HACETTEPE UNIVERSITY
FACULTY OF ENGINEERING
DEPARTMENT OF GEOMATICS ENGINEERING

GRADUATION PROJECT REPORT
2020-2021

<FOREST FIRES INVESTIGATION >

TEAM MEMBERS

<BERK KIVILCIM >
<OKAN DEMİRBİLEK >
<FATİH AFACAN >

SUPERVISOR
<DR. BERK ANBAROGLU>

01/06/2021

ABSTRACT

The project is based on Google Earth Engine (GEE) technology with JavaScript programming language; allow us to reach satellite images and process them with different functions and algorithms. According to calculations and analysis the code generate and illustrate charts and values. To exemplify it; how many hectare burned, when the fire started and ended, temperature graphics, burn severity rate, health situation of vegetation, accuracy assessments of calculations, visual representations of burned areas, pre and post fire images are data we obtained and calculated. With the help of data we obtained, reliability of Google Earth Engine's functions, algorithms and different methods reliabilities tested and compared to each other. The code prepared for public usage so this report include "how to use tutorial" of the code. With the website created; data and analysis can be stored and represented.

TABLE OF CONTENTS

ABSTRACT 2
TABLE OF CONTENTS 3
LIST OF FIGURES 4
LIST OF TABLES 5
1 PROJECT INTRODUCTION 6
1.1 Problem Definition and Motivation 6
1.2 Goals and Objectives 6
1.3 Methodology 7
1.4 Structure of the Report 9
2 BACKGROUND 10
2.1 Web Part 10
2.2 Spectral index and Programming Language Selection 10
3 DESIGN OF THE <PROPOSED SOLUTION/PRODUCT/SYSTEM> 11
4 IMPLEMENTATION AND TESTS (or Application and Results) 23
5 DISCUSSION 36
6 CONCLUSION 38
REFERENCES 39
APPENDICES 40

LIST OF FIGURES

Figure 1 6
Figure 2 7
Figure 3 7
Figure 4 8
Figure 5 8
Figure 6 Example Figure ..... **Hata! Yer işareti tanımlanmamış.**
Figure 7 12
Figure 8 12
Figure 9 13
Figure 10 13
Figure 11 13
Figure 12 13
Figure 13 14
Figure 14 14
Figure 15 14
Figure 16 15
Figure 17 15
Figure 18 16
Figure 19 16
Figure 20 16
Figure 21 18
Figure 22 18
Figure 23 18
Figure 24 19
Figure 25 19
Figure 26 19
Figure 27 20
Figure 28 20
Figure 29 20
Figure 30 22
Figure 31 24
Figure 32 24
Figure 33 24
Figure 34 25
Figure 35 25
Figure 36 25
Figure 37 26
Figure 38 26
Figure 39 27
Figure 40 27
Figure 41 28
Figure 42 28
Figure 43 36

LIST OF TABLES

Table 1 Example Table ..... **Hata! Yer işareti tanımlanmamış.**

1 PROJECT INTRODUCTION

In this project; forest fires investigated and analyzed with Google Earth Engine (GEE) technology. Google Earth Engine developed by google during 2010 and this technology allow us to reach and process on some satellite's (sentinel2, Landsat, MODIS) image catalogs with different platforms such as python, JavaScript and REST. JavaScript is the most popular and common programming language used by google earth engine users beside it has a lot of algorithm functions so our gee codes written with JavaScript language. JavaScript layout illustrated below here in as figure.

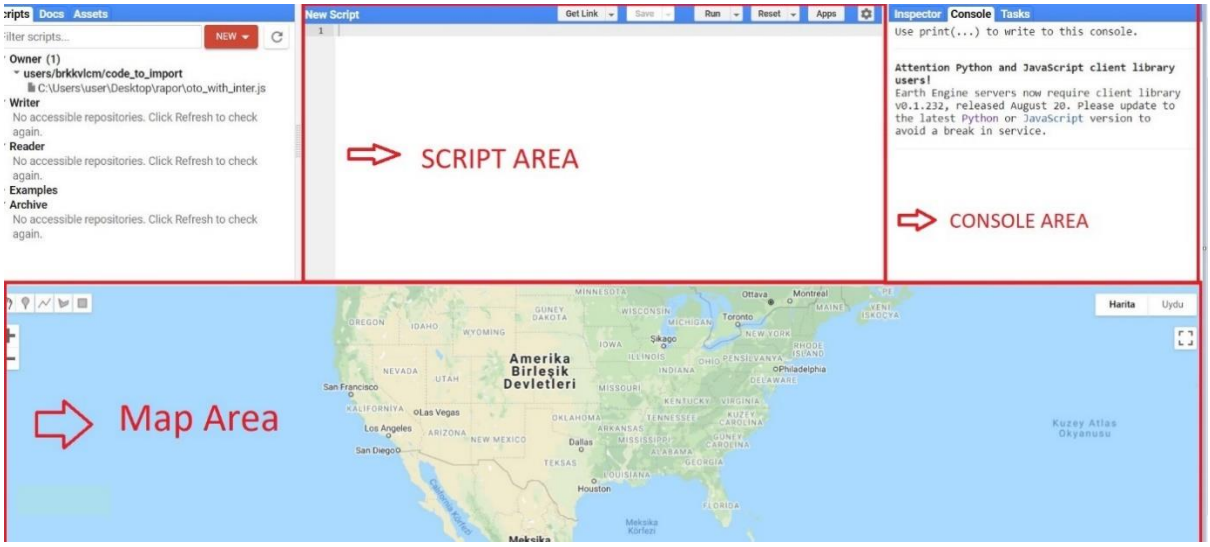


Figure 1

- **Script Area:** The area is where we put our code.
- **Console Area:** The area is where graphical and numerical results displayed.
- **Map Area:** The area is where results of visualization displayed

1.1 Problem Definition and Motivation

Forest are very important objects for earth's ecology, nature and life cycle. Due to this forest fires damages can be devastate and effect all of the living things on earth. Analysis the reason and statistical data of forest fires is very crucial for estimate and prevent the damage. Beside some forest fires deliberately started by humans. The reason behind this to start a construct on the burned field after the post fire tree cutting. GEE could help to detect it beside it could be used by anyone like environment societies and presses include who also doesn't know programming.

1.2 Goals and Objectives

In this project we compare the most of the classification methods and algorithms to find most optimal results which will be help on future researches. The main goal of the project that allow to everyone for easily obtain and analyze satellite images with free of charge accessing to correct neutral data (data which is not manipulated by media). For all that, creating a awareness about forest fires aimed. The main goal of the website building more user friendly interface and inform the users better.

1.3 Methodology

Gee is a free of charge platform. However; users must identify their mails to google. Code send request to google servers and servers calculate data and send back to us results. There is two different input type needed for run the code. First user need to select a field as polygon geometry which he/she want to study on and second one is the date. User must enter a date before burn and after burn. Closer time interval means better results. As classification methods we use supervised, unsupervised and spectral indices based thresholding. Some supervised algorithms used on this project are "Minimum Distances", "Smile Cart", "Support Vector Machine". Unsupervised algorithms are (LCV and K means). As spectral indices based thresholding we used NDVI. Best methods for this indices researched on HERE; however our code filtered the atmosphere effect by themself so NDVI is gives very correct results just by alone. Supervised and unsupervised methods work with clustering principles but thresholding apply on pixels one by one.

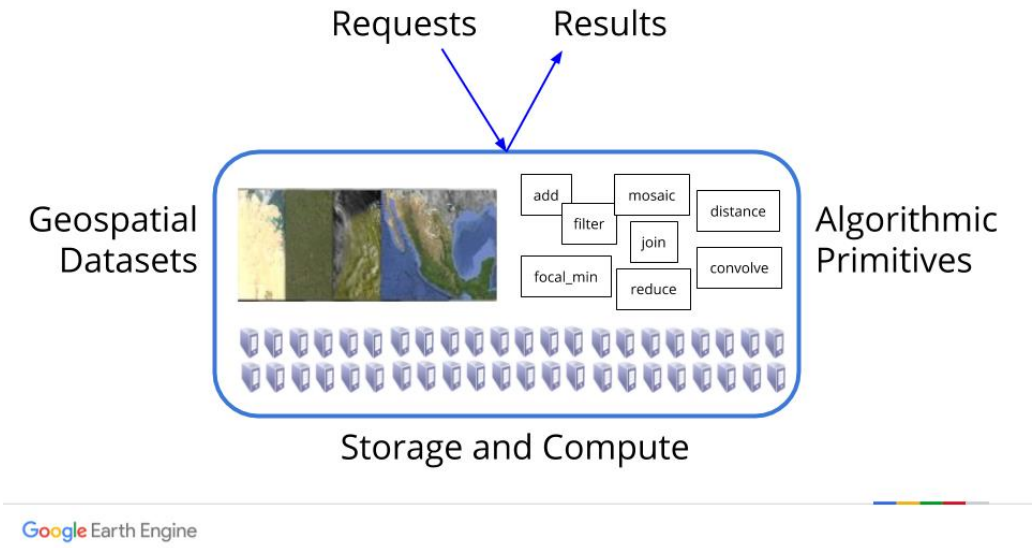


Figure 2

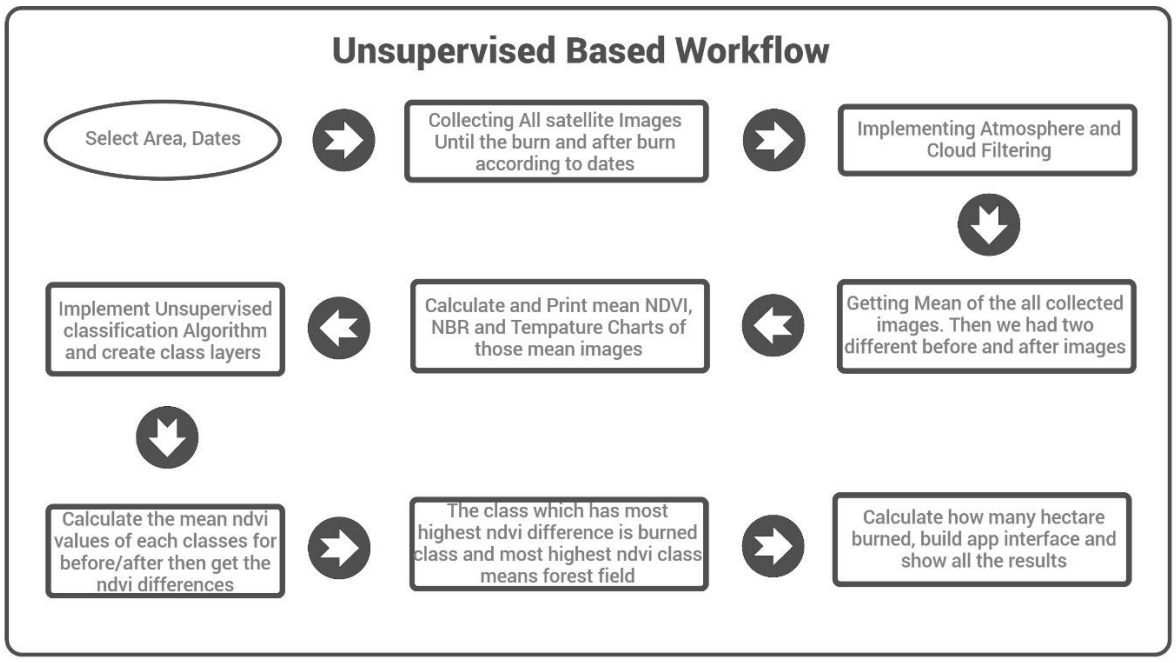


Figure 3

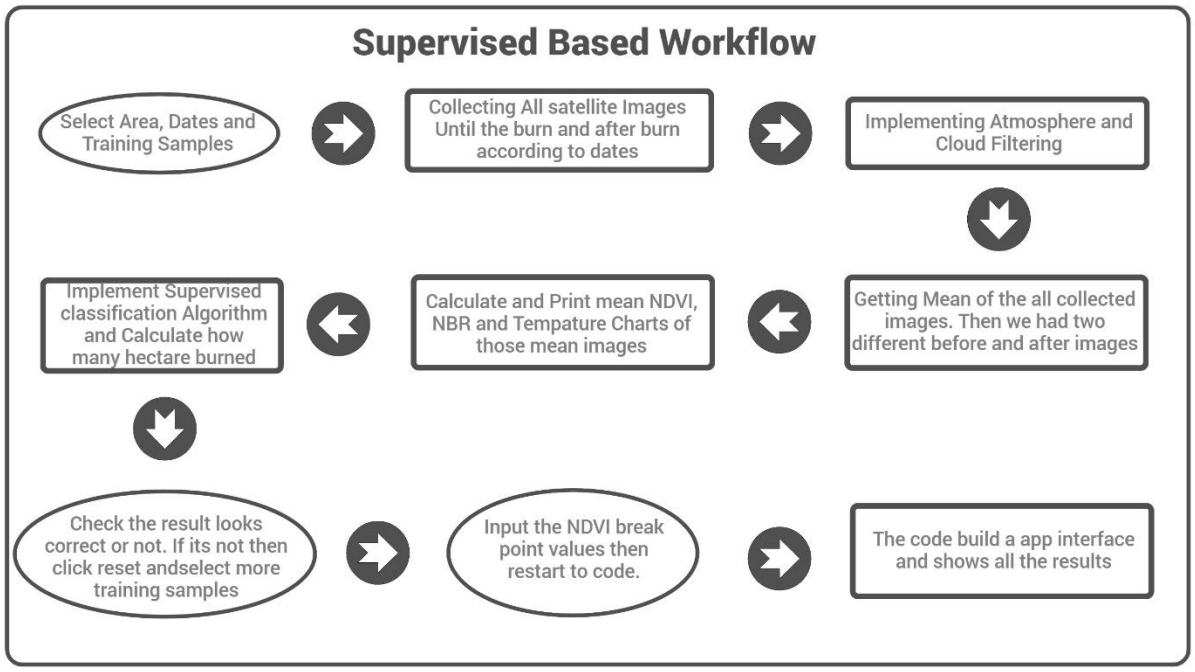


Figure 4

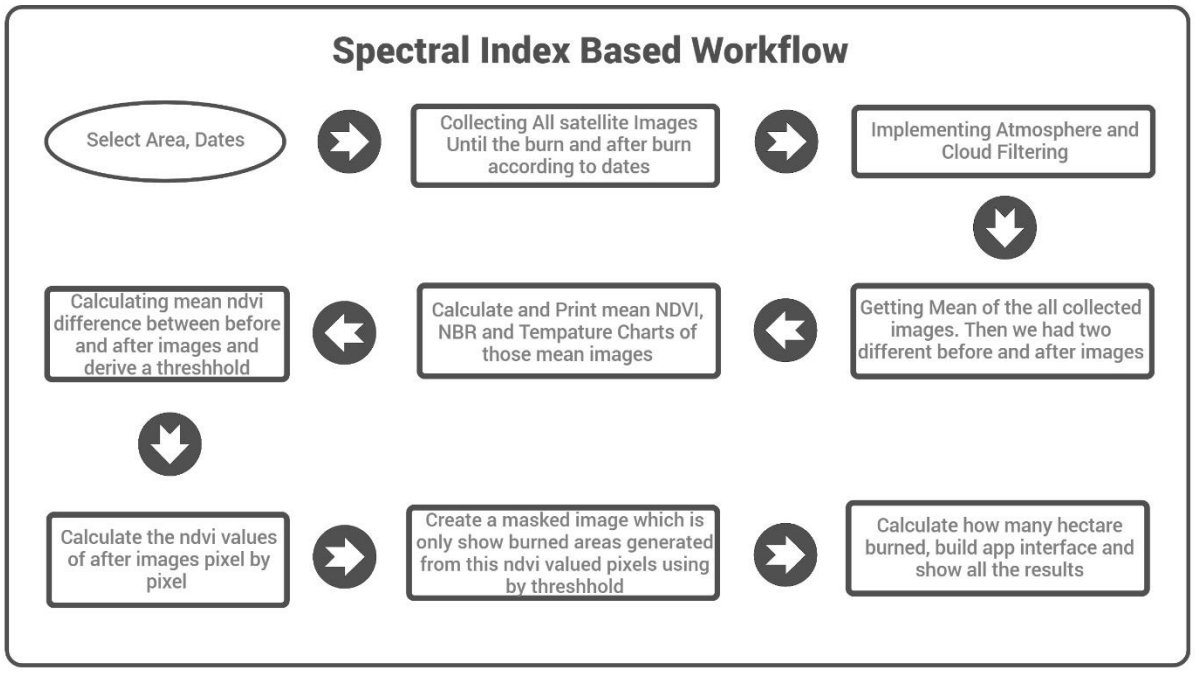


Figure 5

1.4 Structure of the Report

In backgraund part we are giving information about how we got here, searchings we made before we made a summary. We seperated the parts we take so you can observe it wisely. We told why we tried to make it simple user interface for users and according to that why we choose JavaScript in “User Input” part. Also gave information about inputs and how to select them better. In “Image Collection” part we described the layers that we use. “Chart Generation” is all about how to calculate our output graphs. “Classification Algorithms” part we gave all the algorithms we use with links with it so you can use it yourself too. About how to we define and set our clusters is in “Class Identification” part. “How many hectare burned” part is the function provided from our project. “Burn Severity Calculation” is the part we define Burn Severity and giving informations about its` levels. In “User Interface” part you can learn about functuality about our buttons and how to download your data with them. In “WebSite” part we are giving information about “InvestigatingForestFires.com”. In “Implementation and Test” part we are telling the advantages and disadvantages about methods, tests, analysis etc. In “Accuracy Assesment” we are giving information about how we measure the accuracy. “Discussion” is the part we told where we got hard to solve a problem in many circumstances. How our project`s path will be further in “Future Goals” part. In “Conclusion” part we make a summary about our project.

2 BACKGROUND

2.1 Web Part

First of all that the idea comes to us with the website, we tried to make it with the common ways. Tried a WordPress website builder for better use but that way was hard to implement our Google Earth Engine app into WordPress based page.
Google Earth Engine is very new in REST API buildings. There is no useful source or tutorial has made in REST API usage. There was only a Google`s has own sources and tutorial was able to access with it and that source was:
https://developers.google.com/earth-engine/tutorials/tutorial_api_01
After that, we tried to move on with Googles` own sources. Started a tutorial but we could not adapt our all functions into this tutorial. When you need to code something complex, there was a barrier coming to your work from Google Earth Engine`s JavaScript library. That library was only meant to run simple basic functions of Google Earth Engine. We tried this method with ReactJS and Angular but we could not process further. Then we realize there was a App builder of the Google Earth Engine itself and it is cooperate with other Google networks.
App builder had some requirements from us like some terms and using Google Sites to make the website. After making some changes in the Google Cloud you will be ready to apply your Google Earth Engine App to your Google Sites website. There is the path you need to follow when you applying your Google Earth Engine App:
https://developers.google.com/earth-engine/guides/app_engine_intro

2.2 Spectral index and Programming Language Selection

There is so many spectral indices exist like Burned Area Index (BAI), Char Soil Index (CSI), Dry-Bare Surface Index (DBSI), Normalized Burn Ratio (NBR), Normalized Difference Vegetation Index (NDVI), Optimized Soil Adjusted Vegetation Index (OSAVI), Normalized Burn Index (NBI). According to searches we find out Normalized Burn Index is best solution for direct uses because it effect minimally from atmospheric effects however; in our project we apply some atmospheric effect filters and researches shows that after the filtering there is no huge differences between spectral index errors and distortions so we decided to use most common methods like ndvi and nbr on this project.

To selecting programming language we discuss and compare our knowledge about python and javascript. After the searches about difference between this two languages we decided to use javascript because if we try to make a user friendly interface with python it will be very slow beside javascript is more common for google earth engine applications.

3 DESIGN OF THE <PROPOSED SOLUTION/PRODUCT/SYSTEM>

The most important thing about our design being user friendly. This means code workflow background could be confusing however; the code must be easy to use. Hence we choose JavaScript. The reason behind that JavaScript does not need any download and all of the works can be accessed with just one web link while python need libraries and some python versions. We think about creating a .exe with python but this will make the code much more slower so this decision eliminated. JavaScript is most common platform for GEE so this JavaScript code would help other GEE researcher and users instead of python.

3.1 User Input:

Users must input 2 dates (any date time before burn and after burn) and a study field with polygon drawing. In unsupervised classification users must select training samples (for the better results training samples should select on RGB after burn image instead of initial satellite image of google; this means training samples should select after the first run). For the second run users must input Normalized Burn Ratio (NBR) values for burn severity rate calculation and normalized vegetation index (NDVI) for spectral indices classification method. (Values can be seen on charts)

3.1.1 Dates: The dates are in 'yyyy-mm-dd' format.

```
var Start=ee.Date('2019-07-05'); //date before burn
var End=ee.Date('2019-10-05');  //date after burn
var END =End.advance(50,'day');
```

Some Date Selection Tips:

Tip - 1: If your "var End" date is to close to present day you should change the "var END's" 50. This 50 means 50 days later from the "var End's" so sometimes it exceed the present day.

Tip - 2: Trees will show different health situation with different seasons and also some exterior parameters like snow can effect the results. For example; you think the fire starts at july. Then select your time interval between june and August not like the January to December. In conclusion if Start and End dates close to burn date results will be more correct. If your region not tropical and you know the burn not start during the winter season you can apply the filter below here for winter-autumn changes.

Tip - 3: Google Earth Engine is a new technology so If you try to get old fire data like 1999-2007-2010... There would be no data catalog so the code will not work and If you try to get the newest fire datas like today's date the satellite probably not collect the data yet so you should wait a couple days or check the other satellite's datas.

Tip - 4: The season based erros which mentioned in tip-2; erros in charts can be ignorable. Initial date filter shows the all data during the whole year (first day to 365th day) but If you change this initial values in the code's image collection section then you can filter the data . For example If you write 122 and 275 as start and end dates of the filter then the code only count the dates between 1 May to 1 October so not count the winter days and in the conclusion there would be no winter data in charts.

In the example given here; 122 represent 122th day of the year which is means 1 May and 275 means 275th day of the year and its the 1 October.

.filter(ee.Filter.dayOfYear(122, 275))

Unfiltered and Filtered 2018-2020 graphic for 2019 forest fire:

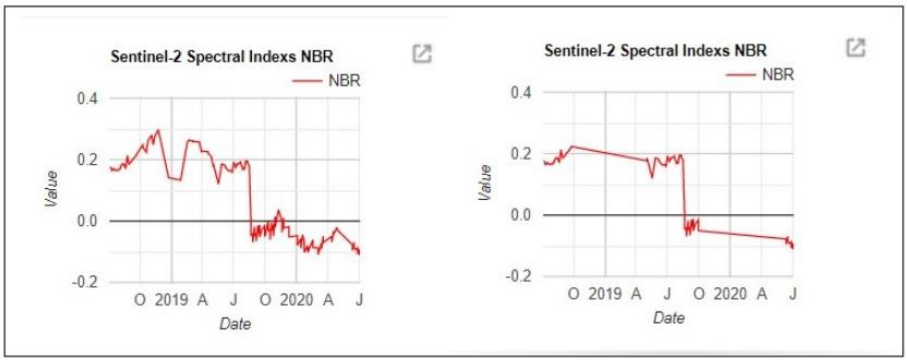


Figure 6

3.1.2 Area Selection:

- There are two different methods to identify polygons in other name areas.
- As default an example area sample identified which is cover to Izmir/Karabaglar region.

3.1.2.1 First method: If you estimate or know the corner coordinates of the polygon which is you want to work it can be directly input by manually.

Just replace those coordinates with which coordinates you want to use. You can add or remove more polygon corner. **Those coordinates must be in Geodetic Coordinate format (latitude and longitude) which is based on WGS84 Ellipsoid .**

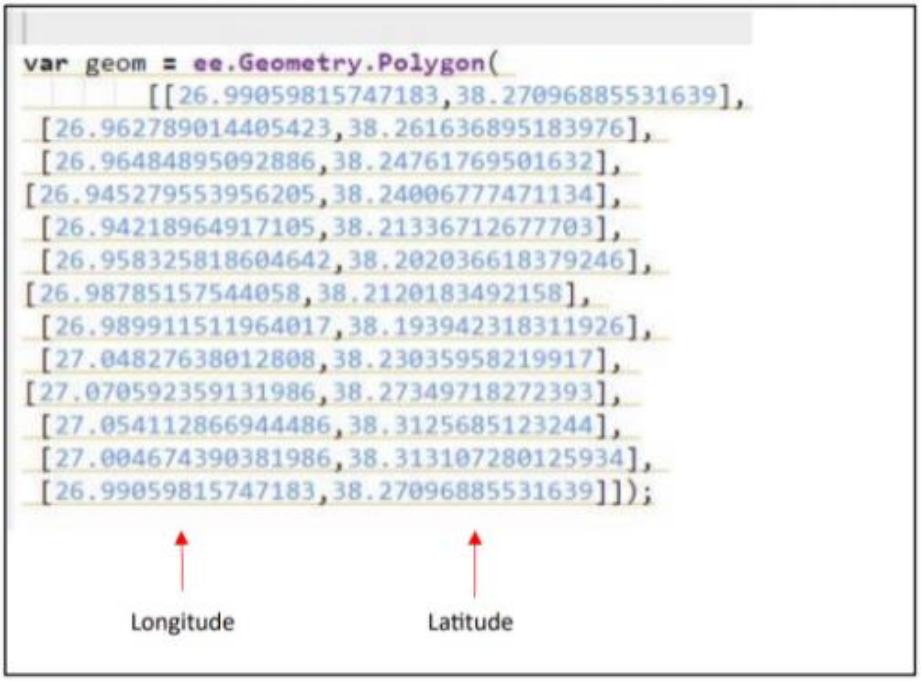


Figure 7

3.1.2.2 Second Method: If you want to select your area manually follow the steps below here. Also you can easily find your work area with search bar just typing your adresses.

Figure 8

Step - 1: Click to draw polygon button.

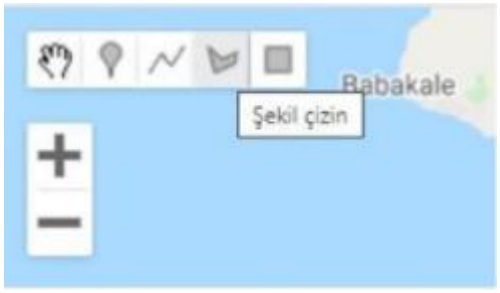


Figure 9

Step – 2: Draw your polygon on map area

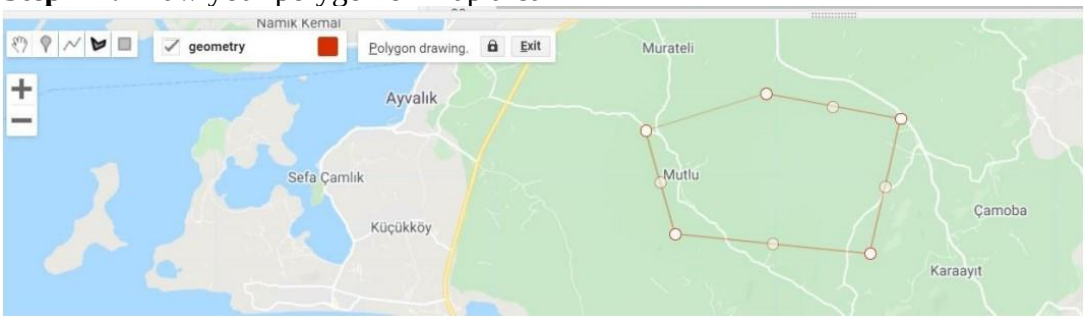


Figure 10

Also you can change the back layer of the visualization area for better visual understand. Two different options in here. Map and Satellite. If you select the satellite; the program will show the area with high quality satellite images.

Figure 11

When you finish the drawing polygon, the properties of this polygon will Show up above the script. Just don't forget delete the whole initial 'var geometry' because we identify new polygon.

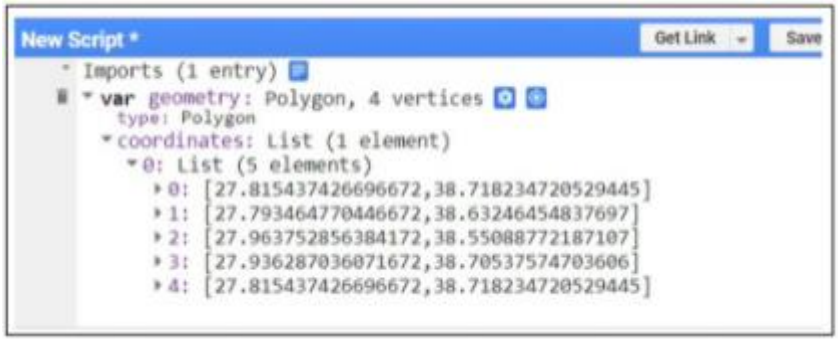


Figure 12

3.1.3 Training Samples:

The code uses supervised classification for calculting how much area burned. For this process code needs to select some training areas. We can select it with point or polygon methods. Polygon method is recommended.

Figure 13

Example of NBR Value Before Burn:

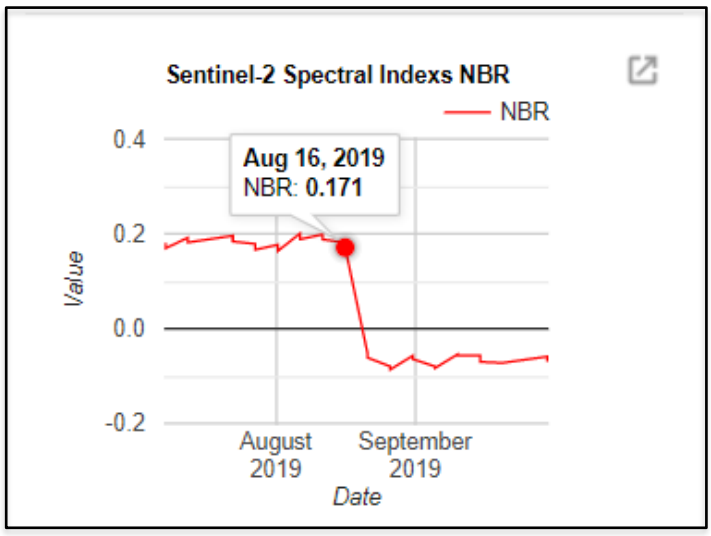


Figure 14

3.2 Image Collection:

There is many different image collection here. One of it sentinel-2 images before burn, other is sentinel-2 images after burn and the last one for temperature graphs with Modis (meteorological satellite). In this project sentinel-2 preferred instead of Landsat because sentinel-2 has higher spatial resolution. Program collect all the images took between our time interval. After the atmospheric effect filtering, we reduce the before burn and after burn images into one image with mean reducer. Finally we just have one before burn and one after burn image.

How reducer Works: Reducer is pixel based statistical function which is apply same location of different collection. This statistics can be median, min, max, mean etc.

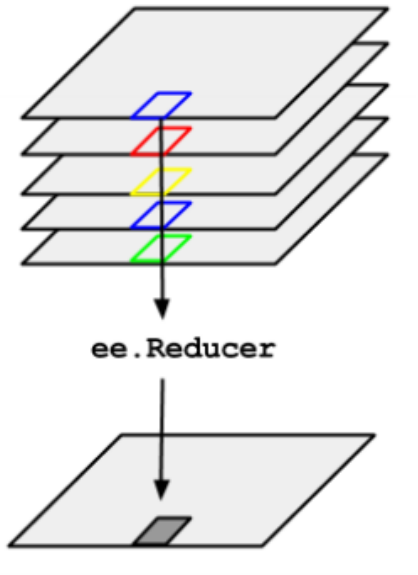


Figure 15

Unsupervised and supervised methods implement classification algorithms on Sentinel-2 RGB image of after burn which is include red (band4), green (band3), blue (band2) bands of satellite image. Sentinel-2's camera band given below here as figure.

Sentinel-2 Bands	Central Wavelength (µm)	Resolution (m)
Band 1 - Coastal aerosol	0.443	60
Band 2 - Blue	0.490	10
Band 3 - Green	0.560	10
Band 4 - Red	0.665	10
Band 5 - Vegetation Red Edge	0.705	20
Band 6 - Vegetation Red Edge	0.740	20
Band 7 - Vegetation Red Edge	0.783	20
Band 8 - NIR	0.842	10
Band 8A - Vegetation Red Edge	0.865	20
Band 9 - Water vapour	0.945	60
Band 10 - SWIR - Cirrus	1.375	60
Band 11 - SWIR	1.610	20
Band 12 - SWIR	2.190	20

Figure 16

Satellite image has cloud coverage and atmospheric effects so product must apply a atmospheric effect filter

3.3 Chart Generation:

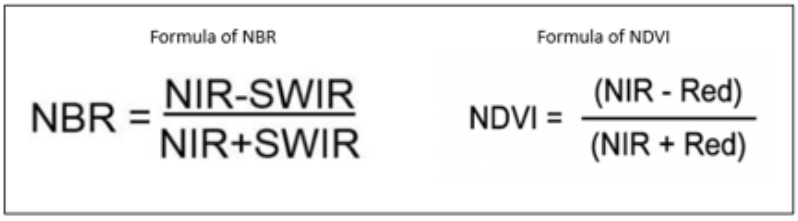


Figure 17

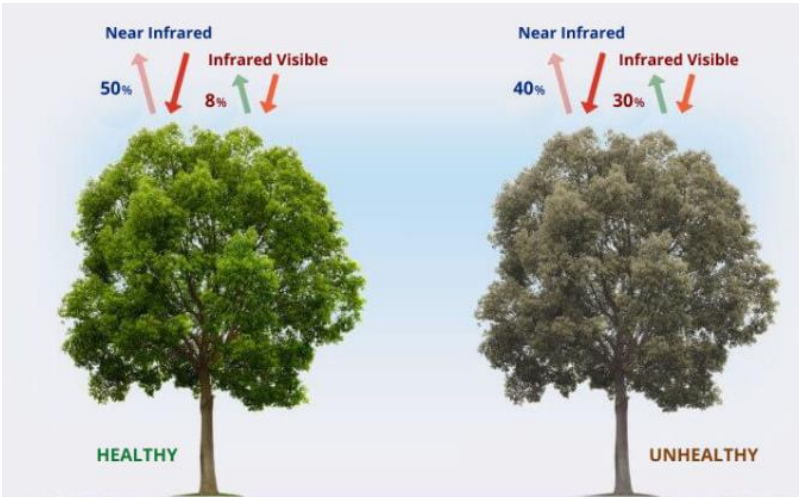


Figure 18

Charts are created with the NBR and NDVI formulas given on table? With sentinel-2 images however; temperature charts created from MODIS data.

Chart Examples:

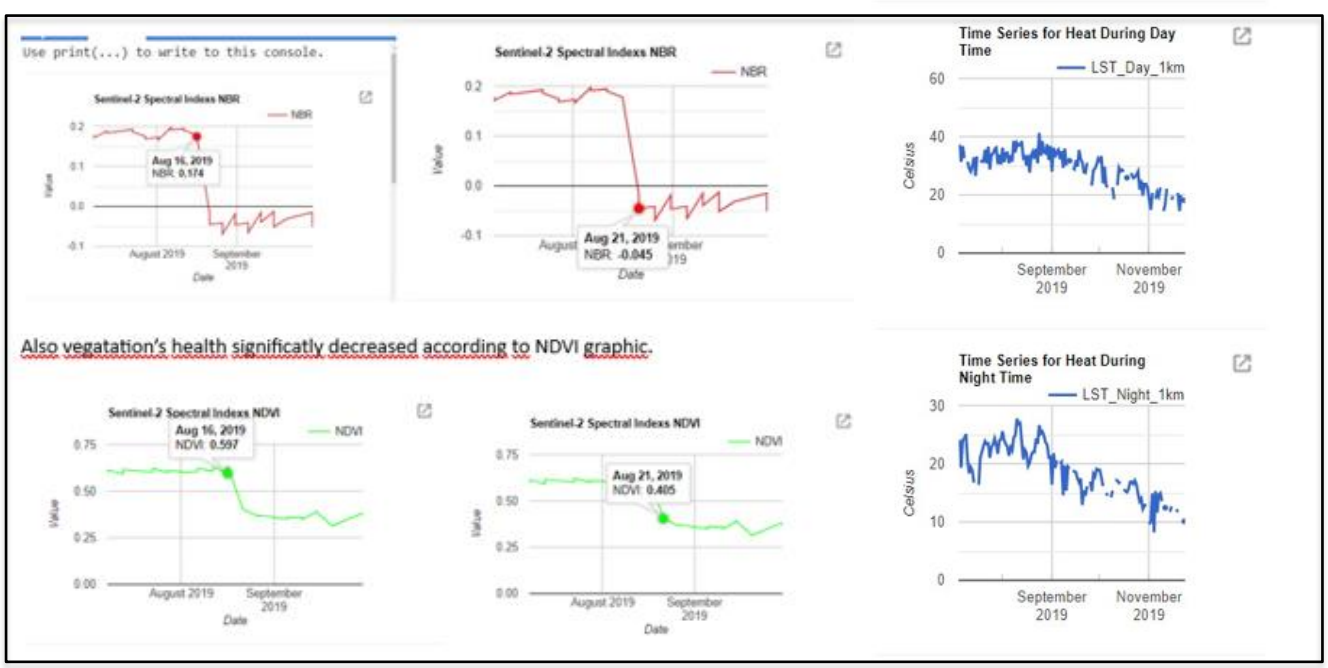


Figure 19

3.4 Classification Algorithms:

Gee have many classification algorithms functions and we applied minimum distance, smile-cart, support vector machine, lcv, k-means algorithims separetly. Projects can be acceses with link below here.

Supervised Algorithms:
MinimumDistance:
https://code.earthengine.google.com/d0c0491477c2b16ad26994abc4997614
Smile-cart:
https://code.earthengine.google.com/59d373f4757325bf8d645c395967758c
Support-Vector Machine: (Algorithm does not work steady)
https://code.earthengine.google.com/db0818173539532e7044cfbb576110a1

Unsupervised Algorthims:
LCV:
https://code.earthengine.google.com/c64499922e3f2d9665788a61b651e223
K-Means:
https://code.earthengine.google.com/b5fa3637a9c8f4d2c62e8df00586a26c

Ndvi based threshhold classification:
https://code.earthengine.google.com/b224a83aa6bb8d9ee723d392c757ec6e

3.5 Class Identification:

It is easy to identify supervised classes because its already identified in tranining sample selection However on unsupervised method code can not identify which class represent which area so with the help of mean NDVI differences of before and after burn; code could identify classes. If the differences is highest then this layer means burned class, if after burn layers mean ndvi value highest then it is a healthy vegetation. Last class identify as soil. At the other side of the coin spectral index based method can only identify burned areas. This method does not apply any clustering algorithms but thresholding each burned pixel one by one so only burned areas detected but this method as high accuracy which will be explain in discussion section. The biggest problem is in here finding a threshold value for thresholding. The code calculate the mean NDVI values of whole before and after image then subtract after image from before. Also subtract the standart deviation differences from this result so this threshhold will give statistically %95 confidence interval to us.

3.6 How many hectare burned:

How many hectare burned calculation work with same principles on all methods. Firstly, how code count how many pixels in burned area layer. Sentinel-2 sattellites GSD size 10m so one pixel represent 10m*10m area. Amount of pixel * pixel area gives us how many area burned in unit of meter-square. Later that code convert meter-square to hectare.

Unit transformation ratio:

```
// Calculating Burned area
var pixel_area=10*10;
var hectar_m2_ratio=10000
```

An example of Results:

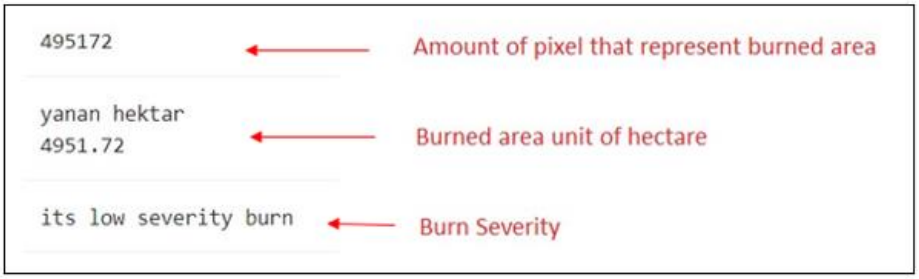


Figure 20

3.7 Burn Severity Calculation:

Burn severity calculated from NBR value differences of before and after burn. The scale of the burn severity rates given below here on figure.

Severity Level	dNBR Range (scaled by 10^3)	dNBR Range (not scaled)
Enhanced Regrowth, high (post-fire)	-500 to -251	-0.500 to -0.251
Enhanced Regrowth, low (post-fire)	-250 to -101	-0.250 to -0.101
Unburned	-100 to +99	-0.100 to +0.99
Low Severity	+100 to +269	+0.100 to +0.269
Moderate-low Severity	+270 to +439	+0.270 to +0.439
Miderate-high Severity	+440 to +659	+0.440 to +0.659
High Severity	+660 to +1300	+0.660 to +1.300

Figure 21

3.8 User Interface:

Lastly; a user face created in map area which is seperate the window two pieces. Left window shows before burn visual datas and right one shows after burn. Also there is some extra buttons added. Those button functionalities are turning back to initial interface and centering map.

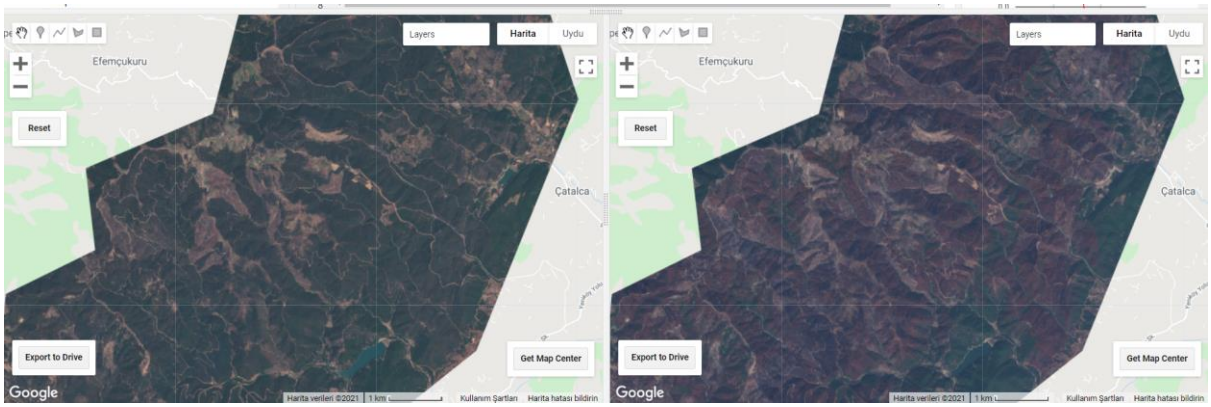


Figure 22

3.9 Saving and downloading datas:

Exporting before and after RGB images quite simple. If you want before the burn images then click to “Export to Drive” button of left side map. If you want to after images then click to right side’s button. The difference between getting screenshot and pressing export; Exported images will be geotiff format which means images also have coordinate data.

Figure 23

After the clicking button check the Task Section and clik to RUN.

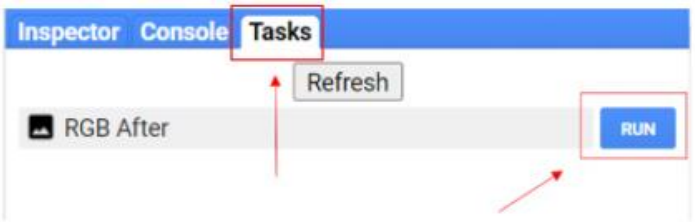
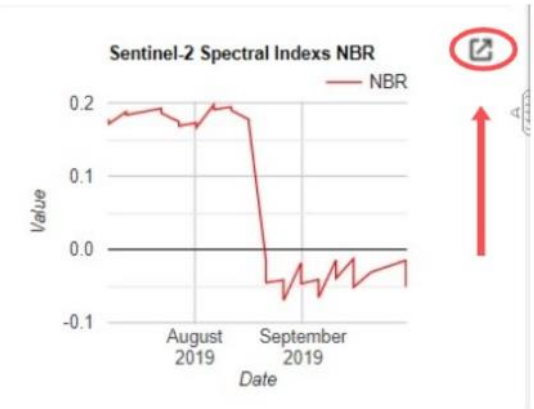


Figure 24

Then click to Run again. This will export the image into your Google Drive adress.

Note: If the code tells, you passed the pixel limit you can select smaller area or increase the scale. Scale also is GSD. **That will decrease your image’s resolution.**

If you want to export charts click to upper-right button of the chart.



Then select to which format you want to download.

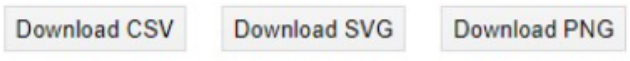


Figure 25

If you wanted to save your project click get link it will save all your work with everything included code, sample, geometry changes to a web site link. If you click to save it will saved to your account and can be accessed from left side of the layout with any computer which has internet connection.

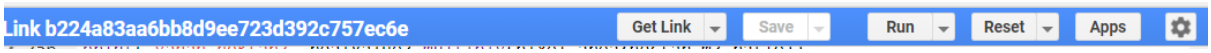


Figure 26

Link:

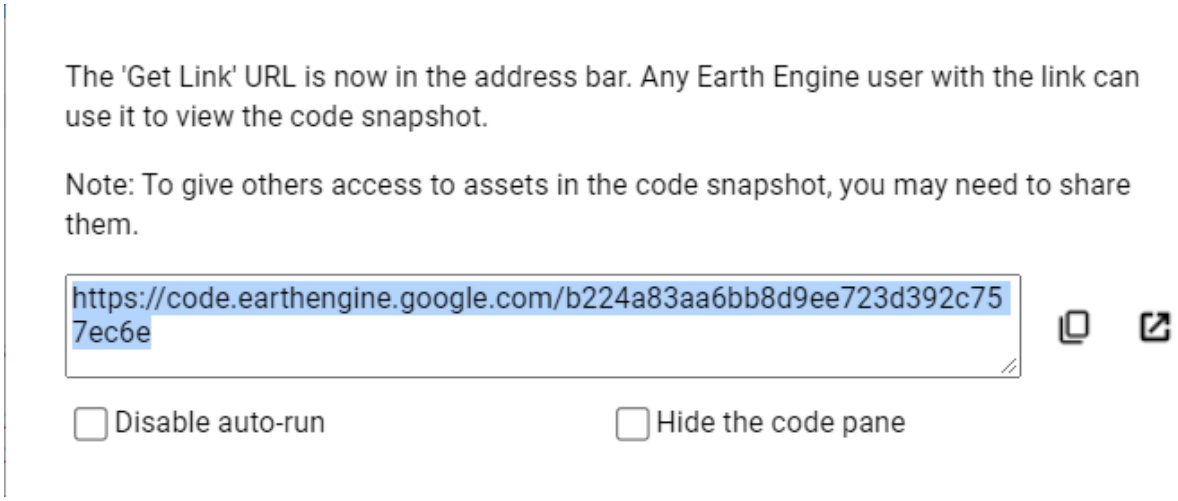


Figure 27

Saving on gmail Account:

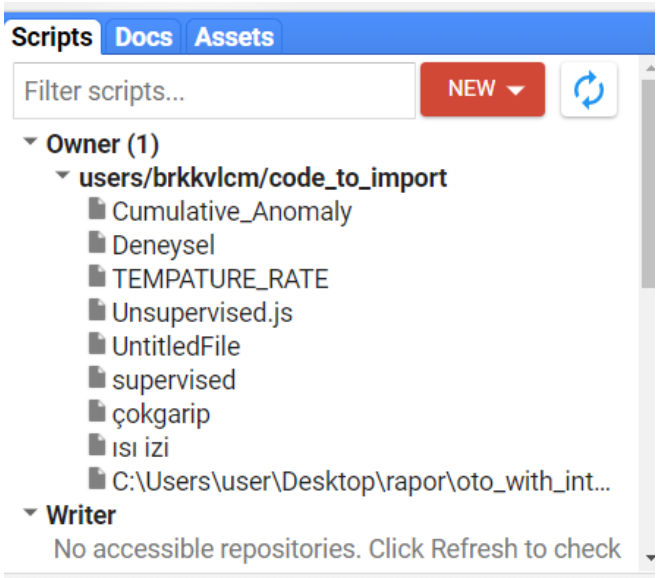


Figure 28

3.10 WebSite:

It is impossible to think The World without an internet these days. Our project is not even using our own computers or our servers. It only takes our script that we coded and polygons that our users` has provided. With all of the usage of internet, there need to be website of our project.
We made this website for better usage. In Google Earth Engine user interface has a confusion about everywhere. There is a script that we coded is shown upper, there is a functions screen left side we never use and right side of the screen there is a console that we are using it only show outputs. This is a very confusing screen for user. We think our user can be only coder , but our user can be the person that he/she never used a computer for coding applications. We need to think of them because of our project. That is why we made this website.
Our website has three parts : Main , How to Use and About Us.
In main part is the part that we applied our Google Earth Engine App and you can use it on this page. There is also have information about Google Earth Engine, some information and video about our project`s content , "what is our output`s meaning?" and some information about wildfires too.
In How to Use part we tried to show how to process is going for people does not know how to use.
And about us part we are giving information about ourselves.
This website has made with Google Sites and publishing with Google Domain because of the Google Earth Engine is still in developing part and its API is not efficient properly. For a more productive result we used Google services.
Web Link: www.InvestigatingForestFires.com

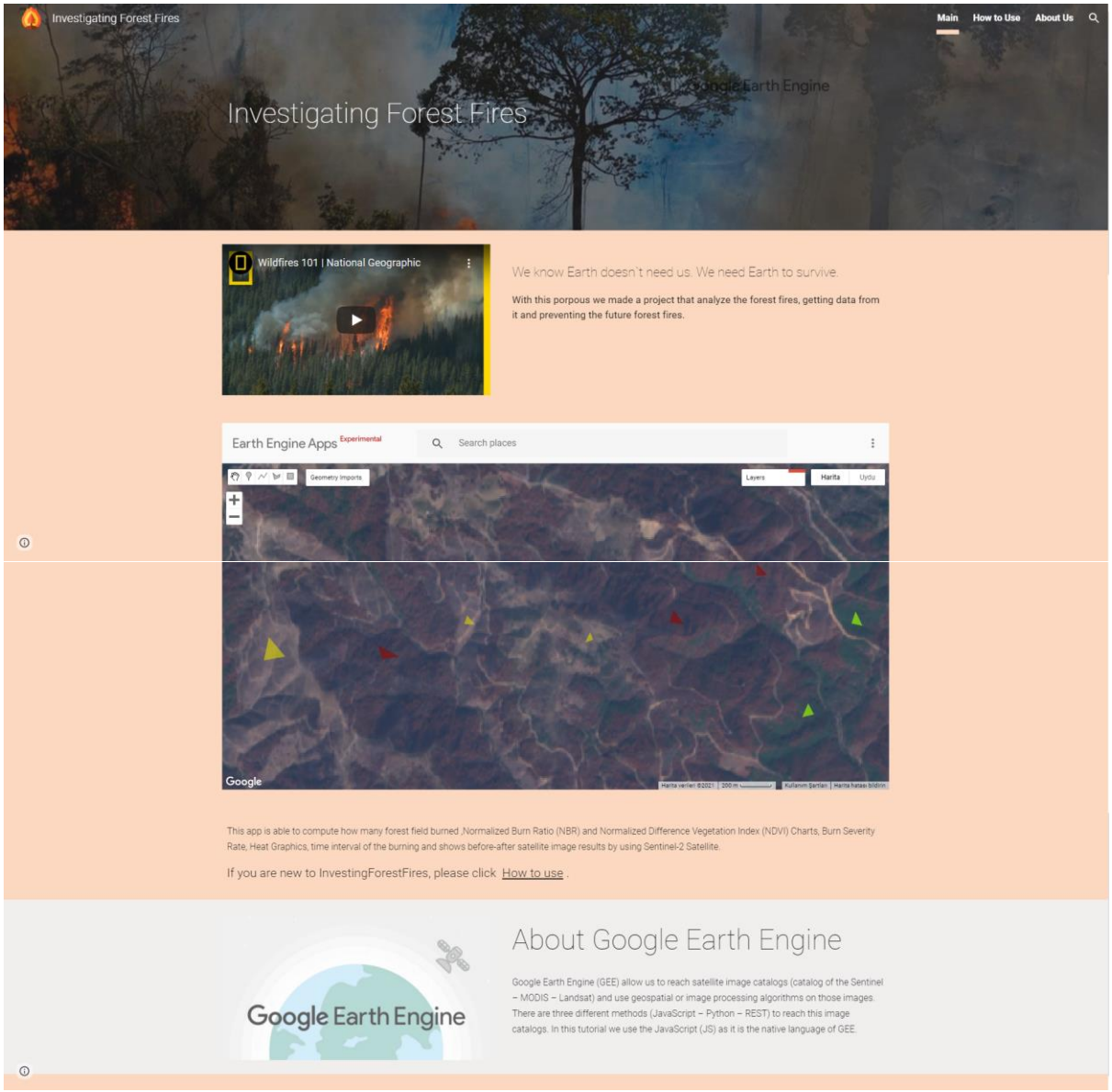

Investigating Forest Fires
Main
How to Use
About Us
Investigating Forest Fires
Wildfires 101 | National Geographic
We know Earth doesn't need us. We need Earth to survive.
With this porpous we made a project that analyze the forest fires, getting data from it and preventing the future forest fires.
Earth Engine Apps
Experimental
Search places
Geometry Imports
Layers
Harita
Uydu
Google
This app is able to compute how many forest field burned ,Normalized Burn Ratio (NBR) and Normalized Difference Vegetation Index (NDVI) Charts, Burn Severity Rate, Heat Graphics, time interval of the burning and shows before-after satellite image results by using Sentinel-2 Satellite.
If you are new to InvestingForestFires, please click How to use .
Google Earth Engine
About Google Earth Engine
Google Earth Engine (GEE) allow us to reach satellite image catalogs (catalog of the Sentinel – MODIS – Landsat) and use geospatial or image processing algorithms on those images. There are three different methods (JavaScript – Python – REST) to reach this image catalogs. In this tutorial we use the JavaScript (JS) as it is the native language of GEE.

Figure 29

4 IMPLEMENTATION AND TESTS (OR APPLICATION AND RESULTS)

4.1 Advantages and Disadvantages of Methods:

Methods would effect the advantages however algorithms does not effect it well so this section separate on methods not algorithms.

4.1.1 Supervised Method:

Advantages:

- Theoretically better results then unsupervised methods.
- Results are quite well on huge working fields with non burned areas
- Fast implementation time

Disadvantages:

- Need to select training samples which could be challenging.
- Result quality depends on training samples

4.1.2 Unsupervised Method:

Advantages:

- Most easy to use according to other methods.
- All automated. Minimum user input needed
- The results just depends on algorithm not the human so results will be more standardized

Disadvantages:

- Very slow. One run took almost 3-6 minutes.
- Hard to debug and develop due to run time.
- Very bad results on huge working fields with non burned areas
- Theoretically worst results according to other methods.

4.1.3 Spectral Index Based Thresholding:

Advantages:

- Theoretically most correct results
- Best results on huge working fields with non burned areas
- Fast implementation time

Disadvantages:

- Finding a good threshold value will be challenging.
- Result quality depends on threshold value

4.2 General Conclusion about advantages and disadvantages:

Run Time:

- Unsupervised > Supervised ~= Spectral index based threshholding

Theoretically Accuracy:

- Spectral index based thresholding > Supervised > Unsupervised

Difficulty of Use:

- Spectral index based thresholding > Supervised > Unsupervised

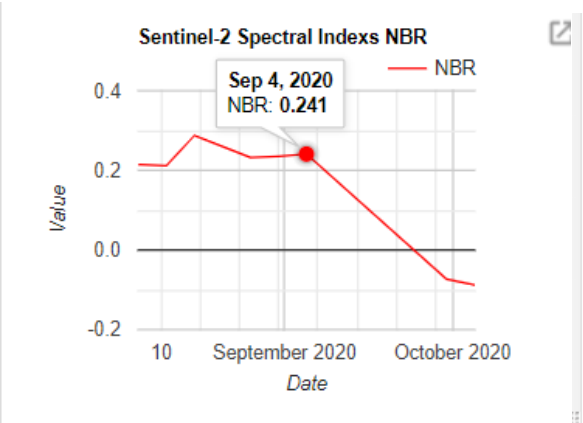
4.3 Tests and Analysis:

4.3.1 Tests from News:

The data when forest fire started and ended compared with news. Some results shown below here.

USA – Oregon Fire: According to our project, forest fires started at 4th September 2020 and ended on 29th September 2020. According to news this fire burned a huge area during September of 2020.

Code Link: https://code.earthengine.google.com/1472dd411ca11d47c4b332fe467f1bfe
News Link: https://www.bbc.com/news/world-us-canada-54180049



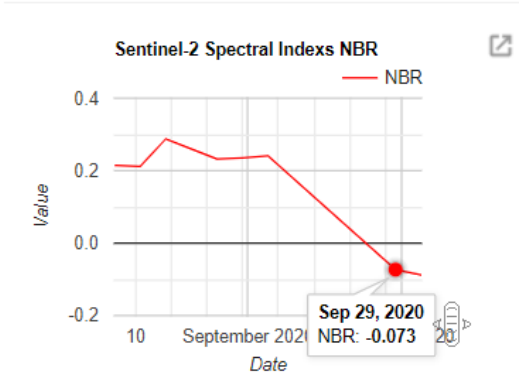


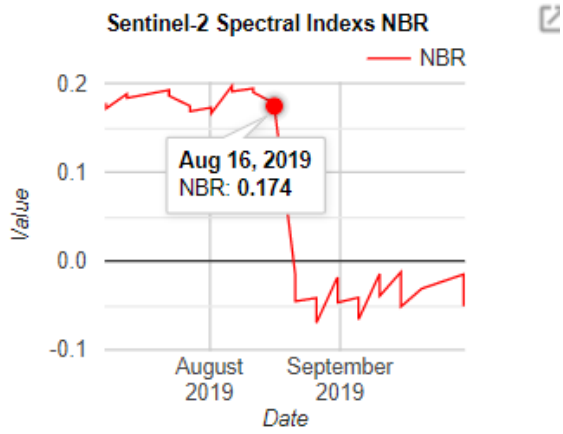
Figure 30

Figure 31

Figure 32

Turkey – İzmir / Karabağlar: According to our project forest fires start at 16th August 2019 and continued until 21st August of 2019. According to news this fire continued between 17th August 2019 to 20th August 2019.

News Link: https://www.trthaber.com/haber/turkiye/izmir-ve-mugladaki-orman-yanginlari-uydudan-goruntulendi-427564.html



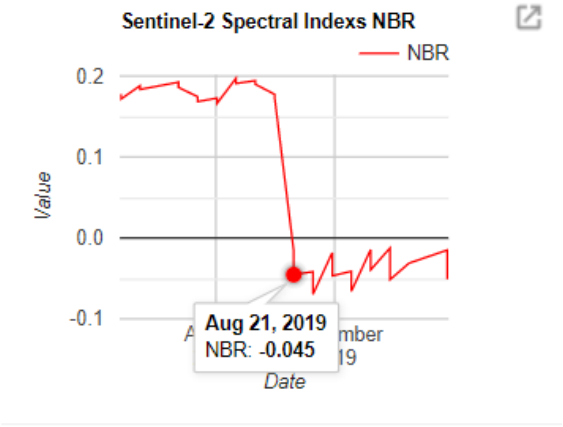


Figure 33

Figure 34

Figure 35

4.3.2 Visual Tests:

4.3.2.1 Errors due to training sample selection:

Sometimes supervised results can be bad especially with bad training sample selection. There could be several reason for bad sample selection. One of it lack of samples and other is wrong selection. If classification does not look correct then more samples should be selected. Some examples given below here.

Original Burned Area:

Figure 36

Only Burned Area Layer: (Good Results)

Amount of training samples: tree: 6, burned: 6, soil: 7

It is clearly seen burned areas mostly covered but healthy vegetation and soil areas not covered.

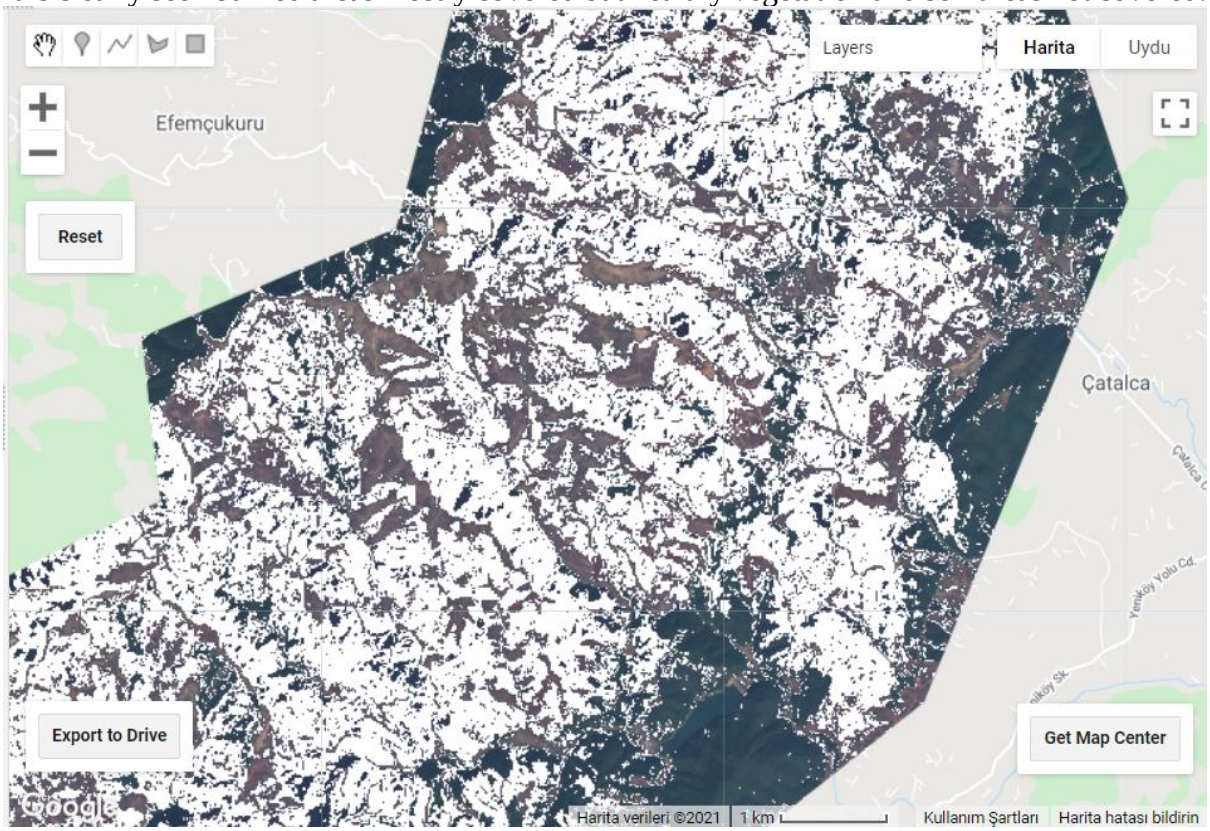


Figure 37

Only Burned Area Layer: (Bad Result)

Amount of training samples: tree: 2, burned: 2 soil: 2

As table given below here, we can understand some dark burned areas not counted as burned because of lack of training samples. In this situation more samples should select carefully.

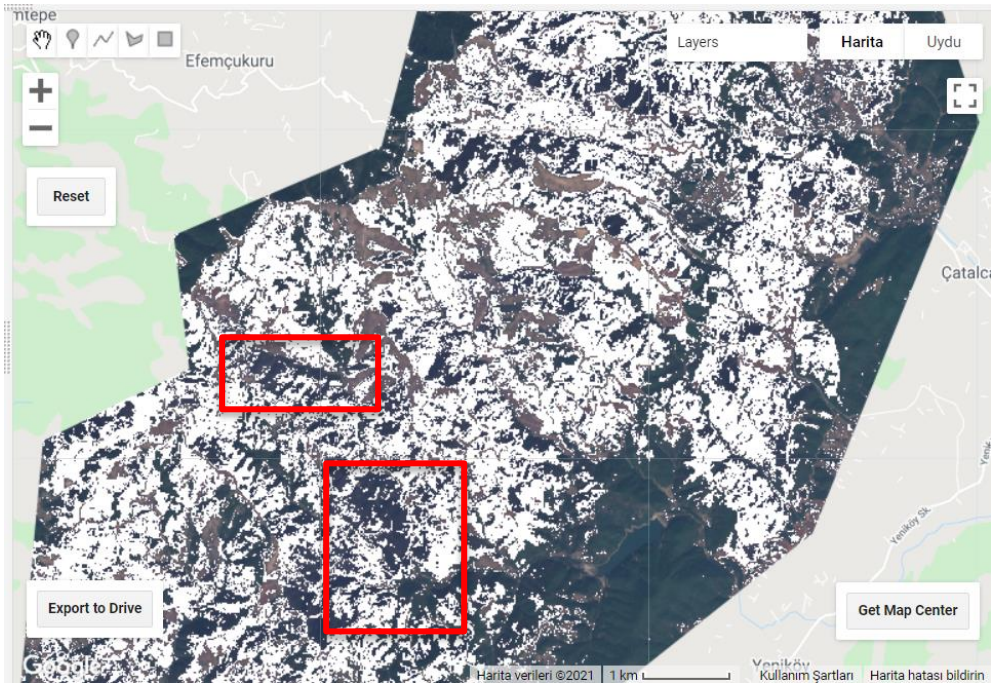


Figure 38

4.3.2.2 Errors due to study field selection:

Additionally If working area has very huge field that not contain burned field then unsupervised algorithms will give terrible results because of clustering algorithm. Even supervised algorithm can calculate the burned areas wrong with this situation. However, spectral index based thresholding method will give correct results because it not apply any clustering algorithm.

NDVI thresholding method with huge study field: It is clearly seen the other parts which is not burned not counted as burned.

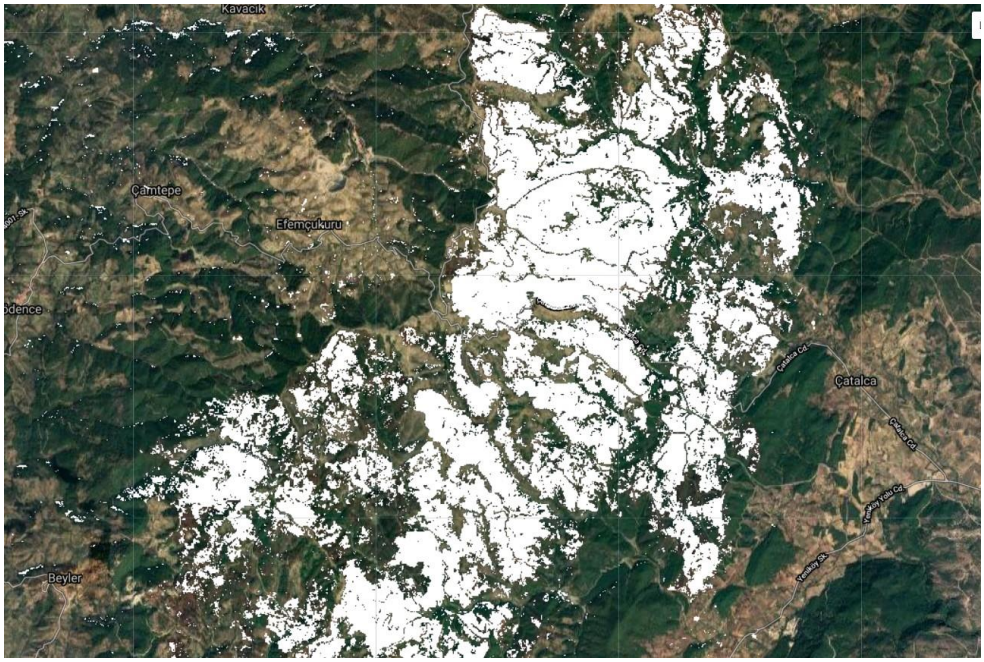


Figure 39

Supervised method with huge study field: Some clustering errors appeared.

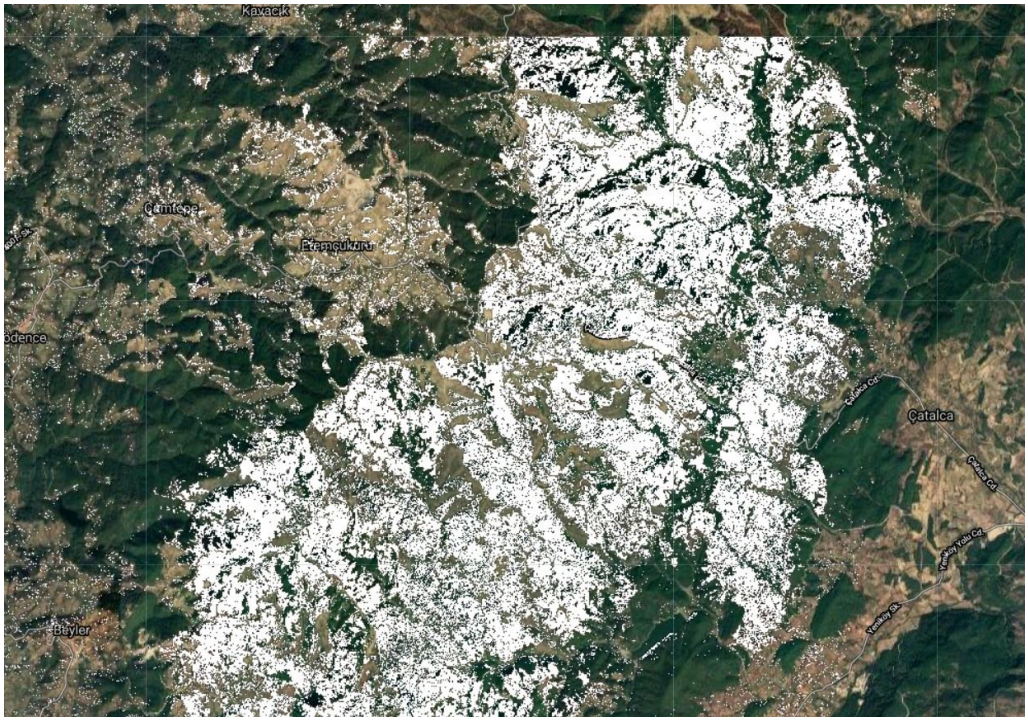


Figure 40

Unsupervised method with huge study field: Tremendous clustering errors appeared.

This healthy vegetation areas counted as burned and this situation cause a huge error at amount of burned area calculation

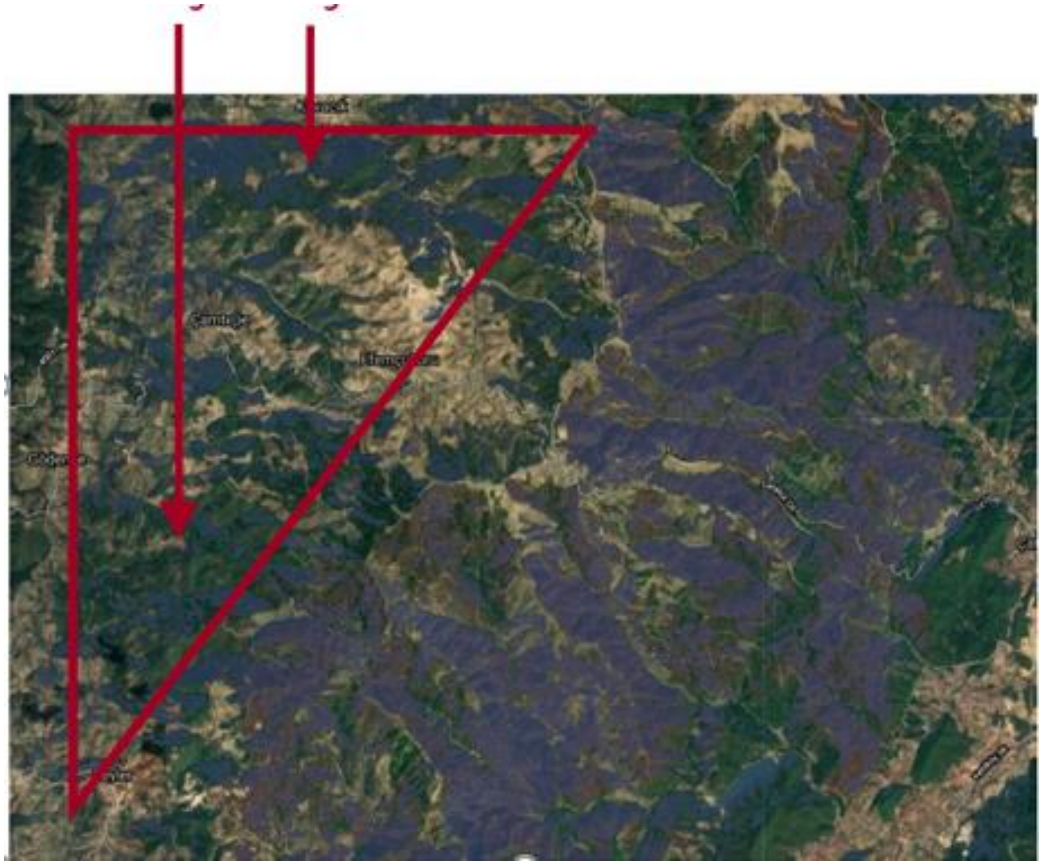


Figure 41

Working field's size effects comparison with numerical values:
If the specific field selection and huge field selection's how many hectare burned results close to each other then method work well on huge field selections.

Burned Area (hectare) calculated with unsupervised method on huge study field:
6335.02 hectare (K-means)
https://code.earthengine.google.com/1d85921cb40eb1af56218929ae8f4720
Burned Area (hectare) calculated with unsupervised method on specific selected area:
4983.59 hectare (K-means)
https://code.earthengine.google.com/b5fa3637a9c8f4d2c62e8df00586a26c

Burned Area in unit of hectare calculated with supervised method on huge study field:
5907.3 hectare (smileCart)
https://code.earthengine.google.com/736262e7e219eaec69bd37f8d8116352
Burned Area (hectare) calculated with supervised method on specific selected area:
5144.55 hectare (smileCart)
https://code.earthengine.google.com/59d373f4757325bf8d645c395967758c

Burned Area in unit of hectare calculated with thresholding method on huge study field:
4653.26 hectare (ndvi)
https://code.earthengine.google.com/b224a83aa6bb8d9ee723d392c757ec6e
Burned Area (hectare) calculated with thresholding method on specific selected area:
4707.62 hectare (ndvi)
https://code.earthengine.google.com/d57c72e261b49891cff8d87f2e08de50

In a brief conclusion, Spectral index based thresholding method does not much effected from study field's size but unsupervised method very effected from it.

4.3.3 Statistical Tests:

There is two main idea about statistically tests. One of it using accuracy assessment methods for supervised classification and the other one is assume the spectral thresholding true and compare others with it. Because theoretically this thresholding has highest accuracy rate and also it based on some statistic rules like confidence interval. 4983.59 is %7 higher than 4653.26 and 5144.55 %10.5 higher than 4653.26.

There is exceptional situations about algorithms. On support vector machine (supervised algorithm) and LCV (unsupervised algorithm) a small change on study field could change the whole clustering results so in this analysis support vector machine and LCV not mentioned due to its unsteady results. Just a little change on field burned area values jump to 6960.87 from 3446.04 for LCV and 4512.24 to 8448.09. Theoretically this is impossible so those algorithms not reliable.

In a brief conclusion, K-means, smile-cart and minimum distance methods more reliable then LCV and support vector machine algorithms.

Accuracy assesment

Confusion matrix

To assess the accuracy of an image classification, generally a confusion matrix is created. In a confusion matrix, classification results are compared to additional ground truth information. The strength of a confusion matrix is that it identifies the nature of the classification errors (how accurately you classify the pixels), as well as their quantities.We are using the google earth engine method which is "ee.ConfusionMatrix, it creates a confusion matrix.

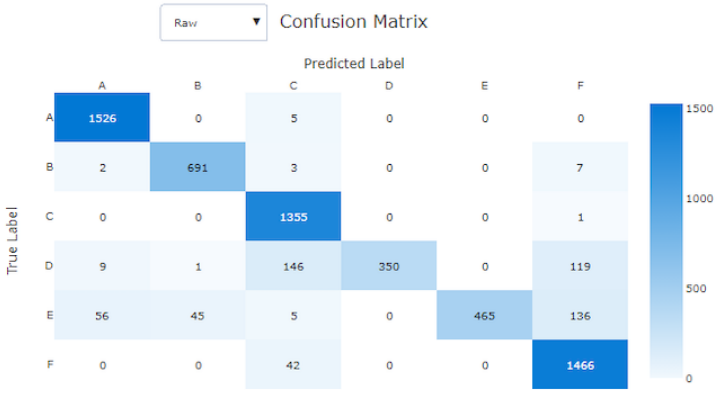


Figure 42

We are trying to find accuracy of the image classification we already done.The Classify operation performs a multi-spectral image classification according to training pixels in a sample set . A sample set stores training pixels for supervised classification. Prior to an image classification, sample pixels or training pixels have to be selected in a sample set. To create a sample set, first a map list and a domain have to be specified. Then, with sampling, assign class names to groups of pixels that are supposed to represent a known feature on the ground(like water,burned areas,healty vegetation) and that have similar spectral values in the maps in the map list.

Preparation of confusion matrix

To calculate confusion matrix we need training pixels that we use for classification (its comes from "classifier"in our code and it contains training values which we selected by our own hands over the google earth engine interface).More about details (ref1).We also need the test values (validation:it gives ground truth information) to compare with the classification results.

Why do we need 2 different sample data ?

The results of confusion matrix highly depend on the selection of ground truth / test set pixels. You may find yourself in a situation of the chicken-egg problem with your sample set, the classification result and your test set. On the one hand, you want to have as many correct sample set pixels as possible so that the classification will be okey; on the other hand, you also need to have an ample number of correct ground truth pixels for the test set to be able to assess the accuracy and reliability of your classification. Using the same data for both the sample set and the test set will produce far too optimistic figures in the confusion matrix, Inspect the output to see that the overall accuracy estimated from the training data is much higher than the validation data. The accuracy estimated from training data is an overestimate because the random forest is “fit” to the training data. The expected accuracy on unknown data is lower, as indicated by the estimate from the validation data. It is strongly advised that the test set raster map does not contain the same pixels as the sample set raster map from the training phase. It is mathematically correct to use half of your ground truth data for the sample set and the other half for the test set.(ref2).So we have to divide our training samples into 2 parts, training and validation.in our code we select randomly 70% training, 30% testing areas and reclassify them.

```
//canfusion matrix for overall accuracy
var sample_rc = training.randomColumn('rand')
var training2 = sample_rc.filter(ee.Filter.lt('rand',0.7))// Roughly 70% training, 30% testing.
var validation = sample_rc.filter(ee.Filter.gte('rand',0.7))
```

Interpretation of confusion matrix

Consider the following example of a confusion matrix:

	classsified water	classsified soil	classsified tree	classsified burned	producers accuracy
accually water	0	0	0	0	0
accually soil	0	425	0	6	0.986
accually tree	0	0	651	0	1
accually burned	0	8	1	359	0.975
consumer accuracy	0	0.981	0.998	0.983	

Figure 43

* Axis 1 (the rows) of the matrix correspond to the actual values or classes in the ground truth map (or test set).
*Axis 0 (the columns) correspond to the predicted values or classes in the classification result.
* The diagonal elements in the matrix represent the number of correctly classified pixels of each class, i.e. the number of ground truth pixels with a certain class name that actually obtained the same class name during classification. In the example above, 425 pixels of 'soil' in the test set were correctly classified as 'soil' in the classified image.
* The *off-diagonal elements* represent misclassified pixels or the classification errors, i.e. the number of ground truth pixels that ended up in another class during classification. In the example above, 6 pixels of 'soil' in the test set were classified as 'burned areas' in the classified image.
Note: The smaller the number of off-diagonal elements, the better model of the confusion matrix we have.This figure is a good model for confusion matrix because test and training pixels selectet from same training samples.

*Off-diagonal row elements represent ground truth pixels of a certain class which were excluded from that class during classification. Such errors are also known as errors of omission or exclusion. For example, 8 ground truth pixels of 'burned areas' were excluded from the 'burned areas' class in the classification and ended up in the 'soil' class.

* Off-diagonal column elements represent ground truth pixels of other classes that were included in a certain classification class. Such errors are also known as errors of commission or inclusion. For example, 1 ground truth pixels of 'burned areas' were included in the 'tree' class by the classification.

Now we have the confusion matrix and we can find the accuracy.We are using the google earth engine code function called "confusion Matrix.accuracy()" and it calculates the accuracy.(ref2)
Accuracy
Accuracy (also known as producer's accuracy): The figures in column Accuracy (ACC) present the accuracy of your classification: it is the fraction of correctly classified pixels with regard to all pixels of that ground truth class. For each class of ground truth pixels (row), the number of

correctly classified pixels is divided by the total number of ground truth or test pixels of that class. For example, for the 'soil' class, the accuracy is 425/431 = 0.986 meaning that approximately 98% of the 'soil' ground truth pixels also appear as 'soil' pixels in the classified image.

```
producer accuracy                                                    JSON

List (4 elements)                                                    JSON
  0: [0]
  1: [0.9860788863109049]
  2: [1]
  3: [0.9755434782608695]
```

Reliability (also known as user's (consumer) accuracy): The figures in row *Reliability* (REL) present the reliability of classes in the classified image: it is the fraction of correctly classified pixels with regard to all pixels classified as this class in the classified image. For each class in the classified image (column), the number of correctly classified pixels is divided by the total number of pixels which were classified as this class. For example, for the 'soil' class, the reliability is 425/433 = 0.981 meaning that approximately 98% of the 'forest' pixels in the classified image actually represent 'forest' on the ground.

```
consumer accuracy                                                    JSON

List (1 element)                                                     JSON
  0: List (4 elements)
     0: 0
     1: 0.9815242494226328
     2: 0.9984662576687117
     3: 0.9835616438356164
```

Overall Accuracy

The *overall accuracy* is calculated as the total number of correctly classified pixels (diagonal elements) divided by the total number of test pixels.425+651+359=1435(diagonal elements), 1435+6+8+1=1450.Overall accuracy=1435/1450=0,98965517.

```
overall accuracy                                                     JSON

0.9896551724137931
```

Tests and deductions
TEST-1
Condition-1 =Using the same training sample for test and training value
Condition-2= dividing into 2 part like %70 training %30 testing.
! classifier is "smileRandomForest(10)"
Google earth engine code links :
- https://code.earthengine.google.com/e09af0017bd50255316600416a36958c
Results
Condition-1 results:

```
Confusion Matrix                                                    JSON

training and tests values selected from same pixels                 JSON

▾ [[0,0,0,0],[0,425,0,6],[0,0,651,0],[0,8,1,359]]                   JSON
  ▸ 0: [0,0,0,0]
  ▸ 1: [0,425,0,6]
  ▸ 2: [0,0,651,0]
  ▸ 3: [0,8,1,359]

overall accuracy                                                    JSON

0.9896551724137931
```

Condition-2 results:

```
Confusion Matrix %70 training %30 testing                           JSON
▾ [[0,0,0,0],[0,100,0,14],[0,0,198,0],[0,13,1,103]]                 JSON
  ▸ 0: [0,0,0,0]
  ▸ 1: [0,100,0,14]
  ▸ 2: [0,0,198,0]
  ▸ 3: [0,13,1,103]

Overall accuracy                                                    JSON
0.9347319347319347
```

Note: We decided to use contition2 in rest of the our code because the chicken-egg problem (you can find it in Why do we need 2 different sample data ? part) and first condition is too optimistic.

****

TEST2: Effect of different classifiers on results

Condition-1: Classifier.smileCart()

- https://code.earthengine.google.com/ca2dfd83b9a33dbdd5067512778640f1

Condition-2: Classifier.smileRandomForest(10)

- https://code.earthengine.google.com/a7f80b296db48e3feede0b66c0375c97

Condition-3: Classifier.minimumDistance()

-https://code.earthengine.google.com/41a0eaec8cf45bf5e8344c7a2df2ff0b

Results

Condition-1 results

Confusion Matrix %70 training %30 testing JSON

[[0,0,0,0],[0,101,0,13],[0,5,53,3],[0,12,0,88]] JSON

- 0: [0,0,0,0]
- 1: [0,101,0,13]
- 2: [0,5,53,3]
- 3: [0,12,0,88]

Overall accuracy JSON

0.8986013986013986

burned area with unit of hectare JSON

5144.55

Condition-2 results

Confusion Matrix %70 training %30 testing JSON

[[0,0,0,0],[0,119,0,10],[0,1,61,1],[0,16,0,10… JSON

- 0: [0,0,0,0]
- 1: [0,119,0,10]
- 2: [0,1,61,1]
- 3: [0,16,0,102]

Overall accuracy JSON

0.9096774193548387

burned area with unit of hectare JSON

5834.57

Condition-3 results

Confusion Matrix %70 training %30 tes… JSON

[[0,0,0,0],[0,98,10,23],[0,0,45,8],[… JSON

- 0: [0,0,0,0]
- 1: [0,98,10,23]
- 2: [0,0,45,8]
- 3: [0,3,18,93]

Overall accuracy JSON

0.7635658914728682

burned area with unit of hectare JSON

5365.91

Note:We are using the random function to select %70 and %30 parts of the samples, so every time we run the function it selects different parts as a %70,%30 part. So we wanted to find average overall accuracy and we run the each function 5 times then calculated the mean of the results.

Condition-1 : smileCart()

0.898601398601398	0.8916967509025271
0.9131832797427653	0.8943894389438944
0.89568345323741	0.8989169675090253
0.868000000000000	0.9124087591240876
0.9253731343283582	0.9124087591240876

Table 1

Average Overall accuracy=0.9011 :%90

Condition-2 : smileRandomForest(10)

0.9096774193548387	0.8833922261484098
0.89419795221843	0.8833922261484098
0.9283276450511946	0.9269102990033222
0.9209621993127147	0.910958904109589
0.9039145907473309	0.9084507042253521

Table 2

Average Overall accuracy=0.9070: %90

Condition-3 : minimumDistance()

0.7635658914728682	0.7180327868852459
0.7441077441077442	0.7509157509157509
0.7515923566878981	0.7509157509157509
0.7300613496932515	0.7692307692307693
0.7353951890034365	0.7395498392282959

Table 3

Average Overall accuracy=0.7453 : %74

5 DISCUSSION

Gee's unstable syntax and function formats:

Google Earth Engine very useful product but it develop and change itself. During the developing our project some syntaxes are changed and our products suddenly gives syntax errors which is means codes stopped to work. Some examples of it;

- Cart algorithm changed to smile-cart so cart functions does not available anymore
- Normally "01" means "1" and it work without any problem however syntax changed. So developers must use 1 instead of 01.
- LCV algorithm mechanism developed so a results obtained from lcv algorithm changed. In our opinion this algorithm still not reliable.

Also google earth engine has its own structure of syntax which is independent from javascript. For example some JavaScript basic operations (add, subtract…) does not work with gee values and gee has its own unique object type which only can process with gee functions. We had several questions about this project so this questions asked on gis.stackexchange and answered from other developers. Example of questions below here;

- https://gis.stackexchange.com/questions/378297/google-earth-engine-ee-number-function-produce-an-object-instead-of-number-val
- https://gis.stackexchange.com/questions/376362/extracting-geometry-data-of-classes-in-classified-image-or-masking-class-layers
- https://gis.stackexchange.com/questions/377592/meaning-of-numpixels-parameter-for-unsupervised-clustering-in-google-earth-engin

Unsupervised codes could crush during running. In this situation users should select "wait" button instead of "close the site". After several click the code will give results.

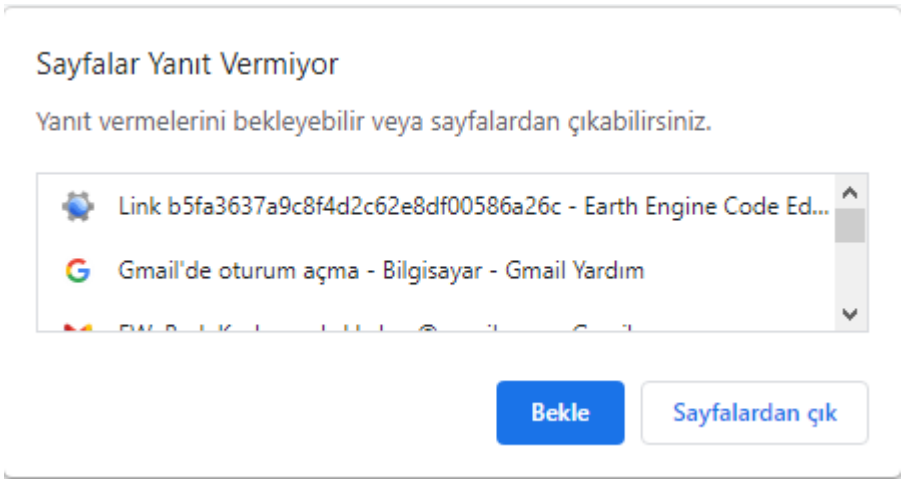


Figure 44

Future goals:

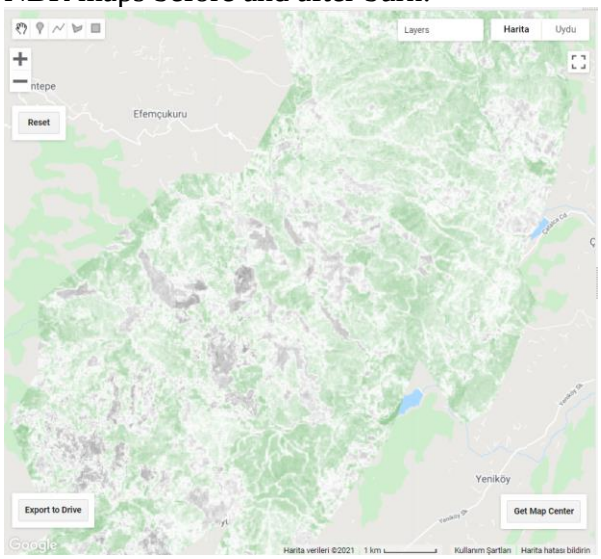
- Better user friendly interface
- Forest fire risk estimating of regions
- Applying geospatial statistic analysis on maps
- Converting raster cluster layers into vector layers
- Tree cutting date detection with yearly time interval inputs
- Calculate forest fire spreading rate
- Analysis spreading direction of forest fire
- Reach this project to more user
- Analysis effects of meteorological phenomena and anomalies
- Comparing results with higher resolution satellites

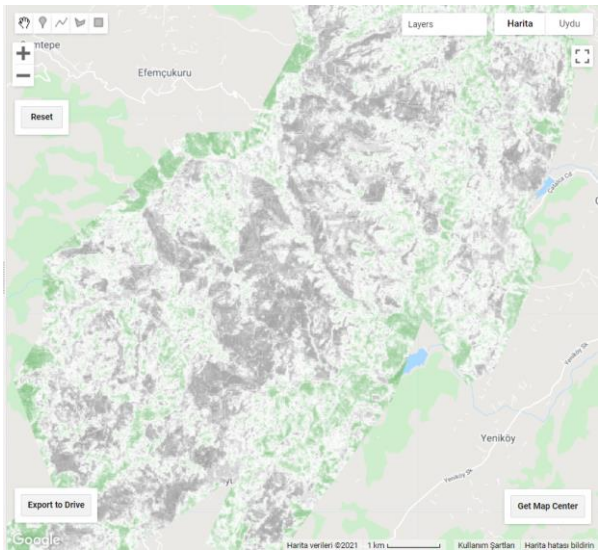
6 CONCLUSION

In conclusion many algorithms, methods and implemented and tested in just using one platform which is google earth engine. Our comment about google earth engine it is very fast cloud based technology so old generation computers can also make fast implementation. However; gee syntaxes and methods could be change. A working code could stop work and need some changes on it. So we can say that a written codes need constant updates according to google earth engine updates. Some gee algorithms not reliable too like svm and lcv. Their results could be wrong and also this functions constantly changing nowadays so they can give different results on different implementation time with same inputs. The other methods like k-means, minimum distances, smile-cart tested and approved by us. The datas like when fire started-ended, how many hectare burned calculated with gee also tested and verified. To optimal results geometry and date interval should select close to true values.

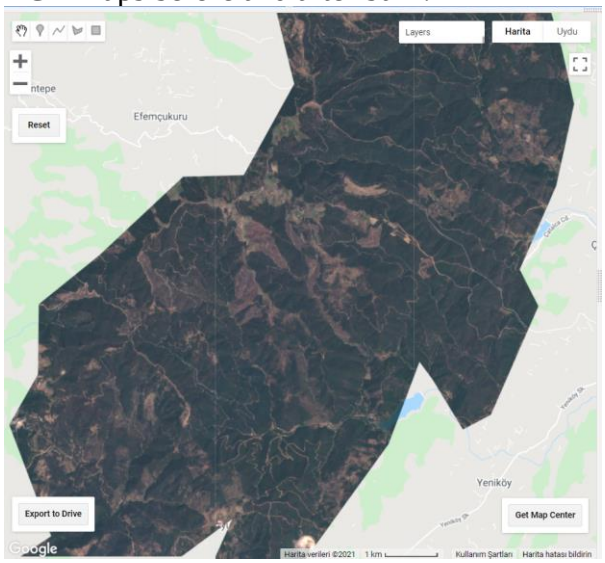
For our main investigation sample (Izmir/Karabağlar) results shown below here:
Burn Severity Rate: Low Severity Burn
When Fire Started: 16 August 2019
When Fire Ended: 21 August 2019
Approximate How many area burned (hectare): 4653.26 - 5144.55
Temperature at day time when fire started and ended (Celsius): 38.4 – 35.9
Temperature at night time when fire started and ended (Celsius): 23.8 - 24.3

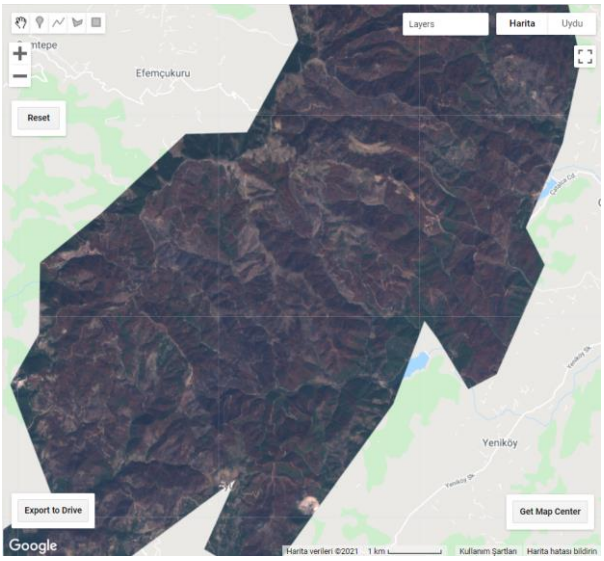
NBR maps before and after burn:





RGB maps before and after burn:





REFERENCES

http://spatialanalyst.net/ILWIS/htm/ilwis/how_to_calculate_confusion_matrix.htm

http://spatialanalyst.net/ILWIS/htm/ilwis/how_to_calculate_confusion_matrix.htm

https://docs.microsoft.com/tr-tr/azure/machine-learning/how-to-understand-automated-ml

https://www.sciencedirect.com/science/article/abs/pii/S235293852030166X

http://un-spider.org/advisory-support/recommended-practices/recommended-practice-burn-severity/in-detail/normalized-burn-ratio#:~:text=The%20Normalized%20Burn%20Ratio%20(NBR,shortwave%20infrared%20(SWIR)%20wavelengths.

https://eos.com/make-an-analysis/ndvi/

https://developers.google.com/earth-engine/apidocs

APPENDICES