



Norges miljø- og
biovitenskapelige
universitet

Masteroppgave 2021 30 stp
Fakultet for realfag og teknologi

Endringsanalyse av hogstfelt med Sentinel-1 SAR-bilder

Change detection of deforestation with Sentinel-1
SAR imagery

Ulrik Samdahl Melhuus
Institutt for Geomatikk

Sammendrag

Kartlegging av vegetasjon med Sentinel-2 multispektrale bildesatellitter er velrenommert, og mye brukt den dag i dag. Fordi Sentinel-2 er en bildesatellitt krever den klar sikt til bakken med tilstrekkelig sollys for å tydelig detektere jordoverflaten. Forholdene og årstidene i Norge byr på sine utfordringer med mye skydekke, mørketid og snødekke store deler av året.

I denne oppgaven er endringsanalyse på VV- og VH-polariserte bilder fra Sentinel-1 SAR forsøkt med objekt klassifisering. Forsøket går ut på å se om Sentinel-1 kan utføre endringsanalyse på vintersesongen, da Sentinel-2 multispektrale bilder ikke kan brukes til kartlegging grunnet snødekke. Klassifiseringen er gjort med tre forskjellige algoritmer, Tilfeldig treutvalg (RFC), Støttevektormaskin (SVM) og Nærmeste nabo (KNN).

Området betraktet i oppgaven er skogkommunen Aurskog-Høland. Trenings-/testdatasettet kommer fra endringsanalyse utført med U-Net “deep learning” på Sentinel-2 bilder, levert av Blom oppgaven skrives for. Treningsdatasettet består av totalt 356 hogstflater felt mellom sommeren 2019 og 2020. Bare hogstflatene felt mellom oktober 2019 og april 2020 er tatt med i oppgaven for å se om endringer på vintersesong kan detekteres med SAR-bilder. Klassifiseringen er utført med binært utvalg som er skog- eller hogstflate. Datasettet brukt i oppgaven består totalt av 130 hogstpolygoner og 77 skogpolygoner. En middelvei av hvert polygon på alle SAR-bildene brukes som trenings- og testdata i klassifisering.

Det er tre datasett som testes separat. Datasett en består av SAR-bilder før og etter vintersesongen, altså før og etter hogst på hogstflatene tatt til betraktning i oppgaven. Datasett to består av SAR-bilder før og vintersesongen, altså før og under tiden hogsten i oppgaven er utført. Datasett tre er samme tidspunkt som datasett to, men med bare VH-polarisering. En enkel pikselklassifisering er utført med alle maskinlæringene trent opp på objektklassifiseringene utført på polygonene.

Resultatene med SVM på datasett en og to var så å si helt like med f1-tall på 87,3%. Resultatene samlet indikerer at SAR har tilstrekkelig med informasjon til å detektere endringer i skogområder. Bildeklassifiseringen viser også med Sentinel-1 klarer maskinlæringene å skille ut store deler av hogstene i et utvalgt studieområde. Konklusjonen er at Sentinel-1 har et mulig bruksområde for skogovervåking i Norge, men dette krever videre arbeid.

Abstract

Vegetation mapping with Sentinel-2 multispectral image satellites is reputable and widely used to this day. Because Sentinel-2 is an imaging satellite, it requires a clear view of the ground with sufficient sunlight to detect the earth's surface. The conditions and seasons in Norway offer their challenges with a lot of cloud cover, dark time, and snow cover most of the year.

This thesis attempts to apply a change detection with VV- and VH-polarized images from Sentinel-1 SAR, with object-orientated classification. The experiment examined whether Sentinel-1 can perform change analysis during the winter season, as Sentinel-2 multispectral images can not deliver mapping due to snow cover. Three different classifications were used, Random Forest Classifier (RFC), Support Vector Machine (SVM), and K-Nearest Neighbor (KNN).

The area considered in the thesis is the forest municipality Aurskog-Høland. The training/test dataset comes from change analysis performed with U-Net “ deep learning ” on Sentinel-2 images, provided by Blom, for whom the thesis was written. The training data set consists of 356 harvested fields between the summer of 2019 and 2020. Only the harvested fields between October 2019 and April 2020 are included in the task to see if changes in the winter season could be done with SAR images. The classifications including a binary sample, forest, or deforested area. The dataset used in the thesis consists of a total of 130 deforestation polygons and 77 forest polygons. The raining and test data consist of a median value taken from each polygon on all SAR images.

Three data sets were separately tested. Dataset one consists of SAR images before and after the winter season, which means before and after the deforestations used in this thesis took place. Dataset two consists of SAR images before and during the winter season, which means before and during deforestation. Data set three uses the exact measurements as data set two but with only VH- polarization. As well as object classification, a simple pixel classification with all the machine learning algorithms was tested to output classified images.

The results with SVM on data sets one and two were almost the same with f1 numbers of 87.3 %. The overall results indicate that SAR has sufficient information to detect changes in forest areas. The image classification also shows that with Sentinel-1, machine learning can distinguish large parts of the harvests in a selected study area. The conclusion is that Sentinel-1 can be used in forest monitoring in Norway, but this requires further work.

Forord

Denne masteroppgaven markerer slutten på alle mine fantastiske år ved NMBU. Oppgaven er skrevet i samarbeid med Blom, og det har vært en utrolig lærerik og spennende prosess. SAR fagfeltet har vært helt fremmed og ukjent for meg, og har bydd på sine utfordringer.

Først og fremst vil jeg rekke ut en stor takk til veilederne Floris Jan Groesz og Stian Rostad i Blom for mange gode innspill, og tett oppfølging. Jeg har alltid kunne tatt kontakt når jeg står fast både faglig og teknisk.

Stor takk til hovedveileder Ivar Maalen-Johansen som har gitt gode råd og forslag, og veiding ved behov.

Det siste ett og et halvt året har vært svært begrenset og bydd på utfordringer for meg selv, og de rundt meg. Tusen takk til kjæresten min Hanna, mine nære venner, kollektivet og gjengen på lesesalen som har vært kjempeviktige og gode støttespillere igjennom hele master- og koronaperioden. Takk til Mannskoret Over Rævne, Trøndernes Fagforening, UKA i Ås, Samfunnet i Ås og alle menneskene jeg har blitt kjent med, som har gitt meg opplevelser og erfaring jeg vil BEvare livet ut.

Til slutt vil jeg takke min familie, foreldre og søster som har støttet og hjulpet meg fram til det punktet jeg har kommet til nå.

Ulrik Samdahl Melhuus

Forkortelser

SAR: Synthetic-aperture radar

ML: Maskinlæring

TP: True positive

TN: True negative

FP: False positive

FN: False negative

ESA: European Space Agency

SNAP: Sentinel Application Platform

RFC: Random forest classifier

SVM: Support vector machine

KNN: K-nearest neighbors

VV: Verikal-vertikal

VH: Vertikal-horisontal

HH: Horisontal-horisontal

HV: Horisontal-vertikal

ColHub: Collaborative Data Hub

DTM: Digital terrengmodell

GRD: Ground Range Detection

RGB: Rødt, Grønt, Blått

Innhold

Figurer	viii
Tabeller	x
1 Innledning	1
1.1 Bakgrunn	1
1.2 Problemstilling	1
1.3 Aktualitet og tidligere arbeid	2
1.4 Oppsett og struktur	4
2 Teori	5
2.1 Hogst i Norge	5
2.2 Sentinel-1	5
2.3 Sentinel-2	5
2.3.1 MultiSpectral Instrument	6
2.4 Radar	6
2.5 SAR	7
2.5.1 SAR prinsippet	7
2.5.2 Bånd og bølgelengder	9
2.5.3 Polarisering	9
2.5.4 SAR bilde	10
2.6 Maskinlæring	11
2.6.1 Kryssvalidering	15
2.6.2 GridSearchCV for tilpasning av modell	16
2.6.3 Over- og undertilpasing	16
2.6.4 Forvirringsmatrise	17
2.6.5 Klassifiseringrapporter	18
3 Material og Metode	20

3.1	Materialer	20
3.1.1	Filformater	20
3.1.2	Forsøksområde	21
3.1.3	Sentinel-1 SAR datasettet	21
3.1.4	Hogstdatasett	23
3.2	Programvarer og moduler	25
3.2.1	ESA SNAP	25
3.2.2	QGIS	25
3.2.3	Lucidchart	25
3.2.4	Python	25
3.3	Prosessering av data i SNAP	27
3.4	Metode	28
3.4.1	Datagrunnlag	28
3.4.2	Produksjonsløype maskinlæring	30
3.4.3	Bildeklassifisering	30
4	Resultat	33
4.1	Resultat datasett	33
4.1.1	Datasett en: Før og etter vintersesong	33
4.1.2	Datasett to: Før og under hogst	34
4.1.3	Datasett tre: Før og under hogst med VH-polarisering	35
4.2	Resultat sammenligning	35
4.2.1	Bildeklassifisering	36
5	Diskusjon	40
5.1	Resultater	40
5.2	Usikkerhet i metode	40
5.2.1	Datagrunnlag	40
5.2.2	Preprosessering	41
5.2.3	Strekking av bilder	41
5.2.4	Trenings- og testdata	41
5.2.5	Bildeklassifisering	42
6	Konklusjon	43
6.1	Videre arbeid	43
	Bibliografi	45

Appendix	48
A Vedlegg	48
A.1 Forvirringsmatriser	48
A.2 Tabeller	52
A.3 Pythonkode	53

Figurer

1.1	Resultat tidligere arbeid.	2
1.2	Skygger i SAR måling.	2
2.1	SAR prinsippet.	7
2.2	SAR-penetrering.	9
2.3	SAR refleksjonstyper.	10
2.4	SAR refleksjon.	11
2.5	Valgtre eksempel.	12
2.6	Tilfeldig treutvalg.	13
2.7	Nærmeste nabo.	14
2.8	Støttevektor maskin.	15
2.9	Støttevektor maskin, C variabel.	15
2.10	Kryssvalidering.	16
2.11	Over- og undertilpasning.	17
2.12	Forvirringsmatrise.	17
3.1	Aurskog-Høland.	21
3.2	Filter eksempel til nedlasting.	22
3.3	Eksempel utvalgt nedlasting data.	23
3.4	Utsnitt av trenings-data.	24
3.5	Klassifiserte hogstfelt fra Sentinel-2.	24
3.6	Markert hogst og skog.	25
3.7	SAR-bilde før prosessering	27
3.8	SAR-bilde etter prosessering	27
3.9	Visuell sammenligning av SAR VV-, VH-polarisering og RGB bilde fra april 2020.	29
3.10	Klassifisering illustrasjon.	30
3.11	Klassifisering utvalgt område.	31
3.12	Utsnitt av bilde brukt til bildeklassifisering.	31

3.13	Eksempel på bildeklassifisering.	32
3.14	Produksjonsløype bildeklassifisering.	32
4.1	F1-tall før og etter vintersesong.	34
4.2	F1-tall før og under hogst.	34
4.3	F1-tall før og under hogst med VH-bånd.	35
4.4	Sammenligning av F1-tall.	36
4.5	Gjennomsnitt F1-tall.	36
4.6	RGB-bilde av predikert område.	37
4.7	Bildeklassifisering dårligste resultat med RFC.	38
4.8	Bildeklassifisering beste resultat med SVM.	39
5.1	RGB bilde med SAR data.	41
5.2	Bildeklassifisering med vektorer.	42
A.1	Forvirringsmatriser av klassifiseringer på før og under -hogst data.	49
A.2	Forvirringsmatriser av klassifisering på før og etter -hogst data.	50
A.3	Forvirringsmatriser av klassifiseringer på før og under -hogst, med VH-bånd.	51

Tabeller

2.1	Tabell av båndene i Sentinel-2 A og B MSI sensor.	6
2.2	Oversikt over forskjellige radar bånd.	9
2.3	SAR refleksjonstyper.	10
A.1	Klassifiseringsrapport datasett en.	52
A.2	Klassifiseringsrapport datasett to.	52
A.3	Klassifiseringsrapport datasett tre.	53

1. Innledning

Innledningen beskriver bakgrunnen for oppgaven, problemstillingen, tidligere forskning og aktualitet på forsøket i oppgaven.

1.1 Bakgrunn

Bakgrunnen for oppgaven er at Blom ønsker å se på nye måter å detektere endringer av hogstfelt. En endringsanalyse er et viktig verktøy brukt for å kunne føre statistikk på endring i arealbruk. Manuell innmåling av arealflater er veldig tidkrevende og kostbart, derfor er nyere teknologi med fjernmåling fra fly og satellitt hyppigere brukt for en generell analyse. I denne oppgaven er fokusområdet Aurskog-Høland, en skogkommune på Østlandet. De mest brukte metodene for kartlegging av vegetasjon i dag innebærer i stor grad bruk av optiske multispektrale bildesatellitter som Sentinel-2 brukt av Blom.

Majoriteten av hogst i Norge skjer på vintersesongen [Carlsson and Rönnqvist, 2005]. De optiske sensorene er avhengig av eksterne lyskilder, som sollys for å kunne fange opp objekter i rommet. De krever klar sikt mot bakken og tilstrekkelig med sollys. I Norge er store deler av landet påvirket av snø, mye nedbør (overskyet) og mørketid lange perioder av året.

Satellitt gruppen Sentinel-1 bruker radar målinger, og et system kalt syntetisk aperture-radar (SAR). SAR-målinger bruker et radarinstrument til å måle overflaten på jorden. Radarinstrumentet er en aktiv sensor som sender ut egne signaler, og leser av returen på radarbølgene er den ikke avhengig av sollys og klar sikt. Fordi radarbølger trenger igjennom både skydekke, snø og kan måle dag og natt har Sentinel-1 SAR-målinger en mulig funksjonalitet når Sentinel-2 ikke rekker til.

1.2 Problemstilling

Målet med denne oppgaven er å se om endringsanalyse med Sentinel-1 SAR-målinger har en nytteverdi. Kan SAR-data supplerer og eller erstatte endringsanalyse av hogstfelt på vinteren når, Sentinel-2 multispektrale bilder ikke kan brukes. Det kan være viktig for oppdragsgivere å vite en tilnærmet sanntids oppdatering på hogstflater, og der kan SAR-bilder komme inn.

En objekt basert klassifisering av polygoner med hogst- og skogflater skal testes i lag med en pikselbasert klassifisering på SAR-bilder. Alle datasettene testet ut i oppgaven bruker SAR-målinger med vertikal/vertikal (VV)- og/eller vertikal/horisontal (VH) -polarisering på et C-bånd med midtels penetrering igjennom vegetasjon og bakke. Få studier tester ut kartlegging med SAR-data på snødekke, og nordiske forhold. Derfor er det ønskelig å svare på følgende problemstillinger:

- I hvilken grad klarer en med maskinlæring å skille hogst- og skogfelt med SAR-bilder generelt, og på vintersesong.
- Hvor godt kan en pikselbasert bildeklassifisering utføres på SAR-bilder.

1.3 Aktualitet og tidligere arbeid

SAR-satellitter ble tatt i bruk med SeaSat-programmet i 1978 [sea]. Tidligere har SAR-data vært privat, statlig eller tilgjengelig mot betaling. Omfattende forskning er utført på bruk av SAR-data til klassifisering og overvåking av jordoverflaten siden dens introduksjon. I motsetning til andre SAR-tjenester er Sentinel-1 gratis med åpen data og kilde, som gjør det tilgjengelig for alle å både laste ned og ta i bruk siden første satellitt ble skutt opp i 2014 [ESA, e]. Fordi alle målinger utført med Sentinel satellittene i Copernicus programmet er gratis, har bilder fra Sentinel-1 SAR i større grad aktuelt å teste ut til generelle formål både for forskning og kommersielt bruk. Med gratis programvarer som gjør behandling av SAR-data enkelt og lett tilgjengelig har de mulige bruksområdene til SAR-bilder begynt å se dagens lys med mye ny forskning og teknikker.

Forskningsartikkelen publisert av Vaglio Laurin et al. [2021] sammenligner klassifisering på Sentinel-1 og Sentinel-2 data for å detektere endringer i skog etter en kraftig storm i Nord Italia. Vaglio Laurin et al. [2021] tester om man klarer å skille polygoner med frisk eller ødelagt skog. Nærmeste nabo og tilfeldig treutvalg er begge brukt i artikkelen, på 4 forskjellige datasett. Selve maskinlæringen brukt i forsøket er utført i R, et programmeringsspråk hovedsakelig brukt til statistikk. Sentinel-1 data preprocessingen er lik som i denne oppgaven, med unntak av koregistrering, disse prosessene beskrives videre i Kapittel 3.

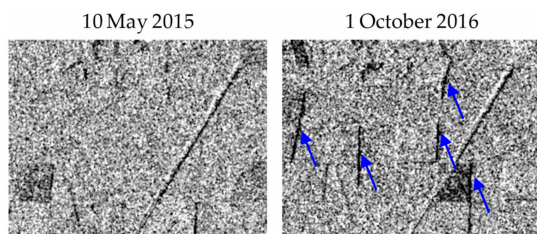
S2_Set1: Sentinel-2 bilder etter storm.

S2_Set2: Sentinel-2 bilder etter storm med vegetasjons indeks.

S1_Set3: Sentinel-1 SAR etter storm med VV- og VH- polarisering.

S1_Set4: Sentinel-1 SAR differanse mellom før og etter storm, med VV- og VH- polarisering.

Tabell 1.1 viser et utsnitt av resultatene presentert i forskningartikkelen med nøyaktighet vist under "OA" (over all accuracy). Nøyaktigheten på Sentinel-2 klassifisering er høyere enn Sentinel-1 med rundt 20 til 30 prosentpoeng.



Figur 1.2: Skygger i SAR måling. [Bouvet et al., 2018]

viste global nøyaktighet 94,36% med MLC og 95,91% med ALT, og stabiliseringsfilter ga ikke bedre resultater.

	Predictor set	OA
BGLM	S2_Set1	0.77
	S2_Set2	0.82
	S1_Set3	0.50
	S1_Set4	0.55
KNN	S2_Set1	0.86
	S2_Set2	0.82
	S1_Set3	0.50
	S1_Set4	0.64
RF	S2_Set1	0.82
	S2_Set2	0.86
	S1_Set3	0.64
	S1_Set4	0.68

Figur 1.1: Resultat tidligere arbeid. [Vaglio Laurin et al., 2021]

Artikkel skrevet av Doblas et al. [2020], om avskoging i Amazonas detektert med Sentinel-1 brukte terskelklassifisering. Algoritmene brukt i denne artikkelen er maximum likelihood classification (MLC) og Adaptive Linear Thresholding (ALT). Trenings/Testdatasettet i denne oppgaven besto av nærmere 6000 lokasjoner over hele Brazil. Halvparten av testområdene var plassert over uberørt skog, andre halvdel var plassert over områder som ble felt i 2019. En av målene med forskningen var også å teste ut stabiliseringsmetoder på Sentinel-1 SAR-dataen. Sluttresultatene presentert i artikkelen

En veldig nytenkende metode brukt i en artikkel skrevet av Bouvet et al. [2018]. I artikkelen betraktes skyggeeffekten forutsagt av innfallsvinkel og bevegelsesretning på SAR-instrumentet, og høydeforskjellen mellom objekter på måleflaten. Forsøksområdet er over et 600 000 hektar stort område av regnskogen i Peru. Totalt 91 målinger, 43 i nordgående, 49 sørgående retning.

Vist i figur 1.2 finner de først endring av nye skygger som kastes rundt omrisset til hogstflatene, her markert med blå piler. Skyggene vil havne på hver sin side (øst og vest) av hogstfeltene i nord- og sørgående retning. Etter deteksjon rekonstruerer hogstflater der en tydelig endring har oppstått, for å beregne areal. Resultatene viser en total deteksjonsrate på 95%, bedre enn kjente teknikker som bruker optiske sensorer til skogovervåking til samme formål.

1.4 Oppsett og struktur

Oppsettet og strukturen til oppgaven er som følger:

Kapittel 1: Innledning Forklarer bakgrunn for oppgaven, problemstilling og tidligere forskning på feltet.

Kapittel 2: Teori Beskriver teorien bak multispektrale bilder, radar, SAR og maskinlæring.

Kapittel 3: Material og Metode Metodedelen beskriver programmer, studieområde, og grunnlagsdata brukt. Metoden forklare fremgangsmåten brukt i preprosessering av SAR-data, og selve klassifiseringen.

Kapittel 4: Resultat Resultatene fra metoden legges frem med plot og sammenligninger av både objektbasert klassifisering, og pikselbasert bildeklassifisering.

Kapittel 5: Diskusjon Diskuterer resultatene, og mulige svakheter i metoden.

Kapittel 6: Konklusjon Oppsummerer oppgaven, resultatene og hva de forteller. Til slutt presenteres konkrete forslag til videre arbeid.

2. Teori

I teorien beskrivelse satellittene, måleinstrumentene og maskinlæringene bruke i oppgaven.

2.1 Hogst i Norge

I Norge leverer skognæringen tremasser til blant annet fyringsved, tømmer, produksjon av papir og eksport. Tall fra SSB [b] forteller at Norge består av totalt 37,4 % skog, som blir til sammen 121043km^2 . Totalt er det 82,8 millioner dekar med drivverdig skog, med en samlet produksjon på rundt 13 millioner kubikkmeter med hogst i 2019.

2.2 Sentinel-1

Sentinel-1 er et europeisk synthetic aperture radar(SAR) oppdrag, som er en del av Copernicus programmet til europeiske romfarts organisasjonen (ESA). Sentinel-1 konstellasjonen består av to satellitter, Sentinel-1A og Sentinel-1B [ESA, b]. Satellittene går i samme bane med 180° faseforskyvning. Hver av satellittene oppnår full dekking av jorden på 12 dager alene, men bruker 6 dager til sammen noe som gir en høy temporal oppløsning [ESA, h]. Begge bruker SAR teknikken for å hente inn data, dette skal videre utdypes under delkapittelet om SAR. Forbruksmålene til Sentinel-1 er som følger:

- Overvåke havis og polare områder
- Kartlegging for humanitær hjelp og kriser
- Overvåkning av marine områder
- Kontroll av landbevegelser
- Kartlegging av land overfalter som skog, vann, jord og jordbruk

[Fletcher, 2012]

2.3 Sentinel-2

Copernicus Sentinel-2 oppdraget består av et samarbeid av to satellitter i nærpolar bane. Begge går i en solsynkron bane med en 180° forskyvning fra hverandre. Hver satellitt bruker 10 dager på å dekke hele jordoverflaten alene, og 5 dager til sammen ved et skyfritt dekke [ESA, e]. Sentinel-2 satellittene bærer et høyoppløselig, vidstrakt(wide-swath) multi-spektral kamera med 13 bånd hver. Båndene har en oppløsning fra 10-60 meter med et fotavtrykk på 290km i bredde [ESA, f]. Bruksområdet til Sentinel-2 omfatter blant annet arealplanlegging, skog- og vegetasjonskartlegging, vannovervåkning og global landbrukskartlegging [ESA, g].

	S2A		S2B		
Band Number	Central wavelength (nm)	Bandwidth (nm)	Central wavelength (nm)	Bandwidth (nm)	Spatial resolution (m)
1	442.7	21	442.3	21	60
2	492.4	66	492.1	66	10
3	559.8	36	559.0	36	10
4	664.6	31	665.0	31	10
5	704.1	15	703.8	16	20
6	740.5	15	739.1	15	20
7	782.8	20	779.7	20	20
8	832.8	106	833.0	106	10
8a	864.7	21	864.0	22	20
9	945.1	20	943.2	21	60
10	1373.5	31	1376.9	30	60
11	1613.7	91	1610.4	94	20
12	2202.4	175	2185.7	185	20

Tabell 2.1: Tabell av båndene i Sentinel-2 A og B MSI sensor.
[ESA, d]

2.3.1 MultiSpectral Instrument

Multi-spektralt kamera/instrument (MSI) er en passiv sensor som bruker push-broom teknikken ved å samle rader av bilder langs banen til satellitten [ESA, d]. MSI måler jordens reflekterte stråling og lys på 13 forskjellige romlige bånd fra synlig nær-infrarødt (VNIR) til kortbølge infrarødt (SWIR) [ESA, d]. Tabell 2.1 viser de forskjellige båndene med bølgelengde og bakkeoppløsning. Det er veldig små forskjeller på Sentinel-2 A og B som utgjør liten til ingen forskjell.

2.4 Radar

Radar er en aktiv sensor, dette betyr at den sender ut sitt eget signal, og er uavhengig av eksterne energikilder som sollys. Det betyr i praksis at radar kan måle små objekter på lang avstand uansett vær, skydekke og lysforhold Skolnik [1962].

Radar er en elektromagnetisk sensor for detektering og lokalisering av reflekterende objekter [Skolnik [2008] kapittel 1.1]. Strålingen reflekteres av objekter på bakken eller i luften beskrevet som et måleobjekt en gitt distanse fra radaren. De elektromagnetiske bølgene blir reflektert i mange forskjellige retninger avhengig av objektets størrelse og form. Noe av strålingen reflekteres tilbake til radarantennen kjent som ekko. Ekkosignalet blir så prosessert og brukt til å beregne posisjonen

til måleobjektet Skolnik [2008].

2.5 SAR

2.5.1 SAR prinsippet

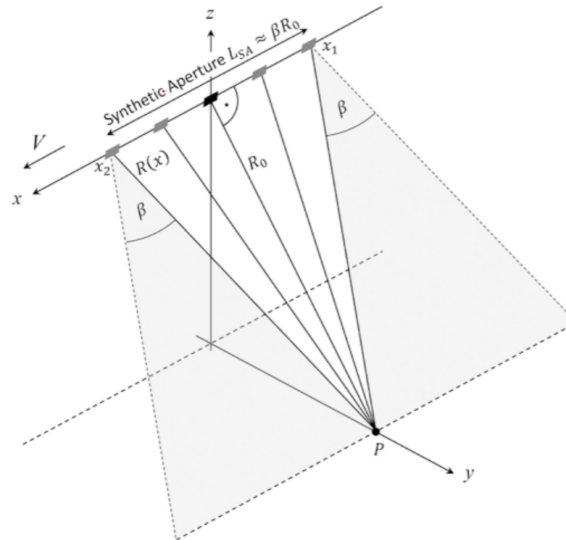
Synthetic aperture radar (SAR) benytter radarteknologi for innhenting av posisjonsdata, men med en syntetisk antenne størrelse. I realiteten er avsender- og mottakerantennen til satellittene for liten til å dekke store landområder fortløpende. Forklart i Flores et al. [2019], for å løse problemet har Carl Wiley utviklet metoden som tillater å lage en lengre syntetisk antenne. Den syntetiske antennen lengden lages ved å slå sammen en rekke målinger utført langs linjen mottakerantennen beveger seg.

Antennelengde påvirker direkte oppløsningen til radar systemet, vist med formel 2.1. Oppløsningen på SAR-målingene fra satellitt, er mellom 5 – 20 meter avhengig av målemetoden [Flores et al., 2019].

$$S \approx \frac{\lambda}{L} R = \beta \cdot R[m] \quad (2.1)$$

- S : Størrelsen på fotavtrykket i lengde eller bredde.
- λ : Bølgelengde
- L : Sidelengde av antennen
- β : Definerer av strålebredde λ/L .
- R : Distanse fra antenne til bakke.

Flores et al. [2019]



Figur 2.1: SAR prinsippet.
[Flores et al., 2019]

Figur 2.1 illustrerer hvordan flere målinger av bakkepunkt P langs satellittens bane måles inn. Antennen beveger seg fra posisjon x₁ (første observasjon av P) til x₂ (siste observasjon av P). Når målingene er utført starter en etter-prosessering for å kombinere alle målinger mellom x₁ og x₂

til ett og samme datasett, dette danner en syntetisk antennelengde. Lengden av den syntetiske antennen kan regnes ut ved formelen 2.2 [Flores et al., 2019].

$$L_{SA} = \frac{\lambda}{L} \cdot R_0 \approx \beta \cdot R_0 \quad (2.2)$$

- λ : Bølgelengde
- L : Sidelengde av antennen
- β : Definerer av strålebredde λ/L .
- R_0 : Distanse fra antenne til bakke.

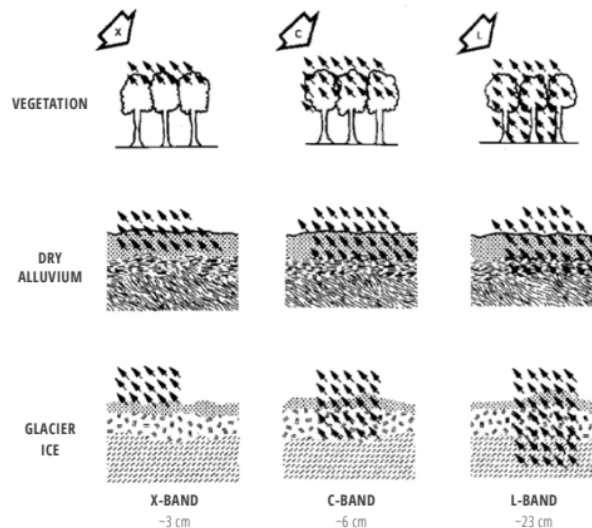
[Flores et al., 2019]

2.5.2 Bånd og bølgelengder

SAR-instrumenter kan bruke forskjellige bølgelengder med varierende egenskaper vist i figur 2.2. C-båndet, brukt i Sentinel-1 har egenskapen til å penetrere overflaten til skogdekket, men ikke ned til bakkenivå i områder med høy bonitet. Penetreringen til C-båndet gjør den egnet til å se endring i skogdekket som hogstfelt hvilken denne oppgaven tar for seg. Bånd som brukes mye av andre SAR satellitter er X- og L- båndet vist i figur 2.2 [Flores et al., 2019]. Oversikt over de forskjellige radarbåndene og bruksområdene vises i tabell 2.2.

Bånd	Frekvens	Bølgelengder	Bruksområder
Ka	27 - 40 GHz	1.1 - 0.8 cm	Flyradar
K	18 - 27 GHz	1.7 - 1.1 cm	Atmosfærisk H ₂ O absorpsjon.
Ku	12 - 18 GHz	2.4 - 1.7 cm	Satellitt altimetri
X	8 - 12 GHz	3.8 - 2.4 cm	Høy oppløselig SAR: Urban overvåkning, is, snø, lite vegetabilsk penetrering; Hurtig kartlegging av store endringer.
C	4 - 8 GHz	7.5 - 3.8 cm	Hoved båndet benyttet i SAR. Global kartlegging; ; overvåkning av områder med lav til moderat vegetasjon.
S	2 - 4 GHz	15 - 7.5 cm	Økt bruk i SAR basert jord observasjon; landbruks overvåkning.
L	1 - 2 GHz	30 - 15 cm	Høy penetrering og vegetasjons kartlegging, medium oppløsning.
P	0.3 - 1 GHz	100 - 30 cm	Nytt eksperimentelt bånd. Biomasser; vegetasjons kartlegging og analyse.

Tabell 2.2: Oversikt over forskjellige radar bånd.
[Flores et al., 2019]



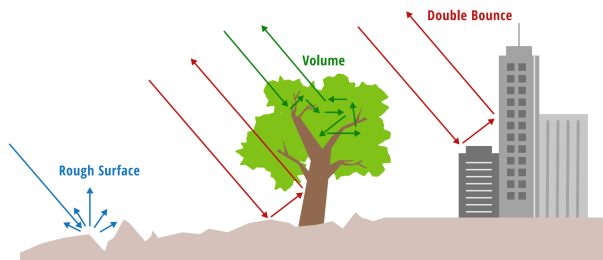
Figur 2.2: SAR-penetrering.
[Flores et al., 2019]

2.5.3 Polarisering

SAR-instrumentet sender ut bølger i enten horisontal (H) eller vertikale (V) polarisering. Polarisering beskriver orienteringen en bølges svingning har i planet [Flores et al., 2019].

Mange gamle SAR instrumenter er enveis polariserte, altså de mottar og sender samme polarisering,

horisontal/horisontal(HH) eller vertikal/vertikal(VV). Majoriteten av nye SAR sensorer sender og mottar horisontale og/eller vertikale polariseringer [Flores et al., 2019]. Sentinel-1 instrumentene er fire-polarisert med egenskapen til å sende og motta både vertikale og horisontale bølger VV, VH, HH og HV [ESA, c] beskrevet som polarisering og krysspolarisering(VH og HV) senere i oppgaven.



Figur 2.3: SAR refleksjonstyper.
[Flores et al., 2019]

Relativ spredning i polarisering:

Spredning ujevne overflater	$ S_{VV} > S_{HH} > S_{HV} $ eller $ S_{VH} $
Dobbel reflektert spredning	$ S_{HH} > S_{VV} > S_{HV} $ eller $ S_{VH} $
Voluminøs spredning	$ S_{HV} $ eller $ S_{VH} $

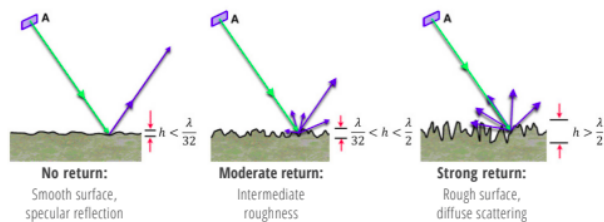
Tabell 2.3: SAR refleksjonstyper.
Flores et al. [2019]

Vist med tabell 2.3 og figur 2.3 er valg av polarisering viktig. Polariseringer virker forskjellig ut i fra hvilke type objekter signalene treffer Flores et al. [2019]. I oppgaven brukes VV og VH -polarisering. VV-polarisering relaterer til ujevne overflater som vann og terreng. HV/VH-krysspolarisering relaterer til mengde spredning i mjuke overflater som vegetasjon, jord og sand.

2.5.4 SAR bilde

SAR måler returen på radarbølgene som sendes ut, og visualiseres med gråskala bilde basert på returstyrken til hvert punkt under den syntetiske blenderåpningen. I denne oppgaven tilsvarer mørke overflate lavt til ingen retursignal, og lysere farger tilsvarer sterkere/mange retursignal.

Vist i figur 2.4 påvirker strukturen til målte overflater mengden retursignal, en glatt overfalte som vann og kort gress reflekterer lite til ingen signal tilbake til satellitten. Ruglete overflater med litt variasjon som ulent terreng, enkelt bygninger og klynger med trær vil gi noen retursignaler. Ujevne overflater med høy bonitet som skog, fjellskrenter og byområder vil gi sterke retursignal [Flores et al., 2019].



Figur 2.4: SAR refleksjon.
[Flores et al., 2019]

2.6 Maskinl ring

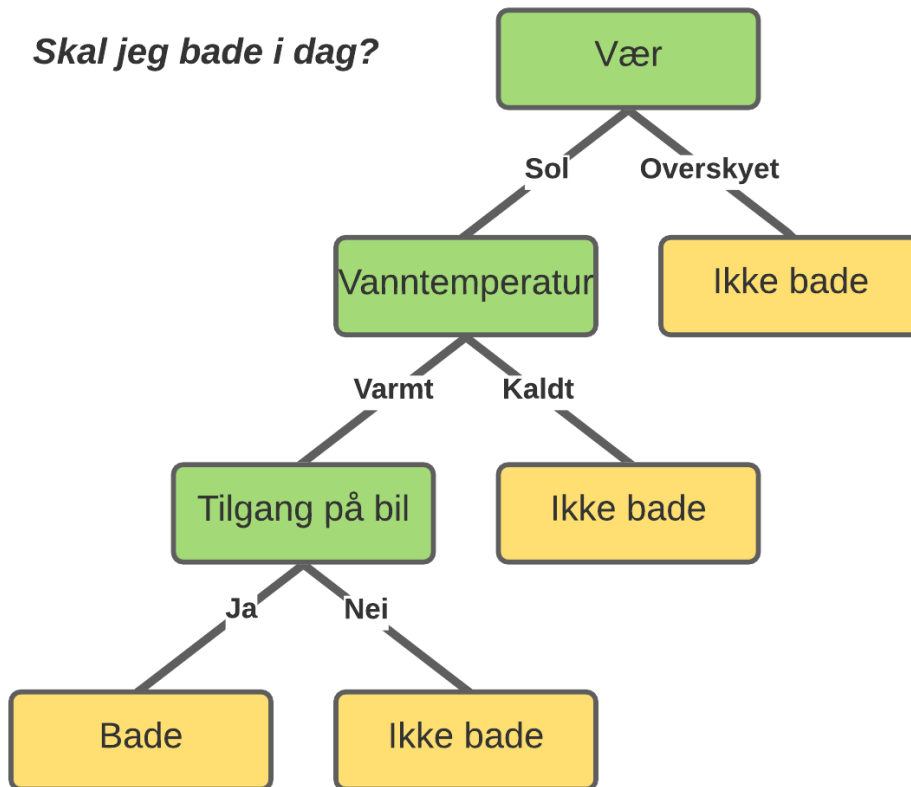
Maskinl ring (ML) kan defineres som en kalkulerende metode, som bruker erfaring til    ke effektivitet, eller   utf re n ye prediksjoner. Erfaring refererer til tidligere informasjon tilgjengelig for algoritmen, som typisk kommer i en form av elektronisk data samlet og gjort tilgjengelig for analyse. Dataen kan v re i form av digitalisert menneskelig markerte treningsett, eller andre typer informasjon hentet inn ved interaksjon med tilgjengelig data. I alle tilfeller er kvalitet og st rrelse essensielt for n yaktigheten av prediksjonene til algoritmen. N yaktigheten p  prediksjonene er avhengig av kvaliteten p  den markerte trenings dataen [Mohri et al., 2018].

I metoden benyttes tre forskjellige ML, Tilfeldig treutvalg, N rmeste nabo og St ttevektormaskin, forklart videre i delkapittelet.

Tilfeldig treutvalg

Tilfeldig treutvalg kjent som Random Forest Classifier (RFC) er en videreutviklet valgte algoritme. Raschka and Mirjalili [2019] forklarer valgte som en m te   bryter ned data til forskjellige klasser ved   stille en rekke sp rsm l eller kriterier som vist i figur 2.5. Figuren viser kategorisk inndeling med sp rsm l. Den fungerer p  samme m te med tallverdier, hvor sp rsm l ang ende st rrelse eller spesifikke verdier blir stilt for   skille forskjellig data.

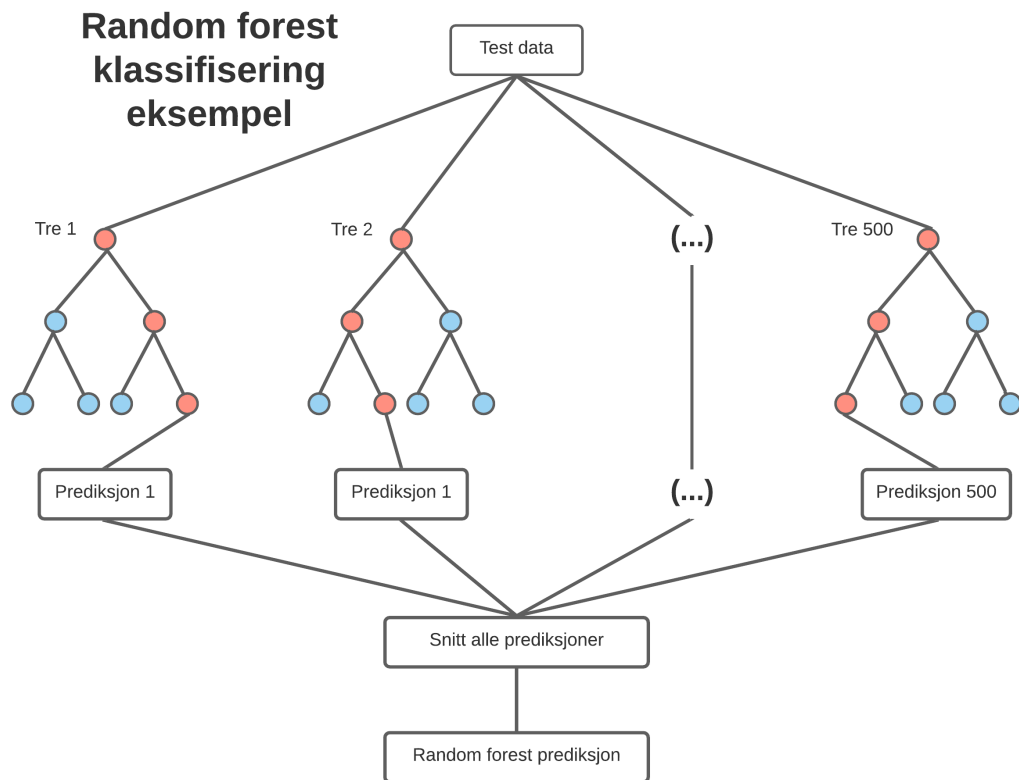
Eksempel valgtre



Figur 2.5: Valgtre eksempel.

Algoritmen har parametere som kan justeres for å definere hvor dypt et tre kan være, og antall noder det skal inneholde. Med for mange iterasjoner og inndelinger kan valgtreet bli overtilpasset, og predikering av ny data vil slå ut feil [Raschka and Mirjalili, 2019].

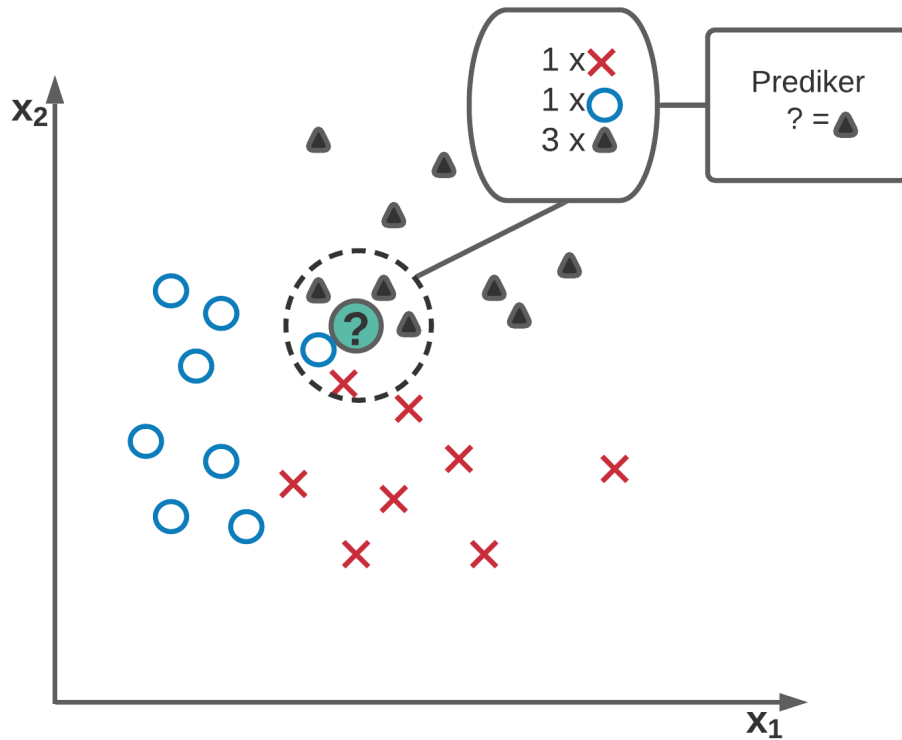
RFC kan betraktes som en sammenslåing av valgtreer vist i figur: 2.6. Ideen bak RFC er å ta gjennomsnittet av mange dype valgtreer som individuelt har høy varians, dette for å bygge en mer robust modell som har bedre generalisert treffsikkerhet og er mindre utsatt for overtilpassing [Raschka and Mirjalili, 2019].



Figur 2.6: Tilfeldig treutvalg.

Nærmeste nabo

Nærmeste nabo kjent som K-nearest neighbor (KNN) er en “lazy learner”, dette fordi den ikke lærer en diskriminerende funksjon, men memorerer treningsdataen. Figur 2.7 viser enkelt hvordan algoritmen jobber. En ukjent variabel blir klassifisert basert på de x-nærmeste verdiene hvor majoriteten bestemmer klassen. Hvor stor distanse fra et ukjent punkt, og antall naboer algoritmen skal ta i betraktning kan justeres for å oppnå best mulig resultat. [Raschka and Mirjalili, 2019]



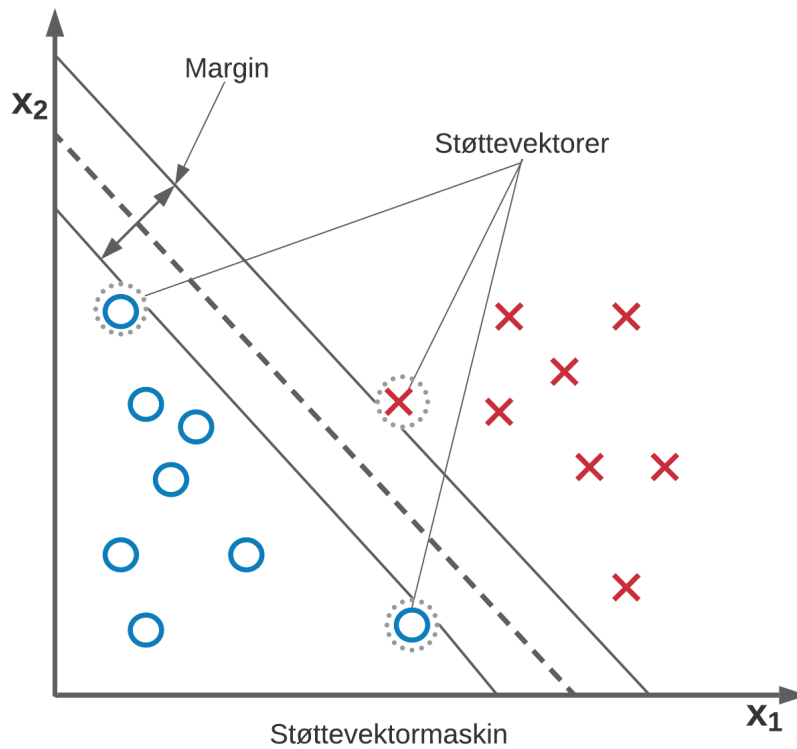
Figur 2.7: Nærmeste nabo.

Støttevektormaskin

Støttevektormaskin kjent som Support vector machine (SVM). SVM forsøker å skille klasser ved å finne ytterpunktene til hver klasse, og danne en skillelinje med margin. Målet til SVM er at alle variabler som er i “negative” klasser plasseres under skillelinjen, og alle “positive” klasser plasseres over skillelinjen, eksempel på dette vises i figur 2.8.

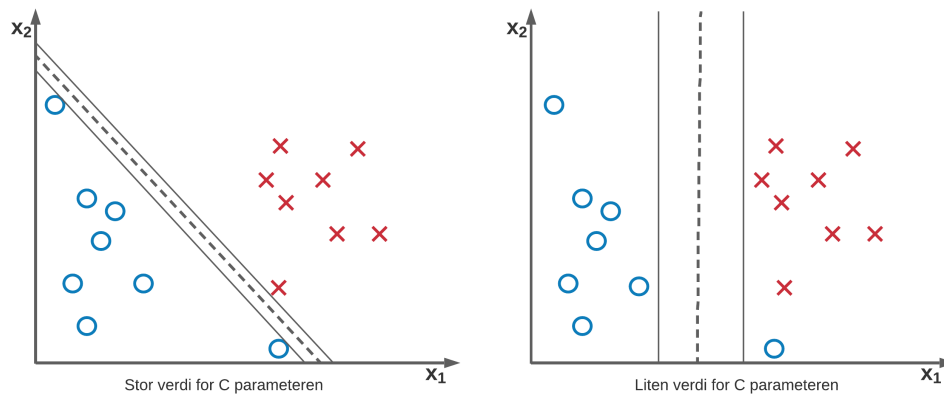
En viktig parameter å betrakte i SVM er C -variabelen som bestemmer god tatt feilmargin. I figur 2.9 vises at stor verdi for C vil være mye strengere enn en lav verdi for C , som i større grad tillater feil [Raschka and Mirjalili, 2019].

Om maskinlæringen er for streng med en høy C -verdi kan modellen bli overtilpasset, som resulterer i feilklassifisering straks det er små variasjoner i hver av klassene. C -verdien må tilpasses hvert enkelt datasett.



Figur 2.8: Støttevektor maskin.

KNN liten kontra stor C verdi

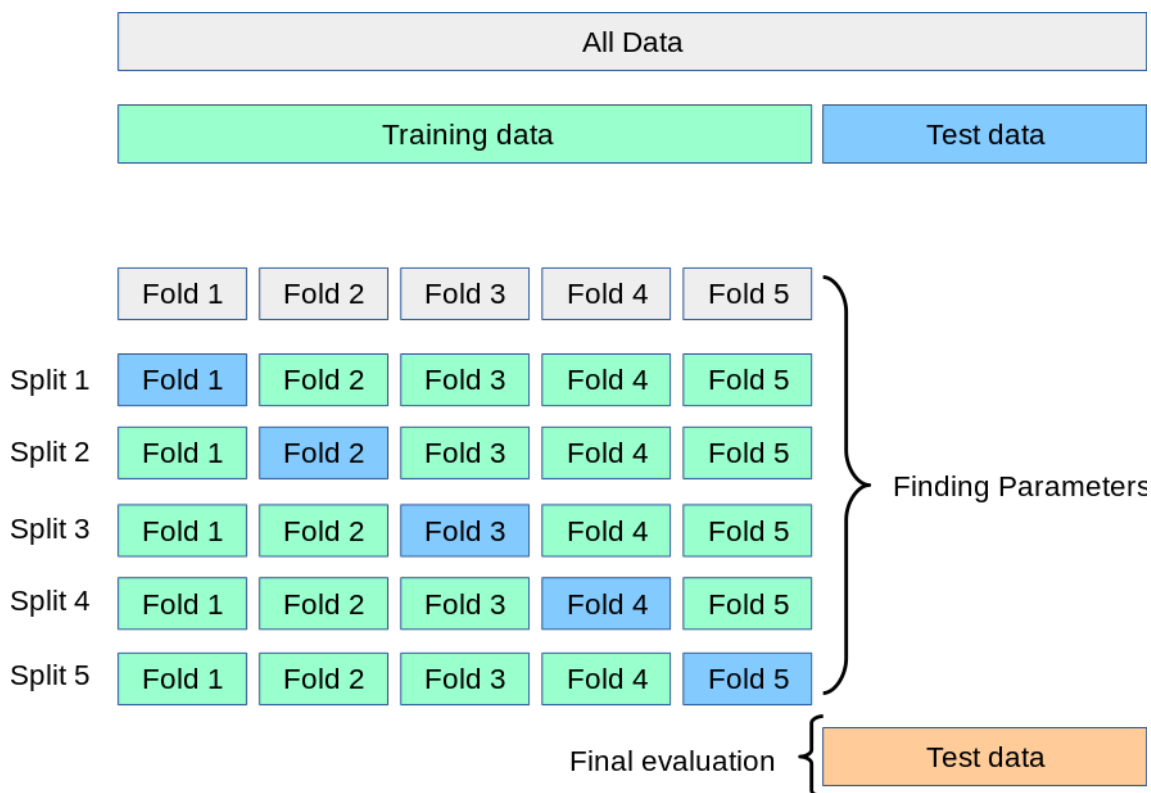


Figur 2.9: Støttevektor maskin, C variabel.

2.6.1 Kryssvalidering

Kryssvalidering er en teknikk tatt i bruk for å unngå overtilpasning til treningsdatasettet [Pedregosa et al., 2011]. K-fold brukt i oppgaven og vist i figur 2.10 benytter “leave one out” strategien. K-fold deler inn treningsdata k -antall splitter. Hver splitt deles inn i $k - 1$ deler, hvor en del holdes tilbake for å brukes til testing. Eksempel i oppgaven deles treningsdata inn i $k = 5$, som vil si at hver splitt deles inn i 80% treningsdata, og 20% testdata. For hver splitt tas en ny del av treningsdata som

testdata, det vil si at all treningsdata bli brukt som testdata en gang hver når kryssvalideringen er ferdig [Pedregosa et al., 2011].



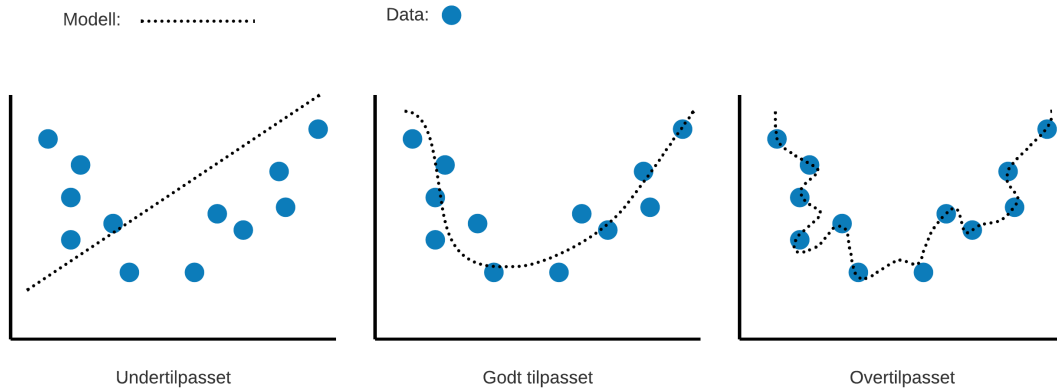
Figur 2.10: Kryssvalidering.
[Pedregosa et al., 2011]

2.6.2 GridSearchCV for tilpasning av modell

GridSearchCV er en modul i Python brukt for å teste forskjellige kombinasjoner av parametere i maskinlæringsalgoritmene. Eksempel er å teste ut forskjellige verdier for C, gamma og kernel i SVM. De kombinasjonene av forhåndsvalgte testparametere som gir de beste resultatene brukes videre for å predikere test datasettet. For å unngå over- eller undertilpassing brukes kryssvalidering av datasettet sammen med hver parameter test i GridSearchCV [Pedregosa et al., 2011].

2.6.3 Over- og undertilpassing

Figur 2.11 viser eksempel på tre situasjoner som kan oppstå ved klassifisering. Første til venstre viser eksempel på en modell som ikke klarer å predikere mønster eller klasser. Dette er som regel et resultat av manglende eller dårlig treningsdata. Midterste figur viser en godt tilpasset modell med treningsdata som er godt fordelt og gir maskinlæringen nødvendig datagrunnlag. Siste figur til høyre viser en overtilpasset modell, dette er et resultat av modellen som tilpasser seg støy og eventuelle feil i datasettet. Overtilpassing skjer når modellen kjører for mange iterasjoner, og parametere er for finjustert til datasettet. En overtilpasset modell vil ikke kunne predikere testdata som har variasjoner fra treningsdata [Pedregosa et al., 2011].



Figur 2.11: Over- og undertilpasning.

2.6.4 Forvirringsmatrise

Maskinlæringsalgoritmenes klassifiseringsresultat visualiserer med forvirringsmatriser (confusion matrix). Forvirringsmatrise viser hvor godt algoritmen klassifiserer hver klasse, og hva de feilklassifiserer. I denne oppgaven benyttes 2 klasser, skog og hogst. På bilde 2.12 ser vi et eksempel på hvordan en forvirringsmatrise er satt opp.

Som forklart av Powers [2008] grønn farge indikerer riktig klassifisering. True positive (TP) viser hogst klassifisert riktig, False positive (FP) viser hogst klassifisert som skog, altså feil klassifisering. False negative (FN) og True negative (TN) er det samme bare motsatte hvor TN er riktig klassifisert, og FN er feil for klassen skog. I oppgavens tilfelle er actual negative det samme som hogst, og actual positive er skog.

		Predikerte klasser	
		Hogst	Skog
Sanne klasser	Hogst	TN	FP
	Skog	FN	TP

TN = Hogst klassifisert som hogst	FP = Hogst klassifisert som skog
FN = Skog klassifisert som hogst	TP = Skog klassifisert som skog

Figur 2.12: Forvirringsmatrise.

2.6.5 Klassifiseringrapporter

Etter hver av klassifiseringene er kjørt returnerer de forvirringsmatrise og en tabell med fire verdier precision (presisjon), recall (systematisk skjevhet/sensitivitet), f1-score (f1-tall) og support (støtte). Presisjon, sensitivitet og f1-tall er mål på hvor godt en maskinlæring klarer å klassifisere klasser. Alle parameterne regnes ut i fra verdiene returnert i forvirringsmatrisen. Formlene 2.3, 2.4 og 2.5 tar utgangspunkt i de eksempel verdiene vist i figur 2.12. Support er antall elementer i hver klasse, og brukes for å vekte ut snittet til recall, f1-score og precision.

Presisjon

Presisjon kvantifiserer mengden positive klasse prediksjoner som faktisk tilhører den positive klassen vist i formel 2.3 [Powers, 2008]. Om maskinlæringen blir optimalisert for presisjon i hogst klasse vil det føre til mindre feil klassifisering av skog klassen, men på bekostning av hogstklassen selv, altså en høy verdi i FP Raschka and Mirjalili [2019].

$$Presisjon = \left(\frac{TP}{TP + FP}\right)_1 \text{og} \left(\frac{TN}{TN + FN}\right)_2 \quad (2.3)$$

- 1: Presisjon for skog.
- 2: Presisjon for hogst.

Powers [2008] side 2

Sensitivitet

Sensitivitet(Recall) kvantifiserer mengden positive prediksjoner sett mot alle riktige prediksjoner utført av modellen vist i formel 2.4.

Om maskinlæringen blir optimalisert for sensitivitet i hogstklassen vil det resultere i en partiskhet som minimaliserer sjansen for å ikke klassifisere hogst, men kan ende opp med å klassifisere som hogst når det egentlig er skog, altså en høy verdi i FN [Raschka and Mirjalili, 2019].

$$Sensitivitet = \left(\frac{TP}{TP + FN}\right)_1 \text{og} \left(\frac{TN}{TN + FP}\right)_2 \quad (2.4)$$

- 1: Sensitivitet for skog.
- 2: Sensitivitet for hogst.

Powers [2008]

F1-tall

For å balansere ut de positive og negative sidene av sensitivitet og presisjon benyttes F1-tallet for å optimalisere verdiene [[Raschka and Mirjalili, 2019] side 214]. Formel 2.5 viser utregningen til f1-tallet.

$$F1 = 2 \frac{PRE \times REC}{PRE + REC} \quad (2.5)$$

- *PRE*: Presisjon.

-
- *REC*: Sensitivitet

Raschka and Mirjalili [2019]

3. Material og Metode

Dette kapittelet er delt inn i to deler, hvor materialer brukt i oppgaven kommer først, etterfulgt av metoden brukt for å komme fram til resultatene.

3.1 Materialer

Materialer vil ta for seg hva som danner datagrunnlaget brukt i oppgaven, programvarer og moduler brukt vil også bli forklart.

3.1.1 Filformater

Flere filformater er brukt i oppgaven, en kortfattet forklaring av de kommer under.

CSV

“Comma-separated values” (CSV), som på norsk heter kommaseparert fil. CSV-fil er et simpelt tekstformalt og brukes oftest når store mengder av data skal flyttes fra en database til en annen [Vis]. Hver verdi deles inn med komma, og leses lett av i tabeller med programvarer som Excel.

Tiff

TIFF er et filformat brukt for å lage og forklare rasterbilder. TIFF er bygd opp med et hode hvor metadata ligger, etterfulgt av kroppen til filen hvor data med bildefilen ligger [Mahammad and Ramakrishnan, 2003].

geoTiff

GeoTiff er beskrevet som en utvidelse av metadataen til TIFF filene gitt for å beskrive kartografisk informasjon assosiert med TIFF-bilder [Mahammad and Ramakrishnan, 2003]. GeoTIFF bildene er georeferert, og kan legges oppå kart med riktig posisjon.

ESRI: Shapefil

En ESRI shapefile lagrer ikke topologisk geometri og attributt informasjon for romlige objekter. Geometrien til objekter er lagret i en sammensetting av vektorkoordinater. Shapefile støtter punkter, linjer og romlige mønstre som polygoner [ESRI, 1998].

3.1.2 Forsøksområde

I denne oppgaven betraktes området som dekker Aurskog-Høland. Blom som oppgaven skrives for har valgt ut området da det tidligere ble kartlagt for endringer av hogstfelt. Kartleggingen av hogstfelt ble utført med maskinlæring på Sentinel-2 bilder i perioden april 2019 til juli 2020.

Aurskog-Høland er en skogkommune med store forekomster av hogstfelt. I følge SSB [a] har kommunen med et landareal på totalt $1144,8 \text{ km}^2$ og hele $857,57 \text{ km}^2$ (77,9 %) av landarealet er dekket av skog. Figur 3.1 viser Aurskog-Høland og områdene rundt.



Figur 3.1: Aurskog-Høland.


3.1.3 Sentinel-1 SAR datasettet

Alle ESA Sentinel målinger er åpen data og kildekode som gjøres tilgjengelig for offentligheten. Norge har en egen nasjonal database av all historisk Sentinel data over nasjonen. Nettsiden heter "Copernicus Scientific Data Hub" (ColHub) [Col].

For å hente ut data ble en shape fil av forsøksområdet lastet opp på ColHub. Filter og spesifikasjoner ble som vist i figur 3.3 satt. I filteret settes tidsrom for måling, polarisering, datatype og andre spesifikasjoner.

Det er mulig å velge mellom to hoveddataformat RAW og GRD. GRD står for Ground Range Detection, og er det som ble brukt i oppgaven. GRD er prosessert SAR-data som er korrigert på en ellipse i en konstant rekkevidde [GRD]. GRD-produktet er delvis ferdig prosessert, men trenger videre etterprosessering som utføres i SNAP, eller lignende programvarer.

RAW-formatet er ubehandlet, og inneholder verken filtrert eller prosessert data. Rådata må igjennom flere prosesser før den kan visualiseres. Den har egne inndelinger med standard-, kalibrering-, støy- og metadata- produkter [ESA, a].



☰ Insert search criteria... 



▼ Upload Shape File For Geographical Selection

▲ Advanced Search Clear

» Sort By: ▼

» Order By: ▼

» Sensing period From:  to: 

» Ingestion period From:  to: 

Mission: Sentinel-1

Satellite Platform ▼

Product Type ▼

Polarisation ▼

Sensor Mode ▼

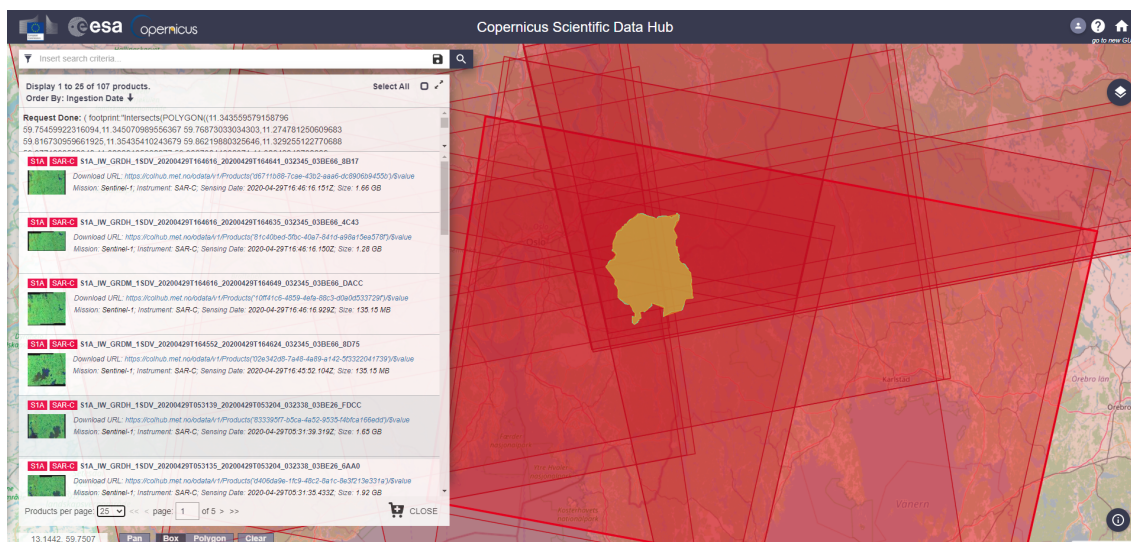
Relative Orbit Number (from 1 to 175)

Collection ▼

Mission: Sentinel-2

Cloud Cover % (e.g.[0 TO 9.4])

Figur 3.2: Filter eksempel til nedlasting.



Figur 3.3: Eksempel utvalgt nedlasting data.

Datasett

Liste over SAR målinger brukt i oppgaven, og hvilke datasett de tilhører. Datasettene er forklart i kapittel 3.4.1

Dato	SAR måling	Datasett
22.05.19	S1A_IW_GRDH_1SDV_20190522T165414_20190522T165439_027343_03157F_9891	1, 2, 3
03.06.19	S1A_IW_GRDH_1SDV_20190603T165414_20190603T165439_027518_031AEC_1D63	1, 2, 3
14.08.19	S1A_IW_GRDH_1SDV_20190814T165419_20190814T165444_028568_033B22_E108	1, 2, 3
01.10.19	S1A_IW_GRDH_1SDV_20191001T165421_20191001T165446_029268_035357_00CD	1, 2, 3
06.12.19	S1B_IW_GRDH_1SDV_20191206T165339_20191206T165358_019247_02456E_8C05	2, 3
23.01.20	S1B_IW_GRDH_1SDV_20200123T165338_20200123T165356_019947_025BB9_A415	2, 3
04.02.20	S1B_IW_GRDH_1SDV_20200204T165337_20200204T165402_020122_026163_3ACD	2, 3
11.03.20	S1B_IW_GRDH_1SDV_20200311T165337_20200311T165355_020647_027235_8EFA	2, 3
04.04.20	S1B_IW_GRDH_1SDV_20200404T165337_20200404T165402_020997_027D45_81FD	1, 2, 3
16.05.20	S1A_IW_GRDH_1SDV_20200516T165420_20200516T165442_032593_03C668_04EC	1, 2, 3
09.06.20	S1A_IW_GRDH_1SDV_20200609T165421_20200609T165441_032943_03D0D8_0B1F	1, 2, 3
09.08.20	S1A_IW_GRDH_1SDV_20200908T053146_20200908T053211_034263_03FB5F_1082	1, 2, 3
01.09.20	S1A_IW_GRDH_1SDV_20200901T165426_20200901T165451_034168_03F80F_9BCA	1, 2, 3

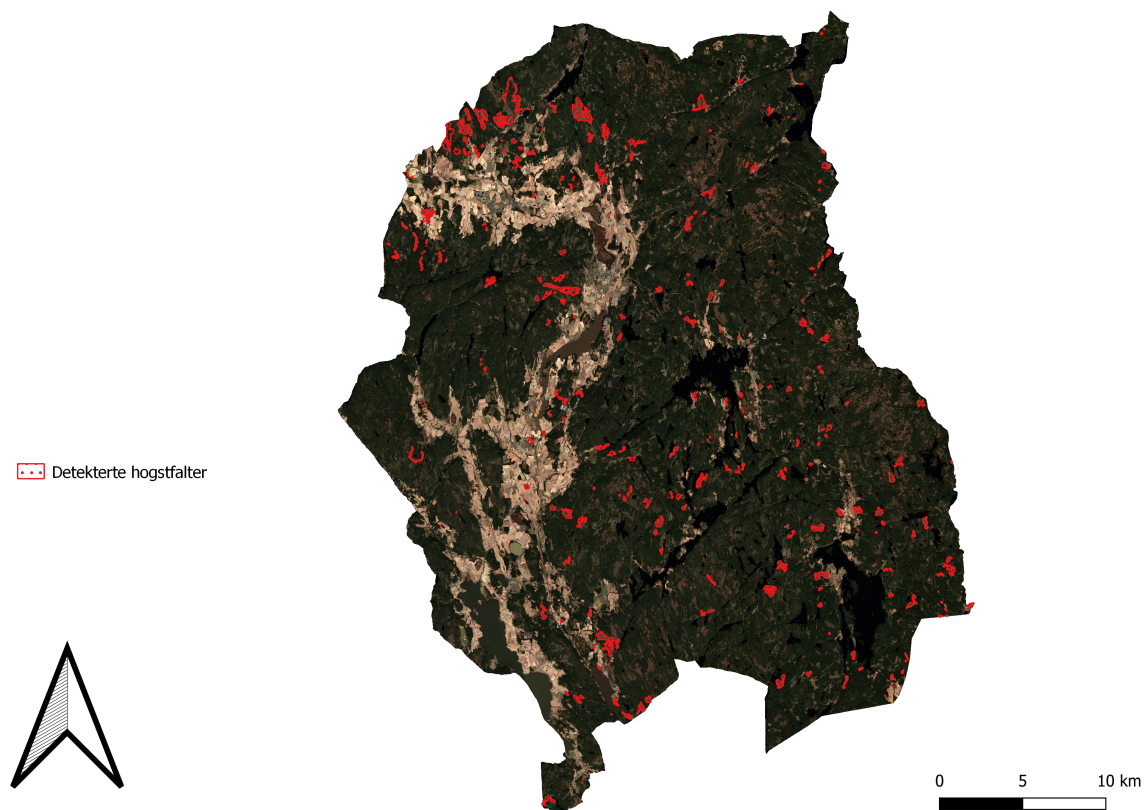
3.1.4 Hogstdatasett

Treningsdata ble hentet fra en ESRI shapefil med polygoner over en rekke hogstfelt klassifisert av Blom. Hver enkelt polygon inneholder informasjon om når hogstfeltene er klassifisert. Et lite utdrag av hogstdatasettet vises i figur 3.4.

Totalt er det utført 356 klassifiseringer mellom 17.06.2019 og 26.06.2020 av Blom med Sentinel-2 bilder. Blom har selv brukt U-Net dyp læring til å utføre klassifiseringen (S. Rostad, personlig kommunikasjon, 28.05.2021). I figur 3.5 er alle hogstfeltene de har klassifisert. I oppgaven ble bare hogst mellom 01.10.2019 og 20.03.2020 tatt i bruk, fordi det er tidsrommet Sentinel-2 ikke kan utføre endringsanalyse på. Fordi klassifisering med Sentinel-2 ikke er helt nøyaktig er de resterende polygonene kontrollert opp mot Sentinel-2 farge (RGB) -bilder før og etter hogst. Polygonene som tilsynelatende er feilklassifisert fjernes. For å gi mest mulig data ble flere u detekterte lagt til manuelt i QGIS, dette ved å se på Sentinel-2 bilde før og etter vintersesongen.

For å teste muligheten å skille hogstfelt fra skogomgivelsene legges egendefinerte skog polygoner

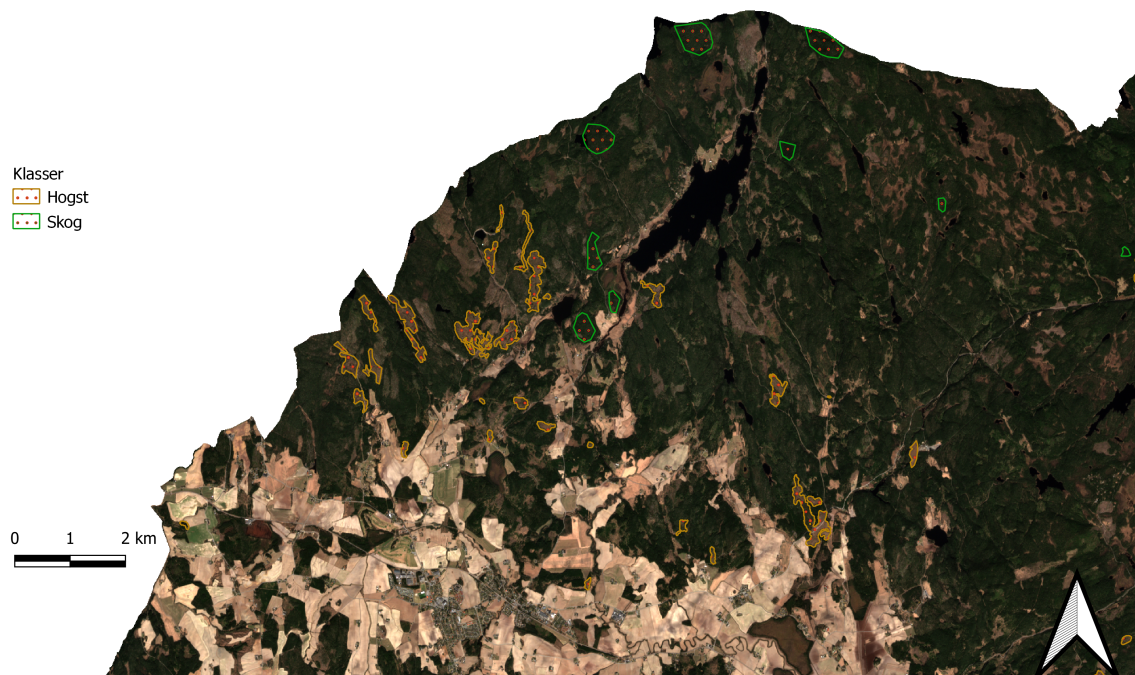
til i datasettet. Hogst og skog tilegnes klassenummerering 1 og 2. Polygoner for skog blir manuelt lagt inn ved å se på et Sentinel-2 bilde som er nyere enn den siste målingen gjort med Sentinel-1, for å unngå at et markert skogområde blir til hogst i løpet av måleperioden. Et lite utsnitt av det endelige datasettet vises i figur 3.6 hvor hogstfelt er markert med oransje omriss, og skog er markert med grønt omriss.



Figur 3.4: Utsnitt av trenings-data.

ID	pre_dato	E_dato	tot_dets	num_dets	Area	check	Hogstfelt	
1	6042	20190727	20190729	51,28378378378...	10,25675675675...	14800,00000000...	0	1
2	4128	20190727	20190729	51,36322869955...	10,77130044843...	44600,00000000...	0	1
3	2282	20191012	20191013	49,41252699784...	8,781857451403...	46300,00000000...	0	1
4	719	20191013	20200409	49,50877192982...	9,684210526315...	5700,000000000...	0	1
5	1991	20191013	20200407	49,66463414634...	9,585365853658...	16400,00000000...	0	1
6	242	20191007	20191013	49,70786516853...	9,955056179775...	17800,00000000...	0	1
7	3066	20191013	20200409	52,84641638225...	10,49829351535...	29300,00000000...	0	1
8	4857	20190922	20191005	53,48582995951...	10,51821862348...	24700,00000000...	0	1

Figur 3.5: Klassifiserte hogstfelt fra Sentinel-2.



Figur 3.6: Markert hogst og skog.

3.2 Programvarer og moduler

3.2.1 ESA SNAP

SNAP står for The Sentinel Application Platform. SNAP er et individuelt verktøy for behandling av Sentinel data som Sentinel -1 og -2.

3.2.2 QGIS

QGIS er et åpent kildekode geografisk informasjonssystem (GIS) program. QGIS støtter en rekke med vektor-, raster-, databaseformater og funksjoner [QGI].

3.2.3 Lucidchart

Lucidchart er et sky-basert program brukt til lage enkle illustrasjoner [luc].

3.2.4 Python

Python er et høy-nivå programmeringsspråk brukt til generelle formål. Python har innebygde datatyper som lister, tekst og matriser. Det eksisterer ett stort bibliotek av funksjoner, klasser, moduler og pakker som hjelper å organisere, og behandle koden Kuhlman [2009].

Python moduler

I Python scriptet benyttes en rekke moduler for å hente inn og behandle data.

Pandas

Pandas tilbyr ett vidt spekter av datastrukturer og funksjoner laget for å gjøre jobb med strukturert data raskt og enkelt. Pandas er en av de kritiske modulene som tillater Python å være et potent og produktivt dataanalytisk verktøy. Hovedoppgaven til Pandas i denne oppgaven er å strukturere data i 2-dimensjonale matriser, rad og kolonne orientert indekser [McKinney, 2012].

Geopandas

GeoPandas er en åpen kildekode-modul for å lettere jobbe med georeferert data i Python. Modulen er en utvidelse av Pandas for å tillate romlige operasjoner på geometriske objekter [geo].

matplotlib

Matplotlib er et omfattende bibliotek brukt for å lage statistikk, animasjoner og interaktive visualiseringer i Python [Hunter, 2007]. I oppgaven brukes matplotlib til visualisering av forvirringsmatriser.

NumPy

NumPy, kort for Numerical Python, er den grunnleggende pakken for vitenskapelige beregninger i Python. NumPy har flere funksjonaliteter. Den kan enkelt og effektivt bruke multidimensjonale lister, tilføre operasjoner mellom lister, lese og skrive lister til intern lagring, regne lineær algebra, transformasjon og generering av tilfeldige verdier [McKinney, 2012]. NumPy brukes til lister og enkle beregninger i oppgaven.

Sklearn

Sklearn også kjent som scikit-learn er en stor samling av mange vel kjente ML-algoritmer, men samtidig opprettholder et enkelt brukergrensesnitt godt implementert i Python språket [Pedregosa et al., 2011].

Os

OS-modulen i Python gir tilgang til funksjoner for å samhandle med operasjonssystemet. OS er en del av Pythons standard pakker, og gir en portabel måte å bruke operasjonssystem avhengige funksjoner som å lese og skrive data fra mapper [OS].

rasterio

Rasterio er et bibliotek for å lese geografiske formater som blant annet GeoTiff, for å organisere og lagre rasterdatasett som satellittbilder og terrengmodeller [ras, a].

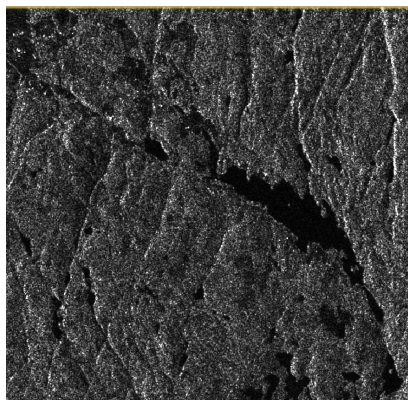
rasterstats

Rasterstats er en Pythonmodul brukt for å summere geografisk informasjon basert på vektorgeometrier. Det inkluderer funksjoner for områdestatistikk og interpolasjon av punktdata [ras, b].

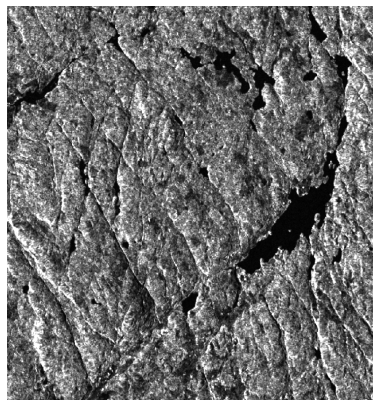
3.3 Prosessering av data i SNAP

Før SAR data kan brukes i Python for klassifisering må det preprosesserer i SNAP. Forskjellen mellom SAR bildet før og etter preprosessering av omtrent samme område kan sees i figur 3.7 og 3.8. Bildet før prosessering inneholder mye støy som gjør det vanskelig å skille objekter i terrenget, det mangler også orientering så bildet blir opp ned og speilvent. Bildet etter prosessering har redusert støy og riktig orientering i koordinatsystemet Euref89 UTM sone 32.

For å få sluttproduktet i figur 3.8 måtte bildet igjennom en rekke med prosesser som beskrives i rekkefølge videre i teksten, med navn på prosesseringen i SNAP. Prosessene er valgt basert på brukerveiledning fra ESA forklart i Braun [2020], og Filipponi [2019]. Formålet er å sitte igjen med et GeoTiff-bilde hvor man visuelt kan skille mellom hogstflater og skogområder.



Figur 3.7: SAR-bilde før prosessering



Figur 3.8: SAR-bilde etter prosessering

Subset

Datasettet beskjæres ned til et rektangulært utvalg som dekker forsøksområdet, dette for å gjøre de resterende prosesseringene enklere med mindre data å behandle.

Applying orbit information

I metadataen til SAR ligger informasjon om satellitt banen, men den er generelt ikke presis. Presis data om satellitt bane lastes ned fra ESA, og er typisk tilgjengelig noen dager etter måling. Den presise satellittbanen lastes ned automatisk, og oppdaterer til den presise posisjonen satellitten hadde i måleøyeblikket [Filipponi, 2019].

Thermal noise removal

SAR-bildenes intensitet er fordelt med termisk støy, spesielt i krysspolariserte bånd som VH. I prosesseringen fjernes dette støyet, og normaliserer retursignalet [Filipponi, 2019].

Ratiometric calibration

Radiometisk kalibrering konverterer digitale pikselverdier til radiometrisk kalibrert SAR-retur. Kalibreringen bruker innfallsvinkel, og andre sensor-spesifikke konstanter [Braun, 2020].

Coregistration

Koregistrering legger flere målinger sammen til en samlet filstabel (file stack). Måledata har små tidsdifferanser og forskjeller i piksel posisjon. Disse små differansene korrigeres til en tilnærmet nøyaktig overlapp [Braun, 2020].

Speckle filtering

SAR-bildene inneholder mye støy og små variasjoner på de reflekterte overflatene, også kjent som “speckle”. Støyet er et resultat av forstyrrelser i bølgene reflektert på overflater, dette kan gi homogene flater stor differanse i retursignal, og kan reduseres ved utvalgte filtermetoder [LEE et al., 1994]. I oppgaven brukes Lee-sigma filet, en “edge sharpening” som fremhever ytterkantene til objekter på bakken [LEE et al., 1994]. Med Lee-sigma blir hogstfelt mye tydeligere, og fremheves fra omgivelsene.

Terrain correction

Terreng korreksjon georefererer SAR-bildene med en digital terrengmodell(DTM) og ortorektifiserer bildet. Den interne geometrien til bildet vil også rettes opp, og orientering bli korrigert [Filipponi, 2019]. Det er mulig å bruke både en automatisk nedlastet DTM, og en egen ekstern DTM, men i denne oppgaven brukes automatisk nedlastede DTM fra Copernicus på 30 meter i oppløsning.

3.4 Metode

I denne delen beskrives metoden brukt for å prosessere data i Python, og hvordan klassifiseringen blir utført for å komme fram til resultatene.

3.4.1 Datagrunnlag

Sentinel-2 og andre multispektrale-bildesatellitter kan ikke gjøre kartlegging av hogstfelt på vinterhalvåret grunnet snødekket vi opplever i store deler av Norge. Formålet med oppgaven er å teste om Sentinel-1 i teorien kan benyttes som en alternativ kartleggingsmetode når Sentinel-2 ikke rekker til. SAR-prinsippet er ikke avhengig av klar sikt for å kartlegge terrenget, og kan i teorien se endringer som hogstfelt året rundt ved snødekke, mørketid og overskyer.

SAR-målinger har en rekke avanserte formler og metoder liggende til grunn for dens funksjonalitet og bruksområdet. I oppgaven har fokuset vært å teste ut tre klassifiseringsmoduler på SAR-bilder, for å gjøre det har all data blitt preprosessert på lik måte. Preprosesseringen av SAR-data er høyst krevende både i datakraft og tid. Grunnet krevende prosessering er bildene behandlet med de mest grunnleggende prosessene som orienterer, glatter og fjerner støy fra bildene.

I analysen vektlegges hvilke tidspunkt og deler av SAR-dataen som benyttes for å best mulig klassifisere hogstfelt, totalt er det tre datasett som testes.

Datsett en: Målinger før og etter vintersesong

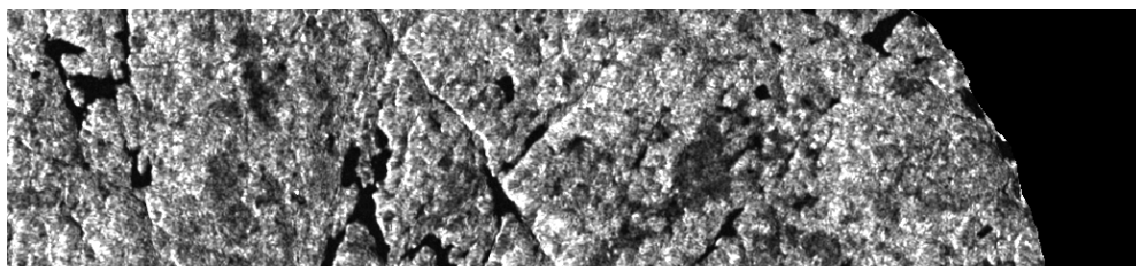
Bilder før og etter vintersesongen inneholder målinger før hogst mellom 22.05.2019 til 01.10.2019, og etter hogst målingene fra 04.04.2020 til 01.09.2020. Disse tidspunktene reproducerer tidspunktene brukt til data i Sentinel-2 klassifisering, og gir et utgangspunkt til hvor godt SAR klarer seg i forhold til multispektrale målinger. Datasettet bruker både VV- og VH-polarisering.

Datsett to: Målinger før og under vintersesong

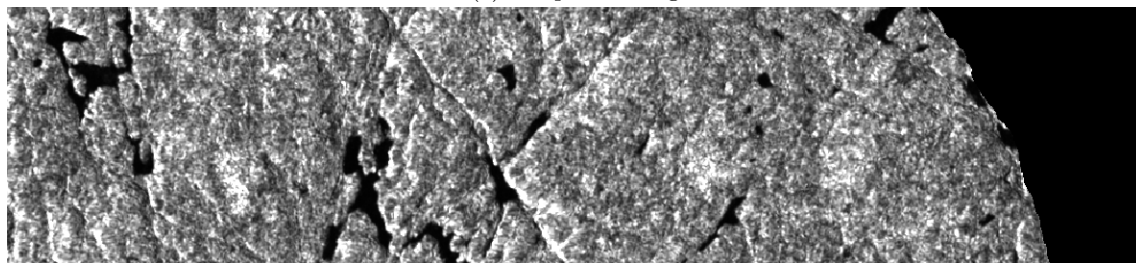
Bildene før og under vintersesong bruker samme før-hogst data som datasett en, men ettermålingene er utført fra 06.12.2019 til og med 04.04.2020 i perioden all hogsten brukt i oppgaven skjer. Formålet er å se om det er mulig å utføre klassifiseringer på Sentinel-1 data som er utenfor sesongen til Sentinel-2. Datasettet bruker både VV- og VH-polarisering.

Datsett tre: Målinger før og under vinter med VH-polarisering

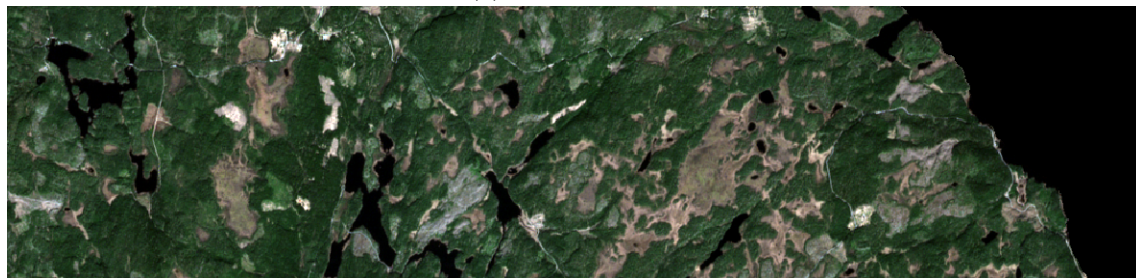
Sentinel-1 bildene forklart i 2.5.3 har flere polariseringer. Ved visuell fremstilling er det tilsynelatende et større skille mellom skog og hogst i VH-polarisering. Figur 3.9 viser eksempel hvor hogstfelt kommer mye tydeligere frem i VH-polarisering som mørke flekker. For å teste om VV-polariseringsverdi i maskinlæringen er datasett 3 bare med målinger mellom høst og vinter med VH bånd.



(a) VH-polarisering



(b) VV-polarisering



(c) RGB

Figur 3.9: Visuell sammenligning av SAR VV-, VH-polarisering og RGB bilde fra april 2020.

3.4.2 Produksjonsløype maskinlæring

Produksjonsløypen til maskinlæringene vises i figur 3.10 med stegvis inndeling, dette utdypes i videre i delkapittelet.

Databehandling av polygoner

I steg en og to får Python scriptet data fra alle polygonene som maskeres på hvert enkelt bilde i datasettene. Hver enkelt polygon har unike piksler med egne verdier som varierer fra nabo-pikslene. For å trekke ut egenskapene til hver polygon blir alle pikslene midlet. Det er mulig å hente ut andre verdier som minimum, maksimum og gjennomsnitt. Median-verdi benyttes fordi SAR-bildene kan inneholde mye støy, som gir ekstreme verdier både i positiv og negativ retning i gjennomsnittsverdi. En slik klassifisering hvor spesifikke verdier hentes ut kalles en objektbasert klassifisering. I teorien skal hogstfeltene ha en relativt lik signatur, og median-verdi betraktes som representativt for denne typen overflate i oppgaven.

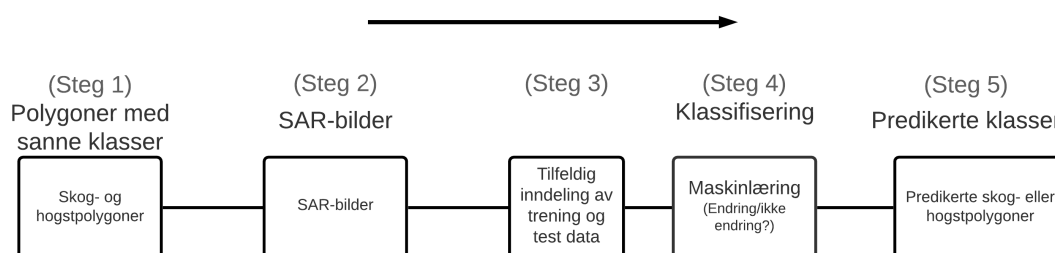
Inndeling av trening- og testdata

I steg tre behandles alle polygonene av en funksjon som splitter datasettene i test og trening data. Fordelingen til test og trening er fast, med like stor inndeling for alle datasettene på 70% trenings- og 30% testdata.

Klassifisering

I steg fire ble datasettene kjørt igjennom tre forskjellige ML for å finne ut hvilke som gir beste resultat, og for kontroll opp mot hverandre. I steg fem predikerer ML hvilke klasser testdatasettet tilhører. Etter predikering valideres klassifiseringene og resultatene kan si om en polygon er riktig eller feilklassifisert.

ML skal i prinsippet se om et felt er hogst eller skog ved endring, differanser og likheter i verdiene til skog- og hogstpolygonene. Alle datasettene som er målt til og med oktober 2019 vil både skog- og hogstpolygonene returnere verdi som tilsier de er skog, men straks et polygon returnerer ny verdi tilsier det en endring, altså skogen er hogget ned i de bestemte polygonene.



Figur 3.10: Klassifisering illustrasjon.

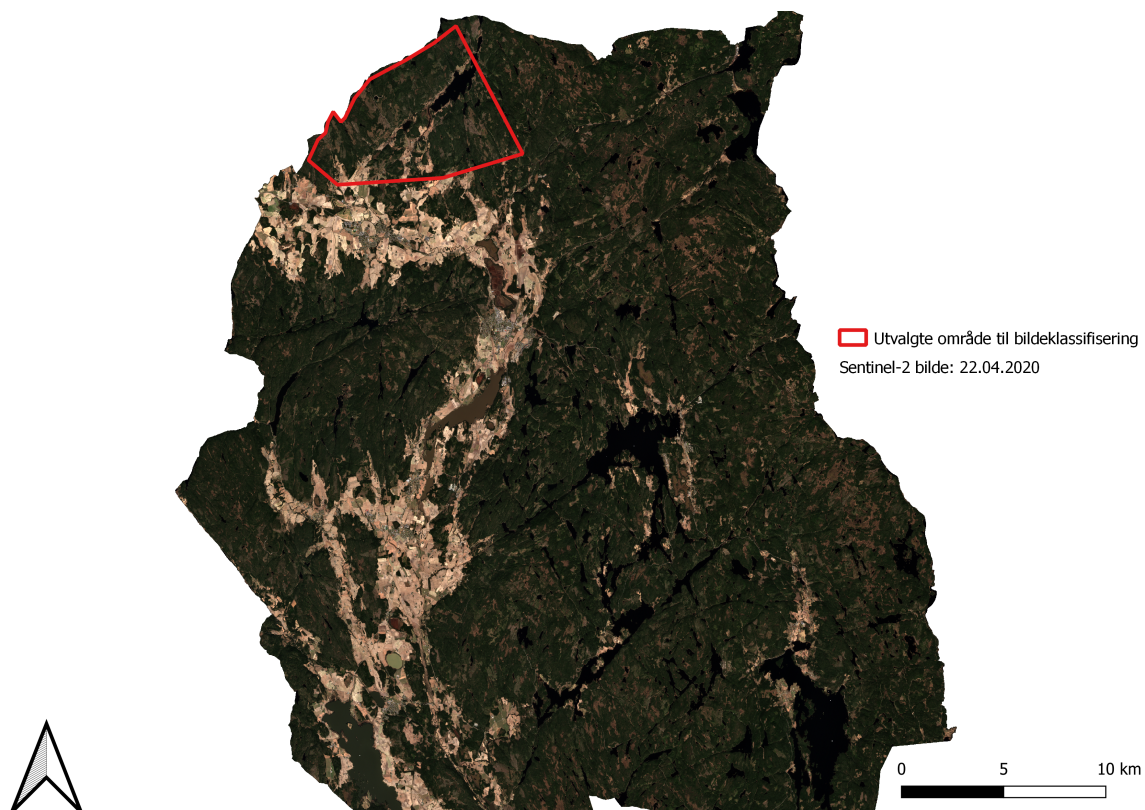
3.4.3 Bildeklassifisering

Etter trening og predikering av hvert enkelt datasett beskjerer bildene i datasettet ned til er område nordvest i Aurskog-Høland vist i figur 3.11. Det utvalgte området omfatter en større samling av nye og store hogstfelt fra vinteren 2019-2020, tidligere vist i figur 3.6.

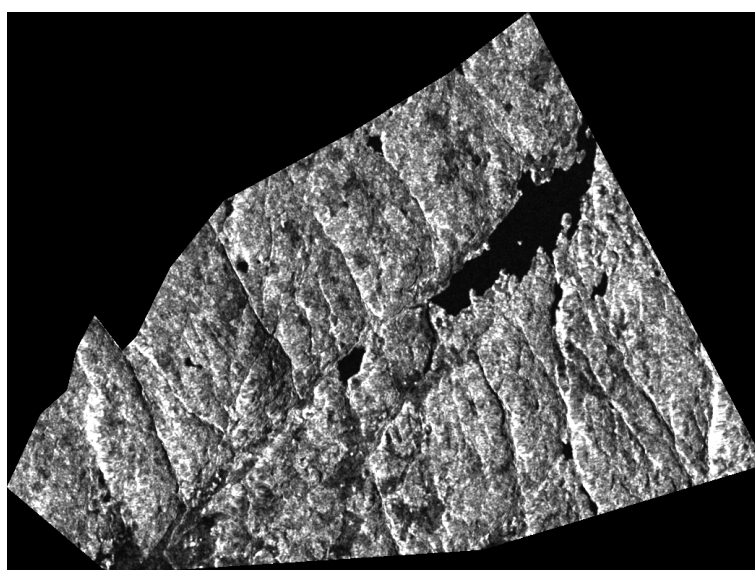
Alle SAR-bildene i datasettene brukt til klassifisering på polygoner klippet ned til området vist i figur 3.11. Sluttresultatet blir noe lignende som bildet vist i figur 3.12.

Prosesseringen utført for å klassifisere på bilder visualiseres i figur 3.14. Maskinlæringsmodellene er ferdig trent etter objektbasert klassifisering på polygonene, og brukes direkte til å klassifisere bildene. Algoritmene utfører piksel basert klassifisering på enkelt piksler i bildet. Alle bildene tas med i klassifiseringen, hvor verdien fra pikslene på samme posisjon samlet brukes for å si om det er skog heller hogst. Bildeklassifiseringen gir et innblikk i hvordan ML-algoritmene tolker bildene, og hvilke overflater de eventuelt bommer på.

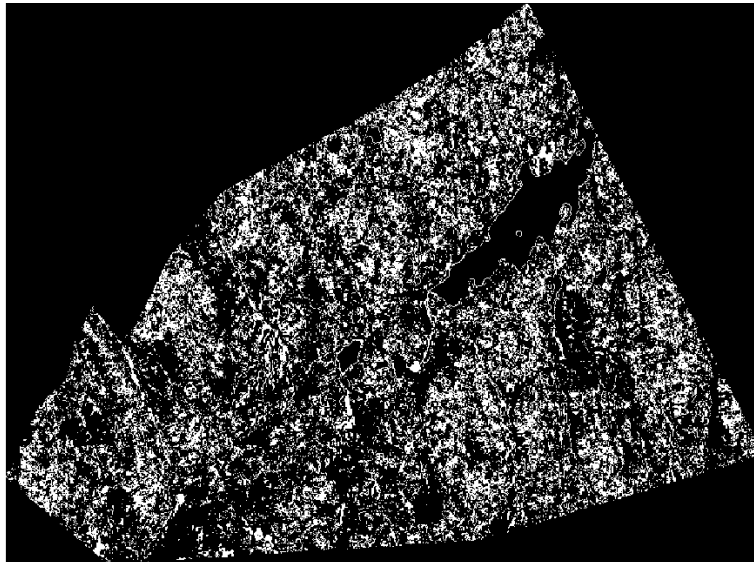
Eksempel på resultat fra pikselklassifisering vises i figur 3.13, det viser klassifisering utført med nærmeste nabo. Bildet returnert inneholder mye støy, noe som er gjengående for alle klassifiseringene utført på denne måten.



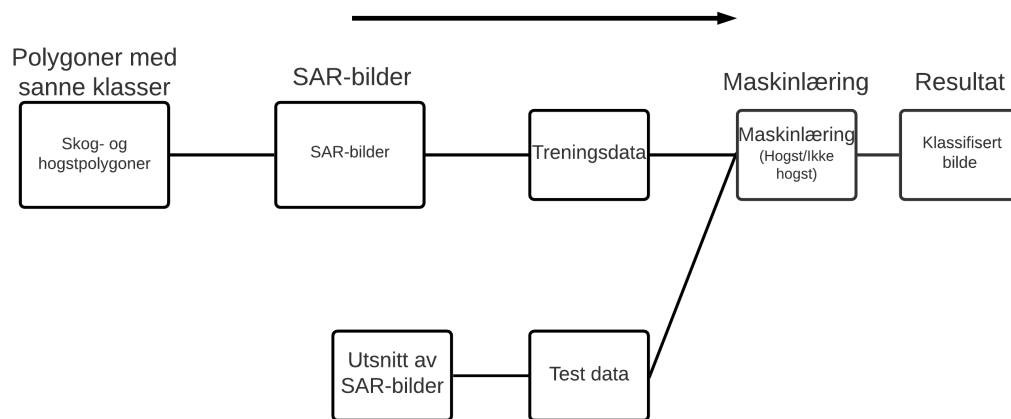
Figur 3.11: Klassifisering utvalgt område.



Figur 3.12: Utsnitt av bilde brukt til bildeklassifisering.



Figur 3.13: Eksempel på bildeklassifisering.



Figur 3.14: Produksjonsl ype bildeklassifisering.

4. Resultat

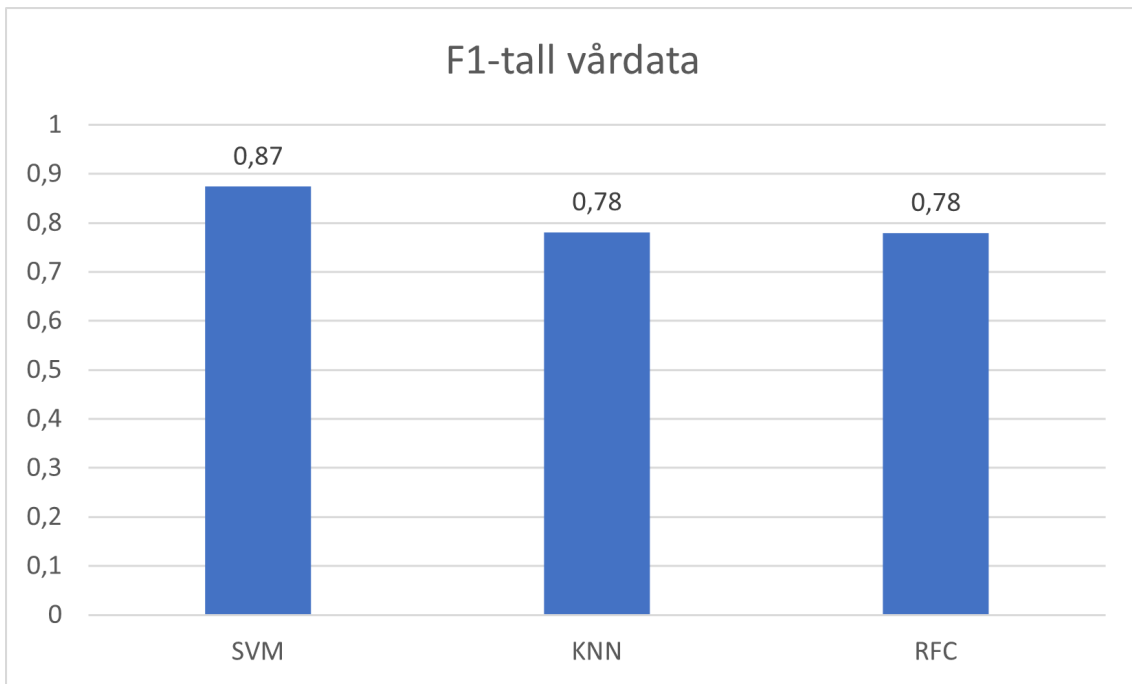
I dette kapitlet presenteres resultatene fra klassifiseringen og metoden lagt frem i kapittel 3. Første del av resultatkapitlet presenterer resultatene underveis i klassifiseringen for å gi en bedre forståelse av sluttresultatet. Resultatene presenteres i grafer som bruker f1-verdien regnet ut i fra forvirringsmtrisene i vedlegg. Andre del vil sammenligne de forskjellige resultatene i form av plot som danner grunnlaget til diskusjon i Kapittel 5.

4.1 Resultat datasett

I denne seksjonen blir resultatene fra datasettene og de forskjellige ML-algortimene presentert. Alle plottene er basert på snittet f1-tall. Hele tabellene og forvirringsmtrisene med resultater ligger som vedlegg.

4.1.1 Datasett en: Før og etter vintersesong

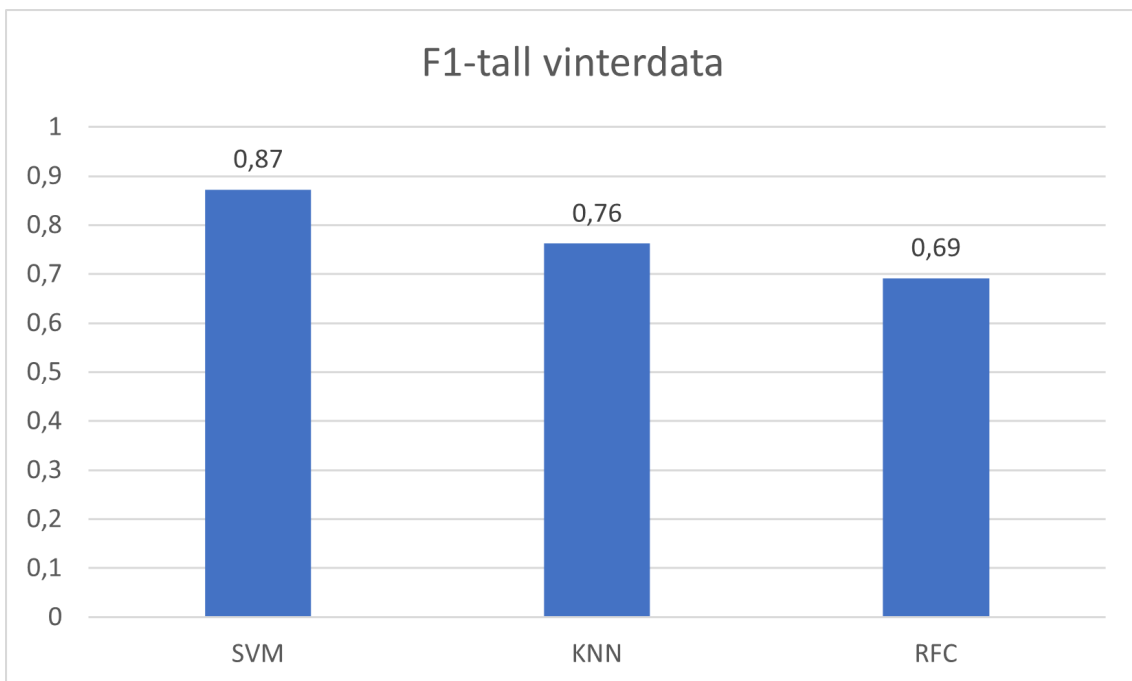
Figur 4.1 viser hvor godt ML klassifiserer hver for seg. Resultatene viser at SVM gir de beste resultatene med f1-tall som tilsier en nøyaktighet på 87%, 9 prosentpoeng bedre enn KNN og RFC. Fovirringsmtrisene i figur A.2 viser hvordan fordelingen av klassifiseringen er, og at støttevektormaskinen treffer bedre enn de to andre på både hogst og skog.



Figur 4.1: F1-tall før og etter vintersesong.

4.1.2 Datasett to: Før og under hogst

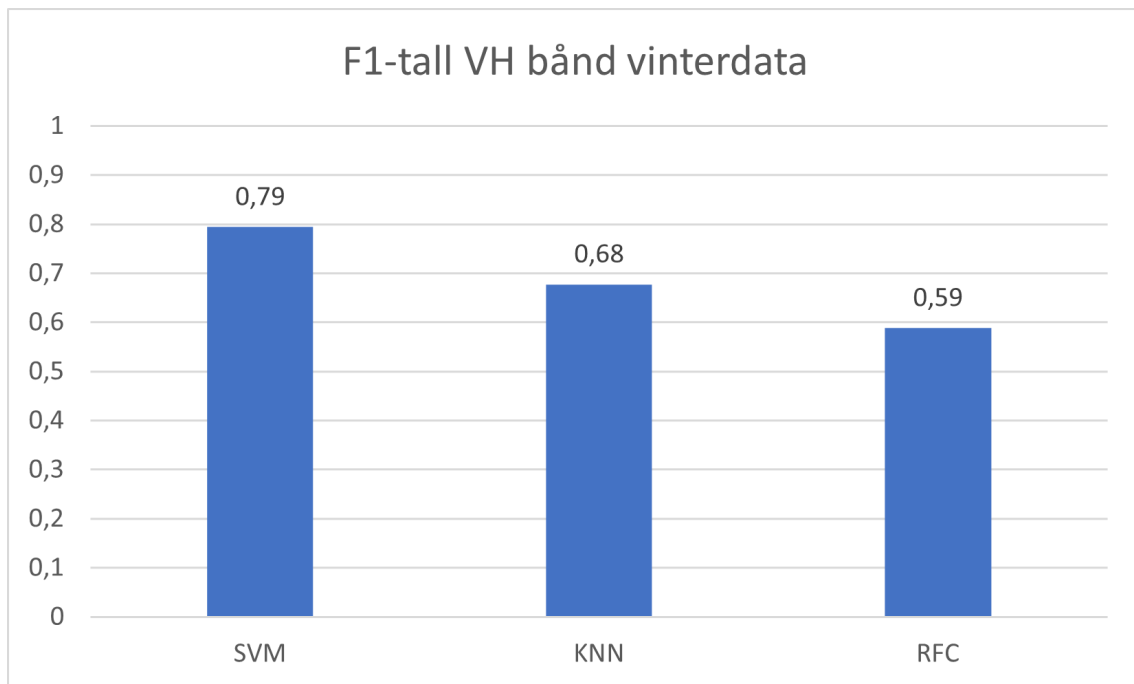
Figur 4.2 viser at SVM også på dette datasettet predikerer mest nøyaktig. I datasett to har SVM en presisjon på 87%, hele 11 og 19 prosentpoeng bedre predikering enn KNN og RFC på 76% og 69%.



Figur 4.2: F1-tall før og under hogst.

4.1.3 Datasett tre: Før og under hogst med VH-polarisering

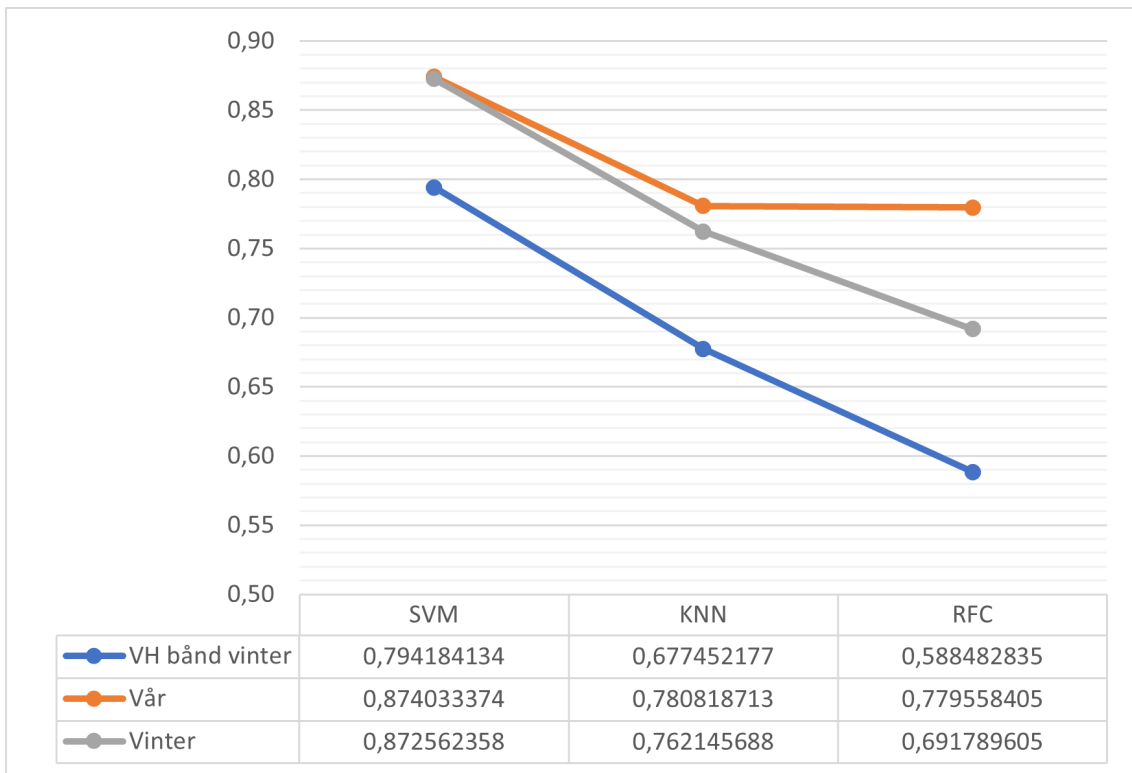
Dette datasettet har en generelt lav nøyaktighet som kan tilsi at ML har et manglende datagrunnlag for både test og trening. Som vist i figur 4.3 skårer SVM best på dette datasettet på 79%, hele 11 og 20 prosentpoeng bedre enn KNN og RFC på 68% og 59%.



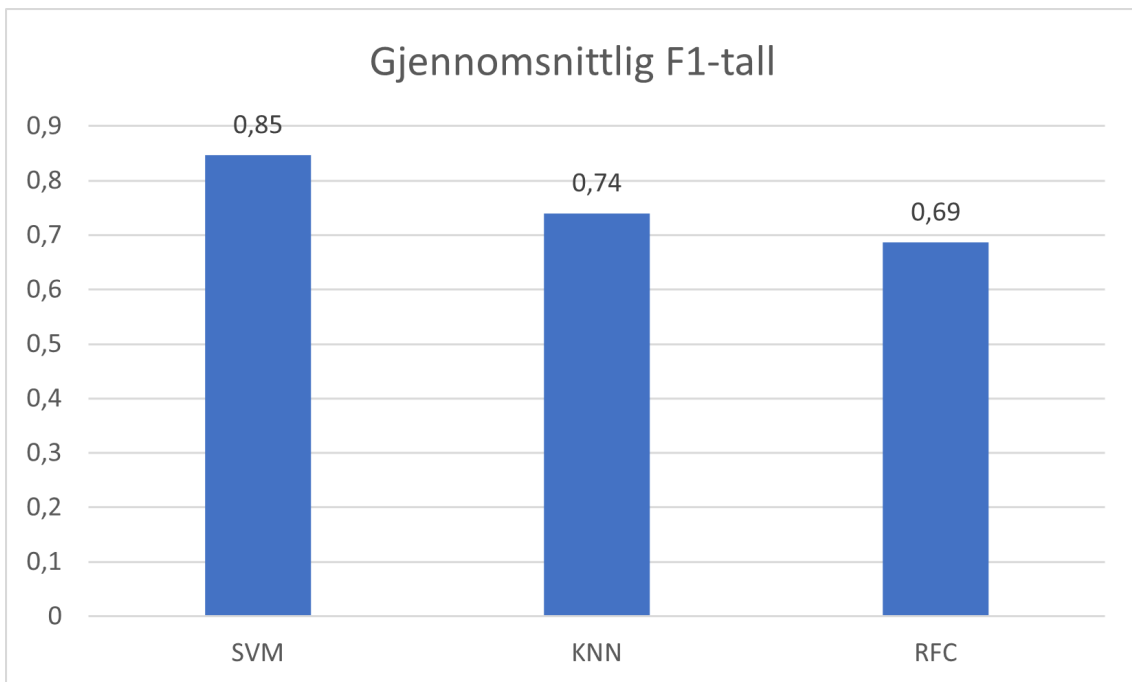
Figur 4.3: F1-tall før og under hogst med VH-bånd.

4.2 Resultat sammenligning

I figur 4.4 sammenlignes alle resultatene. Figur 4.5 viser det gjennomsnittlige F1-tallet til ML. Som vist, presterer SVM i gjennomsnitt best med ett snitt på 85%, hvor KNN kommer ut nest best med 74%, og RFC dårlig med 69%.



Figur 4.4: Sammenligning av F1-tall.

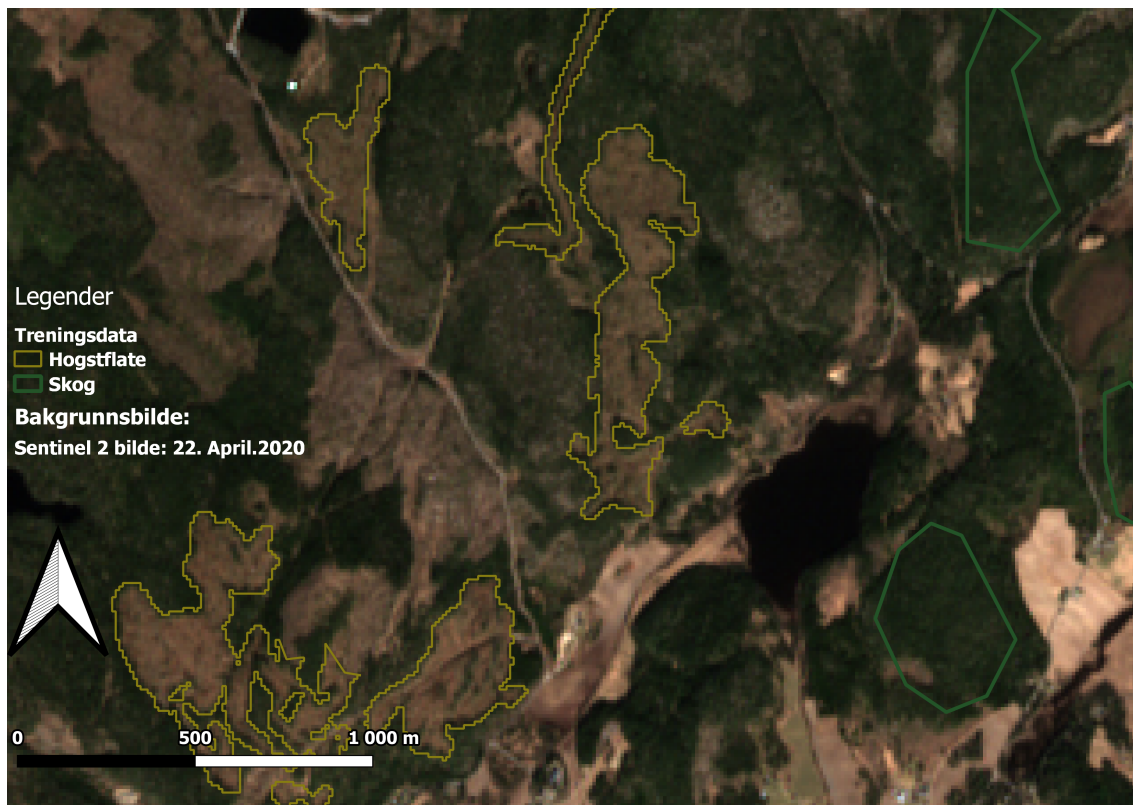


Figur 4.5: Gjennomsnitt F1-tall.

4.2.1 Bildeklassifisering

Totalt er det ni klassifiserte bilder. I resultatene presenteres det dårligste, og det beste resultatet. Korrelasjonen mellom nøyaktighet på polygonklassifisering og bildeklassifisering er tilsynelatende

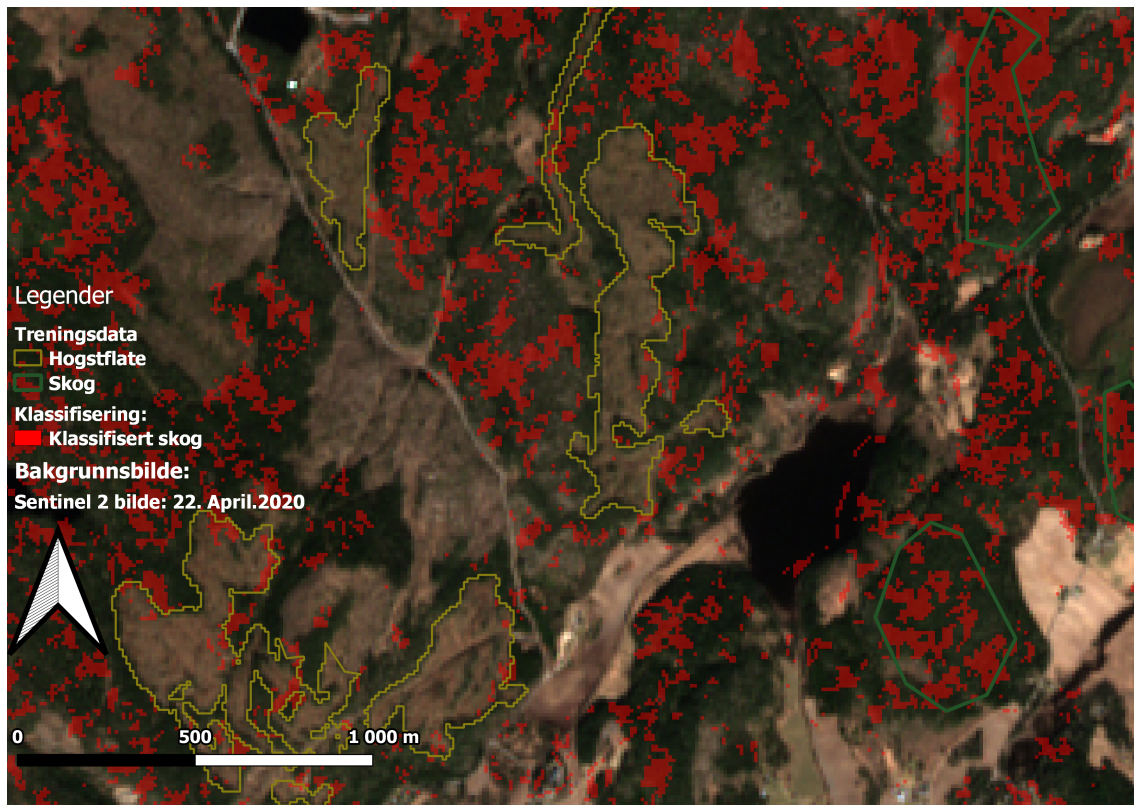
høy, men ikke målbar fordi klassifiseringen er gjort uten et mål og posisjon på treningspolygonene. Bakgrunns-kartet brukt i begge bildene er Sentinel-2 bildet fra 22.04.2020, etter alle hogstflater brukt i oppgaven er felt. Som nevnt i 3.4.3 inneholder alle bildeklassifiseringene mye støy, som et resultat av enkeltpikeseler imellom skog og hogst med en annen retur enn pikslene rundt.



Figur 4.6: RGB-bilde av predikert område.

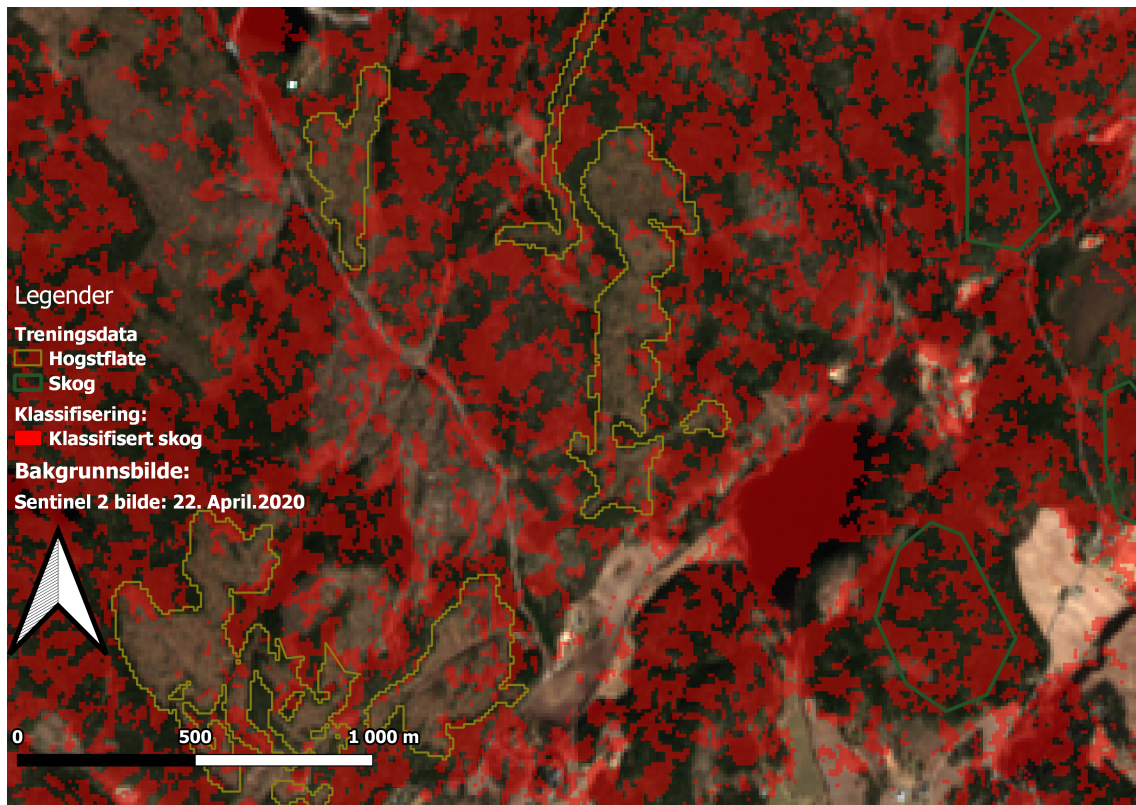
For å lettest visualisere bildeklassifiseringen, vises bare de pikslene klassifisert som skog med rød gjennomsiktig farge. Alle områdene som ikke er farget rødt er klassifisert som hogst. Bildet i figur 4.6 viser utsnittet brukt i bildeklassifisering, det er flere hogstfelt som ikke er markert. Dette fordi hogstfeltene er felt før tidsrommet i oppgaven, men bildeklassifiseringen ser i realiteten om et område er hogst eller skog. Området nede på høyre halvdel av bildet er jordbruk som kan feiltolkes som hogstflater av ML. Jordbruk og andre fremmedelementer i bildene betraktes ikke som feilklassifisering da andre objekter enn skog og hogst ikke er tatt med i klassifiseringen. Alle bildeklassifiseringene har en forskyvning i forhold til RGB-bildene. Forskyvningen er konstant, og lik på alle bildene. Mest sannsynlig skyldes forskyvningen oppdelingen av bildene, som så settes sammen igjen.

Bildet vist i figur 4.7 er bildeklassifiseringen utført av RFC på datasett tre som er den dårligste modellen fra kapittel 4.2. Forvirringsmatrisen i figur A.3 (c) viser at modellen sliter med klassifisering av skog, og tegner store deler av skogområdene som hogst. Resultatene i forvirringsmatrisen gjenspeiler seg i resultatene fra bildeklassifiseringen, hvor den sliter med å markere skog og setter det meste som hogst.



Figur 4.7: Bildeklassifisering dårligste resultat med RFC.

Bildet i figur 4.8 er bildeklassifiseringen fra SVM klassifiseringen på datasett en som har gitt best resultat fra 4.2. Denne klassifiseringen klarer i større grad å klassifisere skog med høyere konsentrasjon, og mindre støy. Sammenlignet er klassifiseringen på datasettene to med SVM veldig like. Vannet vist i bildet er også klassifiser som skog. Både vann og hogst har lav returstyrke, og med det vist som mørke overflater. At modellen klarer å skille de to overflatene tyder på at modellen har et godt nok datagrunnlag på hogstflater til å skille to relativt like objekter. Fordi modellen bare er trent opp på hogst eller skog blir den tvunget til å tegne overflaten som en av klassene. Det er noe mer støy i hogstflatene, men sett i bort fra forskyvningen klart å finne små skogområder som ligger midt i hogsten.



Figur 4.8: Bildeklassifisering beste resultat med SVM.

5. Diskusjon

I dette kapittelet diskuteres resultatene, og metoden brukt. Diskusjon rundt mulige forbedringer og svakheter i metoden kommer som andre del.

5.1 Resultater

Resultatene fra maskinlæringen indikerer en spredning mellom maskinlæringene, men en jevn fordeling på nøyaktighet mellom datasettene. Differansen mellom data målt på tidspunkt bakken er bar, og perioder hvor det er snødekke er relativt liten, men målinger før og etter har litt høyere f1-tall. Med SVM og KNN er resultatene relativt like på datasett en og to. Små variasjoner som dette kan slå ut annerledes ved å tilpasse parameterne gitt til algoritmene i større grad til hvert enkelt datasett, og kjøre en større kompleks GridSearchCV.

Datasett tre med bare VH-polarisering har generelt lavest presisjon. Datasett en og to bruker VV- og VH-polarisering, som gir totalt 2x9 målinger å trene på. Datasett tre har ni bilder. Mindre treningsdata vil oftest medføre lavere nøyaktighet ved klassifisering.

RFC gir overraskende lav nøyaktighet og sliter tilsynelatende med over- eller undertilpassing til hogstflatene, utenom datasett en, som vist i figur A.2 (c) har en jevn fordeling på og skogklassifiseringen. Resultatene fra de første testene var nesten like gode som de fra SVM, men årsaken til plutselig dårlige resultater ble ikke funnet i tide. Flere justeringer på parameterne er forsøkt, men resultatene presentert er de beste etter flere forsøk. På grunn av problemene med RFC konkluderes det med at resultatene fra RFC ikke er representative for problemstillingen og sluttresultatet.

Bildeklassifiseringen vist i figur 4.8 viser at Sentinel-1 bilder har mulighet til å brukes videre i bildeklassifisering. Resultatene er ikke særlig stødige alene, men Sentinel-1 har tilstrekkelig med informasjon til å skille hogst, skog, og muligens andre objekter i terrenget som vann og jorder.

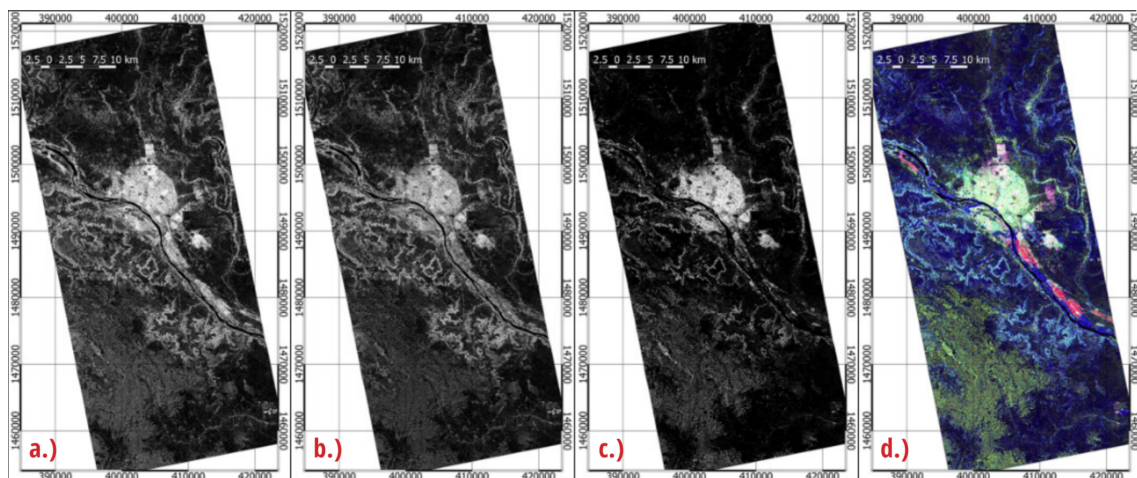
SVM klarer i stor grad å finne nye hogster rett etter felling. Egenskapen SAR har til å kunne utføre målinger hele året rundt viser seg å ha en mulig nytteverdi, mer rundt dette kommer i konklusjon.

5.2 Usikkerhet i metode

5.2.1 Datagrunnlag

I oppgaven ble VV- og VH-polarisering tatt i bruk. Forklart i 2.5.3 har HV- og VH-krysspolarisering samme egenskaper, men HH-polarisering har mulighet til å bidra med ytterligere informasjon. HH-polarisering er egnet til bygninger og noe vegetasjon [Flores et al., 2019]. Selve metoden brukt i oppgaven er ikke avhengig av HH-polarisering, men med totalt tre polariseringer med forskjellige egenskaper er det mulig å gjenskape et RGB-bilde vist som et eksempel fra Flores et al. [2019] i figur 5.1. Et slikt RGB-bilde kan i teorien benyttes som et ytterligere bilde i en klassifisering. Selve målingene fra Sentinel-1 kommer i nord- og sørgående(stigende/synkende) retning. Dette er ikke mulig å spesifisere i nettsystemet ColHub, men tilgjengelig i andre plattformer. Retningen satellittene beveger seg i målingsøyeblikket påvirker skyggen kastet av objekter som trær og variasjon i

terrenget. Ved å ta i bruk både nord og sør gående målinger kan skygge effekten utjevnes, både i pre-prosessering og i selve maskinlæringen. I artikkelen publisert av Bouvet et al. [2018] brukes skyggeendringen fra skog i stigende og synkende retning, og over tid til å finne hvor hogster har skjedd.



Figur 5.1: RGB bilde med SAR data.

[Flores et al., 2019] (a) HH-polarisering (b) VV-polarisering (c) HV-polarisering (d) RGB kombinasjon av polariseringene

5.2.2 Preprosessering

Effekten av forskjellige preprosesseringer ble ikke betraktet som en problemstilling i oppgaven, men det kan påvirke hvordan kontraster mellom forskjellige objekter i terrenget blir i SAR-bildet. Preprosesseringen brukt i oppgaven er grunnleggende, og dekker de mest nødvendige prosessene beskrevet i brukerveiledningen til ESA Braun [2020]. [Vaglio Laurin et al., 2021] brukes tilsvarende preprosessering. I Doblas et al. [2020] brukes tilsvarende prosessering og data som Vaglio Laurin et al. [2021], men med prosessen Remove boarder noises som går i å fjerne støy i utkanten av SAR-bildet, denne prosessen er ignorert da bildet klippes ned videre i Python. De bruker også strekking av bildet i en dB-skala. Tilsvarende pre-prosesseringer som i Doblas et al. [2020] brukes i forsøkene til Cremer et al. [2020] og Bouvet et al. [2018], bare uten dB-skala.

5.2.3 Strecking av bilder

Som forklart av Filipponi [2019], og brukt i Doblas et al. [2020] er det mulig å strekke bildene i den dB-skala som fremhever kontraster ved å bruke en logaritmisk transformasjon. Denne prosessen ble neglisjert for å unngå tap av eventuell data, og kunstig strekking av bildene, noe som i oppgaven ble betraktet som en mulig feilkilde i klassifisering. Senere arbeid kan denne preprosessering sees videre på, og om den muligens kan hjelpe klassifiseringen å skille hogst og skog samt strekke bildene så de blir likere. Målingene har variert utstrekking, noen bilder er mørkere enn andre, men det skal ikke gi noe utfall da det er likt for alle datasettene.

5.2.4 Trenings- og testdata

Metoden benyttet for å hente ut data beskrevet i kapittel 3.4.2, tar medianen av alle pikslene innenfor hvert polygon. I oppgaven ble denne metoden brukt for å kutte ned prosesseringstid grunnet begrenset tilgang på datakraft, og et forsøk på å eliminere støy. Om hele polygonene blir benyttet vil algoritmene ha et bedre grunnlag og mer treningsdata å basere seg på til klassifisering

av test data, men for å gjøre det må dyp læring benyttes (deep learning) som er et annet fagfelt innen maskinlæring.

Ved bruk av kun en middelværdi blir både form og størrelse på hogstfeltene ignorert, det vil si at et lite hogstfelt vil ha like stor uttelling på treningen til algoritmene som ett stort Polygon. Zhou et al. [2017] forteller at standard maskinlæring møter en kritisk utfordring i å avdekke skjulte verdier i “big data”. Ved å benytte dyp læring, med nevralt nettverk forklart i Raschka and Mirjalili [2017], kan oppdatering av vektorer og flere iterasjoner av maskinlæring avdekke skjult informasjon som størrelser, mønster og forskjellige returverdier som kommer til nytte i en bedre og utdypende klassifisering både på polygoner og i bilder.

5.2.5 Bildeklassifisering

Bildeklassifiseringen i oppgaven ble gjort på hver enkelt piksel fra alle bildene sammenlagt. Denne metoden gir et grunnlag for å vise hvordan maskinlæringen jobber, og finne mulige feil. En mulig forbedring av bildeklassifisering kan gjøres ved å betrakte hver enkel piksel, med en vektning av nabo-pikslene. Denne metoden kan eliminere mye av støyet vist i figur 4.7 og 4.8. Illustrasjonen vist i figur 5.2 er et eksempel på hvordan en piksel kan klassifiseres. Vektorer kan bli lagt på pikslene rundt for å bestemme hvor stor innvirkning de skal ha på klassifiseringen av piksel som klassifiseres. Om pikslene enda lengre ut skal bli tatt med kan pikslene som er nærmere bli tilegnet en høyere vekt enn de på avstand, altså de nærmere vil ha høyere innvirkning på resultatet.

Selve bildeklassifiseringen vil slite med å skille ut ny hogst fra tidligere hogst. Det kan derfor være aktuelt å bruke både gamle og nye hogst for å gi bildeklassifiseringen mer data å trene på.

Om U-Net brukt til Sentinel-2 klassifiseringen til Blom anvendes på SAR-bildene, kan fokuset bytte til å se på om en overflate er grodd eller ikke. Om overflaten plutselig endrer seg i noen av bildene kan en med U-Net markere områdene der endring har oppstått [Raschka and Mirjalili, 2017].



Figur 5.2: Bildeklassifisering med vektorer.

6. Konklusjon

De beste resultatene fra polygonklassifisering med SVM gir f1-tall på 87,4% før og etter hogst på våren, og 87,3% før hogst og under hogst på vinter. Hogstfeltene brukt til treningsdata er felt mellom oktober 2019 og april 2020. Dette indikerer at Sentinel-1 klarer å utføre tilstrekkelig gode målinger til å detektere en god del av endringer i hogstflater på årstider, mørketid og værforhold. Sentinel-2 er svært begrenset. Bildeklassifiseringen vist i resultater indikerer også at metoden for å finne hogstfelt har mulighet til å komplementere Sentinel-2 klassifisering ved å finne eventuelle endringer.

Resultatene viser en lik trend mellom hver klassifisering, og relativt like resultater på datasettene med både VV- og VH-polarisering. Datasettet med bare VH-polarisering er mangelfullt, og gir ikke tilstrekkelig med informasjon til algoritmene. Påvirkningen av båndvalg kan peke til en mulig verdi i å legge til HH-polarisering som en ekstra kilde til informasjon spesielt i tilfeller det er urbane områder hvor det er viktig å skille ut bygninger.

Oppløsningen til en SAR-satellitt er med dagens teknologi er ikke høy nok til presis volumberegning av hogstfelt med veldig varierende form og størrelse. Derimot kan det brukes som en indikator for å se endring i trender, og komplementere klassifiseringer gjort med Sentinel-2 som har langt mer informasjon å basere klassifiseringen på.

Resultatene fra bildeklassifiseringen er overraskende gode, siden kun middelverdi av hvert polygon ble brukt i klassifiseringen, og hver piksel kan ha ganske store variasjoner i et SAR-bilde. Treningsdata ble nøye kontrollert og redigert før de ble tatt i bruk. Dette er tidkrevende og ikke nødvendigvis like hensiktsmessig å gjøre i større prosjekter, fordi treningsdatasettet er såpass finpusset vil det også gjøre jobben til klassifiseringen lettere, og unnlate mange naturlige feilkilder.

Konklusjonen er at Sentinel-1 SAR-bilder har tilstrekkelig informasjon til å utføre en endringsanalyse av hogstfelt, men at en kommersiell metode vil kreve mer avansert klassifisering, og mulig større bilde- og polygondatasett.

6.1 Videre arbeid

Utfordringene Sentinel-2 møter i Norge med snødekke, mørketid, mye nedbør og skydekke i deler av landet er en sterk motivasjon for å jobbe videre med Sentinel-1 bilder. Oppgaven indikerer at Sentinel-1 har en stort potensiale for skogkartlegging i Norge.

Her er noen forslag til videre arbeid:

- Benytte nord- og sørgående målinger både for støyreduksjon, og mulig tilføyende informasjon med skygger.
- Tilpasse preprosesseringsprosedyrer av SAR-data i større grad.
- Se på dyp maskinlæring (Deep learning), om det kan hente ut ytterligere informasjon fra polygoner og SAR-bildene.
- Multiklasse klassifisering, for å skille ut andre elementer i bildet som bygninger, vann, jorder.

-
- Samkjøre SAR-data med multispektrale bilder
 - Flere innmålinger totalt, og bruk av tidsserier for å finne et mer nøyaktig tidspunkt for hogst.

Bibliografi

- Copernicus scientific data hub. URL <https://colhub.met.no/>. Accessed: 2021-04-10.
- Esa: Grd format. URL <https://sentinel.esa.int/web/sentinel/technical-guides/sentinel-1-sar/products-algorithms/level-1-algorithms/ground-range-detected>. Accessed: 2021-01-03.
- Os module in python with examples. URL <https://www.geeksforgeeks.org/os-module-python-examples/>. Accessed: 2021-04-09.
- Qgis - the leading open source desktop gis. URL <https://qgis.org/en/site/about/index.html>. Accessed: 2021-04-29.
- Aurskog-høland (viken), a. URL <https://www.ssb.no/kommuneareal/aurskog-holand>. Accessed: 2021-05-03.
- Ssb: Arealbruk og arealressurser, b. URL <https://www.ssb.no/jord-skog-jakt-og-fiskeri/faktaside/skogbruk>. Accessed: 30-05-2021.
- Visma, csv-fil.
- Geopandas 0.9.0. URL <https://geopandas.org/>. Accessed: 2021-04-09.
- The intelligent diagramming application for every team. URL <https://lucid.co/product/lucidchart>. Accessed: 2021-05-18.
- Rasterio: access to geospatial raster data, a. URL <https://rasterio.readthedocs.io/en/latest/>. Accessed: 2021-04-09.
- rasterstats 0.15.0, b. URL <https://pypi.org/project/rasterstats/>. Accessed: 2021-04-09.
- Seasat (seafaring satellite) mission. URL <https://earth.esa.int/web/eoportal/satellite-missions/s/seasat>. Accessed: 2021-05-18.
- Alexandre Bouvet, Stéphane Mermoz, Marie Ballère, Thierry Koleck, and Thuy Le Toan. Use of the sar shadowing effect for deforestation detection with sentinel-1 time series. *Remote Sensing*, 10(8), 2018. ISSN 2072-4292. doi: 10.3390/rs10081250. URL <https://www.mdpi.com/2072-4292/10/8/1250>.
- Andreas Braun. Sar-based landcover classification with sentinel-1 grd products. 2020.
- Dick Carlsson and Mikael Rönnqvist. Supply chain management in forestry—case studies at södra cell ab. *European Journal of Operational Research*, 163(3):589–616, 2005.
- Felix Cremer, Mikhail Urbazaev, José Cortés, John Truckenbrodt, Christiane Schmillius, and Christian Thiel. Potential of recurrence metrics from sentinel-1 time series for deforestation mapping. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 13:5233–5240, 2020.
- Juan Doblaz, Yosio Shimabukuro, Sidnei Sant’Anna, Arian Carneiro, Luiz Aragão, and Claudio Almeida. Optimizing near real-time detection of deforestation on tropical rainforests using sentinel-1 data. *Remote Sensing*, 12(23):3922, 2020.

-
- ESA. Esa: Raw format, a. URL <https://sentinel.esa.int/web/sentinel/technical-guides/sentinel-1-sar/products-algorithms/level-0-products/raw>. Accessed: 2021-01-03.
- ESA. Esa: Sentinel-1 overview, b. URL <https://sentinel.esa.int/web/sentinel/missions/sentinel-1/overview>. Accessed: 2021-01-11.
- ESA. Polarimetry, c. URL <https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar/product-overview/polarimetry>. Accessed: 2021-05-22.
- ESA. Esa: Sentinel-2 technical guide msi instrument, d. URL <https://sentinel.esa.int/web/sentinel/technical-guides/sentinel-2-msi/msi-instrument>. Accessed: 2021-01-28.
- ESA. Esa: Sentinel-2 mission guide, e. URL <https://sentinel.esa.int/web/sentinel/missions/sentinel-2>. Accessed: 2021-01-26.
- ESA. Esa: Sentinel-2 mission guide overview, f. URL <https://sentinel.esa.int/web/sentinel/missions/sentinel-2/overview>. Accessed: 2021-01-26.
- ESA. Esa: Sentinel-2 user guide application land monitoring, g. URL <https://sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi/applications/land-monitoring>. Accessed: 2021-01-26.
- ESA. Esa: Sentinel-1 orbit, h. URL <https://sentinel.esa.int/web/sentinel/missions/sentinel-1/satellite-description/orbit>. Accessed: 2021-01-11.
- ESRI. Esri shapefile technical description. *An ESRI White Paper*, 1998.
- Federico Filippini. Sentinel-1 grd preprocessing workflow. *Proceedings*, 18(1), 2019. ISSN 2504-3900. doi: 10.3390/ECRS-3-06201. URL <https://www.mdpi.com/2504-3900/18/1/11>.
- Karen Fletcher. *SENTINEL 1: ESA's Radar Observatory Mission for GMES Operational Services*. European Space Agency, 2012.
- A Flores, K Herndon, R Thapa, and E Cherrington. The sar handbook: Comprehensive methodologies for forest monitoring and biomass estimation. *NASA Marshall Space Flight Center: Huntsville, AL, USA*, 2019.
- J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3): 90–95, 2007. doi: 10.1109/MCSE.2007.55.
- Dave Kuhlman. *A python book: Beginning python, advanced python, and python exercises*. Dave Kuhlman Lutz, 2009.
- JS LEE, I JURKEVICH, P DEWAELE, P WAMBACQ, and A OOSTERLINCK. Speckle filtering of synthetic aperture radar images: A review. *Remote Sensing Reviews*, 8:313–340, 1994.
- Sk Sazid Mahammad and R Ramakrishnan. Geotiff-a standard image file format for gis applications. *Map India*, pages 28–31, 2003.
- Wes McKinney. *Python for data analysis: Data wrangling with Pandas, NumPy, and IPython*. "O'Reilly Media, Inc.", 2012.
- Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. 2018.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- David Powers. Evaluation: From precision, recall and f-factor to roc, informedness, markedness correlation. *Mach. Learn. Technol.*, 2, 01 2008.
- S. Raschka and V. Mirjalili. *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2, 3rd Edition*. Packt Publishing, 2019. ISBN 9781789958294. URL <https://books.google.no/books?id=sKXIDwAAQBAJ>.
-

Sebastian Raschka and Vahid Mirjalili. *Python Machine Learning: Machine Learning and Deep Learning with Python, Scikit-Learn, and TensorFlow, 2nd Edition*. Packt Publishing, 2nd edition, 2017. ISBN 1787125939.

Merrill I Skolnik. Introduction to radar. *Radar handbook*, 2:21, 1962.

Merrill I Skolnik. *Radar handbook*. McGraw-Hill Education, 2008.

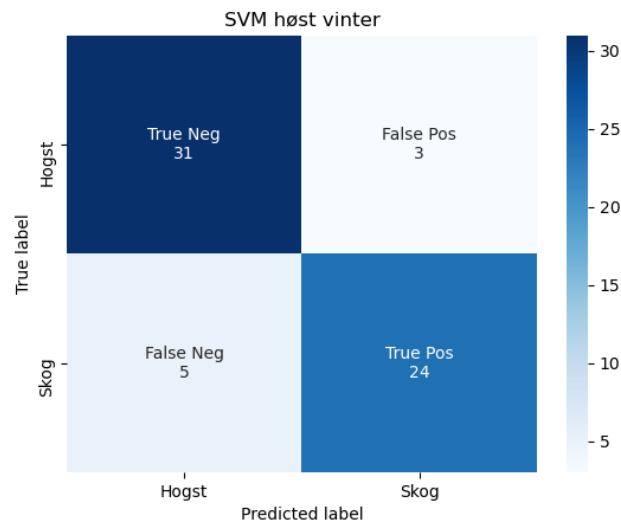
Gaia Vaglio Laurin, Saverio Francini, Tania Luti, Gherardo Chirici, Francesco Pirotti, and Dario Papale. Satellite open data to monitor forest damage caused by extreme climate-induced events: a case study of the vaia storm in northern italy. *Forestry: An International Journal of Forest Research*, 94(3):407–416, 2021.

Lina Zhou, Shimei Pan, Jianwu Wang, and Athanasios V. Vasilakos. Machine learning on big data: Opportunities and challenges. *Neurocomputing*, 237:350–361, 2017. ISSN 0925-2312. doi: <https://doi.org/10.1016/j.neucom.2017.01.026>. URL <https://www.sciencedirect.com/science/article/pii/S0925231217300577>.

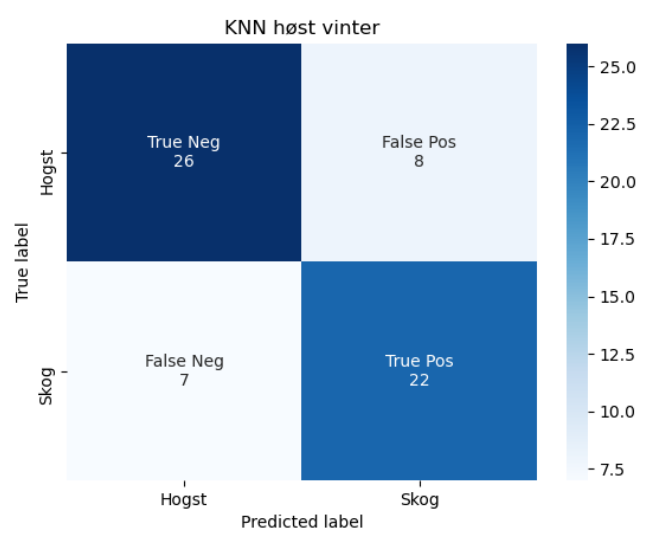
A. Vedlegg

A.1 Forvirringsmatriser

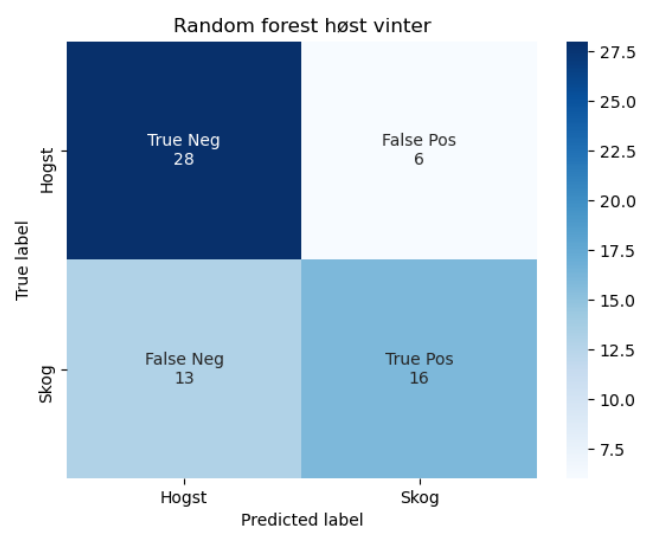
Vedlegget viser resultatet med forvirringsmatrisene fra maskinlæringen.



(a) SVM

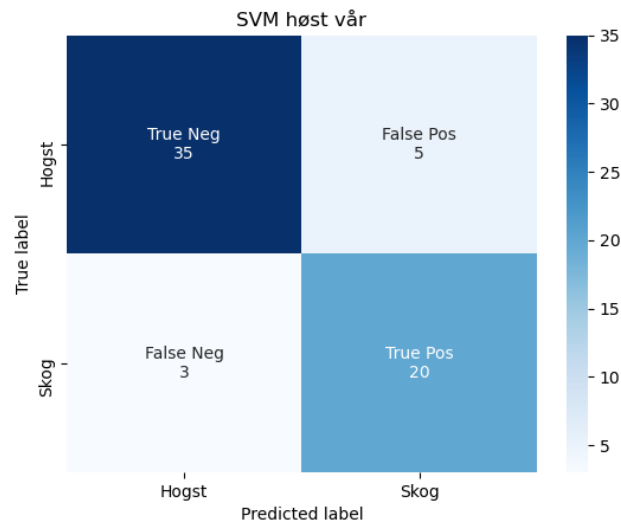


(b) KNN

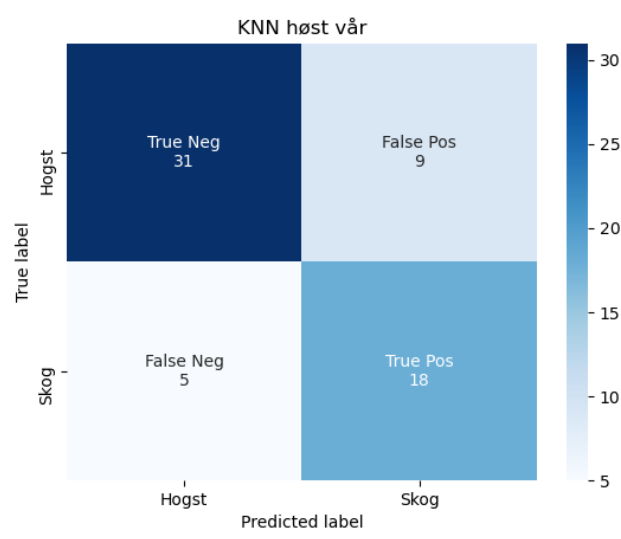


(c) Random forest

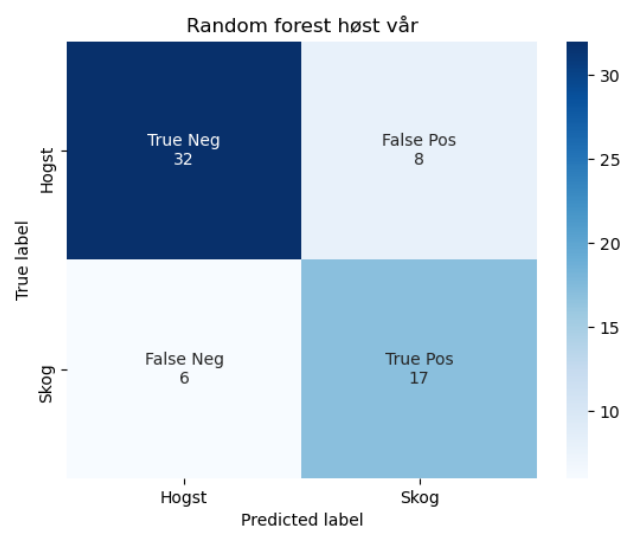
Figur A.1: Forvirringsmatriser av klassifiseringer på før og under -hogst data.



(a) SVM

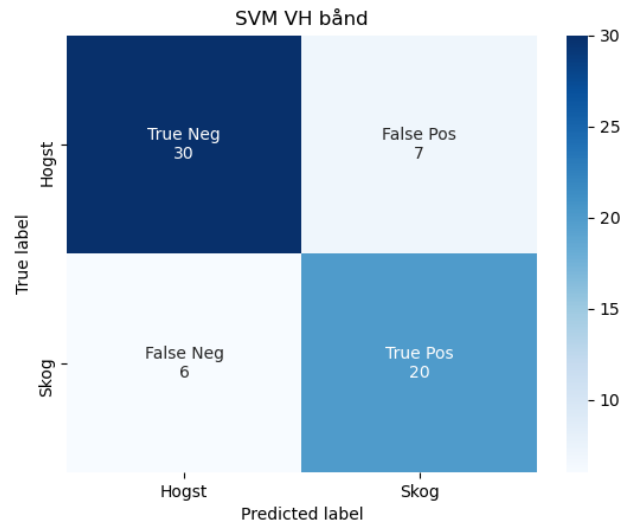


(b) KNN

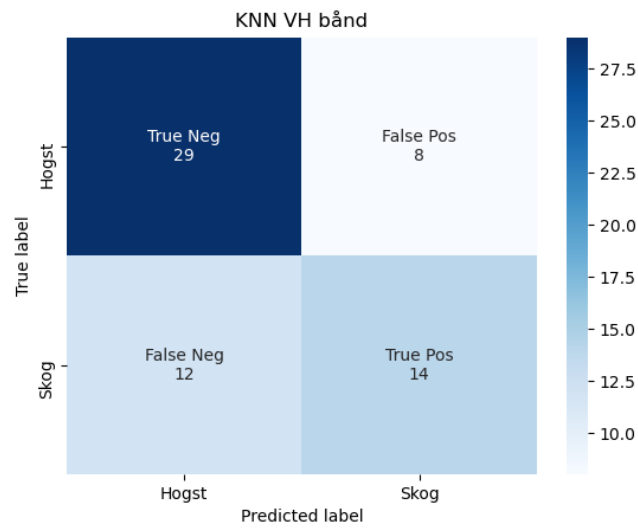


(c) Random forest

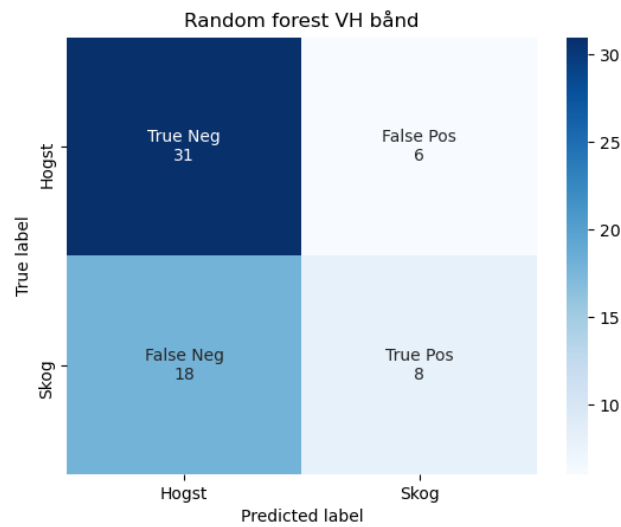
Figur A.2: Forvirringsmatriser av klassifisering på før og etter -hogst data.



(a) SVM



(b) KNN



(c) Random forest

Figur A.3: Forvirringsmatriser av klassifiseringer på før og under -hogst, med VH-bånd.

A.2 Tabeller

Vedlegget viser tabellene fra klassifiseringsrapportene.

SVM vår	precision	recall	f1-score	support
Hogst	0,92105263	0,875	0,8974359	40
Skog	0,8	0,86956522	0,83333333	23
avg / total	0,87685881	0,87301587	0,87403337	63

KNN vår	precision	recall	f1-score	support
Hogst	0,86111111	0,775	0,81578947	40
Skog	0,66666667	0,7826087	0,72	23
avg / total	0,79012346	0,77777778	0,78081871	63

RFC vår	precision	recall	f1-score	support
Hogst	0,84210526	0,8	0,82051282	40
Skog	0,68	0,73913043	0,70833333	23
avg / total	0,78292398	0,77777778	0,7795584	63

Tabell A.1: Klassifiseringsrapport datasett en.

SVM vinter	precision	recall	f1-score	support
Hogst	0,86111111	0,91176471	0,88571429	34
Skog	0,88888889	0,82758621	0,85714286	29
avg / total	0,87389771	0,87301587	0,87256236	63

KNN vinter	precision	recall	f1-score	support
Hogst	0,78787879	0,76470588	0,7761194	34
Skog	0,73333333	0,75862069	0,74576271	29
avg / total	0,76277056	0,76190476	0,76214569	63

RFC vinter	precision	recall	f1-score	support
Hogst	0,68292683	0,82352941	0,74666667	34
Skog	0,72727273	0,55172414	0,62745098	29
avg / total	0,70334002	0,6984127	0,6917896	63

Tabell A.2: Klassifiseringsrapport datasett to.

SVM VH	precision	recall	f1-score	support
Hogst	0,83333333	0,81081081	0,82191781	37
Skog	0,74074074	0,76923077	0,75471698	26
avg / total	0,79512052	0,79365079	0,79418413	63

KNN VH	precision	recall	f1-score	support
Hogst	0,70731707	0,78378378	0,74358974	37
Skog	0,63636364	0,53846154	0,58333333	26
avg / total	0,6780347	0,68253968	0,67745218	63

RFC VH	precision	recall	f1-score	support
Hogst	0,63265306	0,83783784	0,72093023	37
Skog	0,57142857	0,30769231	0,4	26
avg / total	0,60738581	0,61904762	0,58848283	63

Tabell A.3: Klassifiseringsrapport datasett tre.

A.3 Pythonkode

```
1 import pandas as pd
2 import numpy as np
3 import sklearn as sk
4 import geopandas as gp
5 import os
6 import rasterio
7 import seaborn as sns
8 import matplotlib.pyplot as plt
9
10 from sklearn.metrics import
    precision_recall_fscore_support
11 from rasterstats import zonal_stats
12 from rasterio.mask import mask
13 from sklearn.model_selection import train_test_split
14 from sklearn.ensemble import RandomForestClassifier
15 from sklearn.neighbors import KNeighborsClassifier
16 from sklearn.metrics import classification_report,
    confusion_matrix
17 from sklearn import svm
18 from sklearn.model_selection import GridSearchCV
19 from sklearn.metrics import mean_squared_error
20
21
22 class import_data():
23     def __init__(self):
24         """
25         Legger inn globale funksjoner
26         """
27         self.geoTiff = []
28         self.metadata = []
29         self.name = []
30         self.csv_file = ()
31         self.geoTiff_flat = []
32         self.bla = 0
33
34     def sort_data(self, path_img, path_shp, export):
35         """
36         Importerer bilde og shape filer,
37         klipper til bildene og sorterer metadata
38         og billedata
39         """
40         aoi_df = gp.read_file(path_shp)
41         aoi = aoi_df.geometry
42         self.geoTiff = []
```

```

42         self.metadata = []
43         self.name = []
44
45         for fname in os.listdir(path_img):
46             self.name.append(fname) #Liste
med filnavn
47             im = rasterio.open(os.path.join(
path_img, fname)) #Åpner tif filene
48             arr,transform = mask(im, aoi,
all_touched = True, crop = True) #Klipper ned tif
filene til en shape polygonfil for området
49             imarray = np.array(arr) #Legger
den nye tif filen inn i en midlertidlig variabel #0
for å velge riktig bilde
50             self.geoTiff.append(imarray) #
Legger tif filene inn i en liste
51
52             meta = im.profile #lagrer
metadata fra originale tif filen i en midlertidlig
variabel
53             meta["transform"] = transform #
Oppdaterer verdier som blir endret ved klippingen
54             meta["width"] = arr.shape[2]
55             meta["height"] = arr.shape[1]
56             self.metadata.append(meta) #
Legger oppdatert metadata i en liste
57
58             for i in range(len(self.geoTiff
)):
59                 with rasterio.open(
60                     export + self.name[i
], 'w',
61                     **self.metadata[i])
as dst:
62                     dst.write(self.geoTiff[i
][0, :, :], 1)
63
64             return[self.metadata, self.geoTiff, self.
name]
65
66
67
68     def creat_test_data(self, path_shp_test, export):
69         """

```

```

70         Lage treningsdata basert på shape filer og
        bildefiler fra sort_data
71         """
72         test_df_read = gp.read_file(path_shp_test) #
        Leser shape file med treningspolygoner
73         test_df = test_df_read[['type', 'geometry'
        ]] #Legger spesifikk data fra shape filen i en ny
        variabel
74         out_tbl = pd.DataFrame(data=test_df['type'
        ], columns=['type']) #Gir kolonne navn til ny
        dataframe som polygon statistikk legges inn i
75
76         for i in range(len(self.geoTiff)):
77             result = pd.DataFrame(zonal_stats(
        test_df['geometry'], self.geoTiff[i][0, :, :], stats
        =['median'], #Her kan en bytte mellom median mean
        ech.
78                                     affine
        =self.metadata[i]['transform'], nodata=-999)) #
        Kalkulerer statistikken for polygonene
79             out_tbl[[self.name[i] + '_median']] =
        result[0:][["median"]] #Legger polygondata inn i
        out_tbl dataframen.
80             # #Eksporterer csv tabell med mean og median
        statistikk på alle polygonene i hver av tif filene
81             out_tbl.to_csv(export)
82             self.csv_file = out_tbl.to_csv
83
84         def sort_x_y(self, path_csv):
85             """
86             Importerer CSV filen med data og deler den
        inn i X og Y
87             """
88             df_data = pd.read_csv(path_csv)
89             df_train = df_data.drop(['Unnamed: 0'], axis
        = 1)
90
91             X = df_train.drop(['type'], axis=1)
92             Y = df_train[['type']]
93
94             return X, Y
95
96         def flatten_tif(self, tiff):
97             """

```

```
98         Tar bildene klippet ned til subset og flater  
          de ut til en rekke til bruk i predikering  
99         """  
100        tiff_flat = []  
101        geoTiff_import = np.stack(tiff)  
102        for i in range(len(geoTiff_import)):  
103            tiff_flat.append(geoTiff_import[i].  
flatten())  
104  
105        tiff_flat = np.stack(tiff_flat)  
106        tiff_flat = np.moveaxis(tiff_flat, 0, 1)  
107  
108        return tiff_flat  
109  
110    def split_data(self, X, Y):  
111        """  
112        Splitter X og Y data inn i test- og  
113        treningsdata  
114        """  
114        X_train, X_test, Y_train, Y_test =  
train_test_split(X, Y, test_size=0.30, random_state=  
5)  
115        Y_train = Y_train['type'].ravel()  
116  
117        return X_train, X_test, Y_train, Y_test  
118  
119    def random_forest(self, X_train, X_test, Y_train  
, Y_test, tiff_flat, navn):  
120        """  
121        Random forest med gridsearch.  
122        Beste parameterene visualiseres, så legges  
123        de til en rfc som predikerer og trener på samme data  
124        som gridsearch  
125        """  
124        rfc2 = RandomForestClassifier(random_state=1  
)  
125  
126        n_estimators = [100, 300, 500, 400, 700, 800  
]  
127        max_depth = [5, 10, 20]  
128        min_samples_split = [2, 3, 4]  
129        min_samples_leaf = [1, 5, 7]  
130  
131        hyper = dict(n_estimators=n_estimators,
```

```
131 max_depth=max_depth,
132         min_samples_split=
    min_samples_split,
133         min_samples_leaf=
    min_samples_leaf)
134
135     grid = GridSearchCV(rfc2, hyper, cv=5,
    verbose=1, n_jobs=-1)
136     best_model = grid.fit(X_train, Y_train)
137
138     print('Best n_estimators:', best_model.
    best_estimator_.get_params()['n_estimators'])
139     print('Best max_depth:', best_model.
    best_estimator_.get_params()['max_depth'])
140     print('Best min_samples_split:', best_model.
    best_estimator_.get_params()['min_samples_split'])
141     print('Best min_samples_leaf:', best_model.
    best_estimator_.get_params()['min_samples_leaf'])
142
143     rfc = RandomForestClassifier(n_estimators =
    best_model.best_estimator_.get_params()['
    n_estimators'],
144                                 max_depth =
    best_model.best_estimator_.get_params()['max_depth'
    ],
145     min_samples_split = best_model.best_estimator_.
    get_params()['min_samples_split'],
146     min_samples_leaf = best_model.best_estimator_.
    get_params()['min_samples_leaf'])
147     rfc.fit(X_train, Y_train)
148
149     print(rfc.score(X_test, Y_test))
150
151     Y_pred = rfc.predict(X_test)
152     data.store_classification(Y_pred, Y_test,
    navn)
153
154     print(rfc.score(X_test, Y_test))
155     print("Prosseserer Tiff")
156     pred_tiff = rfc.predict(tiff_flat)
157     proba_tiff = rfc.predict_proba(tiff_flat)
158     print("Ferdig")
```

```
159
160     return proba_tiff, pred_tiff, Y_pred
161
162
163     def Nearest_neighbour(self, X_train, X_test,
164     Y_train, Y_test, tiff_flat, navn):
165         """
166         Samme som gjort i RFC, bare med KNN
167         """
168         leaf_size = list(range(1, 50))
169         n_neighbors = list(range(1, 30))
170         p = [1, 2]
171
172         hyperparameters = dict(leaf_size=leaf_size,
173     n_neighbors=n_neighbors, p=p)
174         knn = KNeighborsClassifier()
175         grid = GridSearchCV(knn, hyperparameters, cv
176     =5, n_jobs=-1)
177         best_model = grid.fit(X_train, Y_train)
178
179         Y_pred = grid.predict(X_test)
180         data.store_classification(Y_pred, Y_test,
181     navn)
182
183         print('Best leaf_size:', best_model.
184     best_estimator_.get_params()['leaf_size'])
185         print('Best p:', best_model.best_estimator_.
186     get_params()['p'])
187         print('Best n_neighbors:', best_model.
188     best_estimator_.get_params()['n_neighbors'])
189
190         print(grid.score(X_test, Y_test))
191         print("Prosseserer Tiff")
192         pred_tiff_knn = grid.predict(tiff_flat)
193         proba_tiff_knn = grid.predict_proba(
194     tiff_flat)
195         print("Ferdig")
196
197         return proba_tiff_knn, pred_tiff_knn, Y_pred
198
199     def svm(self, X_train, X_test, Y_train, Y_test,
200     tiff_flat, navn):
201         """
```



```
194         Kjører gridsearch, men predikerer på grid  
195         modellen  
196         """  
197         C = [0.1, 1, 10, 100, 1000]  
198         gamma = [1, 0.1, 0.01, 0.001, 0.0001]  
199         kernel = ['rbf']  
200         hyperparameters = dict(C = C, gamma = gamma  
201         , kernel = kernel)  
202         grid = GridSearchCV(svm.SVC(),  
203         hyperparameters, cv = 5, verbose=3)  
204         grid.fit(X_train, Y_train)  
205         # print best parameter after tuning  
206         print(grid.best_params_)  
207         # print how our model looks after hyper-  
208         parameter tuning  
209         print(grid.best_estimator_)  
210         Y_pred = grid.predict(X_test)  
211         data.store_classification(Y_pred, Y_test,  
212         navn)  
213         print(grid.score(X_test, Y_test))  
214         print("Prosseserer Tiff")  
215         pred_tiff = grid.predict(tiff_flat)  
216         print("Ferdig")  
217  
218         return pred_tiff, Y_pred  
219  
220  
221     def img_sort(self, tiff, export_dir, img,  
222     filename, metadata_clip):  
223         """  
224         Reorganiserer bildet predikert fra  
225         maskinlæringene, bruker dimensjonen fra en av de  
226         andre bildene før bildeklassifisering  
227         """  
228         Img_height = []  
229         Img_width = []  
230         Class_map = []  
231         Img_height = tiff[0].shape[1]  
232         Img_width = tiff[0].shape[2]
```

```
230         Class_map = np.zeros(tiff[0].shape)
231
232         for i in range(Img_height):
233             Start_intervall = i * Img_width
234             Slutt_intervall = i * Img_width +
                Img_width
235             Class_map[0, i, :] = img[Start_intervall
                :Slutt_intervall]
236
237         Class_map = Class_map.astype(dtype='float32'
                )
238         with rasterio.open(export_dir + '/' +
                filename + '.tif', 'w', **metadata_clip[0]) as dst:
239             dst.write(Class_map)
240
241     def pandas_classification_report(self, y_true,
                y_pred):
242         """
243         Lager klassifiseringsrapport
244         """
245         metrics_summary =
                precision_recall_fscore_support(
246             y_true=y_true,
247             y_pred=y_pred)
248
249         avg = list(precision_recall_fscore_support(
250             y_true=y_true,
251             y_pred=y_pred,
252             average='weighted'))
253
254         metrics_sum_index = ['precision', 'recall',
                'f1-score', 'support']
255         class_report_df = pd.DataFrame(
256             list(metrics_summary),
257             index=metrics_sum_index)
258
259         support = class_report_df.loc['support']
260         total = support.sum()
261         avg[-1] = total
262
263         class_report_df['avg / total'] = avg
264
265         return class_report_df.T
266
```

```
267     def store_classification(self, Y_pred, Y_test,
268                             name):
269         """
270         Lager og lagrer forvirringsmatriser fra
271         maskinl ringene
272         """
273         cf_matrix = confusion_matrix(Y_test, Y_pred)
274         print(cf_matrix)
275         labels = ['True Neg', 'False Pos', 'False
276                 Neg', 'True Pos']
277         categories = ['Hogst', 'Skog']
278         fig = make_confusion_matrix(cf_matrix,
279                                   group_names=labels,
280                                   categories=categories,
281                                   title = name,
282                                   percent=False)
283         print(fig)
284         plt.savefig('C:/Users/Bruker/Documents/
285                 Master/Data_maskinlaering/Export/fig/' + name + '.
286                 png')
287         print(classification_report(Y_test, Y_pred))
288         df_class_report = data.
289         pandas_classification_report(Y_test, Y_pred)
290         df_class_report.to_csv('C:/Users/Bruker/
291                 Documents/Master/Data_maskinlaering/Export/table/' +
292                 name + '.csv', sep=',')
293         np.savetxt('C:/Users/Bruker/Documents/Master
294                 /Data_maskinlaering/Export/pred/' + name + '.csv',
295                 Y_pred, delimiter=',')
296         #Kilde: https://stackoverflow.com/questions/39662398/scikit-learn-output-metrics-classification-report-into-csv-tab-delimited-format
```

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 import seaborn as sns
4
5 def make_confusion_matrix(cf,
6                           group_names=None,
7                           categories='auto',
8                           count=True,
9                           percent=True,
10                          cbar=True,
11                          xyticks=True,
12                          xyplotlabels=True,
13                          sum_stats=True,
14                          figsize=None,
15                          cmap='Blues',
16                          title=None):
17     '''
18     This function will make a pretty plot of an
19     sklearn Confusion Matrix cm using a Seaborn heatmap
20     visualization.
21
22     Arguments
23     -----
24     cf:          confusion matrix to be passed in
25
26     group_names: List of strings that represent the
27                  labels row by row to be shown in each square.
28
29     categories:  List of strings containing the
30                  categories to be displayed on the x,y axis. Default
31                  is 'auto'
32
33     count:       If True, show the raw number in
34                  the confusion matrix. Default is True.
35
36     normalize:   If True, show the proportions for
37                  each category. Default is True.
38
39     cbar:        If True, show the color bar. The
40                  cbar values are based off the values in the confusion
41                  matrix.
42
43                  Default is True.
44
45     xyticks:     If True, show x and y ticks.
```

```
35 Default is True.
36
37     xyplotlabels: If True, show 'True Label' and ' Predicted Label' on the figure. Default is True.
38
39     sum_stats: If True, display summary statistics below the figure. Default is True.
40
41     figsize: Tuple representing the figure size . Default will be the matplotlib rcParams value.
42
43     cmap: Colormap of the values displayed from matplotlib.pyplot.cm. Default is 'Blues' See http://matplotlib.org/examples /color/colormaps_reference.html
44
45     title: Title for the heatmap. Default is None.
46
47     '''
48
49
50
51     # CODE TO GENERATE TEXT INSIDE EACH SQUARE
52     blanks = ['' for i in range(cf.size)]
53
54     if group_names and len(group_names)==cf.size:
55         group_labels = ["{}\n".format(value) for
value in group_names]
56     else:
57         group_labels = blanks
58
59     if count:
60         group_counts = ["{0:0.0f}\n".format(value)
for value in cf.flatten()]
61     else:
62         group_counts = blanks
63
64     if percent:
65         group_percentages = ["{0:.2%}".format(value)
for value in cf.flatten()/np.sum(cf)]
66     else:
67         group_percentages = blanks
68
69     box_labels = [f"{v1}{v2}{v3}".strip() for v1, v2
```

```
69 , v3 in zip(group_labels,group_counts,
group_percentages)]
70     box_labels = np.asarray(box_labels).reshape(cf.
shape[0],cf.shape[1])
71
72
73     # CODE TO GENERATE SUMMARY STATISTICS & TEXT FOR
SUMMARY STATS
74     if sum_stats:
75         #Accuracy is sum of diagonal divided by
total observations
76         accuracy = np.trace(cf) / float(np.sum(cf))
77
78         #if it is a binary confusion matrix, show
some more stats
79         if len(cf)==2:
80             #Metrics for Binary Confusion Matrices
81             precision = cf[1,1] / sum(cf[:,1])
82             recall    = cf[1,1] / sum(cf[1,:])
83             f1_score  = 2*precision*recall / (
precision + recall)
84             stats_text = "\n\nAccuracy={:0.3f}\n
Precision={:0.3f}\nRecall={:0.3f}\nF1 Score={:0.3f}"
            .format(
85                 accuracy,precision,recall,f1_score)
86         else:
87             stats_text = "\n\nAccuracy={:0.3f}".
format(accuracy)
88         else:
89             stats_text = ""
90
91
92     # SET FIGURE PARAMETERS ACCORDING TO OTHER
ARGUMENTS
93     if figsize==None:
94         #Get default figure size if not set
95         figsize = plt.rcParams.get('figure.figsize')
96
97     if xyticks==False:
98         #Do not show categories if xyticks is False
99         categories=False
100
101
102     # MAKE THE HEATMAP VISUALIZATION
```

```
103     plt.figure(figsize=figsize)
104     sns.heatmap(cf,annot=box_labels,fmt="",cmap=cmap
    ,cbar=cbar,xticklabels=categories,yticklabels=
    categories)
105
106     if xyplotlabels:
107         plt.ylabel('True label')
108         plt.xlabel('Predicted label' + stats_text)
109     else:
110         plt.xlabel(stats_text)
111
112     if title:
113         plt.title(title)
```



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

Postboks 5003
NO-1432 Ås
Norway