Predicting Forest Fires

Group - 5

Abhishek Premnath (862323549)

Balaji Arunachalam (862321425)

Madhusudhan Tungamitta (862325125)

Shreya Godishala (862313765)

Thirumalai Vinjamoor Akhil Srinivas (862320790)

ABSTRACT

Our project comprises gathering and processing historical records of fire incidents in order to study the fire events and discover the patterns and links with multiple characteristics, including wind speed, air temperature, humidity, precipitation, Rain, etc. The two major objectives of the study will be to predict the predicted fire perimeter and to pinpoint in the map. With the use of this information, we can locate any potential risk areas and, if required, evacuate. In addition, the project will construct and project forest fire hotspots onto a geographical map.

INTRODUCTION

Forest Fires is becoming a bigger problem and recent survey shows that atleast in portugal more than **300 people died** from forest fires spread from year 2016(Click here), and also causing damage to the local economy and environment. Large amounts of environmental harm were done as a result of the burning of more than tenth of an acre of land. Wildfire spread is challenging to predict since it depends on so many variables with intricate relationships. Our project task consists of analysing the data of Montesinho Natural Park Forest area includes x and y spatial coordinates of the map and the factors that affect such as a month, temperature, humidity and DMC, DM by Fire weather index system, this analysis will help in predicting the fire area and our project will also include project the predicted forest fire area in map. The main difficulties we encountered in this project were initially with gathering the weather data. Although we found many real-time weather datasets related to wildfires, it is difficult to find one that contained the necessary x and y spatial coordinates to map them with our fire dataset. To get around this, we discovered the open-source dataset of Portugal's Montesinho Natural Park from the **UC Irvine Machine Learning Repository(Link)**

PROBLEM STATEMENT

Forest fires which are causing the extinction of some species, disrupting water cycles, and taking the lives of residents, are one of the main environmental challenges seen in recent years. We are attempting to predict the area that is burned due to forest fires. It would be quite advantageous to estimate and project the extent of the fire beforehand.

Data Collection

The spread of wildfire is difficult to analyze, and it is very complex to predict because it depends on many parameters. We need information related to fire location, topography of the surrounding area, weather, and environment conditions in order to complete our task. The for the Portugal's Montesinho Natural Park have taken from the UC Irvine Machine Learning Repository and its attributes consist of X and Y spatial coordinates, humidity and FFMC, DMC, DC, location, etc.

Data Cleaning

- Removed unnecessary or irrelevant observations
- Structural errors have been rectified, and any data that had the characters "-", "NA" or "" have been corrected by giving them a default value.
- Records that lacked necessary information were deleted.

Data Processing

For the appropriate locations of the data collected for our model from University of California, Irvine's contains, the dimensions of the dataset had to be reduced and the null values were addressed by either removing the rows associated with it entirely or taking the average values from similar records. Also, to reduce the complexity of the dataset, certain attributes such as rain and relative humidity were combined.

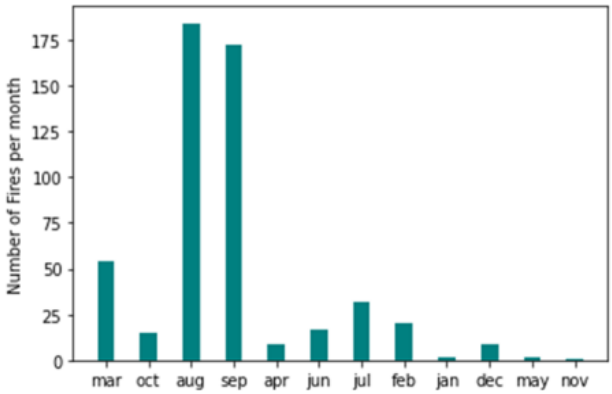
X and Y spatial coordinates within Montesinho park map have also been provided on a scale of 1-9. These values would also be normalized prior to training the model and it also has the attributes such as rain and relative humidity combined and so it made our task easier.

Data Attributes

Attributes	Explanation
X	x-axis spatial coordinate within the Montesinho park map
Y	y-axis spatial coordinate within the Montesinho park map
month	month of the year
day	day of the week
FFMC	Fine Fuel Moisture Code index from the Fire Weather Index system
DMC	Duff Moisture Code index from the Fire Weather Index system
DC	drought code index from the Fire Weather Index system
ISI	Initial spread index from the Fire Weather Index system
temp	temperature in Celsius degrees
RH	relative humidity in %
wind	wind speed in km/h
rain	outside rain in mm/m2
area	the burned area of the forest (in ha)

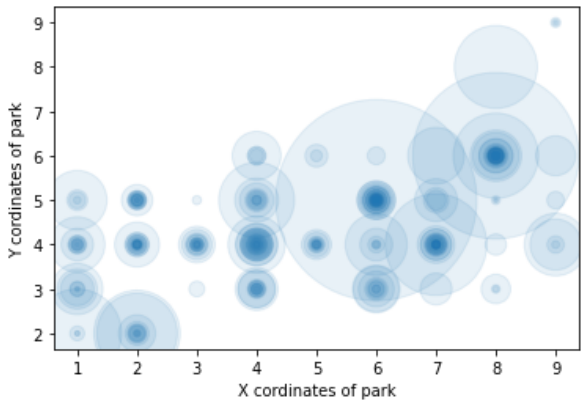
Handling Spatial Data

To get a better understanding of the dataset used and to find the underlying corelation between various attributes used. Some correlated attributes are Duff Moisture Code (DMC) with respect to humidity and the Fire Weather Index (FWI) with respect to the Fine Fuel Moisture Code (FFMC) As a part of initial visualization we are able to see that most of the forest fires happens in the months of August and September months.



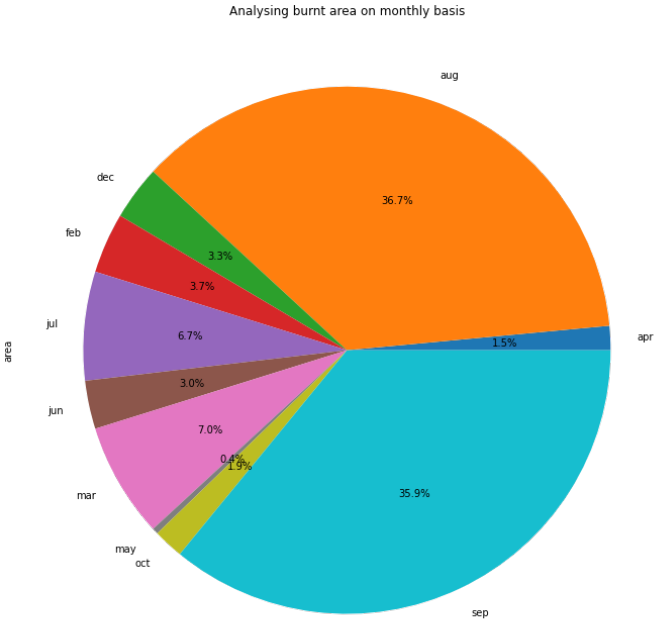
Visualization of the Area Burnt in the X, Y Co-ordinates

BY Visualizing the burnt "area" we observe that there are many hotspots in the maps and it also show the area spread accross and it doesn't spread much



Burnt Area with respect to Months

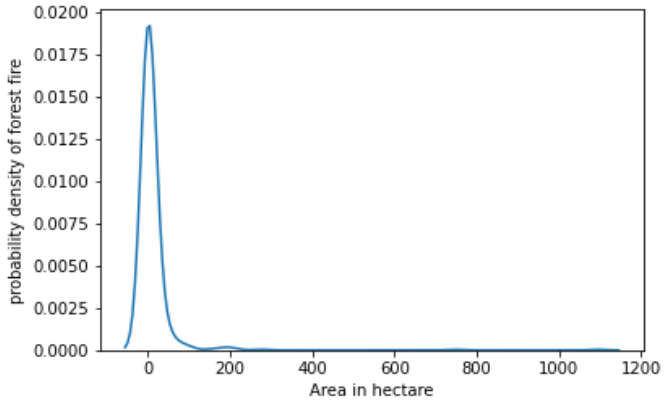
1. Here Months are January to December and we can clearly see that In the months of August and September high area is burnt and it is very severe in those months.
2. we can see that top 4 high temperature also recorded in above those months itself.
3. we can judge, low humidity may also be one of the reasons for the Forest Fires.



	mean					
	DMC	FFMC	RH	rain	temp	wind
month						
1	2.400000	50.400000	89.000000	0.000000	5.250000	2.000000
2	9.475000	82.905000	55.700000	0.000000	9.635000	3.755000
3	34.542593	89.444444	40.000000	0.003704	13.083333	4.968519
4	15.911111	85.788889	46.888889	0.000000	12.044444	4.666667
5	26.700000	87.350000	67.000000	0.000000	14.650000	4.450000
6	93.382353	89.429412	45.117647	0.000000	20.494118	4.135294
7	110.387500	91.328125	45.125000	0.006250	22.109375	3.734375
8	153.732609	92.336957	45.489130	0.058696	21.631522	4.086413
9	120.922674	91.243023	42.843023	0.000000	19.612209	3.557558
10	41.420000	90.453333	37.466667	0.000000	17.093333	3.460000
11	3.000000	79.500000	31.000000	0.000000	11.800000	4.500000
12	26.122222	84.966667	38.444444	0.000000	4.522222	7.644444

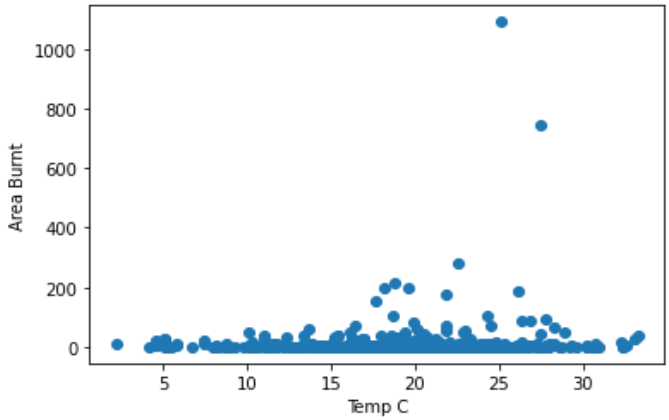
Probability Density of Forest Fire

So, what does the above picture shows us, It shows that forest fires is not spreading very much and it is spreading less than the 200 hectare area, Below Visualization will also shows the area burnt

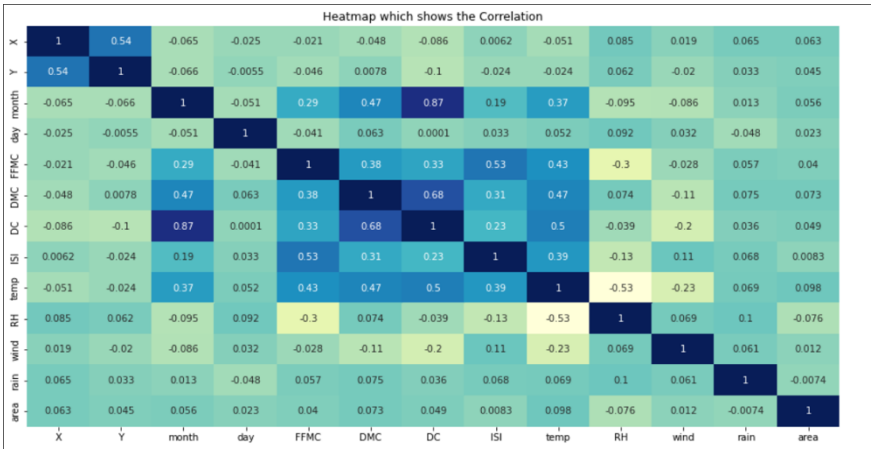


Burnt Area with respect to Temperature

By Visualizing the burnt "area" we observe that there are many hotspots in the maps, and it also show the area spread across and it doesn't spread much so we decided to drop the temperature



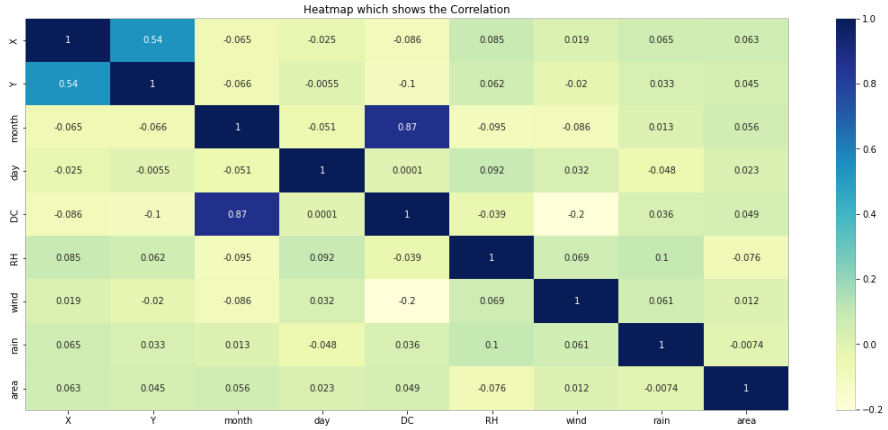
Correlation Matrix:



To determine which traits are associated to one another, a correlation matrix was built. The correlation matrix shows a strong association between the day's average temperature, maximum temperature, and minimum temperature.

So we after seeing the above correlation map we decided to drop DMC, ISI, FFMC, Temp because we see it have very less correlation with respect to the area

New Correlation:



Evaluation and Training of Models

We tried different machine learning algorithms such as SVM, Decision Tree Classifier, Random Forest Classifier, Gaussian NB, Voting Classifier using **area as testing data** by dropping **DMC, ISI, FFMC, Temp.**

Training and Testing Data: We used sklearn to train and test the data where formatted our data because we cannot use the string(month and day feature) and also we can see we have drop the area feature since we are going to use it as testing data

```
from sklearn.model_selection import train_test_split
X=df
X_new_month=recon(X)
X['month']=X_new_month
X_new_days=recon2(X)
X['day']=X_new_days
y = X['area']
y1=ar_cat(X)
y=y1
X = X.drop('area', axis=1)
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, shuffle=True, random_state=1)
```

SVM: Support vector machines are supervised machine learning algorithm which helps in analysing and classifying the data we got F1 score for the first classifier for the svm about 64 for testing data

```
from sklearn import svm
model = svm.SVC(kernel='poly')
model.fit(X_train,y_train)

test_predicted = model.predict(X_test)
train_predicted = model.predict(X_train)

from sklearn.metrics import accuracy_score

train_accuracy = f1_score(y_train, train_predicted,average=None)
test_accuracy = f1_score(y_test, test_predicted,average=None)
print('SVM training accuracy is', train_accuracy)
print('SVM test accuracy is', test_accuracy)
```

```
SVM training accuracy is [0.64540338 0.         0.         0.        ]
SVM test accuracy is [0.64935065 0.         0.         0.        ]
```

Decision Tree Classifier: It is one of the most powerful algorithm and we got an F1 score of 1st classifier of 54 for testing data

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier(max_depth=7 ,min_samples_leaf=6 ,min_samples_split=4)
model.fit(X_train,y_train)
test_predicted = model.predict(X_test)
train_predicted = model.predict(X_train)

from sklearn.metrics import accuracy_score

train_accuracy = f1_score(y_train, train_predicted,average=None)
test_accuracy = f1_score(y_test, test_predicted,average=None)
print('DecisionTreeClassifier training accuracy is', train_accuracy)
print('DecisionTreeClassifier test accuracy is', test_accuracy)
```

```
DecisionTreeClassifier training accuracy is [0.72727273 0.13953488 0.         0.57959184]
DecisionTreeClassifier test accuracy is [0.54644809 0.         0.         0.28865979]
```

Random Forest Classifier:

It is an assembling method which operates by constructing a multi decision trees and it got an 61% F1 score for testing data

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators=100, max_depth=2, random_state=0)
model.fit(X_train,y_train)
test_predicted = model.predict(X_test)
train_predicted = model.predict(X_train)

from sklearn.metrics import accuracy_score
train_accuracy = f1_score(y_train, train_predicted,average=None)
test_accuracy = f1_score(y_test, test_predicted,average=None)
print('RandomForestClassifier training accuracy is', train_accuracy)
print('RandomForestClassifier test accuracy is', test_accuracy)
```

```
RandomForestClassifier training accuracy is [0.68041237 0.         0.         0.36756757]
RandomForestClassifier test accuracy is [0.61244019 0.         0.         0.10526316]
```

Gaussian NB:

It works based on Navies bayes with strong independence assumption and Testing accuracy of 49 for the one of the classifier

```
from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train, y_train)
final = model.predict(X_test)

test_predicted = model.predict(X_test)
train_predicted = model.predict(X_train)

from sklearn.metrics import accuracy_score
train_accuracy = f1_score(y_train, train_predicted,average=None)
test_accuracy = f1_score(y_test, test_predicted,average=None)
print('GaussianNB training accuracy is', train_accuracy)
print('GaussianNB test accuracy is', test_accuracy)
```

```
GaussianNB training accuracy is [0.17525773 0.04761905 0.13793103 0.55579869]
GaussianNB test accuracy is [0.13636364 0.09090909 0.11111111 0.48913043]
```

Voting Classifier:

The voting classifier aggregates the results from the above classifier and takes the majority vote to decide if a particular object belongs to a class or not.

```
from sklearn.ensemble import VotingClassifier
from sklearn import svm
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
m1= GaussianNB()
m2=svm.SVC(kernel='poly')
m3=RandomForestClassifier(n_estimators=100, max_depth=2, random_state=0)
m4=DecisionTreeClassifier(max_depth=7 ,min_samples_leaf=6 ,min_samples_split=4)
eclf1 = VotingClassifier(estimators=[ ('gnb', m1), ('rf', m3), ('dtc', m4),('svm',m2)], voting='hard')
eclf1.fit(X_train, y_train)
test_predicted = eclf1.predict(X_test)
train_predicted = eclf1.predict(X_train)

from sklearn.metrics import accuracy_score
train_accuracy = accuracy_score(y_train, train_predicted)
test_accuracy = accuracy_score(y_test, test_predicted)
print('Ensemble training accuracy is', train_accuracy)
print('Ensemble test accuracy is', test_accuracy)
```

```
Ensemble training accuracy is 0.5373961218836565
Ensemble test accuracy is 0.4423076923076923
```

Experimenting Other Spatial Features:

In this attempt we dropped the area feature because we just wanted to explore other spatial features, and we do not concentrate too heavily on area feature itself. So we instead turn our attention to the df2 correlation heatmap. We can see here that DC may be a good predictor of Rain, and we test to see this

```
df2=df.drop(['DMC','ISI','FFMC','temp'],axis =1)
from pandas._libs.algos import diff_2d
from sklearn.model_selection import train_test_split
X=df2
X_new_month=recon(X)
X['month']=X_new_month
X_new_days=recon2(X)
X['day']=X_new_days
y = X['rain']

X = X.drop('rain', axis=1)
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, shuffle=True, random_state=1)
```

With this new alteration we tried voting classifier for finding the best values

```
from sklearn.ensemble import VotingClassifier
from sklearn import svm
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
m1= GaussianNB()
m2=svm.SVC(kernel='poly')
m3=RandomForestClassifier(n_estimators=100, max_depth=2, random_state=0)
m4=DecisionTreeClassifier(max_depth=7 ,min_samples_leaf=6 ,min_samples_split=4)
eclf1 = VotingClassifier(estimators=[ ('gnb', m1), ('rf', m3), ('dtc', m4),('svm',m2)], voting='hard')
eclf1.fit(X_train, y_train.astype(str))
test_predicted = eclf1.predict(X_test)
train_predicted = eclf1.predict(X_train)

from sklearn.metrics import accuracy_score
train_accuracy = accuracy_score(y_train.astype(str), train_predicted)
test_accuracy = accuracy_score(y_test.astype(str), test_predicted)
print('Ensemble training accuracy is', train_accuracy)
print('Ensemble test accuracy is', test_accuracy)
```

```
Ensemble training accuracy is 0.9889196675900277
Ensemble test accuracy is 0.9743589743589743
```

We got an super good results of the accuracy of 97 but we unfortunately this one got overfitted because of the spare data so we tried looking for solution of altering our data.

```
print(np.unique(df['DC']))
```

```
[  7.9   9.3  15.3  15.5  15.8  16.2  18.7  25.6  26.6  28.3  30.2  30.6
  32.1  34.   36.9  41.1  41.6  43.   43.5  43.6  46.7  48.3  52.8  55.
  55.2  56.9  57.3  58.3  64.7  67.6  70.8  73.7  74.3  77.5  80.8  83.7
  85.3  86.6  87.2  89.4  92.4  94.3  97.1  97.8 100.4 100.7 102.2 103.8
 106.7 113.8 171.4 200.  229.  232.1 233.8 252.6 290.8 296.3 297.7 298.1
 309.9 313.4 316.7 349.7 350.2 352.  352.6 353.5 354.6 355.2 366.7 368.3
 376.6 377.2 395.  411.8 423.4 424.1 430.8 431.6 433.3 437.7 440.9 442.1
 442.9 450.2 458.8 466.3 466.6 474.9 480.8 488.  495.6 503.6 513.3 520.5
 529.8 537.4 542.  550.3 560.  561.6 565.5 567.2 570.5 573.  575.8 578.8
 581.1 586.7 587.1 589.9 594.2 596.3 601.4 605.3 605.8 607.1 608.2 609.6
 613.  614.5 614.7 621.7 624.1 624.2 629.1 631.2 633.6 635.9 638.8 643.
 647.1 649.9 654.1 658.2 661.3 661.8 664.2 664.5 665.3 665.6 666.7 668.
 669.1 671.2 671.9 672.6 673.8 674.4 680.7 680.9 682.6 684.4 685.2 686.5
 686.9 689.1 690.  691.8 692.3 692.6 694.8 696.1 698.6 699.6 700.7 704.4
 706.4 706.6 706.7 706.8 709.9 713.  713.9 714.3 715.1 718.3 721.1 721.4
 723.1 724.3 725.1 726.9 728.6 730.2 730.6 731.7 732.3 735.7 738.1 739.4
 744.4 745.3 750.5 751.5 752.6 753.8 758.1 764.  768.4 770.3 777.1 783.5
 789.7 795.3 795.9 803.3 807.1 811.2 812.1 817.5 819.1 822.8 825.1 844.
 849.3 855.3 860.6]
```

So we tried looking into labels and we find out the DC features has more labels and it seems to be causing trouble

```
[ ]  print(np.unique(df['rain']))

     [0.   0.2 0.4 0.8 1.   1.4 6.4]
```

We have 7 classes, so it seems easier to predict the values of Rain. So we flip the test.

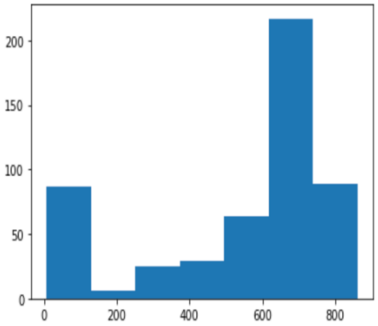
```
from sklearn.ensemble import VotingClassifier
from sklearn import svm
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
m1= GaussianNB()
m2=svm.SVC(kernel='poly')
m3=RandomForestClassifier(n_estimators=100, max_depth=2, random_state=0)
m4=DecisionTreeClassifier(max_depth=7 ,min_samples_leaf=6 ,min_samples_split=4)
eclf1 = VotingClassifier(estimators=[ ('gnb', m1), ('rf', m3), ('dtc', m4),('svm',m2)], voting='hard')
eclf1.fit(X_train, y_train.astype(str))
test_predicted = eclf1.predict(X_test)
train_predicted = eclf1.predict(X_train)

from sklearn.metrics import accuracy_score
train_accuracy = accuracy_score(y_train.astype(str), train_predicted)
test_accuracy = accuracy_score(y_test.astype(str), test_predicted)
print('Ensemble training accuracy is', train_accuracy)
print('Ensemble test accuracy is', test_accuracy)
```

```
Ensemble training accuracy is 0.48476454293628807
Ensemble test accuracy is 0.15384615384615385
```

This is another expected problem when you have something of the order of a 100 classes and sparse data.

```
plt.hist(df['DC'],bins=7)
```

```
(array([ 87.,   6.,  25.,  29.,  64., 217.,  89.]),
 array([  7.9       , 129.71428571, 251.52857143, 373.34285714,
        495.15714286, 616.97142857, 738.78571429, 860.6       ]),
 <a list of 7 Patch objects>)
```




```
def reset(x):
  z=[]
  for i in x['DC']:
    if i>0 and i<=7.9:
      z.append(7.9)
    elif 129.714>=i>7.9:
      z.append(129.714)
    elif 251.52857143>=i> 129.714:
      z.append(251.52857143)
    elif 373.34285714>=i> 251.52857143:
      z.append(373.34285714)
    elif  495.15714286>=i>373.34285714:
      z.append( 495.15714286)
    elif 616.97142857>=i> 495.15714286:
      z.append(616.97142857)
    elif 738.78571429>=i>616.97142857:
      z.append(738.78571429)
    else:
      z.append( 860.6 )
  return z
```

So now we turn this multiclass problem of DC into same range as Number of rains (Ex 7 labels)

```
from sklearn.ensemble import VotingClassifier
from sklearn import svm
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
m1= GaussianNB()
m2=svm.SVC(kernel='poly')
m3=RandomForestClassifier(n_estimators=100, max_depth=2, random_state=0)
m4=DecisionTreeClassifier(max_depth=7 ,min_samples_leaf=6 ,min_samples_split=4)
eclf1 = VotingClassifier(estimators=[ ('gnb', m1), ('rf', m3), ('dtc', m4),('svm',m2)], voting='hard')
eclf1.fit(X_train, np.array(y_train).astype(str))
test_predicted = eclf1.predict(X_test)
train_predicted = eclf1.predict(X_train)

from sklearn.metrics import accuracy_score
train_accuracy = accuracy_score(np.array(y_train).astype(str), train_predicted)
test_accuracy = accuracy_score(np.array(y_test).astype(str), test_predicted)
print('Ensemble training accuracy is', train_accuracy)
print('Ensemble test accuracy is', test_accuracy)
```

```
Ensemble training accuracy is 0.6925207756232687
Ensemble test accuracy is 0.5576923076923077
```

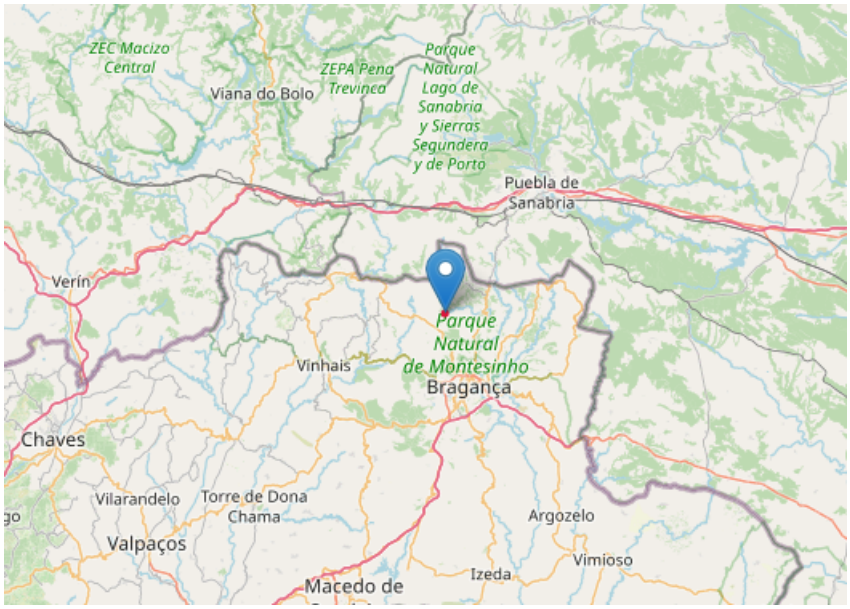
We are just trying to test when looking for accurate features. Correlation is key here and the higher the positive or lower the negative the better the feature.

Why Accuracy is low for the Area but not for Rain:

So previously what we are trying to explain with the above experiment of **'rain'** feature with **'DC'** is we observed from the correlation map **'rain'** and **'DC'** has more correlation and it got good accuracy but when it comes to **'area'** feature no other features has good correlation so with the help of our results we can say it is not easily possible to get good accuracy for predicting area.

Outcome:

Outcome of our project we have found out the spreading point of forest fire in Montesinho Natural Park and we have plotted the point in the map.



Demo in Colab

Taxonomy:

Model	Spatial Component Used	Dataset
Logistic Regression, Boosting Model, Random Forest	Geographical Information System	Aladin and MODIS satellite data(2a)
SVM, KNN, K-SVM	Spatial data on GPU, Spatial Clustering	spatiotemporal datasets, geospatial datasets(6a)

Related Work:

Item	Topic	Link	Type (Academic/Non-Academic)
1	A Data Mining Approach to Predict Forest Fires using Meteorological Data	link	Academic
2	Learning to predict forest fires with different data mining techniques	link	Academic
3	Predicting Forest Fire Using Remote Sensing Data and Machine Learning	link	Academic
4	Are forest fires predictable?	link	Academic
5	Prediction of Forest Fire Using Machine Learning Algorithms: The Search for the Better Algorithm	link	Academic
6	A review of machine learning applications in wildfire science and management	link	Academic
7	Predicting Burned Areas of Forest Fires: An Artificial Intelligence Approach	link	Academic
8	How do wildfire start, and can we predict them?	link	Non-Academic
9	2 Ways in Which Machine Learning Combats Forest Fires	link	Non-Academic
10	Predicting Brazil's Wildfire Patterns in 2020	link	Non-Academic
11	Fighting the Amazon Forest fires with advanced analytics	link	Non-Academic
12	Fighting fire with machine learning: two students use TensorFlow to predict wildfires	link	Non-Academic
13	Leveraging Machine Learning to predict wildfires using PyTorch Lightning	link	Non-Academic

Conclusion

The project's goal is to evaluate and forecast the spread of fires using machine learning models developed on based on the historical fire occurrences. We also learn how to handle spatial data and to make use of it.

Future Works

We intend to use feature engineering to supplement our model and find ways to increase the reliability of sparse data.

Reference

- Forest Fire prediction using Machine Learning https://www.analyticsvidhya.com/blog/2021/10/forest-fire-prediction-using-machine-learning/
- Prediction of Forest Fire Using Machine Learning Algorithms: The Search for the Better Algorithm https://ieeexplore.ieee.org/document/9719887
- Predict forest fires using artificial intelligence https://www.neuraldesigner.com/learning/examples/forest-fires
- Predicting Forest Fire Using Remote Sensing Data And Machine Learning https://www.researchgate.net/publication/347395483_Predicting_Forest_Fire_Using_Remote_Se nsing_Data_And_Machine_Learning
- Predictive modeling of wildfires: A new dataset and machine learning approach https://www.sciencedirect.com/science/article/abs/pii/S0379711218303941

- The Experiment of Forest Fire Prediction using Deep Learning https://medium.com/mlearning-ai/the-experiment-of-forest-fires-prediction-using-deep-learning-d537e8c8e3a2

- Folium : https://python-visualization.github.io/folium/ - :~:text=folium makes it easy to,as markers on the map.