ASSESSMENT OF RANDOM FOREST METHOD IN PIXEL-BASED SNOW COVER CLASSIFICATION IN ALPINE REGION, TATRA MOUNTAINS AND KAÇKAR MOUNTAINS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

CANSU AKSU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
GEODETIC AND GEOGRAPHIC INFORMATION TECHNOLOGIES

NOVEMBER 2022

Approval of the thesis:

ASSESSMENT OF RANDOM FOREST METHOD IN PIXEL-BASED SNOW COVER CLASSIFICATION IN ALPINE REGION, TATRA MOUNTAINS AND KAÇKAR MOUNTAINS

submitted by **CANSU AKSU** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Geodetic and Geographic Information Technologies, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Sevda Zuhal Akyürek
Head of the Department, **Geodetic and Geographic Information Technologies** _____

Prof. Dr. Sevda Zuhal Akyürek
Supervisor, **Civil Engineering, METU** _____

Asst. Prof. Semih Kuter
Co-Supervisor, **Forest Engineering, Çankırı Karatekin University** _____

Examining Committee Members:

Prof. Dr. Lütfi Süzen
Geological Eng., METU _____

Prof. Dr. Sevda Zuhal Akyürek
Civil Eng., METU _____

Prof. Dr. İsmail Yücel
Civil Eng., METU _____

Asst. Prof. Dr. Semih Kuter
Forest Eng., Çankırı Karatekin Uni. _____

Asst. Prof. Dr. Gökçen Uysal
Civil Eng., Eskişehir Technical Uni. _____

Date: 21.11.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last name : Cansu Aksu

Signature :

ABSTRACT

ASSESSMENT OF RANDOM FOREST METHOD IN PIXEL-BASED SNOW COVER CLASSIFICATION IN EUROPEAN ALPS, TATRA MOUNTAINS AND KAÇKAR MOUNTAINS

Aksu, Cansu
Master of Science, Geodetic and Geographic Information Technologies
Supervisor: Prof. Dr. Sevda Zuhal Akyürek
Co-Supervisor: Asst. Prof. Dr. Semih Kuter

November 2022, 234 pages

For most countries in the Northern Hemisphere, the amount of usable water throughout the year is roughly determined by the amount of snow. Climate change and increasing demand on drinking and industrial water due to population growth make the monitoring of snow cover even more crucial than it was in the past. Today, to observe the amount of snow cover, different algorithms are being used on remote sensing data for classification of snow, aside from in-situ data collection techniques. This study presents the evaluation of the performance of Random Forest (RF) algorithm for snow cover classification on Sentinel-2 imagery over three selected mountainous regions: Alpine Region, Tatra Mountains, and Kaçkar Mountains, with different input combinations as independent variables (i.e., predictors). The combinations have been evaluated for three different times of the year for a much better assessment – to observe the differences in when snow cover starts to form (November to December), the time with roughly the maximum amount of snow is observed (January to March), and the time when it starts to melt (April to June). The confusion matrices, overall accuracy (OA) and Kappa coefficient were used for

accuracy assessment. Overall, principle component bands combination (*Pca*) yielded the most accurate results. *Pca* combination also provided the shortest computation time out of all combinations, excluding the process of obtaining principal components, as the combination has the least amounts of input to the RF model, as compared to the other combinations. The overall results revealed that RF algorithm works well with appropriate numbers of principal component bands with NDSI, NDVI and NDWI indices for complex terrains over mountainous areas.

Keywords: Sentinel-2, Remote Sensing of Snow, Classification, Machine Learning, Snow Hydrology, Random Forest

ÖZ

AVRUPA ALPLERİ, TATRA DAĞLARI VE KAÇKAR DAĞLARINDA PİKSEL TABANLI KAR ÖRTÜSÜ SINIFLANDIRMASINDA RANDOM FOREST METODUNUN DEĞERLENDİRİLMESİ

Aksu, Cansu
Yüksek Lisans, Jeodezi ve Coğrafi Bilgi Teknolojileri
Tez Yöneticisi: Prof. Dr. Sevda Zuhal Akyürek
Ortak Tez Yöneticisi: Dr. Öğr. Üy. Semih Kuter

Kasım 2022, 234 sayfa

Kar miktarı, Kuzey yarımküredeki çoğu ülke için yıl boyunca kullanılabilir su miktarını kabaca belirler. İklim değişikliğinin yanı sıra içme ve sanayi suyuna olan talebin sürekli artmasına sebep olan küresel nüfus artışı, kar örtüsünü izlemeyi geçmişte olduğundan daha önemli hale getirmektedir. Günümüzde kar örtüsü miktarını gözlemlemek için, yerinde veri elde etme tekniklerinin yanı sıra, kar sınıflandırması için uzaktan algılama verileri üzerinde farklı algoritmalar kullanılmaktadır. Bu çalışma, seçilen üç dağlık bölgede (Alpler Bölgesi, Tatra Dağları ve Kaçkar Dağları) uzaktan algılama verileri (Sentinel-2) üzerinde kar örtüsü sınıflandırması için Random Forest (RF) algoritmasının bağımsız değişkenlerle oluşturulmuş farklı girdi kombinasyonları ile performansını değerlendirmesini sunmaktadır. Girdi kombinasyonlarının daha iyi değerlendirilmesi için yılın üç farklı zamanı baz alınmıştır - kar örtüsünün oluşmaya başladığı zaman (Kasım-Aralık), kabaca en fazla karın gözlemlendiği zaman (Ocak'tan Mart'a kadar) ve karın erimeye başladığı zaman (Nisan'dan Haziran'a kadar). Değerlendirme sonuçları için hata matrisleri ile genel doğruluk ve Kappa katsayısı kullanılmıştır. Genel olarak, atmosferik ve topografik düzeltme yapılan bantlardan elde edilen ilk üç temel bileşen

bantı ile NDSI, NDVI ve NDWI kombinasyonunun (*Pca*) en doğru sonuçları verdiği gözlemlenmiştir. *Pca* yönteminin ayrıca, temel bileşenleri elde etme süreci hariç, tüm kombinasyonlar arasında en kısa hesaplama süresine sahip olduğu tespit edilmiştir. Bu kombinasyon aynı zamanda RF modeli için, diğer kombinasyonlara kıyasla, en az miktarda girdi değişkenine sahiptir. Elde edilen bu sonuçlar, RF'in, dağlık bölgelerdeki karmaşık arazi yapıları söz konusu olduğunda, NDSI, NDVI ve NDWI indisleri ile uygun sayıda temel bileşen girdisi ile iyi şekilde çalıştığını göstermektedir.

Anahtar Kelimeler: Sentinel-2, Karın Uzaktan Algılaması, Sınıflandırma, Makine Öğrenmesi, Kar Hidrolojisi, Random Forest

To my family

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my dear advisor Prof. Dr. Zuhal Akyürek and my dear co-advisor, Asst. Prof. Dr. Semih Kuter, for their continuous support and understanding throughout a very challenging time in my life. I feel very lucky that I have received their valuable contributions for this study.

I would also like to thank the examining committee, Prof. Dr. Lütfi Süzen, Prof. Dr. İsmail Yücel, and Asst. Prof. Dr. Gökçen Uysal, for their valuable advice and comments.

I would like to give my sincere thanks to Berkay Akpınar for his invaluable contributions to this thesis.

I would like to give thanks to my three greatest supporters, my mother, Gönül Aksu my father Ahmet Aksu and my dear little sister, Hazal Aksu. Without you, this thesis would not be possible.

I owe thanks to İdil Dönderici for her friendship and support since our childhood years, and her motivating presence during this process.

I would like to, finally and especially, thank Buğra Fıratlı, for his unconditional love and encouragement throughout this thesis. You have been with me for everything, and I am eternally grateful.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	
AOT	Aerosol Optical Thickness
AVHRR	Advanced Very High-Resolution Radiometer
BOA	Bottom-of-Atmosphere
BRDF	Bidirectional reflectance distribution function
DEM	Digital elevation model
DMSP	Defense Meteorological Satellite Program
DT	Decision trees
DW	Dynamic World
ESA	European Space Agency
fSCA	Fractional snow covered area
GIS	Goegraphical information systems
GOES	Geostationary Operational Environmental Satellite
MODIS	Moderate Resolution Imaging Spectroradiometer
NDSI	Normalized Difference Snow Index
NDVI	Normalized Difference Vegetation Index,
NDWI	Normalized Difference Water Index
NIR	Near infrared band
NOAA	National Oceanic and Atmospheric Association

OA	Overall Accuracy
OOB	Out-of-bag
PCA	Principal component analysis
SAR	Synthetic aperture radar
SCA	Snow covered area
SCL	Scene classification layer
SNAP	Sentinel Application Platform
SSM/I	Special Sensor Microwave Imagers
SVM	Support Vector Machine
SVR	Support vector regression
SWIR	Short-wave infrared band
RF	Random Forest
TERRA	Earth Observation System-Terra
TOA	Top-of-Atmosphere
VGT	SPOT-Vegetation
VIS	Visible bands
WV	Water Vapor

CHAPTER 1

INTRODUCTION

The amount of snow cover is crucial for humans and all life forms in the Northern Hemisphere, which has 98% of the total snow cover on Earth (Wang et al., 2018). The runoff stemming from snowmelt is deemed as main water resource (Mankin et al., 2015), especially in high altitude locations of North America, Europe and Asia (Takala et al., 2011). The area covered with snow directly correlates with the amount of snow precipitation as well as the temperature, which affects the melting rate of snow on the ground. As temperature and snow precipitation influence the snow cover amount greatly, it is accepted as a key weather and climate indicator (Robinson et al., 1993). Not only the snow precipitation amount is affected by the changes in climate, the amount of snow cover also influences the climate. There are several reasons for the local climate conditions to be affected by the amount of snow cover, to give specific examples: i) the surface albedo can rise with the availability of highly reflective fresh snow cover, ii) due to its high thermal emissivity, snow behaves like a thermal insulator, and iii) melting snow acts as a place that latent heat can submerge (Cohen & Rind, 1991; Armstrong & Brodzik, 2001). In addition to all these, the carbon balance, as well as the soil respiration along with the thickness of the active layer (thawing seasonally) of the soil in permafrost locations is a direct consequence of the snow season length and the period of time which snowmelt occurs (Grogan & Jonasson, 2006; Takala et al., 2011). Thus, snow cover information contributes to the research of hydrological, climatological and greenhouse gas cycle in the mid and upper latitudes of the Northern Hemisphare (Takala et al., 2011; Tekeli et al., 2005).

A study by Wang et.al. (2018) indicates that there is a downward trend of snow cover area in Northern Hemisphere in recent years, which is in line with the results received from other studies, investigated in various time frames (Bormann et al., 2018; Brown, 2000; Brown & Robinson, 2011;McCabe & Wolock, 2010; Kunkel et al.,

2016; Armstrong & Brodzik, 2001). Since the characteristics of snow are significant for investigating climate conditions and climate change, as well as determining the maximum amount of total runoff (Jonas et al., 2009), it is important to monitor the extent of snow accurately and continuously (Takala et al., 2011; Robinson et al., 1993). Such monitoring is conducted by satellites as well as in situ station networks. Monitoring of snow by remote sensing has been increasing, and seen as a necessity for wide spatial extent of the snow (Robinson et al., 1993). Snow data obtained from existing stations within the area of interest, especially for locations with complex terrain, are usually limited (Basang et al., 2017).

Snow cover monitoring with satellite based approaches have been adopted for almost 60 years (Akyürek & Şorman, 2002; Tekeli et al., 2005). Today, snow cover maps can be achieved with adequate temporal resolution, even daily, with multispectral sensors like Moderate Resolution Imaging Spectroradiometer (MODIS) (Parajka & Blöschl, 2006), and sensors like National Oceanic and Atmospheric Association (NOAA) Advanced Very High-Resolution Radiometer (AVHRR) (Liu et al., 2013), and with adequate spatial resolution, with satellites such as Landsat and Sentinel (Gascoin et al., 2019).

Snow covered area (SCA), which has been acknowledged as a crucial hydrological parameter, has been used for predicting streamflow as early as 1940s with aerial photography, and in the beginning stages of using satellite data, by establishing classification algorithms (Maurer et al. 2003). There are two different features that can be derived from optical remote sensing data, that is denoted as SCA, which are (i) binary snow cover maps (snow/nosnow) (Hall et al., 1995; Hall et al., 2002; Maurer et al., 2003), and (ii) fractional snow-covered area (fSCA) (subpixel snow cover mapping) (Metsämäki et al., 2005; Painter et al., 2003). In this study, a version of binary snow cover (no-snow cover is further classified) is implemented. Nowadays, both of these classifications have been obtained by machine learning algorithms (Czyzowska-Wisniewski et al., 2015; Dobreva & Klein, 2011; Kuter

2021; Kuter et al., 2018; Kuter et al., 2022; Liang et al., 2017; Liu et al., 2020a; Moosavi et al., 2014; Zhu et al., 2012).

Features of snow other than SCA and fSCA are also obtained by processing satellite data, such as snow depth, and snow water equivalent, generally retrieved from microwave data (Foster et al., 2005; Tait, 1998; Liu et al., 2013), which are used to ascertain the quantity of runoff in mountainous regions (Jonas et al., 2009). Snow grain size and snow albedo (i.e., reflected solar radiation through snow which correlates to surface temperature, which, in turn, directly affects the climate) can also be acquired by means of remote sensing (Painter et al. 2003; Painter et al. 2009; Guo & Yang, 2022).

The visible part of the electromagnetic spectrum is generally employed for snow cover detection since the reflection of snow is high in visible wavelenghts (Dietz et al., 2012; Tekeli et al., 2005), yet there are some limitations while only relying on the visible bands. Clouds and low brightness (shadows) can restrict or even block the snow cover data within the satellite imagery (Robinson et al., 1993). The dense forests in the area of interest can mask the snow from the view of the satellite, leading to underestimation of the actual snow cover extent (Robinson et al., 1993; Bitner et al., 2002). The solutions that have been offered in literature to mitigate these problems are explored in detail in Section 2.2.

Fast technological developments in the applications of collection and recording of data for the last two decades have influenced the start of the period of "big digital data", which led to studies on forming and finding competent and desirable algorithms of data mining, for obtaining relevant and convenient information (Çevik et al. 2017; Chowdhury et al. 2022; Tsai et al. 2015). In parallel to these developments, data-driven machine learning algorihtms have been evolved and improved in order to manage this big digital data (Kuter et al. 2022; Tsai et al. 2015). Remote sensing applications were not an exception, as machine learning algorithms have begun to be widely used with satellite data as well (Maxwell et al., 2018) since

the 60s (Holloway & Mengersen, 2018). One of the best indicators of the extensive approval of machine learning approaches in remote sensing applications is their usage with operational land-cover mapping, such as 2001 National Land-Cover Database (NLCD), which is generated by decision trees (Homer et al. 2004; Maxwell et al., 2018). In recent years, the accuracy that has been obtained via machine learning algorithms have been yielding impressive results, especially compared to the traditional parametric classifiers (Talukdar et al., 2020; Maxwell et al., 2018; Kuter, 2021; Pal & Mather, 2005).

Snow cover classification through remote sensing data has also been achieved by machine learning algorithms, along with other methods (Tsai et al., 2019a). It has been reported in various studies that machine learning exceeds other common approaches those are used in snow cover mapping, such as statistical regression analysis and spectral unmixing, especially for RF and Support Vector Machine (SVM) algorithms (Hou et al., 2020; Liu et al., 2020a; Hou et al., 2021; Baba et al., 2020; Tsai et al., 2019a).

In this study, RF algorithm is used to classify snow cover, as well as clouds, water bodies, and land over three different mountainous regions, namely, European Alps, Tatra Mountains and Kaçkar Mountains from Sentinel-2 imagery by considering the snow's different morphological phases, i.e., 1) fresh snow, 2) fully-formed snow pack, and 3) melting snow. This thesis aims to investigate the accuracy of RF model with different input combinations (cf. Chapter 3) and mainly seek solutions to the following issues those are often encountered when mapping snow cover with optical remote sensing data:

- Shadows those are mostly prominent with the remote sensing data of mountainous regions, causing misclassification, and
- Clouds in satellite images those are often misidentified as snow.

The thesis starts with Chapter 1, which presents an introduction and the goal of the study. This Chapter also demonstrates the organizational schema of the whole thesis. Chapter 2 reviews the recent and relevant literature. The materials and methods that are used in the thesis are given in detail in Chapter 3. The selected study areas, the inputs and their various combinations, the approaches that are employed while obtaining the input data along with the results are described. In Chapter 4, the results are given and discussed. And finally, Chapter 5 concludes the thesis, summarizing the work that has been utilized with potential future directions and prospects.

CHAPTER 2

LITERATURE REVIEW

2.1 Background on Snow Hydrology

Study of snow hydrology considers the contributions of snow within hydrologic cycle, namely melting of snow, the movement of water from snowmelt within the snow, as well as snowmelt adding to runoff (Singh et al., 2011). The snow hydrology elements are snow cover, snow depth, snow water equivalent, snow stratigraphy, snow albedo, snow-soil interface (Rango, 1993). Figure 2.1 demonstrates the snow hydrology components and processes (Beria et al., 2018).

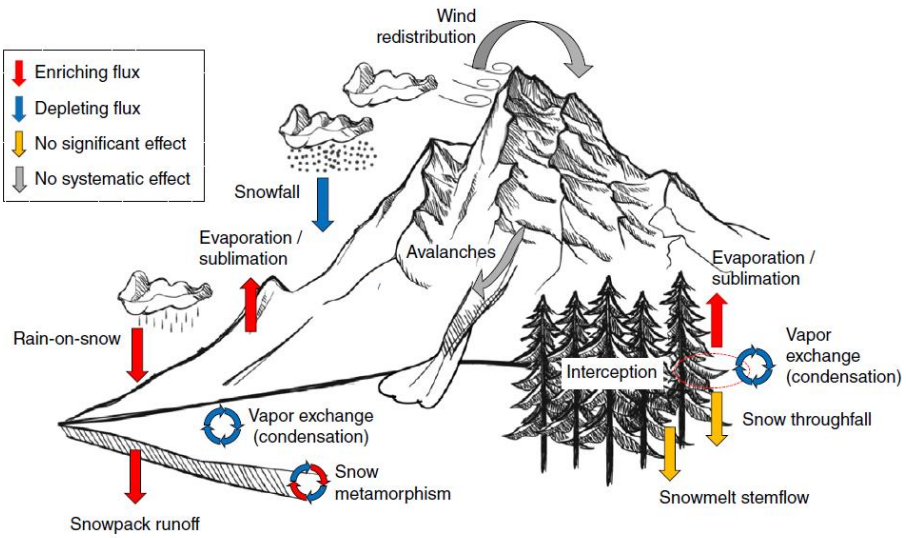


Figure 2.1. Snow hydrology components and processes (Beria et al., 2018)

When snow precipitation occurs, snow starts to accumulate, then redistribution and melt of the snow occur, leading to surface runoff (Beria et al., 2018). Snow accumulation causes snow grain size and shape to transform. This causes snow to subliminate, and its vapor to exchange with the atmosphere (Colbeck, 1982). Wind can redistribute the accumulated snow on the ground. Redistribution of snow can also be triggered by avalanches, which happen due to snowpacks being layered in a structurally weak formation. Wind-induced redistribution can cause snowflakes to shatter upon impacting the surface (Comola et al., 2017). This produces smaller ice fragments, as well as sublimination of the snow. Evaporation of snow is also more likely to occur with the presence of wind (Beria et al., 2018). When snowmelt occurs, it either contributes to surface runoff, infiltrates into the subsurface and contributes to groundwater or other processes in the subsurface, or it re-freezes. The trees also impact the snow cycle such that in forest areas, trees can block snow (Hedstrom & Pomeroy, 1998). The snow intercepted by tree canopy either subliminates or goes underground by snow throughfall of snowmelt stemflow.

In today's world, snow hydrology is deemed as a very important subject since snowmelt contribution to downstream is crucial for many regions (Li et al., 2021; Qi et al., 2022; Nollin, 2010). History of snow cover study was dominated by two subjects: water resources and avalanches. Although there were methodical examinations of snow before 1900s, there were few tools to conduct serious research. One of the earliest snow hydrology studies that was in 17th century by the natural scientist Antonio Vallisnieri. He speculated that the runoff from springs came from rain and snowmelt (Luzzini, 2011). Physical and quantitative observations of snow cover were more possible after year 1936, when International Glaciology Society was established, and many more international organizations and societies formed by snow scientists followed. Untill 1960s, many studies were conducted in detail to analyse snow's physical behaviour, as laboratories for snow research began to be established. Starting from 1970s, more advanced techniques and tools were involved in the research as computers started to become available (Colbeck, 1987), and remote sensing has significantly advanced the observation of snow cover (Rango, 1993).

2.2 Applications of Snow Cover Determination with Remote Sensing Implementations

The snow cover is identified by a few specific occurrences as well as phases that can be determined by satellite-based observations, which have significance in especially mountainous zones (i.e., Alpine Region) (Barry et al., 1995). In recent years, snow cover mapping, and detection of snow cover and its properties for hydrological models have been more frequently used along with traditional methods (i.e., snow pits, probing, ultrasonic snow depth sensors) (Dong, 2018; Schaffhauser et al., 2008). Since data collection by traditional methods, especially in higher elevations, both demands a large amount of time, not feasible, and the obtained data are a point assessment of snow which might not embody the properties of a whole area covered by snow (Snehmani et al., 2015).

Satellite detection of snow was first employed in eastern Canada in the year of 1960 by the TIROS-1 satellite (Akyürek & Şorman, 2002). Although remote sensing techniques were used to detect snow cover in 60s, it had not yet presented sufficient results (Colbeck, 1987). Landsat 1 was first launched to orbit in 1972 (Williams et al., 2006). NOAA-AVHRR, Geostationary Operational Environmental Satellite (GOES) series, and Special Sensor Microwave Imagers (SSM/I) of Defense Meteorological Satellite Program (DMSP) were started to be used in the late 60s and 70s (Hao et al., 2021; Foster et al., 2009; Barry et al., 1995; Armstrong & Brodzik, 2001). All of these instruments have provided a large archive of remote sensing data and extensively used for snow cover mapping (Hao et al., 2021; Rößler & Dietz, 2022; Dech et al., 2021; Akyürek & Şorman, 2002; Lucas & Harrison, 1990; Foster & Rango, 1982). SPOT-VEGETATION (VGT) has been operational since 1998 and it has aided in snow detection of numerous studies as well (Delbart et al., 2006; Dankers & De Jong, 2004). By Terra satellite, which was launched on December 18, 1999, the MODIS started compiling data on February 24, 2000 (Klein, 2003), and since then, it has been used for monitoring of snow cover (Tekeli et al., 2005). Due to the technical limitations of Landsat and NOAA-AVHHR, MODIS - Earth

Observation System-Terra (TERRA) became more preferable. Landsat does not provide daily data, as its temporal resolution is 16 days. Although daily data is available for AVHRR, it requires processing for the classification of snow from other characteristics. Additionally, the geolocation accuracy of MODIS- TERRA is an upgrade over NOAA-AVHRR's (Tekeli et al., 2005).

Sentinel-2A was launched in June 2015, which makes the Sentinel-2 constellation a relatively new source of remote sensing data (Languille et al., 2015). Sentinel-2 grants a higher temporal resolution (5 days with twin satellites, 2A and 2B) as compared to Landsat, and it has high spatial resolution of 10, 20 and 60 meters (Gascon et al., 2009). Setinel-2 data have been widely used in remote sensing applications including snow cover mapping for its high temporal and spatial resolution, in addition being free of charge, i.e., Theia Snow Collection (Gascoin et al., 2019).

The methods and data for deriving snow cover from remote sensing applications have varied throughout the years. The high reflectance of snow in the visible bands and low reflectance in the infrared bands have led to the development of Normalized Difference Snow Index (NDSI), which can separate snow from other land covers (Dong, 2018; Bian et al., 2016). Microwave remote sensing data have also been broadly used to detect snow cover since 1970s, as its ability of identifying snow cover is not influenced by clouds or shadows (Dong, 2018); Barry et al., 1995). Another approach that is frequently used to determine snow cover is integrating the NDSI with threshold classification method (Luo et al., 2022). Spectral mixture analysis (Painter et al., 1998; Hao et al., 2019) and other thresholding methods (Luo et al., 2008) are also some of the common methods of distinguishing snow cover.

Another widely employed approach, nowadays, in snow cover mapping is the use of machine learning algorithms (Tsai et al., 2019b). It has been disclosed in many studies that machine learning classification algorithms surpass other methods of snow cover identification in remote sensing (Hou et al., 2020; Liu et al., 2020b; Hou

et al., 2021; Baba et al., 2020; Tsai et al., 2019a). The most extensively used machine learning algorithms are SVM (Forman & Reichle, 2015; Liu et al., 2020b; Nijhawan et al., 2018a), support vector regression (SVR) (Kuter, 2021; Xiao et al., 2018), decision trees (DT) (Balk & Elder, 2000; Gharaei-Manesh et al., 2016) and RF (Liu et al., 2020b; Nijhawan et al., 2018a; Kuter, 2021). Artificial neural networks (ANN) have also been employed for the purpose of classifying snow (Gharaei-Manesh et al., 2016).

2.2.1 Limitations and Proposed Solutions on Snow Cover Determination in Literature

Snow cover is mainly determined by the visible bands, as the reflection of snow is high in the visible portion of the electromagnetic spectrum, especially for fresh snow (Dietz et al., 2012), but there are some limitations with this approach. Cloud cover, as well as shadows casted by clouds and complex topography may put a limitation on the correct detection of snow cover from the visible bands (Snehmani et al., 2015; Dietz et al., 2012; Barry et al., 1995). Although differentiating cloud from other land cover types is not an issue, snow can show similar characteristics in the visible and thermal bands (Dietz et al., 2012). Cloud cover and shadow problem can be solved by numerous methods, for instance, by the use of satellite microwave data (Barry et al., 1995) and cloud/shadow masking algorithms from different sources (Skakun et al., 2022). Especially for clouds and mountainous regions with slope-casted shadows, microwave radiometers are great devices for deducing land cover, specifically for snow cover. Microwave emissivity of snow is highly reliant on the water content and freshness of the snow (Schanda et al., 1983). Synthetic aperture radar (SAR) can obtain high resolution data at either night or daytime, without the interference of cloud and mountain shadows, unlike other microwave sensors (Snehmani et al., 2015).

In addition to microwave data, other cloud and shadow masking techniques have also been established to overcome these issues (Skakun et al., 2022; Luo et al., 2008)

such as NDSI, which can differentiate snow from clouds, with the exception of ice clouds (Dong, 2018; Bian et al., 2016). Machine learning algorithms, which have already been extensively adopted in snow cover mapping due to their higher accuracy as compared to some other commonly used approaches (Hou et al., 2020; Liu et al., 2020; Hou et al., 2021; Baba et al., 2020; Tsai et al., 2019b), are also used to tackle these issues in various studies with varying but sufficient results (Skakun et al., 2022; Chai et al., 2019; Jeppesen et al., 2019; Gomez-Chova et al., 2017). In addition to the efforts on masking the clouds and shadows with machine learning and deep learning algorithms, there are also various existing masking algorithms that have been developed over the years, such as Fmask (Zhu et al., 2012), Sen2Cor (Main-Knorn et al., 2017) and MAJA (Rouquié et al., 2017) with sufficient labelling accuracy for cloud, as well as other land cover types (i.e., snow, vegetation, water etc.) (Zekoll et al., 2021; Baetens et al., 2019).

CHAPTER 3

MATERIALS AND METHODS

In this study, to test the capabilities of the RF algorithm, Sentinel-2 data from different mountainous regions are used and different input combinations have been formed for detection of four classes: Cloud, No-snow (Land), Snow and Water. The No-snow (Land) class is denoted as the class that contains every element that is not snow, not cloud or not water. It contains different land cover classes, as explained in detail in Section 3.1.1.

The Sentinel-2 data were obtained from Copernicus Open Access Hub (Copernicus Open Access Hub, n.d.; Tona & Boa, 2018).

The band designation of Sentinel-2 is shown in Table 3.1 (Main-Knorn et al., 2017). In this study, Band 2, 3, 4, 5, 6, 7, 8a, 11 and 12 have been used (mentioned as Band 2-12). Band 2, 3 and 4 been resampled to 20 meters of resolution for the purpose of this study with Sen2Cor (cf. Section 3.2.2).

Table 3.1. Spectral bands of Sentinel-2A (Main-Knorn et al., 2017)

Sentinel-2A Bands	*Original Resolution (m)*	*Central Wavelength (μm)*
Band 1	60	0.443
Band 2	10	0.490
Band 3	10	0.560
Band 4	10	0.665
Band 5	20	0.705
Band 6	20	0.740
Band 7	20	0.783
Band 8	10	0.842
Band 8a	20	0.865
Band 9	60	0.945
Band 10	60	1.375
Band 11	20	1.610
Band 12	20	2.190

The RF classification algorithm is applied to the selected spatial subsets on 9 Sentinel-2 images, each of which has an area of 20 × 20 km^2 (1001 × 1001 pixels) (i.e., there are 9 spatial subsets in total) with the following input combinations:

- The composition of resampled Sentinel bands (Band 2-Band 12) and NDSI, NDVI and NDWI, is named as "*Sc_only*",
- The composition of atmospherically and topographically corrected Sentinel bands (Band2-Band12) and NDSI, NDVI and NDWI, is named as "*Atmo_topo*",
- The composition of atmospherically and topographically corrected Sentinel bands (Band2-Band12), NDSI, NDVI and NDWI and digital elevation model (DEM) is named as "*Dem*"
- The composition of the first three principal components of principal component analysis (PCA) with NDSI, NDVI and NDWI, is named as "*Pca*".

The definitions of the compositions are presented in Table 3.2. The approaches to attain some of the inputs are described in the subsequent sections of this chapter. The spectral bands used in all above-mentioned input combinations have been atmospherically and topographically corrected, except for *Sc_only*.

Table 3.2. Main input combinations.

Combination name	Content of the Combination
Sc_only	Band 2-7, Band 8a, Band 11, Band 12, NDSI, NDVI and NDWI
Atmo_topo	Band 2-7, Band 8a, Band 11, Band 12, NDSI, NDVI and NDWI
Dem	Band 2-7, Band 8a, Band 11, Band 12, NDSI, NDVI and NDWI, elevation data
Pca	First three principal components from principal component analysis (PCA) (derived from Band 2-7, Band 8a, Band 11, Band 12), NDSI, NDVI and NDWI

The initial classification results are assessed with OA and Cohen's Kappa coefficient. Two additional input combinations are also applied to the images which produced the best and worst accuracy metrics (Table 3.3):

- The three principal components as well as slope and aspect data derived from DEM named as "*Pca_plus*", and
- The three principal components, slope and aspect data, as well as elevation data acquired from DEM named as "*Pca_plus2"*

Table 3.3. Additional input combinations

Combination name	Content of the Combination
Pca_plus	First three principal components (derived from Band 2-7, Band 8a, Band 11, Band 12), NDSI, NDVI and NDWI, aspect and slope data
Pca_plus2	First three principal components (derived from Band 2-7, Band 8a, Band 11, Band 12), NDSI, NDVI and NDWI, aspect, slope and elevation data

All of the combinations mentioned above are applied on all of the 9 Sentinel-2 images, with number of training data of 300 per class, and 1000 per class (i.e., 2 different training sample size for every image with every method). The process of selection of training/testing datasets is explained in detail in Section 3.5.2.

The *Pca* input combination is conducted on the full tile area (i.e., 100 × 100 km^2 of full tile area) of the 9 Sentinel-2 images, as *Pca* input combination generally yielded the best accuracy assessment results (cf. Chapter 0).

The methodology used in this study is summarized in the flowchart in Figure 3.1.

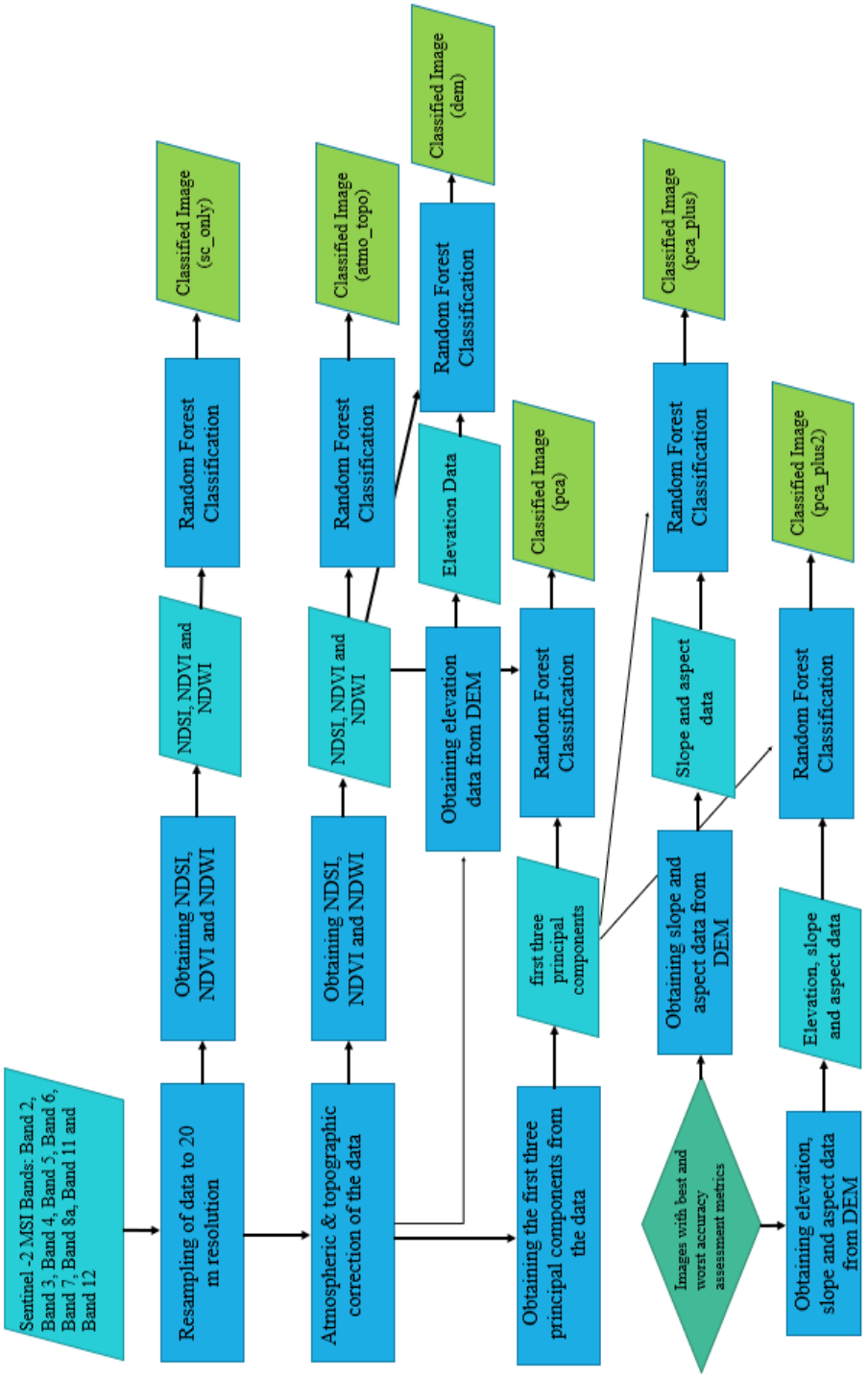


Figure 3.1. The flowchart of the methodology used in the study

The RF classification, preparation of training and test datasets, PCA and calculation of the normalized band indices have been conducted with Python codes (cf.

Appendix A, Appendix B), whereas attaining slope and aspect data, in addition to randomly sampled test data for accuracy assessment, have been done with the aid of ArcMap 10.7 software.

In the subsequent sections, the study areas, the selected satellite image datasets, the image preprocessing stages, PCA, the RF classification algorithm, the selection process of training and test datasets, and the employed statistical metrics for accuracy assessment are explained in detail.

3.1 Study Areas and Satellite Image Dataset

For this study, three different mountainous regions are selected: European Alps, Tatra Mountains and Kaçkar Mountains. For each region, single Sentinel-2 tile with an area of 100 × 100 km^2 is chosen.

The selected study areas can be seen in Figure 3.2.

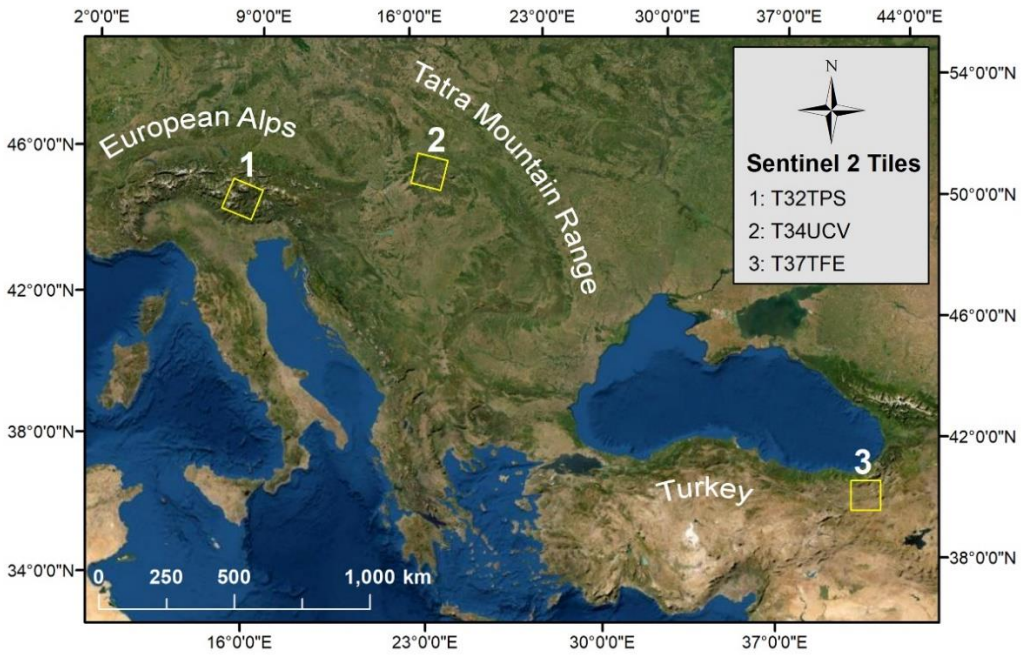


Figure 3.2. The locations of the selected Sentinel-2 tiles.

The three mountainous regions are chosen for this study to observe the differences in the behavior of RF algorithm in different climates, different snow durations and different snow depths. In Kuter et al. (2022), the effectiveness of EUTMETSAT H35 fractional snow cover area product is validated with different data sources. This study further investigates the weaknesses of the detection of snow cover area in these regions with different methods.

The Alpine Region comprises of eight countries in Europe: Austria, France, Germany, Italy, Liechtenstein, Monaco, Slovenia, and Switzerland. The chosen area is mostly at the northeast of Italy and is within the borders of Austria and Switzerland. The mean elevation of the tile from Alpine Region is calculated as 1695 m. The peak elevation of Alps, in general, exceeds 4800 m. The Alps are characterized by distinct climatic gradients, occurrences of extreme precipitation followed by hazardous events, and permanent snow and ice cover in higher altitudes of the mountain range (Gobiet et al., 2014).

The study of Valt and Cianfarra (2010) has analyzed the snow duration (number of snow days with snow depth > 1 cm) and accumulated snowfall with the data gathered from 18 stations located in the Italian Alps for 60 and 50 years, respectively (between the years of 1950 and 2009, and 1960 and 2009). Both analyses have been conducted for the entire snow season, which is from December to April, as well as for spring season (from March to April) and in the case of accumulated snowfall, for winter season (from December to February) separately. The analyses have also been done for two different elevation ranges, which are denoted as between 800 m to 1500 m and 1500 m and above. According to the results of the study, there has not been a significant change in mean snow duration (for decades). For 800-1500 m elevation, the mean duration is between 92-111 days, and for 1500 m elevation and above, the mean duration is between 140-148 days, for the entire snow season in Italian Alps. There is an observed variation; however, in the mean accumulated snowfall. For 800-1500 m elevation, the mean accumulated snowfall changes from 102 to 158 cm, and

for 1500 m elevation and above, the mean snowfall varies from 199 to 280 cm, for winter season in Italian Alps.

A study has determined the temperature changes in Eastern Alps at high and low elevations from 1975 to 2010 with the help of global radiation and remote sensing data (Tudoroiu et al., 2016). According to the study, there has been a warming trend in both higher and lower elevations over the years. The annual median temperature was reported from 6 ^{o}C to 8 ^{o}C in higher altitudes, and in between 10 to 12 ^{o}C in lower altitudes.

Tatra Mountains forge a natural border between Slovakia and Poland. The mean elevation of the tile located in Tatra Mountains is 706 m. Peak elevation of Tatra is 2655 m above sea level (Lapin et al., 2007). Tatra Mountains are the highest mountains within the Carpathian Range, consisting of subalpine and alpine zones. Snow season is usually depicted as October to June (Singh, 2013).

The study of Lapin et al. (2007) has observed the duration of snow as well as maximum snow depth for Tatra Mountains considering between the years of 1921 and 2006, using data from 5 in situ stations. It is reported that, in higher elevations, the duration of snow is almost as high as 60% of the year (i.e., approx. 220 days) in snow season (July-June is considered). Maximum snow depth varies from as low as 5 cm to higher than 160 cm, as observed in the snow season. Annual average temperature is measured in between 0.5 and 3.5 ^{o}C at the meteorological station of Skalnate´ pleso, with the elevation of 1750 m from the year of 1965 to 2002, demonstrating a sharp rise in recent years (Singh, 2013).

Kaçkar Mountains is a mountain range located in the northeastern Turkey, in Trabzon and Rize provinces, close to the coast of Black Sea. The mean elevation of the tile from Kaçkar Mountains is 2112 m. The peak elevation of the mountains is reported as 3932 m, and it is accepted as the highest mountain range in Eastern Black Sea region of Turkey (Bayrakdar & Özdemir, 2014). The mountains are located close to the area of the moisture supply of the Black Sea, and therefore, the annual precipitation is approximately 2000 mm, which is much higher within the valleys of

Kaçkar than the annual average precipitation of Turkey (Reber et al., 2022). The only few glaciers remaining in Northeastern Anatolia are located in Kaçkar mountains (Reber et al., 2022).

For all three regions, ERA5-Land snow depth data, temperature and precipitation data have been presented in Chapter 4 for better comparability of the climate parameters of these regions.

Out of the six seasonal snow cover classes that are proposed by Sturm et al. (1995), seasonal snow on all three Sentinel-2 tiles have been classified as "maritime snow" as can be seen in Figure 3.3.

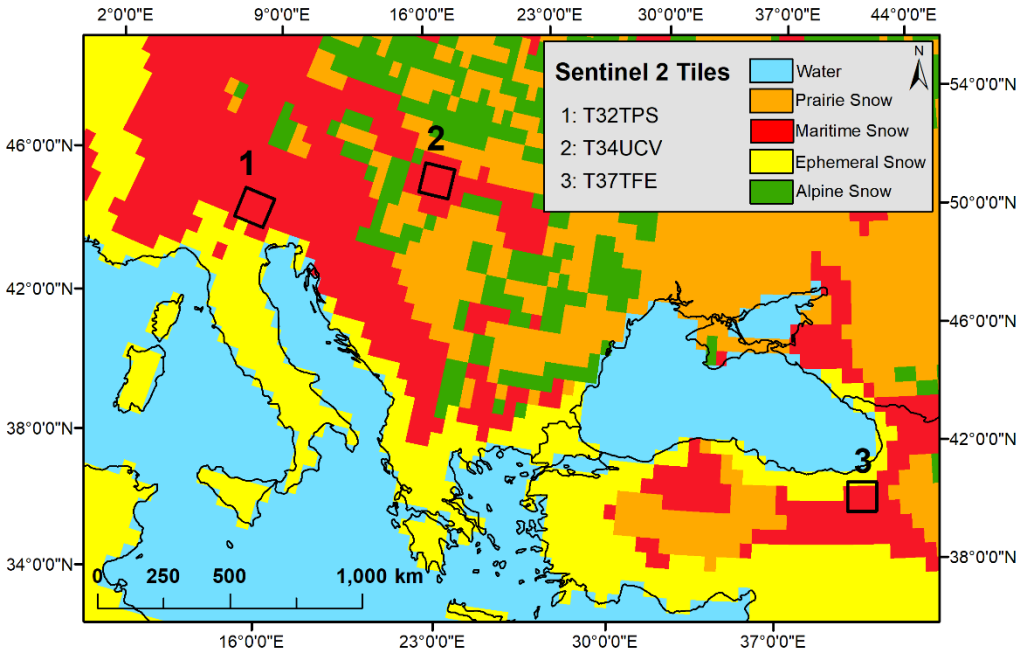


Figure 3.3. Seasonal snow cover types of Sentinel-2 tiles (Sturm et al., 1995).

The satellite image dataset of the study consists of 9 Sentinel-2 Level 1-C images for the 3 selected areas that are described in Section 3.1. For each study area, one Sentinel-2 image for the period when the snow cover starts to accumulate (from November to December), one for the period when the amount of snow is at climax (from January to March) and finally, one for the period when the snow starts to melt

(from April to June) have been used. Sentinel-2 products are composed of 100 km × 100 km tile grids, which have specific tile numbers. The tile numbers, as well as the corresponding dates of acquisition are given in Table 3.4.

Table 3.4. The tile numbers and dates of Sentinel-2 products

Location	Tile Number	Date of acquisition
Alpine Region	32TPS	5 December 2018
		24 January 2019
		13 June 2019
Tatra Mountains	34UCV	29 November 2018
		23 January 2020
		8 April 2019
Kaçkar Mountains	37TFE	19 December 2019
		19 March 2018
		8 April 2018

The Sentinel-2 L1C raw data consist of 13 bands (cf. Table 3.1).

3.1.1 Land Cover

Some studies suggest that land cover affects the detectability of snow cover, especially in forest land cover type (Hall et al., 2001; Robinson et al., 1993; Bitner et al., 2002). The distribution of land cover classes and the changes in these classes between the years of 2018 and 2019 in all three Sentinel-2 tiles that have been observed for this study are presented in Figure 3.4 -Figure 3.9 and Table 3.5. Land cover class distribution have been obtained from Google's Dynamic World (DW) land cover products (Dynamic World, n.d.; Brown et al., 2022) , and the percentage of the land covers are calculated by the help of QGIS software. To present the *trees*

land cover, snow-free season was obtained for land cover data (from May to December).

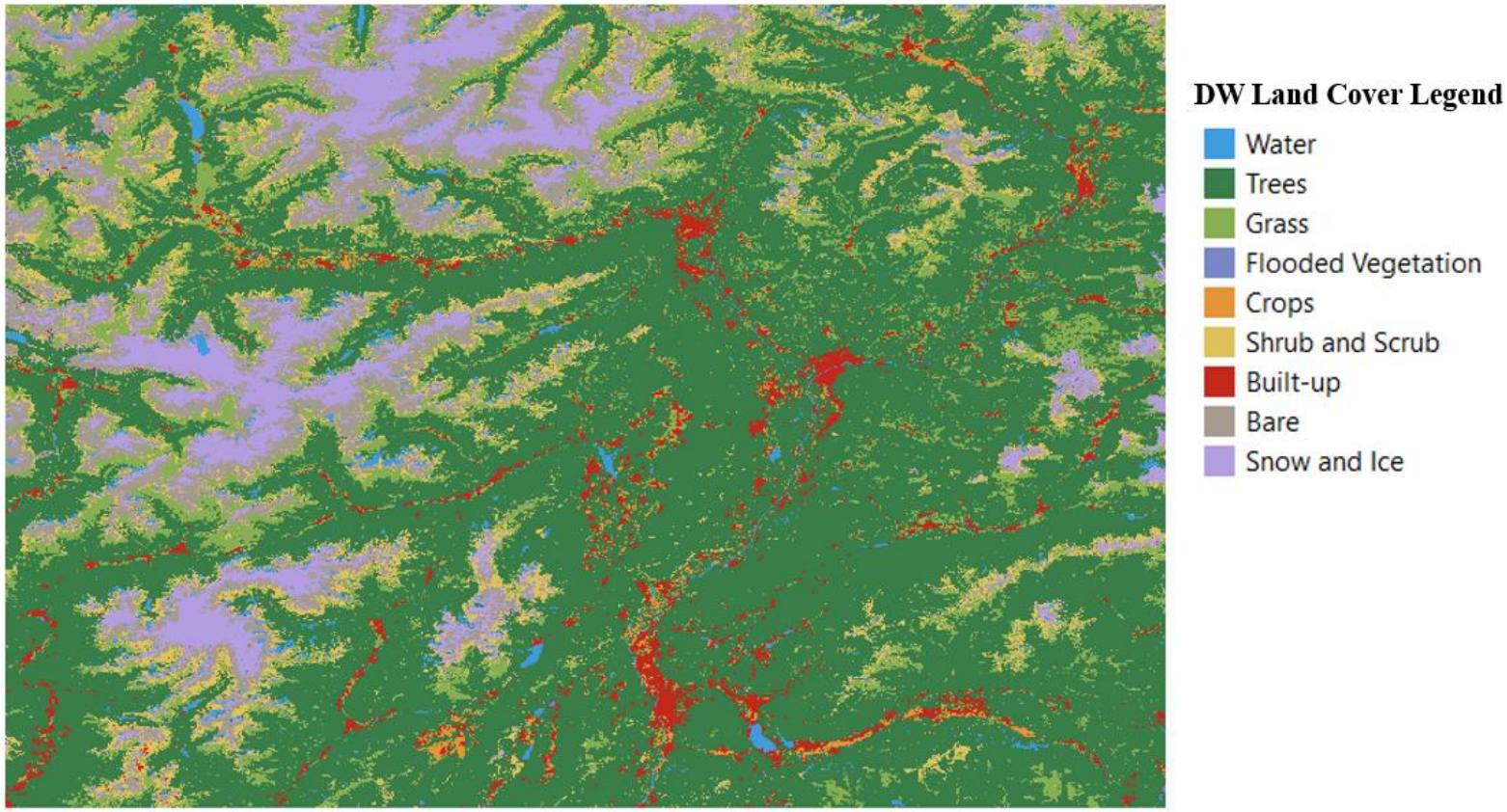


Figure 3.4. DW land cover product of Alps Region tile, 2018

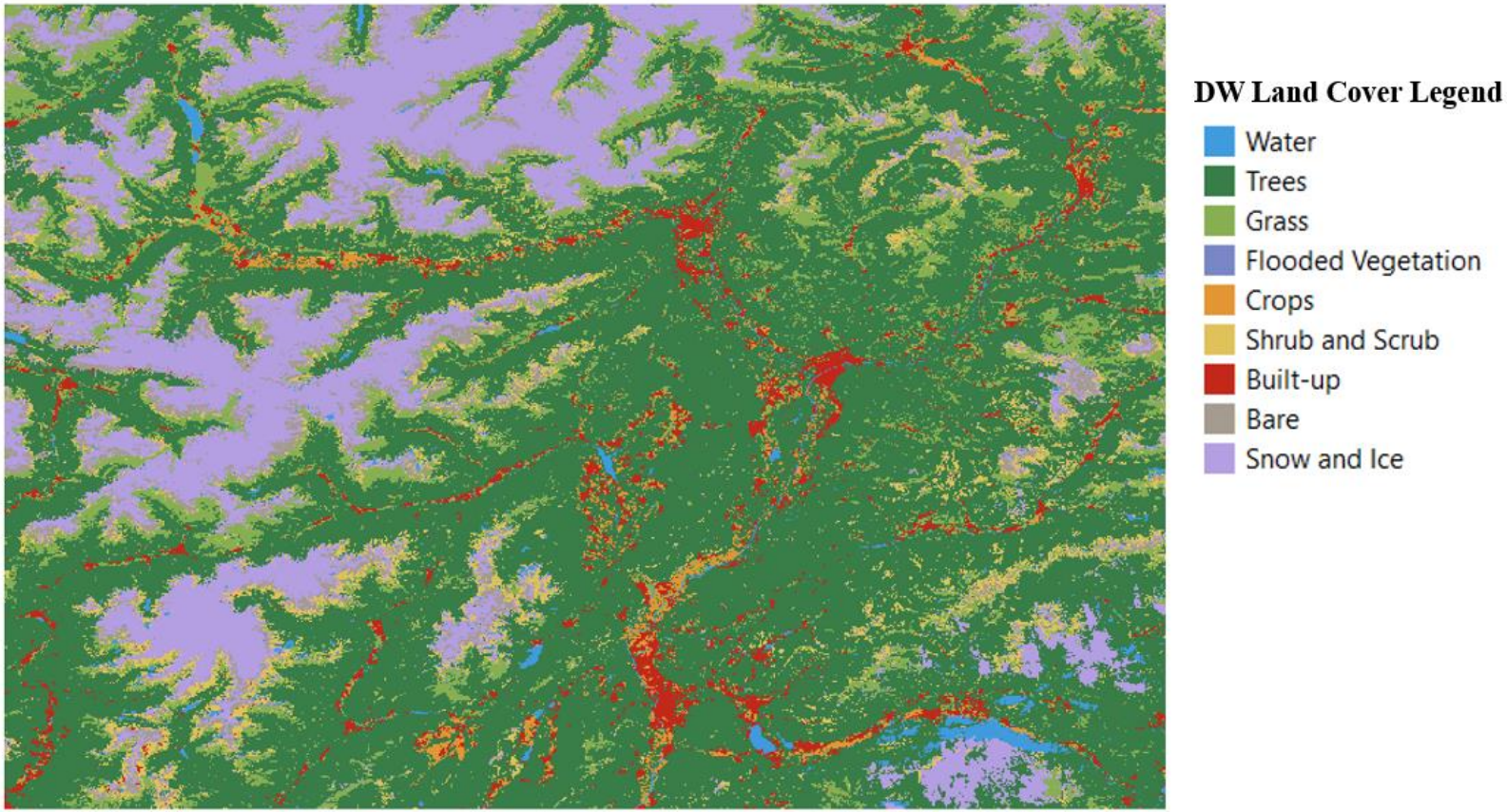


Figure 3.5. DW land cover product of Alps Region tile, 2019

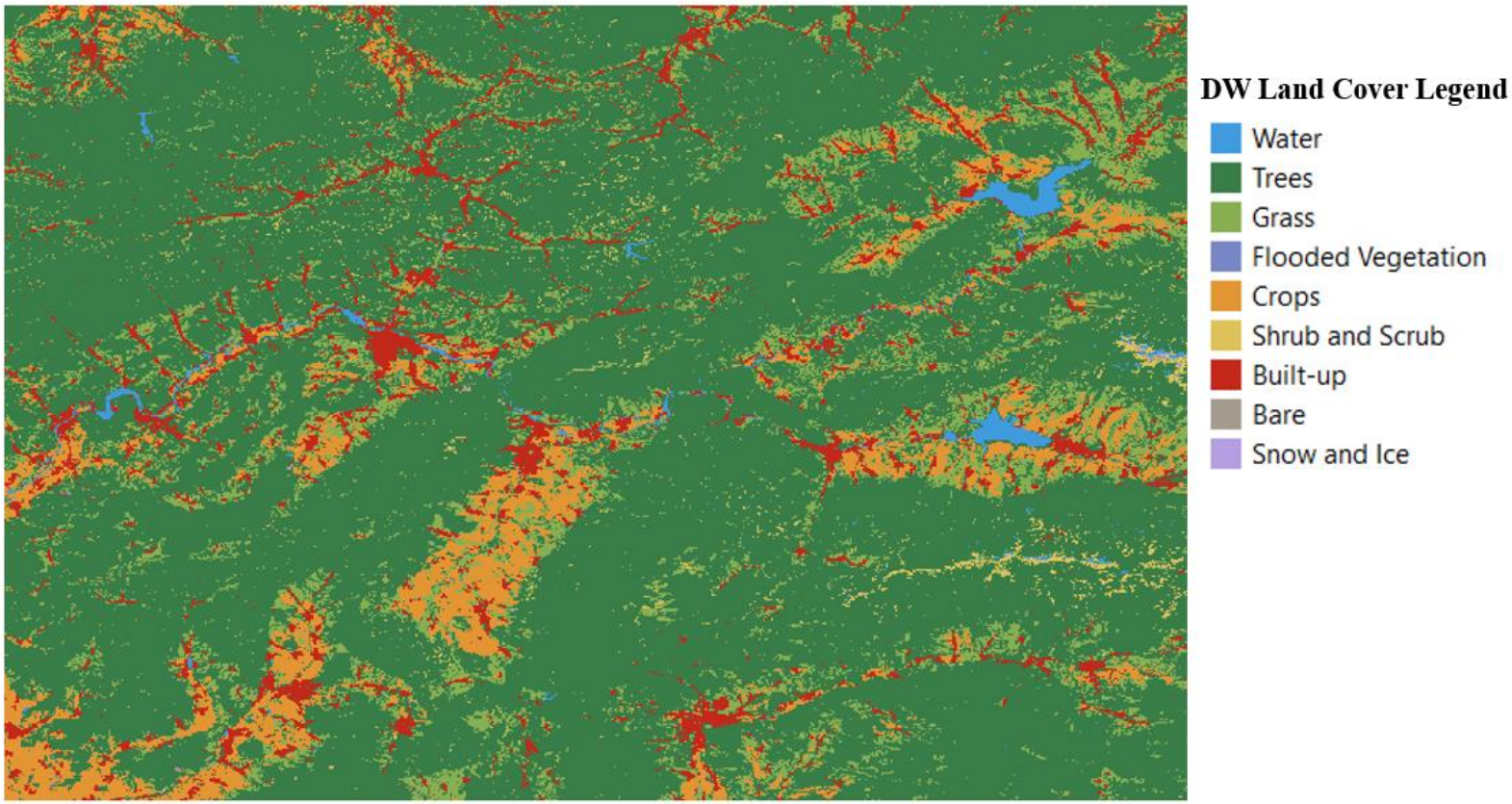


Figure 3.6. DW land cover product of Tatra Mountains tile, 2018

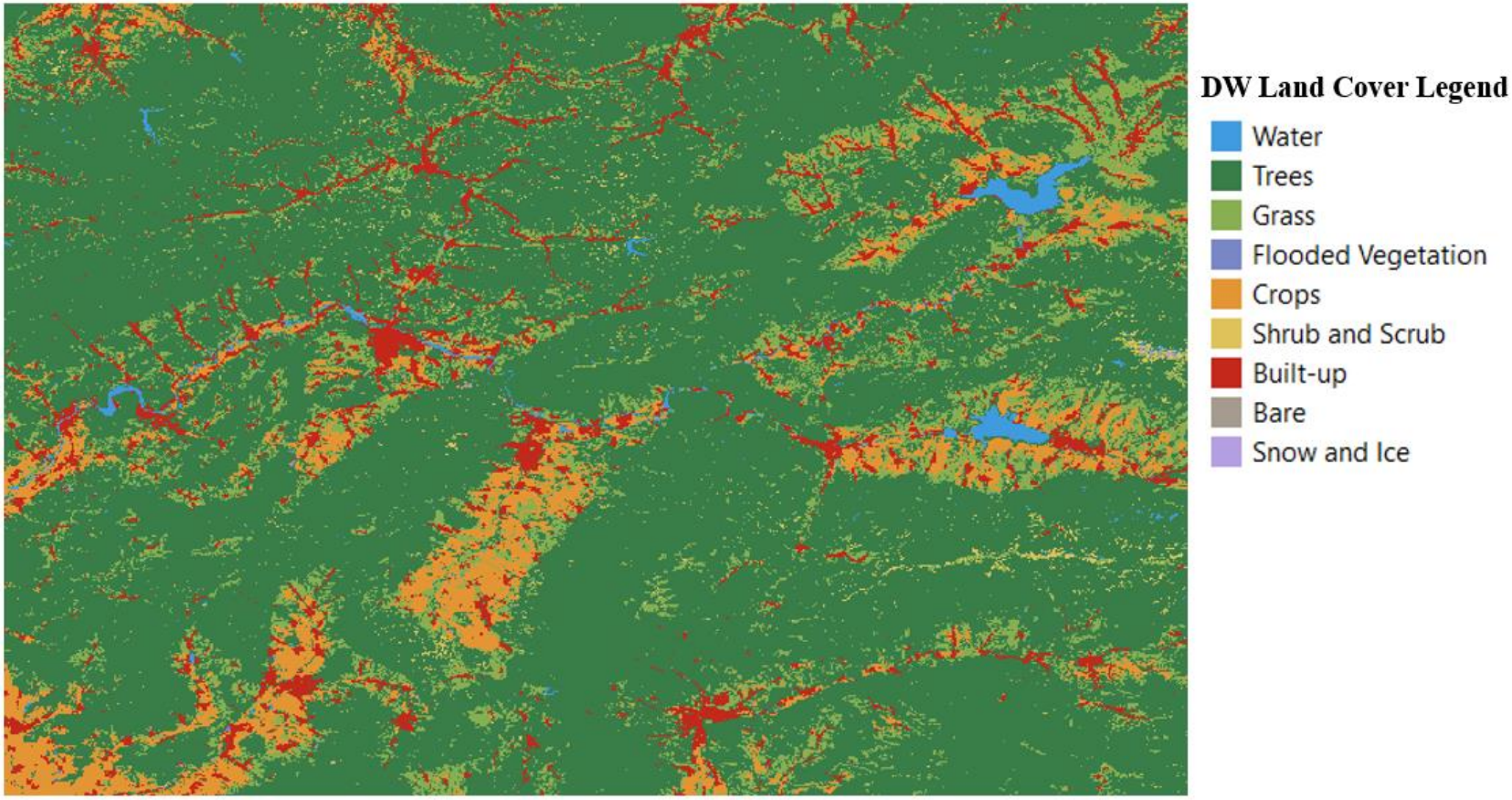


Figure 3.7. DW land cover product of Tatra Mountains tile, 2019

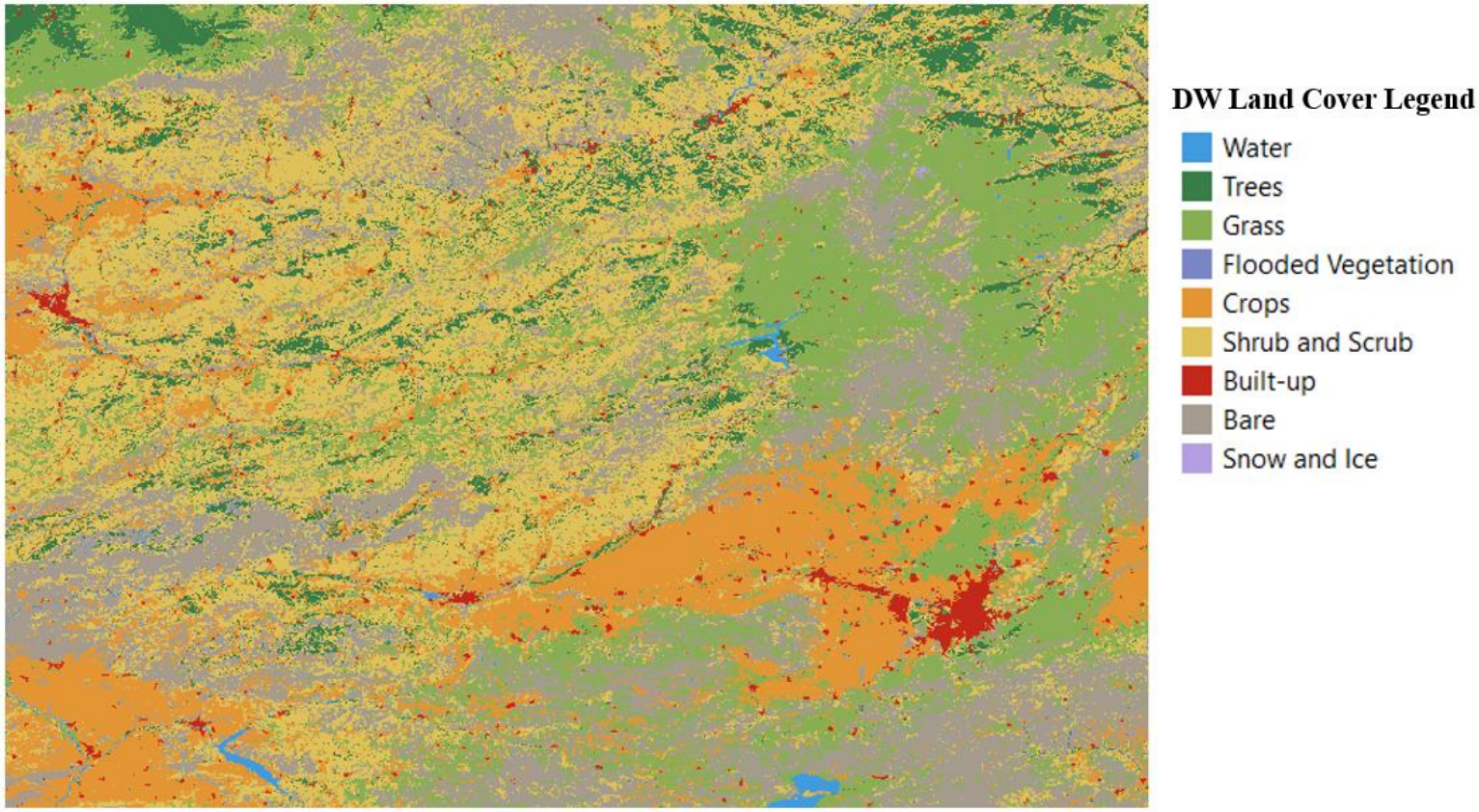


Figure 3.8. DW land cover product of Kaçkar Mountains tile, 2018

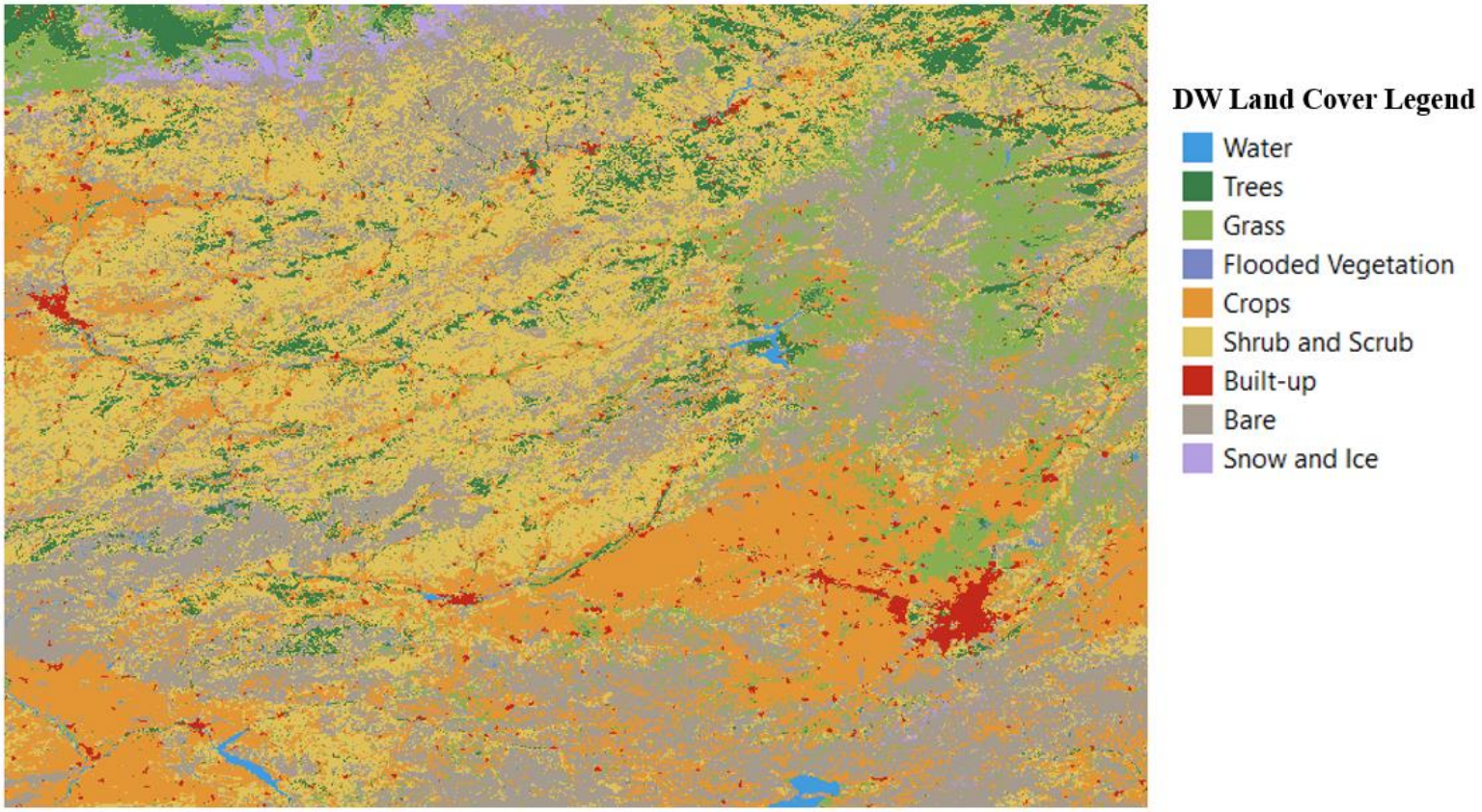


Figure 3.9. DW land cover product of Kaçkar Mountains tile, 2019

Table 3.5. The percentage of the land cover distribution of Sentinel-2 tiles in 2018 and 2019 (May-December).

	Location of the Sentinel-2 Tile					
	European Alps		Tatra Mountains		Kaçkar Mountains	
Land Cover Class	2018	2019	2018	2019	2018	2019
water	1.6	1.2	0.8	0.8	0.5	0.6
trees	60.2	57.9	74.9	75.0	8.2	7.2
grass	11.7	11.9	11.2	11.5	20.2	9.0
flooded vegetation	0.0	0.0	0.0	0.0	0.0	0.0
crops	1.0	1.5	6.4	6.2	15.9	20.8
shrubs and scrub	7.0	5.2	0.9	0.7	29.5	30.1
built-up	3.2	3.4	5.7	5.8	1.6	1.6
bare	9.0	3.5	0.1	0.1	24.1	29.0
snow and ice	6.3	15.5	0.0	0.0	0.1	1.6

The most dominant land cover feature in both tiles from Alpine Region and Tatra Mountains is *trees* land cover class. The tree distribution in the tile from Kaçkar Mountains, differing from the other two regions, have been overshadowed by *shrubs and scrub class*, then followed by *bare class.*

Land cover change between 2018-2019 in three tiles is illustrated in Figure 3.10.

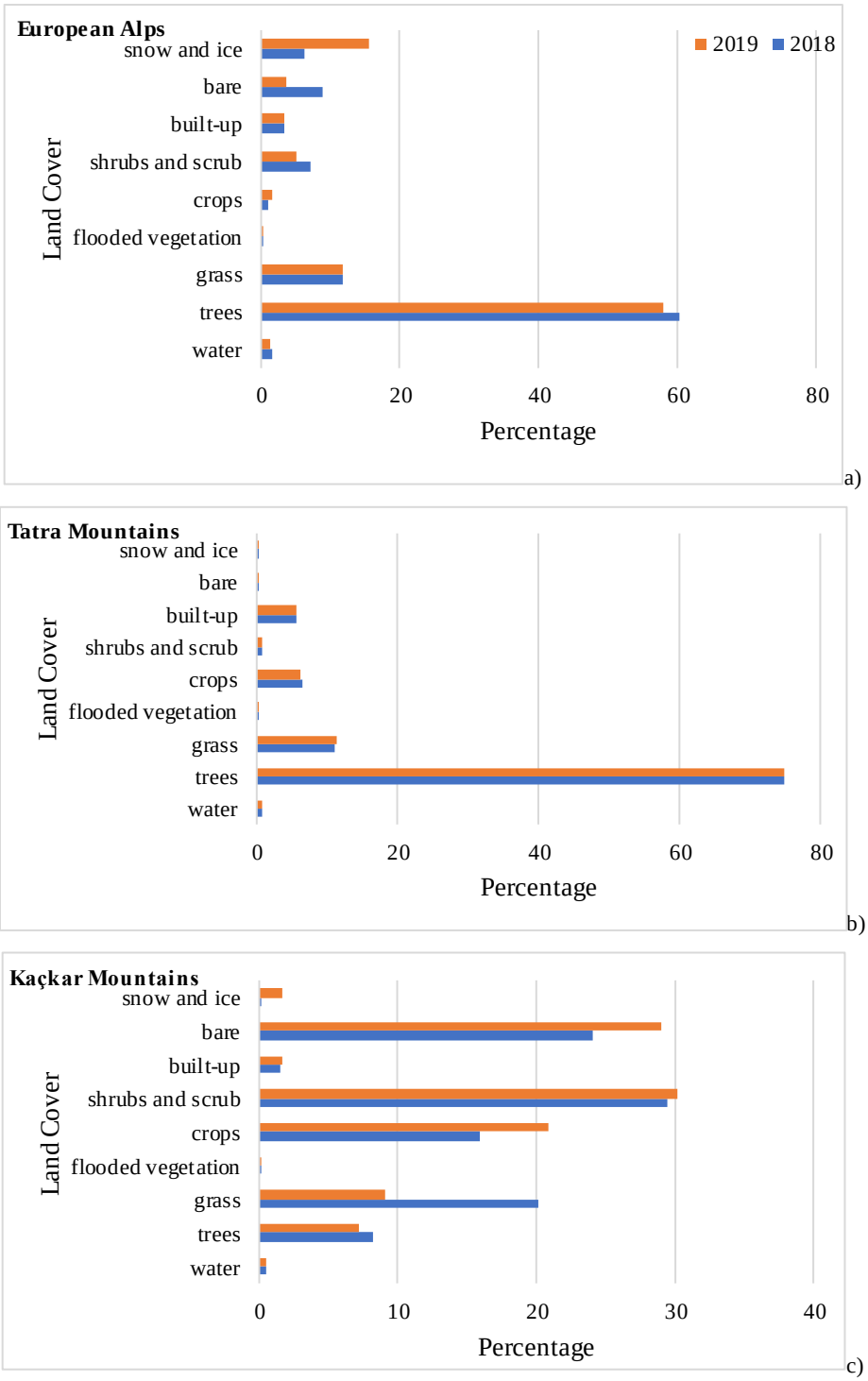


Figure 3.10. The percentage comparison of land cover classes of the Sentinel-2 tiles for the years of 2018 and 2019 for a) European Alps b) Tatra Mountains and c) Kaçkar Mountains

As can be seen in Figure 3.10, there is not a significance change in land covers between the years of 2018 and 2019 for all tiles.

In this study, four classes have been considered for RF algorithm: Cloud, No-snow (Land), Snow and Water. Under No-snow (Land) class, the following land cover classes are considered: *trees*, *grass*, *crops*, *shrubs and scrub*, *bare* and *built-up*.

3.2 Image Pre-Processing Stages

In this study, the Sentinel-2 bands (Band 2, 3 and 4) are resampled to 20 m resolution via the Sentinel-2's Level 2A Processor Sen2Cor (Main-Knorn et al., 2017). The resampling process of Sen2Cor employs cubic spline method. The NDSI, NDVI and NDWI are produced from the resampled data. Then, Sentinel-2 bands are atmospherically and topographically corrected again using Sen2Cor (Main-Knorn et al., 2017).

The bands are then used as input to RF model for different predictor variable combinations that are described in Table 3.2 and Table 3.3.

3.2.1 Normalized Band Indices

This study makes use of three different indices, retrieved from the Sentinel-2 bands, in all input combinations for the RF classification algorithm, to differentiate the four classes: Cloud, No-snow (Land), Snow and Water. These indices are: NDSI, NDVI and NDWI.

NDSI divides the visible (VIS) and near infrared (NIR) or short-wave infrared (SWIR) bands to differentiate clouds from snow cover. In terms of snow cover mapping, NDSI-based approaches have broadly been conducted all around the world, and yielded highly accurate results (Luo et al., 2022), including using the index as an input variable for RF and other machine learning algorithms.

The NDSI is formed by the generalized calculation below (Nagajothi et al., 2019; Hall & Riggs, 2011):

$$NDSI = \frac{(Green-SWIR)}{(Green+SWIR)} \quad \text{(Eq 3.1)}$$

where, for Sentinel-2 data, the green band is band 3 and the SWIR band is band 11.

NDVI is a commonly used index to differentiate vegetation from other land cover types, formed by NIR and red band (Jin et al., 2018). For better mapping of snow cover, in the conditions where snow is at dense vegetation, shadows and low illuminated areas, NDVI, combined with NDSI is used to determine snow, and provide adequate results (Klein et al., 1998). NDVI is frequently used to differentiate cloud from other land covers, as cloud pixel values are zero for both NDVI and NDSI (Ghasemian & Akhoondzadeh, 2018).

The NDVI is realized by the following equation (Ghasemian & Akhoondzadeh, 2018):

$$NDVI = \frac{(NIR-Red)}{(NIR+Red)} \quad \text{(Eq 3.2)}$$

where the red band corresponds to band 4 whereas NIR band corresponds to band 8 for Sentinel-2.

NDWI is the most widely used method to detect surface water features for satellite images (McFeeters, 1996; Burton-Johnson et al., 2016; Acharya et al., 2018).

The NDWI is obtained through the following calculation (McFeeters, 1996):

$$NDWI = \frac{(Green-NIR)}{(Green+NIR)} \quad \text{(Eq 3.3)}$$

where green band is band 3, whereas NIR band is band 8 for Sentinel-2.

3.2.2 Atmospheric and Topographic Correction with Sen2Cor

The reflectance of the objects detected by satellites are often subjected to atmospheric absorption and scattering, instigating distortion of the actual reflectance of the objects, thus affecting the data gathered from remotely sensed images (Mahiny & Turner, 2007). For this reason, atmospheric correction is advised for classification and change detection operations and generally yields better results than non-atmospherically corrected data (Song et al., 2001).

Topographic correction attributes to rectification of the different solar reflectance as a result of the irregular surfaces on the terrain. This difference in reflectance results in a high diversity of reflectance values for similar vegetation categories. This also causes areas with shadow showing less than expected reflectance values as well as illuminated areas giving more than expected reflectance (Riano et al., 2003). Therefore, topographic correction is also essential for correct handling of terrain incident angles and shadows, especially in areas with rough terrain, and thus, yielding to more accurate reflectance values (Padró et al., 2018; Riano et al., 2003).

It is reported that the accuracy results of classification conducted on atmospherically and topographically corrected images resulted well (Vanonckelen et al., 2013). To that end, both atmospheric and topographic corrections are worthwhile to execute before any land cover classification application.

Atmospheric and topographic correction in Sen2Cor is mainly based on ATCOR algorithm (Richter et al., 2006). In several radiometric correction processes, such as Sen2Cor, topographic correction is applied by utilizing a DEM as secondary data (Padró et al., 2018). In Sen2Cor, SRTM DEM data with 90 m resolution is used for topographic correction automatically (as preferred in this study), but it can be changed within the application (Mueller-Wilm, 2020). Topographic correction is done with empirical bidirectional reflectance distribution function (BRDF) method with three adjustable variables. Topographic correction is executed when the elevation difference of the granule is bigger than 50 meters or at least 1% of the pixels of an image have a slope more than 6 degrees, since the effect of the terrain

irregularity can be examined when slope value is equal to or surpasses the threshold of 7 degrees. Sen2Cor does not conduct topographic correction in flat locations to prevent artefacts (Louis, 2021).

The predominant Level-2A processing phases of Sen2Cor are presented in Figure 3.11. Level-2A processing is implemented to the Top-of-Atmosphere (TOA) reflectances of Sentinel Level-1C data. The processing of Sentinel-2 products begins with cloud detection and scene classification. Then, aerosol optical thickness (AOT) and water vapor (WV) are retrieved from the L1C data. Finally, TOA reflectance values are transformed to Bottom-of-Atmosphere (BOA) reflectances. In addition to the main processing steps, Sen2Cor also consists of additional steps that are optional, such as cirrus correction, terrain correction, adjacency correction and empirical BRDF corrections (Main-Knorn et al., 2017).

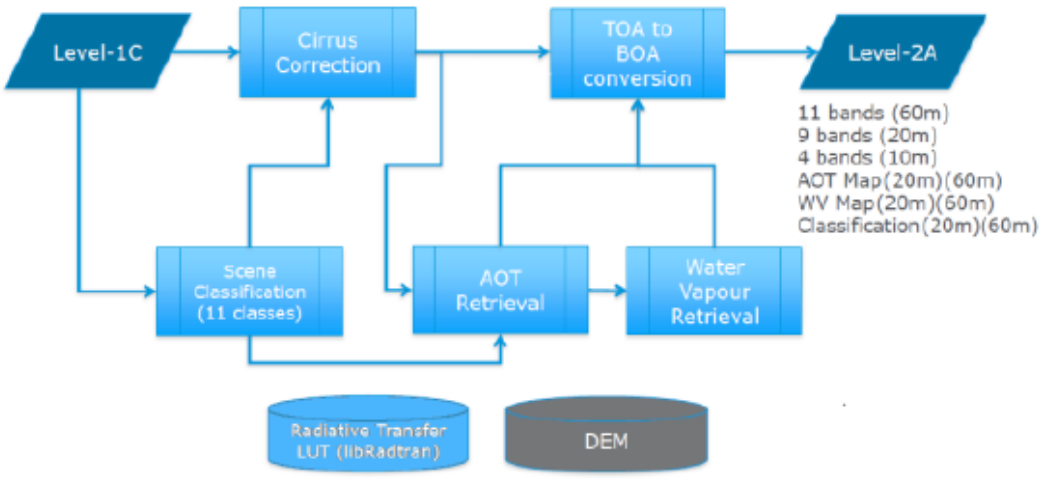


Figure 3.11. Sen2Cor main processing steps (Main-Knorn et al., 2017)

In this study, Sen2Cor is used for both atmospheric and topographic correction with the aid of SNAP (Sentinel Application Platform) software. SNAP is a free open-source toolbox for Sentinel image processing and Sentinel time series images,

developed by European Space Agency (ESA) (European Space Agency, n.d.; Gomarasca et al., 2019).

3.2.2.1 Scene Classification Layer (SCL)

The Scene Classification Layer (SCL) is obtained from the pre-processing phase of Sen2Cor, while Level-1C product is converted to Level-2A (Main-Knorn et al., 2017). SCL consists of 11 classes, which are given together with their associated color codes in Figure 3.12.

0	No Data (Missing data on projected tiles) (black)	
1	Saturated or defective pixel (red)	
2	Dark features / Shadows (very dark grey)	
3	Cloud shadows (dark brown)	
4	Vegetation (green)	
5	Bare soils / deserts (dark yellow)	
6	Water (dark and bright) (blue)	
7	Cloud low probability (dark grey)	
8	Cloud medium probability (grey)	
9	Cloud high probability (white)	
10	Thin cirrus (very bright blue)	
11	Snow or ice (very bright pink)	

Figure 3.12. The class labels and their color codes in Sentinel-2's SCL (Main-Knorn et al., 2017)

In this study, SCL is used in selection of test datasets:

- Cloud class is extracted from SCL attributes 8 and 9,
- No-snow (Land) class is extracted from SCL attributes 4 and 5,
- Snow class is extracted from SCL attribute 11, and

- Water class is extracted from SCL attribute 6.

There have been observed accuracy issues of SCL in other studies (Raiyani et al., 2021), as well as in this study. For that reason, the extracted attributes are further visually checked and adjusted manually, with the help of ArcGIS 10.7 software, especially for the attributes selected as Water class and No-snow (Land) class, to obtain more accurate results.

3.3 Digital Elevation Model (DEM)

DEM is a measurable depiction of terrain, and it is crucial for Earth science, hydrological applications and analysis of remote sensing data in general. DEM can be obtained by employing photogrammetry, interferometry, ground and laser surveying, as well as other methods (Mukherjee et al., 2013). DEMs were started to be produced in the 1960s. The elevation of the surface can be obtained by employing two different angles in the visible or near-infrared regions of the electromagnetic spectrum (Bühler et al., 2011).

Topography affects snow cover distribution; therefore, DEM assists in improving the accuracy of snow cover classification (Tsai et al., 2019b). In this study, DEM data is obtained for the three regions from SRTM data (FABDEM, n.d.; Hawker et al., 2022) and resampled to 20 m resolution to be used in several input combinations, with cubic spline method. The slope and aspect data, and hypsometric curves have been obtained in ArcMap 10.7 software. The visual representation of elevation, slope and aspect data, and hypsometric curves derived from the resampled SRTM DEMs for all three study areas is shown in Figure 3.13. These DEMs have also been automatically used as input for topographic correction by the SNAP toolbox (cf. Section 3.2.2).

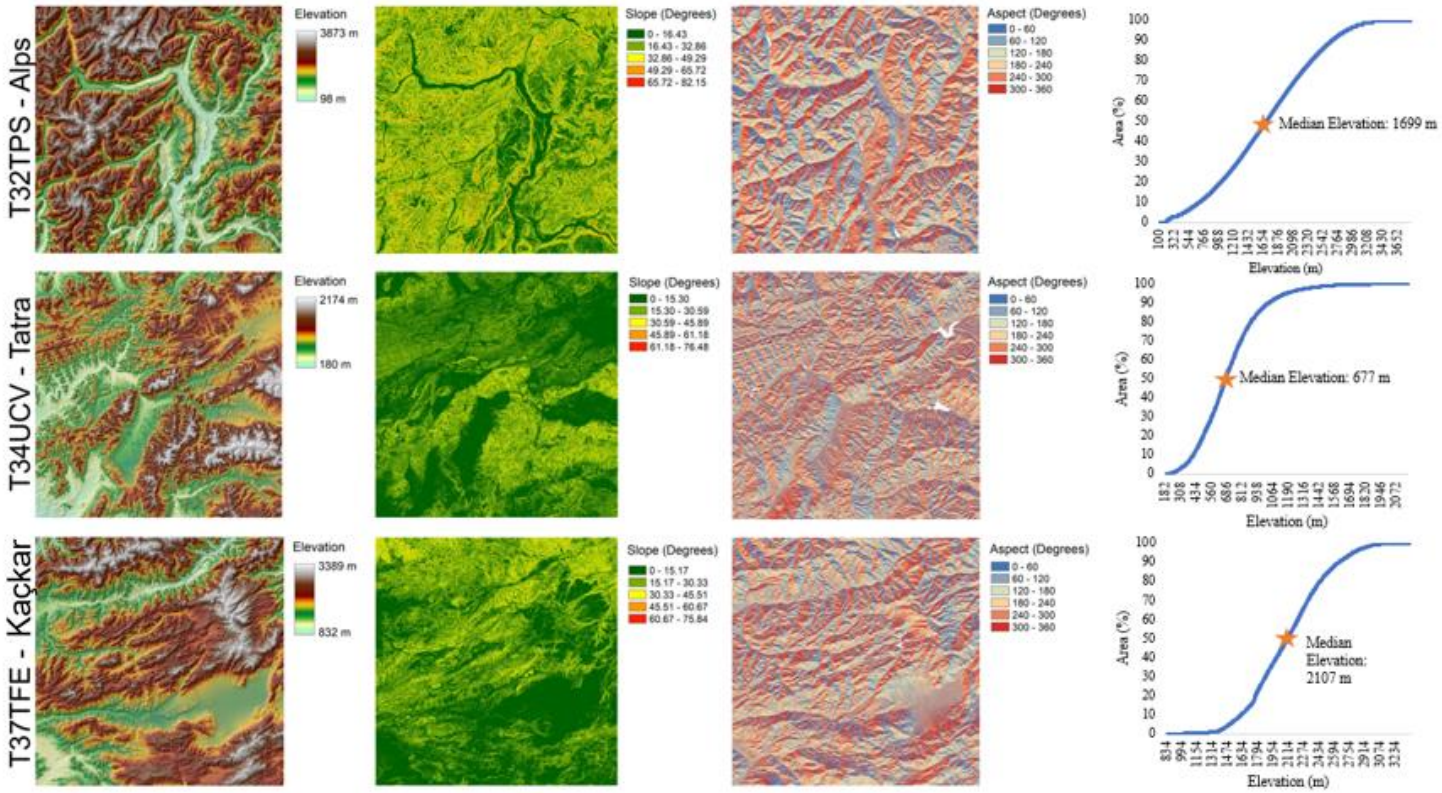


Figure 3.13 Elevation, slope and aspect maps, and hypsometric curves of the three study areas.

3.4 Principal Component Analysis (PCA)

PCA is used as a procedure of re-articulating the data content of a multispectral set on a number of images (i.e., *m*) as a set of *m* number of principal components. There are three procedures to compute principal components in remote sensing, which are (i) Obtaining covariance/correlation matrix with the aid of input image data sets, (ii) Obtaining eigenvalues and eigenvectors that are orthogonal to each other and (iii) obtaining principal components (Eklundh & Singh, 1993).

Principal components possess two notable features:

- The principal components are statistically unrelated (or orthogonal to each other)
- The principal components have maximum variance, which implies that the principal components are derived in the decreasing sequence of variability (Mather & Koch, 2011; Eklundh & Singh, 1993).

The PCA has been employed in remote sensing applications for a variety of goals (Rodarmel & Shan, 2002). In this study, PCA is employed in several input combinations, as the usage of the first several principal component images can result in higher accuracy in remote sensing classification applications (Rodarmel & Shan, 2002).

In this study, the first three PCA components are derived with the aid of sci-kit learn library of Python (The relevant Python script can be observed in Appendix C). The dataset for the PCA is determined as the 9 bands of Sentinel-2 L2A data, which atmospherically and topographically corrected with the help of Sen2Cor. The bands are as follows: band 2-7, band 8A, band 11 and band 12 (cf. Table 3.1).

3.5 Random Forest (RF)

RF is a supervised classification and regression algorithm presented by Breiman in 2001 (Breiman, 2001). RF consists of an ensemble of Classification and Regression

Trees (CARTs). It is primarily rooted in the bagging concept of Breiman (Breiman, 1996), also known as "bootstrap aggregation". Each tree in ensemble of bootstrap aggregation consists of a bootstrap element randomly selected with a substitute from the training data. The trees are accumulated (averaged) to yield the "bagged" estimation (Cutler, 2010).

A critical asset of RF is the out-of-bag (OOB) error. OOB is generally regarded as an acceptable estimator of the error expected for unseen data. OOB data attributes to the observations that are not included in the building process of some of the trees. OOB dataset is applied internally with the aid of the algorithm in the role of a validation dataset in the course of the training process. The relevant OOB error of an RF model is basically the mean error frequency that is calculated during the observations derived out of the set are determined with the usage of trees as they are OOB (Boulesteix et al., 2012).

A few noteworthy features that established RF as attractive in engineering and scientific studies and applications can be outlined as follows:

- RF, as an algorithm, is applicable for multi-class classification in addition to regression problems,
- RF can model interactions between predictor variables,
- The algorithm is fast due to the fact that only a subset of features is necessary for the algorithm during the training,
- Model tuning is dependent on one or two parameters alone: The number of trees in the forest and the number predictors that are randomly picked at each individual split,
- An elemental estimate of the generalization error, OOB error, is given,
- High dimensional complex datasets can be managed due to the fact that the algorithm trains on subsets of data,
- The algorithm grants means of variable importance (Cutler et al., 2012).

3.5.1 Training of the RF Models

The RF algorithm makes use of three essential model tuning parameters. The parameters are following:

- The number of trees: Ntree,
- The number of predictor variables that is randomly picked at individual splits: Mtry, and,
- The minimum number of observations for each tree: Lsize.

Ntree has an inverse proportional relation with the variance of the RF model. The generalization error of an RF model ultimately converges to a limit as Ntree number rises, as also Breiman states that Strong Law of Large Numbers omit the possibility of overfitting for RF (Breiman, 2001). This indicates that a selection of Ntree number that is considerably large will not yield to an overfitting (Biau & Scornet 2016). In literature, for satellite imagery, a broad extent of Ntree number have been used, from as small as 50 to as big as 5000 (Colditz 2015; Guan et al. 2013; Houborg & McCabe 2018; Nitze et al. 2015; Pelletier et al. 2016), although the acceptable number of Ntrees is deemed as 500. The reason for it is that the generalization error often reaches to a stability as Ntree number goes up to 500 (Lawrence et al. 2006). Thus, in this study, Ntree is taken as 500 with every different input combination.

Mtry is generally denoted as the singular feature that RF is affected (Cutler et al. 2012). Mtry is the number of input variables that are randomly selected, and the split where the best is determined only in this subset of data (Tyralis & Papacharalampous, 2017). As observed with the study Díaz-Uriarte and De Andres (2006) presented the susceptibility of overfitting because of the selection of Mtry is negligible. The default number of Mtry is denoted as the square root of the total number of variables for classification. This default number of Mtry is generally used in classification tasks in remote sensing applications (Belgiu & Drăguţ, 2016; Houborg and McCabe 2018).

The number that is used as default for Lsize is 1 for classification (Biau & Scornet 2016; Cutler et al. 2012). The default number of Lsize is denoted as a good selection; however, there is lack of strong proof on the matter (Díaz-Uriarte & De Andres, 2006). This default number is also often recognized when it comes to the classification of satellite data (Abdel-Rahman et al. 2014, Mutanga et al. 2012).

In this study, the default values for Mtry and Lsize is used for all input combinations. The classification was done for four class labels: Cloud, No-snow (Land), Snow and Water. The training dataset that was prepared with the expert judgement was broken down into two subsets as 70% of the dataset for training and 30% for testing, by simple random sampling, the most basic random sampling technique, where every sample has an equal chance of being chosen that are independent from other samples (Wadoux et al., 2019) The RF model is then conducted with the different combinations of inputs with the aid of *sklearn.ensemble.RandomForestClassifier*, a sci-kit learn Python library (the relevant code can be found in Appendix D)

3.5.2 Selection of Training and Testing Datasets

For remote sensing applications, various methodologies are employed to obtain training and testing datasets:

- Collecting in-situ data
- Collecting data as polygons with the help of a GIS application, and
- Seeding of training data (Jensen, 2006)

In this study, selection of training and test datasets for all of the 9 images is conducted via polygon shapefiles with the aid of ArcMap for the classification with the four classes: Cloud, No-snow (Land), Snow and Water. The data are collected depending on the expert visual analyses on the Sentinel-2 images, which are atmospherically and topographically corrected by Sen2Cor. The dataset is then divided into two subsets as 70% (training), and 30% (testing). The training and test datasets are then formed with the Python code presented in Appendix A. This code extracts the input

data (for all the input combinations) to the manually drawn polygons, to later derive randomly picked samples of training and test datasets of (i) 300 samples for each class, and (ii) 1000 samples for each class (for every input combination), and prepares the training and test datasets for the RF model. The reason for choosing 300 samples for each class is that, the number of training samples per class is usually chosen in between *10* number of bands* to *30* number of bands* while classifying remote sensing images (Van Niel et al., 2005), whereas the reason for choosing 1000 samples to train for each class is to observe whether there is a significant change in accuracy when increasing the size of the usual training sample per class by more than three times.

3.6 Accuracy Assessment

Thematic maps obtained by the classification of remote sensing images eventually inherit errors. These errors might be coming from the data receiving process, or due to unwanted atmospheric effects, and some of the errors may remain even after atmospheric and topographic correction (Jensen, 2006). Errors of human interpretation are also possible. Thus, a detailed accuracy assessment should be conducted on the classified image (Jensen, 2006). An accuracy assessment is employed with the presence of classified data and ground reference data. In this section, the test data and the related statistical metrics employed for the accuracy assessment are presented.

3.6.1 Test Data

An accuracy assessment should be conducted with an unbiased test dataset. The ideal test dataset should be obtained as ground reference test points or polygons, as Jensen (2006) suggests, but it is unfortunately not possible for every study. For that reason, in this study, the test data points for all classes are obtained randomly from the SCL with the help of ArcMap 10.7 software (for details of SCL, cf. Section 3.2.2.1).

The sample sizes for the test data are obtained in accordance with the sample size based on multinominal distribution formula (Ozdarici Ok & Akyurek, 2012; Jensen, 2006)

$$N = \frac{B\Pi_i(1-\Pi_i)}{b_i^2} \qquad \text{(Eq 3.4)}$$

where:

- N is the sample size
- i refers to the amount (in proportion) from the population within the ith class out of k classes that possess the percentage of the population nearest to 50%
- b_i is the aspired precession for this class
- B refers to the upper $(\alpha/k) \times 100$th percentile of the chi-square ($\chi 2$) distribution with one degree of freedom (confidence level)
- k is the class number (Ozdarici Ok & Akyurek, 2012).
- In this study, the confidence level (B) is depicted as 95% whereas the precision is depicted as 5% (Jensen, 2006). Thus, B and b become 7.568 and 0.05, respectively.

In accordance with the findings of Hashemian et al. (2004) regarding the sampling of test data, 70 samples per class is observed to yield more accurate results when the image size is 512 × 512 pixels or larger. The study (Hashemian et al., 2004) indicates that this result is consistent with the recommendation of Congalton (1991) of using 75 to 100 number of sample size while employing classification models on larger areas in remote sensing. In this study, the RF classification algorithm is conducted on all 1001 × 1001 spatial subsets for all input combinations. For full tiles (i.e. 5490 × 5490 pixels), a minimum of 100 samples is determined for each class, for the best-case input combination.

3.6.2 Employed Statistical Metrics for Accuracy Assessment

The relation between the classified and the ground reference data (i.e. test data) can be observed by obtaining a confusion matrix (i.e., error matrix) (Jensen, 2006). In this study, confusion matrix along with the Cohen's Kappa coefficient and overall accuracy (OA) are employed. A confusion matrix consists of two dimensions, one dimension is denoted to the true values of the classes, and the other represents the predicted values (Deng et al., 2016). Structure of a sample confusion matrix is given in Table 3.6

Table 3.6. Structure of a confusion matrix (Deng et al., 2016)

		Predicted		
		A_1	A_j	A_n
Actual	A_1	N_{11}	N_{1j}	N_{1n}
	A_j	N_{i1}	N_{ij}	N_{in}
	A_n	N_{n1}	N_{nj}	N_{nn}

The assessment of overall performance of the classifier can be obtained by OA. It depicts the percentage of total correctly classified pixels (Jog & Dixit, 2016). The calculation of OA is shown below:

$$\text{Overall Accuracy} = \frac{\text{Number of correctly classified pixels}}{\text{Number of classified pixels}} \qquad \text{(Eq 3.5)}$$

Kappa coefficient is a discrete multivariate method, discovered by Cohen (1960), which measures the agreement between two evaluators, which are test values and predicted values in terms of classification. A value of Kappa greater than 0.80 are denoted to show a strong agreeement, values between 0.80-0.40 present moderate

agreement, and finally, values below 0.40 indicate poor agreement (Congalton, 2001).

The formula of Kappa coefficient is as follows:

$$\text{Kappa Coefficient} = \frac{N\sum_{i=1}^{k} x_{ii} - \sum_{i=1}^{k} x_{i+} \times x_{+j}}{N^2 - \sum_{i=1}^{k} x_{i+} \times x_{+j}} \qquad \text{(Eq 3.6)}$$

where k is the number of classes, x_{ii} is the number of observations in row i and column i in the confusion matrix, x_{i+} and x_{+j} are the marginal totals for row i and column j, and N is the total number of samples (Congalton & Green, 2009).

CHAPTER 4

RESULTS AND DISCUSSIONS

20 km × 20 km area of 9 Sentinel 2 images from three different tile locations over *i*) European Alps, *ii*) Tatra Mountains, and *iii*) Kaçkar Mountains are classified with RF algorithm as four land cover classes (Cloud, No-snow, Snow, Water) with different input combinations, and with different numbers of training data, i.e., 300 and 1000. The input combinations are denoted as:

- Sc_only
- Atmo_topo
- Dem
- Pca
- Pca_plus (only for the images with best and worst accuracies), and
- Pca_plus2 (only for the images with best and worst accuracies),

which are described in detail in Chapter 3. The full areas of the tiles are then classified with the input combination of *Pca*.

In the subsequent sections, the atmospheric and topographic correction assessment is presented, the classified images for each input combination are shown, and finally, the accuracy assessment results are discussed.

A flowchart of the discussions that are derived from the obtained results are presented in Figure 4.1.

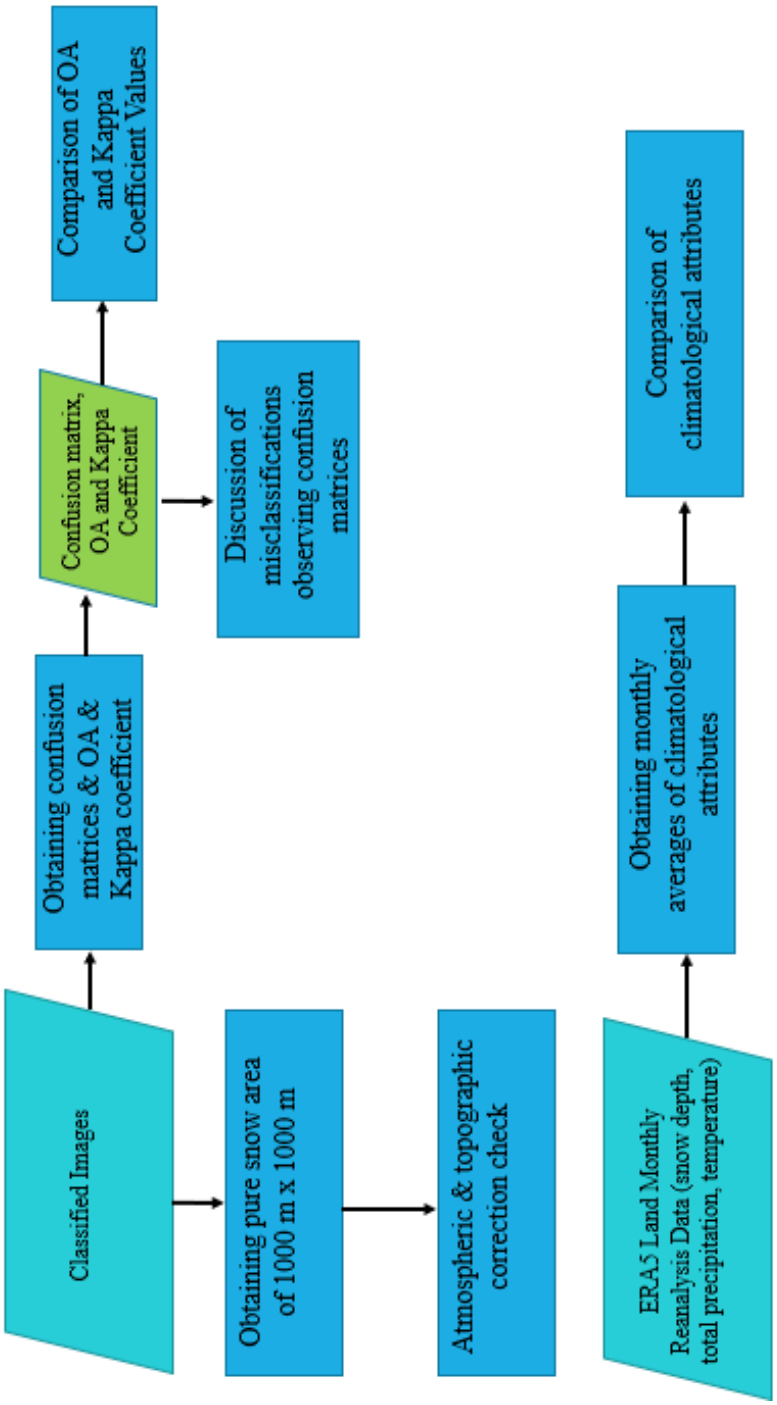


Figure 4.1. Flowchart of discussions

4.1 Atmospheric and Topographic Correction Assessment

The two fundemental effects of the atmosphere, i.e. absorption and scattering, should be corrected in remotely sensed images in order to obtain more accurate analysis results (Lu et al., 2002). The topographic correction, similarly, should be employed to attain better results in classification of satellite data.Topographic correction fixes the irregularity of the terrain as well as the shadow areas due to the shape of the terrain (Riano et al., 2003). In this study, atmospheric and topographic correction have been applied with the help of Sen2Cor interface (cf. Section 3.2.2.1).

The study of Vanonckelen et al. (2013) validates the effect of atmospheric and topographic correction of land cover classes through comparing means of each land cover class and band wavelengths. In this study, a 1000 m × 1000 m pure snow area has been acquired from each of the 9 Sentinel 2 images, and then the mean reflectance values of the area in Band 2, Band 3 and Band 4 (i.e., visible bands) of raw and atmospherically and topographically corrected data are compared, with the aid of a Python script (cf. Appendix E). The results of the comparisons can be observed in Figure 4.2 -Figure 4.4.

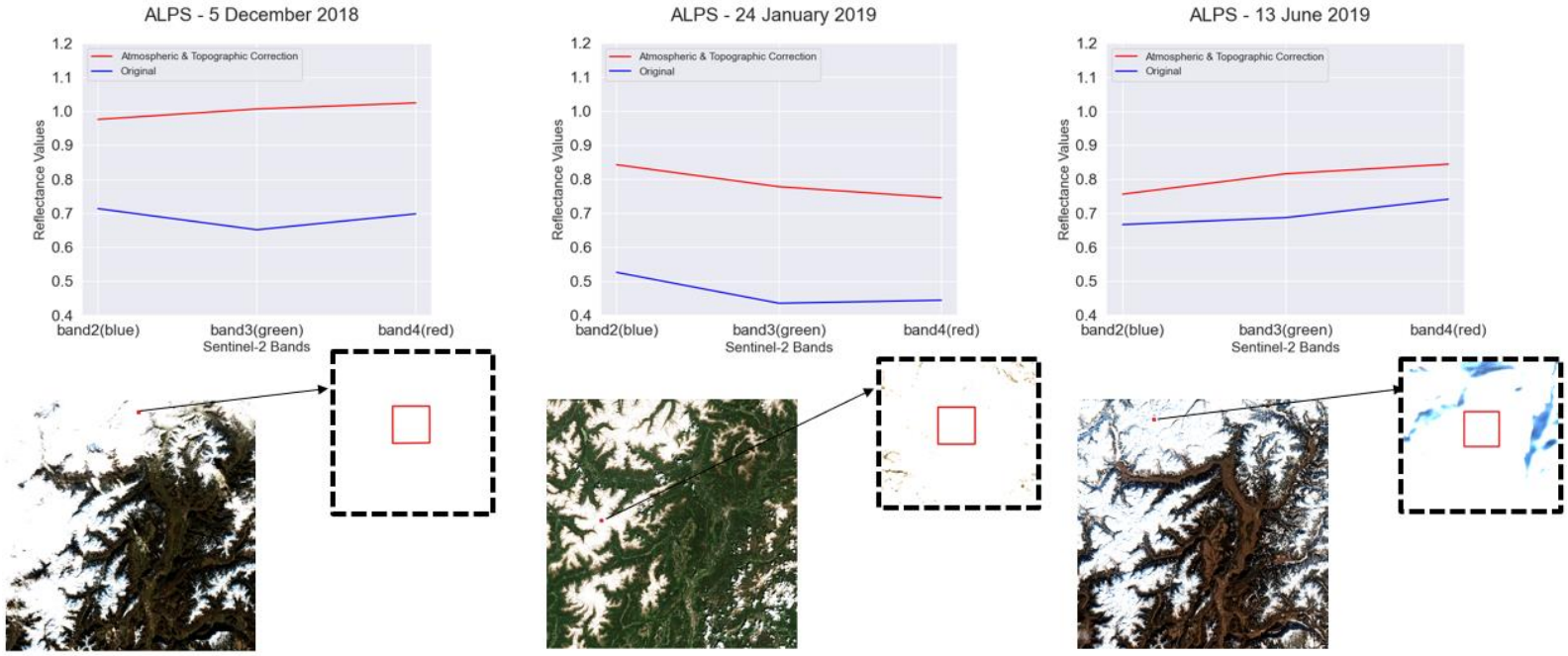


Figure 4.2. The comparison of raw and atmospherically and topographically corrected data of bands 2-4 over Alps Region.

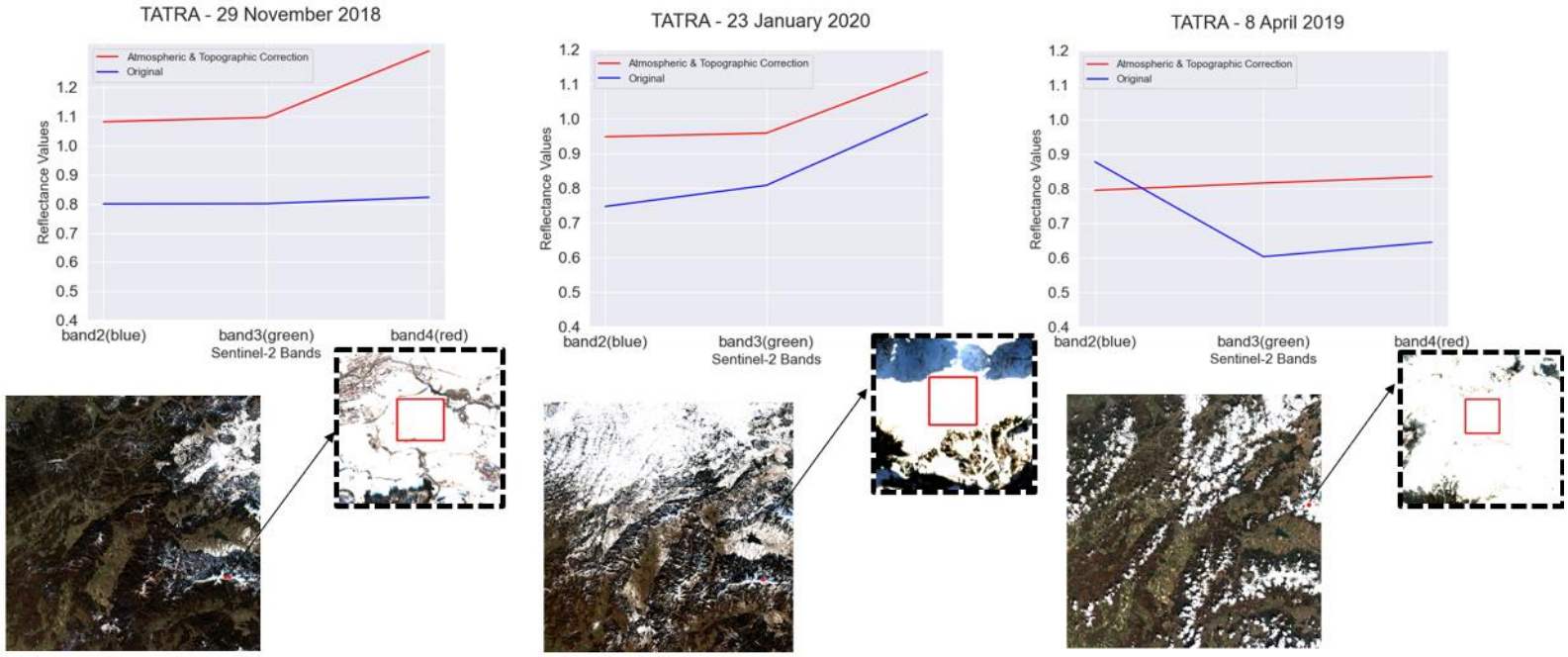


Figure 4.3. The comparison of raw and atmospherically and topographically corrected data of bands 2 -4 over Tatra Mountains.

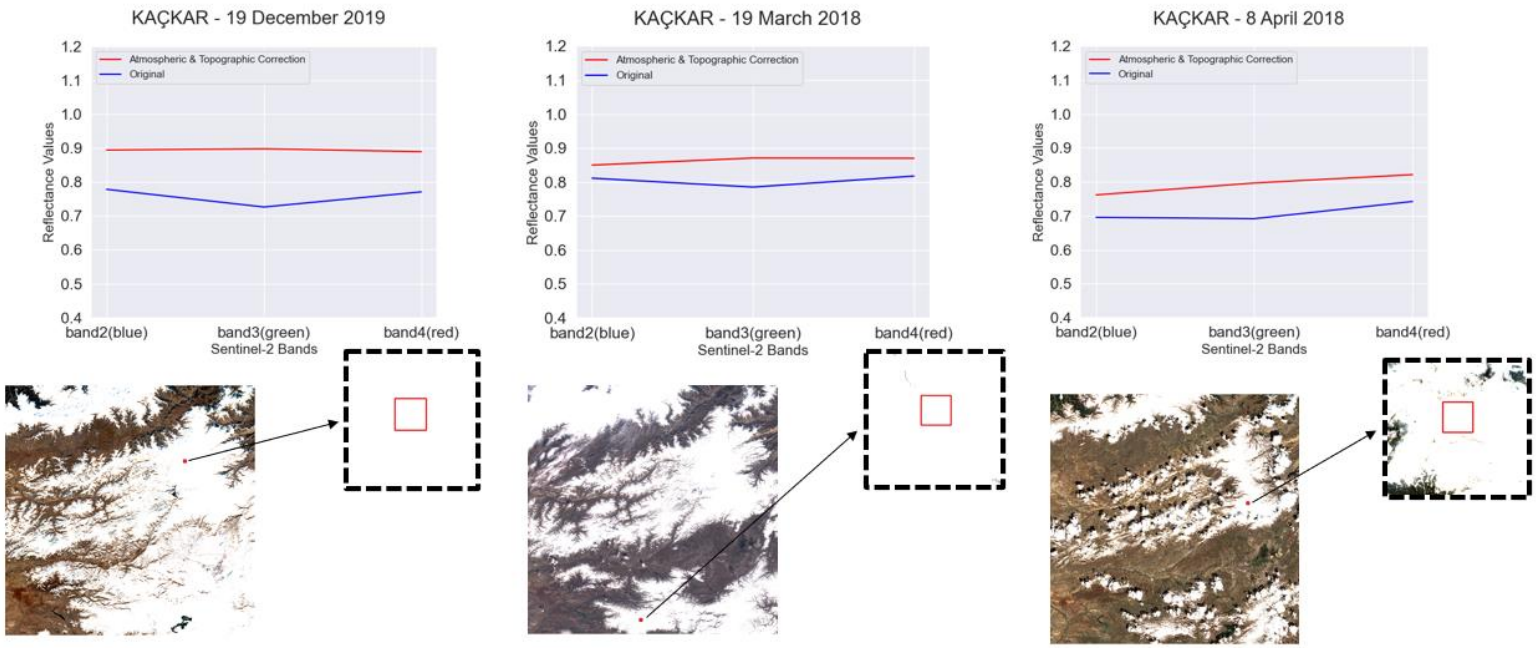


Figure 4.4. The comparison of raw and atmospherically and topographically corrected data of bands 2-4 over Kaçkar Mountains.

Snow can possess a reflectance up to or over 80% in the visible bands (Hall et al., 2002). A study conducted by Singh & Chaudhary (2010) examines the reflectance of snow with differing grain sizes using hyperspectral satellite imagery and the resultant reflectance profiles are given in Figure 4.5.

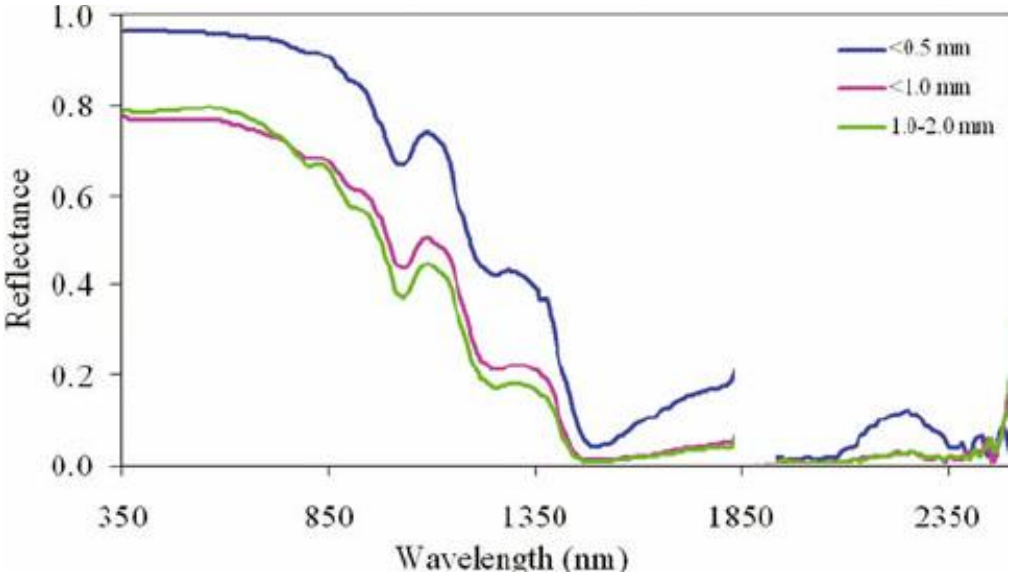


Figure 4.5. Reflectance values of snow with varying grain sizes (Singh & Chaudhary, 2010)

The results in Figure 4.2 - Figure 4.4 show that the data have been atmospherically and topographically corrected, given that results are similar to what is obtained by Singh & Chaudhary (2010) for visible bands in Figure 4.5.

4.2 Classification Results

The classified images that are 20 km × 20 km spatial subsets of Sentinel 2 tiles for all input combinations, as well as the whole tiles classified with *Pca* input combination are presented in this section, side by side with the original false-color RGB images (R: band 11, G: band 8A, B: band 3).

The classified tiles of Alpine Region are shown in Figure 4.6 - Figure 4.8. The classified images of Tatra Mountains are presented in Figure 4.9 - Figure 4.11. The classified subsets along with the whole classified tile of Kaçkar Mountains are given in Figure 4.12 - Figure 4.14

The best accuracy is achieved with the *Pca* input combination with 1000 training samples for each class (i.e., method *Pca_1000*) in most of the scenes. Thus, the whole tiles are classified with the input combination of *Pca_1000*. The details of the accuracy assessment results are discussed in the next section (i.e., Section 0).

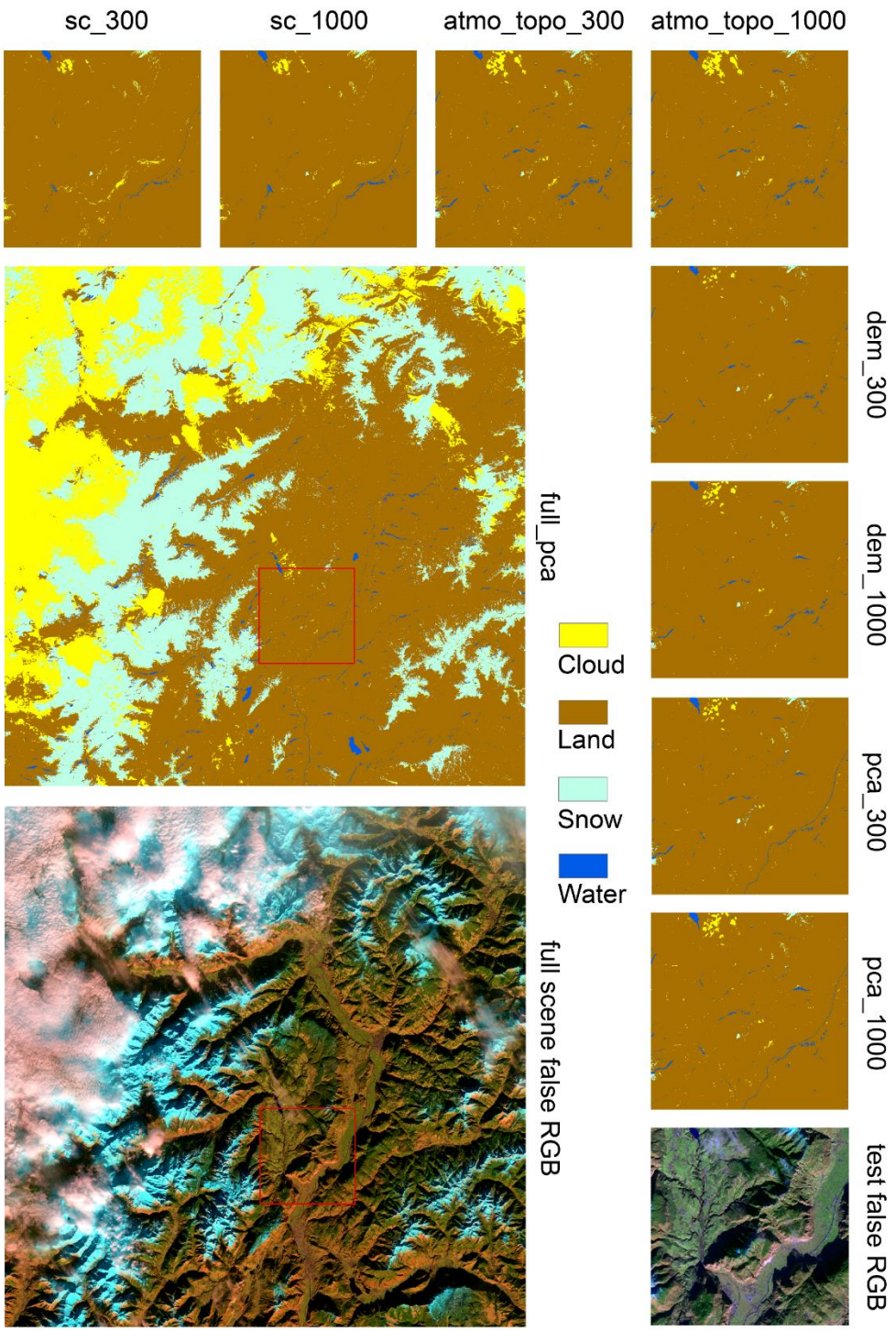


Figure 4.6. Classified images of partial tiles of Alps Region (5 December 2018) for different input combinations

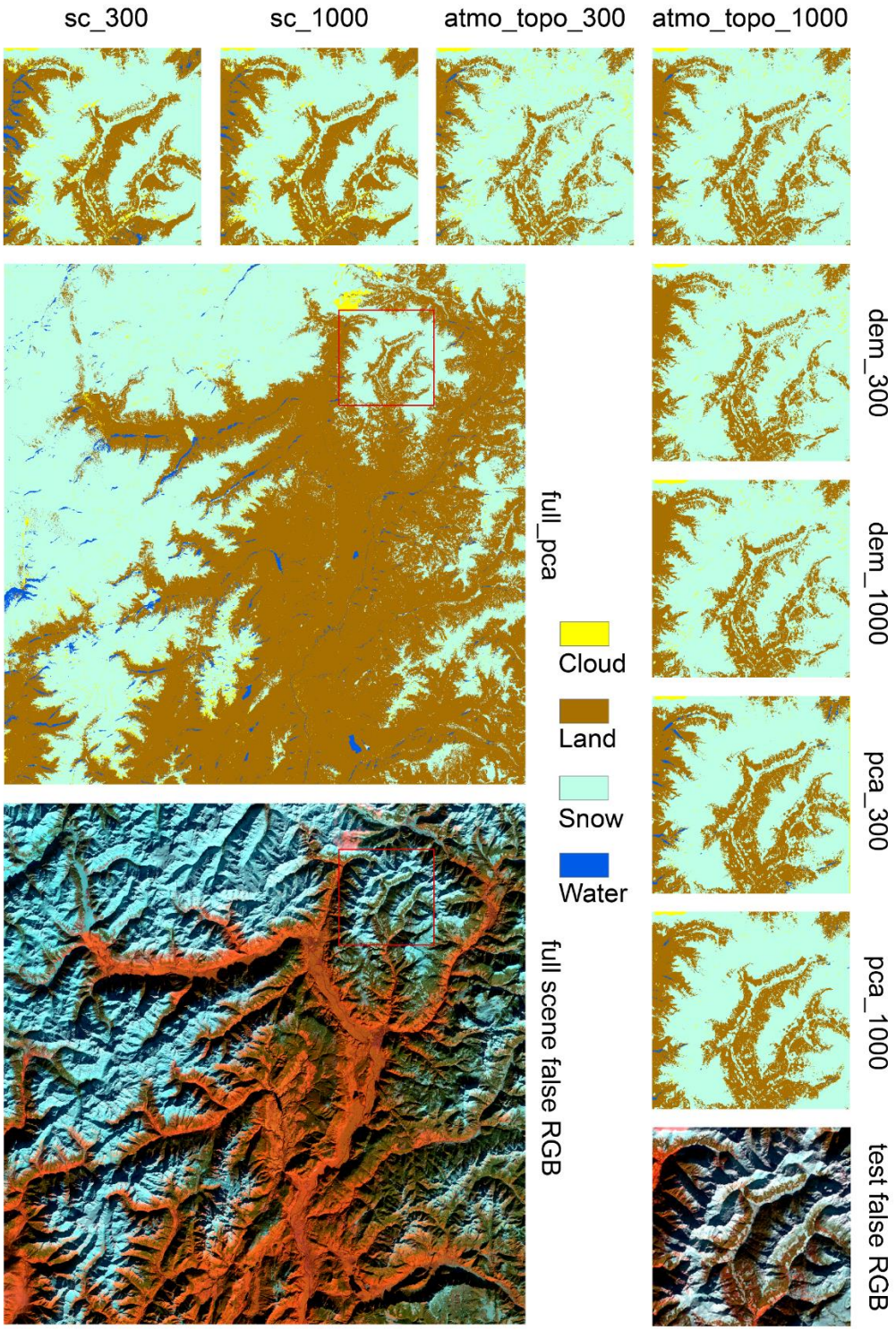


Figure 4.7. Classified images of partial tiles of Alps Region (24 January 2019) for different input combinations

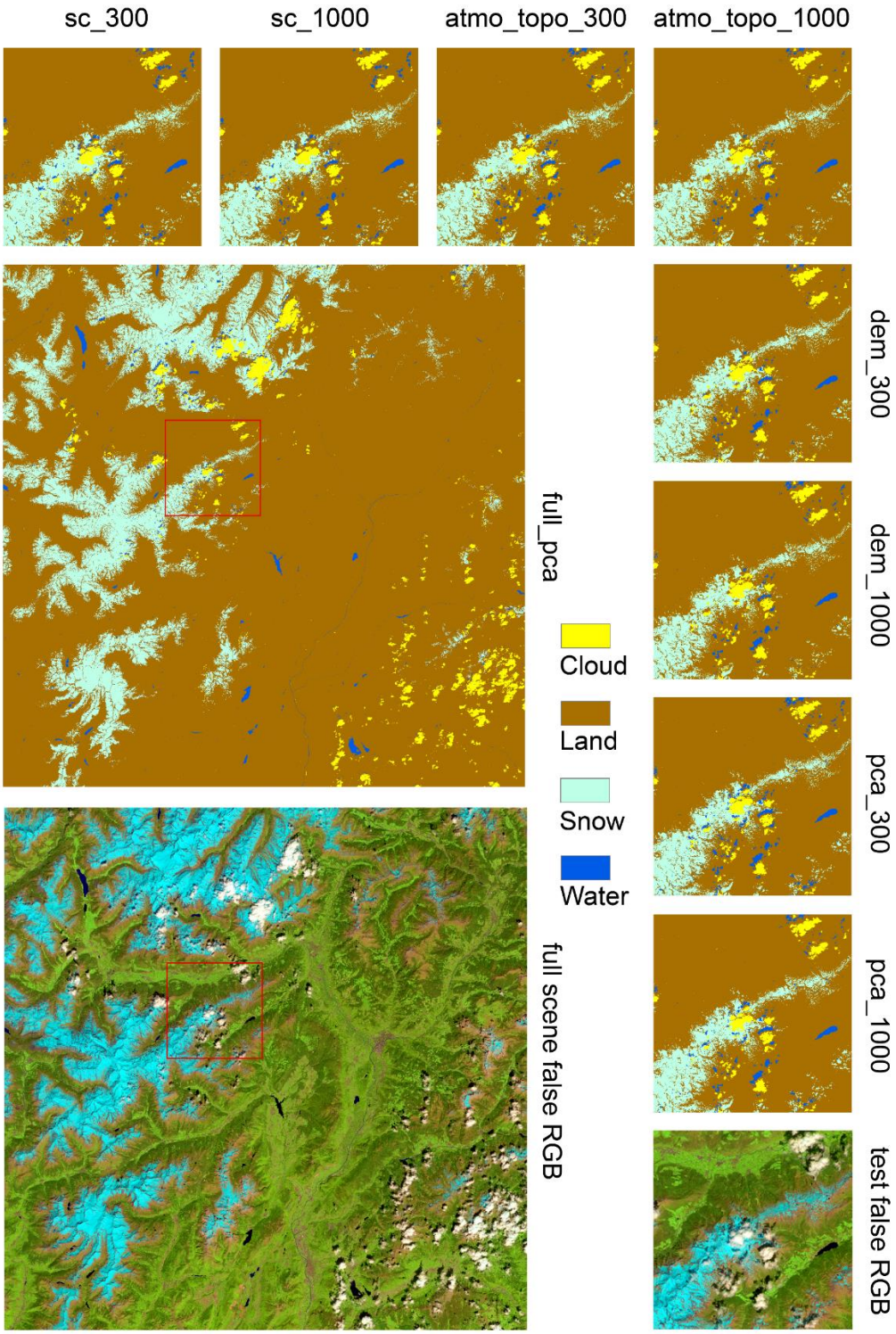


Figure 4.8. Classified images of partial tiles of Alps Region (13 June 2019) for different input combinations

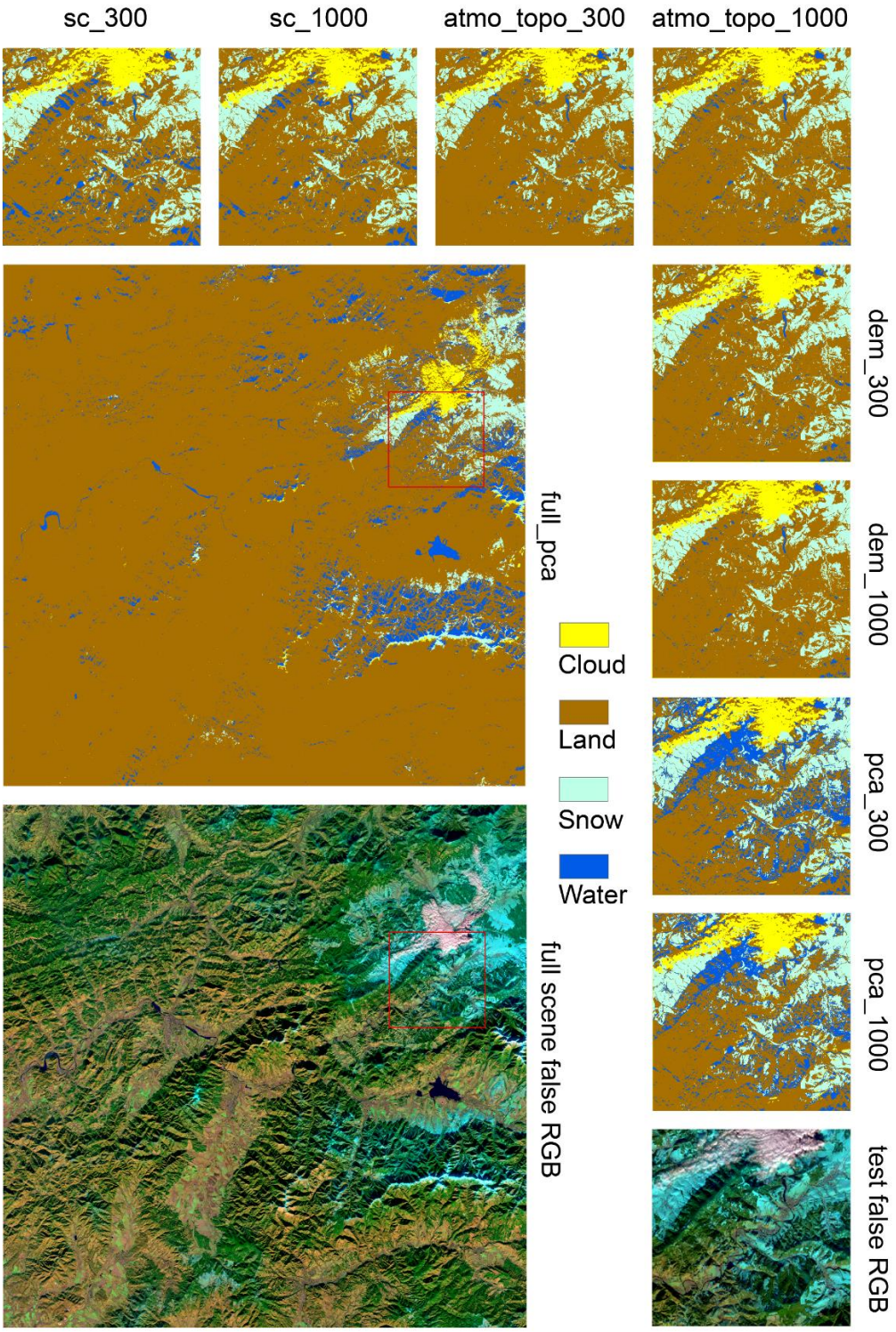


Figure 4.9. Classified images of partial tiles of Tatra Mountains (29 November 2018) for different input combinations

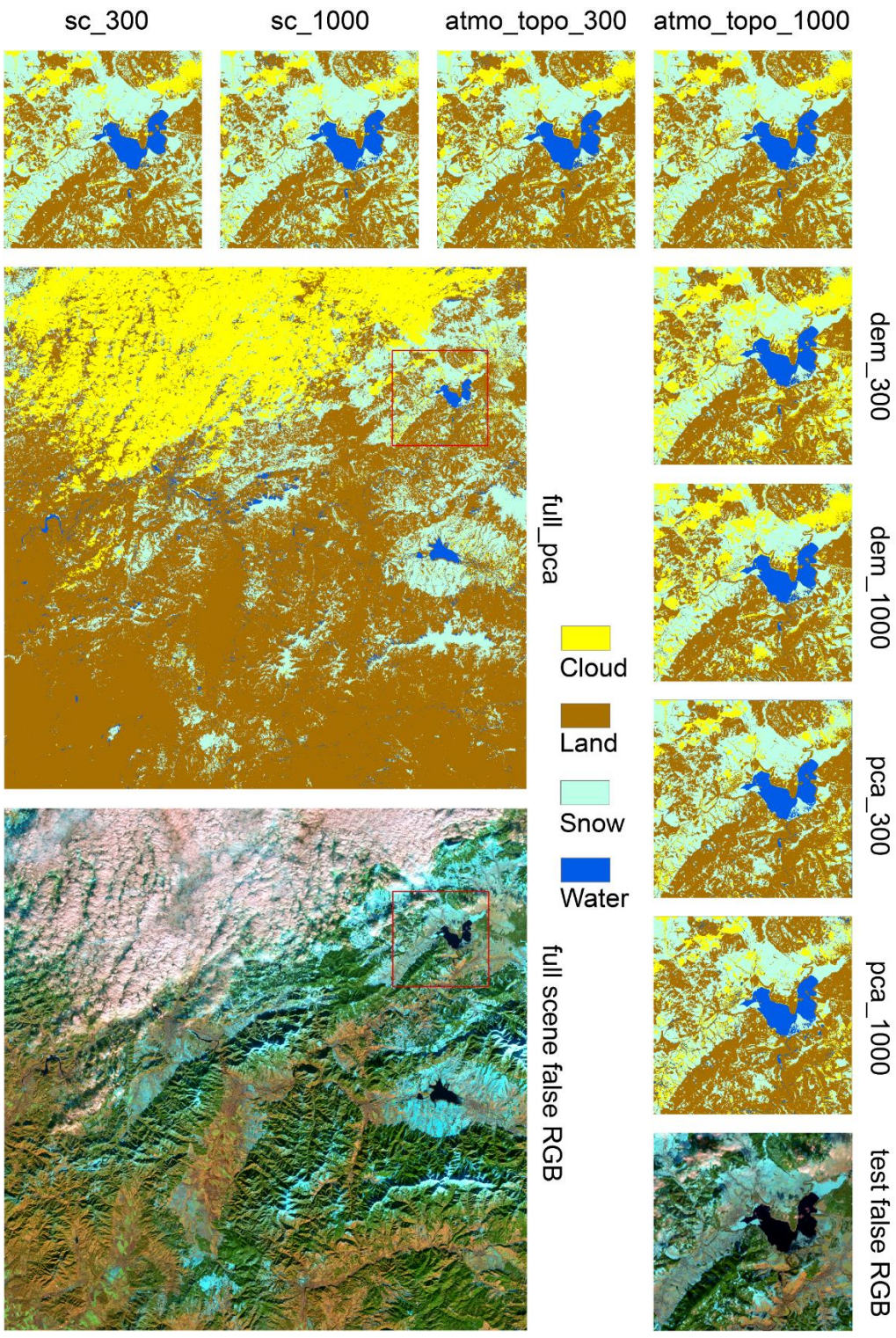


Figure 4.10. Classified images of partial tiles of Tatra Mountains (23 January 2020) for different input combinations

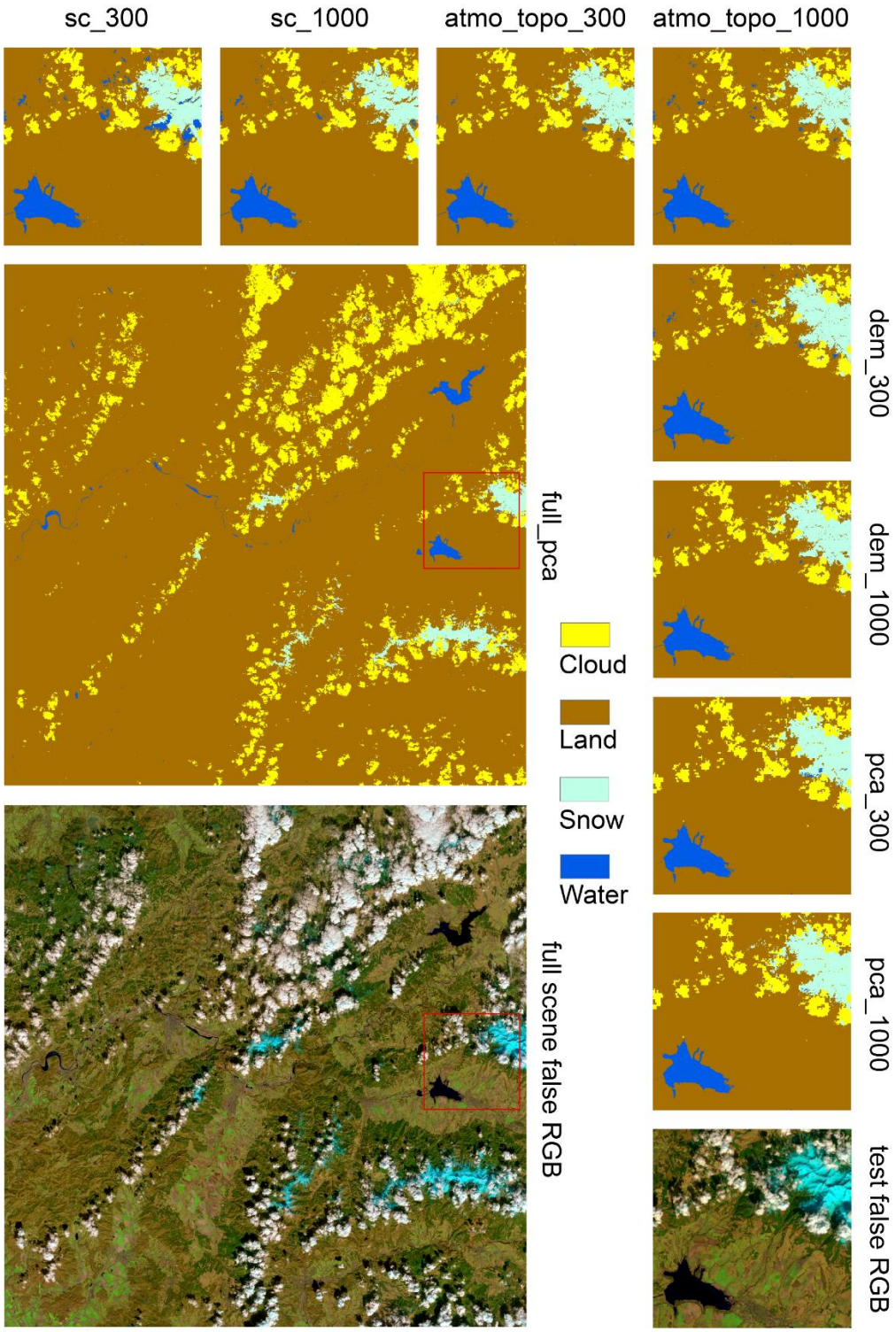


Figure 4.11. Classified images of partial tiles of Tatra Mountains (8 April 2019) for different input combinations

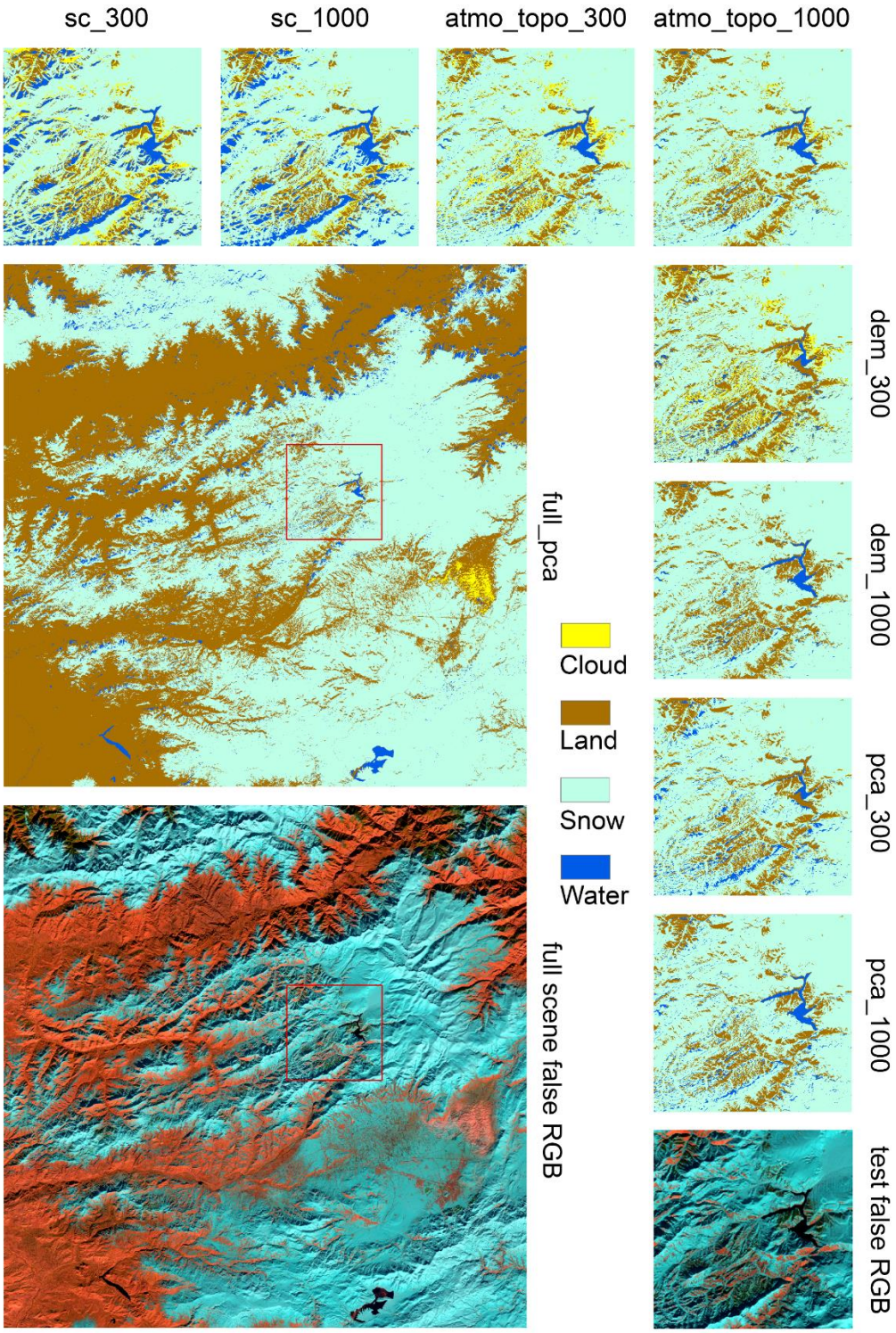


Figure 4.12. Classified images of partial tiles of Kaçkar Mountains (19 December 2019) for different input combinations

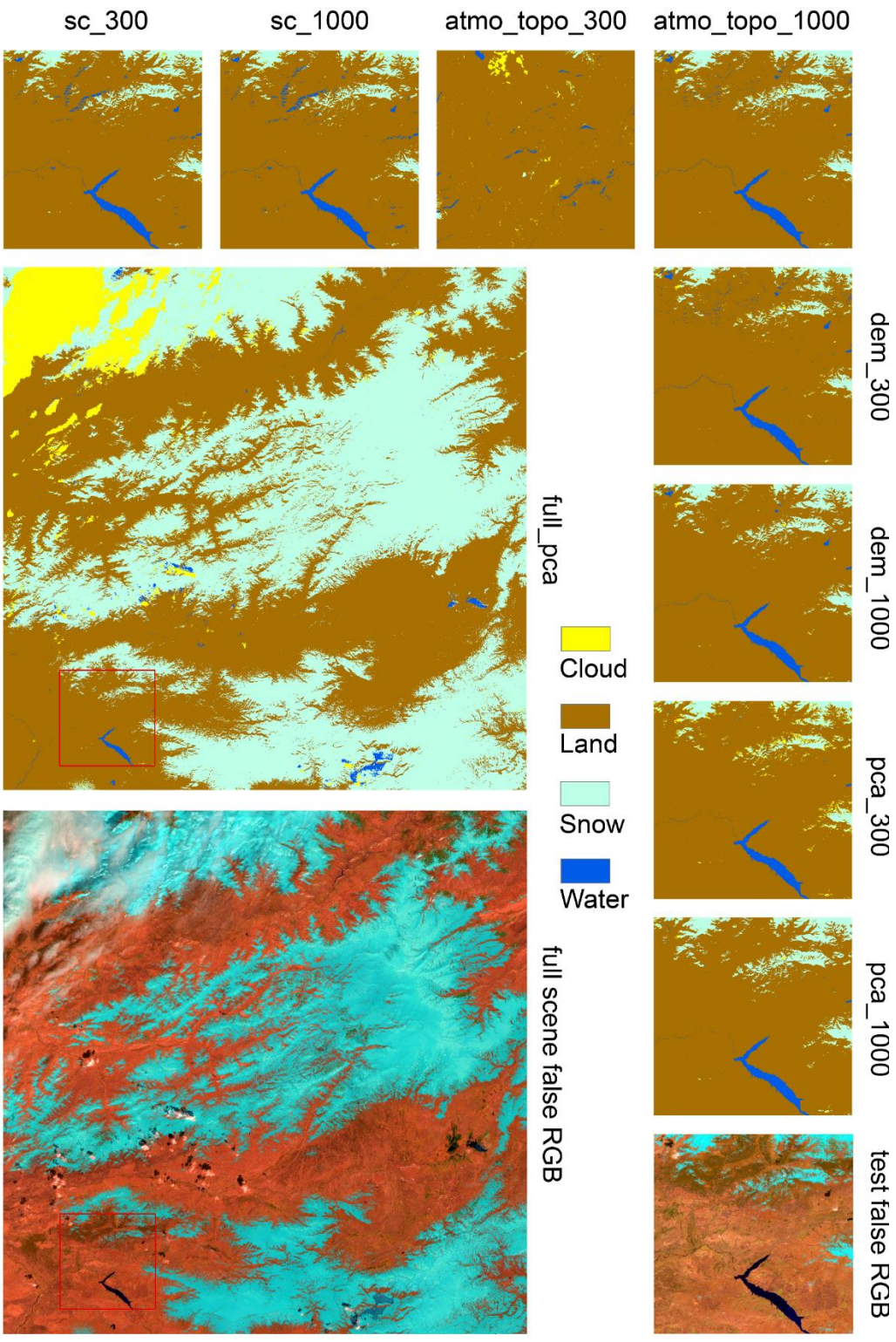


Figure 4.13. Classified images of partial tiles of Kaçkar Mountains (19 March 2018) for different input combinations

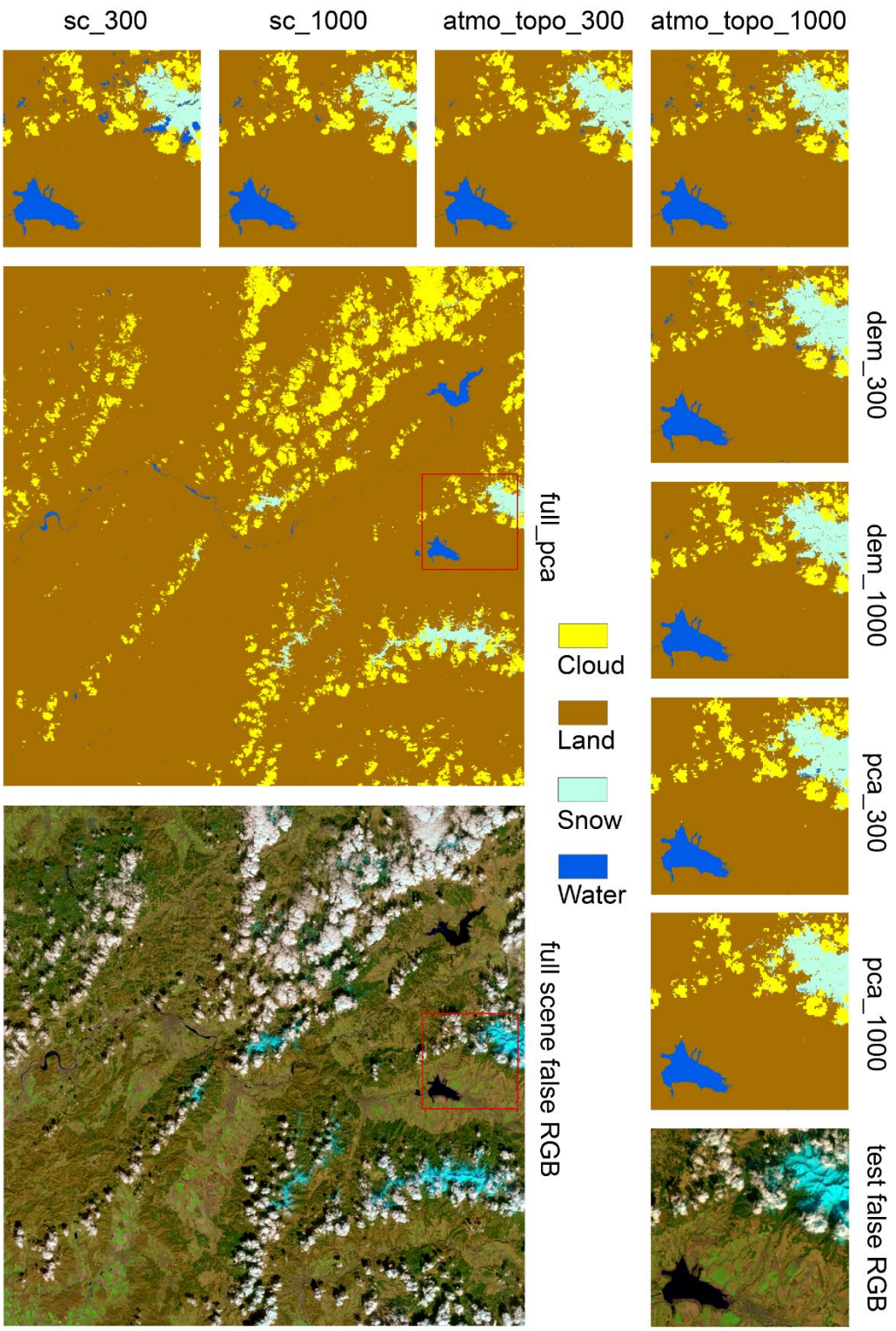


Figure 4.14. Classified images of partial tiles of Kaçkar Mountains (8 April 2018) for different input combinations

4.3 Accuracy Assessment Results

The accuracy assessment is conducted by collecting random test data using SCL of Sen2Cor and with the aid of ArcMap 10.7 software. The number of test data is sampled by multinominal distribution formula, and the details are explained in Section 3.6.1. The test data in each spatial subsample of 20 km × 20 km area are represented in Table 4.1, whereas the sample sizes of the test data employed in the accuracy assessment of full tiles are shown in Table 4.2.

Table 4.1 Number of test samples per class for 20 km × 20 km subsets of Sentinel-2 images, calculated by multinomial distribution formula (Ozdarici Ok & Akyurek, 2012; Jensen, 2006)

Location	Date	Number of Points (classes)				
		Cloud	No snow (Land)	Snow	Water	Total
ALPS REGION	5 December 2018	100	415	100	100	745
	24 January 2019	100	100	360	100	755
	13 June 2019	100	450	120	100	660
TATRA MOUNTAINS	29 November 2018	100	100	100	100	440
	23 January 2020	100	205	270	100	740
	8 April 2019	100	440	100	100	710
KAÇKAR MOUNTAINS	19 December 2019	100	100	365	100	450
	19 March 2019	100	440	100	100	600
	8 April 2018	165	235	135	100	685

Table 4.2 Number of test samples per class for full Sentinel-2 tiles, calculated by multinomial distribution formula (Ozdarici Ok & Akyurek, 2012; Jensen, 2006)

Location	Date	Number of Points (classes)				
		Cloud	Nosnow (Land)	Snow	Water	Total
ALPS REGION	5 December 2018	105	125	130	100	545
	24 January 2019	100	450	100	100	625
	13 June 2019	100	435	100	100	595
TATRA MOUNTAINS	29 November 2018	100	230	100	100	670
	23 January 2020	120	210	105	100	655
	8 April 2019	100	450	100	100	690
KAÇKAR MOUNTAINS	19 December 2019	100	140	430	100	735
	19 March 2019	100	250	345	100	750
	8 April 2018	100	425	100	100	740

The minimum number of test samples for each class is 100, as explained in Section 3.6.1. The classified data for each input combination along with the random test data are then used for the accuracy assessment. The confusion matrix and the obtained OA and Kappa coefficient metrics (cf. Section 3.6.2), are given in Figure 4.15-Figure 4.23 for the corresponding spatial subset, for the input combinations presented in Table 3.2 (*Sc_only, Atmo_topo, Dem, Pca*). Both confusion matrices and OA and Kappa coefficient values have been obtained by the Python script presented in Appendix F.

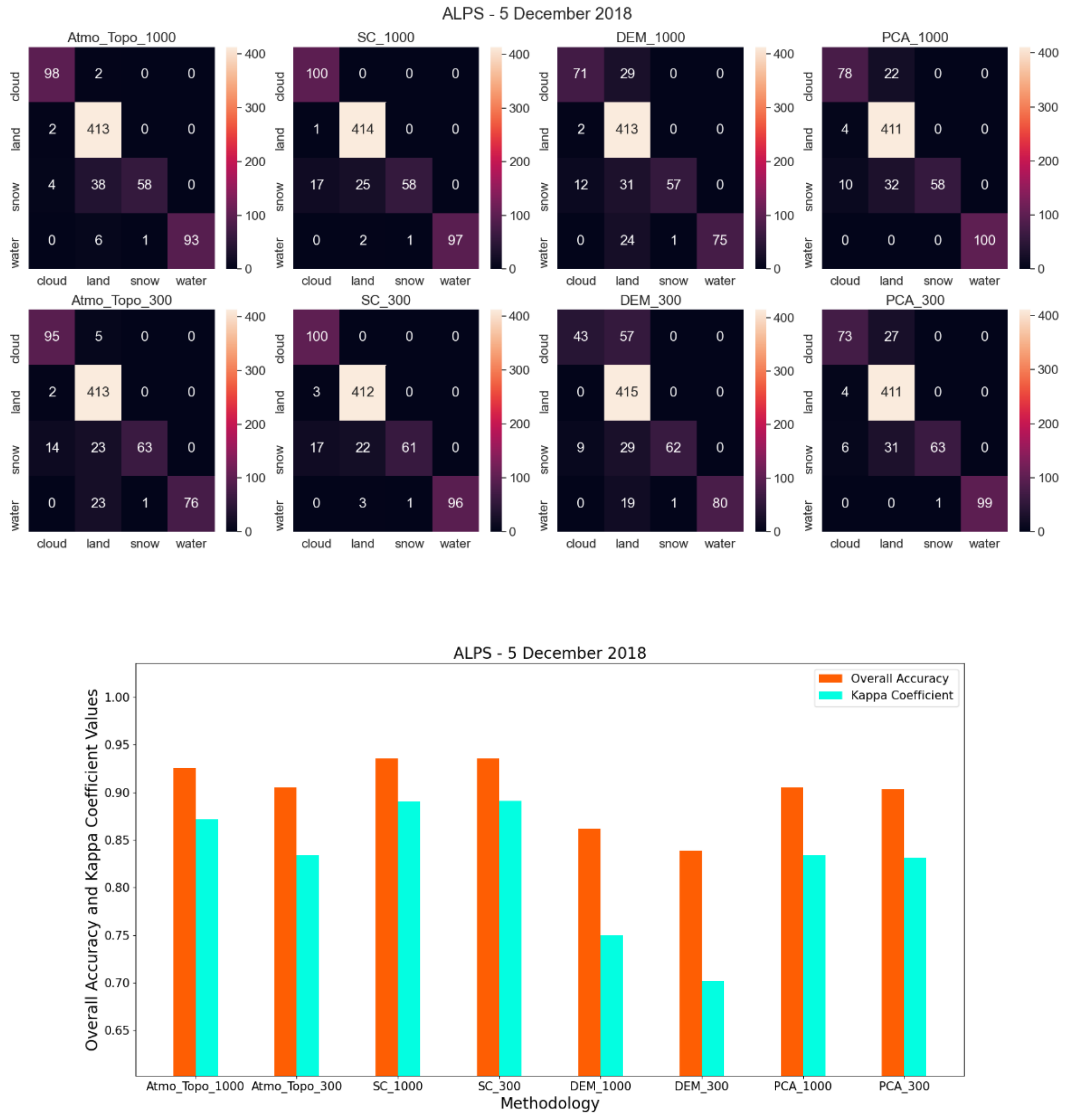


Figure 4.15. Confusion matrices, OA and Kappa coefficient values of spatial subset for Alps Region on 5 December 2018 for main input combinations

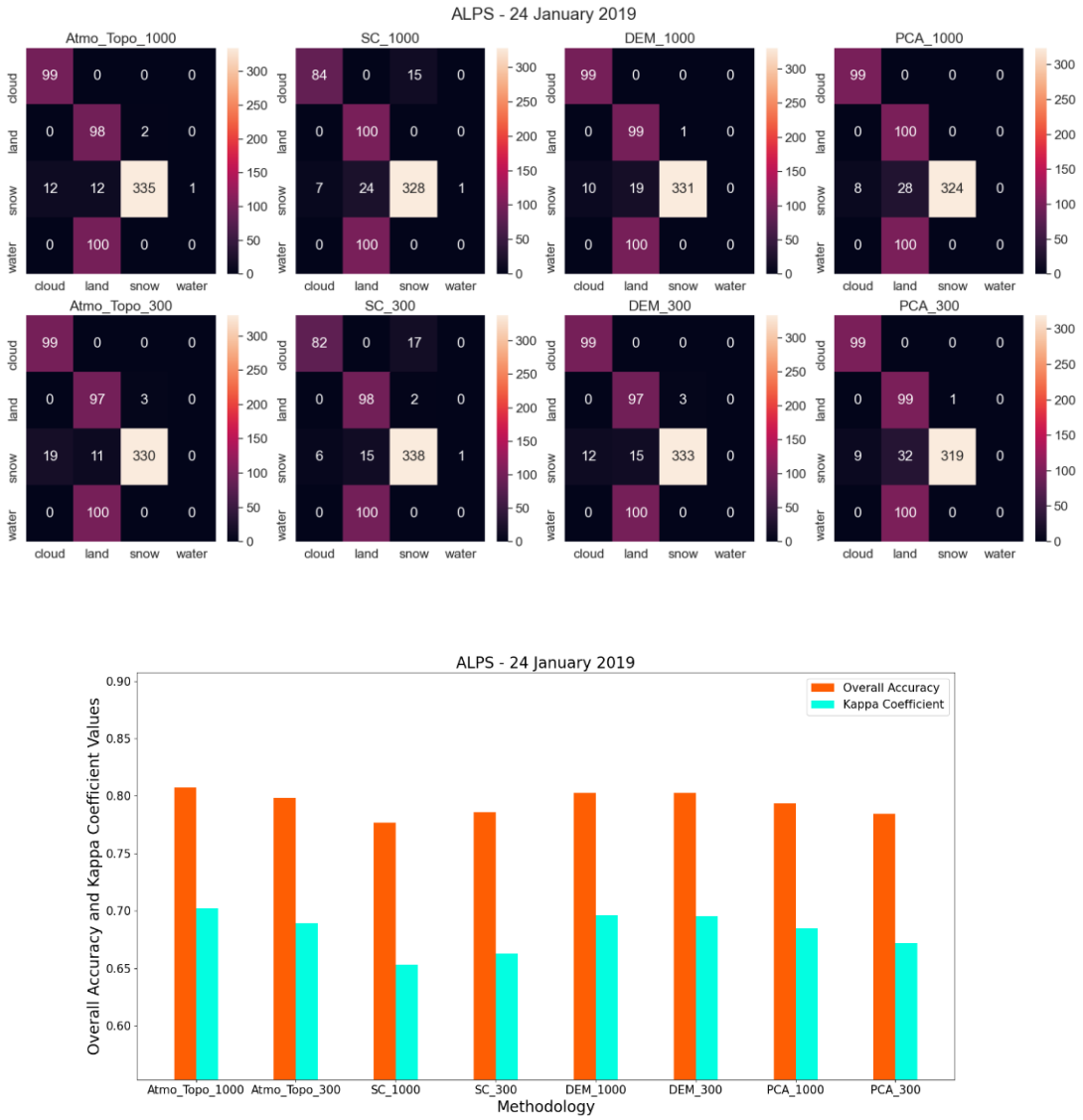


Figure 4.16. Confusion matrices, OA and Kappa coefficient values of spatial subset for Alps Region on 24 January 2019 for main input combinations

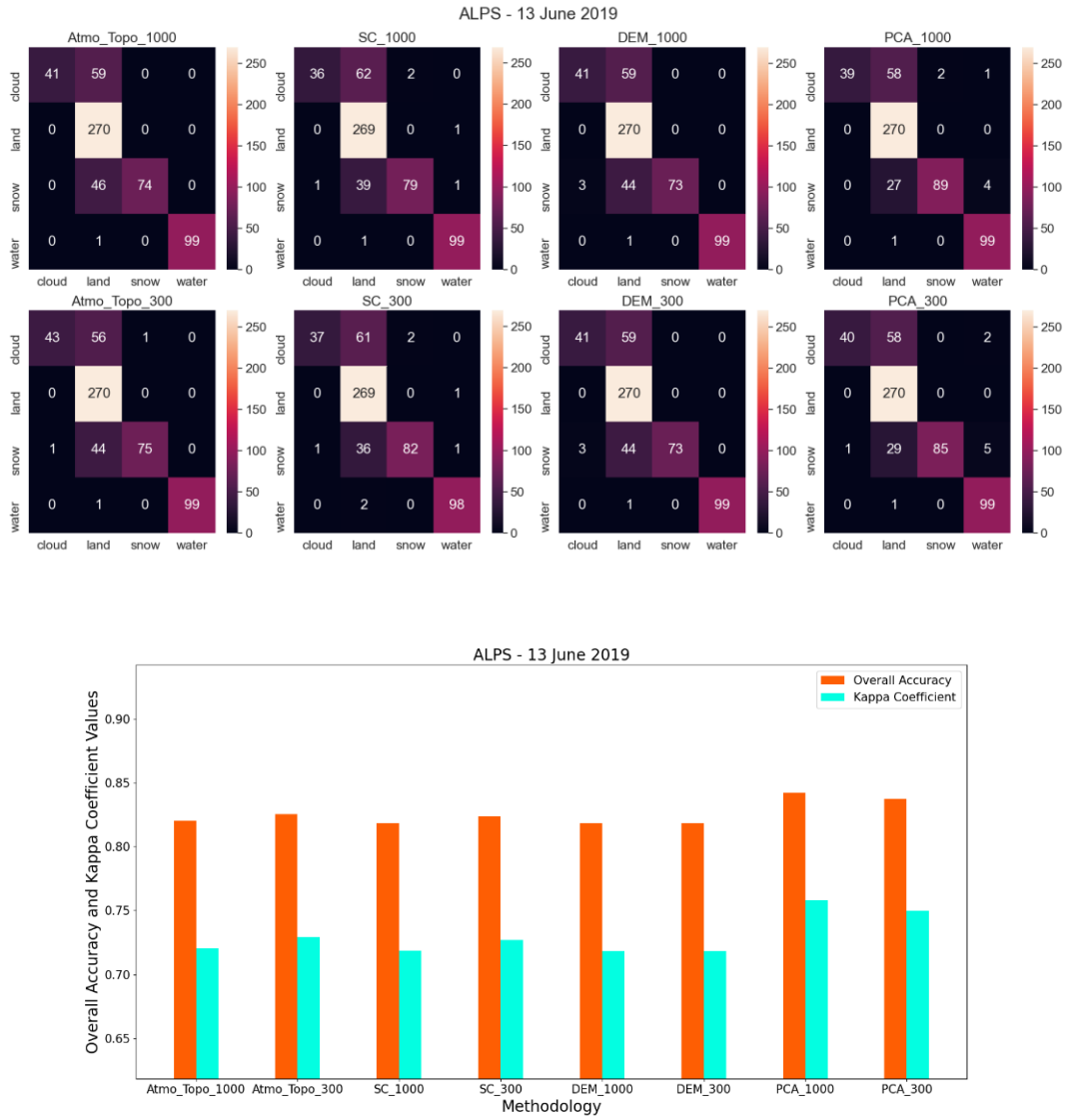


Figure 4.17. Confusion Matrices, OA and Kappa coefficient values of spatial subset for Alps Region on 13 June 2019 for main input combinations

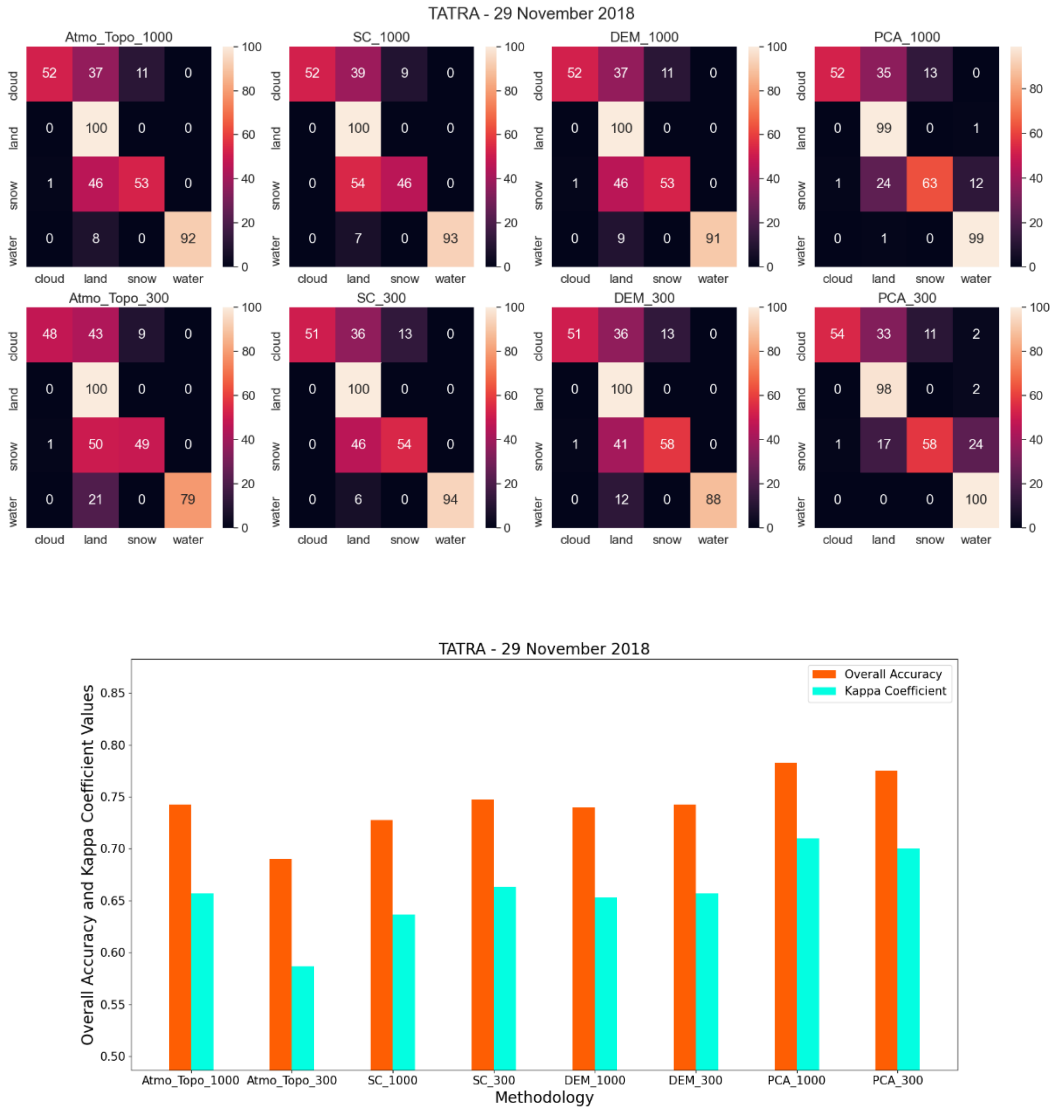


Figure 4.18. Confusion matrices, OA and Kappa coefficient values of spatial subset for Tatra Mountains on 29 November 2018 for main input combinations

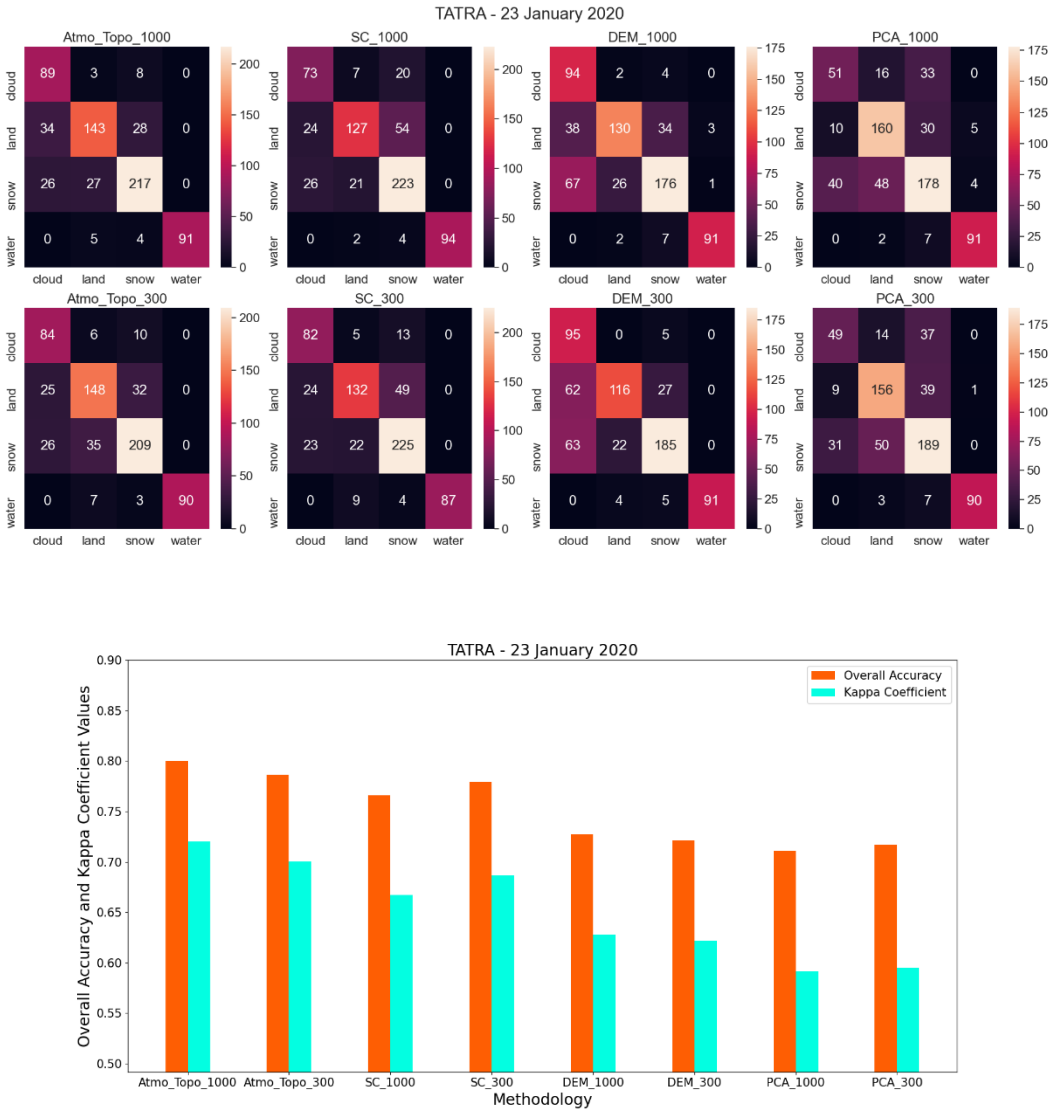


Figure 4.19. Confusion matrices, OA and Kappa coefficient values of spatial subset for Tatra Mountains on 23 January 2020 for main input combinations

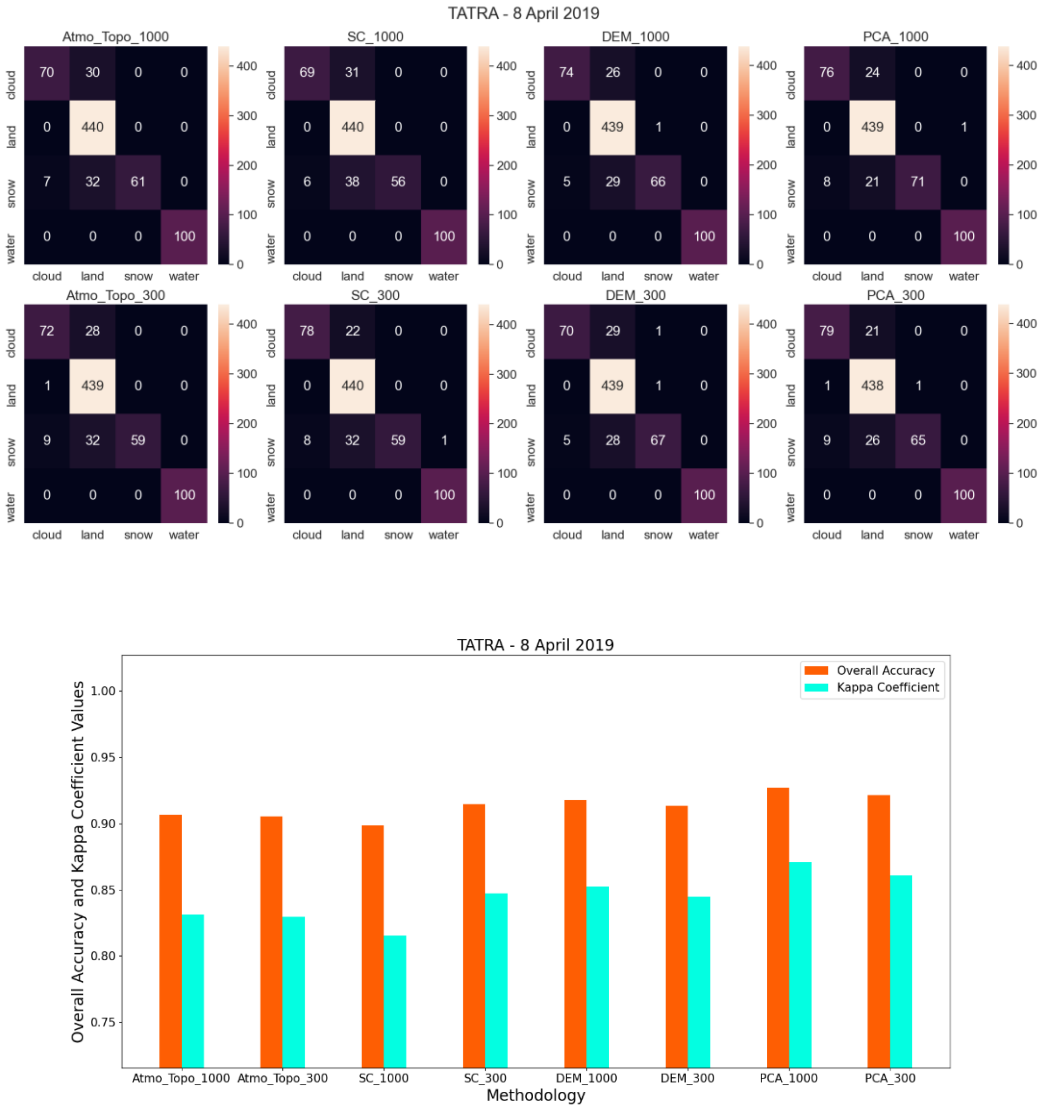


Figure 4.20. Confusion matrices, OA and Kappa coefficient values of spatial subset for Tatra Mountains on 8 April 2019 for main input combinations

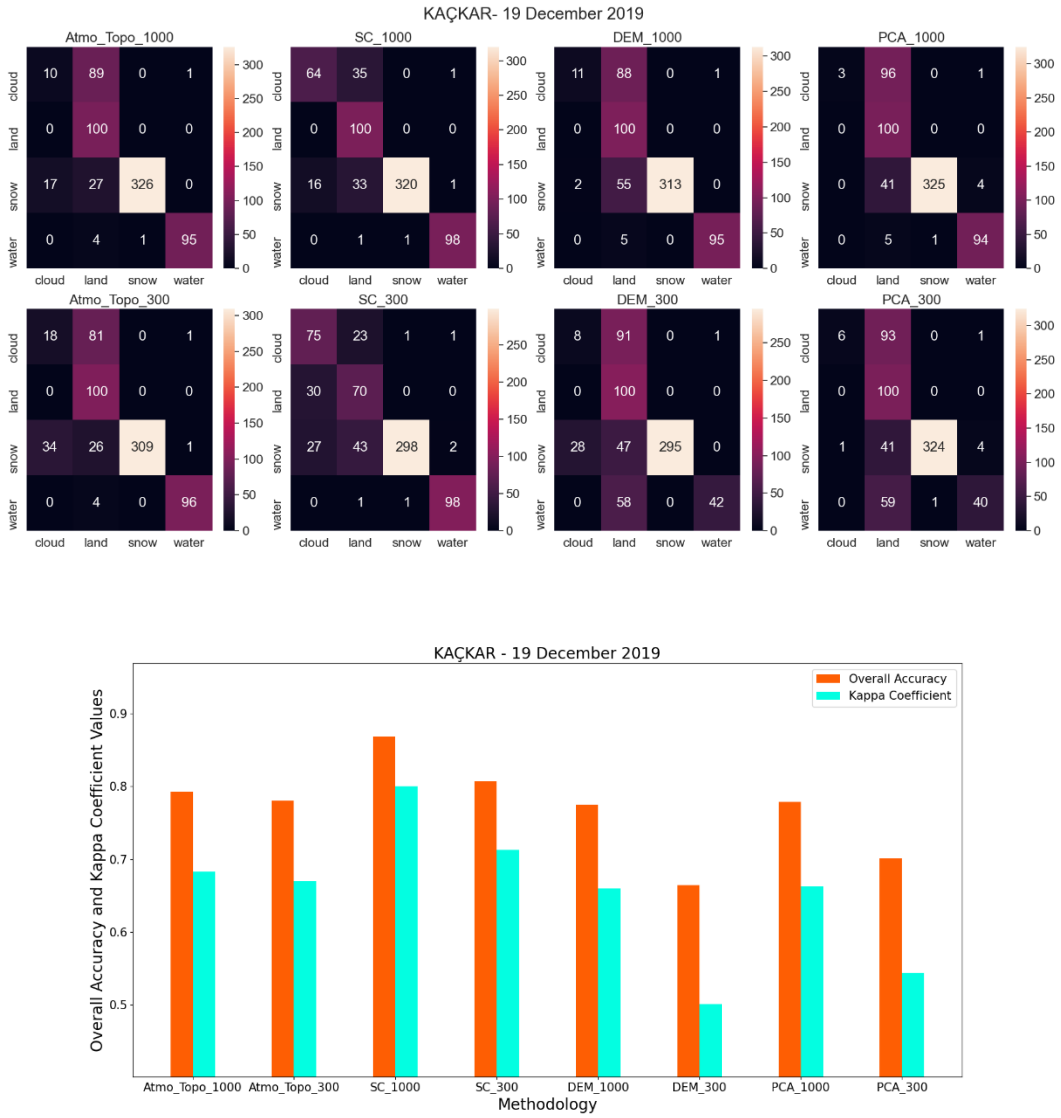


Figure 4.21. Confusion matrices, OA and Kappa coefficient values of spatial subset for Kaçkar Mountains on 19 December 2019 for main input combinations

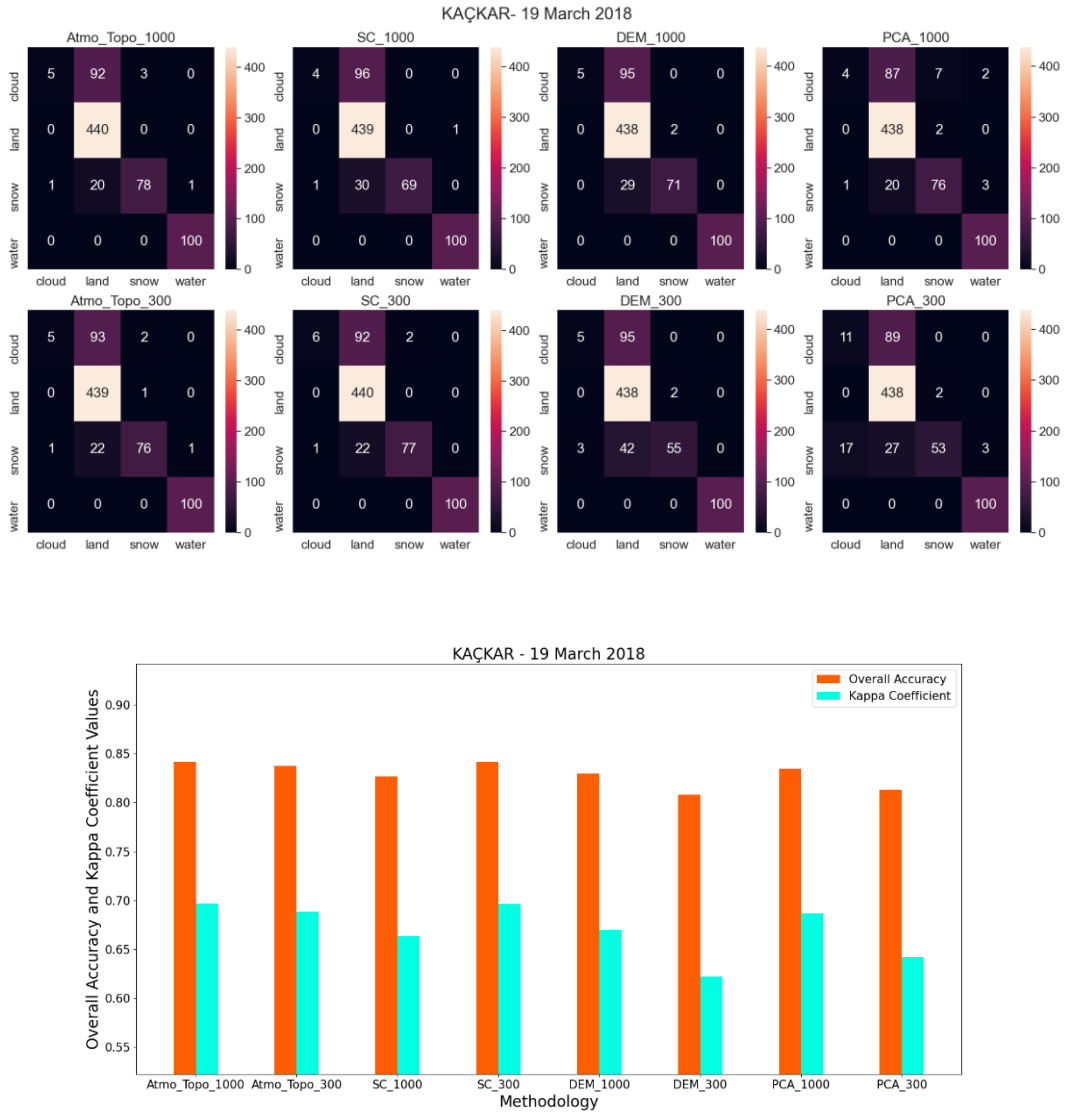


Figure 4.22. Confusion matrices, OA and Kappa coefficient values of spatial subset for Kaçkar Mountains on 19 March 2018 for main input combinations

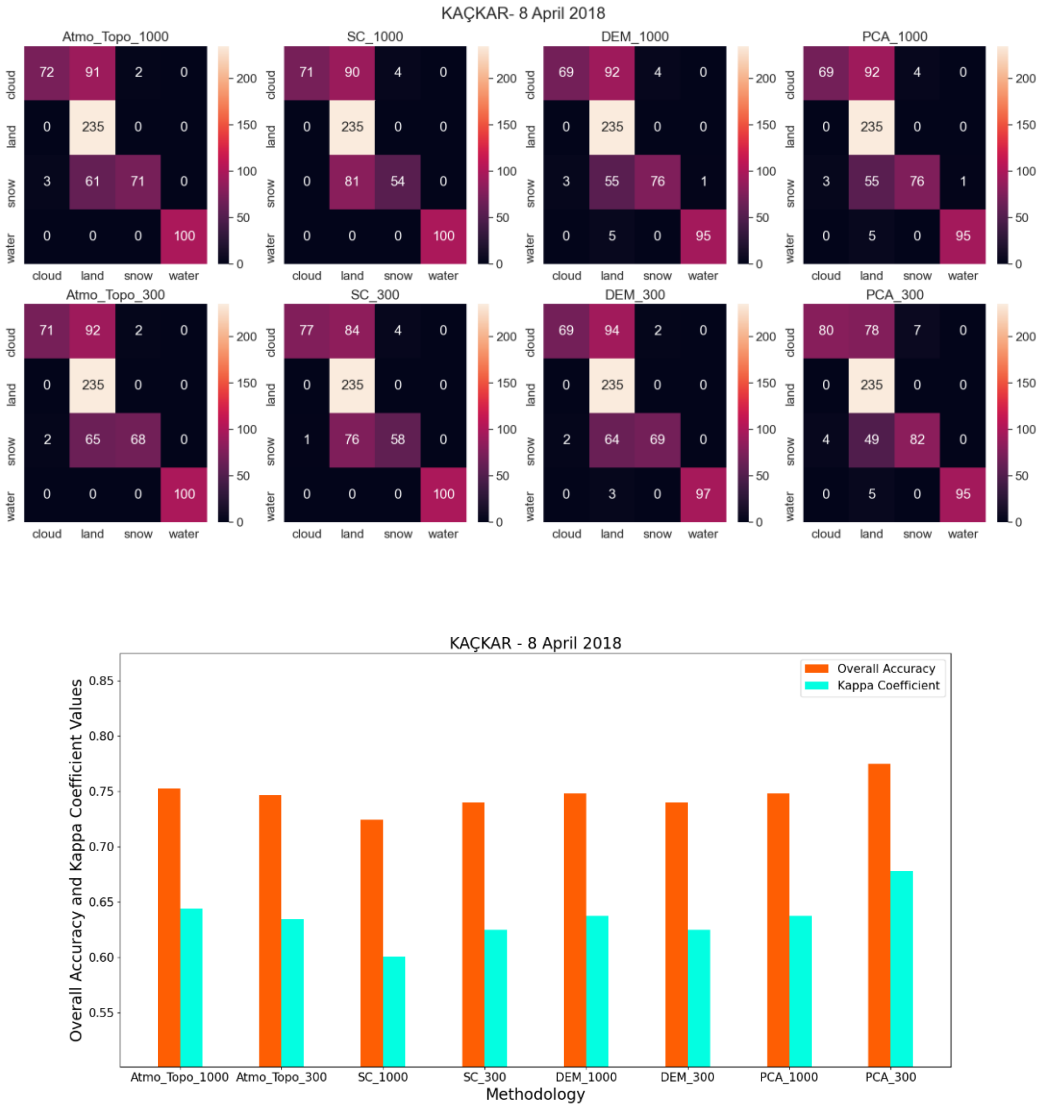


Figure 4.23. Confusion matrices, OA and Kappa coefficient values of spatial subset for Kaçkar Mountains on 8 April 2018 for main input combinations

For the additional input combinations that are shown in Table 3.3, the confusion matrices and the obtained OA and Kappa coefficients of two images that have yielded the best and worst accuracy results can be observed in Figure 4.24 and Figure 4.25.

The OA and Kappa coefficient results of all classified spatial subsets are shown in Table 4.3.

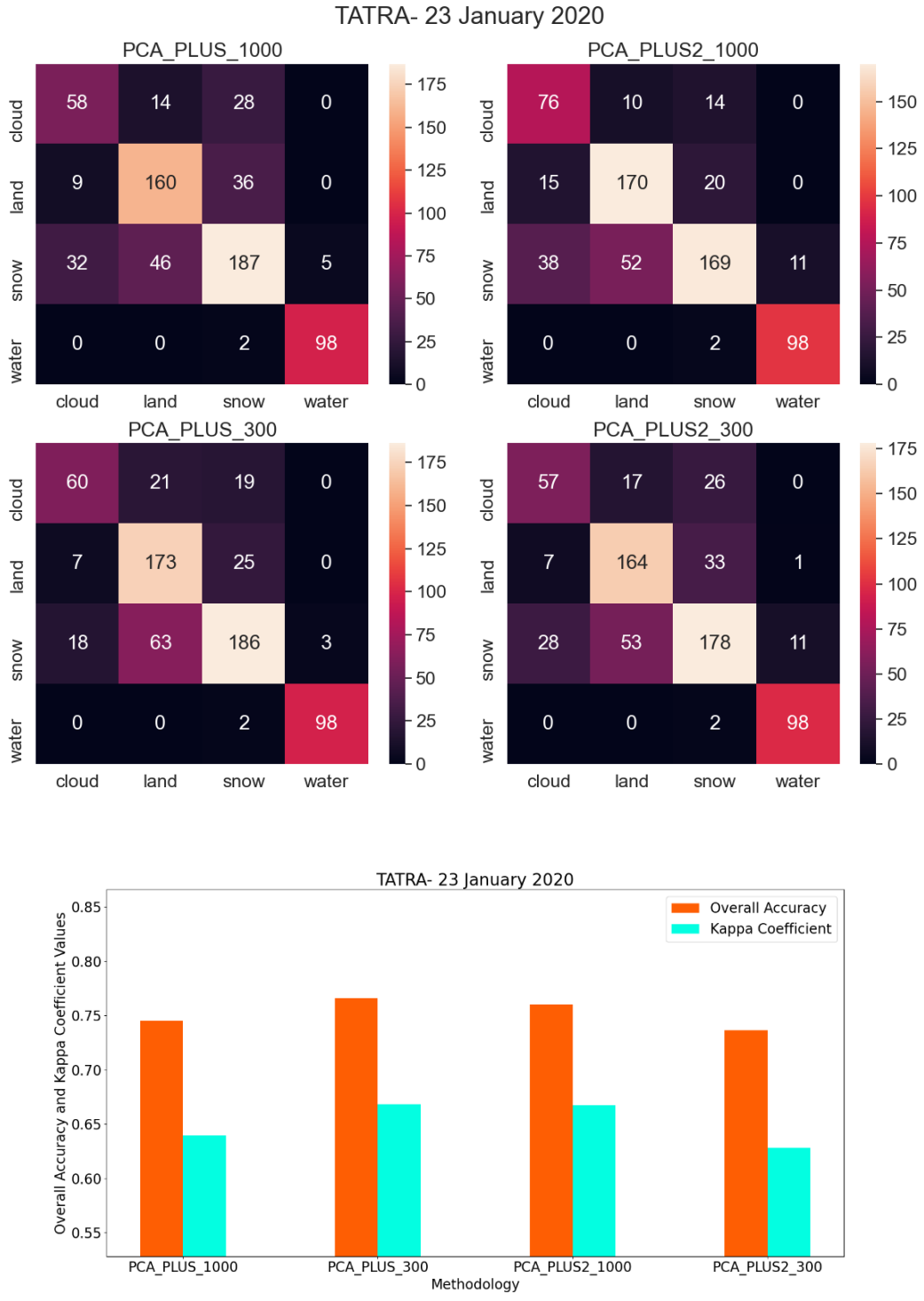


Figure 4.24 Confusion matrices, OA and Kappa coefficient values of spatial subset for Kaçkar Mountains on 8 April 2018 for main input combinations

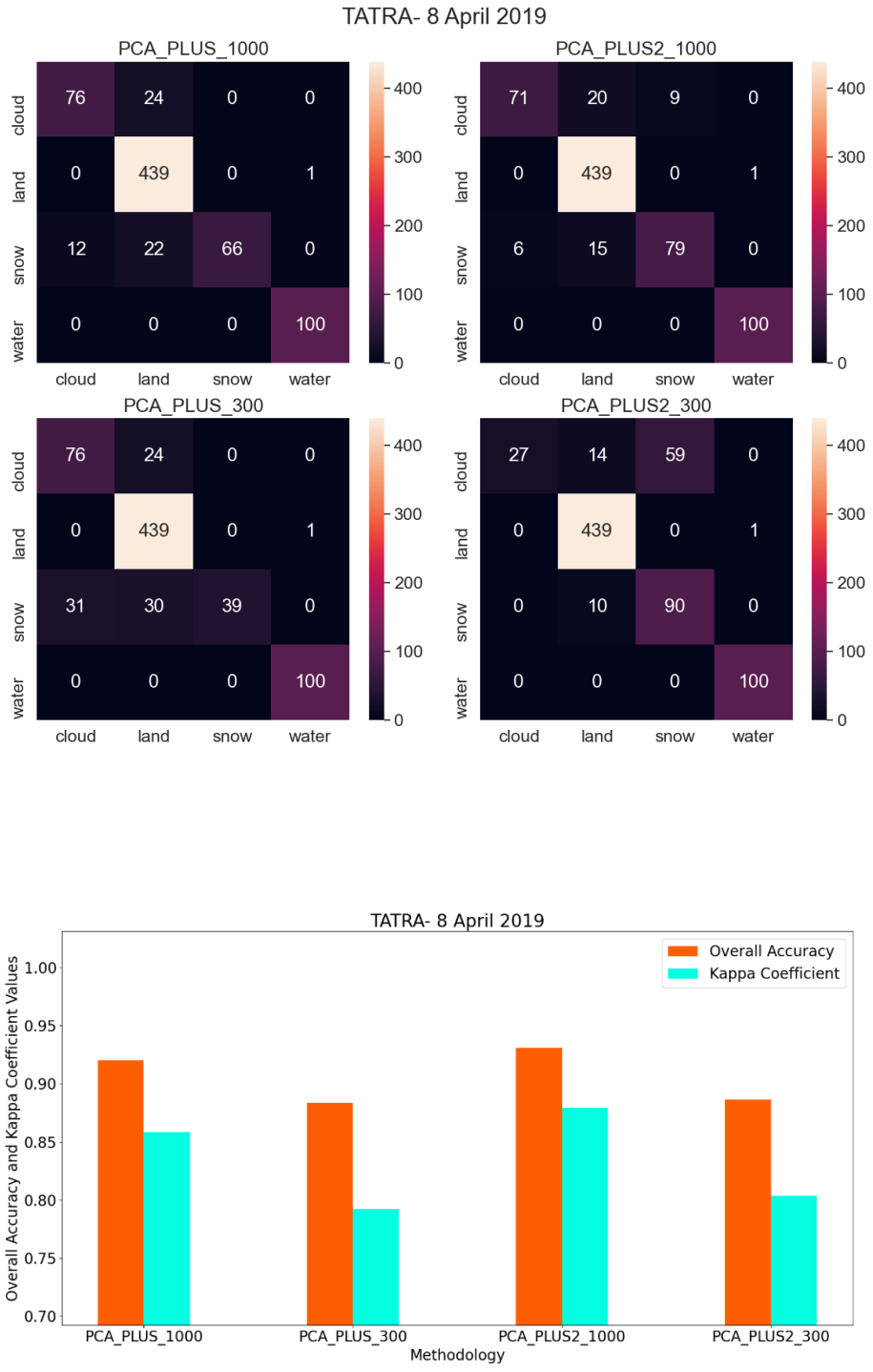


Figure 4.25. Confusion matrices, OA and Kappa coefficient values of spatial subset for Tatra Mountains on 8 April 2019 for additional input combination

Table 4.3 OA and Kappa coefficient values of 20 km × 20 km spatial subsets with respect to different input combinations.

Location	Date	Coefficient	Atmo_Topo_1000	Atmo_Topo_300	SC_1000	SC_300	DEM_1000	DEM_300	PCA_1000	PCA_300	PCA_PLUS_1000	PCA_PLUS_300	PCA_PLUS2_1000	PCA_PLUS2_300
ALPS REGION	5 December 2018	OA	0.93	0.90	0.94	0.94	0.86	0.84	0.90	0.90	-	-	-	-
		Kappa	0.87	0.83	0.89	0.89	0.75	0.70	0.83	0.83	-	-	-	-
	24 January 2019	OA	0.81	0.80	0.78	0.79	0.80	0.80	0.79	0.78	-	-	-	-
		Kappa	0.70	0.69	0.65	0.66	0.70	0.70	0.68	0.67	-	-	-	-
	13 June 2019	OA	0.82	0.83	0.82	0.82	0.82	0.82	0.84	0.84	-	-	-	-
		Kappa	0.72	0.73	0.72	0.73	0.72	0.72	0.76	0.75	-	-	-	-
TATRA MOUNTAINS	29 November 2018	OA	0.74	0.69	0.73	0.75	0.74	0.74	0.78	0.78	-	-	-	-
		Kappa	0.66	0.59	0.64	0.66	0.65	0.66	0.71	0.70	-	-	-	-
	23 January 2020	OA	0.80	0.79	0.77	0.78	0.73	0.72	0.71	0.72	0.75	0.77	0.76	0.74
		Kappa	0.72	0.70	0.67	0.69	0.63	0.62	0.59	0.60	0.64	0.67	0.67	0.63

Table 4.3 (continued)

Location	Date	Coefficient	Atmo_Topo_1000	Atmo_Topo_300	SC_1000	SC_300	DEM_1000	DEM_300	PCA_1000	PCA_300	PCA_PLUS_1000	PCA_PLUS_300	PCA_PLUS2_1000	PCA_PLUS2_300
TATRA MOUNTAINS	8 April 2019	OA	0.91	0.91	0.90	0.91	0.92	0.91	0.93	0.92	0.92	0.88	0.93	0.89
		Kappa	0.83	0.83	0.82	0.85	0.85	0.84	0.87	0.86	0.86	0.79	0.88	0.80
KAÇKAR	19 December	OA	0.79	0.78	0.87	0.81	0.77	0.66	0.78	0.70	-	-	-	-
MOUNTAINS	2019	Kappa	0.68	0.67	0.80	0.71	0.66	0.50	0.66	0.54	-	-	-	-
	19 March 2019	OA	0.84	0.84	0.83	0.84	0.83	0.81	0.84	0.81	-	-	-	-
		Kappa	0.70	0.69	0.66	0.70	0.67	0.62	0.69	0.64	-	-	-	-
	8 April 2018	OA	0.75	0.75	0.72	0.74	0.75	0.74	0.75	0.77	-	-	-	-
		Kappa	0.64	0.63	0.60	0.62	0.64	0.62	0.64	0.68	-	-	-	-

The full tiles are assessed by randomly selecting ground truth data from SCL, with the same method as applied on spatial subsets (cf. Section 3.6). The confusion matrices of full tiles classified with the *Pca* method is presented in Figure 4.26, whereas the OA and Kappa coefficient values of the classified full tiles are shown in Figure 4.27.

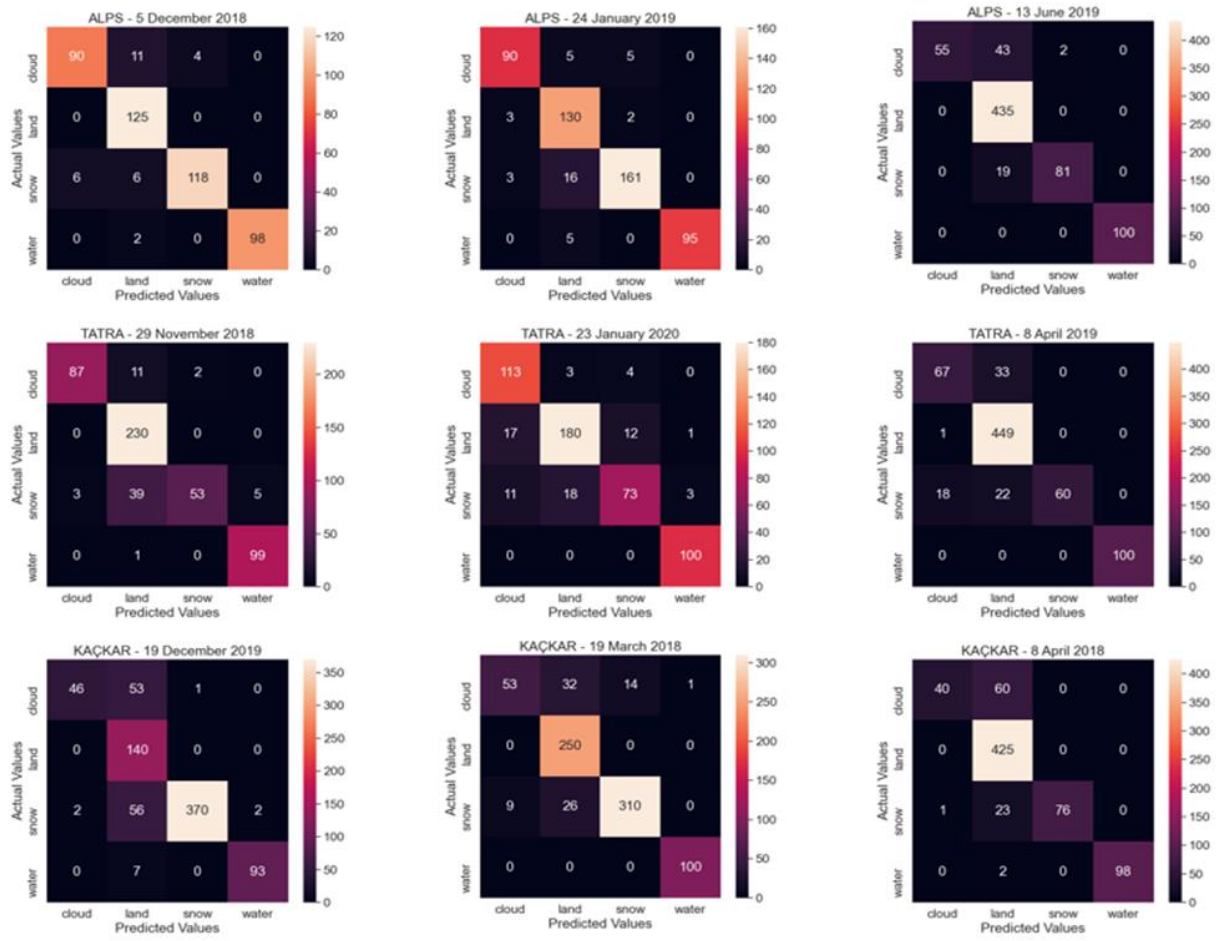


Figure 4.26. Confusion matrices of the classified full tiles (with *Pca* input combination)

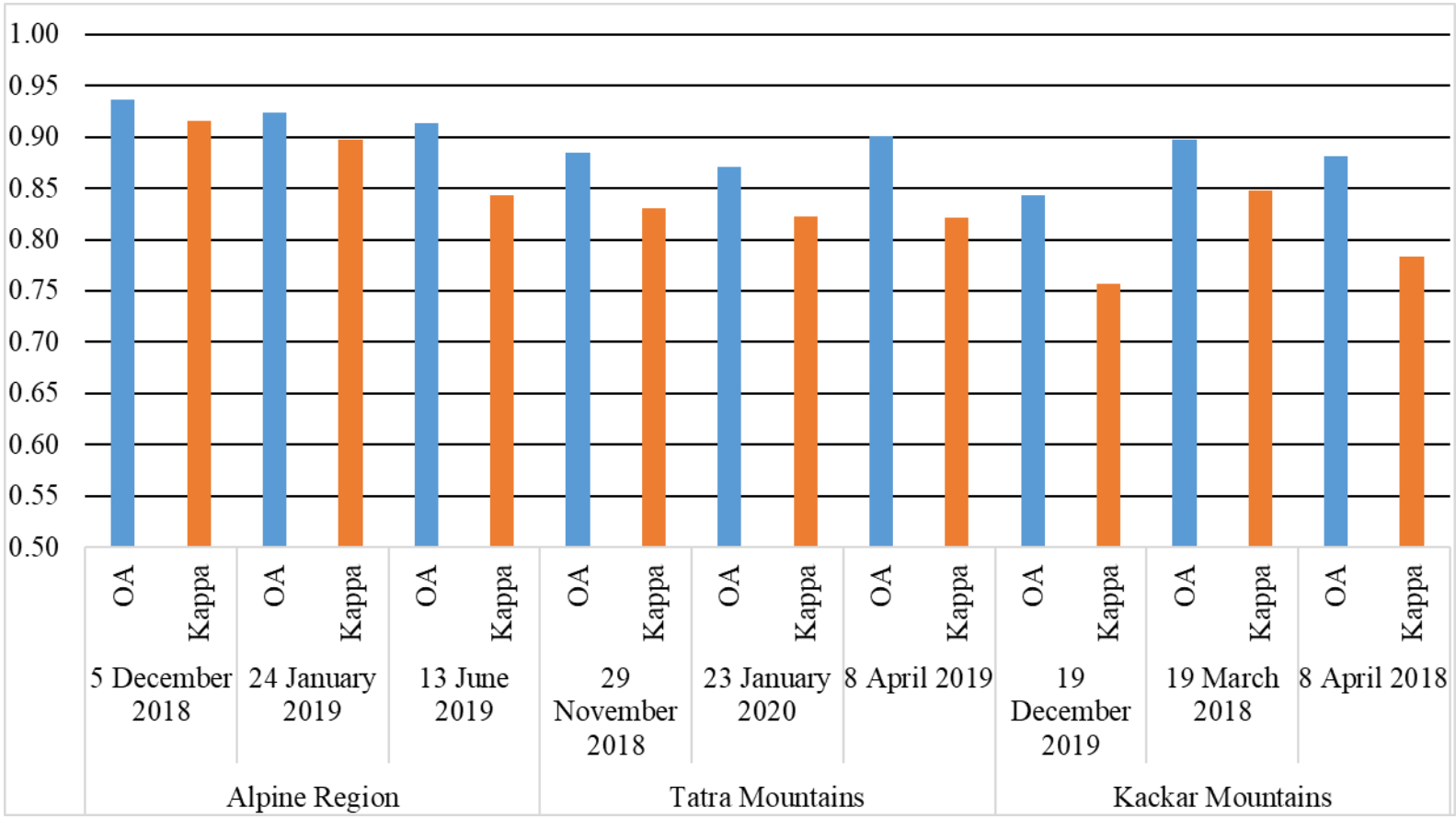


Figure 4.27. OA and Kappa coefficient values of for the full tiles classified with *Pca*

4.4 Discussion of the Results

As mentioned in detail in Chapter 3, the 20 km × 20 km subsets of the 9 Sentinel 2 images are classified with RF algorithm with different input combinations. The input combinations giving the best and worst results, with respect to the obtained OA and Kappa coefficient values for each subset, are presented in Table 4.4, excluding the results of *pca_plus* and *pca_plus2*, as these combinations have been conducted after determining the best and worst statistical metric results out of all subsets. The input combination that yields the best result has been determined as *Pca* with training sample size of 1000 per class, whereas the input combination yielding the worst performance has been concluded as *Dem* with 300 training samples.

Table 4.4. The best and worst input combinations for each 20 km × 20 km spatial subset with respect to OA and Kappa values.

Location	Date	Best Input Combination	Best OA Value	Best Kappa coefficient Value	Worst Input Combination	Worst OA Value	Worst Kappa coefficient Value
ALPS REGION	5 December 2018	Sc_1000	0.94	0.89	Dem_300	0.84	0.70
	24 January 2019	Atmo_topo_1000	0.81	0.7	Sc_1000	0.78	0.65
	13 June 2019	Pca_1000	0.84	0.82	Dem_300,	0.76	0.72
TATRA MOUNTAINS	29 November 2018	Pca_1000	0.78	0.71	Atmo_topo_300	0.69	0.59
	23 January 2020	Atmo_topo_1000	0.80	0.72	Pca_1000	0.71	0.59
	8 April 2019	Pca_1000	0.93	0.87	Sc_1000	0.90	0.82
KAÇKAR MOUNTAINS	19 December 2019	Sc_1000	0.87	0.80	Dem_300	0.66	0.50
	19 March 2018	Pca_1000	0.84	0.70	Dem_300	0.81	0.62
	8 April 2018	Pca_300	0.77	0.68	Sc_1000	0.72	0.60

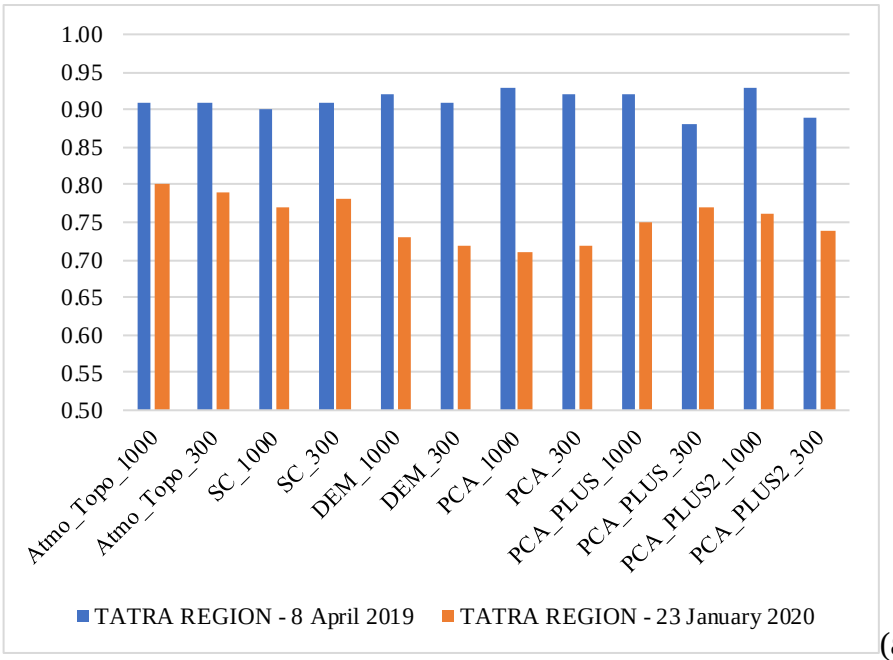
The most accurate results, in general, have been obtained over the subset of Tatra Mountains for 8 April 2019. The least accurate results have been observed over the subset of Tatra Mountains on 23 January 2020, considering the input combination of Pca with training data size of 1000, since it has been determined to be the best input combination in general.

The feature importance of RF for the input combinations have been presented in Appendix I. In general:

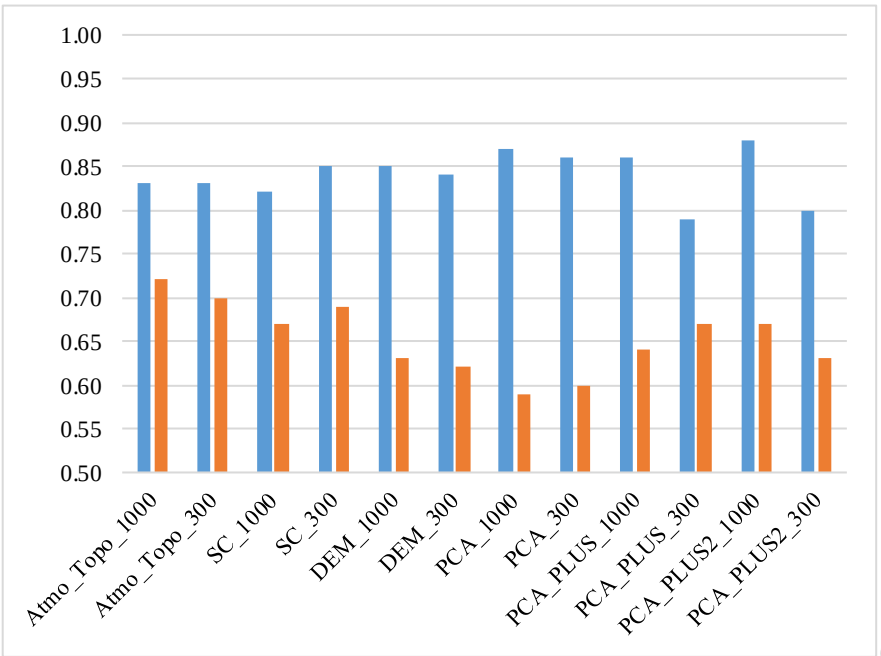
- ndsi feature, overall, generates very high importance. Bands 7,8A and 11 are also features with overall high importance.
- pca1 (first principal component) feature have very high importance for *Pca* input combination
- dem feature (elevation data) has very low importance, mostly the lowest importance out of all features for *Dem* input combination, which can be the cause of worse accuracy results.

Considering the results of feature importances, a base study is conducted, considering only of NDSI accuracy assessment results. Only classes that were considered are "snow" and "nosnow", different from other input combinations. "snow" class is denoted where NDSI >= 0.4 and "nosnow" class is denoted where NDSI < 0.4 (Zhang et al., 2019). The Python code for the classification is given in Appendix A. The accuracy assessment is conducted by collecting random test data using SCL of Sen2Cor and with the aid of ArcMap 10.7 software, same with other input combinations. Number of test samples per class which are calculated by multinomial distribution formula (Ozdarici Ok & Akyurek, 2012; Jensen, 2006) is given in Appendix N. OA and Kappa coefficient results of this study is presented in Appendix K. According to the results, although in some spatial subsets, NDSI have a very high accuracy (i.e., Alps region, 13 June 2019 subset), overall, the obtained results are lower compared to results of RF input combinations, especially considering *Pca* input combination.

Pca_plus and Pca_plus2 methods have been conducted on both tiles after determining the overall best and worst accuracy metrics across all image subsets. The comparison of OA and Kappa coefficient values of Pca_plus, Pca_plus2 and other input combinations of both tiles are available in Figure 4.28.



(a)



(b)

Figure 4.28. The comparison of OA (a) and Kappa coefficient (b) values of all input combinations conducted on the partial tiles of Tatra Region, with the dates of a) "8 April 2019" and b) "23 January 2020"

As can be observed from Figure 4.28, the *Pca_plus2* method yields almost the same result as the *Pca* method, the best method, for the tile of 8 April 2019; however, it does not exhibit a significant difference from *Pca* method. Since both *Pca_plus2* and *Pca_plus* methods yield worse results as compared to some other input combinations for the 23 January 2020 subset, *Pca* method is decided to be the best method to use out of all different input combinations. Thus, all tiles are fully subjected to RF classification with *Pca* input combination. The OA and Kappa coefficient values of the classified tiles can be seen in Figure 4.27. According to the accuracy assessment, the tile from Alps region acquired on 5 December 2018 yielded the best results with OA value of 0.94, and Kappa value of 0.92. On the other hand, the tile from Kaçkar Mountains on 19 December 2019 achieved the worst results with OA value of 0.84, and Kappa coefficient value of 0.76.

An input combination of only the first three principal components have also been assessed, to see the effect of NDSI, NDVI and NDWI. The results are shown in Appendix L. The accuracy assessment results of the input combination (with training sample size of 1000) have yielded slightly higher accuracy assessment results than *Pca* input combination with same training sample size, on same spatial subsets. Although that is the case, it is still decided that *Pca* input combination with the indices NDSI, NDVI and NDWI, is considered to be the best out of all input combinations.

For the RF-classified full tiles, the regions and the phases of the snow cover are then compared (i.e., when snow starts to form, snow amount at its peak, and when snow melts) in Figure 4.29. To make these comparisons, the means of the OA and Kappa coefficient values have been attained per region.

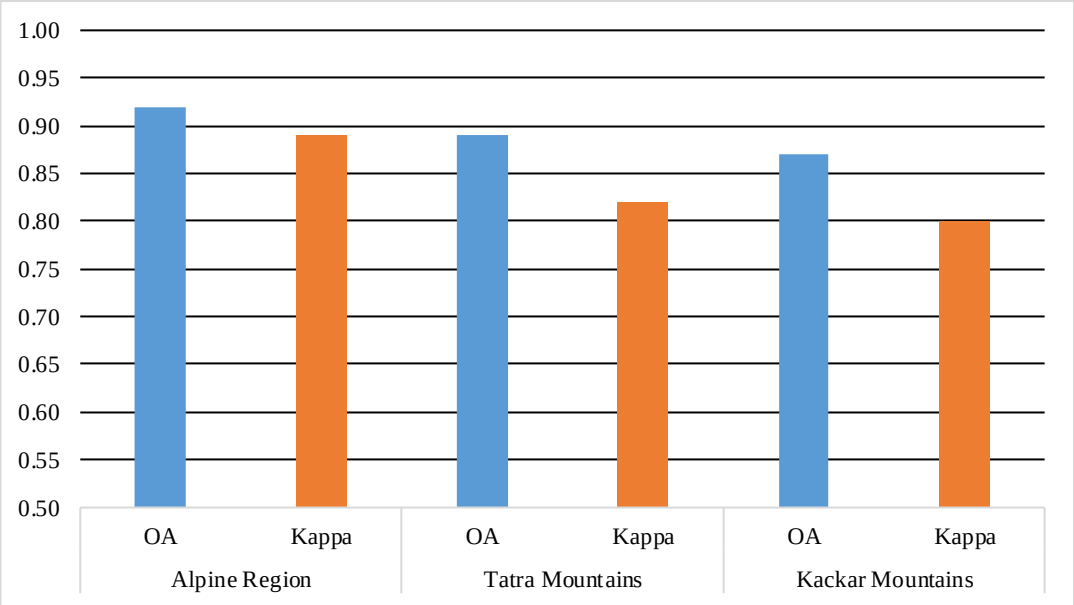


Figure 4.29. Average OA and Kappa coefficient values of the regions

According to the results, the region with the best values is Alps Region and the worst is Kaçkar Mountains. The general expectation is, when snow is at its peak, the accuracy of snow mapping is at its highest, compared to the beginning and end of the snow season (Romanov et al., 2002; Parajka & Blöschl, 2006), and in this study the only region that fulfills that expectation is Kaçkar Mountains. In Alps Region, the best results are obtained at the beginning of the snow season, and in Tatra Mountains, the end of the snow season is when the results have been better than other parts of the season. Incidentally, the land cover type with the largest area in Alps region and Tatra mountains tiles is *trees*, while Kaçkar Mountains' tile has the least tree cover (*trees* land cover) area out of all tiles (cf. Section 3.1.1). It might be concluded that, since dense forests may lead misrepresentation of the snow-covered area (Luo et al., 2022; Robinson et al., 1993; Bitner et al., 2002), the accuracy results of the tiles over Alps Region and Tatra Mountains are skewed. Snow cover mapping over forested areas is discussed in more detail in Section 4.4.3 in terms of the scope of this study.

Several studies for snow cover mapping with RF algorithm are observed to point out the similarities and/or differences in results. Nijhawan et al. (2018b) combined several algorithms to enhance the accuracy of the obtained snow cover maps. One of the base classifiers of this study was RF algorithm, and the OA result was 87.34%. The algorithms were trained on multispectral Landsat ETM+ data. Compared to the best obtained accuracy assessment result of this thesis (Alps region, 5 December 2018, OA: 0.94 with *Pca* input combination), the result obtained for this study is a bit low. Similarly, Haq et al. (2021) compared three classification algorithms on Hyperion images. OA of RF algorithm was found as 90.98%.

The confusion matrices of the spatial subsets have been observed in Section 0 (Figure 4.15 -Figure 4.23). Table 4.5 examines the classes that are mostly misclassified and what these classes are mostly misclassified as, for all spatial subsets and all input combinations. Table 4.6 presents the same for full tiles.

Table 4.5. Misclassified classes for all input combinations in classified spatial subsets

Location	Date	Input Combination											
		Atmo_Topo_1000	*Atmo_Topo_300*	*SC_1000*	*SC_300*	*DEM_1000*	*DEM_300*	*PCA_1000*	*PCA_300*	*PCA_PLUS_1000*	*PCA_PLUS_300*	*PCA_PLUS2_1000*	*PCA_PLUS2_300*
ALPS REGION	5 December 2018	Snow - Land	Snow - Land	Snow - Land	Snow - Land	Snow - Land	Snow - Land	Snow - Land	Snow - Land	-	-	-	-
	24 January 2019	Water - Land	Water - Land	Water - Land	Water - Land	Snow - Cloud	Snow - Cloud	Water - Land	Water - Land	-	-	-	-
	13 June 2019	Cloud - Land	Cloud - Land	Cloud - Land	Cloud - Land	Cloud - Land	Cloud - Land	Cloud - Land	Cloud - Land	-	-	-	-
TATRA MOUNTAINS	29 November 2018	Snow - Land	Snow - Land	Snow - Land	Snow - Land	Snow - Land	Snow - Land	Cloud - Land	Cloud - Land				
	23 January 2020	Land - Cloud	Snow - Land	Land - . Snow	Land - Snow	Land-. Snow	Snow - Cloud	Snow - Land	Snow - Land	Snow- Land	Snow- Land	Snow- Land	Snow- Land

Table 4.5 (continued)

Location	Date	Input Combination											
		Atmo_Topo_1000	*Atmo_Topo_300*	*SC_1000*	*SC_300*	*DEM_1000*	*DEM_300*	*PCA_1000*	*PCA_300*	*PCA_PLUS_1000*	*PCA_PLUS_300*	*PCA_PLUS2_1000*	*PCA_PLUS2_300*
TATRA MOUNTAINS	8 April 2019	Snow-Land	Snow-Land	Snow-Land	Snow-Land	Snow-Land	Snow-Land	Cloud-Land	Snow-Land	Cloud-Land	Snow-Cloud	Cloud-Land	Cloud-. Snow
KAÇKAR MOUNTAINS	19 December 2019	Cloud-Land	Cloud-Land	Cloud-Land	Snow-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	-	-	-	-
	19 March 2018	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	-	-	-	-
	8 April 2018	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	Cloud-Land	-	-	-	-

Table 4.6. The most misclassified classes for all input combinations in classified full tiles

Location	Date	Misclassified Classes
ALPS REGION	5 December 2018	Cloud miscl. No snow
	24 January 2019	Snow miscl. No snow
	13 June 2019	Cloud miscl. No snow
TATRA MOUNTAINS	29 November 2018	Snow miscl. No snow
	23 January 2020	Snow miscl. No snow
	8 April 2019	Cloud miscl. No snow
KAÇKAR MOUNTAINS	19 December 2019	Cloud miscl. No snow
	19 March 2018	Cloud miscl. No snow
	8 April 2018	Cloud miscl. No snow

The most frequent misclassifications observed in this study are discussed below.

Snow Misclassified as No-snow (Land)

Snow misclassified as No-snow (Land) class is one of the most common misclassifications that have been observed in this study, especially for the tiles from Alps Region and Tatra Mountains. As explained in the beginning of Chapter 3, No-snow (Land) class is denoted as more of a composite of land cover classes that do not consist of snow, water or cloud, which naturally contains the *trees* land cover type. *Trees* cover is the most frequent land cover type in Alps Region and Tatra Mountain tiles, in both years of 2018 and 2019 (cf. Section 3.1.1). Trees, especially dense forests, are commonly noted as a limitation in the literature while mapping snow cover from remote sensing data (Robinson et al., 1993; Bitner et al., 2002) such that snow can be underrepresented in forests. It is guessed that this limitation is the

cause of the misclassification of snow as No-snow (Land) class. It can be noted that the *Pca* input combination decrease the extent of the issue but it is not valid for all the instances. The extent of forests minimizing the snow cover representation is discussed in Section 4.4.3 as a limitation for this study and as a possible future research based on this thesis, with visual examples.

Cloud Misclassified as No-snow (Land)

Another common misclassification, for this study, is cloud class being wrongly denoted as No-snow (Land) class, especially for the tiles located in Kaçkar Mountains. There have been many studies to either determine or to eliminate cloud cover from remote sensing data by developing and examining various techniques, including RF (Deng et al., 2016; Ghasemian & Akhoondzadeh, 2018). In these studies, results obtained with RF algorithms generate good accuracy, but usually paired with another method, such as *K*-means clustering method, segmentation and thresholding. Therefore, to avert misclassification of cloud cover, RF can be used along with one of these methods, or another method to achieve better accuracy in the future. Furthermore, Ghasemian and Akhoondzadeh (2018) state that while RF algorithm yields better results than other methods in literature, it still does not consider textural features. Therefore, it can be said that the misrepresentation of cloud might stem from this, and in future works, features of cloud cover might be implemented to the input algorithms as proposed in this study.

The examples of misclassification between Cloud and No-snow (Land) can be observed in Figure 4.30 - Figure 4.32. The fully classified images with *Pca* input combination are selected from Table 4.6, where the most misclassified classes are Cloud with No-snow (Land). The red rectangles in the Figures are to emphasize the locations where Cloud class is misrepresented as No-snow (Land).

Figure 4.30. Example of misclassification of Cloud as No-snow (Land) class in fully classified image of Alps Region tile, dated 13 June 2019 (full tile and the cropped area (red rectangle) are represented at the lower right)

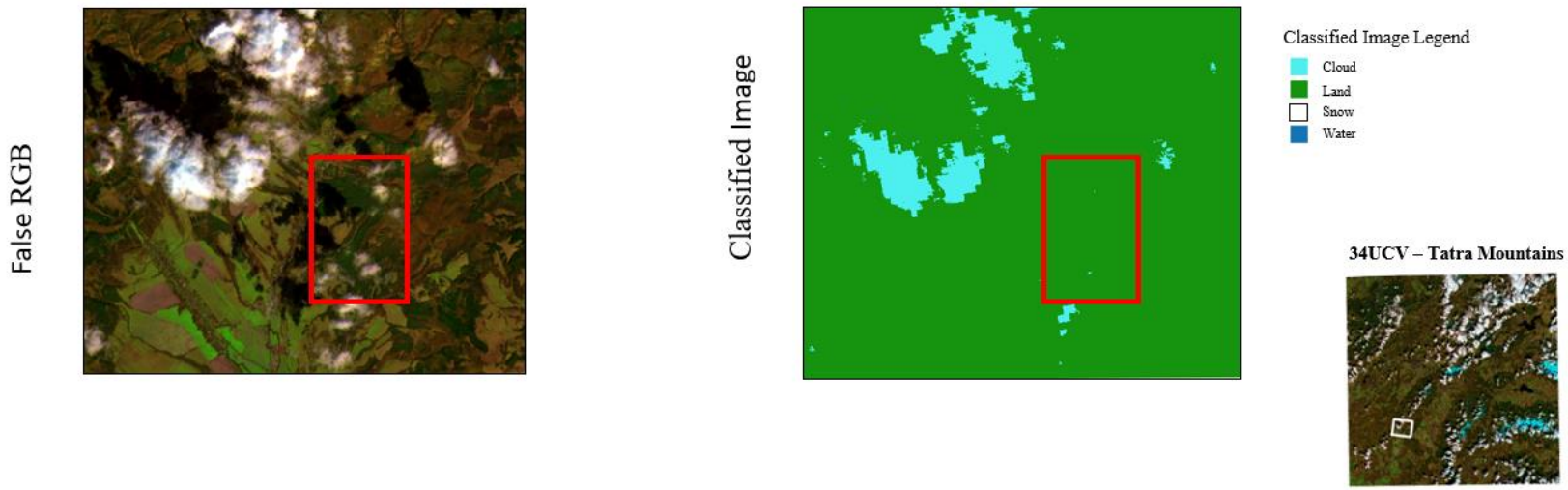


Figure 4.31. Example of misclassification of Cloud as No-snow (Land) class in fully classified image of Tatra Mountains tile, dated 8 April 2019 (full tile and the cropped area (white rectangle) are represented at the lower right)

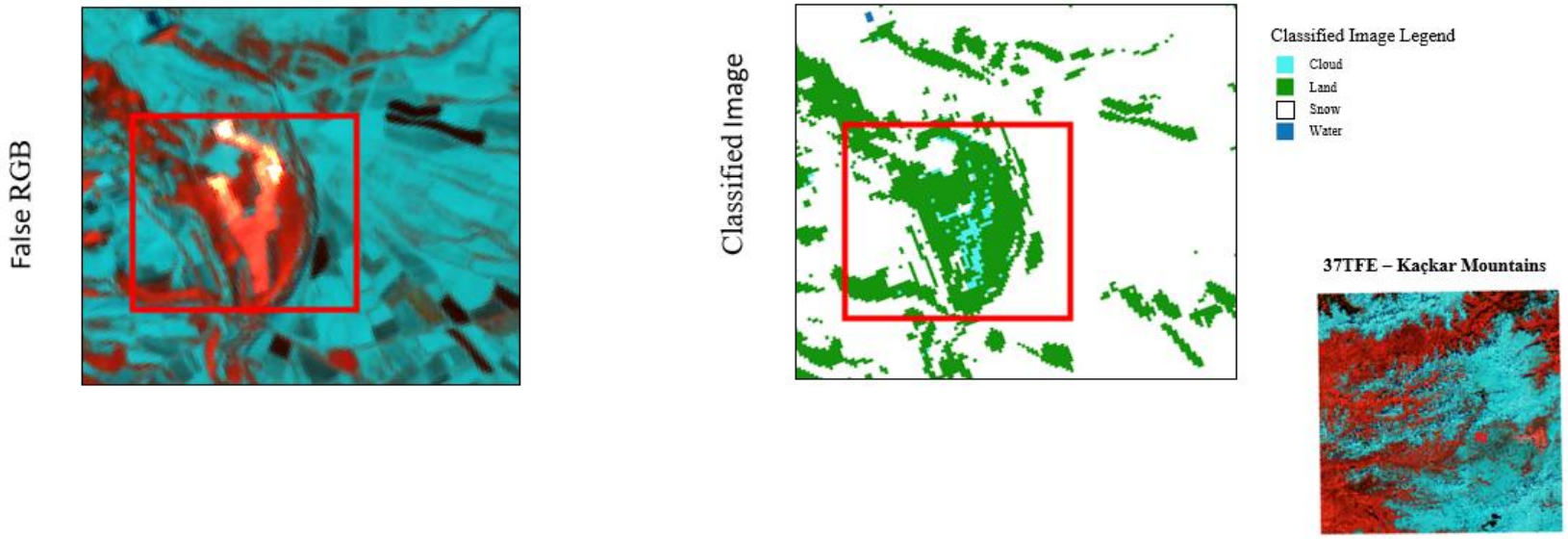


Figure 4.32. Example of misclassification of Cloud as No-snow (Land) class in fully classified image of Kaçkar Mountains tile, dated 19 December 2019 (full tile and the cropped area (red rectangle) are represented at the lower right)

Water Misclassified as No-snow (Land)

Water class being misclassified as No-snow (Land) class, and vice versa, have regularly been observed in this study. There are studies showing that the water bodies can be very well determined with the aid of RF (Gislason et al., 2006). In this study, selected tiles are from regions with complex topographies, and Water class, more often than not, consists of narrow rivers in the areas of interest. This may be the cause of misclassification between those two classes, especially within areas of small water bodies and mountain shadows (Ko et al., 2015). For this study, it can be noted that the *Pca* combination is observed to ease this problem the most out of all input combinations, with almost all spatial subsets. Unlike with the case with spatial subsets, water class has not been misclassified with the No-snow (Land) class as much, and it is observed that the *Pca* mostly eliminates the confusion between water and shadow areas, but there are also some areas which is not the case. The examples of misclassification between water and shadowy areas are presented in Figure 4.33-Figure 4.35. The misclassified areas are emphasized with white or red rectangles. QGIS Google Terrain base map is used to underline the fact that the areas that are emphasized in the figures are not water.

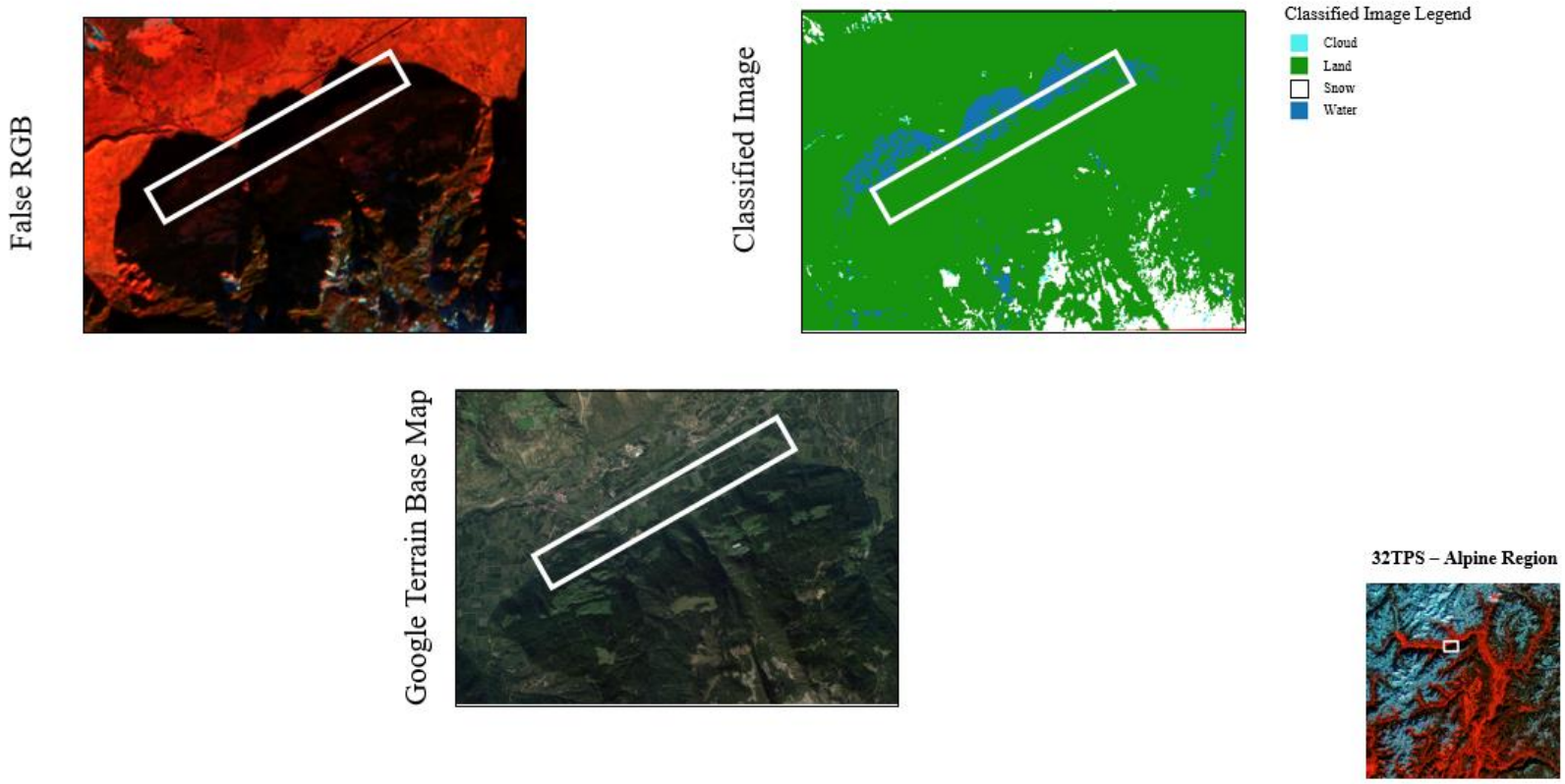


Figure 4.33. Example of misclassification of mountain shadows as Water class in fully classified image of Alpine Region tile, dated 24 January 2019 (full tile and the cropped area (white rectangle) are represented at the lower right)

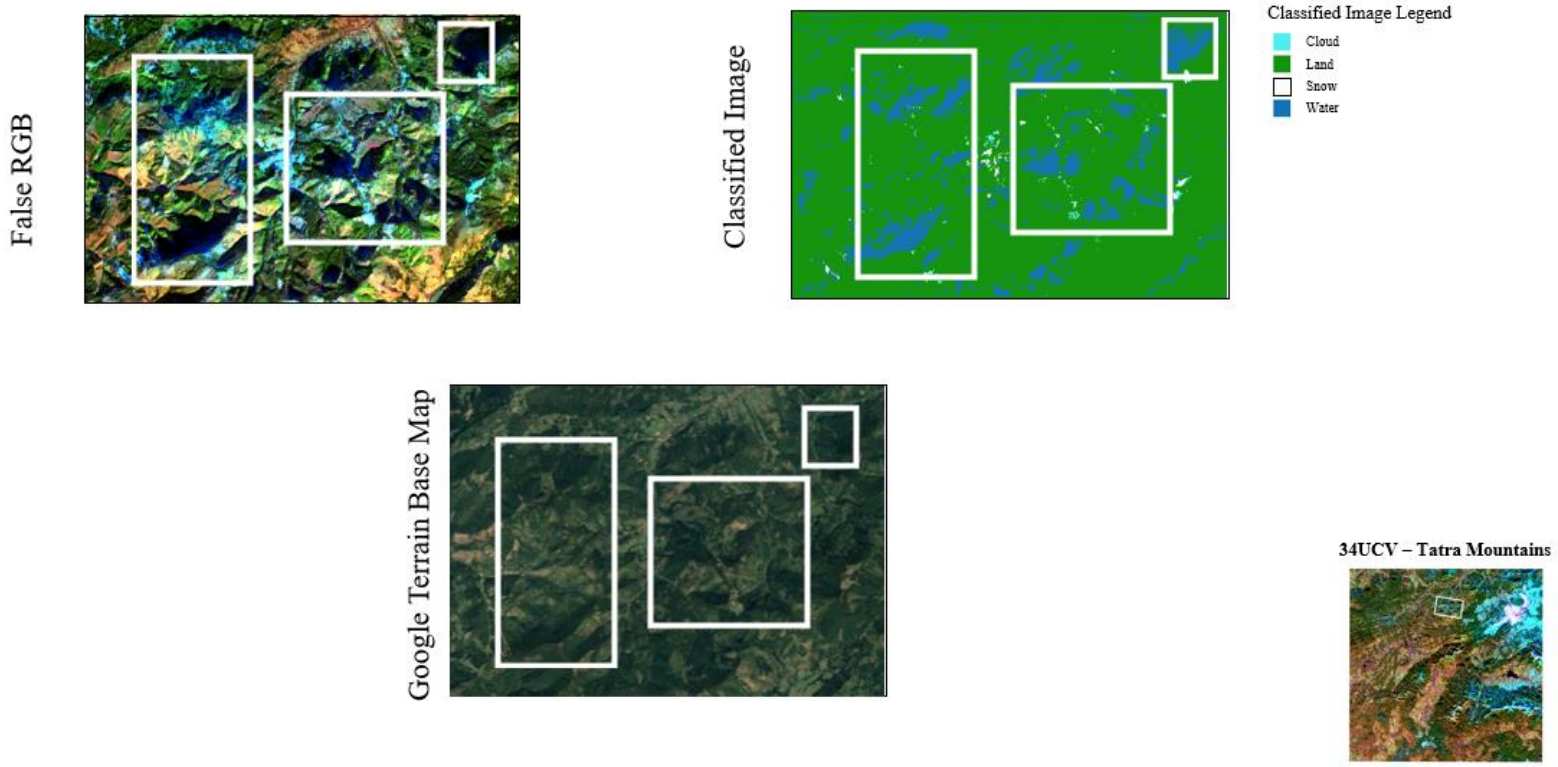


Figure 4.34. Example of misclassification of mountain shadows as Water class in fully classified image of Tatra Mountains tile, dated 29 November 2018 (full tile and the cropped area (white rectangle) are represented at the lower right)

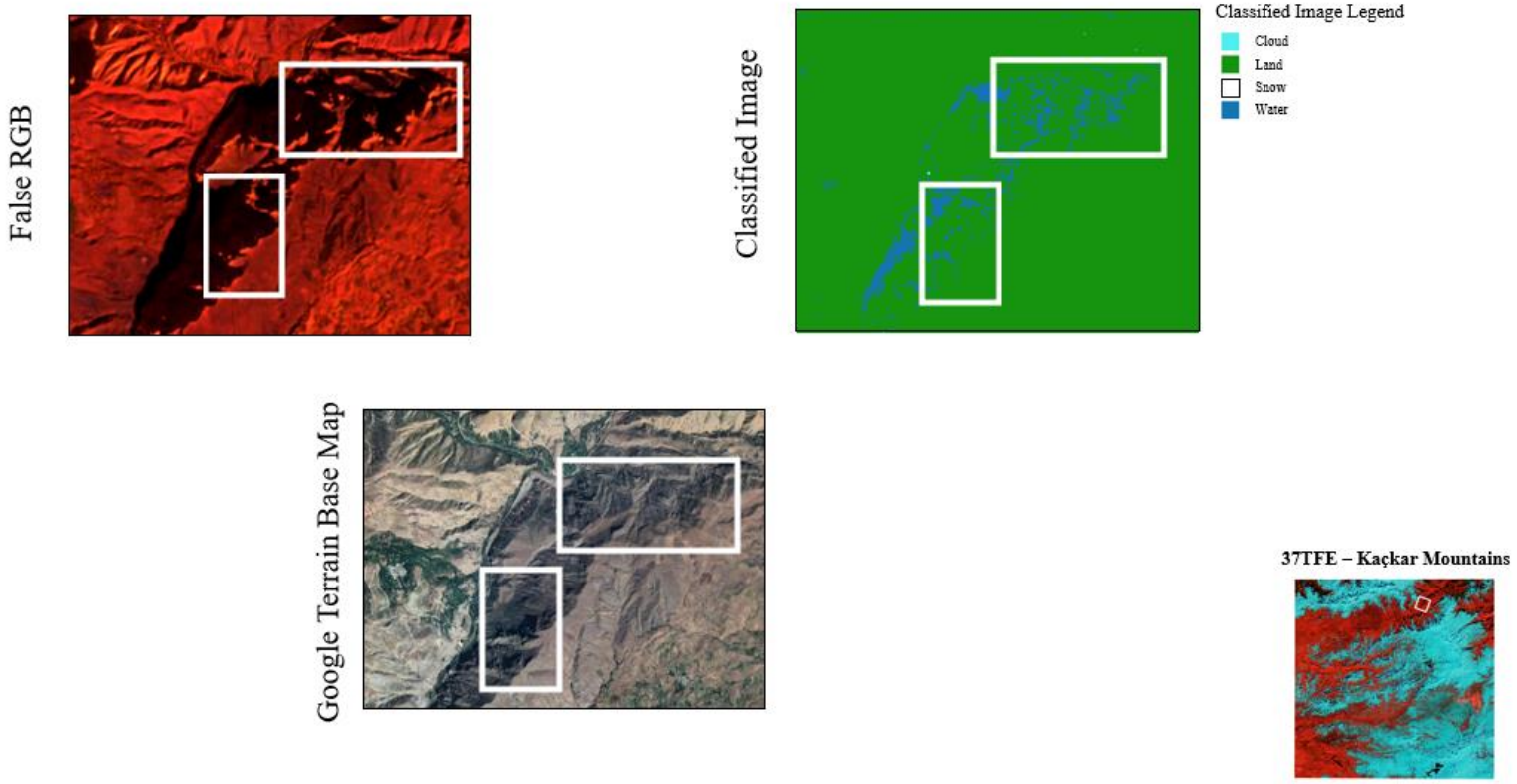


Figure 4.35. Example of misclassification of mountain shadows as Water class in fully classified image of Kaçkar Mountains tile, dated 19 December 2019 (full tile and the cropped area (white rectangle) are represented at the lower right)

Snow class in shadow areas, on the other hand, have been classified well with *Pca* input combination. Examples of well-classified areas which have been presented in Figure 4.36 - Figure 4.38, are emphasized with red rectangles. DEMs (hillshade) are used in these figures to show the difference in elevation.

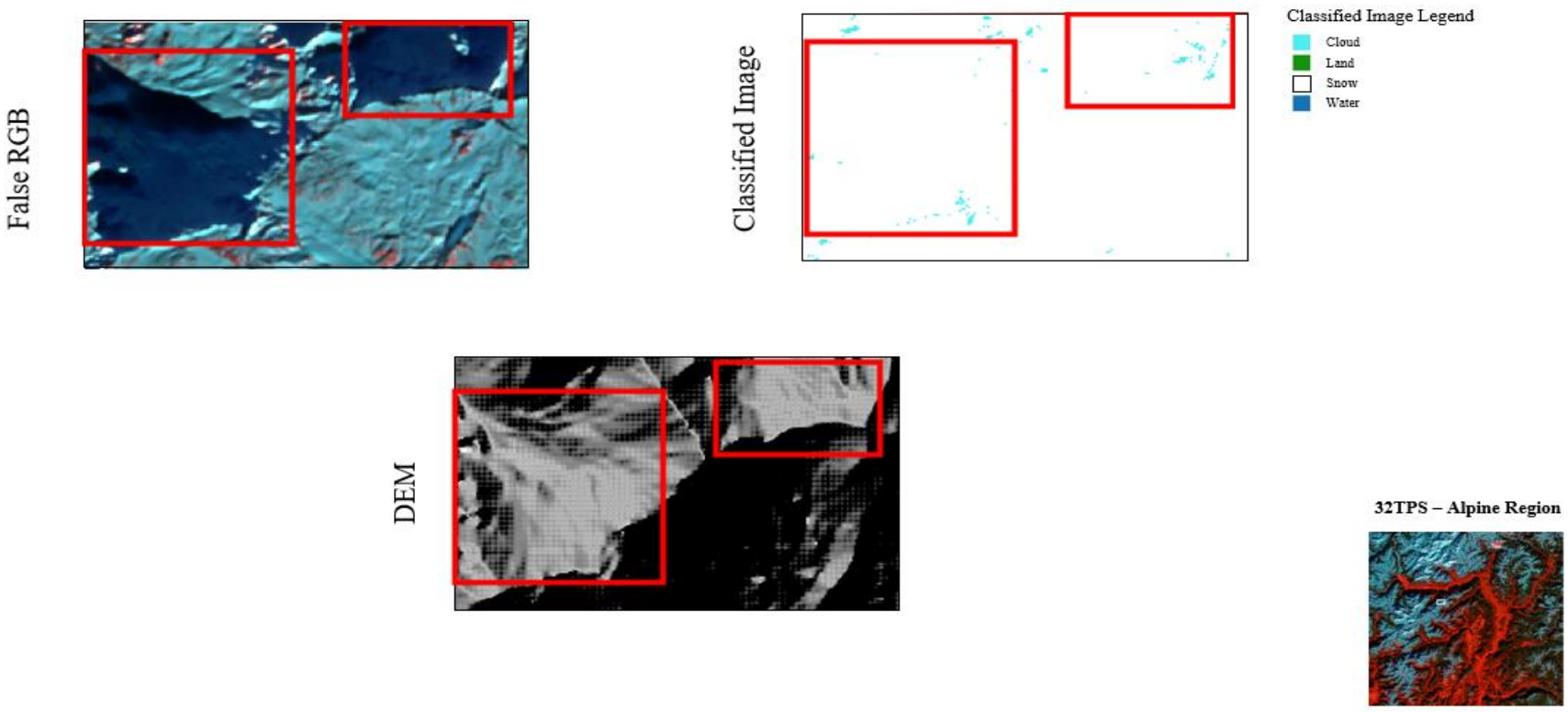


Figure 4.36. Example of classification of snow in shadowy areas in fully classified image of Alps Region tile, dated 24 January 2019 (full tile and the cropped area (white rectangle) are represented at the lower right)

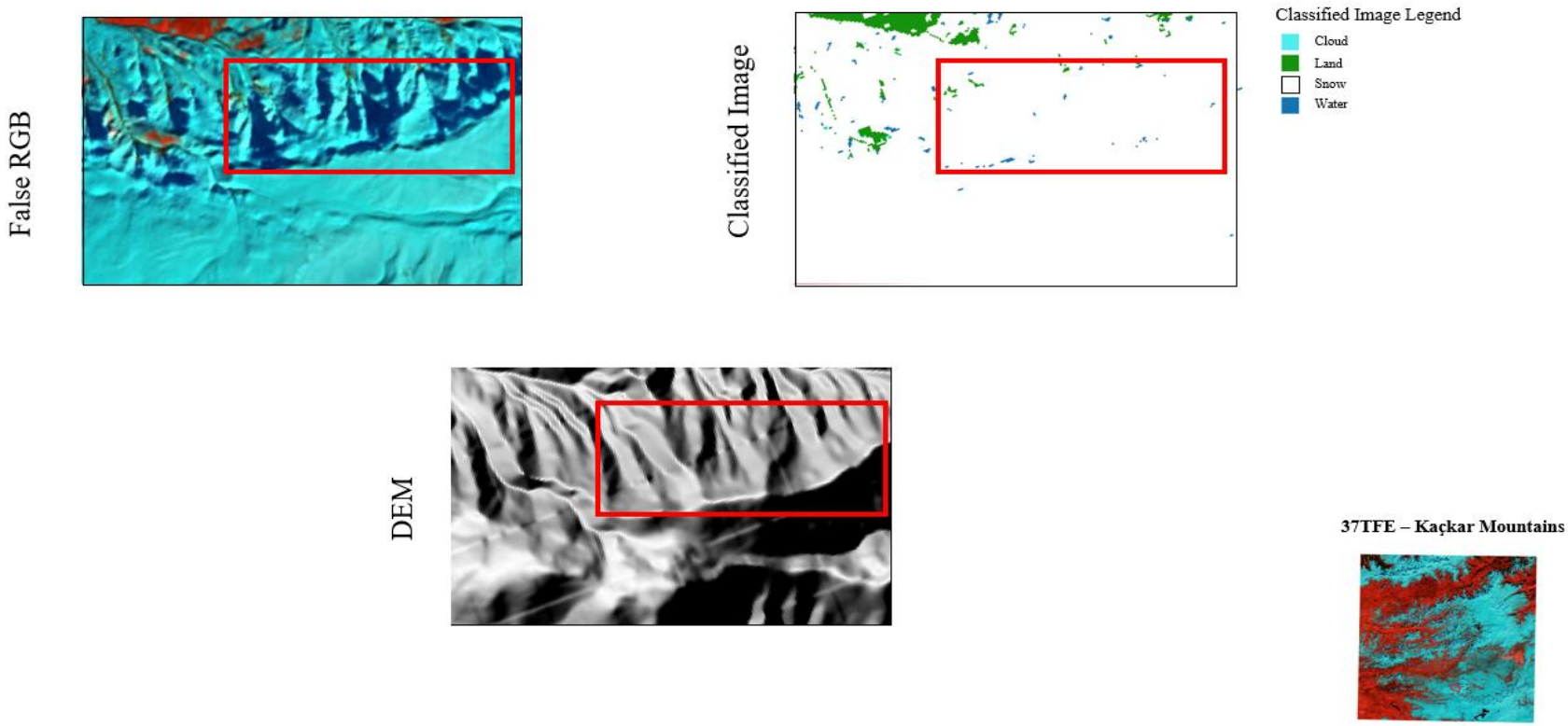


Figure 4.37. Example of classification of snow in shadowy areas in fully classified image of Tatra Mountains tile, dated 23 January 2019 (full tile and the cropped area (white rectangle) are represented at the lower right)

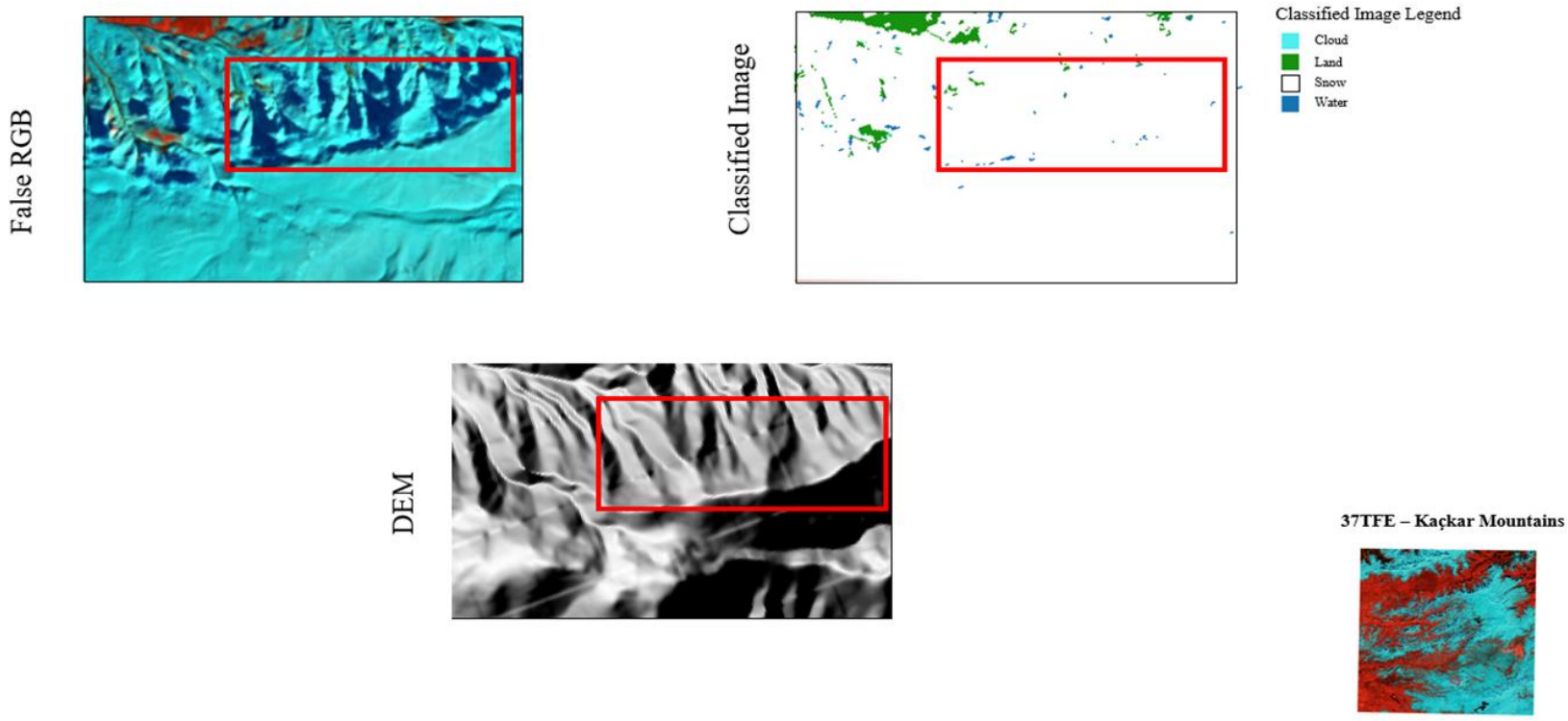


Figure 4.38. Example of classification of snow in shadowy areas in fully classified image of Kaçkar Mountains tile, dated 19 December 2019 (full tile and the cropped area (red rectangle) are represented at the lower right)

For randomly sampled training data, visually checked and manually adjusted SCL was used (cf. Section 3.2.2.1). SCL have been observed to perceive a lot of shadow areas as water, and the layer have been especially altered for that reason. Some example shadowy areas of original, untouched SCL compared with fully classified images with *Pca* input combination (with training sample size of 1000) have been given in Figure 4.39 - Figure 4.42, with shadowy areas emphasized with red and white rectangles.

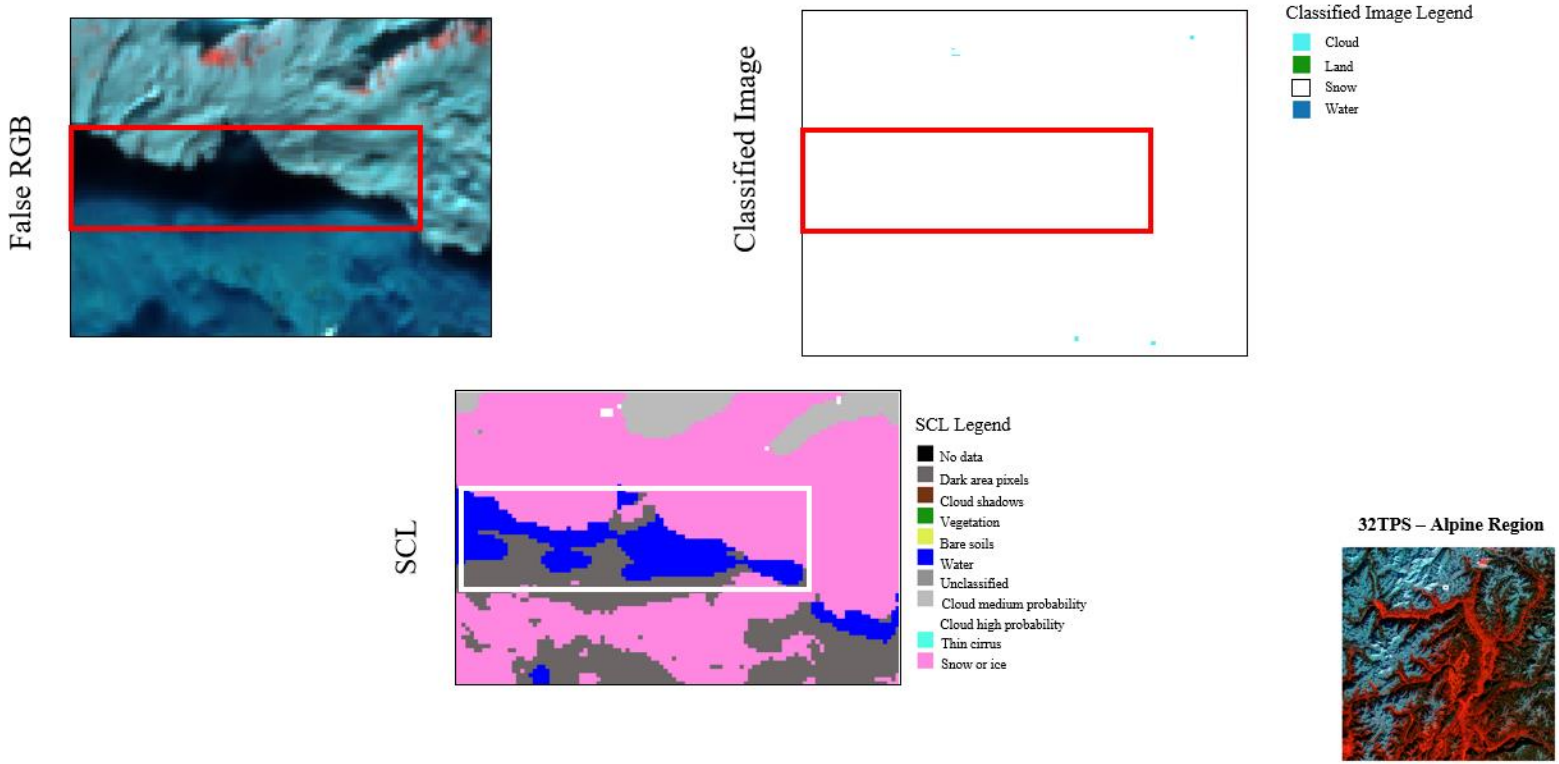


Figure 4.39. Example of classification in shadowy areas in fully classified image of Alpine Region tile, compared to SCL, dated 24 January 2019 (full tile and the cropped area (white rectangle) are represented at the lower right)

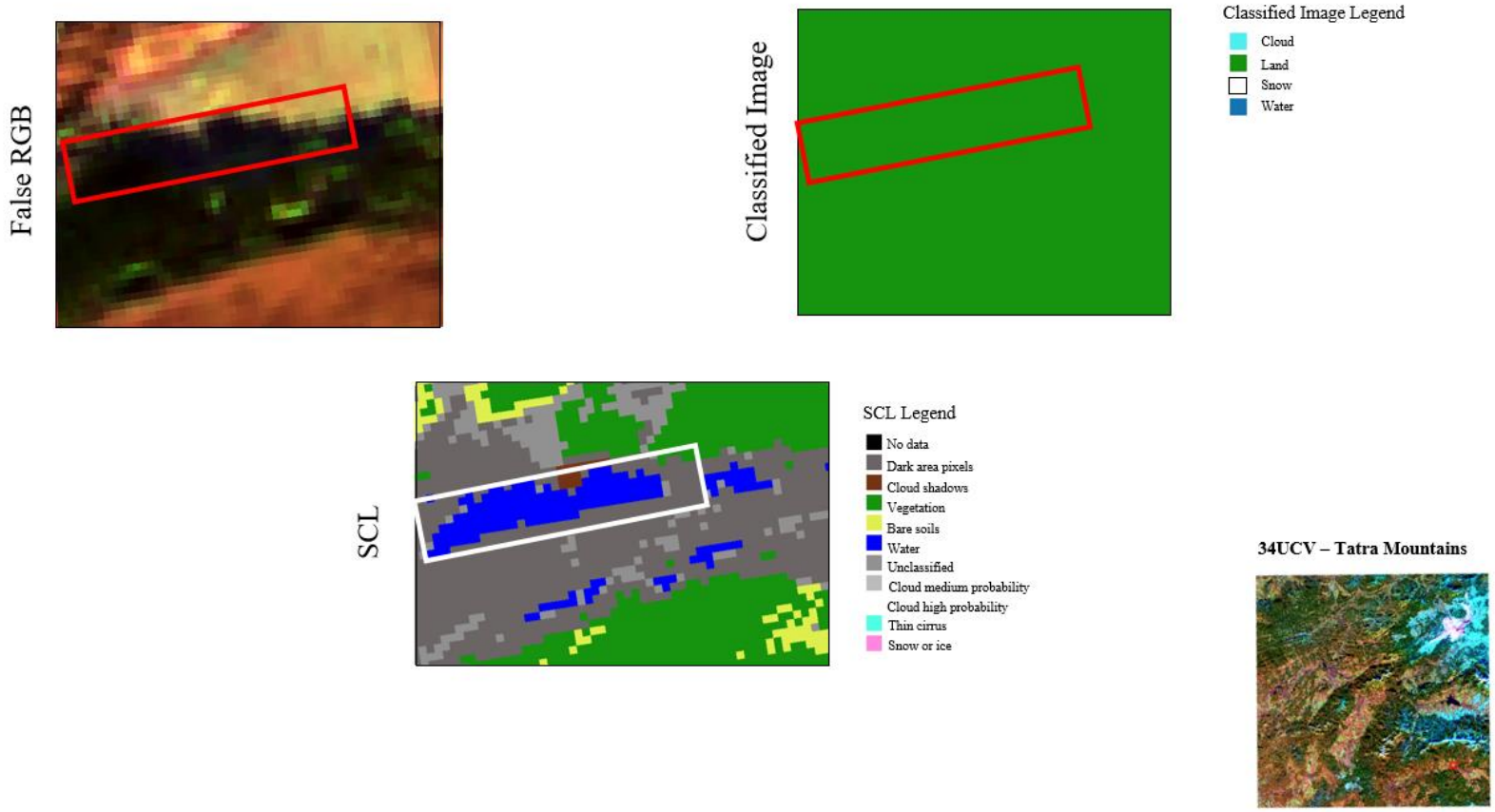


Figure 4.40. Example of classification in shadowy areas in fully classified image of Tatra Mountains tile, compared to SCL, dated 29 November 2018 (full tile and the cropped area (red rectangle) are represented at the lower right)

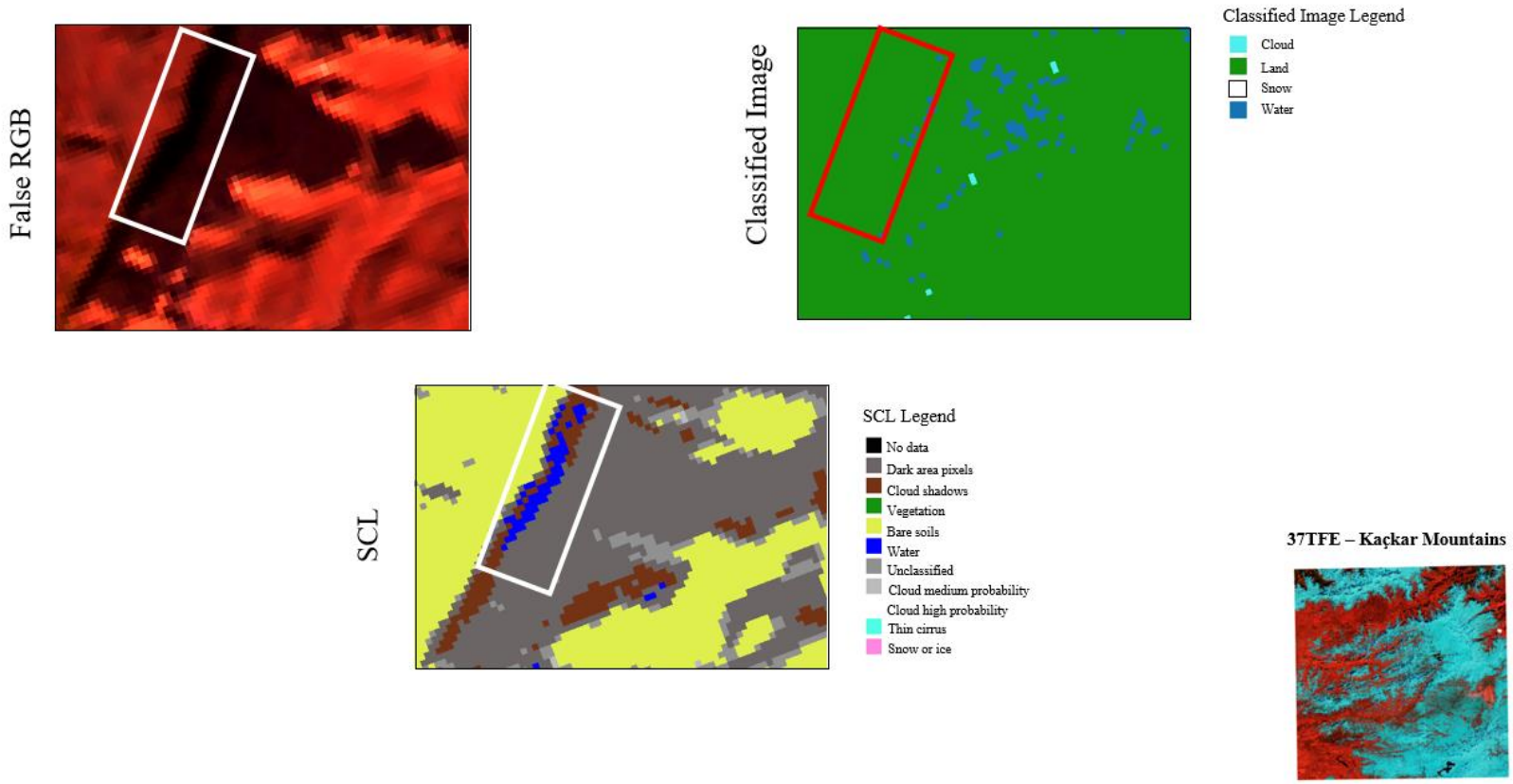


Figure 4.41. Example of classification in shadowy areas in fully classified image of Kaçkar Mountains tile, compared to SCL, dated 19 December 2019 (full tile and the cropped area (white rectangle) are represented at the lower right

Although the issue of cloud and snow confusion is common (Snehmani et al., 2015; Dietz et al., 2012; Barry et al., 1995), in this study, it was not a considerable concern, for all input combinations in most cases. The reason for this might be that RF algorithm, and other machine learning algorithms in general, surpass other approaches when it comes to alleviate this problem (Wang et al., 2022).

300 samples for each class were employed because the number of training samples per class is usually chosen in between *10x number of bands* to *30x number of bands* with remotely sensed images (Van Niel et al., 2005). The sample size that is three times larger of the usually chosen size is also tested in this study, and it is found that training sample size has a modest effect on the accuracy of RF algorithm. Although without exceptions, 1000 per class as training size have yielded better results in all input combinations than the number of training data of 300 per class, for spatial subsets. A study conducted by Li et al. (2014) studied the accuracy of classification on Landsat images with different training sample sizes, ranging from 20 to 240 per class, for a 4-band dataset and a 6-band dataset, with different algorithms. The study indicates that, RF is one of the mostly affected algorithms when it comes to training sample sizes, and the accuracy is observed to increase with the size of training samples, similar to the results achieved in this study.

The obtained ratio of eigenvalues for *Pca* input combination on full tiles (cf. Appendix J) show that the first principal component has the most impact for all of the images. The ratio of first principal component is generally above 0.90.

4.4.1 Runtime of Input Combinations

In this study, most of the work have been done within the Python environment, including RF classification for all input combinations. The average runtimes of RF algorithm for different input combinations are compared in Figure 4.42 for image subsets. It should be noted that the runtime of the code for obtaining the first three

principal components are added to the total runtime for *Pca*, *Pca_plus* and *Pca_plus2* methods.

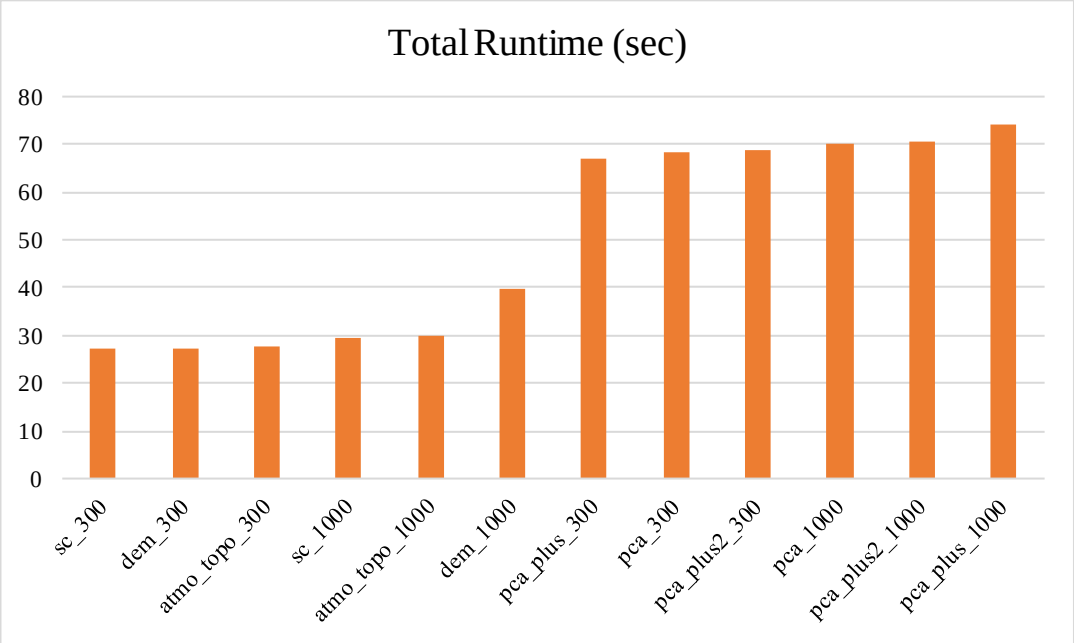


Figure 4.42. Total runtimes (in seconds) for all input combinations with image subsets

Obtaining principal components causes the biggest difference in runtime. Aside from that, RF algorithm runtime only slightly changes with different input combinations. It is observed that number of training samples also alters the runtime, yet slightly. The number of inputs seems to be more crucial when it comes to runtime while training sample size is 1000 per class, where the number of inputs are, with exceptions, proportional to the runtime. However, when the number of training data is 300 per class, the number of inputs is noted to be not as important. The full tiles are expectedly taking a much longer time to be classified than the image subsets. Each spatial subset has roughly 1/25 of the area of a full tile, and a runtime of, approximately, 1/15, as compared to the *Pca* method runtime with training size of 1000 per class.

4.4.2 Climatological Attributes of the Regions and the Effect on the Accuracy of the Results

The study of Kuter et al. (2022) shows that different regions with the same snow classification algorithm show some significant changes in the accuracy, and this might be attributed to the different characteristics of the terrain (Deems et al., 2013), as well as different climatological properties of the regions. In Section 3.1, the three areas of interest for this study have been briefly introduced with their characteristics, such as average precipitation, temperature, and snow depth for snow season for that region and/or annually, from different studies. In order to compare the climatic characteristics of the regions more effectively, ERA5-Land monthly reanalysis data was used for the timeframe between January 1990 to June 2022 for snow depth (in meters), 2 m temperature, which is the air temperature 2 meters above the surface (in Kelvin), and total precipitation (in meters) (Muñoz Sabater, 2021). The mean values that were obtained for the study areas (i.e., for full tiles) for each region monthly, with the aid of a Python script (cf. Appendix G), are presented in Table 4.7 - Table 4.9 (with no-data values omitted from the equation where necessary). The mean values of snow depth and total precipitation have been converted from meters to mm, and the mean values of 2 m temperature have been converted from Kelvin to ^{o}C.

Table 4.7.The average snow depth (cm) values derived from ERA5-Land monthly reanalysis data for the study areas (January 1990 to June 2022)

	Location		
Months	Alpine Region	Tatra Mountains	Kaçkar Mountains
January	97.2	N/A	3.5
February	104.9	0.1	9.8
March	109.4	0.7	10.2
April	104.0	N/A	4.5
May	80.8	N/A	0.1
June	71.4	N/A	0.0
July	70.2	N/A	0.0
August	70.1	N/A	0.0
September	70.2	N/A	0.0
October	70.4	N/A	0.0
November	72.5	N/A	0.0
December	85.9	N/A	0.4

Table 4.8.The average 2 m temperature (°C) values derived from ERA5-Land monthly reanalysis data for the study areas (January 1990 to June 2022)

	Location		
Months	Alpine Region	Tatra Mountains	Kaçkar Mountains
January	-7.6	-4.3	-8.9
February	-5.7	-3.0	-7.1
March	-1.9	0.6	-3.0
April	1.7	6.0	2.0
May	6.4	11.0	7.6
June	10.7	14.7	12.2
July	12.7	16.4	15.6
August	12.5	16.2	16.0
September	8.6	11.5	12.1
October	4.3	6.6	6.4
November	-1.6	1.8	-1.0
December	-6.5	-3.0	-6.8

Table 4.9.The average total precipitation (mm) values derived from ERA5 Land monthly reanalysis data for the study areas (January 1990 to June 2022)

	Location		
Months	Alpine Region	Tatra Mountains	Kaçkar Mountains
January	1.7	2.1	1.9
February	2.0	2.3	2.0
March	2.4	2.3	2.5
April	3.6	2.5	3.1
May	4.6	3.6	3.5
June	5.2	3.8	2.5
July	5.1	4.0	1.8
August	4.8	3.3	1.5
September	3.8	3.0	1.2
October	3.9	2.4	1.8
November	4.3	2.3	1.9
December	2.5	2.2	1.7

From Table 4.7, it can be observed that, the average snow depth of Alps Region tile is substantially higher than the other regions. Tatra Mountains tile, also, observed to have little to no snow depth. It was observed from the original Sentinel-2 images that, Tatra Mountains have significantly less snow cover than the other two regions, and from ERA5 Land monthly reanalysis data, snow depth is only present in the months of February and March. These results also might be because of the ERA5-Land monthly reanalysis products have a native spatial resolution of 9 km (Muñoz

Sabater, 2021), which might have produced some skewed data. It is also reported in some studies that ERA5-Land Monthly reanalysis products can yield to less accurate results in comparison to other similar products (Jiang et al., 2021; Renfrew et al., 2021). In Table 4.8, it can be seen that Tatra Mountains are also warmer than the other two study areas in winter months compared to the other two study areas, which can be the reason of less snow cover and depth compared to the other two regions.

The results show that, Alps Region have mostly permanent snow cover all year round, whereas Tatra and Kaçkar Mountains only have seasonal snow. Alpine Region also have slightly more total average precipitation than the other two area of interests, as can be seen from Table 4.8, which is likely the cause of permanent snow (Saavedra et al., 2017). High permanent snow depth amount can be the cause of higher accuracy results of Alpine Region.

4.4.3 Limitations and Future Work

In this study, some challenging aspects of mapping snow cover has been reduced, such as cloud cover misrepresented as snow and vica versa due to the visible reflectance similarities of both, as well as mountain shadows in complex terrains (Snehmani et al., 2014; Dietz et al., 2012; Barry et al., 1995). One difficult element to consider for the detection of snow cover has been left out of the scope of this study, which is the detection of snow cover in forested areas. It should be considered as a limitation rather than a problem at optical wavelengths due to the canopy interference of dense forest stands, which causes the snow cover to be underrepresented (Luo et al., 2022; Robinson et al., 1993; Bitner et al., 2002). It is perceived, from the obtained results, that the *Pca* input combination yields more improved results in forests than other combinations. However, it does not totally solve the problem. It is demonstrated by Klein et al. (1998) and Nolin (2010) that the forest areas with snow cover generates lower NDSI values than normal. Although Klein et al. (1998) finds that the combined NDSI and NDVI approach is more helpful in terms of snow cover mapping, the combination of these two indices is not yielding

adequate results in dense forest areas (Luo et al., 2022; Rittger et al., 2013). Therefore, *Pca* input combination falling short at snow cover mapping in forest areas is due to the fact that it contains both indices in addition to the first three principal components and NDWI.

To examine the issue of misrepresentation of snow in the presence of forests further, *Fractional Snow Cover on Ground* (FSCOG) layer of Copernicus High Resolution Snow & Ice Monitoring Service, which is canopy-adjusted fractional snow cover (FSC) at 20 m resolution, derived from Sentinel-2 products (Barrou Dumont et al., 2021) have been used. The layers have been classified into four classes: snow, no-snow, cloud and no-data (if available) with the help of a Python script (cf. Appendix H), and then compared with the classified full tiles (with *Pca* combination). It is observed that the snow is underrepresented in some areas with *trees* land cover in all of the fully classified images. The example locations where underestimation of snow in *trees* land cover occurs have been demonstrated with the images of fully classified tiles alongside with the original false color RGB Sentinel 2 images, Google's DW land cover maps, and FSCOG layers in Figure 4.43- Figure 4.48 for once for each year (2018 and 2019). The red rectangles in the Figures indicate the snow cover areas that are underestimated due to *trees* land cover.

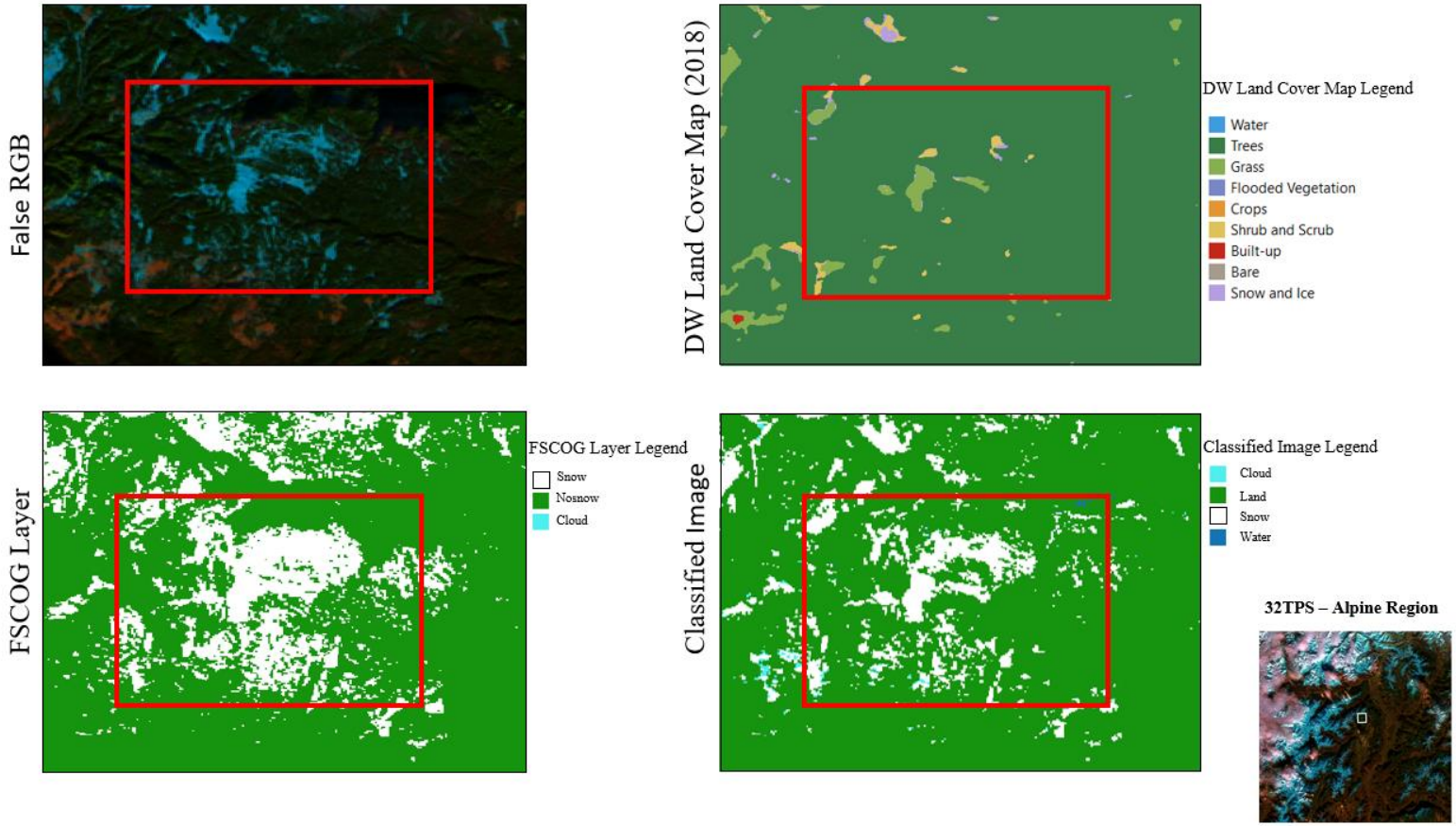


Figure 4.43. Example of underestimation of snow in classified image of Alps Region, 5 December 2018

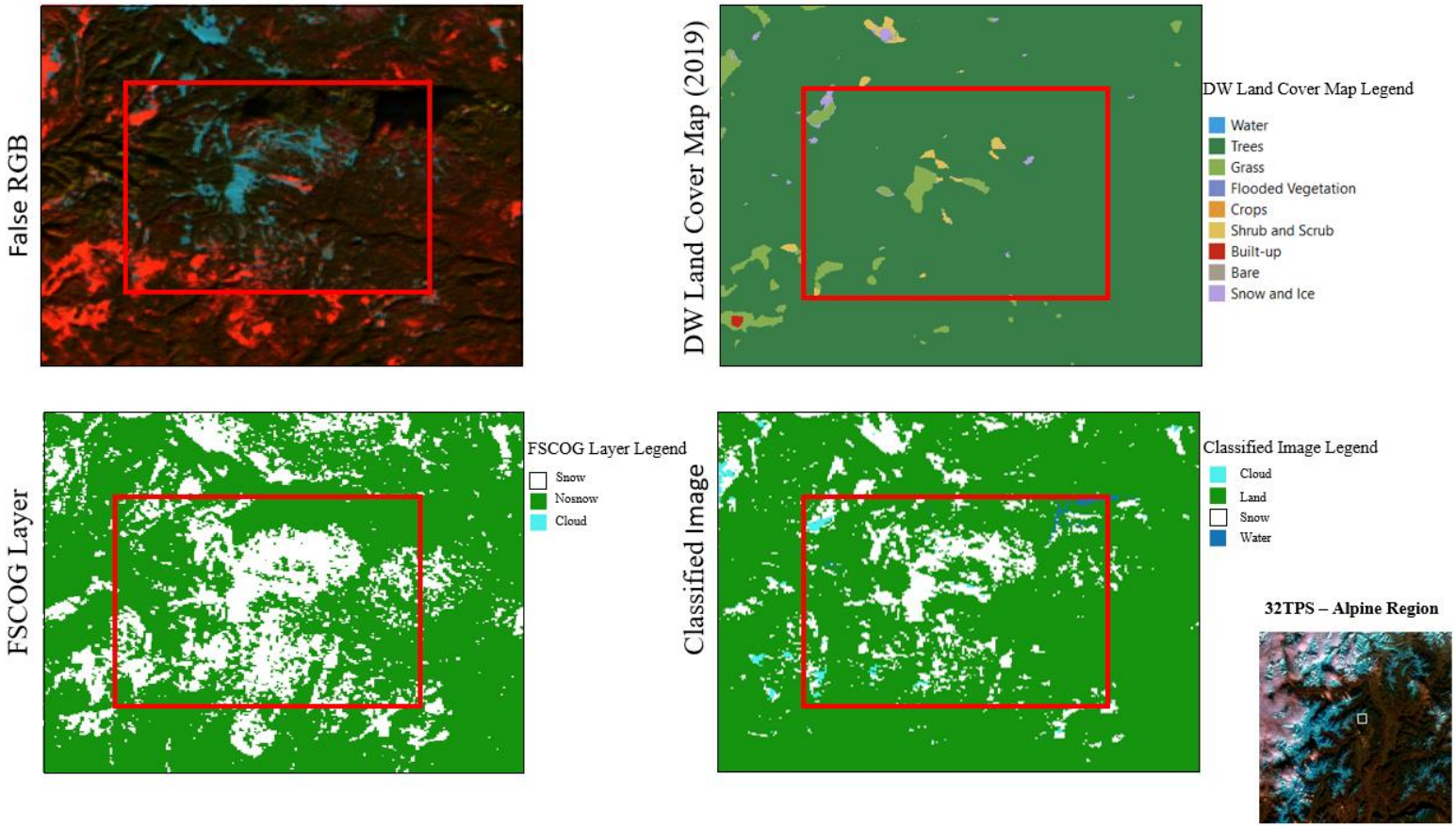


Figure 4.44. Example of underestimation of snow in classified image of Alps Region, 24 January 2019

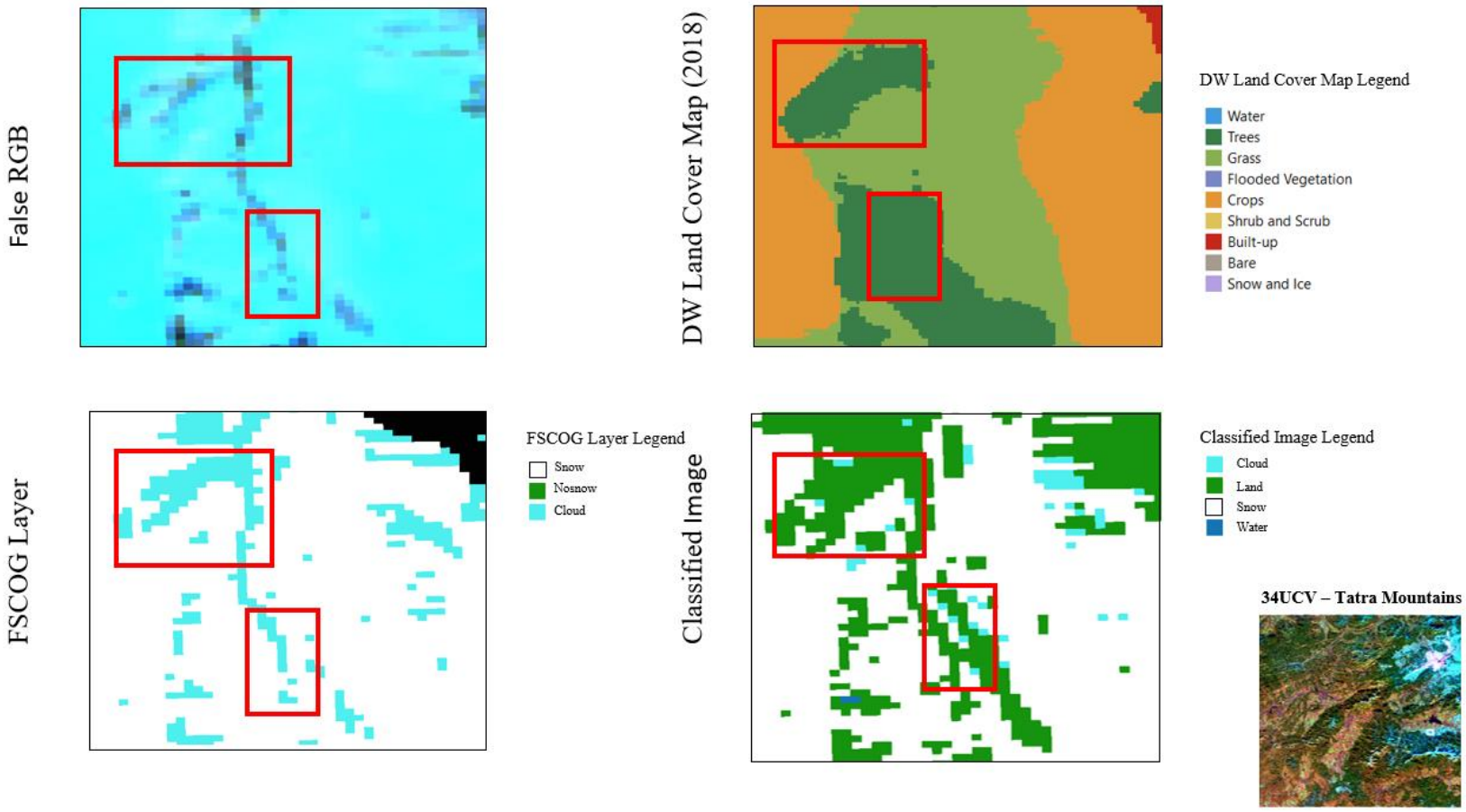


Figure 4.45. Example of underestimation of snow in classified image of Tatra Mountains, 29 November 2018

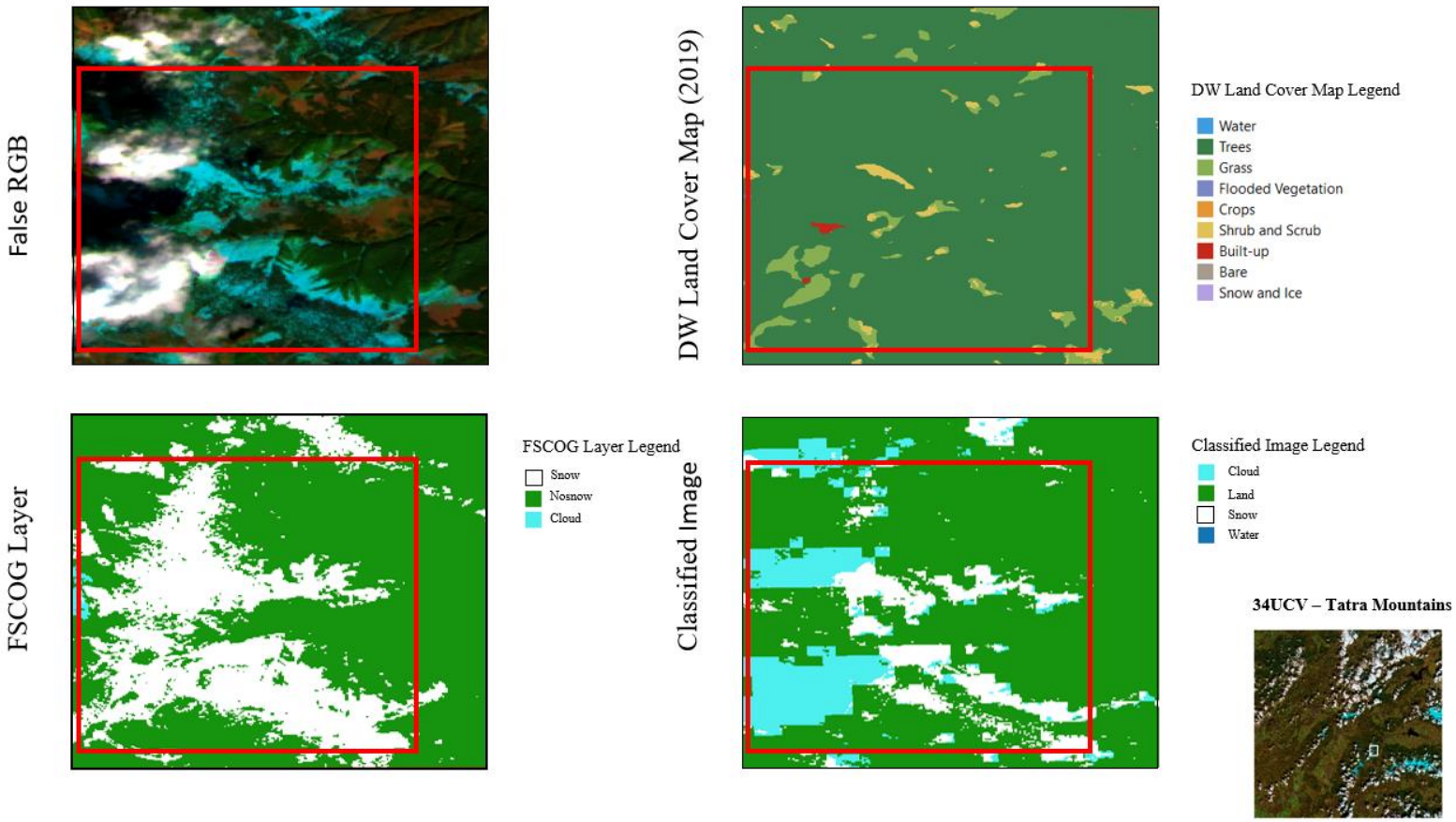


Figure 4.46. Example of underestimation of snow in classified image of Tatra Mountains tile, dated 8 April 2019

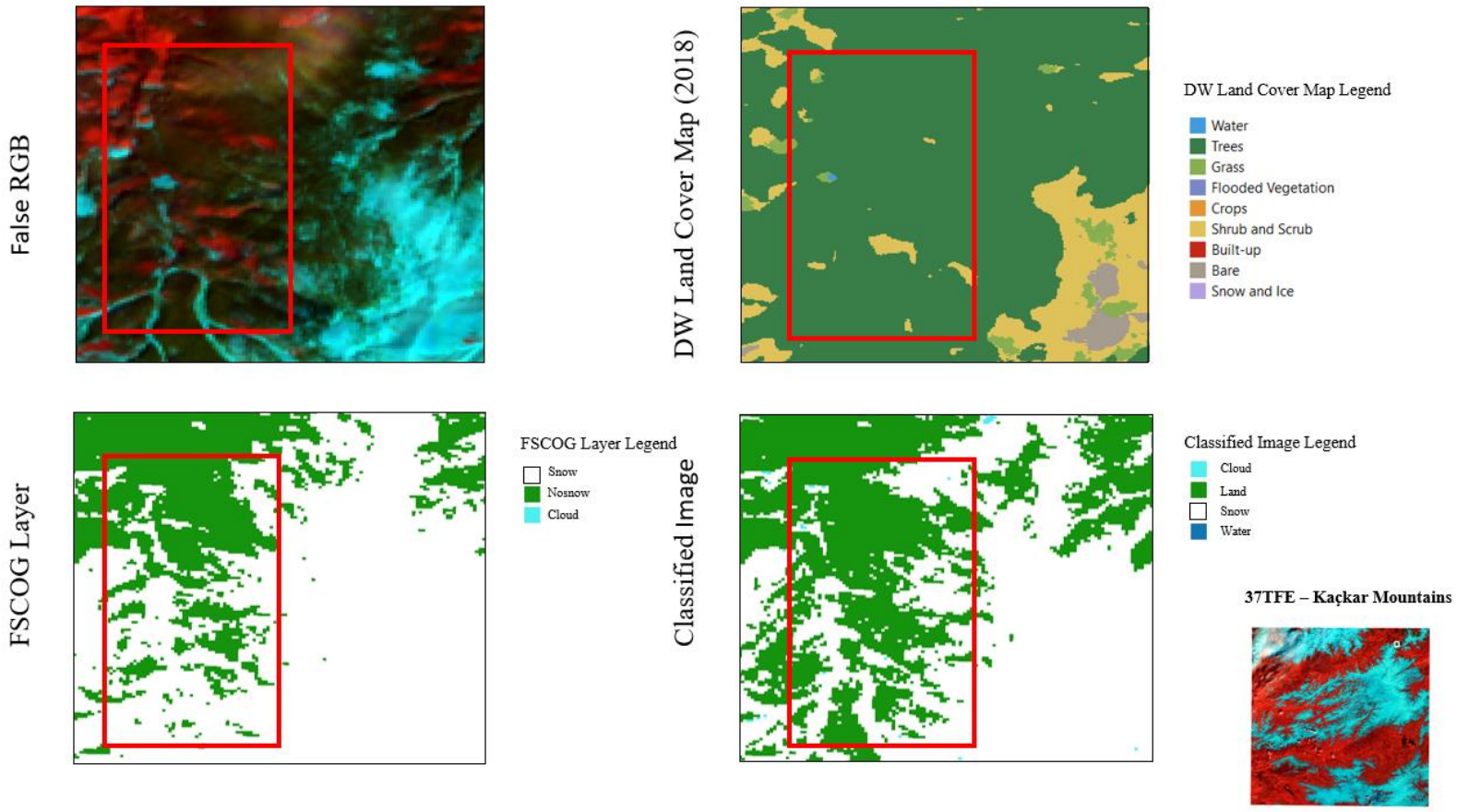


Figure 4.47. Example of underestimation of snow in classified image of Kaçkar Mountains, 19 March 2018

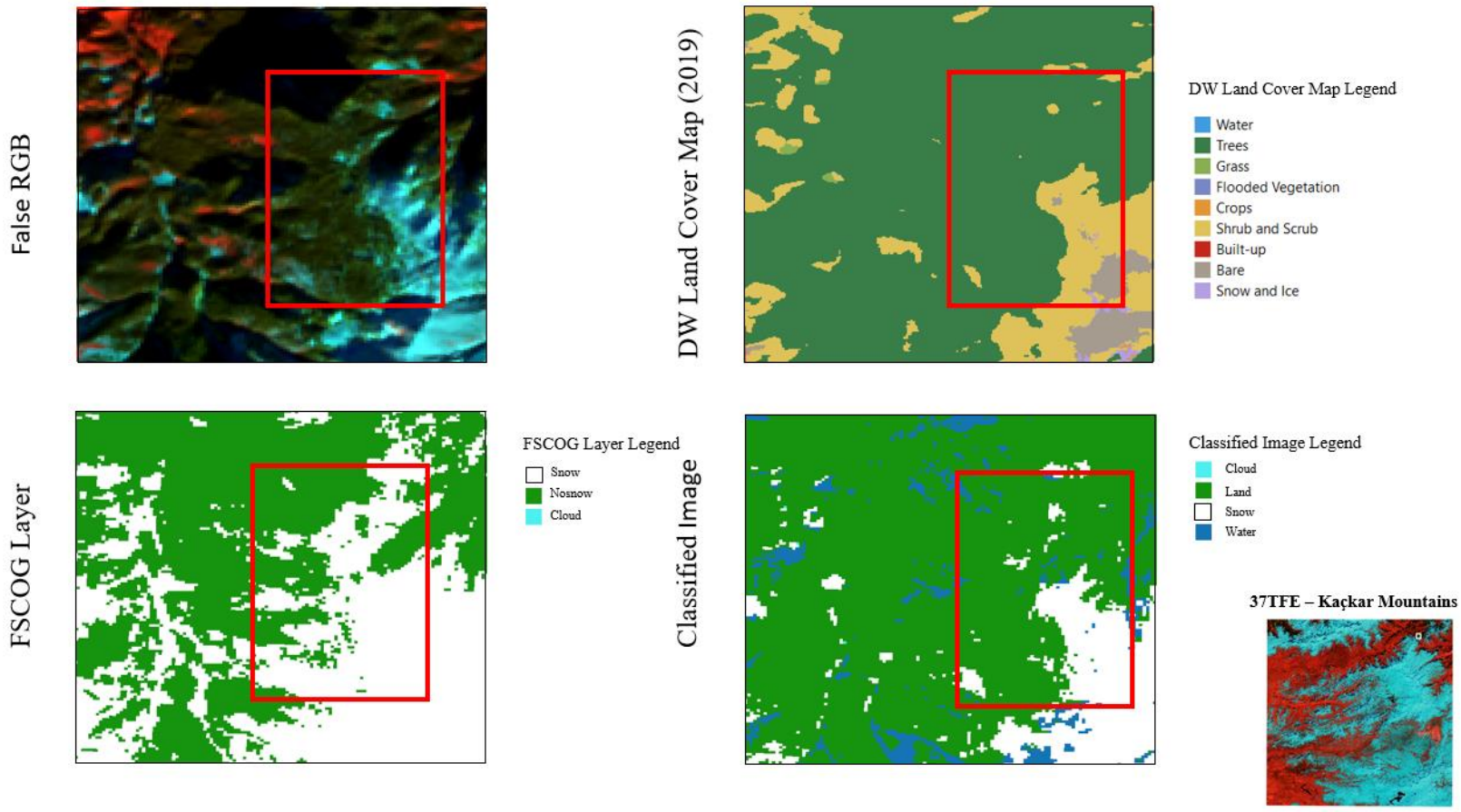


Figure 4.48. Example of underestimation of snow in classified image of Kaçkar Mountains, 19 December 2019

Additionally, it has been observed that, FSCOG layers have a tendency to overestimate cloud cover in some locations, especially in areas with snow. Figure 4.45 shows this overestimation clearly, comparing FSCOG with the original false color RGB image.

In the example locations that are selected in Figure 4.43 - Figure 4.48, percentage of land cover areas (obtained from DW land cover products, cf. Section 3.1.1) have been calculated within only the snow areas. The results of the calculation are shown in Table 4.10. The snow areas have been determined from the fully classified images with *Pca* input combination with 1000 training sample size. The calculation of land cover is done with the aid of ArcMap 10.7 software.

Table 4.10. Percentage of land cover within the snow area (for selected locations in Figure 4.43 - Figure 4.48)

	Location and Date of Classified Sentinel-2 Image					
	Alps Region		Tatra Mountains		Kaçkar Mountains	
Land Cover Class	5 December 2018	24 January2019	29 November 2018	8 April 2019	19 March 2018	19 December 2019
trees	76.6	84.3	5.1	87.6	75.0	12.8
grass	14.6	11.2	27.8	6.3	3.5	0.4
crops	0.0	0.0	65.2	0.0	0.0	0.2
shrubs and scrub	5.2	4.5	0.0	6.0	18.4	66.1
built-up	3.7	0.0	2.0	0.1	0.0	0.0
bare	0.0	0.1	0.0	0.0	3.2	20.5

It is observed from Table 4.10 that, *trees* land cover is the most prominent among all of the land covers in almost all of the selected locations with underestimated snow areas.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

Detection of snow parameters is essential. Accurate classification of snow cover through remote sensing is crucial since it is a much cheaper and easier alternative than the other methods which are currently used for snow cover mapping at both regional and global scales.

The main goal of this study was to evaluate the performance of RF algorithm for determination of snow cover with different input combinations of Sentinel-2 bands (as raw or atmospherically and topographically corrected reflectances), well-known normalized band indices, principal component bands, and DEM data, as well as with different number of training samples per class, on three different regions with different climatic properties, and three different phases of snow cover (snow cover starting to form, snow amount at its peak and lastly, snow starting to melt).

Out of the all three regions, Alps Region yielded the most accurate results, while Kaçkar Mountains yielded the worst. Kaçkar Mountains produce the best results when the peak amount of snow occurs, which is expected, but Tatra Mountains and Alps Region do not. The reason for it is concluded as the difference in the amount of forest cover: Alps Region and Tatra Mountains have significantly more forest than Kaçkar Mountains, which has distorted the results, since forest areas often cause underestimation of snow cover.

Training sample size of 1000 have generally resulted much better accuracy over training sample size of 300, with all of the input combinations. It is determined, with this study, that RF is sensitive to the number of training samples with classification of remote sensing data.

The input combination with best accuracy results were determined as *Pca*, the combination with NDVI, NDSI and NDWI and first three principal components derived from Sentinel-2 bands. Since *Pca* input combination consists of the least number of inputs out of all combinations, this result is foreseen to be very beneficial for future RF classification with remote sensing, as the smaller number of inputs are causing RF algorithm to have less runtime. *Pca* combination is also the best out of all combinations to minimize the confusion between snow and shadow, water and shadow and cloud and snow. *Pca* underestimates snow cover in forest areas, but the underestimation of snow cover in forests is better than the other input combinations. This study recommends another approach combined with RF classification by *Pca* input combination to overcome this problem.

All of the Python scripts that are used in this study can be found in: https://github.com/cansuaksu.

5.2 Recommendations for Future Work

Pca input combination minimizes the underestimation of snow in forested areas, but it does not totally eliminate the problem. Therefore it is proposed, for future studies based on this thesis, to have an additional approach along with RF algorithm and the *Pca* input combination (first three principal components derived from atmospherically and topographically corrected Sentinel-2 bands, NDVI, NDSI and NDWI as inputs), to improve the snow cover identification over forested areas. Other studies suggest some tecniques to mitigate this problem, such as linear mixture or nonlinear mixture analysis (Painter et al., 2003;Rosenthal & Dozier, 1996) and extracting ground-object endmembers (Hao et al., 2019). It is proposed, by Kuter et al. (2018) that *Multivariate Adaptive Regression Splines* yields decent results in forest land cover, with the exception of evergreen forest type while determining snow cover. For evergreen conifereous forests, Wang et al. (2015) presents *Normalized Difference Forest Snow Index* (NDFSI) for better differentiation of snow-covered forests, while Wang et al. (2020) uses an approach of combined

NDFSI and NDSI for the same goal. These are some of the approaches that might be considered for increasing the accuracy of snow detection over forest land cover in future endeavors based on this thesis.

REFERENCES

Abdel-Rahman, E. M., Mutanga, O., Adam, E., & Ismail, R. (2014). Detecting Sirex noctilio grey-attacked and lightning-struck pine trees using airborne hyperspectral data, random forest and support vector machines classifiers. ISPRS Journal of Photogrammetry and Remote Sensing, 88, 48-59. https://doi.org/10.1016/j.isprsjprs.2013.11.013

Acharya, T. D., Subedi, A., & Lee, D. H. (2018). Evaluation of Water Indices for Surface Water Extraction in a Landsat 8 Scene of Nepal. Sensors, 18(8). 10.3390/s18082580

Akyürek, Z., & Şorman, A. Ü. (2002). Monitoring snow-covered areas using NOAA-AVHRR data in the eastern part of Turkey. Hydrological Sciences Journal, 47(2), 243-252.

Armstrong, R. L., & Brodzik, M. J. (2001). Recent northern hemisphere snow extent: A comparison of data derived from visible and microwave satellite sensors. Geophysical Research Letters, 28(19), 3673-3676. https://doi.org/10.1029/2000GL012556

Baba, M. W., Gascoin, S., Hagolle, O., Bourgeois, E., Desjardins, C., & Dedieu, G. (2020). Evaluation of Methods for Mapping the Snow Cover Area at High Spatio-Temporal Resolution with VENμS. Remote Sensing, 12(18). 10.3390/rs12183058

Baetens, L., Desjardins, C., & Hagolle, O. (2019). Validation of Copernicus Sentinel-2 Cloud Masks Obtained from MAJA, Sen2Cor, and FMask Processors Using Reference Cloud Masks Generated with a Supervised Active Learning Procedure. Remote Sensing, 11(4). 10.3390/rs11040433

Balk, B., & Elder, K. (2000). Combining binary decision tree and geostatistical methods to estimate snow distribution in a mountain watershed. Water Resources Research, 36(1), 13-26. https://doi.org/10.1029/1999WR900251

Barrou Dumont, Z., Gascoin, S., Hagolle, O., Michaël, A., Jugier, R., Salgues, G., Morin, S. (2021). Brief communication: Evaluation of the snow cover detection in the Copernicus High Resolution Snow & Ice Monitoring Service.

Barry, R. G., Fallot, J. M., & Armstrong, R. L. (1995). Twentieth-century variability in snow- cover conditions and approaches to detecting and monitoring changes: status and prospects. Progress in Physical Geography: Earth and Environment, 19(4), 520-532. 10.1177/030913339501900405

Basang, D., Barthel, K., & Olseth, J. A. (2017). Satellite and ground observations of snow cover in Tibet during 2001–2015. Remote Sensing, 9(11), 1201.

Bayrakdar, C., & Özdemir, H. (2014). Kaçkar Dağı'nda bakı faktörünün glasiyal ve periglasiyal topografya gelişimi üzerindeki etkisi. Turkish Geographical Review, 0(54), 1-13.

Belgiu, M., & Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. ISPRS Journal of Photogrammetry and Remote Sensing, 114, 24-31. https://doi.org/10.1016/j.isprsjprs.2016.01.011

Beria, H., R. Larsen, J., Ceperley, N. C., Michelon, A., Vennemann, T., & Schaefli, B. (2018). Understanding snow hydrological processes through the lens of stable water isotopes. WIREs Water. https://doi.org/10.1002/wat2.1311

Bian, J., Li, A., Liu, Q., & Huang, C. (2016). Cloud and Snow Discrimination for CCD Images of HJ-1A/B Constellation Based on Spectral Signature and Spatio-Temporal Context. Remote Sensing, 8(1). 10.3390/rs8010031

Biau, G., & Scornet, E. (2016). A random forest guided tour. TEST, 25(2), 197-227. 10.1007/s11749-016-0481-7

Bitner, D., Carroll, T., Cline, D., & Romanov, P. (2002). An assessment of the differences between three satellite snow cover mapping techniques. Hydrological Processes, 16(18), 3723-3733. https://doi.org/10.1002/hyp.1231

Bormann, K. J., Brown, R. D., Derksen, C., & Painter, T. H. (2018). Estimating snow-cover trends from space. Nature Climate Change, 8(11), 924-928.

Boulesteix, A.-L., Janitza, S., Kruppa, J., & König, I. R. (2012). Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. WIREs Data Mining and Knowledge Discovery, 2(6), 493-507. https://doi.org/10.1002/widm.1072

Breiman, L. (1996). Bagging predictors. Machine Learning, 24(2), 123-140. 10.1007/BF00058655

Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5-32. 10.1023/A:1010933404324

Brown, C. F., Brumby, S. P., Guzder-Williams, B., Birch, T., Hyde, S. B., Mazzariello, J., . . . Tait, A. M. (2022). Dynamic World, Near real-time global 10 m land use land cover mapping. Scientific Data, 9(1), 251. 10.1038/s41597-022-01307-4

Brown, R. D. (2000). Northern Hemisphere snow cover variability and change, 1915–97. Journal of climate, 13(13), 2339-2355.

Brown, R. D., & Robinson, D. A. (2011). Northern Hemisphere spring snow cover variability and change over 1922–2010 including an assessment of uncertainty. The Cryosphere, 5(1), 219-229.

Burton-Johnson, A., Black, M., Fretwell, P. T., & Kaluza-Gilbert, J. (2016). An automated methodology for differentiating rock from snow, clouds and sea in Antarctica from Landsat 8 imagery: a new rock outcrop map and area estimation for the entire Antarctic continent. The Cryosphere, 10(4), 1665-1677. 10.5194/tc-10-1665-2016

Bühler, Y., Christen, M., Kowalski, J., & Bartelt, P. (2011). Sensitivity of snow avalanche simulations to digital elevation model quality and resolution. Annals of Glaciology, 52(58), 72-80. 10.3189/172756411797252121

Chai, D., Newsam, S., Zhang, H. K., Qiu, Y., & Huang, J. (2019). Cloud and cloud shadow detection in Landsat imagery based on deep convolutional neural networks. Remote Sensing of Environment, 225, 307-316. https://doi.org/10.1016/j.rse.2019.03.007

Chowdhury, S., Dey, P., Joel-Edgar, S., Bhattacharya, S., Rodriguez-Espindola, O., Abadie, A., & Truong, L. (2022). Unlocking the value of artificial intelligence in human resource management through AI capability framework. Human Resource Management Review, 100899. https://doi.org/10.1016/j.hrmr.2022.100899

Cohen, J. (1960). A coefficient of agreement for nominal scales. Educational and psychological measurement, 20(1), 37-46.

Cohen, J., & Rind, D. (1991). The effect of snow cover on the climate. Journal of Climate, 4(7), 689-706.

Colbeck, S. C. (1982). An overview of seasonal snow metamorphism. Reviews of Geophysics, 20(1), 45-61. https://doi.org/10.1029/RG020i001p00045

Colbeck, S. C. (1987). History of snow-cover research. Journal of Glaciology, 33(S1), 60-65. 10.3189/S0022143000215839

Colditz, R. R. (2015). An Evaluation of Different Training Sample Allocation Schemes for Discrete and Continuous Land Cover Classification Using Decision Tree-Based Algorithms. Remote Sensing, 7(8), 9655-9681. 10.3390/rs70809655

Comola, F., Kok, J. F., Gaume, J., Paterna, E., & Lehning, M. (2017). Fragmentation of wind-blown snow crystals. Geophysical Research Letters, 44(9), 4195-4203.https://doi.org/10.1002/2017GL073039

Congalton, R. G. (1991). A review of assessing the accuracy of classifications of remotely sensed data. Remote Sensing of Environment, 37(1), 35-46. https://doi.org/10.1016/0034-4257(91)90048-B

Congalton, R. G. (2001). Accuracy assessment and validation of remotely sensed and other spatial information. International Journal of Wildland Fire, 10(4), 321-328.

Congalton, R. G., & Green, K. (2009). *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices* (2nd ed.). Boca Raton, FL: CRC press.

Copernicus Open Access Hub. (n.d.) Retrieved April 5, 2020, from https://scihub.copernicus.eu/dhus/#/home

Cutler, A. (2010). Remembering Leo Breiman. The Annals of Applied Statistics, 4(4), 1621-1633. 10.1214/10-AOAS427

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012). Random Forests. In C. Zhang & Y. Ma (Eds.), Ensemble Machine Learning: Methods and Applications (pp. 157-175). Boston, MA: Springer US.

Czyzowska-Wisniewski, E. H., van Leeuwen, W. J., Hirschboeck, K. K., Marsh, S. E., & Wisniewski, W. T. (2015). Fractional snow cover estimation in complex alpine-forested environments using an artificial neural network. Remote Sensing of Environment, 156, 403-417.

Çevik, A., Weber, G.-W., Eyüboğlu, B. M., & Oğuz, K. K. (2017). Voxel-MARS: a method for early detection of Alzheimer's disease by classification of structural brain MRI. Annals of Operations Research, 258(1), 31-57. 10.1007/s10479-017-2405-7

Dankers, R., & De Jong, S. M. (2004). Monitoring snow-cover dynamics in Northern Fennoscandia with SPOT VEGETATION images. International Journal of Remote Sensing, 25(15), 2933-2949. 10.1080/01431160310001618374

Dech, S., Holzwarth, S., Asam, S., Andresen, T., Bachmann, M., Boettcher, M., Kuenzer, C. (2021). Potential and Challenges of Harmonizing 40 Years of AVHRR Data: The TIMELINE Experience. Remote Sensing, 13(18). 10.3390/rs13183618

Deems, J. S., Painter, T. H., & Finnegan, D. C. (2013). Lidar measurement of snow depth: a review. Journal of Glaciology, 59(215), 467-479. 10.3189/2013JoG12J154

Delbart, N., Le Toan, T., Kergoat, L., & Fedotova, V. (2006). Remote sensing of spring phenology in boreal regions: A free of snow-effect method using NOAA-AVHRR and SPOT-VGT data (1982–2004). Remote Sensing of Environment, 101(1), 52-62. https://doi.org/10.1016/j.rse.2005.11.012

Deng, X., Liu, Q., Deng, Y., & Mahadevan, S. (2016). An improved method to construct basic probability assignment based on the confusion matrix for classification problem. Information Sciences, 340-341, 250-261. https://doi.org/10.1016/j.ins.2016.01.033

Díaz-Uriarte, R., & Alvarez de Andrés, S. (2006). Gene selection and classification of microarray data using random forest. BMC Bioinformatics, 7(1), 3. 10.1186/1471-2105-7-3

Dietz, A. J., Kuenzer, C., Gessner, U., & Dech, S. (2012). Remote sensing of snow – a review of available methods. International Journal of Remote Sensing, 33(13), 4094-4134. 10.1080/01431161.2011.640964

Dobreva, I. D., & Klein, A. G. (2011). Fractional snow cover mapping through artificial neural network analysis of MODIS surface reflectance. Remote Sensing of Environment, 115(12), 3355-3366.

Dong, C. (2018). Remote sensing, hydrological modeling and in situ observations in snow cover research: A review. Journal of Hydrology, 561, 573-583.https://doi.org/10.1016/j.jhydrol.2018.04.027

Dynamic World. (n.d.). Retrieved September 14, 2022, from https://developers.google.com/earthengine/datasets/catalog/GOOGLE_DYNAMICWORLD_V1

Eklundh, L., & Singh, A. (1993). A comparative analysis of standardised and unstandardised Principal Components Analysis in remote sensing. International Journal of Remote Sensing, 14(7), 1359-1370. 10.1080/01431169308953962

European Space Agency. (n.d.). Retrieved May 3, 2022, from https://step.esa.int/main/download/snap-download/

FABDEM. (n.d.). Retrieved June 12, 2022, from https://data.bris.ac.uk/data/dataset/25wfy0f9ukoge2gs7a5mqpq2j7#:~:text=The%20FABDEM%20dataset%20is%20licensed,NC%2DSA%204.0%22%20license.&text=This%20dataset%20is%20published%20in,%2F1748%2D9326%2Fac4d4f

Ferran, G., Philippe, M., & François, S. (2009). Sentinel-2 optical high resolution mission for GMES land operational services. Paper presented at the Proc.SPIE.

Foster, J. L., Hall, D. K., Kelly, R. E. J., & Chiu, L. (2009). Seasonal snow extent and snow mass in South America using SMMR and SSM/I passive microwave data (1979–2006). Remote Sensing of Environment, 113(2), 291-305. https://doi.org/10.1016/j.rse.2008.09.010

Foster, J. L., & Rango, A. (1982). Snow cover conditions in the northern Hemisphere during the winter of 1981. Journal of Climatology, 2(2), 171-183. https://doi.org/10.1002/joc.3370020207

Foster, J. L., Sun, C., Walker, J. P., Kelly, R., Chang, A., Dong, J., & Powell, H. (2005). Quantifying the uncertainty in passive microwave snow water equivalent observations. Remote Sensing of Environment, 94(2), 187-203.

Gascoin, S., Grizonnet, M., Bouchet, M., Salgues, G., & Hagolle, O. (2019). Theia Snow collection: High-resolution operational snow cover maps from Sentinel-2 and Landsat-8 data. Earth System Science Data, 11(2), 493-514.

Gharaei-Manesh, S., Fathzadeh, A., & Taghizadeh-Mehrjardi, R. (2016). Comparison of artificial neural network and decision tree models in estimating spatial distribution of snow depth in a semi-arid region of Iran. Cold Regions Science and Technology, 122, 26-35. https://doi.org/10.1016/j.coldregions.2015.11.004

Ghasemian, N., & Akhoondzadeh, M. (2018). Introducing two Random Forest based methods for cloud detection in remote sensing images. Advances in Space Research, 62(2), 288-303. https://doi.org/10.1016/j.asr.2018.04.030

Gislason, P. O., Benediktsson, J. A., & Sveinsson, J. R. (2006). Random Forests for land cover classification. Pattern Recognition Letters, 27(4), 294-300. https://doi.org/10.1016/j.patrec.2005.08.011

Gobiet, A., Kotlarski, S., Beniston, M., Heinrich, G., Rajczak, J., & Stoffel, M. (2014). 21st century climate change in the European Alps—A review. Science of The Total Environment, 493, 1138-1151. https://doi.org/10.1016/j.scitotenv.2013.07.050

Gomarasca, M. A., Tornato, A., Spizzichino, D., Valentini, E., Taramelli, A., Satalino, G., . . . Villa, F. (2019). Sentinel for applications in agriculture. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences,, XLII-3(W6), 91–98. https://doi.org/10.5194/isprs-archives-xlii-3-w6-91-2019

Grogan, P., & Jonasson, S. (2006). Ecosystem CO2 production during winter in a Swedish subarctic region: the relative importance of climate and vegetation type. Global Change Biology, 12(8), 1479-1495.

Guan, H., Li, J., Chapman, M., Deng, F., Ji, Z., & Yang, X. (2013). Integration of orthoimagery and lidar data for object-based urban thematic mapping using random forests. International Journal of Remote Sensing, 34(14), 5166-5186. 10.1080/01431161.2013.788261

Guo, H., & Yang, Y. (2022). Spring snow-albedo feedback from satellite observation, reanalysis and model simulations over the Northern Hemisphere. Science China Earth Sciences, 1-14.

Hall, D. K., & Riggs, G. A. (2011). Encyclopedia of Snow, Ice and Glaciers Netherlands: Dordrecht: Springer

Hall, D. K., Riggs, G. A., & Salomonson, V. V. (1995). Development of methods for mapping global snow cover using moderate resolution imaging spectroradiometer data. Remote Sensing of Environment, 54(2), 127-140.

Hall, D. K., Riggs, G. A., Salomonson, V. V., DiGirolamo, N. E., & Bayr, K. J. (2002). MODIS snow-cover products. Remote Sensing of Environment, 83(1-2), 181-194.

Hao, H., Baireddy, S., Bartusiak, E. R., Konz, L., LaTourette, K., Gribbons, M., Comer, M. L. (2021a, 11-16 July 2021). An Attention-Based System for Damage Assessment Using Satellite Imagery. Paper presented at the 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS.

Hao, S., Jiang, l., Shi, J., Wang, G., & Liu, X. (2019). Assessment of MODIS-Based Fractional Snow Cover Products Over the Tibetan Plateau. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 12(2), 533-548. 10.1109/JSTARS.2018.2879666

Hao, X., Huang, G., Che, T., Ji, W., Sun, X., Zhao, Q., Yang, Q. (2021b). The NIEER AVHRR snow cover extent product over China – a long-term daily snow record for regional climate research. Earth Syst. Sci. Data, 13(10), 4711-4726. 10.5194/essd-13-4711-2021

Haq, M. A., Alshehri, M., Rahaman, G., Ghosh, A., Baral, P., & Shekhar, C. (2021). Snow and glacial feature identification using Hyperion dataset and machine learning algorithms. *Arabian Journal of Geosciences, 14*(15), 1525. doi:10.1007/s12517-021-07434-3

Hashemian, M. S., Abkar, A. A., & Fatemi, S. B. (2004). Study of sampling methods for accuracy assessment of classified remotely sensed data. Paper presented at the International congress for photogrammetry and remote sensing

Hawker, L., Uhe, P., Paulo, L., Sosa, J., Savage, J., Sampson, C., & Neal, J. (2022). A 30 m global map of elevation with forests and buildings removed. Environmental Research Letters, 17(2), 024016. 10.1088/1748-9326/ac4d4f

Hedstrom, N. R., & Pomeroy, J. W. (1998). Measurements and modelling of snow interception in the boreal forest. Hydrological Processes, 12, 1611-1625.

Holloway, J., & Mengersen, K. (2018). Statistical Machine Learning Methods and Remote Sensing for Sustainable Development Goals: A Review. Remote Sensing, 10(9). 10.3390/rs10091365

Homer, C., Huang, C., Yang, L., Wylie, B., & Coan, M. (2004). Development of a 2001 National Land-Cover Database for the United States. Photogrammetric Engineering & Remote Sensing, 70(7), 829-840. 10.14358/PERS.70.7.829

Hou, J., Huang, C., Chen, W., & Zhang, Y. (2021). Improving Snow Estimates Through Assimilation of MODIS Fractional Snow Cover Data Using Machine Learning Algorithms and the Common Land Model. Water Resources Research, 57(7), e2020WR029010. https://doi.org/10.1029/2020WR029010

Hou, J., Huang, C., Zhang, Y., & Guo, J. (2020). On the Value of Available MODIS and Landsat8 OLI Image Pairs for MODIS Fractional Snow Cover Mapping Based on an Artificial Neural Network. IEEE Transactions on Geoscience and Remote Sensing, 58(6), 4319-4334.10.1109/TGRS.2019.2963075

Houborg, R., & McCabe, M. F. (2018). A hybrid training approach for leaf area index estimation via Cubist and random forests machine-learning. ISPRS Journal of Photogrammetry and Remote Sensing, 135, 173-188. https://doi.org/10.1016/j.isprsjprs.2017.10.004

Jensen, J. R. (2006). Introductory digital image processing: a remote sensing perspective: Prentice-Hall Inc.

Jeppesen, J. H., Jacobsen, R. H., Inceoglu, F., & Toftegaard, T. S. (2019). A cloud detection algorithm for satellite imagery based on deep learning. Remote

Sensing of Environment, 229, 247-259. https://doi.org/10.1016/j.rse.2019.03.039

Jiang, Q., Li, W., Fan, Z., He, X., Sun, W., Chen, S., Wen, J., Gao, J., & Wang, J. (2021). Evaluation of the ERA5 reanalysis precipitation dataset over Chinese Mainland. *Journal of Hydrology*, *595*, 125660. https://doi.org/https://doi.org/10.1016/j.jhydrol.2020.125660

Jin, Y., Liu, X., Chen, Y., & Liang, X. (2018). Land-cover mapping using Random Forest classification and incorporating NDVI time-series and texture: a case study of central Shandong. International Journal of Remote Sensing, 39(23), 8703-8723. 10.1080/01431161.2018.1490976

Jog, S., & Dixit, M. (2016, 9-11 June 2016). Supervised classification of satellite images. Paper presented at the 2016 Conference on Advances in Signal Processing (CASP).

Jonas, T., Marty, C., & Magnusson, J. (2009). Estimating the snow water equivalent from snow depth measurements in the Swiss Alps. Journal of Hydrology, 378(1-2), 161-167.

Klein, A. G., & Barnett, A. C. (2003). Validation of daily MODIS snow cover maps of the Upper Rio Grande River Basin for the 2000–2001 snow year. Remote Sensing of Environment, 86(2), 162-176.https://doi.org/10.1016/S0034-4257(03)00097-X

Klein, A. G., Hall, D. K., & Riggs, G. A. (1998). Improving snow cover mapping in forests through the use of a canopy reflectance model. Hydrological Processes, 12(10-11), 1723-1744. https://doi.org/10.1002/(SICI)1099-1085(199808/09)12:10/11<1723::AID-HYP691>3.0.CO;2-2

Ko, B. C., Kim, H. H., & Nam, J. Y. (2015). Classification of Potential Water Bodies Using Landsat 8 OLI and a Combination of Two Boosted Random Forest Classifiers. Sensors, 15(6), 13763-13777. 10.3390/s150613763

Kunkel, K. E., Robinson, D. A., Champion, S., Yin, X., Estilow, T., & Frankson, R. M. (2016). Trends and extremes in Northern Hemisphere snow characteristics. Current Climate Change Reports, 2(2), 65-73.

Kuter, S. (2021). Completing the machine learning saga in fractional snow cover estimation from MODIS Terra reflectance data: Random forests versus support vector regression. Remote Sensing of Environment, 255, 112294.

Kuter, S., Akyurek, Z., & Weber, G.-W. (2018). Retrieval of fractional snow covered area from MODIS data by multivariate adaptive regression splines. Remote Sensing of Environment, 205, 236-252.

Kuter, S., Bolat, K., & Akyurek, Z. (2022). A machine learning-based accuracy enhancement on EUMETSAT H-SAF H35 effective snow-covered area product. Remote Sensing of Environment, 272, 112947.

Languille, F., Déchoz, C., Gaudel, A., Greslou, D., De Lussy, F., Trémas, T., & Poulain, V. (2015). Sentinel-2 geometric image quality commissioning: first results. Paper presented at the Proc.SPIE.

Lapin, M., Faško, P., & Pecho, J. (2007). Snow cover variability and trends in the tatra mountains in 1921-2006. Paper presented at the Proceedings of the 29th International Conference on Alpine Meteorology Chambéry.

Lawrence, R. L., Wood, S. D., & Sheley, R. L. (2006). Mapping invasive plants using hyperspectral imagery and Breiman Cutler classifications (randomForest). Remote Sensing of Environment, 100(3), 356-362. https://doi.org/10.1016/j.rse.2005.10.014

Li, C., Wang, J., Wang, L., Hu, L., & Gong, P. (2014). Comparison of Classification Algorithms and Training Sample Sizes in Urban Land Classification with Landsat Thematic Mapper Imagery. Remote Sensing, 6(2), 964-983. 10.3390/rs6020964

Li, Q., Yang, T., & Li, L.-h. (2021). Impact of forcing data and land surface properties on snow simulation in a regional climate model: a case study over

the Tianshan Mountains, Central Asia. Journal of Mountain Science, 18(12), 3147-3164. 10.1007/s11629-020-6621-2

Liang, H., Huang, X., Sun, Y., Wang, Y., & Liang, T. (2017). Fractional snow-cover mapping based on MODIS and UAV data over the Tibetan Plateau. Remote Sensing, 9(12), 1332.

Liu, C., Huang, X., Li, X., & Liang, T. (2020a). MODIS fractional snow cover mapping using machine learning technology in a mountainous area. Remote Sensing, 12(6), 962.

Liu, Y., Chen, X., Hao, J.-S., & Li, L.-h. (2020b). Snow cover estimation from MODIS and Sentinel-1 SAR data using machine learning algorithms in the western part of the Tianshan Mountains. Journal of Mountain Science, 17(4), 884-897. 10.1007/s11629-019-5723-1

Liu, Y., Peters-Lidard, C. D., Kumar, S., Foster, J. L., Shaw, M., Tian, Y., & Fall, G. M. (2013). Assimilating satellite-based snow depth and snow cover products for improving snow predictions in Alaska. Advances in Water Resources, 54, 208-227.

Louis, J. (2021). Sen2Cor ATBD Document. Retrieved December 16, 2022, from https://step.esa.int/thirdparties/sen2cor/2.10.0/docs/S2-PDGS-MPC-L2A-ATBD-V2.10.0.pdf

Lu, D., Mausel, P., Brondizio, E., & Moran, E. (2002). Assessment of atmospheric correction methods for Landsat TM data applicable to Amazon basin LBA research. International Journal of Remote Sensing, 23(13), 2651-2671. 10.1080/01431160110109642

Lucas, R. M., & Harrison, A. R. (1990). Snow observation by satellite: A review. Remote Sensing Reviews, 4(2), 285-348.10.1080/02757259009532109

Luis, G.-C., Julia, A.-L., Gonzalo, M.-G., Jordi, M.-M., & Gustau, C.-V. (2017). Cloud masking and removal in remote sensing image time series. Journal of Applied Remote Sensing, 11(1), 015005. 10.1117/1.JRS.11.015005

Luo, J., Dong, C., Lin, K., Chen, X., Zhao, L., & Menzel, L. (2022). Mapping snow cover in forests using optical remote sensing, machine learning and time-lapse photography. Remote Sensing of Environment, 275, 113017. https://doi.org/10.1016/j.rse.2022.113017

Luo, Y., Trishchenko, A. P., & Khlopenkov, K. V. (2008). Developing clear-sky, cloud and cloud shadow mask for producing clear-sky composites at 250-meter spatial resolution for the seven MODIS land bands over Canada and North America. Remote Sensing of Environment, 112(12), 4167-4185. https://doi.org/10.1016/j.rse.2008.06.010

Luzzini, F. (2011). Matrices, not seeds. Vallisneri's research on mines: between empiricism and philosophy. History of Research in Mineral Resources, 13.

Magdalena, M.-K., Bringfried, P., Jerome, L., Vincent, D., Uwe, M.-W., & Ferran, G. (2017). Sen2Cor for Sentinel-2. Paper presented at the Proc.SPIE.

Mahiny, A. S., & Turner, B. J. (2007). A Comparison of Four Common Atmospheric Correction Methods. Photogrammetric Engineering & Remote Sensing, 73(4), 361-368. 10.14358/PERS.73.4.361

Mankin, J. S., Viviroli, D., Singh, D., Hoekstra, A. Y., & Diffenbaugh, N. S. (2015). The potential for snow to supply human water demand in the present and future. Environmental Research Letters, 10(11), 114016.

Mather, P. M., & Koch, M. (2011). Computer processing of remotely-sensed images: an introduction: John Wiley & Sons.

Maurer, E. P., Rhoads, J. D., Dubayah, R. O., & Lettenmaier, D. P. (2003). Evaluation of the snow-covered area data product from MODIS. Hydrological Processes, 17(1), 59-71.

Maxwell, A. E., Warner, T. A., & Fang, F. (2018). Implementation of machine-learning classification in remote sensing: an applied review. International

Journal of Remote Sensing, 39(9), 2784-2817. 10.1080/01431161.2018.1433343

McCabe, G. J., & Wolock, D. M. (2010). Long-term variability in Northem Hemisphere snow cover and associations with warmer winters. Climatic Change, 99(1), 141-153.

McFeeters, S. K. (1996). The use of the Normalized Difference Water Index (NDWI) in the delineation of open water features. International Journal of Remote Sensing, 17(7), 1425-1432. 10.1080/01431169608948714

Metsämäki, S. J., Anttila, S. T., Markus, H. J., & Vepsäläinen, J. M. (2005). A feasible method for fractional snow cover mapping in boreal zone based on a reflectance model. Remote Sensing of Environment, 95(1), 77-95.

Moosavi, V., Malekinezhad, H., & Shirmohammadi, B. (2014). Fractional snow cover mapping from MODIS data using wavelet-artificial intelligence hybrid models. Journal of Hydrology, 511, 160-170.

Mukherjee, S., Joshi, P. K., Mukherjee, S., Ghosh, A., Garg, R. D., & Mukhopadhyay, A. (2013). Evaluation of vertical accuracy of open source Digital Elevation Model (DEM). International Journal of Applied Earth Observation and Geoinformation, 21, 205-217. https://doi.org/10.1016/j.jag.2012.09.004

Mueller-Wilm, U. (2020). Sen2Cor Configuration and User Manual. Retrieved December 16, 2022, from http://step.esa.int/thirdparties/sen2cor/2.9.0/docs/S2-PDGS-MPC-L2A-SUM-V2.9.0.pdf

Muñoz-Sabater, J., Dutra, E., Agustí-Panareda, A., Albergel, C., Arduini, G., Balsamo, G., Thépaut, J. N. (2021). ERA5-Land: a state-of-the-art global reanalysis dataset for land applications. Earth Syst. Sci. Data, 13(9), 4349-4383.10.5194/essd-13-4349-2021

Mutanga, O ., Adam, E., & Cho, M. A. (2012). High density biomass estimation for wetland vegetation using WorldView-2 imagery and random forest regression algorithm. International Journal of Applied Earth Observation and Geoinformation, 18, 399-406. https://doi.org/10.1016/j.jag.2012.03.012

Nagajothi, V., Priya, M. G., & Sharma, P. (2019). Snow cover estimation of western himalayas using sentinel-2 high spatial resolution data. Indian Journal of Ecology, 46(1), 88-93.

Nijhawan, R., Das, J., & Balasubramanian, R. (2018a). A Hybrid CNN + Random Forest Approach to Delineate Debris Covered Glaciers Using Deep Features. Journal of the Indian Society of Remote Sensing, 46(6), 981-989. 10.1007/s12524-018-0750-x

Nijhawan, R., Raman, B., & Das, J. (2018b). Meta-Classifier Approach with ANN, SVM, Rotation Forest, and Random Forest for Snow Cover Mapping. Paper presented at the Proceedings of 2nd International Conference on Computer Vision & Image Processing, Singapore.

Nitze, I., Grosse, G., Jones, B. M., Romanovsky, V. E., & Boike, J. (2018). Remote sensing quantifies widespread abundance of permafrost region disturbances across the Arctic and Subarctic. Nature Communications, 9(1), 5423. 10.1038/s41467-018-07663-3

Nolin, A. W. (2010). Recent advances in remote sensing of seasonal snow. Journal of Glaciology, 56(200), 1141-1150. 10.3189/002214311796406077

Ozdarici Ok, A., & Akyurek, Z. (2012). A segment-based approach to classify agricultural lands by using multi-temporal optical and microwave data. International Journal of Remote Sensing, 33(22), 7184-7204. 10.1080/01431161.2012.700423

Padró, J.-C., Muñoz, F.-J., Ávila, L. Á., Pesquer, L., & Pons, X. (2018). Radiometric Correction of Landsat-8 and Sentinel-2A Scenes Using Drone Imagery in Synergy with Field Spectroradiometry. *Remote Sensing*, *10*(11). 10.3390/rs10111687

Painter, T. H., Dozier, J., Roberts, D. A., Davis, R. E., & Green, R. O. (2003). Retrieval of subpixel snow-covered area and grain size from imaging spectrometer data. Remote Sensing of Environment, 85(1), 64-77.

Painter, T. H., Rittger, K., McKenzie, C., Slaughter, P., Davis, R. E., & Dozier, J. (2009). Retrieval of subpixel snow covered area, grain size, and albedo from MODIS. Remote Sensing of Environment, 113(4), 868-879.

Painter, T. H., Roberts, D. A., Green, R. O., & Dozier, J. (1998). The Effect of Grain Size on Spectral Mixture Analysis of Snow-Covered Area from AVIRIS Data. Remote Sensing of Environment, 65(3), 320-332. https://doi.org/10.1016/S0034-4257(98)00041-8

Pal, M., & Mather, P. M. (2005). Support vector machines for classification in remote sensing. International Journal of Remote Sensing, 26(5), 1007-1011. 10.1080/01431160512331314083

Parajka, J., & Blöschl, G. (2006). Validation of MODIS snow cover images over Austria. Hydrology and Earth System Sciences

Pelletier, C., Valero, S., Inglada, J., Champion, N., & Dedieu, G. (2016). Assessing the robustness of Random Forests to map land cover with high resolution satellite image time series over large areas. Remote Sensing of Environment, 187, 156-168. https://doi.org/10.1016/j.rse.2016.10.010

Prado, Álvaro J., Michałek, Maciej M., & Cheein, Fernando A. (2018). Machine-learning based approaches for self-tuning trajectory tracking controllers under terrain changes in repetitive tasks. Engineering Applications of Artificial Intelligence, 67, 63-80. https://doi.org/10.1016/j.engappai.2017.09.013

Qi, W., Feng, L., Kuang, X., Zheng, C., Liu, J., Chen, D., . . . Yao, Y. (2022). Divergent and Changing Importance of Glaciers and Snow as Natural Water Reservoirs in the Eastern and Southern Tibetan Plateau. Journal of Geophysical Research: Atmospheres, 127(7), e2021JD035888. https://doi.org/10.1029/2021JD035888

Raiyani, K., Gonçalves, T., Rato, L., Salgueiro, P., & Marques da Silva, J. R. (2021). Sentinel-2 Image Scene Classification: A Comparison between Sen2Cor and a Machine Learning Approach. Remote Sensing, 13(2). 10.3390/rs13020300

Rango, A. (1993). II. Snow hydrology processes and remote sensing. Hydrological Processes, 7(2), 121-138. https://doi.org/10.1002/hyp.3360070204

Reber, R., Akçar, N., Tikhomirov, D., Yesilyurt, S., Vockenhuber, C., Yavuz, V., Schlüchter, C. (2022). LGM Glaciations in the Northeastern Anatolian Mountains: New Insights. Geosciences, 12(7). 10.3390/geosciences12070257

Renfrew, I. A., Barrell, C., Elvidge, A. D., Brooke, J. K., Duscha, C., King, J. C., Kristiansen, J., Cope, T. L., Moore, G. W. K., Pickart, R. S., Reuder, J., Sandu, I., Sergeev, D., Terpstra, A., Våge, K., & Weiss, A. (2021). An evaluation of surface meteorology and fluxes over the Iceland and Greenland Seas in ERA5 reanalysis: The impact of sea ice distribution. *Quarterly Journal of the Royal Meteorological Society*, *147*(734), 691–712. https://doi.org/https://doi.org/10.1002/qj.3941

Riano, D., Chuvieco, E., Salas, J., & Aguado, I. (2003). Assessment of different topographic corrections in Landsat-TM data for mapping vegetation types (2003). IEEE Transactions on Geoscience and Remote Sensing, 41(5), 1056-1061. 10.1109/TGRS.2003.811693

Richter, R., Schläpfer, D., & Müller, A. (2006). An automatic atmospheric correction algorithm for visible/NIR imagery. *International Journal of Remote Sensing, 27*(10), 2077-2085.

Rittger, K., Painter, T. H., & Dozier, J. (2013). Assessment of methods for mapping snow cover from MODIS. Advances in Water Resources, 51, 367-380. https://doi.org/10.1016/j.advwatres.2012.03.002

Robinson, D. A. J. G. d. (1993). Monitoring northern hemisphere snow cover.

Rodarmel, C., & Shan, J. (2002). Principal component analysis for hyperspectral image classification. Surveying and Land Information Science, 62(2), 115-122.

Romanov, P., Gutman, G., & Csiszar, I. (2002). Satellite-derived snow cover maps for north America: Accuracy assessment. Advances in Space Research, 30(11), 2455-2460. https://doi.org/10.1016/S0273-1177(02)80304-0

Rosenthal, W., & Dozier, J. (1996). Automated Mapping of Montane Snow Cover at Subpixel Resolution from the Landsat Thematic Mapper. Water Resources Research, 32(1), 115-130. https://doi.org/10.1029/95WR02718

Rouquié, B., Hagolle, O., Bréon, F.-M., Boucher, O., Desjardins, C., & Rémy, S. (2017). Using Copernicus Atmosphere Monitoring Service Products to Constrain the Aerosol Type in the Atmospheric Correction Processor MAJA. Remote Sensing, 9(12). 10.3390/rs9121230

Rößler, S., & Dietz, A. J. (2022). Detection of Snow Cover from Historical and Recent AVHHR Data Thematic TIMELINE Processor. Geomatics, 2(1), 144-160. 10.3390/geomatics2010009

Rouquié, B., Hagolle, O., Bréon, F.-M., Boucher, O., Desjardins, C., & Rémy, S. (2017). Using Copernicus Atmosphere Monitoring Service Products to Constrain the Aerosol Type in the Atmospheric Correction Processor MAJA. *Remote Sensing*, *9*(12). 10.3390/rs9121230

Saavedra, F. A., Kampf, S. K., Fassnacht, S. R., & Sibold, J. S. (2017). A snow climatology of the Andes Mountains from MODIS snow cover data. *International Journal of Climatology*, *37*(3), 1526–1539. https://doi.org/https://doi.org/10.1002/joc.4795

Schaffhauser, A., Adams, M., Fromm, R., Jörg, P., Luzi, G., Noferini, L., & Sailer, R. (2008). Remote sensing based retrieval of snow cover properties. Cold Regions Science and Technology, 54(3), 164-175. https://doi.org/10.1016/j.coldregions.2008.07.007

Schanda, E., Matzler, C., & Kunzi, K. (1983). Microwave remote sensing of snow cover. International Journal of Remote Sensing, 4(1), 149-158. 10.1080/01431168308948536

Singh, B. (2013). Climate Change: Realities, Impacts Over Ice Cap, Sea Level and Risks: BoD–Books on Demand.

Singh, S. K., Kulkarni, A. V., & Chaudhary, B. S. (2010). Hyperspectral analysis of snow reflectance to understand the effects of contamination and grain size. Annals of Glaciology, 51(54), 83-88. 10.3189/172756410791386535

Singh, S. K., Rathore, B. P., Bahuguna, I. M., & Ajai. (2014). Snow cover variability in the Himalayan–Tibetan region. International Journal of Climatology, 34(2), 446-452. https://doi.org/10.1002/joc.3697

Skakun, S., Wevers, J., Brockmann, C., Doxani, G., Aleksandrov, M., Batič, M., . . . Žust, L. (2022). Cloud Mask Intercomparison eXercise (CMIX): An evaluation of cloud masking algorithms for Landsat 8 and Sentinel-2. Remote Sensing of Environment, 274, 112990. https://doi.org/10.1016/j.rse.2022.112990

Snehmani, Bhardwaj, A., Singh, M. K., Gupta, R. D., Joshi, P. K., & Ganju, A. (2015). Modelling the hypsometric seasonal snow cover using meteorological parameters. Journal of Spatial Science, 60(1), 51-64. 10.1080/14498596.2014.943310

Song, C., Woodcock, C. E., Seto, K. C., Lenney, M. P., & Macomber, S. A. (2001). Classification and Change Detection Using Landsat TM Data: When and How to Correct Atmospheric Effects? Remote Sensing of Environment, 75(2), 230-244. https://doi.org/10.1016/S0034-4257(00)00169-3

Sturm, M., Holmgren, J., & Liston, G. E. (1995). A seasonal snow cover classification system for local to global applications. Journal of Climate, 8(5), 1261-1283.

Tait, A. (1998). Estimation of snow water equivalent using passive microwave radiation data. Remote Sensing of Environment, 64(3), 286-291.

Takala, M., Luojus, K., Pulliainen, J., Derksen, C., Lemmetyinen, J., Kärnä, J.-P., . . . Bojkov, B. (2011). Estimating northern hemisphere snow water equivalent for climate research through assimilation of space-borne radiometer data and ground-based measurements. Remote Sensing of Environment, 115(12), 3517-3529.

Talukdar, S., Singha, P., Mahato, S., Shahfahad, Pal, S., Liou, Y.-A., & Rahman, A. (2020). Land-Use Land-Cover Classification by Machine Learning Classifiers for Satellite Observations—A Review. Remote Sensing, 12(7). 10.3390/rs12071135

Tekeli, A. E., Akyürek, Z., Şorman, A. A., Şensoy, A., & Şorman, A. Ü. (2005). Using MODIS snow cover maps in modeling snowmelt runoff process in the eastern part of Turkey. Remote Sensing of Environment, 97(2), 216-230.

Tona, C., & Bua, R. (2018). Open Source Data Hub System: free and open framework to enable cooperation to disseminate Earth Observation data and geo-spatial information. In *Geophysical Research Abstracts* (Vol. 20). Copernicus Open Access Hub. April 5,2020, from https://scihub.copernicus.eu/dhus/#/home

Tsai, C.-W., Lai, C.-F., Chao, H.-C., & Vasilakos, A. V. (2015). Big data analytics: a survey. Journal of Big Data, 2(1), 21. 10.1186/s40537-015-0030-3

Tsai, Y.-L. S., Dietz, A., Oppelt, N., & Kuenzer, C. (2019a). Remote Sensing of Snow Cover Using Spaceborne SAR: A Review. Remote Sensing, 11(12). 10.3390/rs11121456

Tsai, Y.-L. S., Dietz, A., Oppelt, N., & Kuenzer, C. (2019b). Wet and Dry Snow Detection Using Sentinel-1 SAR Data for Mountainous Areas with a Machine Learning Technique. Remote Sensing, 11(8). 10.3390/rs11080895

Tudoroiu, M., Eccel, E., Gioli, B., Gianelle, D., Schume, H., Genesio, L., & Miglietta, F. (2016). Negative elevation-dependent warming trend in the Eastern Alps. Environmental Research Letters, 11(4), 044021. 10.1088/1748-9326/11/4/044021

Tyralis, H., & Papacharalampous, G. (2017). Variable Selection in Time Series Forecasting Using Random Forests. Algorithms, 10(4). 10.3390/a10040114

Valt, M., & Cianfarra, P. (2010). Recent snow cover variability in the Italian Alps. Cold Regions Science and Technology, 64(2), 146-157. https://doi.org/10.1016/j.coldregions.2010.08.008

Van Niel, T. G., McVicar, T. R., & Datt, B. (2005). On the relationship between training sample size and data dimensionality: Monte Carlo analysis of broadband multi-temporal classification. Remote Sensing of Environment, 98(4), 468-480. https://doi.org/10.1016/j.rse.2005.08.011

Vanonckelen, S., Lhermitte, S., & Van Rompaey, A. (2013). The effect of atmospheric and topographic correction methods on land cover classification accuracy. International Journal of Applied Earth Observation and Geoinformation, 24, 9-21. https://doi.org/10.1016/j.jag.2013.02.003

Wadoux, A. M. J. C., Brus, D. J., & Heuvelink, G. B. M. (2019). Sampling design optimization for soil mapping with random forest. Geoderma, 355, 113913. https://doi.org/10.1016/j.geoderma.2019.113913

Wang, X., Wu, C., Peng, D., Gonsamo, A., & Liu, Z. (2018). Snow cover phenology affects alpine vegetation growth dynamics on the Tibetan Plateau: Satellite observed evidence, impacts of different biomes, and climate drivers. Ecological Indicators, 256, 61-74.

Wang, X.-Y., Wang, J., Jiang, Z.-Y., Li, H.-Y., & Hao, X.-H. (2015). An Effective Method for Snow-Cover Mapping of Dense Coniferous Forests in the Upper Heihe River Basin Using Landsat Operational Land Imager Data. Remote Sensing, 7(12), 17246-17257. 10.3390/rs71215882

Wang, Y., Su, J., Zhai, X., Meng, F., & Liu, C. (2022). Snow Coverage Mapping by Learning from Sentinel-2 Satellite Multispectral Images via Machine Learning Algorithms. Remote Sensing, 14(3). 10.3390/rs14030782

Williams, D. L., Goward, S., & Arvidson, T. (2006). Landsat. Photogrammetric Engineering & Remote Sensing, 72(10), 1171-1178. 10.14358/PERS.72.10.1171

Xiao, X., Zhang, T., Zhong, X., Shao, W., & Li, X. (2018). Support vector regression snow-depth retrieval algorithm using passive microwave remote sensing data. Remote Sensing of Environment, 210, 48-64. https://doi.org/10.1016/j.rse.2018.03.008

Zekoll, V., Main-Knorn, M., Alonso, K., Louis, J., Frantz, D., Richter, R., & Pflug, B. (2021). Comparison of Masking Algorithms for Sentinel-2 Imagery. Remote Sensing, 13(1). 10.3390/rs13010137

Zhang, H., Zhang, F., Zhang, G., Che, T., Yan, W., Ye, M., & Ma, N. (2019). Ground-based evaluation of MODIS snow cover product V6 across China: Implications for the selection of NDSI threshold. *Science of The Total Environment, 651*, 2712-2726. doi:https://doi.org/10.1016/j.scitotenv.2018.10.128

Zhu, J., Shi, J., & Wang, Y. (2012). Subpixel snow mapping of the Qinghai–Tibet Plateau using MODIS data. International Journal of Applied Earth Observation and Geoinformation.

APPENDICES

A. Python Script for Obtaining Training and Test Datasets

```
import rasterio.mask
from shapely.geometry import shape, mapping
from shapely.ops import unary_union
import fiona
import itertools
import pandas as pd
import numpy as np
import glob
import os


'''THIS SCRIPT IS FOR THE INPUT COMBINATIONS OF ATMO_TOPO AND SC_ONLY. SIMILAR '''

'''BEFORE THIS SCRIPT, GETTING A COMPOSITE IMAGE VIA ARCMAP IS NEEDED'''


path = "D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/23_Jan_2020_sc_only/"

def get_test_data(path):
    with fiona.open(path + "/Test_Region/Clip_Frame.shp", "r") as shapefile:
        for feature in shapefile:
            shapes = [feature["geometry"]]
```

```
    with rasterio.open(path + "/S2_Bands_TIFF/Composite.tif") as src:
        out_image, out_trasform = rasterio.mask.mask(src, shapes, crop="True")
        out_meta = src.meta

    out_meta.update({
        'driver': 'Gtiff',
        'height': out_image.shape[1],
        'width': out_image.shape[2],
        'transform': out_trasform
    })

    with rasterio.open(path + "/Test_Region/Composite_Clipped.tif", "w",
**out_meta) as dst:
        dst.write(out_image)
    array = rasterio.open(path + "Test_Region/Composite_Clipped.tif").read()
    shape_list = list(array.shape)
    array = array.reshape(shape_list[0], shape_list[1] * shape_list[2])
    array = array.transpose()
    df = pd.DataFrame(array,columns=['band2', 'band3', 'band4', 'band5', 'band6',
'band7', 'band8a', 'band11', 'band12','ndsi', 'ndvi', 'ndwi'])
    df.to_csv(path + "Test_Region/Test_Data_original.csv", index=False)

#
# get_test_data(path)


def training_polygons(path, choice="Cloud"):
    with fiona.open(path+"/Training_Polygons/" + choice + ".shp") as input:
        # preserve the schema of the original shapefile, including the crs
```

```
        meta = input.meta
        with fiona.open(path+ "/Training_Polygons/" + choice + "_dissolve.shp" ,'w',
**meta) as output:
            # groupby clusters consecutive elements of an iterable which have the same
key so you must first sort the features by the 'STATEFP' field
            e = sorted(input, key=lambda k: k['properties']['Id'])
            # group by the 'STATEFP' field
            for key, group in itertools.groupby(e, key=lambda x: x['properties']['Id']):
                properties, geom = zip(*[(feature['properties'],
shape(feature['geometry'])) for feature in group])
                # write the feature, computing the unary_union of the elements in the
group with the properties of the first element in the group
                output.write({'geometry': mapping(unary_union(geom)), 'properties':
properties[0]})

    with fiona.open(path+ "/Training_Polygons/" + choice + "_dissolve.shp", "r") as
shapefile:
        for feature in shapefile:
            shapes = [feature["geometry"]]

    with rasterio.open(path + "/S2_Bands_TIFF/Composite.tif") as src:
        out_image, out_trasform = rasterio.mask.mask(src, shapes, crop="True")
        out_meta = src.meta

    out_meta.update({
        'driver': 'Gtiff',
        'height': out_image.shape[1],
        'width': out_image.shape[2],
        'transform': out_trasform
    })
```

```
    with rasterio.open(path +"Training_Polygons/python/Composite_Polygons/" +
choice + ".tif", "w", **out_meta) as dst:
        dst.write(out_image)

# training_polygons(path, choice="Water")

arr = rasterio.open(path +
"/Training_Polygons/python/Composite_Polygons/Water.tif").read()
# print(arr)
nodata=arr[0][0][0] ###For obtaining nodata.
print(nodata)

def training_polygon_to_csv(path, choice="Cloud", choice2=1):
    array= rasterio.open(path + "/Training_Polygons/python/Composite_Polygons/"
+ choice + ".tif").read()
    shape_list = list(array.shape)
    array = array.reshape(shape_list[0], shape_list[1]*shape_list[2])
    array = array.transpose()
    df = pd.DataFrame(array, columns = ['band2','band3','band4', 'band5', 'band6',
'band7', 'band8a', 'band11', 'band12','ndsi', 'ndvi', 'ndwi'])
    df.drop(df.loc[df['band2'] == nodata].index, inplace=True)
    df.insert(0, "class", choice2)
    df.to_csv(path + "/Training_Polygons/python/Composite_Polygons/" + choice +
".csv", index=False)
#
# training_polygon_to_csv(path, choice="Water", choice2=4)

def sample_csv(path, choice="Cloud", sample_num=1000):
    total_training = pd.read_csv(path +
```

```
"/Training_Polygons/python/Composite_Polygons/" + choice + ".csv")

    total_training= total_training.sample(sample_num)
    total_training.to_csv(path + "/Training_Polygons/Original/" + choice + "_" +
str(sample_num) + ".csv", index=False)

# sample_csv(path, choice="Water", sample_num=1000)

def concat_csvs(path, sample_num=1000):
    files = os.path.join(path + "/Training_Polygons/Original/" , "*" +
str(sample_num) + ".csv")

    # list of merged files returned
    files = glob.glob(files)


    df = pd.concat(map(pd.read_csv, files), ignore_index=True)
    df.to_csv(path + "/Training_Polygons/Original/Training_Data" + "_" +
str(sample_num) + ".csv", index=False)
# #
# concat_csvs(path, sample_num=1000)
```

B. Python Script for Obtaining Normalized Band Indices

```
import rasterio
import os
import numpy as np
import pandas as pd

# NDWI (Sentinel 2) = (B3 – B8) / (B3 + B8)
# NDVI (Sentinel 2) = (B8 – B4) / (B8 + B4)
# NDSI (Sentinel 2) = (B3 – B11) / (B3 + B11)

'''THIS CODE IS FOR OBTAINING NDSI, NDVI AND NDWI FROM
SENTINEL-2 BANDS THAT ARE CONVERTED FROM .JP2 FORMAT TO
.TIFF BEFOREHAND'''

path =
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
23_Jan_2020_sc_only/"

b_3 = rasterio.open(path + 'S2_Bands_TIFF/B03.tif').read()
b_8 = rasterio.open(path +"S2_Bands_TIFF/B08A.tif").read()
b_4 = rasterio.open(path +'S2_Bands_TIFF/B04.tif').read()
b_11 = rasterio.open(path + 'S2_Bands_TIFF/B11.tif').read()
b_12= rasterio.open(path + 'S2_Bands_TIFF/B12.tif').read()

b_3_r = rasterio.open(path + 'S2_Bands_TIFF/B03.tif')
b_4_r = rasterio.open(path + 'S2_Bands_TIFF/B04.tif')


outpath = path + 'S2_Bands_TIFF/'
```

```
b_3= b_3.astype(float)
b_8= b_8.astype(float)
b_11= b_11.astype(float)
b_4= b_4.astype(float)
b_12 = b_12.astype(float)
b_3[b_3==0] = 0.0000001
b_8[b_8==0] = 0.0000001
b_4[b_4==0] = 0.0000001
b_11[b_11==0] = 0.0000001


ndvi = np.zeros(b_3_r.shape, dtype=rasterio.float32)
ndvi = (b_8.astype(float)-b_4.astype(float))/(b_8+b_4)

print(ndvi.shape)
ndvi=ndvi.reshape(5490,5490)

kwargs = b_4_r.meta
kwargs.update(
    dtype=rasterio.float32,
    count=1,
    compress='lzw')

with rasterio.open(os.path.join(outpath, 'NDVI.tif'), 'w', **kwargs) as dst:
    dst.write_band(1, ndvi.astype(rasterio.float32))

ndwi = np.zeros(b_3_r.shape, dtype=rasterio.float32)
ndwi = (b_3.astype(float)-b_8.astype(float))/(b_3+b_8)

print(ndwi.shape)
```

```
ndwi=ndwi.reshape(5490,5490)

kwargs = b_3_r.meta
kwargs.update(
    dtype=rasterio.float32,
    count=1,
    compress='lzw')

with rasterio.open(os.path.join(outpath, 'NDWI.tif'), 'w', **kwargs) as dst:
    dst.write_band(1, ndwi.astype(rasterio.float32))

ndsi = np.zeros(b_3_r.shape, dtype=rasterio.float32)
ndsi = (b_3.astype(float)-b_11.astype(float))/(b_3+b_11)
print(ndsi.shape)
ndsi=ndsi.reshape(5490,5490)

kwargs = b_3_r.meta
kwargs.update(
    dtype=rasterio.float32,
    count=1,
    compress='lzw')

with rasterio.open(os.path.join(outpath, 'NDSI.tif'), 'w', **kwargs) as dst:
    dst.write_band(1, ndsi.astype(rasterio.float32))
```

C. Python Script for Obtaining PCA Bands

```
from sklearn.decomposition import PCA
import skimage
import os
import rasterio
import numpy as np
import time

''' COMPOSITE RASTER SHOULD BE OBTAINED BEFOREHAND'''

start = time.time()

path_composite =
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
8_Apr_2019_atmo_topo/S2_Bands_PCA/Comp_PCA.tif"
save_path =
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
8_Apr_2019_atmo_topo/S2_Bands_PCA/"

meta_data = rasterio.open(path_composite)
comp_array = rasterio.open(path_composite).read()
# print(comp_array.shape) # 6, 5490, 5490
comp=np.reshape(comp_array,(6,5490*5490))

comp_final = np.transpose(comp)
# comp_final[np.isnan(comp_final)] = 32767
pca = PCA(n_components = 3)

a=pca.fit_transform(comp_final)
```

```
pca1_np = a[:,0]
pca1 = np.reshape(pca1_np, (5490,5490))
kwargs = meta_data.meta

with rasterio.open(os.path.join(save_path, 'PCA1_runtime.tif'), 'w', **kwargs) as
dst:
    dst.write_band(1, pca1)

pca2_np = a[:,1]
pca2 = np.reshape(pca2_np, (5490,5490))

with rasterio.open(os.path.join(save_path, 'PCA2_runtime.tif'), 'w', **kwargs) as
dst:
    dst.write_band(1, pca2)

pca3_np = a[:,2]
pca3 = np.reshape(pca3_np, (5490,5490))

with rasterio.open(os.path.join(save_path, 'PCA3_runtime.tif'), 'w', **kwargs) as
dst:
    dst.write_band(1, pca3)

end = time.time()
print((end - start)/60)
```

D. Python Script for RF Classification

```
import rasterio
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
import time

'''THIS CODE IS FOR ATMO_TOPO AND SC_ONLY INPUT
COMBINATIONS ONLY'''
start = time.time()

img_path=
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
8_Apr_2019_sc_only/Test_Area/"
train_path=
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
8_Apr_2019_sc_only/Training_Polygons/Original/Training_Data_1000.csv"
save_path=
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
8_Apr_2019_sc_only/220610_Classified_01/"
# img = rasterio.open(img_path).read()
# print(img.shape)
def random_forest_og(img_path, train_path, save_path, sample_size = 300,
choice="atmo_topo"):
    img = rasterio.open(img_path)
    img_as_array = img.read()
    input_crs = img.crs
    input_gt = img.transform
```

```
    df_train = pd.read_csv(train_path)

    # Initialize our model with 500 trees
    X = df_train.drop('class', axis = 1)
    y = df_train['class']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
    # Fit our model to training data
    rf = RandomForestClassifier(n_estimators=500, oob_score=True)
    rf_model = rf.fit(X_train, y_train)
    arr1 = img_as_array.reshape(12,1001*1001).transpose()
    class_prediction = rf_model.predict(arr1)
    class_prediction = class_prediction.reshape(1001,1001)
    class_prediction = class_prediction.astype(np.float32)
    with rasterio.open(
        save_path + '/RF_classification_'+ choice +'_'+ str(sample_size) +'runtime.tif',
        'w',
        driver='GTiff',
        height=img.shape[0],
        width=img.shape[1],
        count=1,
        dtype=np.float32,
        crs=input_crs,
        transform=input_gt,
    ) as dest_file:
        dest_file.write(class_prediction, 1)
    dest_file.close()
random_forest_og(img_path, train_path, save_path,sample_size=1000,
choice='atmo_topo')
end = time.time()
print((end - start)/60)
```

```
# print(save_path + '/RF_classification_'+ "atmo_topo" +'_'+ str(1000) +'new.tif')
```

E. Python Script for Checking Atmospheric and Topographic Correction

```
import rasterio
import fiona
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import rasterio.mask
import glob
import os

'''A POLYGON REPRESENTING PURE SNOW COVER SHOULD BE
OBTAINED BEFOREHAND'''

path=
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
23_Jan_2020_atmo_topo"
path_sc=
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
23_Jan_2020_sc_only"



def pure_snow_polygon(path, path_sc, band_choice = "B02"):
   #THIS CODE IS FOR ATMOSPHERICALLY AND TOPOGRAPHICALLY
CORRECTED IMAGES
   with fiona.open(path+"/Atmo_Topo_Check/puresnow.shp", "r") as shapefile:
      for feature in shapefile:
         shapes = [feature["geometry"]]
```

```
    with rasterio.open(path + "/S2_Bands_TIFF/"+ band_choice + ".tif") as src:
        out_image, out_trasform = rasterio.mask.mask(src, shapes, crop="True")
        out_meta = src.meta

    out_meta.update({
        'driver': 'Gtiff',
        'height': out_image.shape[1],
        'width': out_image.shape[2],
        'transform': out_trasform
    })
    with rasterio.open(path +"/Atmo_Topo_Check/" + band_choice +
"_puresnow.tif", "w", **out_meta) as dst:
        dst.write(out_image)
    #THIS CODE IS FOR ORIGINAL/RESAMPLED ONLY IMAGES


    with fiona.open(path+"/Atmo_Topo_Check/puresnow.shp", "r") as shapefile:
        for feature in shapefile:
            shapes = [feature["geometry"]]

    with rasterio.open(path_sc + "/S2_Bands_TIFF/"+ band_choice + ".tif") as src:
        out_image, out_trasform = rasterio.mask.mask(src, shapes, crop="True")
        out_meta = src.meta

    out_meta.update({
        'driver': 'Gtiff',
        'height': out_image.shape[1],
        'width': out_image.shape[2],
        'transform': out_trasform
    })
```

```
    with rasterio.open(path +"/Atmo_Topo_Check/" + band_choice +
"_puresnow_sconly.tif", "w", **out_meta) as dst:
        dst.write(out_image)

def plot_mean(path):
    b_2 = rasterio.open(path + "/Atmo_Topo_Check/" +
'B02_puresnow.tif').read().flatten()
    b_3 = rasterio.open(path + "/Atmo_Topo_Check/" +
'B03_puresnow.tif').read().flatten()
    b_4 = rasterio.open(path + "/Atmo_Topo_Check/" +
'B04_puresnow.tif').read().flatten()

    b2 = rasterio.open(path + "/Atmo_Topo_Check/" +
'B02_puresnow_sconly.tif').read().flatten()
    b3 = rasterio.open(path + "/Atmo_Topo_Check/" +
'B03_puresnow_sconly.tif').read().flatten()
    b4 = rasterio.open(path + "/Atmo_Topo_Check/" +
'B04_puresnow_sconly.tif').read().flatten()

    #nodata= -32768

    b_2 = b_2[b_2!=-32768]
    b_3 = b_3[b_3!=-32768]
    b_4 = b_4[b_4!=-32768]
    b2 = b2[b2!=-32768]
    b3 = b3[b3!=-32768]
    b4 = b4[b4!=-32768]

    mean_b_2= np.mean(b_2)/10000
    mean_b_3= np.mean(b_3)/10000
```

```
    mean_b_4= np.mean(b_4)/10000

    mean_b2= np.mean(b2)/10000
    mean_b3= np.mean(b3)/10000
    mean_b4= np.mean(b4)/10000


    x = ["band2(blue)","band3(green)","band4(red)"]
    y1=[mean_b_2,mean_b_3,mean_b_4]
    y2=[mean_b2,mean_b3,mean_b4]

    sns.lineplot(x=x, y=y1, legend="brief", label= "Atmospheric & Topographic
Correction",color="red")
    sns.lineplot(x=x, y=y2, legend="brief", label="Original", color="blue")
    sns.set(rc = {'figure.figsize':(10,20)})
    plt.legend(loc="center")
    plt.yticks(np.arange(min(y2)+0.0003, max(y1), 0.02))
    plt.suptitle('TATRA - 23 January 2020') #change it as you go
    plt.xlabel('Sentinel-2 Bands')
    plt.ylabel('Reflectance Values')
    # function to show the plot
    # plt.show()

plt.savefig("E:/TEZ/Presentations/070722/atmo_topo/Tatra_23_Jan_2020_new.png
")

# band_names = ['B02', 'B03', 'B04']
#
# for name in band_names:
#     pure_snow_polygon(path, path_sc, band_choice=name)
```

```
plot_mean(path)
```

F. Python Script for Obtaining Accuracy Assessment Metrics

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.metrics import cohen_kappa_score



path =
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/02_TATRA/Work_Folder/
8_Apr_2019_atmo_topo"

def colors_from_values(values, palette_name):
    # normalize the values to range [0, 1]
    normalized = (values - min(values)) / (max(values) - min(values))
    # convert to indices
    indices = np.round(normalized * (len(values) - 1)).astype(np.int32)
    # use the indices to get the colors
    palette = sns.color_palette(palette_name, len(values))
    return np.array(palette).take(indices, axis=0)

def conf_matrix_batch(path=path):
    path_read = path + "/Accuracy_Assessment/ALL/"
    # df_atmo_topo_1000 = pd.read_csv(path_read +
"classified_atmo_topo_1000.csv")
    # df_atmo_topo_300=pd.read_csv(path_read + "classified_atmo_topo_300.csv")
    # df_sc_1000=pd.read_csv(path_read + "classified_sc_1000.csv")
```

```
    # df_sc_300 = pd.read_csv(path_read + "classified_sc_300.csv")
    # df_dem_1000 = pd.read_csv(path_read + "classified_dem_1000.csv")
    # df_dem_300 = pd.read_csv(path_read + "classified_dem_300.csv")
    # df_pca_1000 = pd.read_csv(path_read + "classified_pca_1000.csv")
    # df_pca_300 = pd.read_csv(path_read + "classified_pca_300.csv")
    df_pca_plus_1000 = pd.read_csv(path_read + "classified_pca_plus_1000.csv")
    df_pca_plus_300 = pd.read_csv(path_read + "classified_pca_plus_300.csv")
    df_pca_plus2_1000 = pd.read_csv(path_read +
"classified_pca_plus_10002.csv")
    df_pca_plus2_300 = pd.read_csv(path_read + "classified_pca_plus_3002.csv")

    df_gt = pd.read_csv(path_read + "groundtruth.csv")

    # cf_matrix_atmo_topo_1000 = confusion_matrix(df_gt, df_atmo_topo_1000)
    # cf_matrix_atmo_topo_300 = confusion_matrix(df_gt, df_atmo_topo_300)
    # cf_matrix_sc_1000 = confusion_matrix(df_gt, df_sc_1000)
    # cf_matrix_sc_300 = confusion_matrix(df_gt, df_sc_300)
    # cf_matrix_dem_1000 = confusion_matrix(df_gt, df_dem_1000)
    # cf_matrix_dem_300 = confusion_matrix(df_gt, df_dem_300)
    # cf_matrix_pca_1000 = confusion_matrix(df_gt, df_pca_1000)
    # cf_matrix_pca_300 = confusion_matrix(df_gt, df_pca_300)
    cf_matrix_pca_plus_1000 = confusion_matrix(df_gt, df_pca_plus_1000)
    cf_matrix_pca_plus_300 = confusion_matrix(df_gt, df_pca_plus_300)
    cf_matrix_pca_plus2_1000 = confusion_matrix(df_gt, df_pca_plus2_1000)
    cf_matrix_pca_plus2_300 = confusion_matrix(df_gt, df_pca_plus2_300)

    # cm_df_atmo_topo_1000 = pd.DataFrame(cf_matrix_atmo_topo_1000,
    #                                     index=["cloud", "land", "snow", "water"],
    #                                     columns=["cloud", "land", "snow", "water"])
    #
```

```
# cm_df_atmo_topo_300 = pd.DataFrame(cf_matrix_atmo_topo_300,
#                                    index=["cloud", "land", "snow", "water"],
#                                    columns=["cloud", "land", "snow", "water"])
#
# cm_df_sc_1000 = pd.DataFrame(cf_matrix_sc_1000,
#                                    index=["cloud", "land", "snow", "water"],
#                                    columns=["cloud", "land", "snow", "water"])
#
# cm_df_sc_300= pd.DataFrame(cf_matrix_sc_300,
#                                    index=["cloud", "land", "snow", "water"],
#                                    columns=["cloud", "land", "snow", "water"])
# cm_df_dem_1000 = pd.DataFrame(cf_matrix_dem_1000,
#                                    index=["cloud", "land", "snow", "water"],
#                                    columns=["cloud", "land", "snow", "water"])
# cm_df_dem_300 = pd.DataFrame(cf_matrix_dem_300,
#                                    index=["cloud", "land", "snow", "water"],
#                                    columns=["cloud", "land", "snow", "water"])
#
# cm_df_pca_1000 = pd.DataFrame(cf_matrix_pca_1000,
#                                    index=["cloud", "land", "snow", "water"],
#                                    columns=["cloud", "land", "snow", "water"])
# cm_df_pca_300 = pd.DataFrame(cf_matrix_pca_300,
#                                    index=["cloud", "land", "snow", "water"],
#                                    columns=["cloud", "land", "snow", "water"])

cm_df_pca_plus_1000 = pd.DataFrame(cf_matrix_pca_plus_1000,
                                   index=["cloud", "land", "snow", "water"],
                                   columns=["cloud", "land", "snow", "water"])
cm_df_pca_plus_300 = pd.DataFrame(cf_matrix_pca_plus_300,
                                  index=["cloud", "land", "snow", "water"],
```

```
                        columns=["cloud", "land", "snow", "water"])
    cm_df_pca_plus2_1000 = pd.DataFrame(cf_matrix_pca_plus2_1000,
                         index=["cloud", "land", "snow", "water"],
                         columns=["cloud", "land", "snow", "water"])
    cm_df_pca_plus2_300 = pd.DataFrame(cf_matrix_pca_plus2_300,
                        index=["cloud", "land", "snow", "water"],
                        columns=["cloud", "land", "snow", "water"])

    fig, axes = plt.subplots(nrows=2, ncols=2, figsize= (12,12))

    # sns.heatmap(cm_df_atmo_topo_1000, annot=True, fmt='g',
ax=axes[0,0]).set(title="Atmo_Topo_1000")
    # sns.heatmap(cm_df_sc_1000, annot=True, fmt='g',
ax=axes[0,1]).set(title="SC_1000")
    # sns.heatmap(cm_df_dem_1000, annot=True, fmt='g',
ax=axes[0,2]).set(title="DEM_1000")
    # sns.heatmap(cm_df_pca_1000, annot=True, fmt='g',
ax=axes[0,3]).set(title="PCA_1000")
    sns.heatmap(cm_df_pca_plus_1000, annot=True, fmt='g', ax=axes[0,
0]).set(title="PCA_PLUS_1000")
    sns.heatmap(cm_df_pca_plus2_1000, annot=True, fmt='g', ax=axes[0,
1]).set(title="PCA_PLUS2_1000")

    # sns.heatmap(cm_df_atmo_topo_300, annot=True, fmt='g',
ax=axes[1,0]).set(title="Atmo_Topo_300")
    # sns.heatmap(cm_df_sc_300, annot=True, fmt='g',
ax=axes[1,1]).set(title="SC_300")
    # sns.heatmap(cm_df_dem_300, annot=True, fmt='g',
ax=axes[1,2]).set(title="DEM_300")
    # sns.heatmap(cm_df_pca_300, annot=True, fmt='g',
```

```
ax=axes[1,3]).set(title="PCA_300")
    sns.heatmap(cm_df_pca_plus_300, annot=True, fmt='g', ax=axes[1,
0]).set(title="PCA_PLUS_300")
    sns.heatmap(cm_df_pca_plus2_300, annot=True, fmt='g', ax=axes[1,
1]).set(title="PCA_PLUS2_300")

    fig.suptitle("TATRA - 8 April 2019")
    fig.tight_layout()
    plt.show()



def acc_kappa(path):
    path_read = path + "/Accuracy_Assessment/ALL/"
    # df_atmo_topo_1000 = pd.read_csv(path_read +
"classified_atmo_topo_1000.csv")
    # df_atmo_topo_300 = pd.read_csv(path_read +
"classified_atmo_topo_300.csv")
    # df_sc_1000 = pd.read_csv(path_read + "classified_sc_1000.csv")
    # df_sc_300 = pd.read_csv(path_read + "classified_sc_300.csv")
    # df_dem_1000 = pd.read_csv(path_read + "classified_dem_1000.csv")
    # df_dem_300 = pd.read_csv(path_read + "classified_dem_300.csv")
    # df_pca_1000 = pd.read_csv(path_read + "classified_pca_1000.csv")
    # df_pca_300 = pd.read_csv(path_read + "classified_pca_300.csv")
    df_pca_plus_1000 = pd.read_csv(path_read + "classified_pca_plus_1000.csv")
    df_pca_plus_300 = pd.read_csv(path_read + "classified_pca_plus_300.csv")
    df_pca_plus2_1000 = pd.read_csv(path_read +
"classified_pca_plus_10002.csv")
    df_pca_plus2_300 = pd.read_csv(path_read + "classified_pca_plus_3002.csv")
    df_gt = pd.read_csv(path_read + "groundtruth.csv")
```

```
# overall_acc_atmo_topo_1000 = accuracy_score(df_gt, df_atmo_topo_1000)
# kappa_atmo_topo_1000 = cohen_kappa_score(df_gt, df_atmo_topo_1000)
#
# overall_acc_atmo_topo_300 = accuracy_score(df_gt, df_atmo_topo_300)
# kappa_atmo_topo_300 = cohen_kappa_score(df_gt, df_atmo_topo_300)
#
# overall_acc_sc_1000 = accuracy_score(df_gt, df_sc_1000)
# kappa_sc_1000 = cohen_kappa_score(df_gt, df_sc_1000)
#
# overall_acc_sc_300 = accuracy_score(df_gt, df_sc_300)
# kappa_sc_300 = cohen_kappa_score(df_gt, df_sc_300)
#
# overall_acc_dem_1000 = accuracy_score(df_gt, df_dem_1000)
# kappa_dem_1000 = cohen_kappa_score(df_gt, df_dem_1000)
#
# overall_acc_dem_300 = accuracy_score(df_gt, df_dem_300)
# kappa_dem_300 = cohen_kappa_score(df_gt, df_dem_300)
#
# overall_acc_pca_1000 = accuracy_score(df_gt, df_pca_1000)
# kappa_pca_1000 = cohen_kappa_score(df_gt, df_pca_1000)
#
# overall_acc_pca_300 = accuracy_score(df_gt, df_pca_300)
# kappa_pca_300 = cohen_kappa_score(df_gt, df_pca_300)

overall_acc_pca_plus_1000 = accuracy_score(df_gt, df_pca_plus_1000)
kappa_pca_plus_1000 = cohen_kappa_score(df_gt, df_pca_plus_1000)

overall_acc_pca_plus_300 = accuracy_score(df_gt, df_pca_plus_300)
kappa_pca_plus_300 = cohen_kappa_score(df_gt, df_pca_plus_300)
```

```
    overall_acc_pca_plus2_1000 = accuracy_score(df_gt, df_pca_plus2_1000)
    kappa_pca_plus2_1000 = cohen_kappa_score(df_gt, df_pca_plus2_1000)

    overall_acc_pca_plus2_300 = accuracy_score(df_gt, df_pca_plus2_300)
    kappa_pca_plus2_300 = cohen_kappa_score(df_gt, df_pca_plus2_300)

    # Numbers of pairs of bars you want
    N = 4
    # Y values
    list_overall_acc = [overall_acc_pca_plus_1000, overall_acc_pca_plus_300,
overall_acc_pca_plus2_1000, overall_acc_pca_plus2_300]
    list_kappa = [kappa_pca_plus_1000, kappa_pca_plus_300,
kappa_pca_plus2_1000, kappa_pca_plus2_300]
    # list_overall_acc = [overall_acc_atmo_topo_1000, overall_acc_atmo_topo_300,
overall_acc_sc_1000, overall_acc_sc_300,
    #                     overall_acc_dem_1000, overall_acc_dem_300,
overall_acc_pca_1000, overall_acc_pca_300]
    # list_kappa = [kappa_atmo_topo_1000, kappa_atmo_topo_300,
kappa_sc_1000, kappa_sc_300, kappa_dem_1000, kappa_dem_300,
    #               kappa_pca_1000, kappa_pca_300]

    # X values

    # x_values = ['Atmo_Topo_1000', 'Atmo_Topo_300', 'SC_1000', 'SC_300',
'DEM_1000', 'DEM_300', 'PCA_1000', 'PCA_300']
    x_values = ['PCA_PLUS_1000', 'PCA_PLUS_300', 'PCA_PLUS2_1000',
'PCA_PLUS2_300']

    # Position of bars on x-axis
```

```
    ind = np.arange(N)
    plt.figure(figsize=(20, 10))
    width = 0.22

    plt.bar(ind, list_overall_acc, width, label='Overall Accuracy', color="#FE5E03")
    plt.bar(ind + width, list_kappa, width, label='Kappa Coefficient',
color="#03FEE1")

    plt.xlabel('Methodology')
    plt.ylabel('Overall Accuracy and Kappa Coefficient Values')
    plt.title("TATRA - 8 April 2019")
    plt.xticks(ind + width / 2, x_values)
    plt.ylim(min(list_kappa) - 0.1, max(list_overall_acc) + 0.1)
    # plt.yticks(np.arange(min(list_kappa)+0.1,max(list_overall_acc), 0.01))

    # Finding the best position for legends and putting it
    plt.legend(loc='best')
    plt.show()


def to_csv(path):
    path_read = path + "/Accuracy_Assessment/ALL/"
    df_atmo_topo_1000 = pd.read_csv(path_read +
"classified_atmo_topo_1000.csv")
    df_atmo_topo_300 = pd.read_csv(path_read + "classified_atmo_topo_300.csv")
    df_sc_1000 = pd.read_csv(path_read + "classified_sc_1000.csv")
    df_sc_300 = pd.read_csv(path_read + "classified_sc_300.csv")
    df_dem_1000 = pd.read_csv(path_read + "classified_dem_1000.csv")
    df_dem_300 = pd.read_csv(path_read + "classified_dem_300.csv")
    df_pca_1000 = pd.read_csv(path_read + "classified_pca_1000.csv")
```

```
    df_pca_300 = pd.read_csv(path_read + "classified_pca_300.csv")
    # df_pca_plus_1000 = pd.read_csv(path_read + "classified_pca_plus_1000.csv")
    # df_pca_plus_300 = pd.read_csv(path_read + "classified_pca_plus_300.csv")
    # df_pca_plus2_1000 = pd.read_csv(path_read +
"classified_pca_plus_10002.csv")
    # df_pca_plus2_300 = pd.read_csv(path_read + "classified_pca_plus_3002.csv")

    df_gt = pd.read_csv(path_read + "groundtruth.csv")

    overall_acc_atmo_topo_1000 = accuracy_score(df_gt, df_atmo_topo_1000)
    kappa_atmo_topo_1000 = cohen_kappa_score(df_gt, df_atmo_topo_1000)

    overall_acc_atmo_topo_300 = accuracy_score(df_gt, df_atmo_topo_300)
    kappa_atmo_topo_300 = cohen_kappa_score(df_gt, df_atmo_topo_300)

    overall_acc_sc_1000 = accuracy_score(df_gt, df_sc_1000)
    kappa_sc_1000 = cohen_kappa_score(df_gt, df_sc_1000)

    overall_acc_sc_300 = accuracy_score(df_gt, df_sc_300)
    kappa_sc_300 = cohen_kappa_score(df_gt, df_sc_300)

    overall_acc_dem_1000 = accuracy_score(df_gt, df_dem_1000)
    kappa_dem_1000 = cohen_kappa_score(df_gt, df_dem_1000)

    overall_acc_dem_300 = accuracy_score(df_gt, df_dem_300)
    kappa_dem_300 = cohen_kappa_score(df_gt, df_dem_300)

    overall_acc_pca_1000 = accuracy_score(df_gt, df_pca_1000)
    kappa_pca_1000 = cohen_kappa_score(df_gt, df_pca_1000)
```

```
    overall_acc_pca_300 = accuracy_score(df_gt, df_pca_300)
    kappa_pca_300 = cohen_kappa_score(df_gt, df_pca_300)

    # overall_acc_pca_plus_1000 = accuracy_score(df_gt, df_pca_plus_1000)
    # kappa_pca_plus_1000 = cohen_kappa_score(df_gt, df_pca_plus_1000)
    #
    # overall_acc_pca_plus_300 = accuracy_score(df_gt, df_pca_plus_300)
    # kappa_pca_plus_300 = cohen_kappa_score(df_gt, df_pca_plus_300)
    #
    # overall_acc_pca_plus2_1000 = accuracy_score(df_gt, df_pca_plus2_1000)
    # kappa_pca_plus2_1000 = cohen_kappa_score(df_gt, df_pca_plus2_1000)
    #
    # overall_acc_pca_plus2_300 = accuracy_score(df_gt, df_pca_plus2_300)
    # kappa_pca_plus2_300 = cohen_kappa_score(df_gt, df_pca_plus2_300)

    dict_ult = {'Name': ['Overall Accuracy', 'Kappa'], 'Atmo_Topo_1000':
[overall_acc_atmo_topo_1000, kappa_atmo_topo_1000],'Atmo_Topo_300':
[overall_acc_atmo_topo_300, kappa_atmo_topo_300], 'SC_1000':
[overall_acc_sc_1000, kappa_sc_1000], 'SC_300':[overall_acc_sc_300,
kappa_sc_300], 'DEM_1000': [overall_acc_dem_1000,kappa_dem_1000],
'DEM_300': [overall_acc_dem_300, kappa_dem_300], 'PCA_1000':
[overall_acc_pca_1000, kappa_pca_1000], 'PCA_300':
[overall_acc_pca_300,kappa_pca_300] }

    df = pd.DataFrame(dict_ult)
    df.to_csv("E:/TEZ/Presentations/140722/Alps_24_Jan_2019_jun.csv")

conf_matrix_batch(path)
# acc_kappa(path)
# to_csv(path)
```

G. Python Script for Obtaining Monthly Average Values from ERA5-Land Monthly Reanalysis Data

```
from shapely.geometry import mapping
import xarray as xr
import rioxarray
import geopandas as gpd
import numpy as np
import pandas as pd
import statistics as st


shp_path_alps='D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/04_Sentinel_2_Tiles_Borders/T32TPS_Alps.shp'
shp_path_tatra='D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/04_Sentinel_2_Tiles_Borders/T34UCV_Tatra.shp'
shp_path_turkey='D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/04_Sentinel_2_Tiles_Borders/T37TFE_Turkey.shp'
list_range = list(range(776,865)) #covering from January 1990 to December 2021


#lists for individual months - needed to get the mean of the months for 31 years for each region
list_january = list (range(480,865,12))
list_february = list (range(480+1,865,12))
list_march = list (range(480+2,865,12))
list_april = list (range(480+3,865,12))
list_may = list (range(480+4,865,12))
list_june = list (range(480+5,865,12))
list_july = list (range(480+6,865,12))
```

```
list_august = list(range(480+7,865,12))
list_september = list (range(480+8,865,12))
list_october = list (range(480+9,865,12))
list_november = list (range(480+10,865,12))
list_december = list (range(480+11,865,12))



#region= "alps", "tatra", "turkey"
def get_raster_sde(i):
# Open the NetCDF

    sde_full = xr.open_dataset('E:/indirilenler/sde_full.nc')
    sde= sde_full['sde']
    sde.rio.write_crs("epsg:4326", inplace=True)
#(Optional) convert longitude from (0-360) to (-180 to 180) (if required)
    sde.coords['longitude'] = (sde.coords['longitude'] + 180) % 360 - 180
    sde = sde.sortby(sde.longitude)

#Define lat/long
    sde = sde.rio.set_spatial_dims('longitude', 'latitude')
    sde[i].rio.to_raster("E:/indirilenler/sde/sde_" + str(i) + ".tiff", compress='LZMA',
tiled=True, dtype="int32")



def clip_and_get_mean_sde(shp_path, i):
    sde_alps_try = rioxarray.open_rasterio("E:/indirilenler/sde/sde_"+str(i)+".tiff",
                                masked=True).squeeze()
```

```
    shapes_alps = gpd.read_file(shp_path)
    sde_clipped = sde_alps_try.rio.clip(shapes_alps.geometry.apply(mapping),
shapes_alps.crs)
    sde_array = sde_clipped.to_masked_array().flatten().compressed()
    sde_array = list(sde_array)
    # return sde_array
    return np.mean(sde_array)

# '''GET RASTERS FIRST'''
# for i in list_range:
#     get_raster_sde(i)
#     print("done:" + str(i))

'''MEAN OF MONTHS FOR ALPINE REGION'''

# mean_list_sde_alps_january=[]
# for i in list_january:
#     mean_list_sde_alps_january.append(clip_and_get_mean_sde(shp_path_alps,
i))
#     print("done:" + str(i))
#
# mean_list_sde_alps_february=[]
# for i in list_february:
#     mean_list_sde_alps_february.append(clip_and_get_mean_sde(shp_path_alps,
i))
#     print("done:" + str(i))
#
# mean_list_sde_alps_march=[]
# for i in list_march:
#     mean_list_sde_alps_march.append(clip_and_get_mean_sde(shp_path_alps, i))
```

```
#     print("done:" + str(i))
#
# mean_list_sde_alps_april=[]
# for i in list_april:
#     mean_list_sde_alps_april.append(clip_and_get_mean_sde(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_sde_alps_may=[]
# for i in list_may:
#     mean_list_sde_alps_may.append(clip_and_get_mean_sde(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_sde_alps_june=[]
# for i in list_june:
#     mean_list_sde_alps_june.append(clip_and_get_mean_sde(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_sde_alps_july=[]
# for i in list_july:
#     mean_list_sde_alps_july.append(clip_and_get_mean_sde(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_sde_alps_august=[]
# for i in list_august:
#     mean_list_sde_alps_august.append(clip_and_get_mean_sde(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_sde_alps_september=[]
# for i in list_september:
#
```

```
mean_list_sde_alps_september.append(clip_and_get_mean_sde(shp_path_alps, i))
#     print("done:" + str(i))
#
#
# mean_list_sde_alps_october=[]
# for i in list_october:
#     mean_list_sde_alps_october.append(clip_and_get_mean_sde(shp_path_alps,
i))
#     print("done:" + str(i))
#
#
# mean_list_sde_alps_november=[]
# for i in list_november:
#     mean_list_sde_alps_november.append(clip_and_get_mean_sde(shp_path_alps,
i))
#     print("done:" + str(i))
# #
# mean_list_sde_alps_december=[]
# for i in list_december:
#     mean_list_sde_alps_december.append(clip_and_get_mean_sde(shp_path_alps,
i))
#     print("done:" + str(i))
#
# alps_sde= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
'August', 'September', 'October', 'November', 'December'],
#         'Means': [st.mean(mean_list_sde_alps_january),
st.mean(mean_list_sde_alps_february), st.mean(mean_list_sde_alps_march),
st.mean(mean_list_sde_alps_april), st.mean(mean_list_sde_alps_may),
st.mean(mean_list_sde_alps_june), st.mean(mean_list_sde_alps_july),
st.mean(mean_list_sde_alps_august), st.mean(mean_list_sde_alps_september),
```

```
st.mean(mean_list_sde_alps_october), st.mean(mean_list_sde_alps_november),
st.mean(mean_list_sde_alps_december)]}
# alps_sde
=pd.DataFrame(alps_sde).to_csv("E:/indirilenler/sde/alps_sde_means.csv")

'''MEAN OF MONTHS FOR TATRA MOUNTAINS'''
#
# mean_list_sde_tatra_january=[]
# for i in list_january:
#     mean_list_sde_tatra_january.append(clip_and_get_mean_sde(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# mean_list_sde_tatra_february=[]
# for i in list_february:
#     mean_list_sde_tatra_february.append(clip_and_get_mean_sde(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# mean_list_sde_tatra_march=[]
# for i in list_march:
#     mean_list_sde_tatra_march.append(clip_and_get_mean_sde(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# mean_list_sde_tatra_april=[]
# for i in list_april:
#     mean_list_sde_tatra_april.append(clip_and_get_mean_sde(shp_path_tatra, i))
#     print("done:" + str(i))
#
```

```
# mean_list_sde_tatra_may=[]
# for i in list_may:
#     mean_list_sde_tatra_may.append(clip_and_get_mean_sde(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_sde_tatra_june=[]
# for i in list_june:
#     mean_list_sde_tatra_june.append(clip_and_get_mean_sde(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_sde_tatra_july=[]
# for i in list_july:
#     mean_list_sde_tatra_july.append(clip_and_get_mean_sde(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_sde_tatra_august=[]
# for i in list_august:
#     mean_list_sde_tatra_august.append(clip_and_get_mean_sde(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# mean_list_sde_tatra_september=[]
# for i in list_september:
#
mean_list_sde_tatra_september.append(clip_and_get_mean_sde(shp_path_tatra, i))
#     print("done:" + str(i))
#
#
# mean_list_sde_tatra_october=[]
# for i in list_october:
```

```
#     mean_list_sde_tatra_october.append(clip_and_get_mean_sde(shp_path_tatra,
i))
#     print("done:" + str(i))
#
#
# mean_list_sde_tatra_november=[]
# for i in list_november:
#
mean_list_sde_tatra_november.append(clip_and_get_mean_sde(shp_path_tatra, i))
#     print("done:" + str(i))
# #
# mean_list_sde_tatra_december=[]
# for i in list_december:
#
mean_list_sde_tatra_december.append(clip_and_get_mean_sde(shp_path_tatra, i))
#     print("done:" + str(i))
#
# tatra_sde= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
'August', 'September', 'October', 'November', 'December'],
#         'Means': [st.mean(mean_list_sde_tatra_january),
st.mean(mean_list_sde_tatra_february), st.mean(mean_list_sde_tatra_march),
st.mean(mean_list_sde_tatra_april), st.mean(mean_list_sde_tatra_may),
st.mean(mean_list_sde_tatra_june), st.mean(mean_list_sde_tatra_july),
st.mean(mean_list_sde_tatra_august), st.mean(mean_list_sde_tatra_september),
st.mean(mean_list_sde_tatra_october), st.mean(mean_list_sde_tatra_november),
st.mean(mean_list_sde_tatra_december)]}
# tatra_sde
=pd.DataFrame(tatra_sde).to_csv("E:/indirilenler/sde/tatra_sde_means.csv")
# print(mean_list_sde_tatra_january)
# print(mean_list_sde_tatra_february)
```

```
# print(mean_list_sde_tatra_march)
# print(mean_list_sde_tatra_april)
# print(mean_list_sde_tatra_may)
# print(mean_list_sde_tatra_june)
# print(mean_list_sde_tatra_july)
# print(mean_list_sde_tatra_august)
# print(mean_list_sde_tatra_september)
# print(mean_list_sde_tatra_october)
# print(mean_list_sde_tatra_november)
# print(mean_list_sde_tatra_december)
'''MEAN OF MONTHS FOR KAÇKAR MOUNTAINS'''

# mean_list_sde_kackar_january=[]
# for i in list_january:
#
mean_list_sde_kackar_january.append(clip_and_get_mean_sde(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_sde_kackar_february=[]
# for i in list_february:
#
mean_list_sde_kackar_february.append(clip_and_get_mean_sde(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_sde_kackar_march=[]
# for i in list_march:
#
mean_list_sde_kackar_march.append(clip_and_get_mean_sde(shp_path_turkey, i))
```

```
#     print("done:" + str(i))
#
# mean_list_sde_kackar_april=[]
# for i in list_april:
#     mean_list_sde_kackar_april.append(clip_and_get_mean_sde(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_sde_kackar_may=[]
# for i in list_may:
#     mean_list_sde_kackar_may.append(clip_and_get_mean_sde(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_sde_kackar_june=[]
# for i in list_june:
#     mean_list_sde_kackar_june.append(clip_and_get_mean_sde(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_sde_kackar_july=[]
# for i in list_july:
#     mean_list_sde_kackar_july.append(clip_and_get_mean_sde(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_sde_kackar_august=[]
# for i in list_august:
#
mean_list_sde_kackar_august.append(clip_and_get_mean_sde(shp_path_turkey,
```

```
i))
#     print("done:" + str(i))
#
# mean_list_sde_kackar_september=[]
# for i in list_september:
#
mean_list_sde_kackar_september.append(clip_and_get_mean_sde(shp_path_turke
y, i))
#     print("done:" + str(i))
#
#
# mean_list_sde_kackar_october=[]
# for i in list_october:
#
mean_list_sde_kackar_october.append(clip_and_get_mean_sde(shp_path_turkey,
i))
#     print("done:" + str(i))
#
#
# mean_list_sde_kackar_november=[]
# for i in list_november:
#
mean_list_sde_kackar_november.append(clip_and_get_mean_sde(shp_path_turkey
, i))
#     print("done:" + str(i))
# #
# mean_list_sde_kackar_december=[]
# for i in list_december:
#
mean_list_sde_kackar_december.append(clip_and_get_mean_sde(shp_path_turkey
```

```
, i))
#     print("done:" + str(i))
#
#kackar_sde= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
'August', 'September', 'October', 'November', 'December'],
#         'Means': [st.mean(mean_list_sde_kackar_january),
st.mean(mean_list_sde_kackar_february), st.mean(mean_list_sde_kackar_march),
st.mean(mean_list_sde_kackar_april), st.mean(mean_list_sde_kackar_may),
st.mean(mean_list_sde_kackar_june), st.mean(mean_list_sde_kackar_july),
st.mean(mean_list_sde_kackar_august),
st.mean(mean_list_sde_kackar_september),
st.mean(mean_list_sde_kackar_october),
st.mean(mean_list_sde_kackar_november),
st.mean(mean_list_sde_kackar_december)]}
# kackar_sde
=pd.DataFrame(kackar_sde).to_csv("E:/indirilenler/sde/kackar_sde_means.csv")

def get_raster_t2m(i):
#Open the NetCDF

    sde_full = xr.open_dataset('E:/indirilenler/t2m_full.nc')
    sde= sde_full['t2m']
    sde.rio.write_crs("epsg:4326", inplace=True)
#(Optional) convert longitude from (0-360) to (-180 to 180) (if required)
    sde.coords['longitude'] = (sde.coords['longitude'] + 180) % 360 - 180
    sde = sde.sortby(sde.longitude)

#Define lat/long
    sde = sde.rio.set_spatial_dims('longitude', 'latitude')
#Save individual rasters
```

```
    sde[i].rio.to_raster("E:/indirilenler/t2m/t2m_" + str(i) + ".tiff",
compress='LZMA', tiled=True, dtype="int32")

def clip_and_get_mean_t2m(shp_path, i):
    sde_alps_try = rioxarray.open_rasterio("E:/indirilenler/t2m/t2m_"+str(i)+".tiff",
                                masked=True).squeeze()
    shapes_alps = gpd.read_file(shp_path)
    sde_clipped = sde_alps_try.rio.clip(shapes_alps.geometry.apply(mapping),
shapes_alps.crs)
    sde_array = sde_clipped.to_masked_array().compressed()
    sde_array = list(sde_array)
    return np.mean(sde_array)

# '''GET RASTERS FIRST'''
# for i in list_range:
#     get_raster_t2m(i)
#     print("done:" + str(i))
'''MEAN OF MONTHS FOR ALPINE REGION'''

# mean_list_t2m_alps_january=[]
# for i in list_january:
#     mean_list_t2m_alps_january.append(clip_and_get_mean_t2m(shp_path_alps,
i))
#     print("done:" + str(i))
#
# mean_list_t2m_alps_february=[]
# for i in list_february:
#     mean_list_t2m_alps_february.append(clip_and_get_mean_t2m(shp_path_alps,
i))
#     print("done:" + str(i))
```

```
#
# mean_list_t2m_alps_march=[]
# for i in list_march:
#     mean_list_t2m_alps_march.append(clip_and_get_mean_t2m(shp_path_alps,
i))
#     print("done:" + str(i))
#
# mean_list_t2m_alps_april=[]
# for i in list_april:
#     mean_list_t2m_alps_april.append(clip_and_get_mean_t2m(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_t2m_alps_may=[]
# for i in list_may:
#     mean_list_t2m_alps_may.append(clip_and_get_mean_t2m(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_t2m_alps_june=[]
# for i in list_june:
#     mean_list_t2m_alps_june.append(clip_and_get_mean_t2m(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_t2m_alps_july=[]
# for i in list_july:
#     mean_list_t2m_alps_july.append(clip_and_get_mean_t2m(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_t2m_alps_august=[]
# for i in list_august:
#     mean_list_t2m_alps_august.append(clip_and_get_mean_t2m(shp_path_alps,
```

```
i))
#     print("done:" + str(i))
#
# mean_list_t2m_alps_september=[]
# for i in list_september:
#
mean_list_t2m_alps_september.append(clip_and_get_mean_t2m(shp_path_alps, i))
#     print("done:" + str(i))
#
#
# mean_list_t2m_alps_october=[]
# for i in list_october:
#     mean_list_t2m_alps_october.append(clip_and_get_mean_t2m(shp_path_alps,
i))
#     print("done:" + str(i))
#
#
# mean_list_t2m_alps_november=[]
# for i in list_november:
#
mean_list_t2m_alps_november.append(clip_and_get_mean_t2m(shp_path_alps, i))
#     print("done:" + str(i))
# #
# mean_list_t2m_alps_december=[]
# for i in list_december:
#
mean_list_t2m_alps_december.append(clip_and_get_mean_t2m(shp_path_alps, i))
#     print("done:" + str(i))
#
# alps_t2m= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
```

```
'August', 'September', 'October', 'November', 'December'],
#         'Means': [st.mean(mean_list_t2m_alps_january),
st.mean(mean_list_t2m_alps_february), st.mean(mean_list_t2m_alps_march),
st.mean(mean_list_t2m_alps_april), st.mean(mean_list_t2m_alps_may),
st.mean(mean_list_t2m_alps_june), st.mean(mean_list_t2m_alps_july),
st.mean(mean_list_t2m_alps_august), st.mean(mean_list_t2m_alps_september),
st.mean(mean_list_t2m_alps_october), st.mean(mean_list_t2m_alps_november),
st.mean(mean_list_t2m_alps_december)]}
# alps_t2m
=pd.DataFrame(alps_t2m).to_csv("E:/indirilenler/t2m/alps_t2m_means.csv")

'''MEAN OF MONTHS FOR TATRA MOUNTAINS'''
#
# mean_list_t2m_tatra_january=[]
# for i in list_january:
#     mean_list_t2m_tatra_january.append(clip_and_get_mean_t2m(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# mean_list_t2m_tatra_february=[]
# for i in list_february:
#
mean_list_t2m_tatra_february.append(clip_and_get_mean_t2m(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_t2m_tatra_march=[]
# for i in list_march:
#     mean_list_t2m_tatra_march.append(clip_and_get_mean_t2m(shp_path_tatra,
i))
#     print("done:" + str(i))
```

```
#
# mean_list_t2m_tatra_april=[]
# for i in list_april:
#     mean_list_t2m_tatra_april.append(clip_and_get_mean_t2m(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_t2m_tatra_may=[]
# for i in list_may:
#     mean_list_t2m_tatra_may.append(clip_and_get_mean_t2m(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_t2m_tatra_june=[]
# for i in list_june:
#     mean_list_t2m_tatra_june.append(clip_and_get_mean_t2m(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_t2m_tatra_july=[]
# for i in list_july:
#     mean_list_t2m_tatra_july.append(clip_and_get_mean_t2m(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_t2m_tatra_august=[]
# for i in list_august:
#     mean_list_t2m_tatra_august.append(clip_and_get_mean_t2m(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# mean_list_t2m_tatra_september=[]
# for i in list_september:
#
```

```
mean_list_t2m_tatra_september.append(clip_and_get_mean_t2m(shp_path_tatra,
i))
#     print("done:" + str(i))
#
#
# mean_list_t2m_tatra_october=[]
# for i in list_october:
#     mean_list_t2m_tatra_october.append(clip_and_get_mean_t2m(shp_path_tatra,
i))
#     print("done:" + str(i))
#
#
# mean_list_t2m_tatra_november=[]
# for i in list_november:
#
mean_list_t2m_tatra_november.append(clip_and_get_mean_t2m(shp_path_tatra,
i))
#     print("done:" + str(i))
# #
# mean_list_t2m_tatra_december=[]
# for i in list_december:
#
mean_list_t2m_tatra_december.append(clip_and_get_mean_t2m(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# tatra_t2m= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
'August', 'September', 'October', 'November', 'December'],
#         'Means': [st.mean(mean_list_t2m_tatra_january),
st.mean(mean_list_t2m_tatra_february), st.mean(mean_list_t2m_tatra_march),
```

```
st.mean(mean_list_t2m_tatra_april), st.mean(mean_list_t2m_tatra_may),
st.mean(mean_list_t2m_tatra_june), st.mean(mean_list_t2m_tatra_july),
st.mean(mean_list_t2m_tatra_august), st.mean(mean_list_t2m_tatra_september),
st.mean(mean_list_t2m_tatra_october), st.mean(mean_list_t2m_tatra_november),
st.mean(mean_list_t2m_tatra_december)]}
# tatra_t2m
=pd.DataFrame(tatra_t2m).to_csv("E:/indirilenler/t2m/tatra_t2m_means.csv")

'"MEAN OF MONTHS FOR KAÇKAR MOUNTAINS"'

# mean_list_t2m_kackar_january=[]
# for i in list_january:
#
mean_list_t2m_kackar_january.append(clip_and_get_mean_t2m(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_t2m_kackar_february=[]
# for i in list_february:
#
mean_list_t2m_kackar_february.append(clip_and_get_mean_t2m(shp_path_turkey
, i))
#     print("done:" + str(i))
#
# mean_list_t2m_kackar_march=[]
# for i in list_march:
#
mean_list_t2m_kackar_march.append(clip_and_get_mean_t2m(shp_path_turkey,
i))
#     print("done:" + str(i))
```

```
#
# mean_list_t2m_kackar_april=[]
# for i in list_april:
#
mean_list_t2m_kackar_april.append(clip_and_get_mean_t2m(shp_path_turkey, i))
#     print("done:" + str(i))
#
# mean_list_t2m_kackar_may=[]
# for i in list_may:
#
mean_list_t2m_kackar_may.append(clip_and_get_mean_t2m(shp_path_turkey, i))
#     print("done:" + str(i))
#
# mean_list_t2m_kackar_june=[]
# for i in list_june:
#
mean_list_t2m_kackar_june.append(clip_and_get_mean_t2m(shp_path_turkey, i))
#     print("done:" + str(i))
#
# mean_list_t2m_kackar_july=[]
# for i in list_july:
#     mean_list_t2m_kackar_july.append(clip_and_get_mean_t2m(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_t2m_kackar_august=[]
# for i in list_august:
#
mean_list_t2m_kackar_august.append(clip_and_get_mean_t2m(shp_path_turkey,
i))
```

```
#     print("done:" + str(i))
#
# mean_list_t2m_kackar_september=[]
# for i in list_september:
#
mean_list_t2m_kackar_september.append(clip_and_get_mean_t2m(shp_path_turkey, i))
#     print("done:" + str(i))
#
#
# mean_list_t2m_kackar_october=[]
# for i in list_october:
#
mean_list_t2m_kackar_october.append(clip_and_get_mean_t2m(shp_path_turkey, i))
#     print("done:" + str(i))
#
#
# mean_list_t2m_kackar_november=[]
# for i in list_november:
#
mean_list_t2m_kackar_november.append(clip_and_get_mean_t2m(shp_path_turkey, i))
#     print("done:" + str(i))
# #
# mean_list_t2m_kackar_december=[]
# for i in list_december:
#
mean_list_t2m_kackar_december.append(clip_and_get_mean_t2m(shp_path_turkey, i))
```

```
#     print("done:" + str(i))
#
# kackar_t2m= {'Months': ['January', 'February', 'March', 'April', 'May', 'June',
'July', 'August', 'September', 'October', 'November', 'December'],
#         'Means': [st.mean(mean_list_t2m_kackar_january),
st.mean(mean_list_t2m_kackar_february), st.mean(mean_list_t2m_kackar_march),
st.mean(mean_list_t2m_kackar_april), st.mean(mean_list_t2m_kackar_may),
st.mean(mean_list_t2m_kackar_june), st.mean(mean_list_t2m_kackar_july),
st.mean(mean_list_t2m_kackar_august),
st.mean(mean_list_t2m_kackar_september),
st.mean(mean_list_t2m_kackar_october),
st.mean(mean_list_t2m_kackar_november),
st.mean(mean_list_t2m_kackar_december)]}
# kackar_t2m
=pd.DataFrame(kackar_t2m).to_csv("E:/indirilenler/t2m/kackar_t2m_means.csv")



def get_raster_tp(i):
#Open the NetCDF

    sde_full = xr.open_dataset('E:/indirilenler/tp_full.nc')
    sde= sde_full['tp']
    sde.rio.write_crs("epsg:4326", inplace=True)
#(Optional) convert longitude from (0-360) to (-180 to 180) (if required)
    sde.coords['longitude'] = (sde.coords['longitude'] + 180) % 360 - 180
    sde = sde.sortby(sde.longitude)

#Define lat/long
    sde = sde.rio.set_spatial_dims('longitude', 'latitude')
    sde[i].rio.to_raster("E:/indirilenler/tp/tp_" + str(i) + ".tiff", compress='LZMA',
```

```
tiled=True, dtype="float64")

def clip_and_get_mean_tp(shp_path, i):


    sde_alps_try = rioxarray.open_rasterio("E:/indirilenler/tp/tp_"+str(i)+".tiff",
                                masked=True).squeeze()
    shapes_alps = gpd.read_file(shp_path)
    sde_clipped = sde_alps_try.rio.clip(shapes_alps.geometry.apply(mapping),
shapes_alps.crs)
    sde_array = sde_clipped.to_masked_array().compressed()
    sde_array=list(sde_array)
    return np.mean(sde_array)

# '''GET RASTERS FIRST'''
# for i in list_range:
#     get_raster_tp(i)
#     print("done:" + str(i))


'''MEAN OF MONTHS FOR ALPINE REGION'''

# mean_list_tp_alps_january=[]
# for i in list_january:
#     mean_list_tp_alps_january.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_tp_alps_february=[]
# for i in list_february:
#     mean_list_tp_alps_february.append(clip_and_get_mean_tp(shp_path_alps, i))
```

```
#     print("done:" + str(i))
#
# mean_list_tp_alps_march=[]
# for i in list_march:
#     mean_list_tp_alps_march.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_tp_alps_april=[]
# for i in list_april:
#     mean_list_tp_alps_april.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_tp_alps_may=[]
# for i in list_may:
#     mean_list_tp_alps_may.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_tp_alps_june=[]
# for i in list_june:
#     mean_list_tp_alps_june.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_tp_alps_july=[]
# for i in list_july:
#     mean_list_tp_alps_july.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
# mean_list_tp_alps_august=[]
# for i in list_august:
#     mean_list_tp_alps_august.append(clip_and_get_mean_tp(shp_path_alps, i))
```

```
#     print("done:" + str(i))
#
# mean_list_tp_alps_september=[]
# for i in list_september:
#     mean_list_tp_alps_september.append(clip_and_get_mean_tp(shp_path_alps,
i))
#     print("done:" + str(i))
#
#
# mean_list_tp_alps_october=[]
# for i in list_october:
#     mean_list_tp_alps_october.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
#
# mean_list_tp_alps_november=[]
# for i in list_november:
#     mean_list_tp_alps_november.append(clip_and_get_mean_tp(shp_path_alps,
i))
#     print("done:" + str(i))
# #
# mean_list_tp_alps_december=[]
# for i in list_december:
#     mean_list_tp_alps_december.append(clip_and_get_mean_tp(shp_path_alps, i))
#     print("done:" + str(i))
#
# alps_tp= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
'August', 'September', 'October', 'November', 'December'],
#         'Means': [np.nanmean(mean_list_tp_alps_january),
st.mean(mean_list_tp_alps_february), st.mean(mean_list_tp_alps_march),
```

```
st.mean(mean_list_tp_alps_april), st.mean(mean_list_tp_alps_may),
st.mean(mean_list_tp_alps_june), st.mean(mean_list_tp_alps_july),
st.mean(mean_list_tp_alps_august), st.mean(mean_list_tp_alps_september),
st.mean(mean_list_tp_alps_october), st.mean(mean_list_tp_alps_november),
np.nanmean(mean_list_tp_alps_december)]}
# alps_tp =pd.DataFrame(alps_tp).to_csv("E:/indirilenler/tp/alps_tp_means.csv")

'''MEAN OF MONTHS FOR TATRA MOUNTAINS'''

# mean_list_tp_tatra_january=[]
# for i in list_january:
#     mean_list_tp_tatra_january.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_february=[]
# for i in list_february:
#     mean_list_tp_tatra_february.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_march=[]
# for i in list_march:
#     mean_list_tp_tatra_march.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_april=[]
# for i in list_april:
#     mean_list_tp_tatra_april.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_may=[]
```

```
# for i in list_may:
#     mean_list_tp_tatra_may.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_june=[]
# for i in list_june:
#     mean_list_tp_tatra_june.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_july=[]
# for i in list_july:
#     mean_list_tp_tatra_july.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_august=[]
# for i in list_august:
#     mean_list_tp_tatra_august.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
#
# mean_list_tp_tatra_september=[]
# for i in list_september:
#     mean_list_tp_tatra_september.append(clip_and_get_mean_tp(shp_path_tatra,
i))
#     print("done:" + str(i))
#
#
# mean_list_tp_tatra_october=[]
# for i in list_october:
#     mean_list_tp_tatra_october.append(clip_and_get_mean_tp(shp_path_tatra, i))
#     print("done:" + str(i))
```

```
#
#
# mean_list_tp_tatra_november=[]
# for i in list_november:
#     mean_list_tp_tatra_november.append(clip_and_get_mean_tp(shp_path_tatra,
i))
#     print("done:" + str(i))
# #
# mean_list_tp_tatra_december=[]
# for i in list_december:
#     mean_list_tp_tatra_december.append(clip_and_get_mean_tp(shp_path_tatra,
i))
#     print("done:" + str(i))
#
# tatra_tp= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
'August', 'September', 'October', 'November', 'December'],
#         'Means': [np.nanmean(mean_list_tp_tatra_january),
st.mean(mean_list_tp_tatra_february), st.mean(mean_list_tp_tatra_march),
st.mean(mean_list_tp_tatra_april), st.mean(mean_list_tp_tatra_may),
st.mean(mean_list_tp_tatra_june), st.mean(mean_list_tp_tatra_july),
st.mean(mean_list_tp_tatra_august), st.mean(mean_list_tp_tatra_september),
st.mean(mean_list_tp_tatra_october), st.mean(mean_list_tp_tatra_november),
np.nanmean(mean_list_tp_tatra_december)]}
# tatra_tp =pd.DataFrame(tatra_tp).to_csv("E:/indirilenler/tp/tatra_tp_means.csv")

'"MEAN OF MONTHS FOR KAÇKAR MOUNTAINS"'

# mean_list_tp_kackar_january=[]
# for i in list_january:
#     mean_list_tp_kackar_january.append(clip_and_get_mean_tp(shp_path_turkey,
```

```
i))
#     print("done:" + str(i))
#
# mean_list_tp_kackar_february=[]
# for i in list_february:
#
mean_list_tp_kackar_february.append(clip_and_get_mean_tp(shp_path_turkey, i))
#     print("done:" + str(i))
#
# mean_list_tp_kackar_march=[]
# for i in list_march:
#     mean_list_tp_kackar_march.append(clip_and_get_mean_tp(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_tp_kackar_april=[]
# for i in list_april:
#     mean_list_tp_kackar_april.append(clip_and_get_mean_tp(shp_path_turkey, i))
#     print("done:" + str(i))
#
# mean_list_tp_kackar_may=[]
# for i in list_may:
#     mean_list_tp_kackar_may.append(clip_and_get_mean_tp(shp_path_turkey, i))
#     print("done:" + str(i))
#
# mean_list_tp_kackar_june=[]
# for i in list_june:
#     mean_list_tp_kackar_june.append(clip_and_get_mean_tp(shp_path_turkey, i))
#     print("done:" + str(i))
#
```

```
# mean_list_tp_kackar_july=[]
# for i in list_july:
#     mean_list_tp_kackar_july.append(clip_and_get_mean_tp(shp_path_turkey, i))
#     print("done:" + str(i))
#
# mean_list_tp_kackar_august=[]
# for i in list_august:
#     mean_list_tp_kackar_august.append(clip_and_get_mean_tp(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# mean_list_tp_kackar_september=[]
# for i in list_september:
#
mean_list_tp_kackar_september.append(clip_and_get_mean_tp(shp_path_turkey,
i))
#     print("done:" + str(i))
#
#
# mean_list_tp_kackar_october=[]
# for i in list_october:
#     mean_list_tp_kackar_october.append(clip_and_get_mean_tp(shp_path_turkey,
i))
#     print("done:" + str(i))
#
#
# mean_list_tp_kackar_november=[]
# for i in list_november:
#
mean_list_tp_kackar_november.append(clip_and_get_mean_tp(shp_path_turkey,
```

```
i))
#     print("done:" + str(i))
# #
# mean_list_tp_kackar_december=[]
# for i in list_december:
#
mean_list_tp_kackar_december.append(clip_and_get_mean_tp(shp_path_turkey,
i))
#     print("done:" + str(i))
#
# kackar_tp= {'Months': ['January', 'February', 'March', 'April', 'May', 'June', 'July',
'August', 'September', 'October', 'November', 'December'],
#         'Means': [np.nanmean(mean_list_tp_kackar_january),
st.mean(mean_list_tp_kackar_february), st.mean(mean_list_tp_kackar_march),
st.mean(mean_list_tp_kackar_april), st.mean(mean_list_tp_kackar_may),
st.mean(mean_list_tp_kackar_june), st.mean(mean_list_tp_kackar_july),
st.mean(mean_list_tp_kackar_august), st.mean(mean_list_tp_kackar_september),
st.mean(mean_list_tp_kackar_october), st.mean(mean_list_tp_kackar_november),
np.nanmean(mean_list_tp_kackar_december)]}
# kackar_tp
=pd.DataFrame(kackar_tp).to_csv("E:/indirilenler/tp/kackar_tp_means.csv")
```

H. Python Script for the Reclassification of FSCOG Data

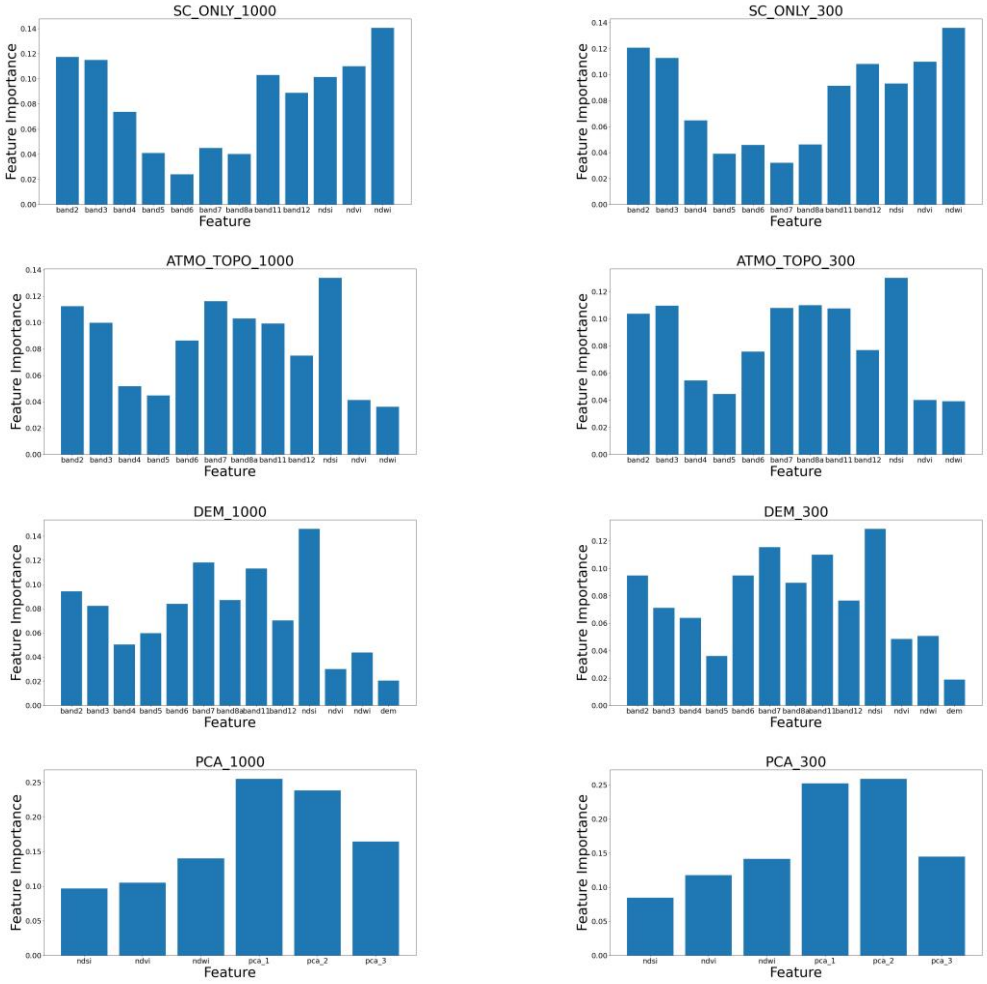
```
import numpy as np
# define location for raster
fsc_path =
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/Snow_oC_uC/FSC/"
sog_raster_path =
'FSC_20190304T095656_S2B_T34UCV_V100_1/FSC_20190304T095656_S2B_
T34UCV_V100_1/FSC_20181214T095652_S2B_T34UCV_V100_1_FSCOG.tif'

# use rasterio to open the raster for reading

import gdal

#1.
tiff_file = gdal.Open(fsc_path + sog_raster_path)
#2.
geotransform = tiff_file.GetGeoTransform()
projection = tiff_file.GetProjection()
band = tiff_file.GetRasterBand(1)
xsize = band.XSize
ysize = band.YSize
#3.
array = band.ReadAsArray()
tiff_file = None #close it
band = None #close it
#4.
array = np.where(np.logical_and(array>=0 , array<=100) , 1 , array) #snow
array = np.where(array==255, 4, array) #nodata
array = np.where(array==205, 3, array) #cloud or cloud shadow
```

```
array = np.where(array==0, 2, array)  #nosnow


print(array)
#5.
output_raster_name = "tatra_14_Dec_2018_sog.tiff"
driver = gdal.GetDriverByName('GTiff')
new_tiff = driver.Create(output_raster_name,xsize,ysize,1,gdal.GDT_Float32)
new_tiff.SetGeoTransform(geotransform)
new_tiff.SetProjection(projection)
new_tiff.GetRasterBand(1).WriteArray(array)
new_tiff.FlushCache() #Saves to disk
new_tiff = None #closes the file

#1.
tiff_file2 = gdal.Open(fsc_path + toc_raster_path)
#2.
geotransform2 = tiff_file2.GetGeoTransform()
projection2 = tiff_file2.GetProjection()
band2 = tiff_file2.GetRasterBand(1)
xsize2 = band2.XSize
ysize2 = band2.YSize
#3.
array2 = band2.ReadAsArray()
tiff_file2 = None #close it
band2 = None #close it
#4.
array2 = np.where(np.logical_and(array>0 , array<=100) , 1 , array)
array2 = np.where(array==255, 4, array)
array2 = np.where(array==205, 3, array)
```

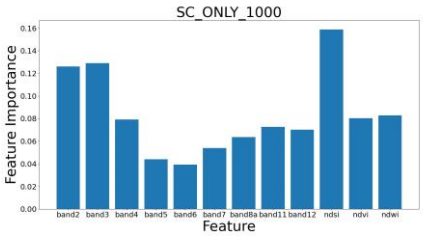
```
array2 = np.where(array==0, 2, array)


print(array)
#5.
output_raster_name2 = "tatra_14_Dec_2018_toc.tiff"
driver2 = gdal.GetDriverByName('GTiff')
new_tiff2 = driver.Create(output_raster_name2,xsize2,ysize2,1,gdal.GDT_Float32)
new_tiff2.SetGeoTransform(geotransform2)
new_tiff2.SetProjection(projection2)
new_tiff2.GetRasterBand(1).WriteArray(array2)
new_tiff2.FlushCache() #Saves to disk
new_tiff2 = None #closes the file
```

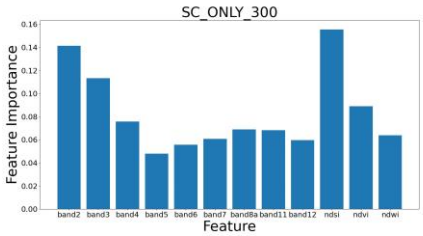
I. Feature Importance of RF

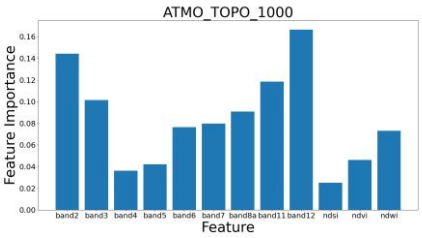
ALPS REGION – 5 DECEMBER 2018

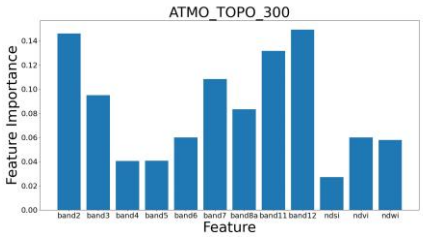


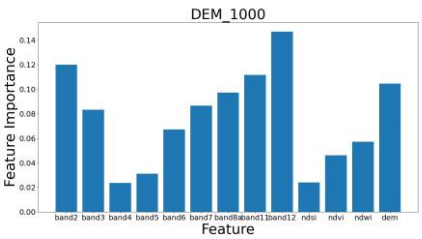
ALPS REGION – 24 JANUARY 2019

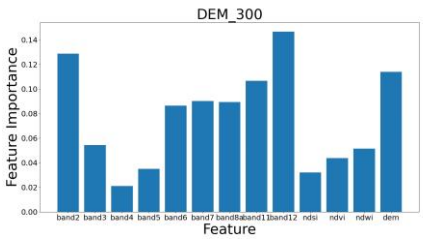


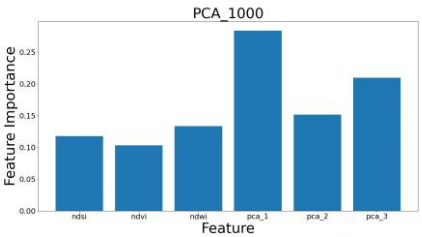


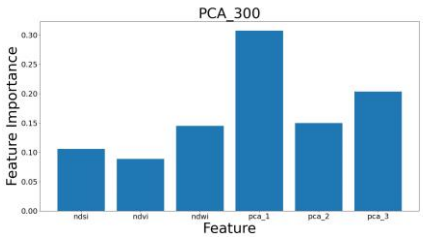




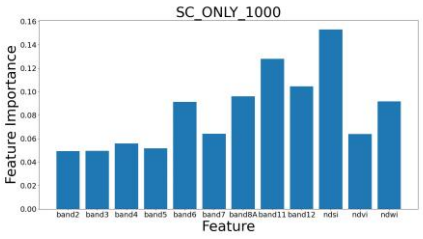


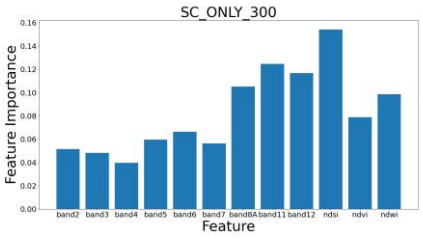


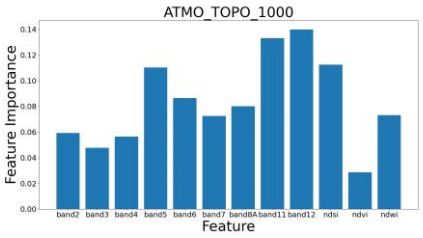


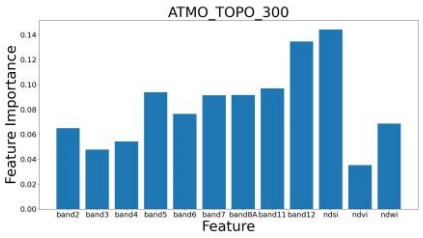


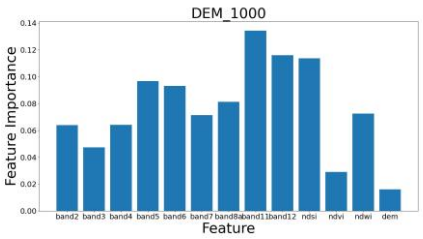
ALPS REGION –13 JUNE 2019

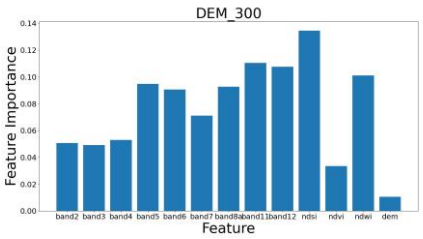


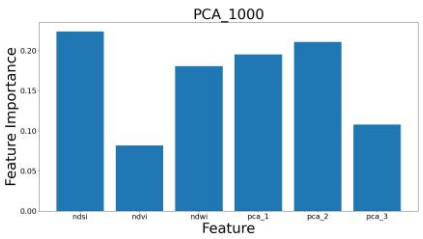


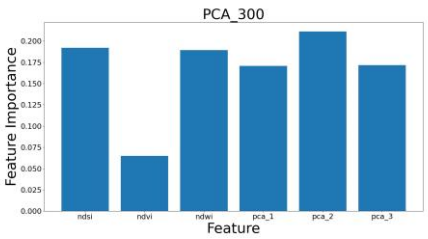




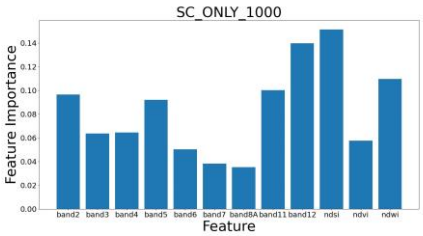


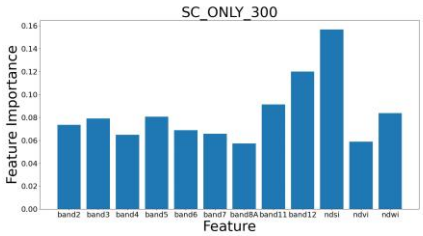


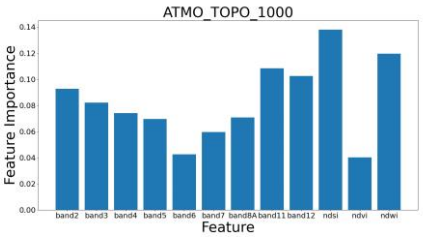


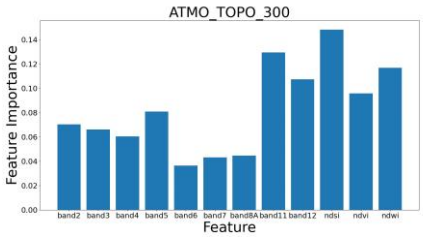


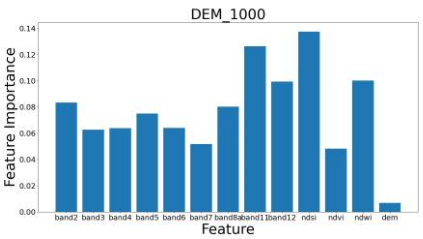
TATRA MOUNTAINS –29 NOVEMBER 2018

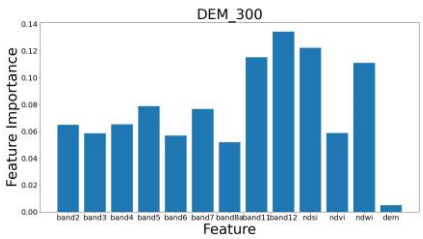


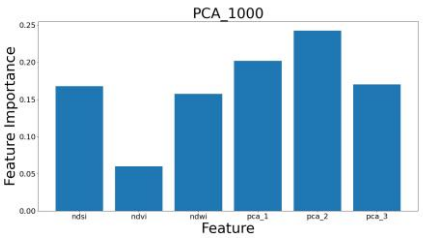


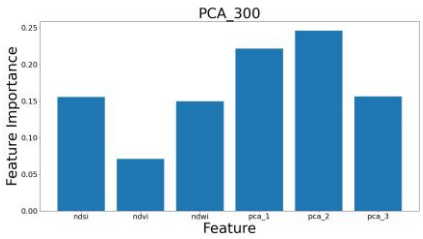




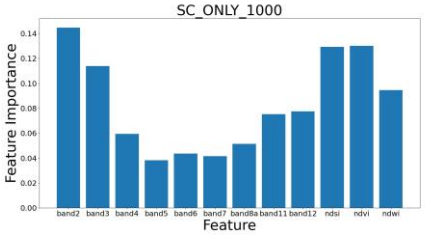


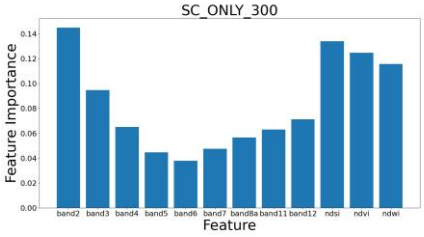


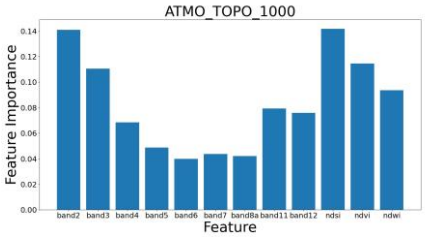


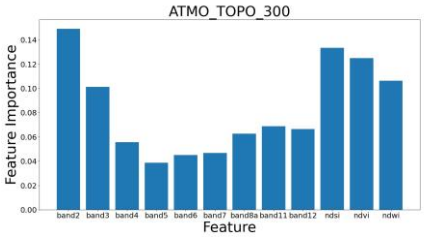


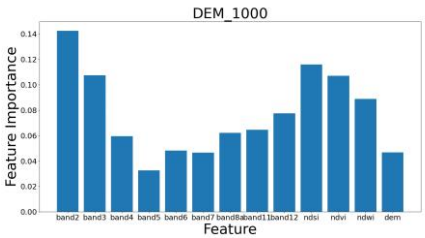
TATRA MOUNTAINS –23 JANUARY 2020

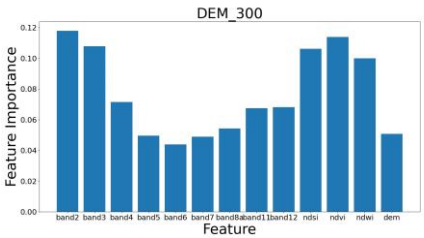


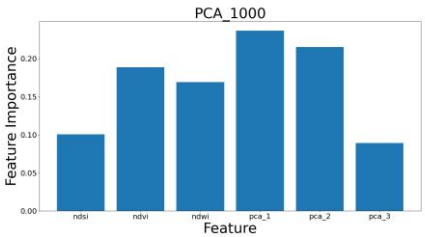


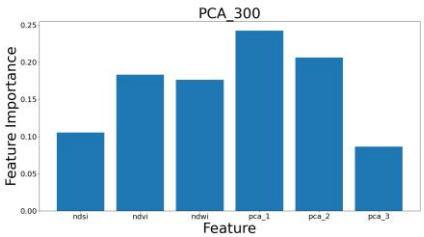




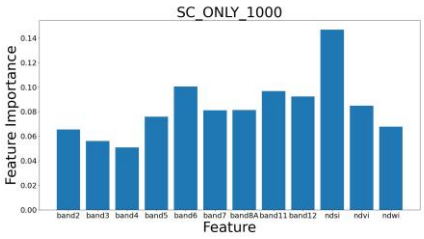


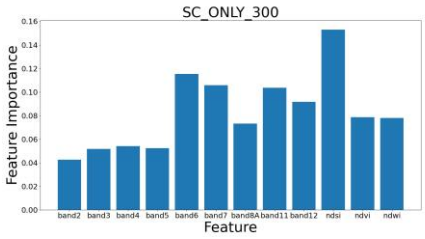


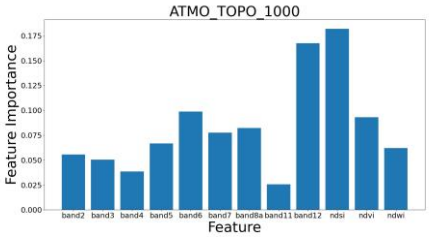


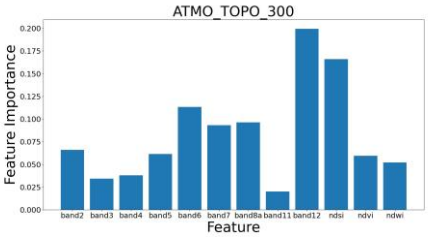


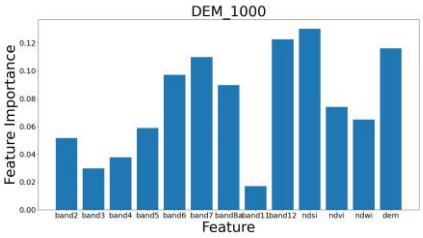
TATRA MOUNTAINS –8 APRIL 2019

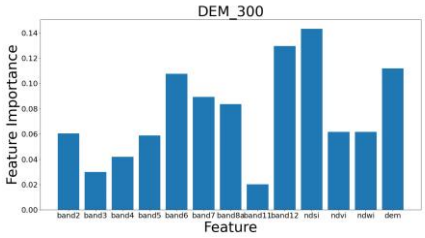


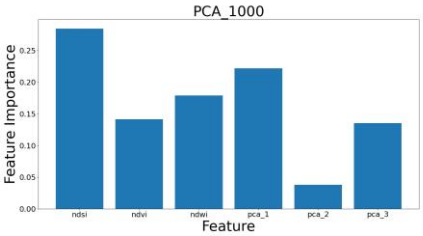


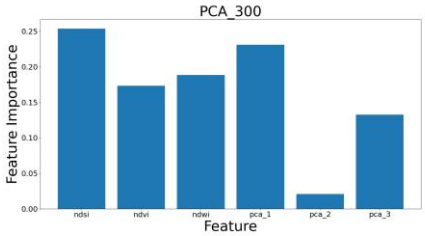




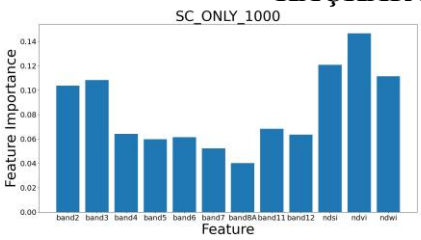


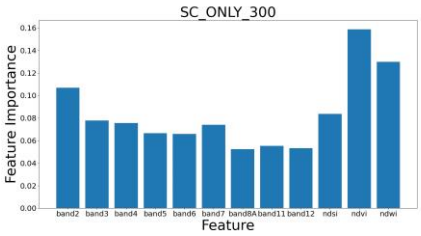


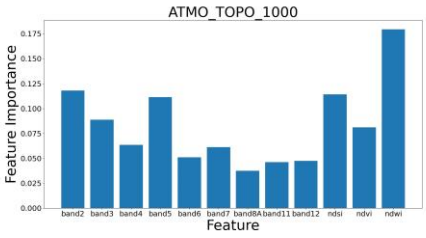


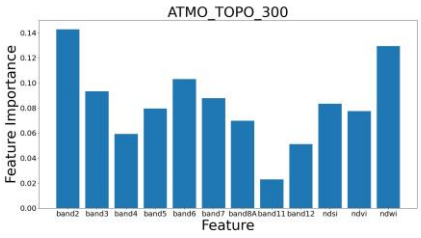


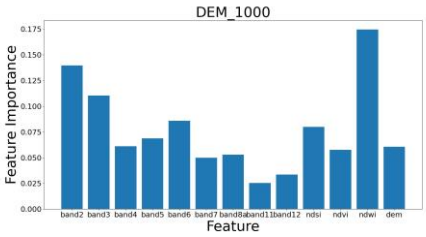
KAÇKAR MOUNTAINS –19 DECEMBER 2019

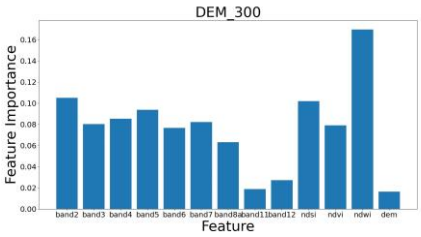


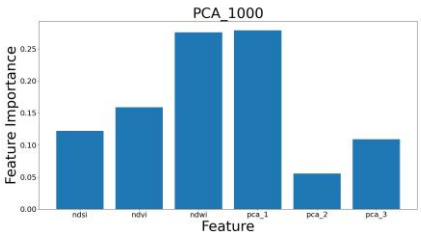


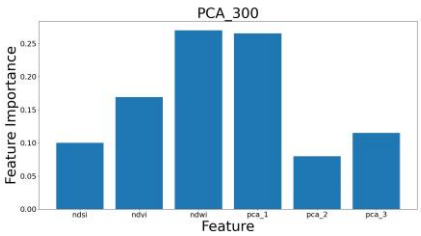




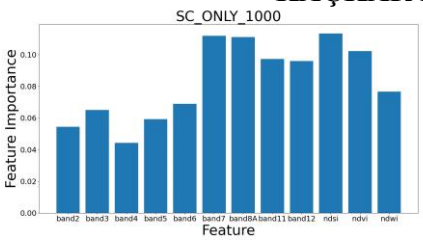


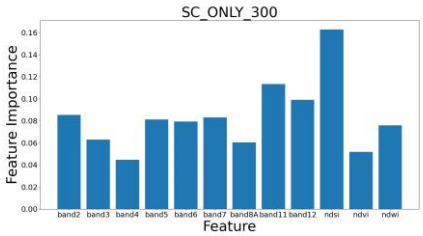


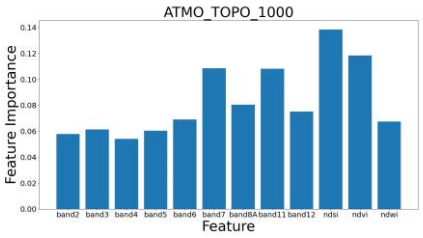


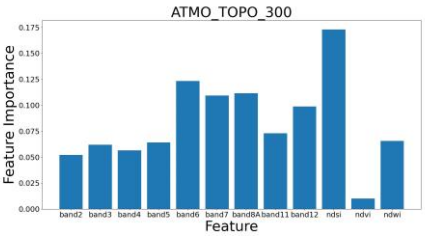


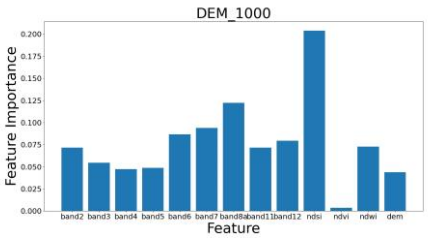
KAÇKAR MOUNTAINS –19 MARCH 2018

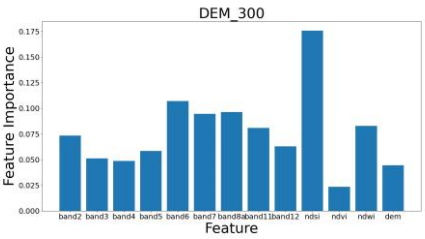


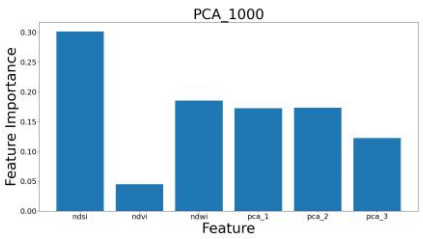


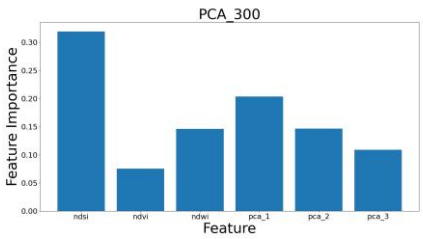




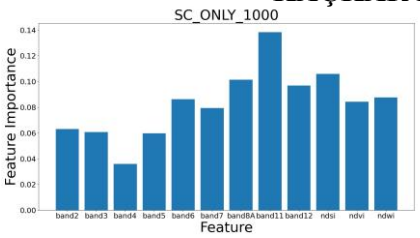


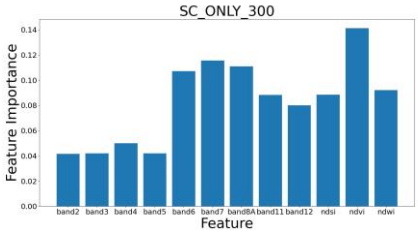


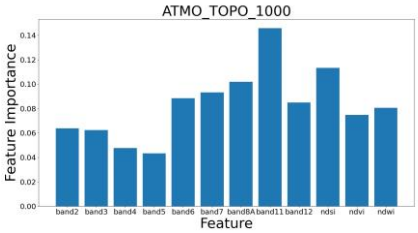


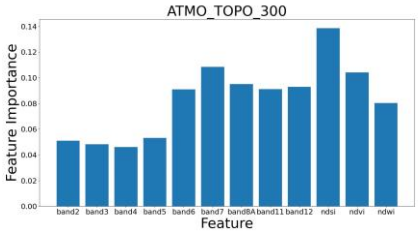


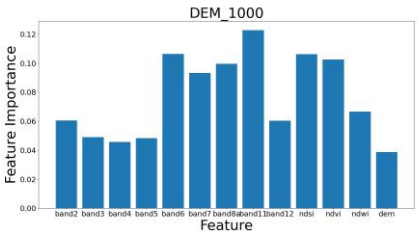
KAÇKAR MOUNTAINS –8 APRIL 2018

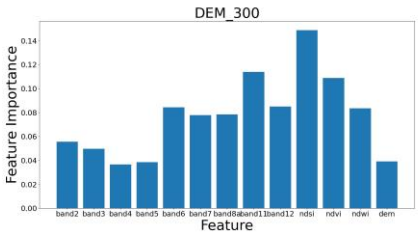


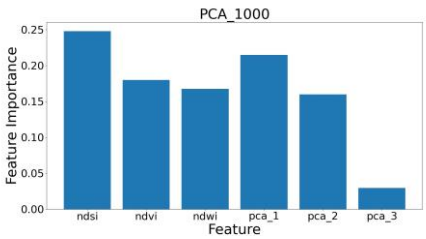


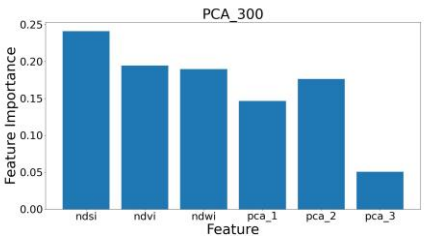








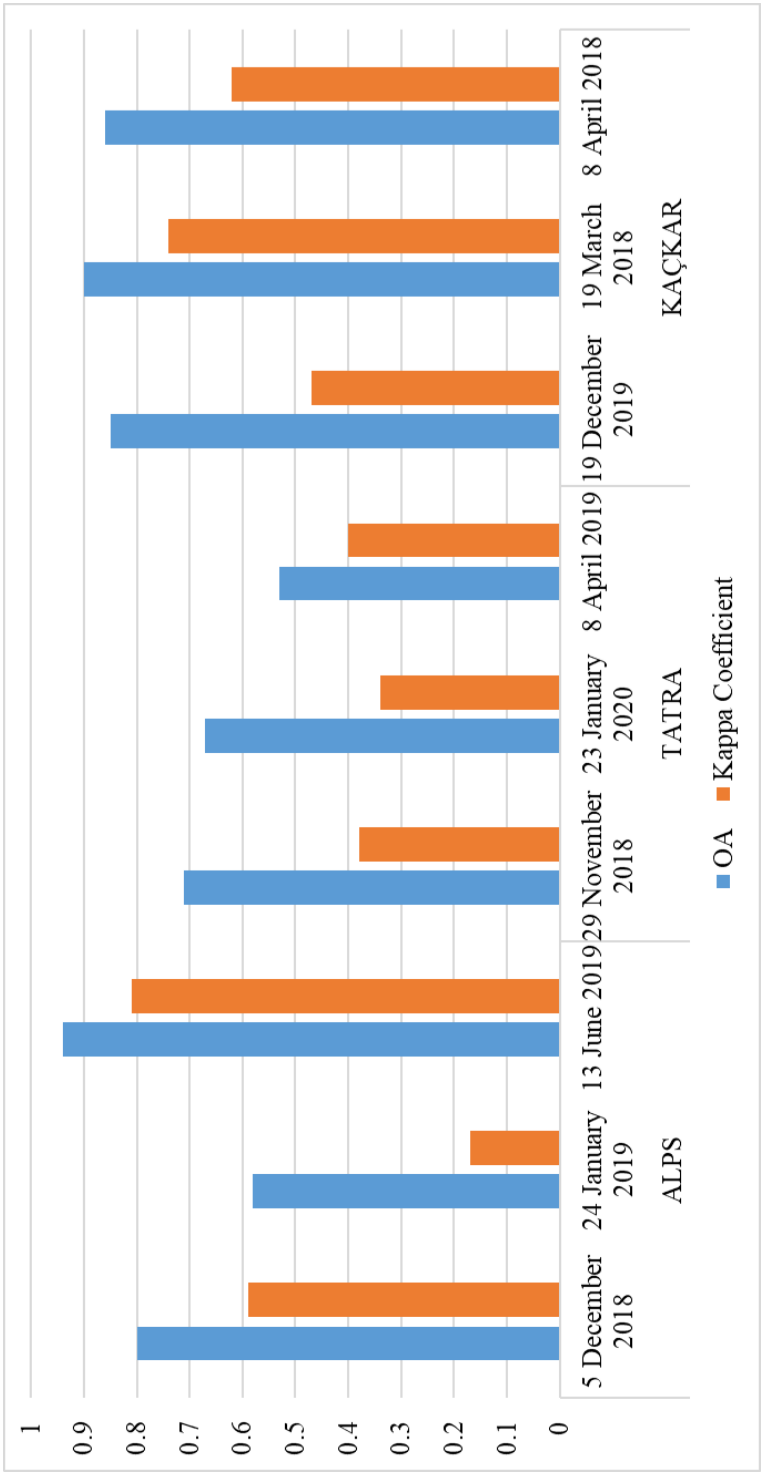




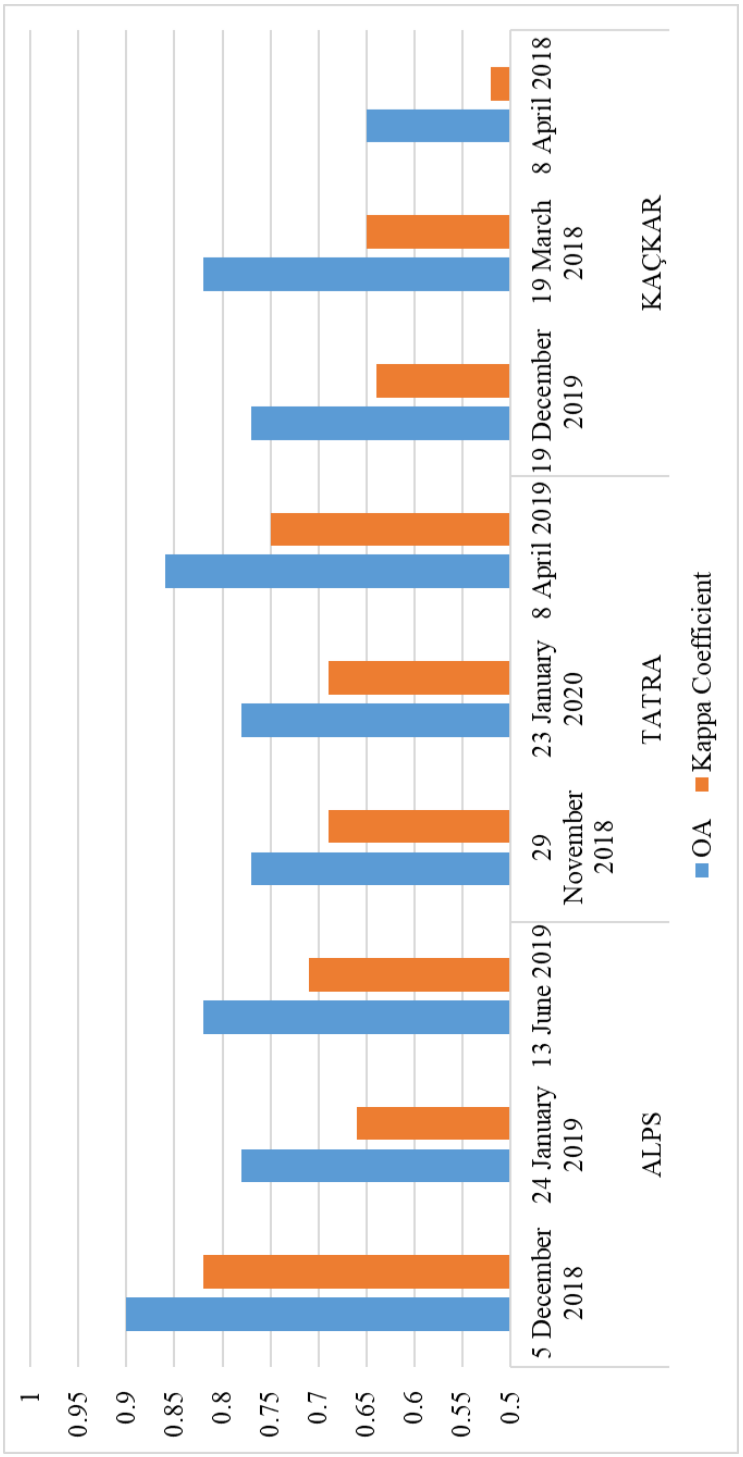
J. Ratio of Eigenvalues of *Pca* Input Combination (for full tiles)

Location	*Date of the Product*	*Ratio of Eigenvalues*	*Cumulative of Eigenvalues*
Alps Region	5 December 2018	**PCA1:** 0.96575223 **PCA2:** 0.026839 **PCA3**: 0.00739108	**PCA1:** 0.96575223 **PCA2:** 0.99259124 **PCA3:** 0.99998231
	24 January 2019	**PCA1:** 0.97626484 **PCA2**: 0.01904372 **PCA3:**0.004668	**PCA1:** 0.97626484 **PCA2:** 0.99530856 **PCA3:** 0.99997656
	13 June 2019	**PCA1:** 0.89541703 **PCA2:** 0.06080271 **PCA3:** 0.0437662	**PCA1:** 0.89541703 **PCA2:** 0.95621974 **PCA3:** 0.99998593
Tatra Mountains	29 November 2018	**PCA1:** 0.86222486 **PCA2**: 0.093171 **PCA3**: 0.04102694	**PCA1:** 0.86222486 **PCA2:** 0.95539586 **PCA3:** 0.99642281
	23 January 2020	**PCA1:** 0.913538 **PCA2:** 0.069525 **PCA3:** 0.016937	**PCA1:** 0.913538 **PCA2:** 0.983063 **PCA3:** 1.00
	8 April 2019	**PCA1:** 0.9221785 **PCA2:** 0.0555026 **PCA3**: 0.0223189	**PCA1:** 0.9221785 **PCA2:** 0.9776811 **PCA3:** 1.00
Kackar Mountains	19 December 2019	**PCA1:** 0.98353166 **PCA2:** 0.01354346 **PCA3:** 0.00286949	**PCA1:** 0.98353166 **PCA2:** 0.99707512 **PCA3:** 0.99994461
	19 March 2018	**PCA1:** 0.98515067 **PCA2:** 0.0129161 **PCA3:** 0.00192669	**PCA1:** 0.98515067 **PCA2:** 0.99806676 **PCA3:** 0.99999345
	8 April 2018	**PCA1:** 0.91735474 **PCA2:** 0.07586615 **PCA3:** 0.00677911	**PCA1:** 0.91735474 **PCA2:** 0.99322089 **PCA3:** 1.00

K. Base Study (NDSI) Accuracy Assessment Results



L. Input Combination of Only First Three Principal Components - Accuracy Assessment Results



M. Python Script for Classification of NDSI

```
mport numpy as np
# define location for raster

import rasterio.mask
from shapely.geometry import shape, mapping
from shapely.ops import unary_union
import fiona
import itertools
import pandas as pd
import numpy as np
import glob
import os
import gdal



path =
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/03_TURKEY/Work_Folde
r/8_Apr_2018_atmo_topo"
ndsi_path =
"D:/Drivers/GGIT/SK_TEZ_110622/Cansu_Tez_Draft/03_TURKEY/Work_Folde
r/8_Apr_2018_atmo_topo/S2_Bands_TIFF/NDSI.tif"
output_path = "E:/TEZ/NDSI/03_TURKEY/8_Apr_2018/"



def get_test_data(path,ndsi_path,output_path):
    with fiona.open(path + "/Test_Region/Clip_Frame1.shp", "r") as shapefile:
        for feature in shapefile:
            shapes = [feature["geometry"]]
```

```
    with rasterio.open(ndsi_path) as src:
        out_image, out_trasform = rasterio.mask.mask(src, shapes, crop="True")
        out_meta = src.meta

    out_meta.update({
        'driver': 'Gtiff',
        'height': out_image.shape[1],
        'width': out_image.shape[2],
        'transform': out_trasform
    })

    with rasterio.open(output_path + "NDSI_clipped.tif", "w", **out_meta) as dst:
        dst.write(out_image)

#
get_test_data(path,ndsi_path,output_path)


# use rasterio to open the raster for reading


def classify(output_path):
    #1.
    tiff_file = gdal.Open(output_path + "NDSI_clipped.tif")
    #2.
    geotransform = tiff_file.GetGeoTransform()
    projection = tiff_file.GetProjection()
    band = tiff_file.GetRasterBand(1)
    xsize = band.XSize
    ysize = band.YSize
    #3.
```

```
    array = band.ReadAsArray()
    tiff_file = None #close it
    band = None #close it
    #4.
    array = np.where(array>0 , 1 , array) #snow
    array = np.where(array<=0, 0, array) #nosnow
    #5.
    output_raster_name = "NDSI_class.tiff"
    driver = gdal.GetDriverByName('GTiff')
    new_tiff = driver.Create(output_path +
output_raster_name,xsize,ysize,1,gdal.GDT_Float32)
    new_tiff.SetGeoTransform(geotransform)
    new_tiff.SetProjection(projection)
    new_tiff.GetRasterBand(1).WriteArray(array)
    new_tiff.FlushCache() #Saves to disk
    new_tiff = None #closes the file
classify(output_path)
```

N. Number Of Test Samples Per Class for 20 Km × 20 Km Subsets of Sentinel-2 Images, Calculated By Multinomial Distribution Formula (Ozdarici Ok & Akyurek, 2012; Jensen, 2006) for NDSI Accuracy Assessment

Location	Date	Number of Points (classes)		
		Snow	No snow	Total
ALPS REGION	5 December 2018	100	100	200
	24 January 2019	100	370	470
	13 June 2019	360	400	760
TATRA MOUNTAINS	29 November 2018	100	360	460
	23 January 2020	290	440	730
	8 April 2019	100	100	200
KAÇKAR MOUNTAINS	19 December 2019	365	100	465
	19 March 2019	100	250	350
	8 April 2018	100	390	490

TEZ İZİN FORMU / THESIS PERMISSION FORM

ENSTİTÜ / INSTITUTE

Fen Bilimleri Enstitüsü / Graduate School of Natural and Applied Sciences	X
Sosyal Bilimler Enstitüsü / Graduate School of Social Sciences	
Uygulamalı Matematik Enstitüsü / Graduate School of Applied Mathematics	
Enformatik Enstitüsü / Graduate School of Informatics	
Deniz Bilimleri Enstitüsü / Graduate School of Marine Sciences	

YAZARIN / AUTHOR

Soyadı / Surname : AKSU
Adı / Name : CANSU
Bölümü / Department : GEODETIC AND GEOGRAPHICAL INFORMATION TECHNOLOGIES

TEZİN ADI / TITLE OF THE THESIS (**İngilizce** / English) : ASSESSMENT OF RANDOM FOREST METHOD IN PIXEL-BASED SNOW COVER CLASSIFICATION IN ALPINE REGION, TATRA MOUNTAINS AND KAÇKAR MOUNTAINS

TEZİN TÜRÜ / DEGREE: Yüksek Lisans / Master [X] **Doktora** / PhD [ ]

1. **Tezin tamamı dünya çapında erişime açılacaktır. /** Release the entire work immediately for access worldwide. [X]
2. **Tez iki yıl süreyle erişime kapalı olacaktır.** / Secure the entire work for patent and/or proprietary purposes for a period of **two year. *** [ ]
3. **Tez altı ay süreyle erişime kapalı olacaktır.** / Secure the entire work for period of **six months**. ***** [ ]

***** *Enstitü Yönetim Kurulu Kararının basılı kopyası tezle birlikte kütüphaneye teslim edilecektir.*
A copy of the Decision of the Institute Administrative Committee will be delivered to the library together with the printed thesis.

Yazarın imzası / Signature ..... **Tarih** / Date .....