POLITECNICO
MILANO 1863

M.Sc. Geoinformatics Engineering
Geoinformatics Project

Project Technical Report

Analysis of factors affecting wildfire distribution

Abbes Madeleine
madeleinesarah.abbes@mail.polimi.it

Rajabi Forough
forough.rajabi@mail.polimi.it

Zallemi Nikolina
nikolina.zallemi@mail.polimi.it

January 12, 2024

Link to the website:
https://github.com/madeleinemadeleinemadeleine/FactorsAffectingWildfires

Contents

Abstract

This project is the premise of a larger study, aiming at understanding what are the factors affecting wildfire distribution around the globe.
We propose two different methods to extract information on wildfires from multi-spectral satellite imagery, in order to study what are the factors that are affecting wildfires' distribution at a global scale. An image containing all the factors will be created using Google Earth-Engine, and we will randomly extract points from both the burnt and unburnt areas, to analyze them through machine learning classification tools (Random Forest, Adaptive Boosting, dimension reduction analysis, etc.).
In order to extract points from the image, we propose two different methods. The first method, more straightforward consists of applying a mask over the area of interest in order to extract points from it. This solution leads to interesting and fast first results.
The second method is using a hexagon tiling to extract the points in a particular area of the globe, which is being distinguished by the amount of burned pixels. This method leads to a higher flexibility in the point extraction and allowed us to reach higher accuracy than the first method in the classification process.
Our work sets the scene and material for further analysis of these factors, while already providing some preliminary experiments on the data.

Keywords : *Google Earth-Engine, Remote Sensing, wildfires global distribution*

1 Introduction

Climate change is expected to modify the geographic distribution of wildfire in the future [3]. In 2023, many regions experienced record-breaking wildfire activity [4]. The damages these wildfires caused to the environment and population are not only caused by the flames themselves. Among the damages triggered by wildfires, we have also smoke pollution, and CO_2 emissions. As 2023 was the hottest year ever recorded for the last 150 years [1], we can imagine that global temperature is an inducing factor for wildfires. However, combustion is a very complex physical reaction, and the set of factors reunited required to induce a wildfire is far from being trivial. This document will provide a detailed description of all the material, ideas, and steps that lead to the development of the project topic and implementation of the necessary codes. Further, it will include the final results, future developments, and improvements, without leaving aside the problems faced during the work process.

1.1 Purpose

The objective of the project is to assess the spatio-temporal distribution of Burned Areas, on a global scale. Having these burned areas depicted, the purpose goes further into listing and analyzing the influential factors, on the distribution of the wildfires around the world.
The chosen tools for making this analysis are Google Earth Engine and Machine Learning Algorithms, specifically Random Forest Algorithm. For obtaining additional products, which were not provided by GEE, the usage of other external, open-source software was allowed, specifically QGIS.
To finalize, GEE Python API, was used to perform the algorithms and visualize the final results.

1.2 Definitions, Acronyms

- **QGIS**: A free and open-source cross-platform desktop geographic information system (GIS) application that supports viewing, editing, printing, and analysis of geospatial data.
- **GEE**: Google Earth Engine (GEE) is a cloud-based platform for global-scale geospatial analysis which allows to process a variety of geographical data at scale and handle large geographical datasets.
- **API**: Stands for Application Programming Interface. In the context of GEE, Python API means that GEE can be accessed from a Python environment such as Jupyter Notebook, which we have used for the final analysis.
- **CRS**: Coordinate reference system is a coordinate-based local, regional, or global system used to locate geographical entities.
- **PCA**: Principal component analysis is a dimensionality reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

2 Input Products

2.1 GEE Collections

For the development and completion of the project we used different image and raster collections provided by GEE:

1. **Hansen Global Forest Change v1.10 (2000-2022)**
   Hansen Global Forest Change refers to a dataset developed by researchers at the University of Maryland, led by Dr. Matthew Hansen. The dataset provides high-resolution information on global forest cover and changes over time, based on Landsat images.
   This raster represents mapped global tree cover extent, loss, and gain at a spatial resolution of 30 m, with loss allocated annually.[2]
   Of all the 13 bands included in this collection, we are making use of the "datamask" band only.

2. **MCD64A1.061 MODIS Burned Area Monthly Global 500m**
   The Terra and Aqua combined Moderate Resolution Imaging Spectroradiometer (MODIS) is the main image collection from where we get the information of the wildfires occurred in the world, in a specified period of time, which can be months or years.
   This is a monthly, global gridded 500meter(m) product containing per-pixel burned-area and quality information. The algorithm uses a burn-sensitive Vegetation Index (VI) and identifies the date of burn for the 500m grid cells within each individual MODIS tile.
   The data layers provided in the MCD64A1 product include Burn Date, Burn Data Uncertainty, and Quality Assurance, along with First Day and Last Day of reliable change detection of the year. From all the above data layers, differently called bands, we are using only the "Burn-Date".

3. **MCD12Q1.061 MODIS Land Cover Type Yearly Global 500m**
   The Terra and Aqua combined MODIS Land Cover collection provides us all global land cover types. It has a yearly temporal scale with 500m grid cells.
   We used this collection to obtain the land cover, which is a crucial factor in analyzing the distribution of fires.

4. **TerraClimate: Monthly Climate and Climatic Water Balance for Global Terrestrial Surfaces, University of Idaho**
   TerraClimate is a collection of rasters of monthly climatic water balance for global terrestrial surfaces. We used this collection to obtain the data of the climatic variables, used as factors for analyzing the distribution of the wildfires. The bands used from the collection are listed in Table 1 below:

BAND	Description	Unit
soil	Soil Moisture	mm
tmmn	Minimum temperature	degrees Celsius
tmmx	Maximum temperature	degrees Celsius
def	Climate water deficit	mm
pr	Precipitation accumulation	mm
vs	Wind-speed	m/s

Table 1: Climatic factors affecting wildfires

5. **NASA SRTM Digital Elevation 30m**
   The Shuttle Radar Topography Mission (SRTM) digital elevation data provides a global scale digital elevation model in a raster form. It allows us to obtain the topographic variables such as slope and aspect, in meters, and use them as additional factors to further complete our study.

6. **USGS Landsat 8 Collection 2 Tier 1 TOA Reflectance**
   The Landsat 8 image collection is a part of the United States Geological Survey's (USGS) Landsat program. It is employed in our analysis due to its capability to capture high-resolution multi-spectral imagery, making it suitable for monitoring land surface dynamical changes.
   Landsat 8 enables the extraction of valuable information about land surface characteristics, such as the Normalized Difference Vegetation Index (NDVI) and Normalized Difference Moisture Index (NDMI).
   NDVI is a key vegetation indicator used to assess vegetation health and density. It was computed using the bands B5 (Near Infrared) and B4 (Red) of the collection (1).

$$\text{NDVI} = \frac{\text{B5 (Near Infrared)} - \text{B4 (Red)}}{\text{B5 (Near Infrared)} + \text{B4 (Red)}} \tag{1}$$

   On the other hand, NDMI is a key moisture indicator, providing insights into the moisture content on the surface. It was computed using the bands B5 (Near Infrared) and B6 (Shortwave Infrared 1) of the collection (2).

$$\text{NDMI} = \frac{\text{B5 (Near Infrared)} - \text{B6 (Shortwave Infrared 1)}}{\text{B5 (Near Infrared)} + \text{B6 (Shortwave Infrared 1)}} \tag{2}$$

7. **Human Impact Index (HII) by Wildlife Conservation Society**
   The human footprint map measures the cumulative impact of human activities on nature. It includes eight inputs:

   - the extent of built environments,

- cropland,
- pasture land,
- human population density,
- night-time lights,
- railways,
- roads, and
- navigable waterways,

which can be visualized on this link.
The images of this collection have a nominal scale of 300, which means that each pixel in the HII dataset represents an area of 300 units on the Earth's surface, based on the chosen coordinate reference system. The values vary from 0, minimum value, which indicates no human impact, to 5588, maximum value, which is a discrete value indicating the maximum impact of humans in the environment.

2.2 External Products

This section includes products that were taken or produced from external software, not connected to GEE.

2.2.1 Hexagon Map Grid

This hexagon grid was created using QGIS tools and it is used in the development of the second approach of the project, where it is also explained its importance.
For our purpose, it was chosen a 300 km horizontal and vertical spacing. Horizontal spacing means that when you move horizontally from the center of one hexagon to the center of its neighboring hexagon, the distance covered will be 300 kilometers. On the other hand, vertical spacing means that when you move vertically from the center of one hexagon to the center of its neighboring hexagon, the distance covered will be 300 kilometers.
This approach can be suitable for representing geographic areas at a broader scale. Since our analysis considers a global scale, we are allowed to analyze or visualize data with a lower level of detail.
In order to reproduce a grid of another scale for a more detailed analysis, for future studies, you can consult the link.

2.2.2 Shapefiles

The shapefiles used to obtain the borders of each continent were downloaded from figshare. All 7 products are uploaded as assets on the GEE project folder:

1. Europe
2. Africa
3. Asia
4. North America
5. South America
6. Australia
7. Oceania

2.3 Factors affecting wildfire distribution

In order to study wildfire distribution, we need to understand better the physical concept behind combustion. Wildfire is an ecological disturbance process controlled by the coincidence of three basic requirements: resources to burn (vegetation), environmental conditions that promote combustion, and ignition [3].
To select the most interesting factors for our case of study, which are listed in Table 2, we used two publications: an article from the French National Research Institute [3] and the materials of a masterclass on fire detection provided by NASA [5].

Variable	**Description**
Slope	Slope of the terrain
Aspect	Aspect (direction the slope faces) of the terrain
Wind Speed	Wind speed
Temperature (Max)	Maximum temperature
Temperature (Min)	Minimum temperature
Water Deficit	Climate water deficit
Precipitation	Precipitation accumulation
Soil Moisture	Soil moisture
NDVI	Normalized Difference Vegetation Index (NDVI)
NDMI	Normalized Difference Moisture Index (NDMI)
Land Cover	Land cover type from MCD12Q1 dataset
Human Impact Index	Human impact index

Table 2: Environmental, Climatic and Social Variables

3 Development approaches

In this section, we outline the steps taken to realize the project's objectives, clarifying the reasoning behind each decision. We present two distinct solutions, each of them representing a unique approach to addressing the identified challenges. A detailed explanation of these solutions is provided, complemented by references to the corresponding codes.
Additionally, for each step, we highlight supplementary suggestions and alternative options, in case future users want to perform modifications according to their final goal.

3.1 Data preprocessing and Initial decisions

3.1.1 Clipping the Area of Interest in continents

The purpose of this project is to develop an analysis on a global scale, using GEE, consequently our AOI is a polygon including the whole world. A crucial problem when using GEE for processing a large number of pixels is the user memory limit. Since the Earth Engine platform has a finite amount of RAM available for computation, it was impossible to process the data for the whole world all at once.
To avoid this obstacle, we decided to use shapefiles (see section 2.2.2) representing the administrative boundaries of each continent (see Figure 1). In this way, we could perform the computations once per continent without risking to exceed the user's memory.
Another positive point is related to the fact that by using shapefiles, we could totally exclude the pixels representing oceans and seas from the processing step. We are interested in fires and the information regarding water areas is not necessary. Consequently, the amount of pixels became smaller and the computation faster.

3.1.2 Removing Cloud Cover and Shadows from Landsat images

Since we are making use of the Landsat 8 images for calculating the NDVI and NDMI factors (see section 6), we have to apply some preprocessing to these images before using them. For this reason, we use the function **'maskL8clouds'** that is **built** to mask out clouds and shadows from the images. This function uses the **QA band** from the Landsat images and defines **bitmasks**[1] to isolate the cloud and shadow bit in the QA-band. Multispectral imagery products, like Landsat, contain one or more QA bands that allow the users to extract information about cloud pixels and mask them, by increasing in this way the data quality for the analysis.
The purpose of removing clouds and shadows is to make sure that the following steps, such as computing the vegetation indexes (NDVI and NDMI),

[1] To learn more about the bitmask, you can consult the link.

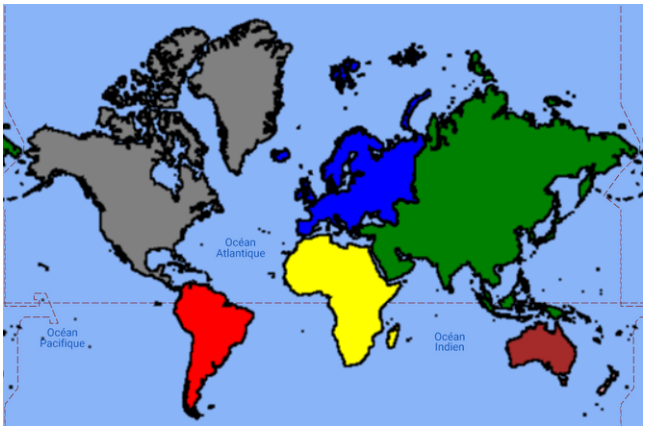

Figure 1: The world divided into 7 parts for reducing computation cost (Africa, North/South America, Asia, Australia, Europe, Oceania)

are not affected by their presence, which can introduce noise and errors in the results.

3.1.3 Aggregation of variables

This step involves preprocessing of the factors listed in Table 2. Since we are considering a large time range, of 13 years, we have to apply some statistics to the data and prepare them for subsequent analysis. Aggregation of the variables, over the specified time range, can provide temporal consistency in the dataset, by offering a meaningful comparison over time.

- **Climate variables**
  Considering the median value for each climatic factor, we can create a consistent and manageable dataset. Simplifying the final set of variables, that is going to be used for training the network, is necessary, since we are considering climate variables of different units and temporal variations.
  To add more, aggregation using median values helps avoid the impact of outliers in the data, ensuring a representative value for each variable during the time interval.
  For all the above reasons we have used the **.median()** function, which calculates the median value for each pixel across the specified time series. If the user wants to change the statistics considered, it can change this function with **.mean()**[2] or continue a deeper study for deciding on a mode value for each factor. Of course, this change has to be suitable for the nature of the data included.

[2]To switch from median to mean just comment the ".median()" and uncomment the ".mean()" line of code

- **Topographic variables**
  For slope and aspect that are topographic data and are static over short periods, no aggregation was applied.

- **Vegetation indices variables**
  The same aggregation as for the climate variables, was applied also for the vegetation indices NDVI and NDMI. The **.median()** function is applied to the processed image collection of Landsat. In this case, the function calculates the median value for each pixel across all the images in the collection. This results in a single composite image that represents the median values for each pixel over the specified time interval. In conclusion, the indices are calculated from the median values of the necessary bands of the new composite image, according to the formulas (1) and (2).

- **Land Cover variable**
  The land cover variable represents categorical values, rather than continuous numerical measurements, like the climatic variables. Also in the case of land cover, we distinguish spatial patterns that are better captured by taking the true values rather than summarizing them with a statistical measure.
  The Land Cover product was reprojected to match the CRS of the other collections.

- **Human Impact Index variable**
  In the case of the Human Impact Index, it was reasonable to consider the mean value, since HII is a composite index that represents the average effect of multiple factors. The mean value represents the central tendency of the distribution of the human impact across the area where the training points of the algorithm are going to be taken.

3.1.4 Creating a Predictor Image

Having a list of variables to input as predictor variables for the machine learning algorithm, we decided to produce a variable called **Predictor Image**[3]. This image serves as a dataset that combines all the preprocessed variables that affect the distribution of fires, which are considered as bands of this predictor image.
The product image serves as an input for wildfire risk assessment and it is further used in the process of sampling to generate training points for the Random Forest algorithm.

[3]Check the code for the creation of Predictor Image in file Stratified_Sampling inside GEE repository

3.2 First approach

3.2.1 Creating a binary mask

burnMask file: As a first approach for the development of the problem, we are creating a binary mask of burned and non-burned pixels. This is a crucial product for performing the Random Forest algorithm, over a classification problem.
The code is separated into three sections:

- Time Range and Land Data Mask:
  The first section of the code establishes the temporal distribution of the analysis by defining a start date (January 1, 2010) and an end date (September 1, 2023). This interval sets the time interval for collecting information on burned pixels related to wildfires.
  We are considering 13 years for the temporal distribution of the wildfires, but according to the preferences of the users and the purpose of the study, this interval can be changed, by just putting the desired dates on the **startDate** and **endDate** variables.
  Further, the script extracts a land data mask from the Global Forest Change dataset (see collection 1). The mask, represented by the variable **MaskLand**, helps filter out irrelevant information and focus the analysis only on areas where wildfires might have occurred.

- Creation of Burn Mask from MODIS Data:
  The code creates a burn mask, named variable **myMask**, using MODIS data (see collection 2) filtered within the specified time range. The mask is derived from the **'BurnDate'** band of the MODIS dataset and identifies areas with at least one occurrence of burning. The resulting mask is then applied to unmask burned areas while setting non-burned areas to zero.
  This process effectively isolates regions affected by wildfires, by creating a binary representation of burned and non-burned pixels.
  The burn mask and the MODIS data are added to the map for a visual inspection.

- Exporting the Burn Mask to GEE Asset:
  The final part of the code enables exporting the generated burn mask as an image, to the GEE assets repository. The *Export.image.toAsset* function is utilized for this purpose.
  The **scale** of **50,000**, determines the spatial resolution of the exported image. Opting for this scale aims to make the analysis faster and minimize computational demands. While a bigger scale would lead to a more detailed analysis, for our wide Area of Interest, a coarser scale still proves to be pragmatic. This choice balances the need for sufficient details with the computational efficiency necessary for processing

The Coordinate Reference System **(CRS)** is specified as **EPSG:4326**. This step enables the preservation and accessibility of the burn mask in the GEE environment, making it easily accessible and facilitating further analysis, visualization, and integration with other Earth observation datasets at any time.

3.2.2 Performing stratified sampling

Stratified_Sampling file: After creating the predictor image and the binary mask we combine these products to be able to distinguish which training points are going to belong to burned pixels and which to unburned pixels. From the binary mask, we select the band "Burn" and add it as a band of the predictor image. In this way now the predictor image has 13 bands.
As the final step a stratified random sampling is performed, using the function **predictorImage.stratifiedSample**, which indicates that the points are sampled from the predictor image.

- We specify the "Burn" band as the band used for stratification.
- The scale for sampling, called also spatial resolution, is the same as the one of the binary mask.
- As the number of points to be sampled we are choosing 100 points for each class of the "Burn" band. This means that the function is going to create 100 points on the burned pixels, with a Burn value equal to 1, and 100 more on the unburned pixels, with a Burn value equal to 0.
- As a region for sampling the points we choose each continent one by one. In this way, for every continent, we are going to receive a dataset of training points.

Finally, each training point is going to contain all the bands of the predictor image, with their values and also the geometry, which means its coordinates.

- Exporting the Training Points to Google Drive:
  This step is useful for further analysis outside of GEE, such as the Python Jupyter Notebook. The training points, together with their bands, are exported in CSV format. At the end, we are receiving 7 CSV files, one per each continent.
- Merging the Training Points in one CSV file:
  This step can alternatively be performed on GEE or in Jupyter Notebook. The function **.merge** helps us to merge all datasets in a unique CSV file.
  The merged points can be also saved as an Asset of GEE repository or as a shapefile by using 'shp' as "fileFormat".

3.3 Second approach

In this section, we are presenting another option for sampling the necessary training points, for algorithm development. We are suggesting an alternative solution, which in our opinion, can lead to better results not only for the purpose of our study but also for more detailed analyses that include smaller areas.
Since our main problem, when coding in GEE, has been the user memory limit, we created and imported a hexagon grid (see product 2.2.1), which offers several advantages for our spatial analysis and for handling raster data. Our purpose is to simplify the sampling process, in such a way that the code will not generate errors and that this approach can be used in the future for projects requiring different temporal and spatial resolutions.
For this method we are creating two files. In one we sample points on the burned pixels and in the other file, we sample points on the non-burned pixels.

3.3.1 Hexagon_method_burned_pixels file

- The imported hexagon grid is being clipped according to the geometry of the shapefile of each continent. Next, also the MODIS Burned Area is clipped to the extent of the clipped hexagon grid.

- Pure fires binary mask:
  The MODIS collection is filtered by the start and end date, just like in the first approach. Further, a burn date threshold equal to 1 is chosen. Choosing this threshold allows the creation of a binary mask. Pixels with a burn date greater than or equal to the threshold are marked as burned and a value of 1 is assigned to them. The rest of the pixels are marked as not burned and a value of 0 is assigned.
  Each image now has a binary mask. All these binary masks are summed up, creating a cumulative burned area image. This image is then processed spatially. It is reprojected and resampled to a 15km scale.
  Finally, all values in 'Pure Fires' that are greater than or equal to 1, are set to 1 and are retained in the image. The rest are masked out.

- Creating an area image and performing zonal statistics:
  The binary burned mask is multiplied by the pixel area of each pixel in the image, giving an area image where each pixel represents the area of burned regions.
  Using the area image, the **'reduceRegions'** function is used to perform zonal statistics, specifically, to aggregate the information from the area image within each hexagon, by using as a reducer the **'sum()'**.
  Finally, the number of the hexagons is calculated and divided, into hexagons with burned pixel sum 0 and burned pixel sum not 0 (see

Figure 2). For each of them is created a Feature Collection.

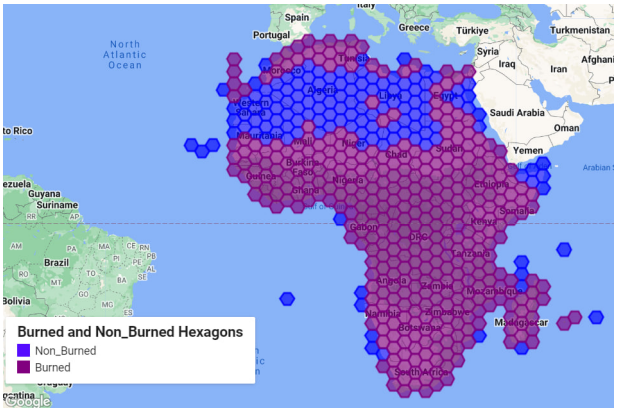

Figure 2: The purple hexagons refer to burned pixel sum Not 0. The blue hexagons refer to the burned pixel sum 0. (Africa example)

- Extracting sample points over hexagons:
  This process starts with creating a Feature Collection representing hexagons with burned pixels. Since we want to access each hexagon with the burned pixel sum not 0 and extract the burned area information within each of them, we extract the 'id' of each hexagon.
  Further, iterating over a subset of the hexagons of the Feature Collection, we can:

  1. Filter hexagons with a specific 'id'.
  2. Generate a number of stratified sample points within the burned areas (see Figure 3). The points are stratified sampled over the Pure Fire mask.
  3. Extract values from the predictor image variables, for each sampled point.

  It is important to mention that for each continent we choose a different number of hexagons and points to sample within each hexagon. This depends on the total area covered by the hexagons of burned pixels. The bigger the area, the higher the number of points we want to collect because we want to make sure the samples come from different parts of each continent. Another reason stands behind the problem of limited memory exceeding. We have to adjust the number of hexagons and points (*see Table 3*), in order to overcome the problems but also achieve our purpose.
  Finally, the results are flattened and merged into a final Feature Col-

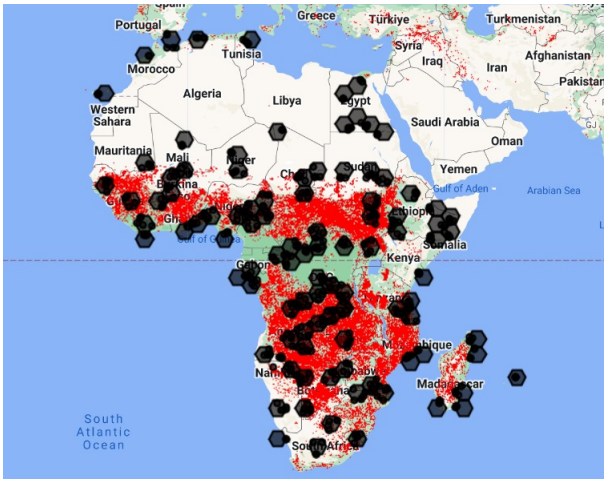

Figure 3: Burned points within each hexagon before sampling. (Africa example)

lection called **'finalresults'**, which contains the sampled points in the burned pixels.

- Exporting the final product to Google Drive:
  The final Feature Collection is exported as a CSV file in a Google Drive folder, to be further processed in Python Jupyter Notebook.

Continent	**Burned**	
	Hexagons	**Points within each hexagon**
Africa	50	5
Asia	50	10
Australia	30	5
Europe	30	5
North America	30	5
South America	50	5
Oceania	20	5

Table 3: Hexagon and Burned Point Distribution by Continent

3.3.2 Hexagon_method_unburned_pixels file

The file has the same steps as in the above file. The steps that differ are:

- Preparing an unburned image:
  Here the Feature Collection of hexagons with burned pixel sum 0 is reduced to an image with a band called 'unburned', within each hexagon.

The image is then converted to bytes, reprojected to the correct CRS, and clipped.

- Looping over hexagons and sampling points in the unburned areas:
  A loop over a set of hexagons, assigns a specified number of unburned points, for each hexagon (see Figure 4). The points are stratified sampled. For the same reasons as above, also for the unburned areas, the number of hexagons and points within each hexagon is adjusted, for each continent (*see Table 4*).
  The predictor variables are then extracted for these sampled points and organized into a Feature Collection, called 'finalresults'.

- Exporting the final product to Google Drive:
  Again the final Feature Collection is exported as a CSV file, for further analysis.

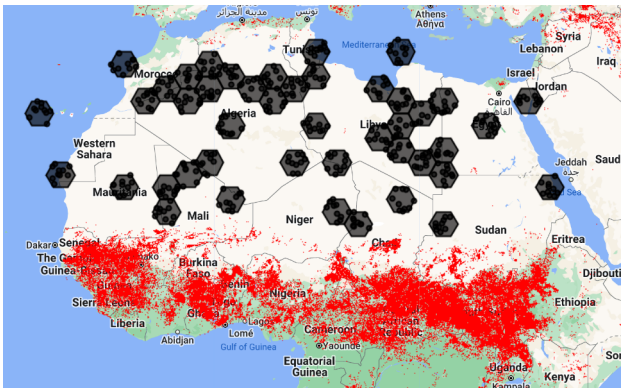

Figure 4: Non-burned points within each hexagon before sampling.(Africa example)

Continent	**Non-burned**	
	Hexagons	**Points within each hexagon**
Africa	50	10
Asia	50	20
Australia	3	100
Europe	50	20
North America	50	20
South America	44	20
Oceania	50	20

Table 4: Hexagon and Non-burned Point Distribution by Continent

4 Analysis of the datasets

4.1 First approach

4.1.1 Data exploration

Once the images with the factors are sampled, the tables of points are exported (to Google Drive or as assets in GEE) and explored in a **python notebook**[4].
The very first goal of the project was to build and train a classifier that would use the chosen factors as features to decide if a point is *burnt* or *unburnt* according to the criteria mentioned above.
To do so, the .csv file containing the sampled points with their features (the factors), was loaded into the code as a Pandas' DataFrame. There are in total 1251 points, collected just like described in section 3.2. Each point, represented by a row in the DataFrame, contains values for all the 14 features listed below (*Table 5*), which are the columns of the DataFrame.
Half of the points (601) belong to the *burn* category, which means they have *burn* = *True* as attribute. The other half (650) has the attribute *burn* = *False*.

feature	type object
point id	int 8
Land cover	int 8
Human Impact Index	float 64
NDMI	float 64
NDVI	float 64
aspect	float 64
precipitation	float 64
slope	float 64
soil moisture content	float 64
wind speed	float 64
water deficit	float 64
max temperature (°C)	float 64
min temperature (°C)	float 64
burn	boolean

Table 5: The features of every point extracted during the first methods. The right-handed column contains the object type of each attribute.

[4]Check file First_approach in the project folder.

4.1.2 Model Training (Random Forest Algorithm)

The data was split into two distinct subsets for training and testing the model. A ratio of 80%-20% was used. This indicates that 80% of the whole dataset was used for training the model and 20% was used for testing the models' accuracy. As the number of total points is more than 1200, we could train our model on a bigger part of the dataset and still have enough points for testing the model.
The split was perfomed randomly, using **sklearn.model_selection.train_test_split** function in Python. Due to the care put in creating a dataset with balanced categories burnt and unburnt, we can expect to get a similar proportion of both categories in the sets randomly generated.
After the train-set/test-set split, we can train our model. We chose to use the Random Forest model, that is built in using the sci-kit library's function **sklearn.ensemble.RandomForestClassifier**. This model performs better on not-normalized input data, hence we chose not to normalize our data in the first place.
With the mentioned settings, we get from the trained model the results shown in Table 6 and Figure 5. The accuracy is acceptable so we accept the model as successful.

Model's accuracy:	0.7370
factor	importance
wind speed	0.104316
precipitation	0.095617
NDMI	0.087743
Soil Moisture Content	0.087285
Max Temperature	0.085794
NDVI	0.085042
slope	0.083885
water deficit	0.077022
Human Impact Index	0.076335
Min Temperature	0.075240
Land Cover	0.074702
aspect	0.067018

Table 6: Model's accuracy on testing set and importance of the factors in the classification.

To increase the accuracy we are feeding to the model some extra points we sampled manually on GEE[5] and saved them as assets. Since we are

[5]Check assets with Id: UnB_points_manually_sum0 for unburned points selected manually, B_points_sum1 for burned points selected manually.

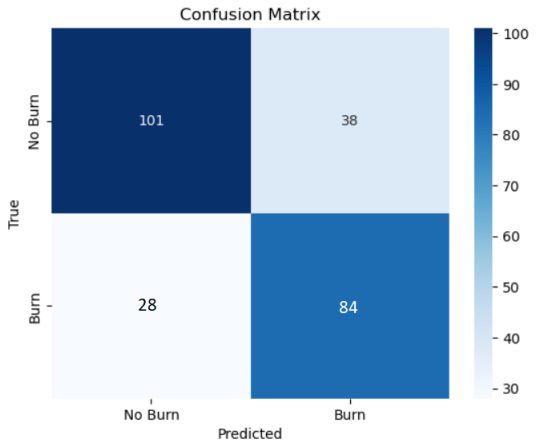

Figure 5: Confusion matrix after classifying our test set with the model

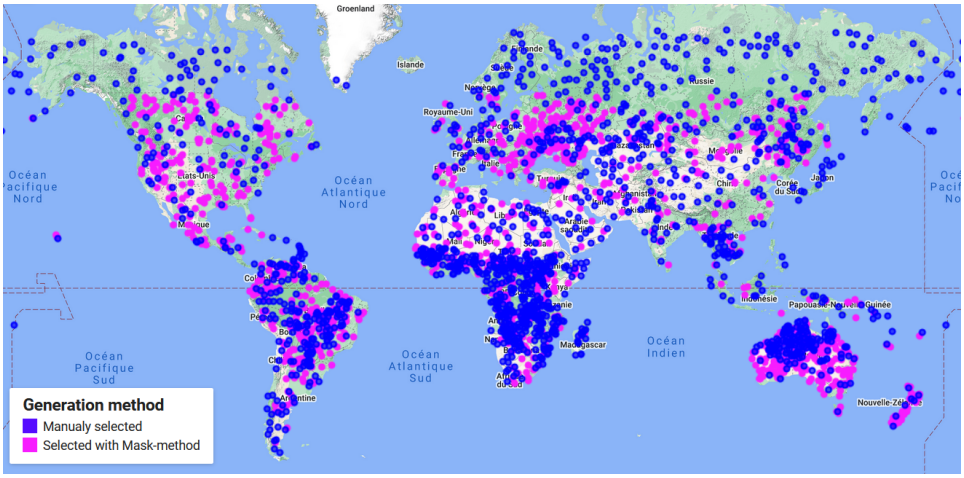

Figure 6: In pink are the points used for the first model training (*Table 6*). In blue are the points we added to the pink one for the second training (*Table 7*)

.

increasing the dataset also the accuracy is increasing. The model has more training and testing points, which makes it increases the learning rate. The results are presented in Table 7 and Figure 7.
Feature importance in the Random Forest algorithm provides us insights into the contribution of chosen features (variables) to the overall performance. The algorithm evaluates each feature's significance by considering how much it reduces the error in the model. Features with higher importance values have a bigger impact on the model's predictions.

Model's accuracy:	0.7526
factor	importance
WindSpeed	0.100515
NDVI	0.095257
NDMI	0.089032
Precipitation	0.087268
HumanImpactIndexMean	0.085429
SoilMoist	0.083466
TempMax	0.083202
Slope	0.081115
LandCover	0.079303
WaterDeficit	0.078512
TempMin	0.073839
Aspect	0.063061

Table 7: Second model's accuracy on testing set and importance of the factors in the classification.

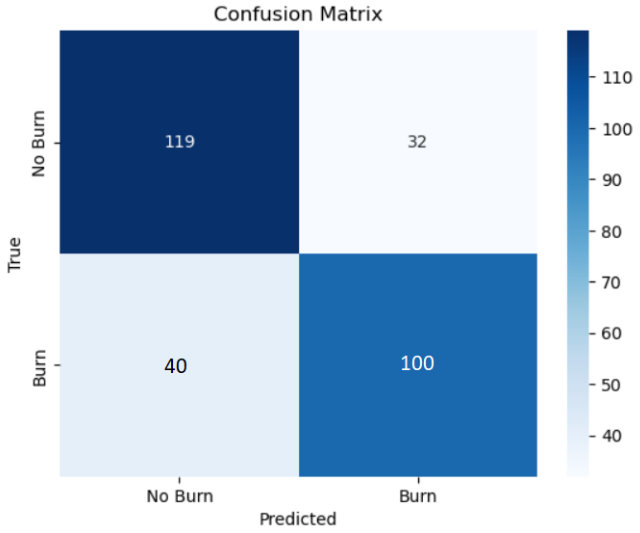

Figure 7: Confusion matrix after classifying our test set with the second model

4.1.3 Dimentionality study (PCA)

In order to investigate the importance of the factor in the classification, we performed a Principal Component Analysis (PCA) (see **python notebook**[6]), in order to see if the dimensionality of the data could be reduced

[6]Check file PCA_first_approach in the project folder.

(hence, one factor is redundant). However, we could observe that all components had a significant impact on dimensionality. Indeed, there is no striking difference between the repartition of the importance of the PCA components in table 8 and of the initial factors in table 6.
Further experiments were led such as reducing the dimensionality of the data, taking only the first d components for training, and testing the classifier. But by doing so, the accuracy of the model would just go down drastically and no improvement of any kind was observed.

Model's accuracy:	0.7012
component	importance
pc1	0.103537
pc11	0.102431
pc2	0.098119
pc6	0.097604
pc4	0.087990
pc7	0.082340
pc10	0.075084
pc9	0.074980
pc8	0.072754
pc12	0.071393
pc3	0.070099
pc5	0.063672

Table 8: Performances of the model after applying PCA to the dataset.

4.2 Second approach

The tables of points are exported to Google Drive as a CSV file and explored in a **python notebook**[7].

4.2.1 Data exploration

The CSV products containing the points of*Table 3* and *Table 4* are presented as Pandas Dataframes. To create class balance, only 1200 unburned and 1200 burned points are randomly extracted from the tables and a final concatenated table is formed.
Each point, represented by a row in the DataFrame, contains values for all the 14 features listed below (*Table 9*), which are the columns of the DataFrame. The attribute hazard represents the burned points if the value is 1, and the unburned points, if the the value is 0.

feature	type object
point id	int 8
Land cover	int 8
Human Impact Index	float 64
NDMI	float 64
NDVI	float 64
aspect	float 64
precipitation	float 64
slope	float 64
soil moisture content	float 64
wind speed	float 64
water deficit	float 64
max temperature (°C)	float 64
min temperature (°C)	float 64
hazard	int

Table 9: The features of every point extracted during the second method. The right-handed column contains the object type of each attribute.

4.2.2 Data Splitting & Cross-Validation

The data is split into training and testings sets, in order to train the models used for predicting the feature importance of the factors affecting wildfires. The features are the input variables and the column hazard is the target variable, so the class that we want to predict, using different classification methods.

[7]Check file Second_approach in the project folder.

The data is randomly shuffled before being split. 20% of the data will be used for testing, and the remaining 80% will be used for training.

We are setting a seed, which ensures that the random shuffling is reproducible. In this way, running the code multiple times with the same seed, will result in the same train-test split.
We evaluated that a Cross-validation technique would improve the performance of the following machine-learning models by avoiding overfitting. The models will be able to generalize better because the dataset will be split into multiple subsets. Some of these subsets will be used to train the model and others in the training process.
We are choosing **StratifiedKFold** cross-validation splitter, that maintains the proportion of target classes in each subset (fold). The dataset is divided into 30 folds.

4.2.3 Classification Methods and Results

We are exploiting various classification algorithms to model and predict the occurrence of wildfires based on listed features. These diverse methods allow us to assess the impact of algorithmic choices on the performance of wildfire prediction.
We use several metrics to evaluate the performance of each classifier. They provide us with insights into different aspects of the model's effectiveness. On Table *10* you can view a detailed description for each of them.

To evaluate the performance of each classifier we used a Receiver Operating Characteristic (ROC) curve, which is a graphical representation, used on binary classifications, that shows the trade-off between the True Positive Rate (TPR) and the False Positive Rate (FPR).
Further, Area Under the Curve (AUC) measures the area under the ROC curve and evaluates the ability of the classifier to distinguish between classes. AUC values close to 1 indicate excellent performance, while values around 0.5 suggest that the classifier performs no better than random guessing.
Figure *8* is a mirror of the corresponding ROC curves for each model. The baseline represents a random classifier with no ability to distinguish between the positive and negative classes. The baseline connects the points (0,0) and (1,1) in the ROC space and is used as a border for evaluating the discrimination ability of the classifiers.

Metric	Description
Cross-Validation Accuracy (CV)	Gives an estimate of how well the model is expected to perform on new, unseen data by simulating the training process on different subsets of the dataset. The values on Table *11* and *Table 12* represent the mean accuracy, and the standard deviation or error of the accuracy across different folds.
Test Accuracy	Measures how well the model generalizes to new data that it hasn't encountered during training.
Precision	Measures the accuracy of the positive predictions. It is the ratio of true positive predictions to the total predicted positives. High precision indicates a low false positive rate.
Recall	Measures the ability of the classifier to capture all the positive instances. It is the ratio of true positive predictions to the total actual positives.
F1 Score	It provides a balance between precision and recall.
Confusion Matrix	Provides a detailed description of the classifier's predictions, including true positives, true negatives, false positives, and false negatives. It's useful for understanding the types of errors made by the model.

Table 10: Metrics and Descriptions

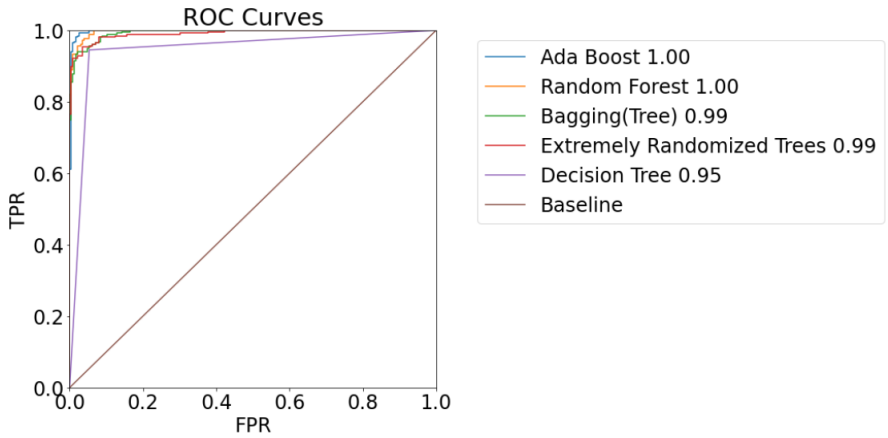

Figure 8: AUC values and the corresponding ROC curves for each classification method

To add more, the plot in Figure 9 provides a visual comparison of cross-validation accuracy scores for different classification methods. Each bar represents the mean accuracy obtained during cross-validation. This graphical representation allows for a quick assessment of the performance of each model in predicting wildfires based on the listed features.

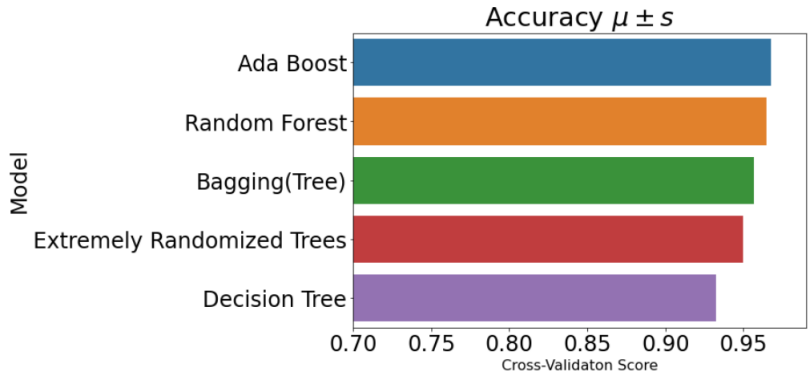

Figure 9: Cross-validation accuracy scores of models.

Below we provide an analysis of the training results (see Table *11* and *Table 12*) for each classification method:

- **Decision Tree:** The Decision Tree classifier performs well with high accuracy and balanced precision and recall. The model shows good generalization to the test set.
- **Bagging Tree:**

  Bagging Tree improves the accuracy compared to the single Decision Tree. It enhances model robustness and reduces overfitting, resulting in higher precision and recall on the test set.
- **Random Forest:**

  Random Forest improves accuracy and maintains a high precision level. It performs well on the test set.
- **Extremely Randomized Trees:**

  Extremely Randomized Trees gives lower accuracy compared to Random Forest. It provides a better balance between precision and recall.
- **Ada Boost:**

  Ada Boost leads to high accuracy and precision, making it one of the top-performing models. It effectively combines weak learners to create a strong ensemble.

Metric	Decision Tree	Bagging (Tree)	Random Forest
Accuracy (CV)	92.2% ± 2.4%	95.3% ± 2.8%	96.3% ± 2.6%
Accuracy (Test)	94.9%	96.8%	97.6%
Precision	94.9%	96.8%	97.6%
Recall	94.5%	95.3%	94.9%
F1 Score	94.7%	96.0%	96.2%

Table 11: Classification Metrics for Decision Tree, Bagging Tree, and Random Forest

Metric	Extremely Randomized Trees	Ada Boost
Accuracy (CV)	95.0% ± 2.2%	97.3% ± 1.6%
Accuracy (Test)	95.6%	97.2%
Precision	95.6%	97.2%
Recall	94.5%	96.5%
F1 Score	95.1%	96.9%

Table 12: Classification Metrics for Extremely Randomized Trees and Ada Boost

Furthermore, based on the listed metrics, we can analyze the confusion matrices.
Ada Boost, the best-performing model, has low false positives and false negatives.
The Random Forest model appears to be the second-best performing model among the ones considered. It has high true positives and true negatives and a low number of false positives.

Method	Decision Tree	Bagging (Tree)	Random Forest
Confusion Matrix	212 13 14 241	217 8 12 243	219 6 13 242

Table 13: Confusion Matrices for Decision Tree, Bagging Tree, and Random Forest

Method	Extremely Randomized Trees	Ada Boost
Confusion Matrix	214 11 14 241	218 7 9 246

Table 14: Confusion Matrices for Extremely Randomized Trees and Ada Boost

4.2.4 Comparing Classifier Performance using Statistical Tests

A pairwise t-test between different models based on cross-validation results is performed. The **ttest_rel** function performs a statistical test, used to produce the p-values of the t-tests.
The created loop prints a message (see figure 10) if the difference between the models is statistically significant. The messages highlight the comparisons where the p-value is less than (1 - confidence_level), suggesting a significant difference with 95% confidence.

```
  Bagging(Tree) vs       Ada Boost => Difference is statistically significant (cf 95.00 p-value=0.0354)
  Random Forest vs Extremely Randomized Trees => Difference is statistically significant (cf 95.00 p-value=0.0118)
Extremely Randomized Trees vs       Ada Boost => Difference is statistically significant (cf 95.00 p-value=0.0031)
```

Figure 10: Ttest conclusions

The results provide insights into the performance of these models, helping to identify which pairs exhibit statistically significant differences in their cross-validation results. They highlight the diversity in the effectiveness of the methods based on the specific metrics we used for evaluating them.

4.2.5 Variable Importance

As a final step, to conclude the purpose of the analysis, we use the **feature_importance_model** dictionary, where each entry contains the model and its associated feature importances. For each model, the features are sorted based on their importance (see figure 11 and figure 12).
A further analysis can be done, in order to explain in deep why the order of the feature importance changes from one method to the other. For this, the variables may be grouped into subgroups by taking into consideration the correlation they might have with each other. Unfortunately, we can not proceed with this analysis, since our knowledge is not complained for this field of study.

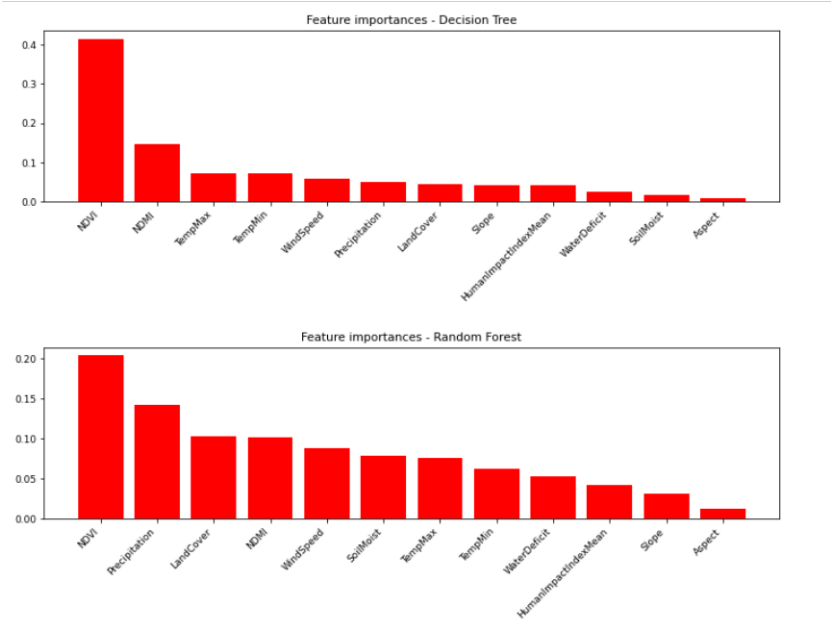

Figure 11: Feature importance plot of Decision Tree and Random Forest

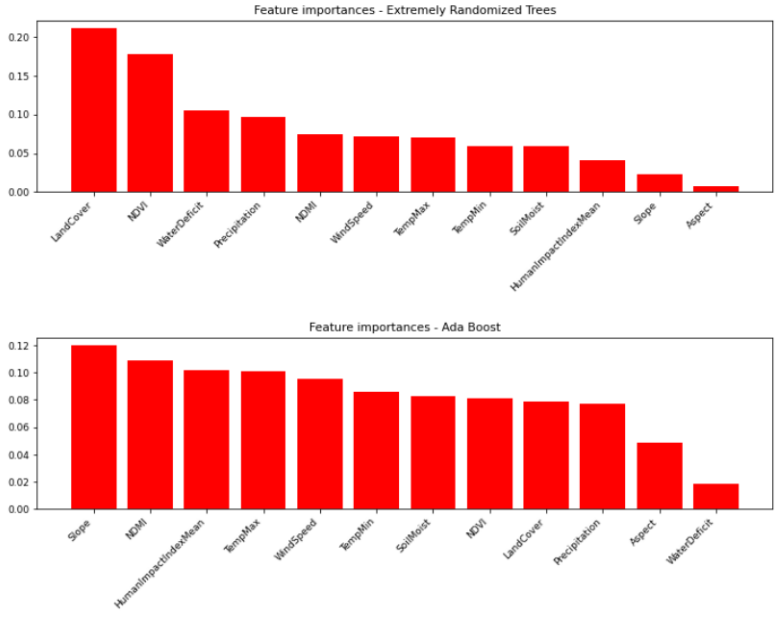

Figure 12: Feature importance plot of Extremely Randomized Trees and Ada Boost

5 Conclusion

The first part of the project was to analyze the existing literature on the factors affecting fires and try to make a summary of all these factors. In a second time, we could find a way to gather all these factors using Google Earth Engine, and we averaged them over the past 10 years.
Two different methods were developed to create random points on the burnt and unburnt areas. These two methods both have three advantages. For the hexagon method (see section 2.2.1) the advantage is one that the scale is easily changeable, and one can consider only a small part of the globe without having to generate points for the whole world. For the mask method (see section 3.2), the advantage consists of a smaller computational time. scale can also be changed in a way to consider only a regional or continental scale.
The difficulty was to find a way to get the value of all factors at the coordinate where the points are located. Once the points were created and the corresponding factor values were extracted, points were then extracted locally with the values of all factors for preprocessing and training the classifier.
Random forest was the classifier giving the best performances among all the classifiers we trained.
Further analysis of the result could show that all the factors that were considered in the analysis have an impact on the prediction of fires, each of them on a different scale of importance.

6 Problems Faced

Throughout the implementation of the project, various challenges were encountered, each requiring unique solutions.
A significant obstacle was encountered in the creation of a hexagon grid on GEE using Python, with difficulties arising from achieving consistent hexagon sizes due to the intricacies of map projections. Despite attempts in Python, QGIS tool was ultimately leveraged to successfully generate a grid with uniform hexagons (see section 2.2.1).

The most challenging obstacle was posed by the limitations of working on a global scale within GEE. To avoid this, a decision was made to narrow the focus to the continent level. This strategic decision proved essential in overcoming the constraints imposed by the extensive geographic area.

The third major hurdle involved the scaling of point sampling for the machine learning algorithm. Balancing the scale was crucial: a larger scale risked clustering points within the same pixel, while a scale identical to the Modis Burned Area map (2) resulted in memory usage errors in GEE. To address this, two distinct approaches were adopted.
In the first approach (see section 3.2), a mask map was created at a larger

scale for sampling, effectively mitigating memory limitations. In the second approach (see section 3.3), a decision was made to opt for a smaller scale to ensure better resolution. Here, sampled points were generated within each hexagon, a more confined area, successfully avoiding the memory constraints.

Another issue appeared when sampling non-burned points, as the Modis Burned Area map lacked values for non-burned regions. To rectify this in the first method, a binary mask for both burned and non-burned pixels was produced, assigning distinct values. In the second method, an image specifically for non-burned hexagons was created, setting their values to 0 for the relevant band.

In conclusion, the approaches adopted not only addressed the technical issues but also optimized the reliability and efficiency of the project implementation.

7 Future Developments

For future developments, we can explore more paths to enhance the project further:

1. **Refinement of Models:**
   Experiment with different algorithms and hyperparameters to improve accuracy and generalization. We can try a different number of trees for the Random Forest algorithm.

2. **Temporal Analysis:**
   Extend the analysis over different temporal scales. Investigate how the factors influencing wildfires evolve over seasons or years. This could provide valuable insights into the dynamics of fire occurrences.

3. **Fine-Tuning Spatial Resolutions:**
   Experiment with different spatial resolutions in your analysis. Fine-tune the hexagon grid or consider other methods to capture variations in factors at different scales, especially for regions with diverse landscapes.

4. **Incorporation of Additional Factors:**
   Explore the inclusion of additional factors that may influence wildfires. This could involve obtaining and integrating new Earth observation datasets or incorporating socio-economic and demographic variables for a more comprehensive analysis.

5. **Integration with Real-Time Data:**
   Explore possibilities for integrating real-time data sources to enable near-real-time monitoring of wildfire events.

6. **Climate Change Impacts:**
   Investigate the potential impacts of climate change on wildfire occurrences. Analyze long-term trends and assess how changing climatic conditions may influence the factors driving wildfires and how these factors may be linked to each other.

Acknowledgments

We would like to express our sincere gratitude to Dr. Guido Ceccherini, devoted tutor and organizer of the course "Introduction to Google Earth Engine for advanced Geo-spatial Analysis" for the time and support he dedicated us throughout this project. His guidance, his patience and his deep knowledge of Google Earth-Engine helped us in all the steps for the development of this project.

Our deep thanks go also to our supervisors for this project, Prof. Stroppiana and Prof. Venuti, who could guide us throughout this project and help us choosing the best direction for our research.

Eventually, we present our special acknowledgement to Prof. Fugini for organising this course and providing us with the opportunity to learn from such an interesting project.

8 Bibliography

References

[1] N. G. S. F. Center. Nasa summer 2023 temperature media resources, 2023. URL https://svs.gsfc.nasa.gov/14407.

[2] M. C. Hansen, P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, D. Thau, S. V. Stehman, S. J. Goetz, T. R. Loveland, A. Kommareddy, A. Egorov, L. Chini, C. O. Justice, and J. R. G. Townshend. High-resolution global maps of 21st-century forest cover change. *Science*, 342(6160):850–853, 2013. doi: 10.1126/science.1244693. URL https://www.science.org/doi/abs/10.1126/science.1244693.

[3] M. A. Krawchuk, M. A. Moritz1, M.-A. Parisien, J. V. Dorn, and K. Hayhoe. Global pyrogeography: the current and future distribution of wildfire. *PLoS ONE*, 4, 2009. doi: 10.1126/science.1244693. URL https://doi.org/10.1371/journal.pone.0005102.

[4] N. Lopez. 2023: A year of intense global wildfire activity, 2023. URL https://atmosphere.copernicus.eu/copernicus-canada-produced-23-global-wildfire-carbon-emissions-2023.

[5] A. J. McCullum, J. L. Torres-Pérez, Q. assistance from Britnay Beaudry, and H. Pippin. Using earth observations for pre- and post-fire monitoring, 2022. URL https://appliedsciences.nasa.gov/get-involved/training/english/arset-using-earth-observations-pre-and-post-fire-monitoring.