al. (2017). *mwaskom/seaborn: v0.8.1 (September 2017)*. Tilgjengelig fra: https://doi.org/10.5281/zenodo.883859 (lest 09.05.2018).

Xiao, Q., Ustin, S. & McPherson, E. (2004). Using AVIRIS data and multiple-masking techniques to map urban forest tree species. *International Journal of Remote Sensing*, 25 (24): 5637-5654.

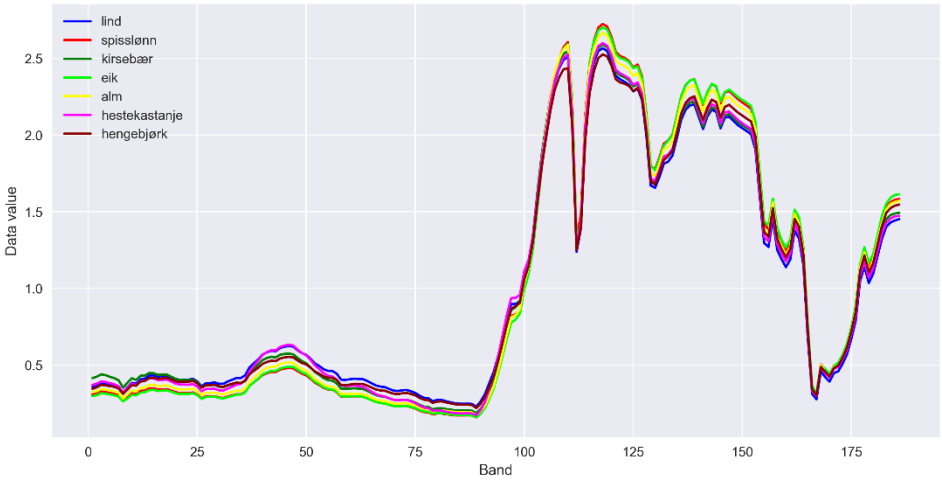
Yu, B., Ostland, M., Gong, P. & Pu, R. (1999). Penalized discriminant analysis of in situ hyperspectral data for conifer species recognition. *Ieee transactions on Geoscience and remote sensing*, 37 (5): 2569-2577.

Vedlegg

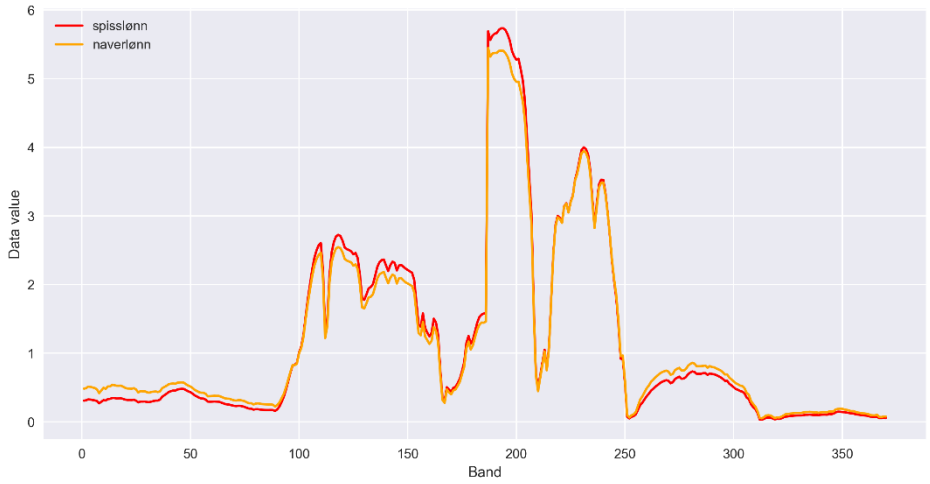
Vedlegg A: Spektralsignaturer



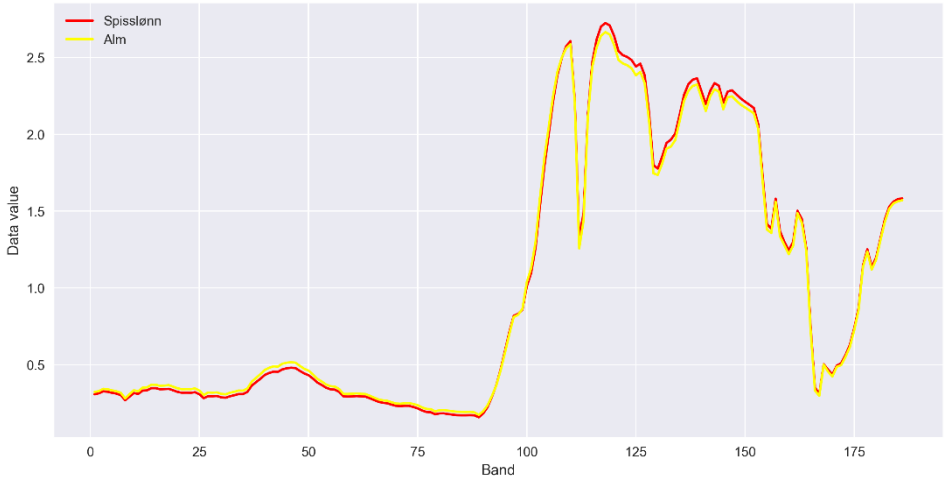
Figur A.1: Spektralsignaturer for de ni mest frekvente artene, vist med datasettet VNIR.



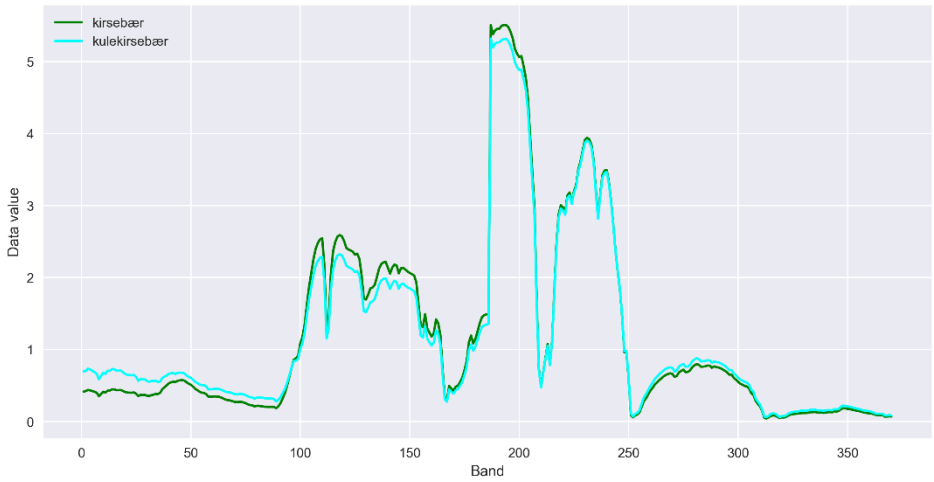
Figur A.2: Spektralsignaturer for utvalgte arter.



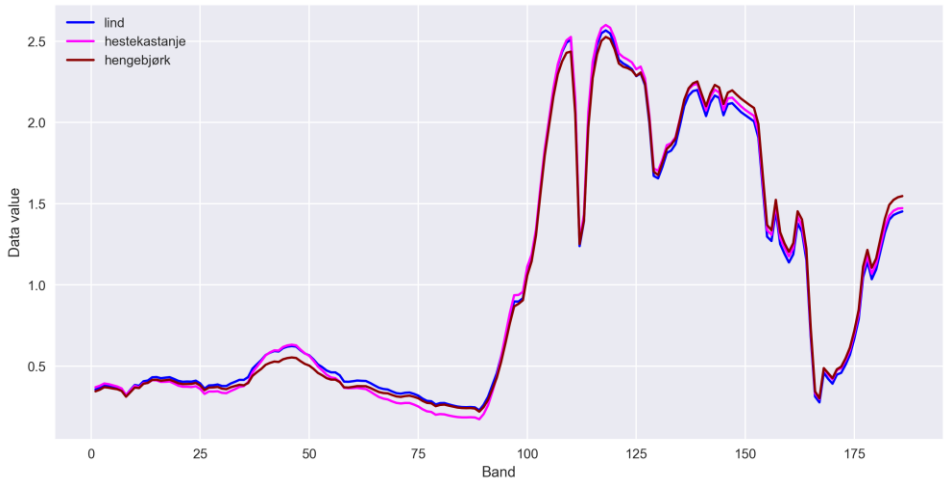
Figur A.3: Spektralsignaturer for spisslønn og naverlønn.



Figur A.4: Spektralsignaturer til spisslønn og alm.



Figur A.5: Spektralsignaturer til kirsebær og kulekirsebær.



Figur A.6: Spektralsignaturer til lind, hestekastanje og hengebjørk.

Vedlegg B: Python script som strukturerer CSV-fil fra ENVI

Les_blandet_format.py

```
# -*- coding: utf-8 -*-
"""
Created on Thu Mar 22 11:41:21 2018

@author: Erik Røstad

Funksjon som leser inn CSV-fil fra ENVI og strukturer filen slik
at den blir lesbar i Python.

Input: filnavn

"""

import pandas as pd

def les_blandet_format(filnavn):
    file = open(filnavn)
    treslag = []

    # Les inn antall ROIs
    linje = file.readline().strip()
    ROIs = int(linje[linje.index(":")+1:])

    # Flytt ned to linjer
    file.readline()
    file.readline()

    # Les inn ROIs
    for i in range(ROIs):
        treslag_linje = file.readline().strip().split(": ")[1]
        file.readline()
        antall_linje = file.readline().strip().split(": ")[1]
        for j in range(int(antall_linje)):
            treslag.append(treslag_linje)
        file.readline()
    treslagDF = pd.DataFrame(treslag, columns=["Treslag"])

    # Les inn overskrifter
    linje = file.readline()
    file.close()

    # Lag overskrifter om til liste
    headers = linje.strip().lstrip("; ").split(", ")

    # Les resterende av filen i CSV-format
    return pd.concat([treslagDF, pd.read_csv(open(filnavn),
skiprows=ROIs*4+5, header=None, names=headers)], axis=1)
```

Vedlegg C: Script som midler verdiene på hvert tre

Middelverdi_CSV.py

```
# -*- coding: utf-8 -*-
"""
Created on Sun Apr  8 10:56:53 2018

@author: Erik Røstad
"""
""" midler pikselverdiene på hvert tre (oppdatert 15.04.2018)"""

# Importerer moduler
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.colors import ListedColormap
import seaborn as sns

# Leser inn CSV fra stort testområde
df = pd.read_csv('nummerert_tredata.csv',sep=',',header=0)

# gir kolonne nytt navn
#df = df.rename(columns={'tre_nr': 'nr'})
df = df.rename(columns={'Originalt': 'Art'})
# lager kopi av df
data = df

# Sletter kolonner som er overflødige
df = df.drop('Lat',1)
df = df.drop('Lon',1)
#df = df.drop('Klasse_CLR',1)
#df = df.drop('Pikselnr',1)


# Velger ut bjørkefam
db = df[df["Originalt"].str.contains("Bjørk|Hengebjørk")]

# Henter ut unike verdier
# u1 er sorterte unike verdier
# ind er indeks til første hendelse av den unike verdien
u1, ind = np.unique(df.nr.values, return_index=True)

# Beregner middelverdi etter nr
dh = df.groupby(['nr']).mean().reset_index()

Count_Row=df.shape[0] #antall rader
Count_Col=df.shape[1] #antall kolonner


#Antall_piksler = df.groupby(['nr']).count()
antall = df['nr'].value_counts()

# Unike arter
unike_arter = df.iloc[ind]

# Legger til data igjen
```

```
dh = pd.merge(dh,
unike_arter[['nr','Originalt','Aggregert','Treslag','BotNavn']], on='nr')

# Endrer kolonnerekkefølge
cols = dh.columns.tolist()
cols = cols[-5:] + cols[:-5] # Flytter de fem sist kolonnene først
dh = dh[cols]


Lagrer til CSV
#df.to_csv('stort_testomraade_num_tredata_full.csv', encoding='utf-8')
#dh.to_csv('stort_tredata_num_tredata_mean.csv', encoding='utf-8')
```

Vedlegg D: Script for optimalisering av parametre ved GridSearchCV

klassifisering_gridsearch.py

```
# -*- coding: utf-8 -*-
"""
Created on Sun Apr 22 17:00:56 2018

@author: errostad
"""

""" GridSearchCV på det hyperspektrale datasettet"""
# GridSearrchCV finner optimale parametre
# logistisk regresjon
# SVM

#  class weight balanced og scoring = accuracy

# cv = 10 # antall folds
# n_jobs: antall modeller som kjøres samtidig

#============================================================================
====
# Importerer moduler
#============================================================================
====
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import svm
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import cohen_kappa_score
from sklearn.learning_curve import validation_curve
from sklearn.model_selection import learning_curve
from sklearn.metrics import precision_score, f1_score, accuracy_score
```

```
#============================================================================
====
# Leser CSV
#============================================================================
====
df = pd.read_csv('num_tredata_mean.csv',sep=',',header=0, encoding='utf-8')
df = df.drop('Unnamed: 0',1)
df = df.drop('TRE_ID_NR',1)
#============================================================================
====
# Antall trær
#============================================================================
====
antall = df['Art'].value_counts()
df["antall"] = [antall[d] for d in df.Art]

#============================================================================
====
# Velger ut klasser med mer enn 24 eksemplar i
#============================================================================
====
data = df[df['antall'] > 24]
#data.to_csv('topp9arter.csv', encoding='utf-8')
# statistikk

#stat.to_csv('stat_topp9arter.csv', encoding='utf-8')
antal_test = data['Art'].value_counts()
stat = data.groupby(["Art", "klasse_ID"]).size().reset_index(name="Antall")
# Plotter fordeling
plt.figure(dpi=300)
sns.countplot(y="Art", data=data)
plt.xlabel('antall')
plt.show()

#============================================================================
====
# Skiller ut spektroskopien (X) og responsen (y)
#============================================================================
====
# VNIR: 1-186, SWIR: 187-370

X = data.iloc[:,7:193] # VNIR
#X = data.iloc[:,7:-1] # SWIR
#X = data.iloc[:,7:-1] # VNIR+SWIR
y = data.iloc[:,0]

## balanserer datasettet
#X, y = make_imbalance(X, y, ratio={1: 60, 19: 60, 39: 60},
#                      random_state=0)


# learning curve function
def plot_learning_curve(estimator, title, X, y, ylim=None, cv=None,
                        n_jobs=1, train_sizes=np.linspace(.1, 1.0, 5)):
    plt.figure(dpi=300)
    plt.title(title)
    if ylim is not None:
        plt.ylim(*ylim)
    plt.xlabel("Training examples")
    plt.ylabel("Score")
    train_sizes, train_scores, test_scores = learning_curve(
```

```
        estimator, X, y, cv=cv, n_jobs=n_jobs, train_sizes=train_sizes)
    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
    test_scores_std = np.std(test_scores, axis=1)
    plt.grid()

    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,
                     train_scores_mean + train_scores_std, alpha=0.1,
                     color="r")
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,
                     test_scores_mean + test_scores_std, alpha=0.1,
color="g")
    plt.plot(train_sizes, train_scores_mean, 'o-', color="r",
             label="Training score")
    plt.plot(train_sizes, test_scores_mean, 'o-', color="g",
             label="Cross-validation score")

    plt.legend(loc="best")
    return plt
#=============================================================================
====
# Deler datasettet i treningssett og testsett, fordeling 70 - 30 %
#=============================================================================
====

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=22,stratify=y)

#scoring = 'f1_macro'
scoring = None
#=============================================================================
====
# Logistisk regresjon
#=============================================================================
====
# C testes opp til verdien 1000
#param_grid = {'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000] }
param_grid = {'C': range(100,1000,100) }
grid = GridSearchCV(LogisticRegression(), param_grid,
cv=10,scoring=scoring)
grid.fit(X_train, y_train) # tilpasser på treningssettet

print("Beste parameterverdi i logreg er %s med en score på %0.2f"
      % (grid.best_params_, grid.best_score_))


#=============================================================================
====
# Plotter C mot nøyaktighet
#=============================================================================
====
grid_mean_scores = [result.mean_validation_score for result in
grid.grid_scores_]
print (grid_mean_scores)

plt.figure()
plt.plot(range(100,1000,100), grid_mean_scores)
#plt.plot([0.001, 0.01, 0.1, 1, 10, 100, 1000], grid_mean_scores)
plt.xlabel('verdi av C for logistisk regresjon')
plt.ylabel('nøyaktighet')
```

```
#=============================================================================
====
# valideringskurve logreg
#=============================================================================
====

#scoring = 'accuracy'
param_range = [0.001, 0.01, 0.1, 1, 10, 100, 1000]
#param_range = range(100,1000,100)
#param_range = [0.001, 0.01, 0.1, 1, 10, 100, 1000]
train_scores, valid_scores = validation_curve(LogisticRegression(),
X_train,y_train, param_name="C", param_range=param_range,
scoring=scoring,cv=10)

print(train_scores)

print(valid_scores)

train_scores_mean = np.mean(train_scores, axis=1)
train_scores_std = np.std(train_scores, axis=1)
valid_scores_mean = np.mean(valid_scores, axis=1)
valid_scores_std = np.std(valid_scores, axis=1)
plt.figure(figsize=(10,10), dpi=300)
plt.title("Validation curves logistic regresion", size=20)
plt.xlabel("Parameter C", size=20)
plt.ylabel("Score", size=20)
plt.tick_params(axis='both', which='major', labelsize=18)

plt.ylim(0.0, 1.1)
plt.semilogx(param_range, train_scores_mean, label="Training score",
color="r")
plt.fill_between(param_range, train_scores_mean - train_scores_std,
                 train_scores_mean + train_scores_std, alpha=0.2,
color="r")
plt.semilogx(param_range, valid_scores_mean, label="Cross-validation
score",
             color="g")
plt.fill_between(param_range, valid_scores_mean - valid_scores_std,
                 valid_scores_mean + valid_scores_std, alpha=0.2,
color="g")
plt.legend(loc="best",prop={'size':18})
plt.show()

#=============================================================================
====
# Evaluering med treningskurve
#=============================================================================
====


title = "Learning Curves (Logistic regression)"
# Cross validation with 100 iterations to get smoother mean test and train
# score curves, each time with 20% data randomly selected as a validation
set.

estimator = LogisticRegression( **grid.best_params_)

plot_learning_curve(estimator, title, X_train, y_train, ylim=(0, 1.01),
cv=10, n_jobs=1)
```

```
#============================================================================
====
# Resultat logistisk regresjon
#============================================================================
====
print("Detaljert klassifiseringsrapport logreg:")
print()
y_true, y_pred = y_test, grid.predict(X_test)
print(classification_report(y_true, y_pred))
print()

print("F1 micro: %1.4f\n" % f1_score(y_test, y_pred, average='micro'))
print("F1 macro: %1.4f\n" % f1_score(y_test, y_pred, average='macro'))
print("F1 weighted: %1.4f\n" % f1_score(y_test, y_pred,
average='weighted'))
print("Accuracy: %1.4f" % (accuracy_score(y_test, y_pred)))

# navn
klasse = [1, 7,14,16,19,22,39,40,41]

print("Forvirringsmatrise logreg")
# Forvirringsmatrise
cm_lr = confusion_matrix(y_test, y_pred)
# plotter confusion matrix i seaborn
plt.figure(dpi=300)
ax= plt.subplot()
lrh = sns.heatmap(cm_lr, annot=True, ax = ax); #annot=True to annotate
cells

# labels, title and ticks
ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix logistic regresion');
ax.set_xticklabels(klasse)
ax.set_yticklabels(reversed(klasse))
plt.show()


# Cohen's kappa score
kappa_lr = cohen_kappa_score(y_true, y_pred)
print("kappa logistisk regresjon:")
print(kappa_lr)

#============================================================================
====
#============================================================================
====
# SVM
##============================================================================
====
param_range = [0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0]
#param_range = [100, 200, 300, 400, 500]
param_grid = [{'C': param_range,'kernel': ['linear']},{'C': param_range,
'gamma': param_range,
'kernel': ['rbf']}]
#svc = svm.SVC()
gs = GridSearchCV(svm.SVC(),param_grid=param_grid,scoring=scoring,cv=10)
gs = gs.fit(X_train, y_train)

print()
print( "SVM:")
```

```
print('Beste C:',gs.best_estimator_.C)
print('Beste Kernel:',gs.best_estimator_.kernel)
print('Beste gamma:',gs.best_estimator_.gamma)

print("Beste parameterverdi er %s med en score på %0.2f"
      % (gs.best_params_, gs.best_score_))

#=============================================================================
====
# Evaluering
#=============================================================================
====
# Valideringskurve
#gamma_range = [10, 1, 0.10, 0.01, 0.001, 0.0001] # gamma
#param_range = [0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0] # C
    # gamma
#train_scores, test_scores = validation_curve(
#    svm.SVC( X_train, y_train, param_name="gamma",
param_range=param_range,
#    cv=10, scoring=scoring, n_jobs=1)
#train_scores_mean = np.mean(train_scores, axis=1)
#train_scores_std = np.std(train_scores, axis=1)
#test_scores_mean = np.mean(test_scores, axis=1)
#test_scores_std = np.std(test_scores, axis=1)
#plt.figure(dpi=300)
#plt.title("Validation Curve with SVM")
#plt.xlabel("$\gamma$")
#plt.ylabel("Score")
#plt.ylim(0.0, 1.1)
#plt.semilogx(param_range, train_scores_mean, label="Training score",
color="r")
#plt.fill_between(param_range, train_scores_mean - train_scores_std,
#                 train_scores_mean + train_scores_std, alpha=0.2,
color="r")
#plt.semilogx(param_range, test_scores_mean, label="Cross-validation
score",
#             color="g")
#plt.fill_between(param_range, test_scores_mean - test_scores_std,
#                 test_scores_mean + test_scores_std, alpha=0.2, color="g")
#plt.legend(loc="best")
#plt.show()

# C (linear)
#scoring = 'f1_macro'
train_scores, test_scores = validation_curve(
    svm.SVC(), X_train, y_train, param_name="C", param_range= param_range,
    cv=10, scoring=scoring, n_jobs=1)
train_scores_mean = np.mean(train_scores, axis=1)
train_scores_std = np.std(train_scores, axis=1)
test_scores_mean = np.mean(test_scores, axis=1)
test_scores_std = np.std(test_scores, axis=1)
plt.figure(dpi=300)
plt.title("Validation Curve with SVM")
plt.ylabel("Score")
plt.xlabel("C")
plt.ylim(0.0, 1.1)
plt.semilogx(param_range, train_scores_mean, label="Training score",
color="r")
plt.fill_between(param_range, train_scores_mean - train_scores_std,
                 train_scores_mean + train_scores_std, alpha=0.2,
color="r")
```

```
plt.semilogx(param_range, test_scores_mean, label="Cross-validation score",
             color="g")
plt.fill_between(param_range, test_scores_mean - test_scores_std,
                 test_scores_mean + test_scores_std, alpha=0.2, color="g")
plt.legend(loc="best")
plt.show()

title = "Learning Curves (SVM)"
estimator = svm.SVC(kernel='linear', C=1000)
plot_learning_curve(estimator, title, X_train, y_train, (0, 1.01), cv=10,
n_jobs=1)
plt.show()

#============================================================================
====
# Resultat
#============================================================================
====
print("Detaljert klassifiseringsrapport SVM:")
print()
y_true, y_pred = y_test, gs.predict(X_test)
print(classification_report(y_true, y_pred))
print()

print("F1 micro: %1.4f\n" % f1_score(y_test, y_pred, average='micro'))
print("F1 macro: %1.4f\n" % f1_score(y_test, y_pred, average='macro'))
print("F1 weighted: %1.4f\n" % f1_score(y_test, y_pred,
average='weighted'))
print("Accuracy: %1.4f" % (accuracy_score(y_test, y_pred)))


# navn
klasse = [1, 7,14,16,19,22,39,40,41]

# Forvirringsmatrise
cm_svm = confusion_matrix(y_test, y_pred)

# plotter confusion matrix i seaborn
plt.figure(dpi=300)
ax= plt.subplot()
svmh = sns.heatmap(cm_svm, annot=True, ax = ax); #annot=True to annotate
cells

# labels, title and ticks
ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix SVM');
ax.set_xticklabels(klasse)
ax.set_yticklabels(reversed(klasse))
plt.show()


# Cohen's kappa score
kappasvm = cohen_kappa_score(y_true, y_pred)
print('kappa SVM')
print(kappasvm)
```

Vedlegg E: Script som kjører klassifisering

klassifisering_eval.py

```
# -*- coding: utf-8 -*-
"""
Created on Fri Apr 27 16:29:25 2018

@author: Erik
"""

""" Klassifisering med logistisk regresjon og SVM"""
#============================================================================
====
# Importerer moduler
#============================================================================
====
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import svm
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import cohen_kappa_score
from sklearn.metrics import precision_score, f1_score, accuracy_score
from sklearn.learning_curve import validation_curve
from sklearn.model_selection import learning_curve

#============================================================================
====
# Leser CSV
#============================================================================
====
df = pd.read_csv('num_tredata_mean.csv',sep=',',header=0, encoding='utf-8')
df = df.drop('Unnamed: 0',1)
df = df.drop('TRE_ID_NR',1)
#============================================================================
====
# Antall trær
#============================================================================
====
antall = df['Art'].value_counts()
df["antall"] = [antall[d] for d in df.Art]

#============================================================================
====
# Velger ut klasser med mere enn 24 eksemplar i
#============================================================================
====
data = df[df['antall'] > 24]
#data.to_csv('topp9arter.csv', encoding='utf-8')
# statistikk
antall2 = data['Art'].value_counts()
stat = data.groupby(["Art", "klasse_ID"]).size().reset_index(name="Antall")
#stat.to_csv('stat_topp9arter.csv', encoding='utf-8')
```

```
# Plotter fordeling
sns.countplot(y="Art", data=data)
plt.show()

#============================================================================
====
# Skiller ut spektroskopien (X) og responsen (y)
#============================================================================
====
# VNIR: 1-186, SWIR: 187-370

#X = data.iloc[:,7:193] # VNIR
X = data.iloc[:,7:-1] # VNIR+SWIR
y = data.iloc[:,0]


#============================================================================
====
# learning curve function
def plot_learning_curve(estimator, title, X, y, ylim=None, cv=None,
                        n_jobs=1, train_sizes=np.linspace(.1, 1.0, 5)):
    plt.figure(dpi=300)
    plt.title(title)
    if ylim is not None:
        plt.ylim(*ylim)
    plt.xlabel("Training examples")
    plt.ylabel("Score")
    train_sizes, train_scores, test_scores = learning_curve(
        estimator, X, y, cv=cv, n_jobs=n_jobs, train_sizes=train_sizes)
    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
    test_scores_std = np.std(test_scores, axis=1)
    plt.grid()

    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,
                     train_scores_mean + train_scores_std, alpha=0.1,
                     color="r")
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,
                     test_scores_mean + test_scores_std, alpha=0.1,
color="g")
    plt.plot(train_sizes, train_scores_mean, 'o-', color="r",
             label="Training score")
    plt.plot(train_sizes, test_scores_mean, 'o-', color="g",
             label="Cross-validation score")

    plt.legend(loc="best")
    return plt

#============================================================================
====
for i in range(1):


#============================================================================
====
    # Deler datasettet i treningssett og testsett, fordeling 70 - 30 %

#============================================================================
====
    X_train, X_test, y_train, y_test = train_test_split(
```

```
        X, y, test_size=0.3, random_state=22,stratify=y)


    # logistisk regresjon
    lr = LogisticRegression(C=10)
    lr.fit(X_train, y_train)

    # svm
    sv = svm.SVC(kernel='linear', C=200)
    sv.fit(X_train,y_train)

## læringskurve
#title = "Learning Curves (Logistic regression)"
#estimator = LogisticRegression( C=10)
#plot_learning_curve(estimator, title, X_train, y_train, ylim=(0, 1.01),
cv=10)
#
#title = "Learning Curves (SVM)"
#estimator = svm.SVC(kernel='linear', C=200)
#plot_learning_curve(estimator, title, X_train, y_train, (0, 1.01), cv=10)
#plt.show()


#=============================================================================
    print()
    print("resultat logistisk regresjon:")
    print()
    print("Detaljert klassifiseringsrapport:")
    print()
    y_true, y_pred = y_test, lr.predict(X_test)
    print(classification_report(y_true, y_pred))
    print()

    print("F1 micro: %1.4f\n" % f1_score(y_test, y_pred, average='micro'))
    print("F1 macro: %1.4f\n" % f1_score(y_test, y_pred, average='macro'))
    print("F1 weighted: %1.4f\n" % f1_score(y_test, y_pred,
average='weighted'))
    print("Accuracy: %1.4f" % (accuracy_score(y_test, y_pred)))

    # Confusion matrix
    cmlr = confusion_matrix(y_test, y_pred)

    # plotter confusion matrix i seaborn
    # navn
    klasse = [1, 7,14,16,19,22,39,40,41]
    plt.figure(dpi=300)
    ax= plt.subplot()
    lrh = sns.heatmap(cmlr, annot=True, ax = ax); #annot=True to annotate
cells

    # labels, title and ticks
    ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels');
    ax.set_title('Confusion Matrix logistic regresion');
    ax.set_xticklabels(klasse)
    ax.set_yticklabels(reversed(klasse))
    plt.show()

    # Calculate Cohen's kappa scores
    kappalr = cohen_kappa_score(y_true, y_pred)
    print("Kappa lr: %1.6f\n" % kappalr)
```

```
================================================================================
===
    print("resultat SVM:")
    print()
    print("Detaljert klassifiseringsrapport:")
    print()
    #y_true, y_pred = y_test, lr.predict(X_test)
    y_truesv, y_predsv = y_test, sv.predict(X_test)
    print(classification_report(y_truesv, y_predsv))
    print()

    print("F1 micro: %1.4f\n" % f1_score(y_test, y_pred, average='micro'))
    print("F1 macro: %1.4f\n" % f1_score(y_test, y_pred, average='macro'))
    print("F1 weighted: %1.4f\n" % f1_score(y_test, y_pred,
average='weighted'))
    print("Accuracy: %1.4f" % (accuracy_score(y_test, y_pred)))

    # Confusion matrix
    cmsv = confusion_matrix(y_test, y_predsv)

    # plotter confusion matrix i seaborn
    plt.figure(dpi=300)
    ax= plt.subplot()
    svh = sns.heatmap(cmsv, annot=True, ax = ax); #annot=True to annotate
cells

    # labels, title and ticks
    ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels');
    ax.set_title('Confusion Matrix SVM');
    ax.set_xticklabels(klasse)
    ax.set_yticklabels(reversed(klasse))
    plt.show()

    # Calculate Cohen's kappa scores
    kappasv = cohen_kappa_score(y_truesv, y_predsv)
    print("Kappa SVM: %1.6f\n" % kappasv)

#prediction = pd.DataFrame(predictions,
columns=['predictions']).to_csv('prediction.csv')

#================================================================================
====
## C (linear)
#param_range = [0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0]
#train_scores, test_scores = validation_curve(
#    svm.SVC(class_weight='balanced'), X_train, y_train, param_name="C",
param_range= param_range,
#    cv=10, scoring='accuracy', n_jobs=1)
#train_scores_mean = np.mean(train_scores, axis=1)
#train_scores_std = np.std(train_scores, axis=1)
#test_scores_mean = np.mean(test_scores, axis=1)
#test_scores_std = np.std(test_scores, axis=1)
#plt.figure(dpi=300)
#plt.title("Validation Curve with SVM")
#plt.ylabel("Score")
#plt.xlabel("C")
#plt.ylim(0.0, 1.1)
#plt.semilogx(param_range, train_scores_mean, label="Training score",
color="r")
#plt.fill_between(param_range, train_scores_mean - train_scores_std,
```

```
#                 train_scores_mean + train_scores_std, alpha=0.2,
color="r")
#plt.semilogx(param_range, test_scores_mean, label="Cross-validation
score",
#             color="g")
#plt.fill_between(param_range, test_scores_mean - test_scores_std,
#                 test_scores_mean + test_scores_std, alpha=0.2, color="g")
#plt.legend(loc="best")
#plt.show()
```

Norges miljø- og biovitenskapelige universitet
Noregs miljø- og biovitskapelege universitet
Norwegian University of Life Sciences

Postboks 5003
NO-1432 Ås
Norway