*A CAL Project Report*

on

**FOREST COVER TYPE PREDICTION**

*to be submitted in partial fulfilling of the requirements for the course on*

**Data Mining and Business Intelligence – ITA5007**

**(B1+TB1 )**

by

**KIRAN SHINDE 21MCA0277**

**AYUSHI KUMARI 21MCA0288**

**DIVYA KUMARI 21MCA0219**



Winter Semester 2021-2022

**TABLE OF CONTENTS**

ABSTRACT

1. Introduction ……………………………………………………………. 04
2. Review 1 (Survey, Analysis)...………………………………….……… 05
   1. Problem definition
   2. Dataset Description
   3. Review on Existing System
   4. Proposed System
3. Review 2 (Design)…………………….…………………………………08
   1. Methodology
      1. Module Description
         1. Data exploration
         2. Pre-processing
      2. Algorithms used
         1. Justification for choosing the models
4. Review 3 (Code)…………………………………..……………………..10
   1. Implementation
      1. Software description
      2. hardware description
      3. Code
      4. Output Screenshots
      5. The histogram
5. Conclusion ………………………………………………………....…… 22
6. References ………………………………………………………………. 22

**ABSTRACT**

The project is based upon data mining techniques for predicting the cover type of forest with the datasets provided. We have predicted cover type of forest from the cartographic variables i.e., visible variables. The actual mapping of forest cover type for a given 30 x 30 meter cell was determined from US Forest Service (USFS) Region 2 Resource Information System data. Independent variables were then derived from data obtained from the US Geological Survey and USFS. The study area includes four wilderness areas located in the Roosevelt National Forest of northern Colorado and seven different cover types. These areas represent forests with minimal human activities, so that existing forest cover types are more a result of ecological processes rather than forest management practices.

1. **INTRODUCTION**

Identifying forest cover types is often necessary for forest management departments so that they could keep an eye on the ecological balance of an area. Often, forest cover types are identified using field personnel or by remote sensing techniques which are most of the times, both time-consuming and expensive. Moreover, applying these techniques is not practical when area being studied is remote or in another state or territory. In these cases, predictive models trained on existing data can be quite useful as they can approximate cover types for unseen data.

The data belongs to four wilderness areas of Roosevelt National Forest namely Rawah, Comanche Peak, Neota and Cache la Poudre. The advantage of using this data is that it belongs to an area that has been untouched by human interaction. The dataset has also been chosen because it presents a complex problem to solve as it contains substantial number of instances i.e. 5 65892(test dataset) and 15120(train dataset). This means that network could be trained well and following that tested well. The sever categories numbered from 1 to 7 in the cover\_type column to be classified as Spruce/Fir, Lodgepole Pine, Ponderosa Pine, Cottonwood/Willow, Aspen, Douglas-fir, Krummholz respectively.

We have used ExtraTreesClassifier, under skcit-learn library which is a type of ensemble learning technique which aggregates the result of multiple de-correlated decision trees collected in a “forest” to input its classification result.

1. **REVIEW-1 (Survey & Dataset Collection)**
   1. **Problem definition:**

Predicting a forest cover type can be helpful in many ways. Identifying forest cover types is often necessary for forest management departments so that they could keep an eye on the ecological balance of an area. The prediction can be helpful to keep a record of which tree is best suited in which region so that maximum of output can be received.

* 1. **Dataset Description:**

The dataset is collected from : <https://www.kaggle.com/competitions/forest-cover-type-prediction/data> which are as follows:

Data Fields:

Elevation - Elevation in meters  
Aspect - Aspect in degrees azimuth  
Slope - Slope in degrees  
Horizontal\_Distance\_To\_Hydrology - Horz Dist to nearest surface water features  
Vertical\_Distance\_To\_Hydrology - Vert Dist to nearest surface water features  
Horizontal\_Distance\_To\_Roadways - Horz Dist to nearest roadway  
Hillshade\_9am (0 to 255 index) - Hillshade index at 9am, summer solstice  
Hillshade\_Noon (0 to 255 index) - Hillshade index at noon, summer solstice  
Hillshade\_3pm (0 to 255 index) - Hillshade index at 3pm, summer solstice  
Horizontal\_Distance\_To\_Fire\_Points - Horz Dist to nearest wildfire ignition points  
Wilderness\_Area (4 binary columns, 0 = absence or 1 = presence) - Wilderness area designation  
Soil\_Type (40 binary columns, 0 = absence or 1 = presence) - Soil Type designation  
Cover\_Type (7 types, integers 1 to 7) - Forest Cover Type designation

The wilderness areas are:

1 - Rawah Wilderness Area  
2 - Neota Wilderness Area  
3 - Comanche Peak Wilderness Area  
4 - Cache la Poudre Wilderness Area

The soil types are:

1 Cathedral family - Rock outcrop complex, extremely stony.  
2 Vanet - Ratake families complex, very stony.  
3 Haploborolis - Rock outcrop complex, rubbly.  
4 Ratake family - Rock outcrop complex, rubbly.  
5 Vanet family - Rock outcrop complex complex, rubbly.  
6 Vanet - Wetmore families - Rock outcrop complex, stony.  
7 Gothic family.  
8 Supervisor - Limber families complex.  
9 Troutville family, very stony.  
10 Bullwark - Catamount families - Rock outcrop complex, rubbly.  
11 Bullwark - Catamount families - Rock land complex, rubbly.  
12 Legault family - Rock land complex, stony.  
13 Catamount family - Rock land - Bullwark family complex, rubbly.  
14 Pachic Argiborolis - Aquolis complex.  
15 unspecified in the USFS Soil and ELU Survey.  
16 Cryaquolis - Cryoborolis complex.  
17 Gateview family - Cryaquolis complex.  
18 Rogert family, very stony.  
19 Typic Cryaquolis - Borohemists complex.  
20 Typic Cryaquepts - Typic Cryaquolls complex.  
21 Typic Cryaquolls - Leighcan family, till substratum complex.  
22 Leighcan family, till substratum, extremely bouldery.  
23 Leighcan family, till substratum - Typic Cryaquolls complex.  
24 Leighcan family, extremely stony.  
25 Leighcan family, warm, extremely stony.  
26 Granile - Catamount families complex, very stony.  
27 Leighcan family, warm - Rock outcrop complex, extremely stony.  
28 Leighcan family - Rock outcrop complex, extremely stony.  
29 Como - Legault families complex, extremely stony.  
30 Como family - Rock land - Legault family complex, extremely stony.  
31 Leighcan - Catamount families complex, extremely stony.  
32 Catamount family - Rock outcrop - Leighcan family complex, extremely stony.  
33 Leighcan - Catamount families - Rock outcrop complex, extremely stony.  
34 Cryorthents - Rock land complex, extremely stony.  
35 Cryumbrepts - Rock outcrop - Cryaquepts complex.  
36 Bross family - Rock land - Cryumbrepts complex, extremely stony.  
37 Rock outcrop - Cryumbrepts - Cryorthents complex, extremely stony.  
38 Leighcan - Moran families - Cryaquolls complex, extremely stony.  
39 Moran family - Cryorthents - Leighcan family complex, extremely stony.  
40 Moran family - Cryorthents - Rock land complex, extremely stony.

* 1. **Review on Existing System**

In the existing system there is a formula to predict the cover type using RandomForestClassifier and accuracy based on provided attributes. These techniques won’t give accurate results because allows uders to bootstrap replicas, but by default it uses the entire input sample. This may increase variance because bootstrapping makes it more diversified. There are very less systems that predict cover type and accuracy.

**d. Proposed System**

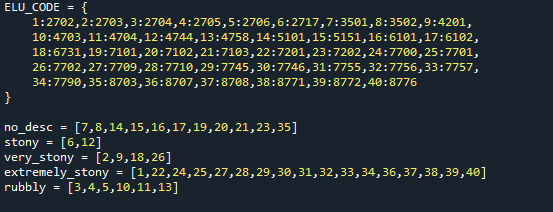
The proposed system uses Extra Trees Classifiers for cover type prediction.. And then we perform Data Cleaning followed by Data Preprocessing. Data Analysis and Visualization is done to make better understanding of the results and exploring various valuable insights.

1. **REVIEW-2 (Design)**
2. **Methodology:**

The proposed method consists of modules like loading the dataset, data exploration pre-processing etc. In light of 565892 data from our dataset, mainly to point on prediction algorithm that considers the results of various trees having “ID” and Cover Type. the prediction is of which tree id falls on which cover type.

1. **Data Exploration:**

It mostly concerns the gathering of the appropriate dataset. A variety of filters must be applied to the dataset that will be used to produce Cover Type Prediction. Data collection also helps to improve the dataset by incorporating more external data. We'll start by examining the Kaggle dataset, and then we'll use the model with the data to examine the predictions accurately based on the accuracy.



1. **Data Preprocessing:**

As of our data is preprocessed still to avoid any discrepancy we used a method to preprocess the data.



1. **Algorithm Used:**

We have used ExtraTreesClassifier. under skcit-learn library which is a type of ensemble learning technique which aggregates the result of multiple de-correlated decision trees collected in a “forest” to input its classification result.

Advantages:

1. Extra Trees uses the whole original sample. In the extra trees sklearn implementation there is an optional parameter that allows user to bootstrap replicas, but by default,it uses the entire input sample.
2. ExtraTrees chooses the selection cut points randomly , in order to split nodes
3. ExtraTrees add randomization but still has optimization.

1. **REVIEW-3 (Code)**
   1. **Hardware Description**

Processor – Intel(R) Core(TM) i3-7th Gen

SDD – 120GB

RAM – 12GB

* 1. **Software Description**

Python – 3.9.12

SPYDER

* 1. **CODE:**

import pandas as pd

import math

import numpy as np

import matplotlib.pyplot as plt

from sklearn.ensemble import ExtraTreesClassifier

from sklearn.model\_selection import cross\_val\_score

from sklearn.metrics import confusion\_matrix,accuracy\_score,f1\_score, precision\_score,recall\_score,roc\_auc\_score

data = pd.read\_csv(r'C:\Users\Ayushi\Desktop\train.csv', index\_col="Id")

data\_test = pd.read\_csv(r'C:\Users\Ayushi\Desktop/test.csv', index\_col="Id")

ELU\_CODE = {

1:2702,2:2703,3:2704,4:2705,5:2706,6:2717,7:3501,8:3502,9:4201,

10:4703,11:4704,12:4744,13:4758,14:5101,15:5151,16:6101,17:6102,

18:6731,19:7101,20:7102,21:7103,22:7201,23:7202,24:7700,25:7701,

26:7702,27:7709,28:7710,29:7745,30:7746,31:7755,32:7756,33:7757,

34:7790,35:8703,36:8707,37:8708,38:8771,39:8772,40:8776

}

no\_desc = [7,8,14,15,16,17,19,20,21,23,35]

stony = [6,12]

very\_stony = [2,9,18,26]

extremely\_stony = [1,22,24,25,27,28,29,30,31,32,33,34,36,37,38,39,40]

rubbly = [3,4,5,10,11,13]

surface\_cover = {i:0 for i in no\_desc}

surface\_cover.update({i:1 for i in stony})

surface\_cover.update({i:2 for i in very\_stony})

surface\_cover.update({i:3 for i in extremely\_stony})

surface\_cover.update({i:4 for i in rubbly})

soil\_features = [f'Soil\_Type{i}' for i in range(1,41)]

wilderness\_features = [x for x in data.columns if x.startswith("Wilderness\_Area")]

def r(x):

if x+180>360:

return x-180

else:

return x+180

def misc\_features(data):

df = data.copy()

df["soil\_type\_count"] = df[soil\_features].sum(axis=1)

df["wilderness\_area\_count"] = df[wilderness\_features].sum(axis=1)

df['Soil\_Type'] = 0

for i in range(1,41):

df['Soil\_Type'] += i\*df[f'Soil\_Type{i}']

df['Climatic\_Zone'] = df['Soil\_Type'].apply(

lambda x: int(str(ELU\_CODE[x])[0])

)

df['Geologic\_Zone'] = df['Soil\_Type'].apply(

lambda x: int(str(ELU\_CODE[x])[1])

)

df['Surface\_Cover'] = df['Soil\_Type'].apply(

lambda x: surface\_cover[x]

)

df['Soil\_12\_32'] = df['Soil\_Type32'] \* df['Soil\_Type12']

df['Soil\_Type23\_22\_32\_33'] = df['Soil\_Type23'] + df['Soil\_Type22'] + df['Soil\_Type32'] + df['Soil\_Type33']

df['Soil29\_Area1'] = df['Soil\_Type29'] + df['Wilderness\_Area1']

df['Soil3\_Area4'] = df['Wilderness\_Area4'] + df['Soil\_Type3']

df['Climate\_Area2'] = df['Wilderness\_Area2']\*df['Climatic\_Zone']

df['Climate\_Area4'] = df['Wilderness\_Area4']\*df['Climatic\_Zone']

df['Surface\_Area1'] = df['Wilderness\_Area1']\*df['Surface\_Cover']

df['Surface\_Area2'] = df['Wilderness\_Area2']\*df['Surface\_Cover']

df['Surface\_Area4'] = df['Wilderness\_Area4']\*df['Surface\_Cover']

for col, dtype in df.dtypes.iteritems():

if dtype.name.startswith('float'):

df[col] = df[col].astype('float64')

df['Horizontal\_Distance\_To\_Roadways\_Log'] = [math.log(v+1) for v in df['Horizontal\_Distance\_To\_Roadways']]

df['Water Elevation'] = df['Elevation'] - df['Vertical\_Distance\_To\_Hydrology']

df['Hydro\_Fire\_1'] = df['Horizontal\_Distance\_To\_Hydrology'] + df['Horizontal\_Distance\_To\_Fire\_Points']

df['Hydro\_Fire\_2'] = abs(df['Horizontal\_Distance\_To\_Hydrology'] - df['Horizontal\_Distance\_To\_Fire\_Points'])

df['Hydro\_Road\_1'] = abs(df['Horizontal\_Distance\_To\_Hydrology'] + df['Horizontal\_Distance\_To\_Roadways'])

df['Hydro\_Road\_2'] = abs(df['Horizontal\_Distance\_To\_Hydrology'] - df['Horizontal\_Distance\_To\_Roadways'])

df['Fire\_Road\_1'] = abs(df['Horizontal\_Distance\_To\_Fire\_Points'] + df['Horizontal\_Distance\_To\_Roadways'])

df['Fire\_Road\_2'] = abs(df['Horizontal\_Distance\_To\_Fire\_Points'] - df['Horizontal\_Distance\_To\_Roadways'])

df['EVDtH'] = df.Elevation - df.Vertical\_Distance\_To\_Hydrology

df['EHDtH'] = df.Elevation - df.Horizontal\_Distance\_To\_Hydrology \* 0.2

df['Elev\_3Horiz'] = df['Elevation'] + df['Horizontal\_Distance\_To\_Roadways'] + df['Horizontal\_Distance\_To\_Fire\_Points'] + df['Horizontal\_Distance\_To\_Hydrology']

df['Elev\_Road\_1'] = df['Elevation'] + df['Horizontal\_Distance\_To\_Roadways']

df['Elev\_Road\_2'] = df['Elevation'] - df['Horizontal\_Distance\_To\_Roadways']

df['Elev\_Fire\_1'] = df['Elevation'] + df['Horizontal\_Distance\_To\_Fire\_Points']

df['Elev\_Fire\_2'] = df['Elevation'] - df['Horizontal\_Distance\_To\_Fire\_Points']

df['EViElv'] = df['Vertical\_Distance\_To\_Hydrology'] \* df['Elevation']

df['Aspect2'] = df.Aspect.map(r)

df.fillna(0, inplace = True)

for col, dtype in df.dtypes.iteritems():

if dtype.name.startswith('int'):

df[col] = pd.to\_numeric(df[col], downcast ='integer')

elif dtype.name.startswith('float'):

df[col] = pd.to\_numeric(df[col], downcast ='float')

df.drop(columns = soil\_features, inplace = True)

df.drop(columns = ["Aspect"], inplace = True)

df.drop(columns = ["Horizontal\_Distance\_To\_Roadways"], inplace = True)

return df

df\_train = misc\_features(data)

df\_test = misc\_features(data\_test)

feature\_cols = df\_train.columns.to\_list()

feature\_cols.remove("Cover\_Type")

def two\_largest\_indices(inlist):

largest = 0

second\_largest = 0

largest\_index = 0

second\_largest\_index = -1

for i in range(len(inlist)):

item = inlist[i]

if item > largest:

second\_largest = largest

second\_largest\_index = largest\_index

largest = item

largest\_index = i

elif largest > item >= second\_largest:

second\_largest = item

second\_largest\_index = i

return largest\_index, second\_largest\_index

df\_train\_1\_2 = df\_train[(df\_train['Cover\_Type'] <= 2)]

df\_train\_3\_4\_6 = df\_train[(df\_train['Cover\_Type'].isin([3,4,6]))]

X\_train = df\_train[feature\_cols]

X\_test = df\_test[feature\_cols]

X\_train\_1\_2 = df\_train\_1\_2[feature\_cols]

X\_train\_3\_4\_6 = df\_train\_3\_4\_6[feature\_cols]

y = df\_train['Cover\_Type']

y\_1\_2 = df\_train\_1\_2['Cover\_Type']

y\_3\_4\_6 = df\_train\_3\_4\_6['Cover\_Type']

test\_ids = df\_test.index

clf = ExtraTreesClassifier(n\_estimators=500, random\_state=42, max\_depth=31, min\_samples\_split=2, criterion='entropy',

max\_features=12, n\_jobs=-1)

clf.fit(X\_train, y)

clf\_1\_2 = ExtraTreesClassifier(n\_estimators=500, random\_state=42, max\_depth=31, min\_samples\_split=2, criterion='gini',

max\_features=12, n\_jobs=-1)

clf\_1\_2.fit(X\_train\_1\_2, y\_1\_2)

clf\_3\_4\_6 = ExtraTreesClassifier(n\_estimators=500, random\_state=42, max\_depth=31, min\_samples\_split=2, criterion='gini',

max\_features=12, n\_jobs=-1)

clf\_3\_4\_6.fit(X\_train\_3\_4\_6, y\_3\_4\_6)

vals\_1\_2 = {}

for e, val in enumerate(list(clf\_1\_2.predict\_proba(X\_test))):

vals\_1\_2[e] = val

vals\_3\_4\_6 = {}

for e, val in enumerate(list(clf\_3\_4\_6.predict\_proba(X\_test))):

vals\_3\_4\_6[e] = val

vals = {}

for e, val in enumerate(list(clf.predict(X\_test))):

vals[e] = val

with open("submission.csv", "w") as outfile:

outfile.write("Id,Cover\_Type\n")

for e, val in enumerate(list(clf.predict\_proba(X\_test))):

val[0] += vals\_1\_2[e][0]/1.3

val[1] += vals\_1\_2[e][1]/1.1

val[2] += vals\_3\_4\_6[e][0]/3.4

val[3] += vals\_3\_4\_6[e][1]/4.0

val[5] += vals\_3\_4\_6[e][2]/3.6

i,j = two\_largest\_indices(val)

v = i + 1

outfile.write("%s,%s\n"%(test\_ids[e],v))

actual=pd.read\_csv(r'C:\Users\Ayushi\Desktop\train.csv', index\_col="Id")

predicted=pd.read\_csv(r'C:\Users\Ayushi\Desktop\python programs\submission.csv', index\_col="Id")

predication\_data\_accuracy\_check=accuracy\_score(actual,predicted)

final\_accuracy\_to\_be\_printed=predication\_data\_accuracy\_check

df=pd.read\_csv(r'C:\Users\Ayushi\Desktop\python programs\submission.csv')

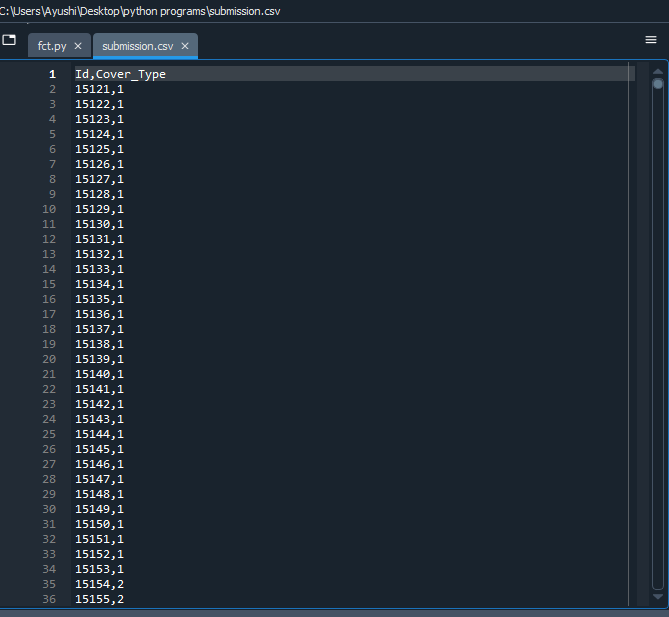
df['Cover\_Type'].hist()

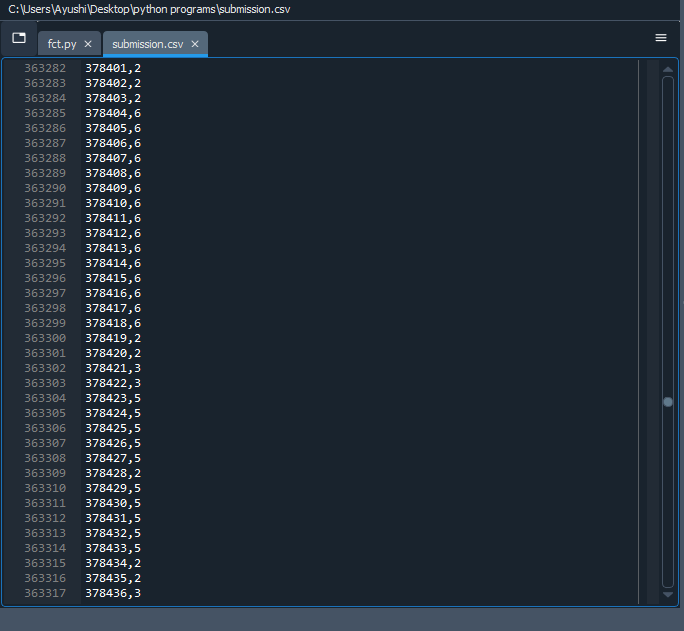
plt.xlabel("Cover Type")

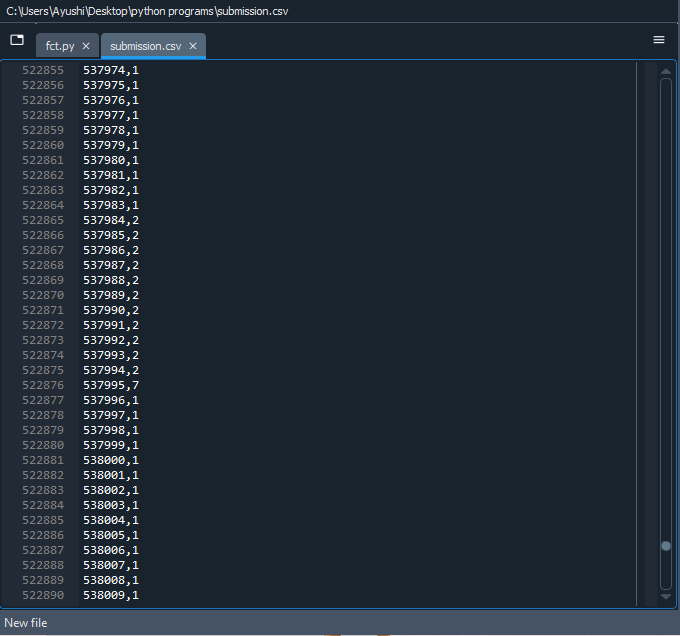
print ("Accuracy",final\_accuracy\_to\_be\_printed)

* 1. **Output Snapshots:**

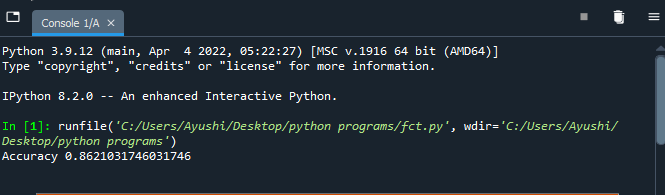
The output of the program is made available in the form of csv file also, an output of accuracy and histogram is shown as follows:

Snapshots of csv file:

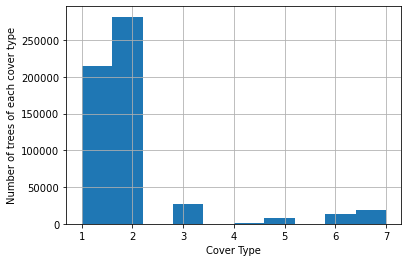




Accuracy:



* 1. **The histogram:**



1. **CONCLUSION:**

The goal of our algorithm is to be able to predict forest cover type based on catographical data.The best result was obtained using ExtraTreesClassifier ,which gave 86.2% accuracy for train dataset and submission dataset. In terms of the social importance, predictive project can be used to help farmers or government to plant more and more trees according to the suitable region so that it can grow well and help us in return.

1. **REFERENCES:**

[Predicting Forest Cover Types with the Machine Learning Workflow | by Ceren Iyim | Towards Data Science](https://towardsdatascience.com/predicting-forest-cover-types-with-the-machine-learning-workflow-1f6f049bf4df)

[Forest Cover Type Prediction (slideshare.net)](https://www.slideshare.net/akrish/forest-cover-type-prediction)

[(268) Dealing With Imbalanced Classes (Forest Cover Type Prediction) - Data Every Day #125 - YouTube](https://www.youtube.com/watch?v=y70bUYIPe2A)

[(268) Decision Tree in Python using Scikit-Learn | Tutorial | Machine Learning - YouTube](https://www.youtube.com/watch?v=uN1G1sZlELs)