greatlearning

FINAL PROJECT REPORT
FOREST COVER TYPE PREDICTION

Post Graduate Program in Data Science Engineering

Location: **Hyderabad** Batch: **PGPDSE-FT-H-July23**

Submitted by

Batchu Heam chandu
S Esha Yadav
Yash Vardhan Khetawat
Mamidi Reshma
V.Sriram
Vishnu Vamsi

Mentored by

Mr. Animesh Tiwari

Table Of Content

1. INDUSTRY REVIEW

1.1 Overview:

The project focuses on predicting forest cover types using a dataset with 55 features, including 11 quantitative variables, 4 binary variables for Wilderness Area, and 40 binary variables for Soil Type. The dataset encompasses diverse aspects such as elevation, slope, distance to hydrology, and more. The target variable is the forest cover type, categorized into seven classes. The objective is to develop a predictive model that accurately classifies the forest cover type based on the given features. The project involves exploratory data analysis, statistical parameterization, and the application of machine learning algorithms to achieve accurate and efficient predictions. The categorical features, including Wilderness Area and Soil Type, offer additional complexity to the prediction task. The project aims to contribute insights into forest cover dynamics, aiding in ecological studies and sustainable forest management.

1.2 Current Practices:

In the field of remote sensing for forest monitoring, current practices involve the utilization of various Earth observation satellites and technologies. Satellite programs like Landsat, Copernicus, and MODIS are widely employed to provide spatial and temporal observations of forest characteristics at landscape and regional scales. Instruments such as light detection and ranging (LiDAR) and hyperspectral sensors are frequently used to quantify forest characteristics at stand to landscape levels.

1.3 Background Research:

The background research in remote sensing for forest monitoring has evolved with innovations in technology and computing methods. Over the last few decades, there has been a continuous improvement in forest monitoring efforts, driven by the need for effective management of forest resources. The research includes the development and application of statistical and machine-learning models derived from plot-level field observations, which are extrapolated to larger areas using remote sensing data.

2. Dataset and Domain

Domain:

The goal of this project is to predict the forest cover type, specifically the predominant kind of tree cover, using cartographic variables. This data is obtained from the US Geological Survey (USGS) and the US Forest Service (USFS) which is in open domain and includes four wilderness areas located in Roosevelt National Forest of northern Colorado and provided by Machine Learning Laboratory of University of California Irvine.

Data Characteristics:

- **Raw Form:** The data is in its raw form and has not been scaled. This implies that feature scaling may be necessary during the preprocessing stage.
- **Qualitative Independent Variables:** The dataset contains binary columns representing qualitative independent variables, such as wilderness areas and soil types.

Study Area:

- **Location:** The study area is the Roosevelt National Forest in northern Colorado.
- **Wilderness Areas:** There are four wilderness areas in the study, representing forests with minimal human-caused disturbances. The existing forest cover types are more a result of ecological processes than forest management practices.

Problem Statement:

Develop an accurate predictive model for classifying seven different land cover types in the Roosevelt National Forest's four wilderness areas in northern Colorado. Each observation corresponds to a 30m x 30m patch. The goal is to enhance understanding of the region's ecological dynamics for sustainable forest management and environmental conservation.

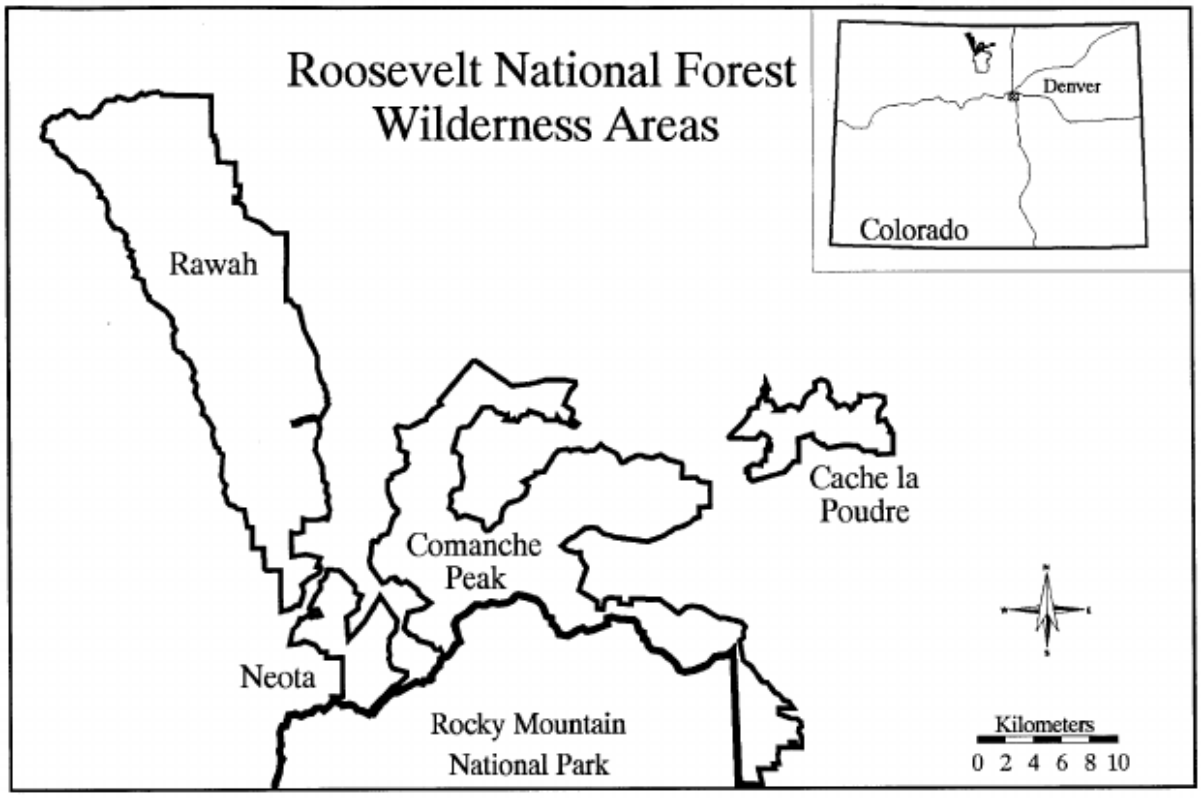
Data Exploration

Name	Measurement	Description
Elevation	meters	Elevation in meters
Aspect	azimuth	Aspect in degrees azimuth
Slope	degrees	Slope in degrees
Horizontal Distance To Hydrology	meters	Horz Dist to nearest surface water features
Vertical Distance To Hydrology	meters	Vert Dist to nearest surface water features
Horizontal Distance To Roadways	meters	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 soltice
Hillshade 3pm	0 to 255 index	Hillshade index at 3pm, summer solstice
Horizontal Distance To Fire Points	meters	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	Classes 1 to 7	Forest Cover Type designation - *Response Variable*

Name Data Type Measurement Description

- Elevation quantitative meters Elevation in meters
- Aspect quantitative azimuth Aspect in degrees azimuth
- Slope quantitative degrees Slope in degrees
- Horizontal_Distance_To_Hydrology quantitative meters Horz Dist to nearest surface water features
- Vertical_Distance_To_Hydrology quantitative meters Vert Dist to nearest surface water features
- Horizontal_Distance_To_Roadways quantitative meters Horz Dist to nearest roadway
- Hillshade_9am quantitative 0 to 255 index Hillshade index at 9am, summer solstice
- Hillshade_Noon quantitative 0 to 255 index Hillshade index at noon, summer soltice
- Hillshade_3pm quantitative 0 to 255 index Hillshade index at 3pm, summer solstice
- Horizontal_Distance_To_Fire_Points quantitative meters Horz Dist to nearest wildfire ignition points
- Wilderness_Area (4 binary columns) qualitative 0 (absence) or 1 (presence) Wilderness area designation
- Soil_Type (40 binary columns) qualitative 0 (absence) or 1 (presence) Soil Type designation
- Cover_Type (7 types) integer 1 to 7 Forest Cover Type designation

Details of Wilderness Areas:



Wilderness_Area1	Rawah Wilderness Area
Wilderness_Area2	Neota Wilderness Area
Wilderness_Area3	Comanche Wilderness Area
Wilderness_Area4	Cache La Poudre Wilderness Area

Background Information on the Four Wilderness Areas:

Neota (Area 2) likely has the highest mean elevation, primarily featuring spruce/fir. Rawah (Area 1) and Comanche Peak (Area 3) have a lower mean elevation, with lodgepole pine as the primary species. Cache la Poudre (Area 4) has the lowest mean elevation, characterized by Ponderosa pine, Douglas-fir, and cottonwood/willow. Rawah and Comanche Peak represent the overall dataset, while Neota and Cache la Poudre stand out due to unique features like elevation range and species composition.

Details of Forest Cover Type Classes:

1	Spruce / Fir
2	Lodgepole Pine
3	Ponderosa Pine
4	Cottonwood / Willow
5	Aspen
6	Douglas-fir
7	Krummholz

The study area has different types of forests, each with its own kinds of trees and plants. Some areas have spruce and fir trees, others have lodgepole pine, and some have a mix of spruce, fir, and aspen. There are also places with Ponderosa pine, Douglas-fir, and cottonwood/willow. The differences in elevation and the types of trees make each forest area unique and affect the plants and animals that live there.

Details of Soil Types:

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, 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

The study area features diverse soil types, from extremely stony rock outcrop complexes (Cathedral, Vanet) to rubbly ones (Haploborolis, Ratake). Complexes like Gothic and Supervisor-Limber contribute to the soil diversity, ranging from extremely stony (Leighcan, Granile-Catamount) to very stony (Rogert). The warm and extremely stony Leighcan family stands out. Various complexes like Cryaquolis and Cryumbrepts add to the intricate soil landscape. Understanding these soil variations is crucial for effective ecological and land management in the area.

3. DATA EXPLORATION

Preview of Dataset: Top 5 Rows

	Id	Elevation	Aspect	Slope	Horizontal_Distance_To_Hydrology	Vertical_Distance_To_Hydrology	Horizontal_Distance_T
0	1	2596	51	3	258	0	510
1	2	2590	56	2	212	-6	390
2	3	2804	139	9	268	65	3180
3	4	2785	155	18	242	118	3090
4	5	2595	45	2	153	-1	391

First columns and rows of the dataset

Soil_Type33	Soil_Type34	Soil_Type35	Soil_Type36	Soil_Type37	Soil_Type38	Soil_Type39	Soil_Type40	Cover_Type
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	3

Last columns and rows of the dataset

3.1. UNIVARIATE ANALYSIS: Numerical variables

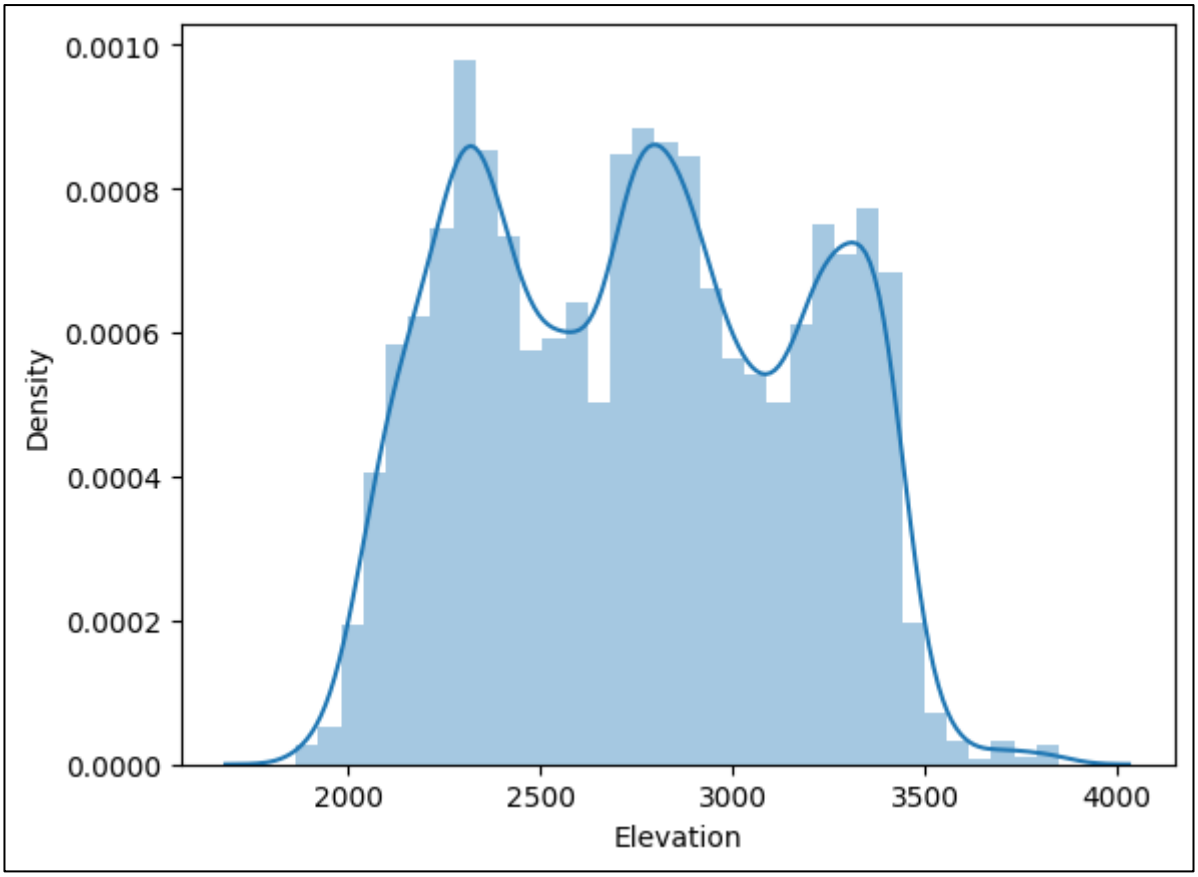
1. ELEVATION:

Quantile statistics

Minimum	1863
5-th percentile	2117
Q1	2376
median	2752
Q3	3104
95-th percentile	3397
Maximum	3849
Range	1986
Interquartile range (IQR)	728

Descriptive statistics

Standard deviation	417.67819
Coefficient of variation (CV)	0.1519204
Kurtosis	-1.0821158
Mean	2749.3226
Median Absolute Deviation (MAD)	367
Skewness	0.075639707
Sum	41569757
Variance	174455.07
Monotonicity	Not monotonic



Here we say that from descriptive statistics, the variable 'Elevation' column has normal distribution with minimal skewness is +0.08 and kurtosis is -1.08. Therefore, we can say that it is normally distributed and platykurtic in nature.

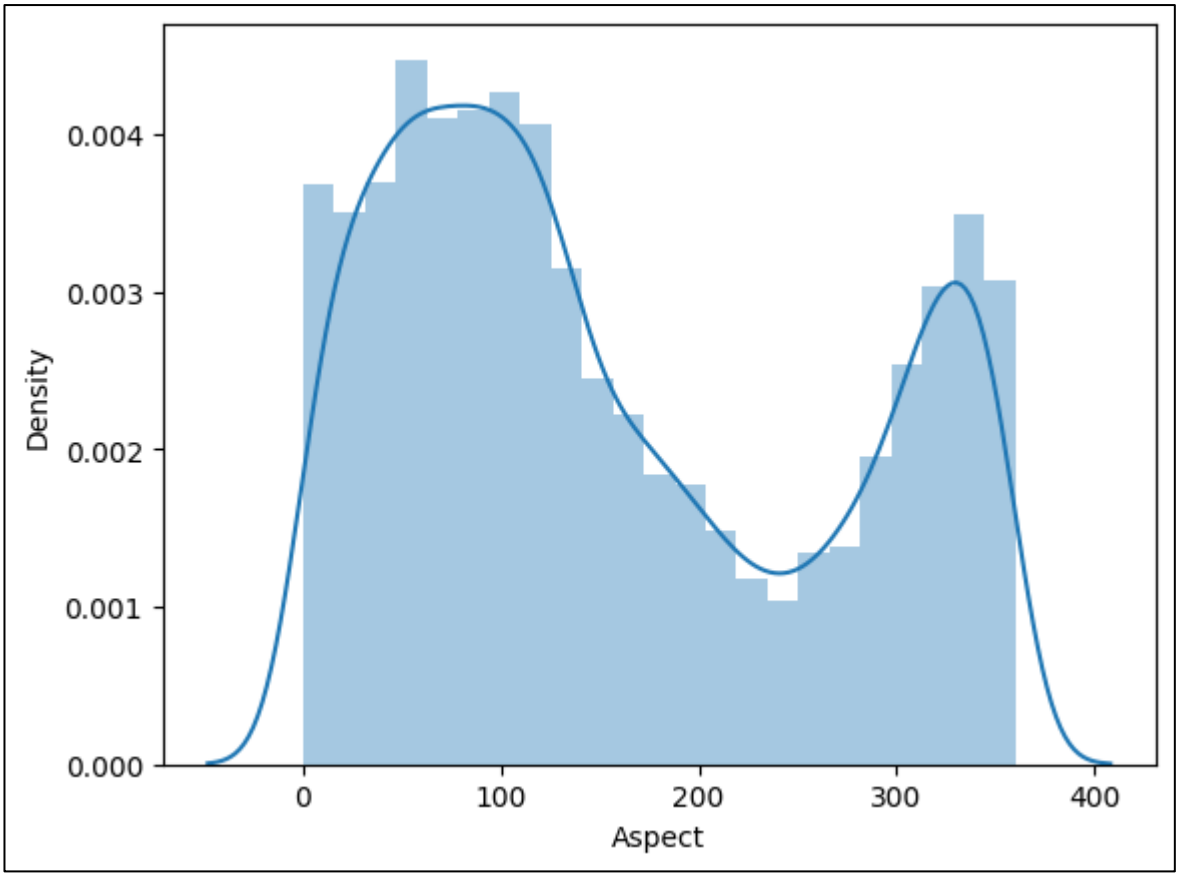
2. ASPECT:

Quantile statistics

Minimum	0
5-th percentile	13
Q1	65
median	126
Q3	261
95-th percentile	344
Maximum	360
Range	360
Interquartile range (IQR)	196

Descriptive statistics

Standard deviation	110.0858
Coefficient of variation (CV)	0.70263054
Kurtosis	-1.1502445
Mean	156.67665
Median Absolute Deviation (MAD)	77
Skewness	0.45093529
Sum	2368951
Variance	12118.884
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Slope' has a skew value of +0.45 and kurtosis value of -1.15. Therefore, we can say that it is normally distributed and platykurtic in nature.

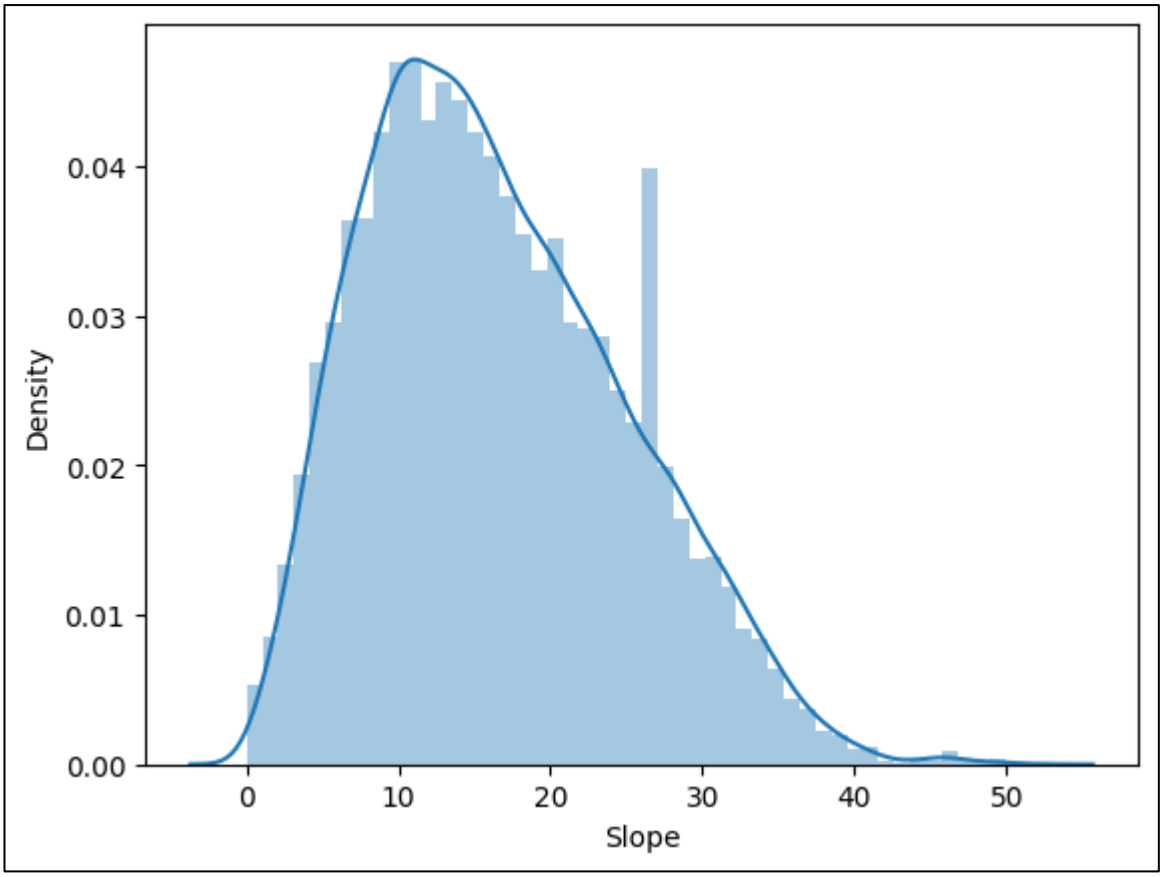
3. SLOPE:

Quantile statistics

Minimum	0
5-th percentile	5
Q1	10
median	15
Q3	22
95-th percentile	32
Maximum	52
Range	52
Interquartile range (IQR)	12

Descriptive statistics

Standard deviation	8.4539268
Coefficient of variation (CV)	0.51230991
Kurtosis	-0.23831014
Mean	16.501587
Median Absolute Deviation (MAD)	6
Skewness	0.52365834
Sum	249504
Variance	71.468878
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Slope' has a skew value of +0.52 and kurtosis value of -0.238. Therefore, we can say that it is normally distributed and platykurtic in nature.

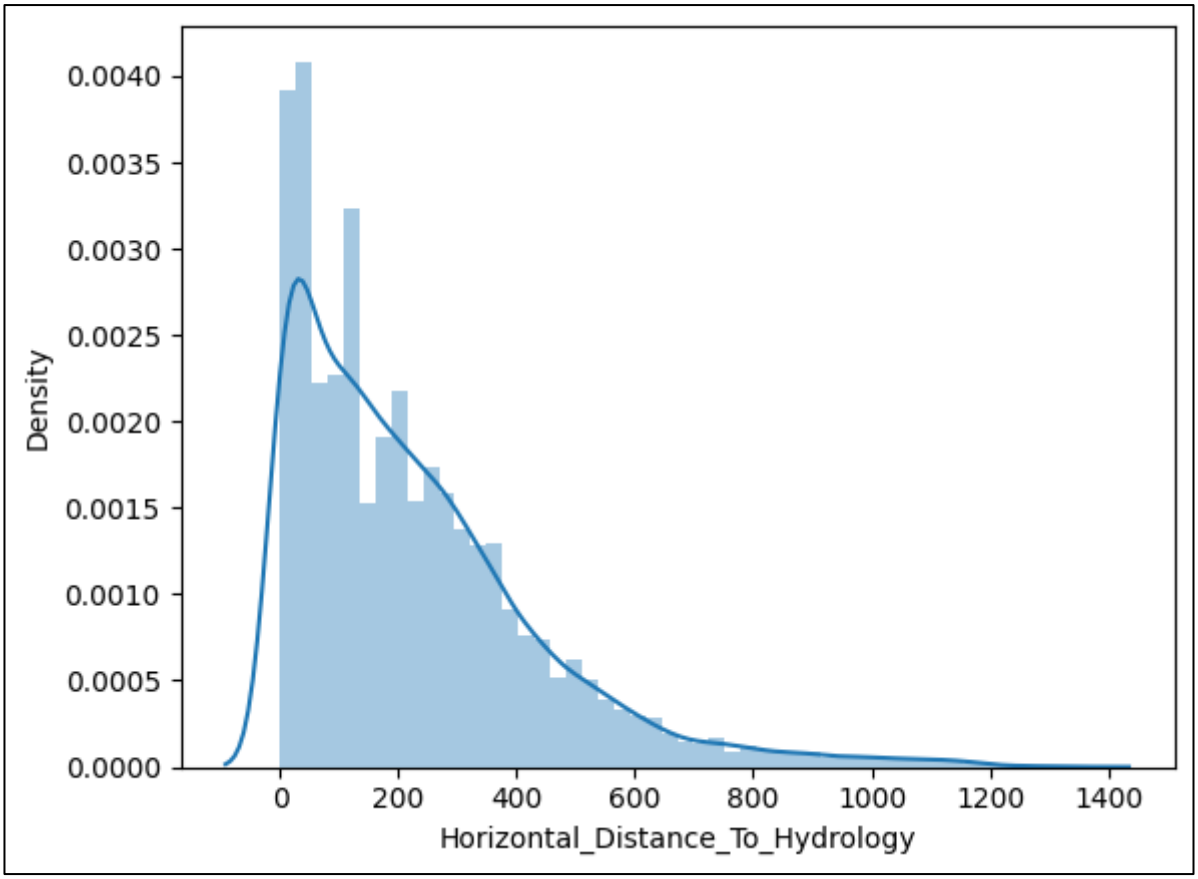
4. HORIZONTAL DISTANCE TO HYDROLOGY:

Quantile statistics

Minimum	0
5-th percentile	0
Q1	67
median	180
Q3	330
95-th percentile	631
Maximum	1343
Range	1343
Interquartile range (IQR)	263

Descriptive statistics

Standard deviation	210.0753
Coefficient of variation (CV)	0.92464468
Kurtosis	2.8039844
Mean	227.1957
Median Absolute Deviation (MAD)	120
Skewness	1.4880525
Sum	3435199
Variance	44131.63
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Slope' has a skew value of +1.488 and kurtosis value of +2.803. Therefore, we can say that it is positively skewed and leptokurtic in nature.

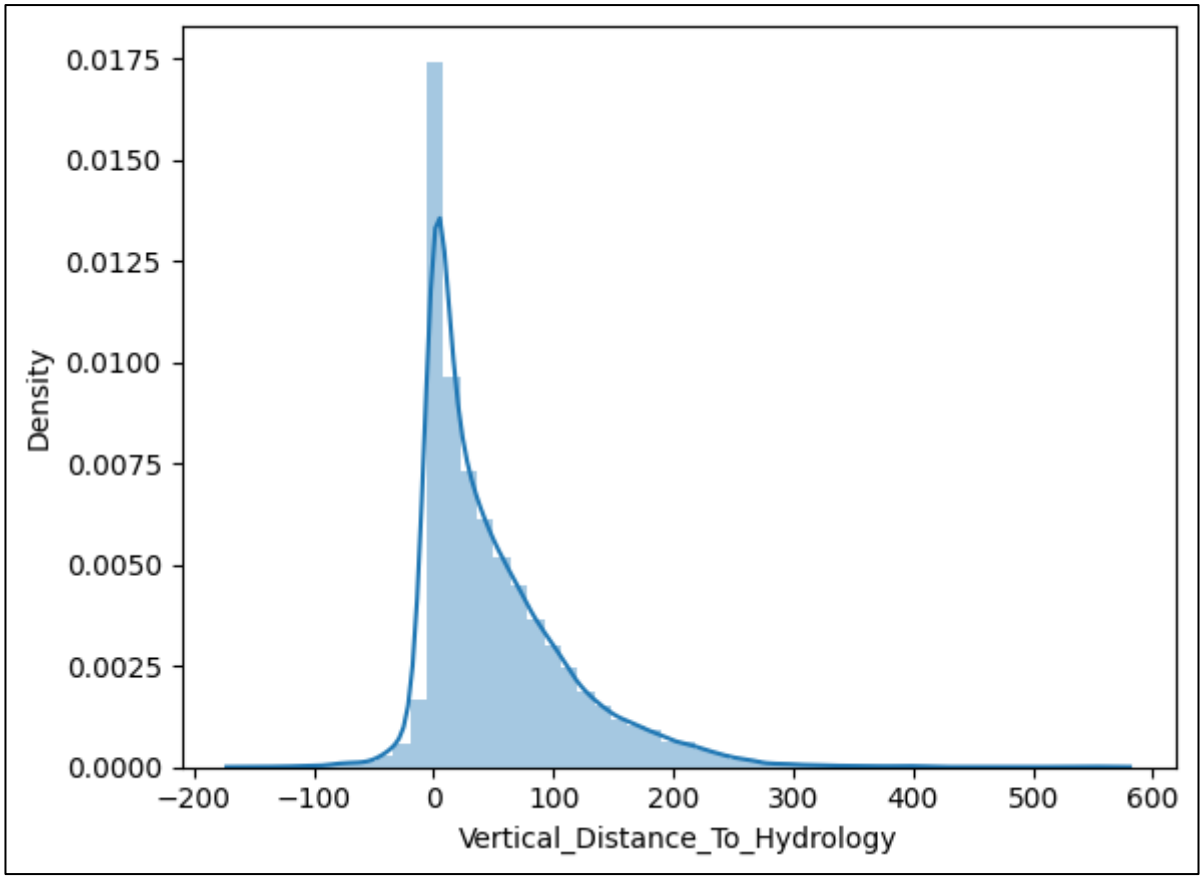
5. VERTICAL DISTANCE TO HYDROLOGY:

Quantile statistics

Minimum	-146
5-th percentile	-4
Q1	5
median	32
Q3	79
95-th percentile	176
Maximum	554
Range	700
Interquartile range (IQR)	74

Descriptive statistics

Standard deviation	61.239406
Coefficient of variation (CV)	1.1989737
Kurtosis	3.4034987
Mean	51.076521
Median Absolute Deviation (MAD)	32
Skewness	1.5377757
Sum	772277
Variance	3750.2649
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Vertical_Distance_To_Hydrology' has a skew value of +1.53 and kurtosis value of +3.40. Therefore, we can say that it is positively skewed and leptokurtic in nature.

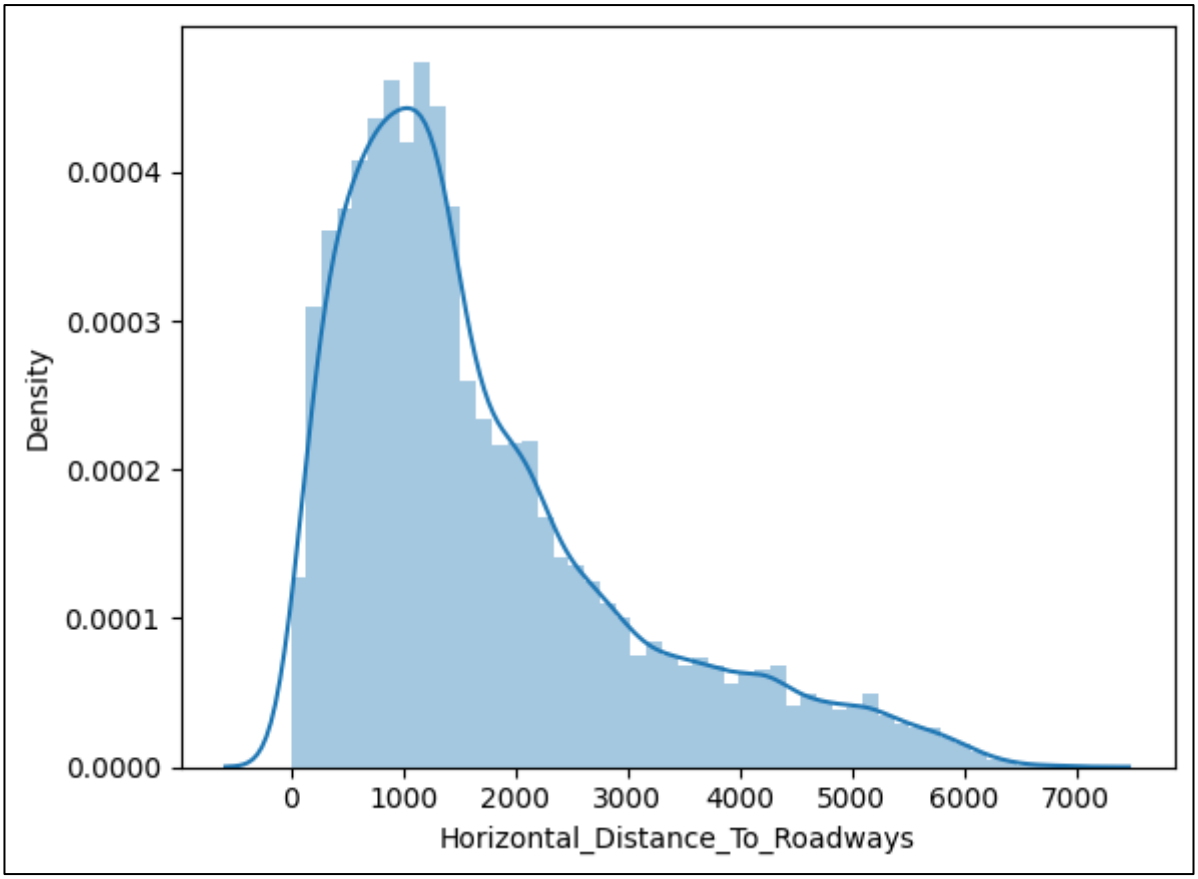
6. HORIZONTAL DISTANCE TO ROADWAYS:

Quantile statistics

Minimum	0
5-th percentile	242
Q1	764
median	1316
Q3	2270
95-th percentile	4635.1
Maximum	6890
Range	6890
Interquartile range (IQR)	1506

Descriptive statistics

Standard deviation	1325.0664
Coefficient of variation (CV)	0.77307375
Kurtosis	1.0224194
Mean	1714.0232
Median Absolute Deviation (MAD)	690
Skewness	1.2478107
Sum	25916031
Variance	1755800.9
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Horizontal_Distance_To_Roadways' has a skew value of +1.24 and kurtosis value of +1.022. Therefore, we can say that it is positively skewed and leptokurtic in nature.

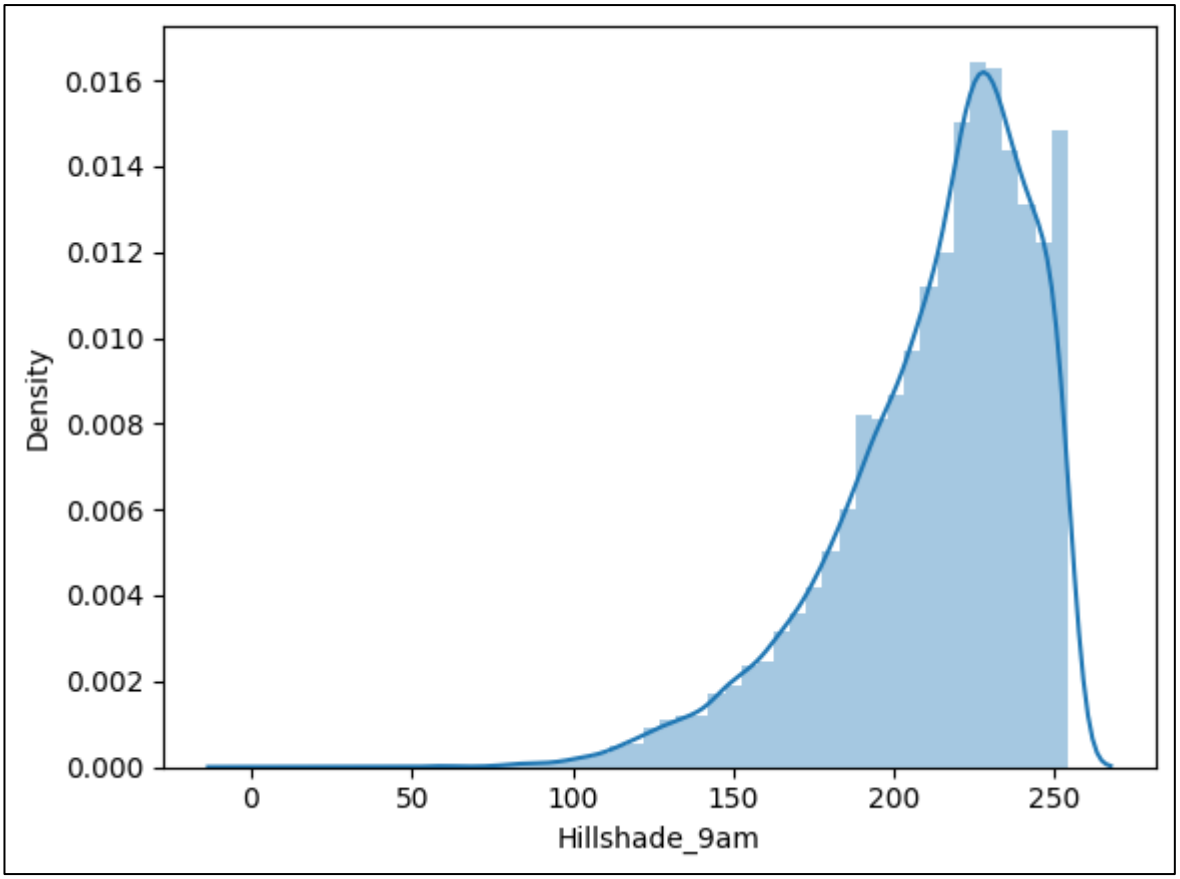
7. HILLSHADE_9AM:

Quantile statistics

Minimum	0
5-th percentile	151
Q1	196
median	220
Q3	235
95-th percentile	250
Maximum	254
Range	254
Interquartile range (IQR)	39

Descriptive statistics

Standard deviation	30.561287
Coefficient of variation (CV)	0.14367969
Kurtosis	1.2188105
Mean	212.7043
Median Absolute Deviation (MAD)	18
Skewness	-1.0936806
Sum	3216089
Variance	933.99226
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Hillshade_9am', has a skew value of -1.093 and kurtosis value of +1.21. Therefore, we can say that it is negatively skewed and leptokurtic in nature.

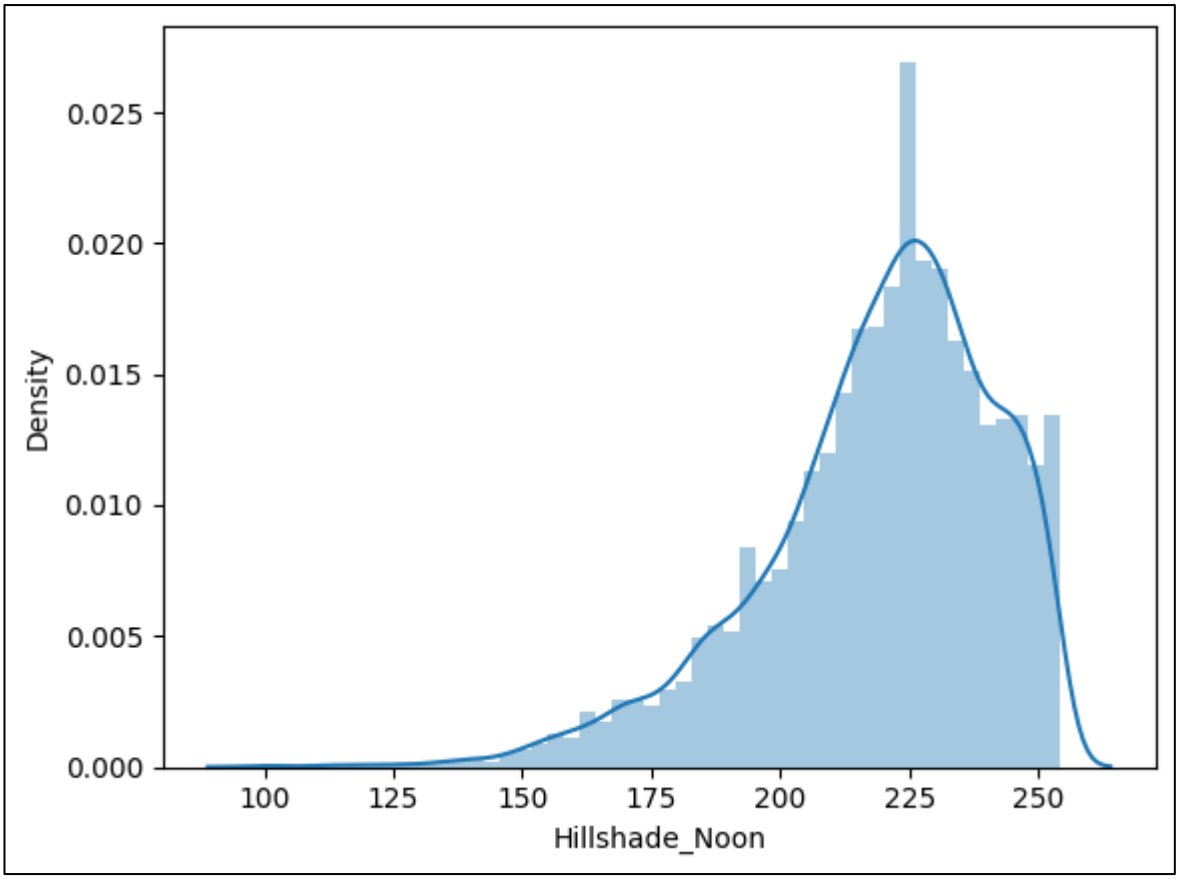
8. HILLSHADE NOON:

Quantile statistics

Minimum	99
5-th percentile	175
Q1	207
median	223
Q3	235
95-th percentile	250
Maximum	254
Range	155
Interquartile range (IQR)	28

Descriptive statistics

Standard deviation	22.801966
Coefficient of variation (CV)	0.10413492
Kurtosis	1.1534842
Mean	218.96561
Median Absolute Deviation (MAD)	14
Skewness	-0.95323171
Sum	3310760
Variance	519.92963
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Hillshade_Noon',has a skew value of -1.095 and kurtosis value of +1.15. Therefore, we can say that it is negatively skewed and leptokurtic in nature.

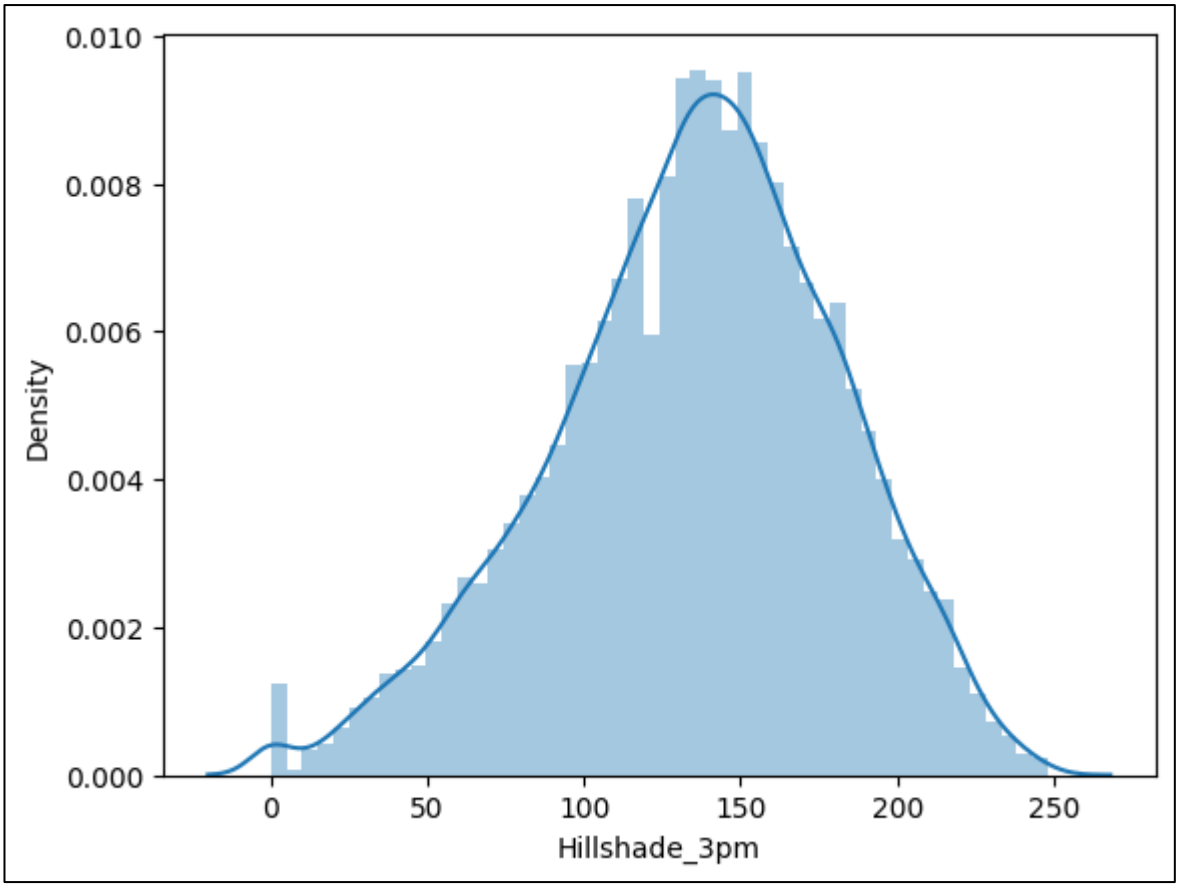
9. HILLSHADE 3PM:

Quantile statistics

Minimum	0
5-th percentile	53
Q1	106
median	138
Q3	167
95-th percentile	207
Maximum	248
Range	248
Interquartile range (IQR)	61

Descriptive statistics

Standard deviation	45.895189
Coefficient of variation (CV)	0.33973285
Kurtosis	-0.087343908
Mean	135.092
Median Absolute Deviation (MAD)	30
Skewness	-0.34082723
Sum	2042591
Variance	2106.3683
Monotonicity	Not monotonic



As we can see from descriptive statistics, the variable 'Hillshade_3pm', 'Hillshade_Noon',has a skew value of -0.34 and kurtosis value of -0.08. Therefore, we can say that it is negatively skewed and mesokurtic in nature.

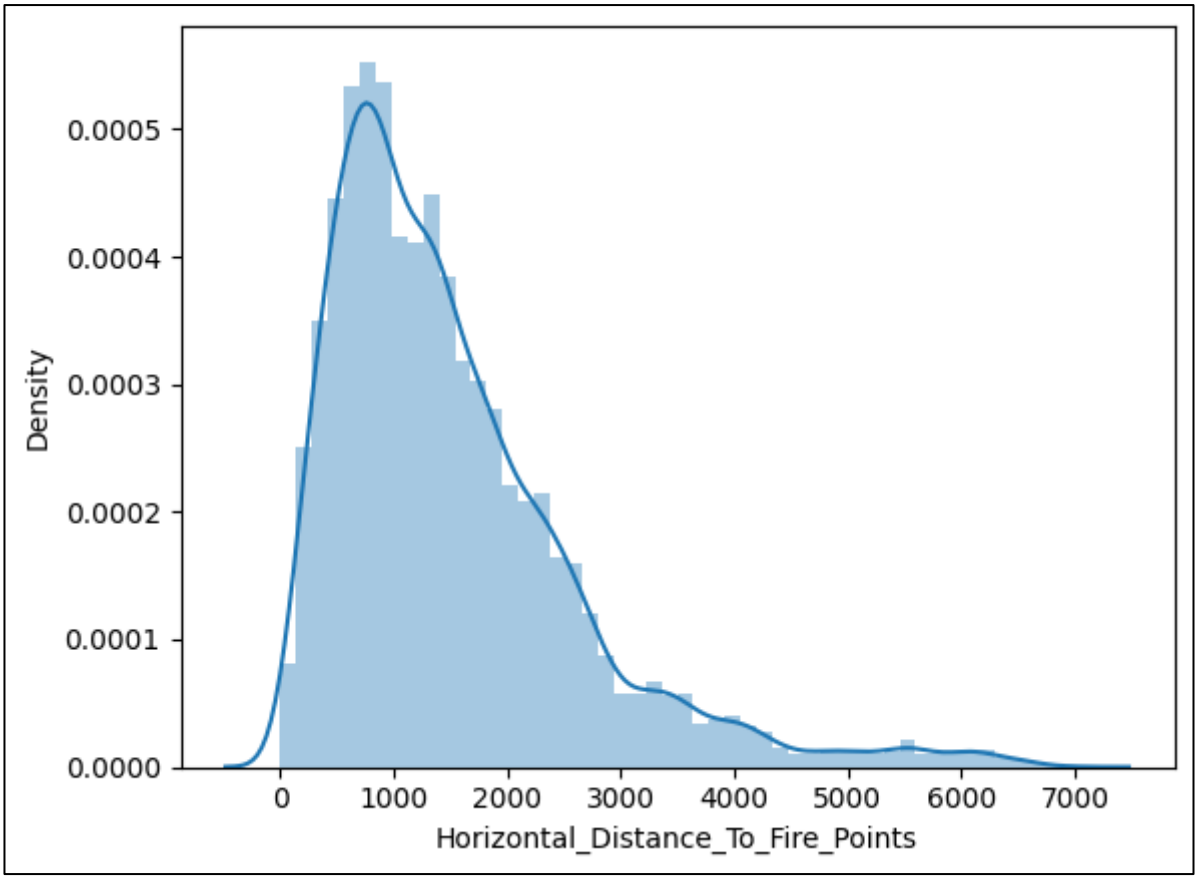
10. HORIZONTAL DISTANCE TO FIRE POINTS:

Quantile statistics

Minimum	0
5-th percentile	296.9
Q1	730
median	1256
Q3	1988.25
95-th percentile	3663.05
Maximum	6993
Range	6993
Interquartile range (IQR)	1258.25

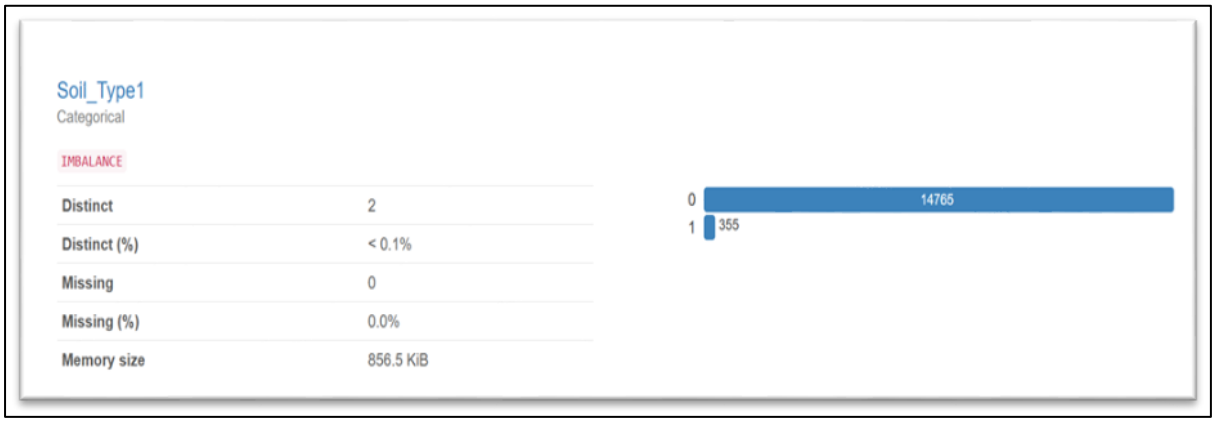
Descriptive statistics

Standard deviation	1099.9365
Coefficient of variation (CV)	0.72788172
Kurtosis	3.3854158
Mean	1511.1473
Median Absolute Deviation (MAD)	595
Skewness	1.6170989
Sum	22848547
Variance	1209860.3
Monotonicity	Not monotonic



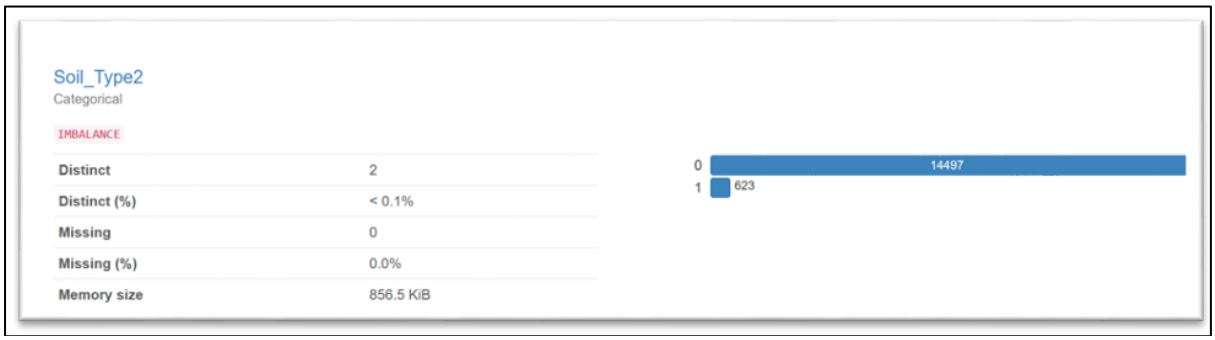
As we can see from descriptive statistics, the variable 'Hillshade_Noon',has a skew value of +1.617 and kurtosis value of +3.38. Therefore, we can say that it is negatively skewed and leptokurtic in nature.

11. Soil Type1:



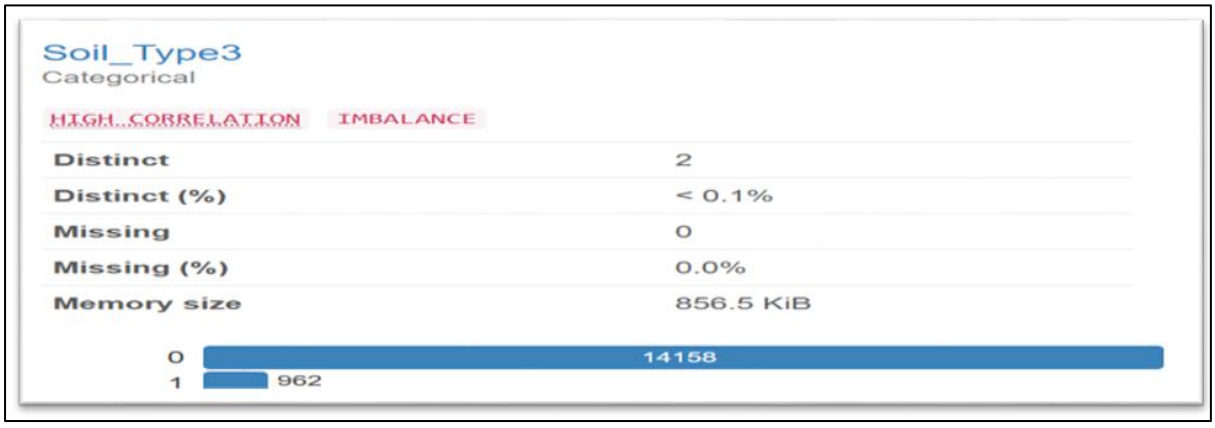
Here the variable 'Soil_Type1' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 97.7% of data is 0 along with 2.3% as 1 which means 2.3% of data contains from 'Soil_Type1' and remaining from rest of the soils.

12. Soil Type 2:



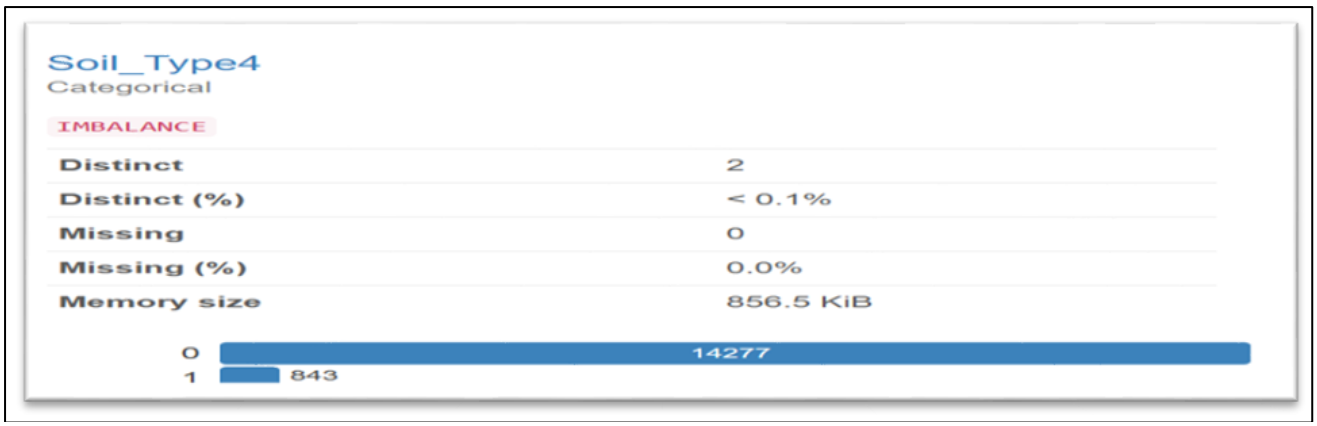
Here the variable 'Soil_Type2' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.9% of data is 0 along with 4.1% as 1 which means 4.1% of data contains from 'Soil_Type2' and remaining from rest of the soils.

13. Soil Type 3:



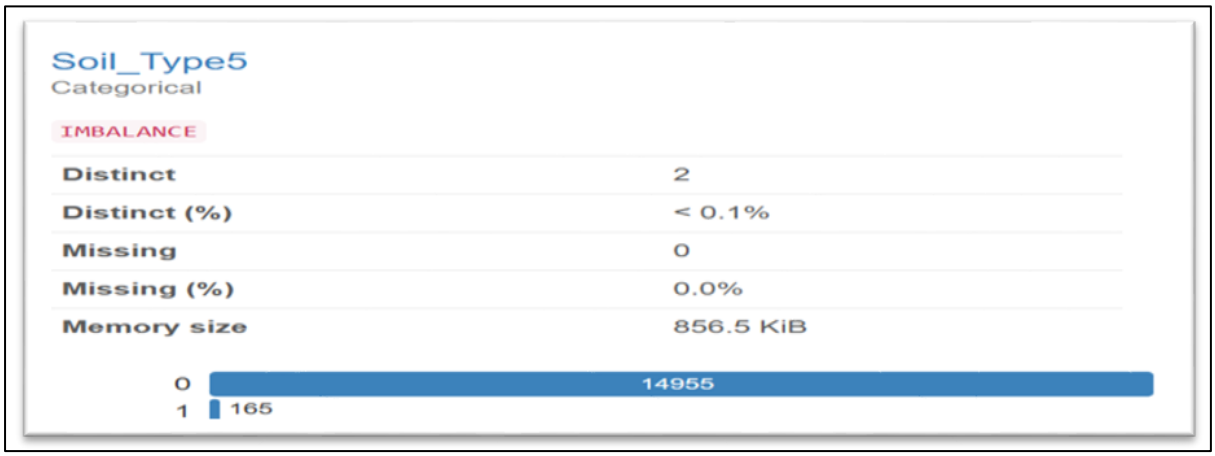
Here the variable 'Soil_Type3' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 93.6% of data is 0 along with 6.4% as 1 which means 6.4% of data contains from 'Soil_Type3' and remaining from rest of the soil.

14. Soil Type 4:



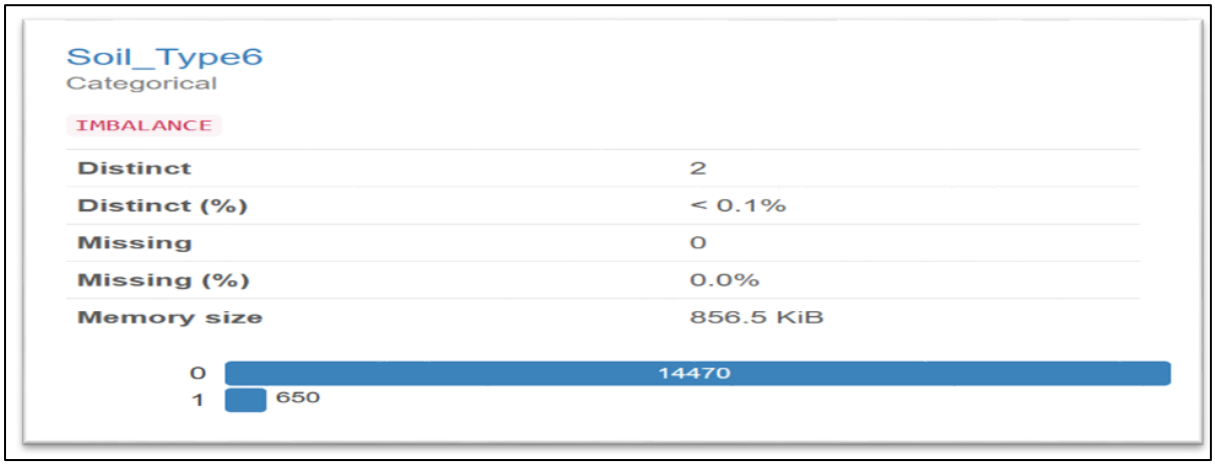
Here the variable 'Soil_Type4' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 94.4% of data is 0 along with 5.6% as 1 which means 5.6% of data contains from 'Soil_Type4' and remaining from rest of the soils

15. Soil Type 5:



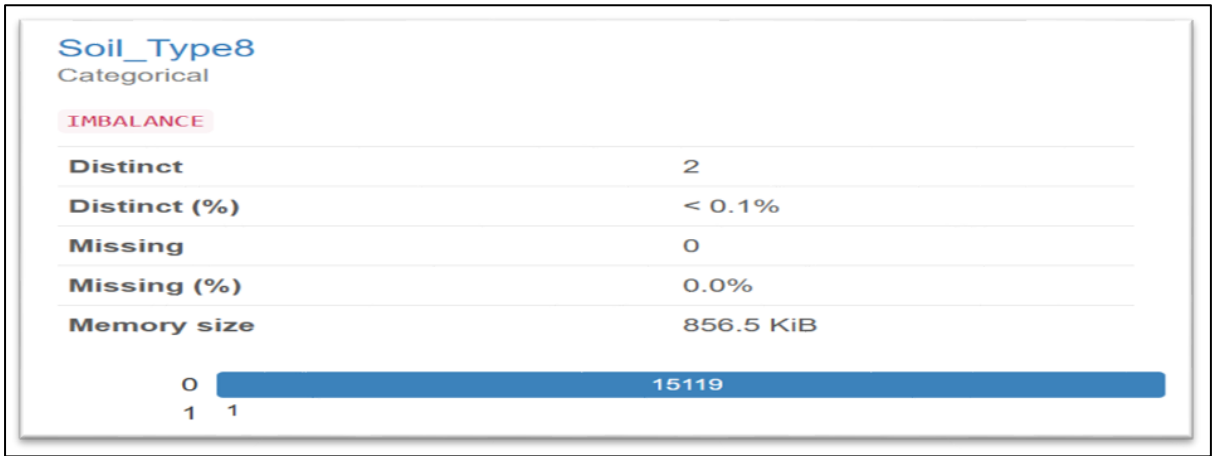
Here the variable 'Soil_Type5' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 98.9% of data is 0 along with 1.1% as 1 which means 1.1% of data contains from 'Soil_Type5' and remaining from rest of the soils.

16. Soil Type 6:



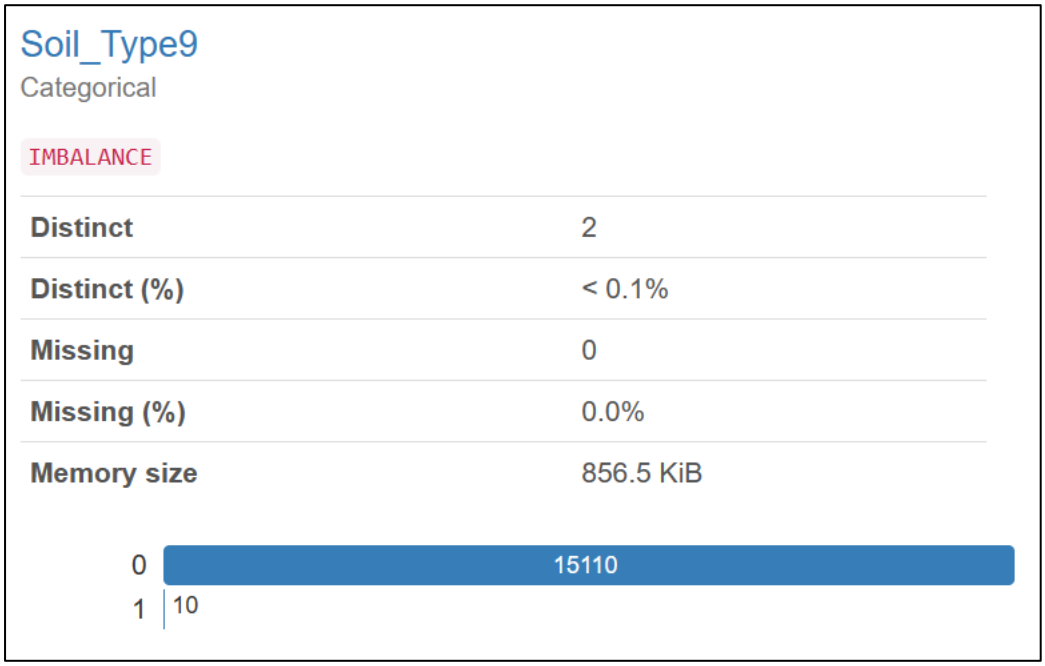
Here the variable 'Soil_Type6' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.7% of data is 0 along with 4.3% as 1 which means 4.3% of data contains from 'Soil_Type6' and remaining from rest of the soils.

17. Soil Type 8:



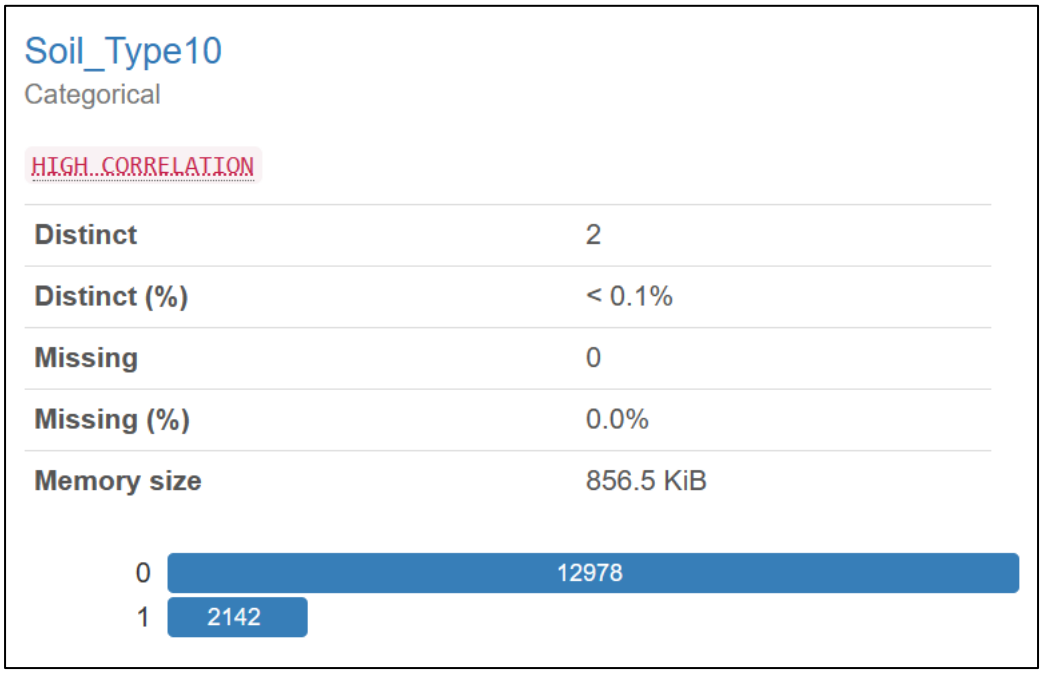
Here the variable 'Soil_Type8' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.9% of data is 0 along with 0.1% as 1 which means 0.1% of data contains from 'Soil_Type8' and remaining from rest of the soils.

18. Soil Type 9:



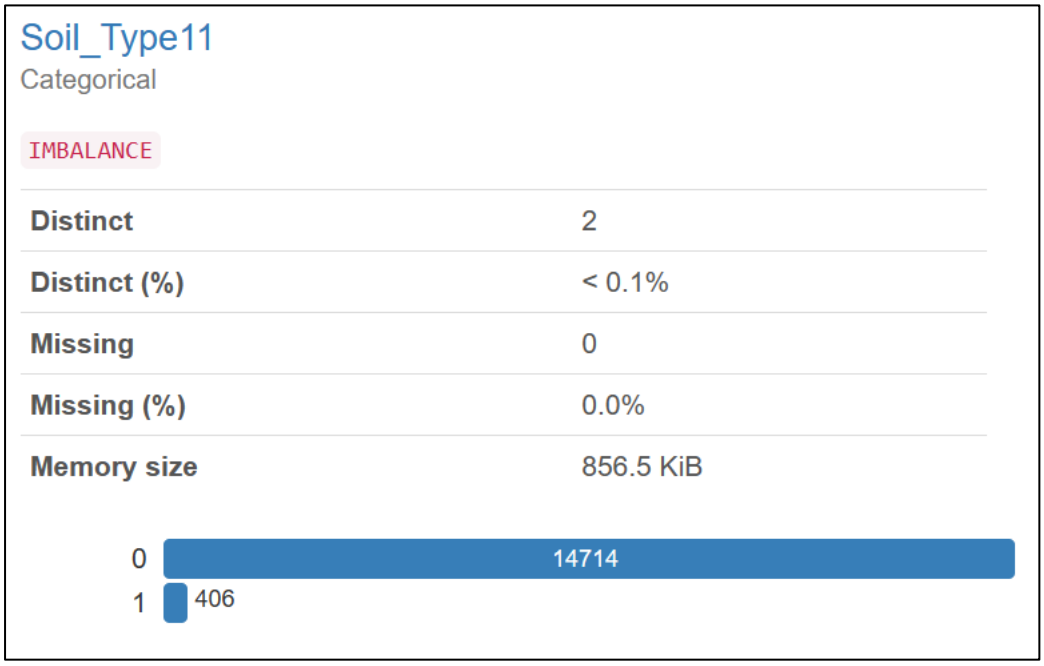
Here the variable 'Soil_Type9' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.9% of data is 0 along with 0.1% as 1 which means 0.1% of data contains from 'Soil_Type9' and remaining from rest of the soils.

19. Soil Type10:



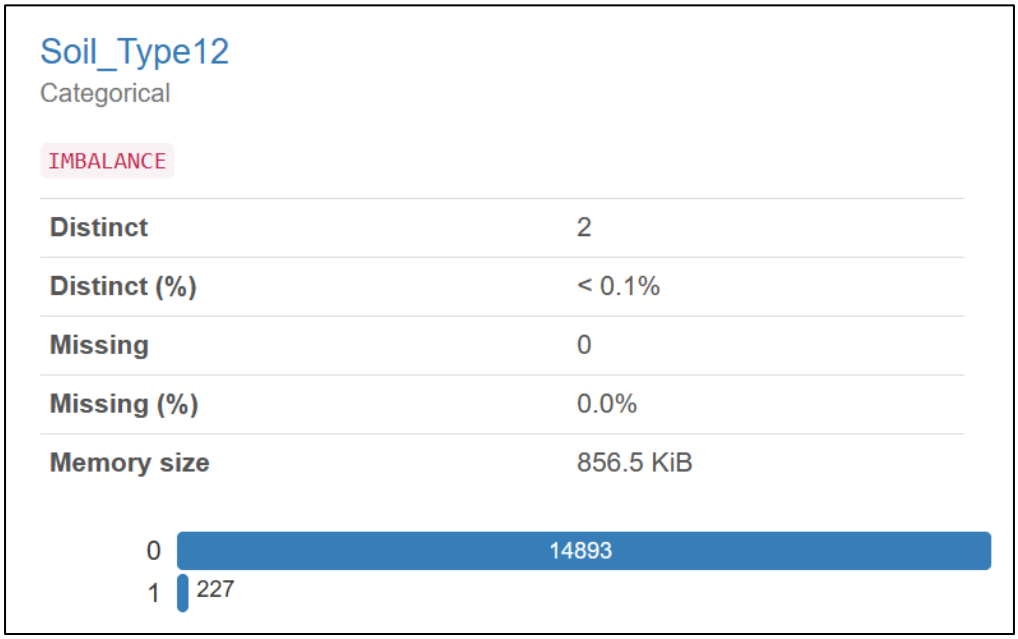
Here the variable 'Soil_Type10' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 85.8% of data is 0 along with 14.2% as 1 which means 14.2% of data contains from 'Soil_Type10' and remaining from rest of the soils.

20. Soil Type 11:



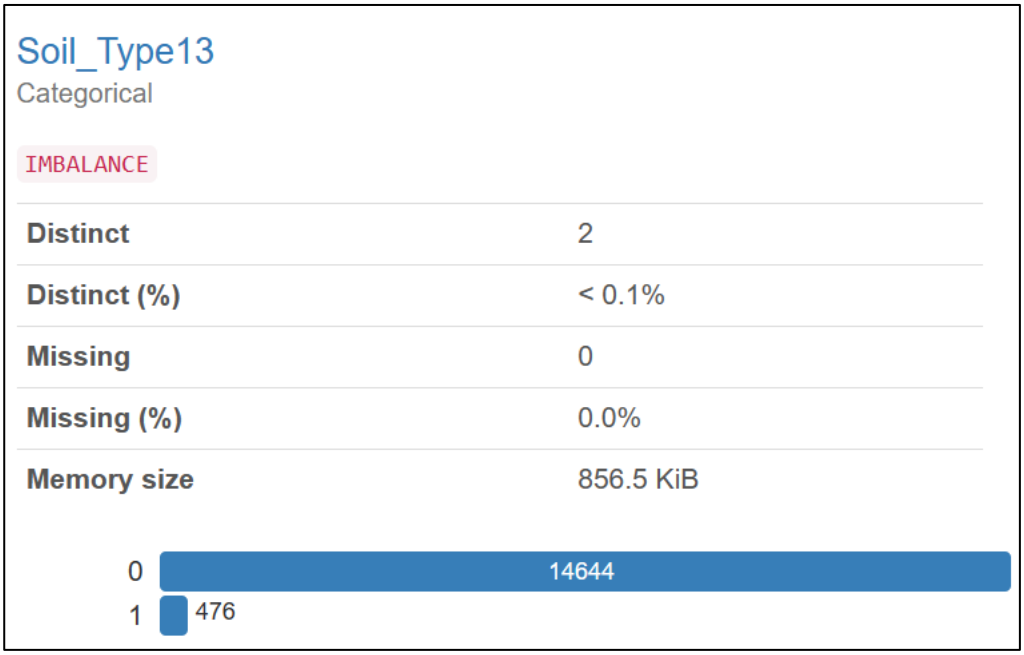
Here the variable 'Soil_Type11' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 97.3% of data is 0 along with 2.7% as 1 which means 2.7% of data contains from 'Soil_Type11' and remaining from rest of the soils.

21. Soil Type12:



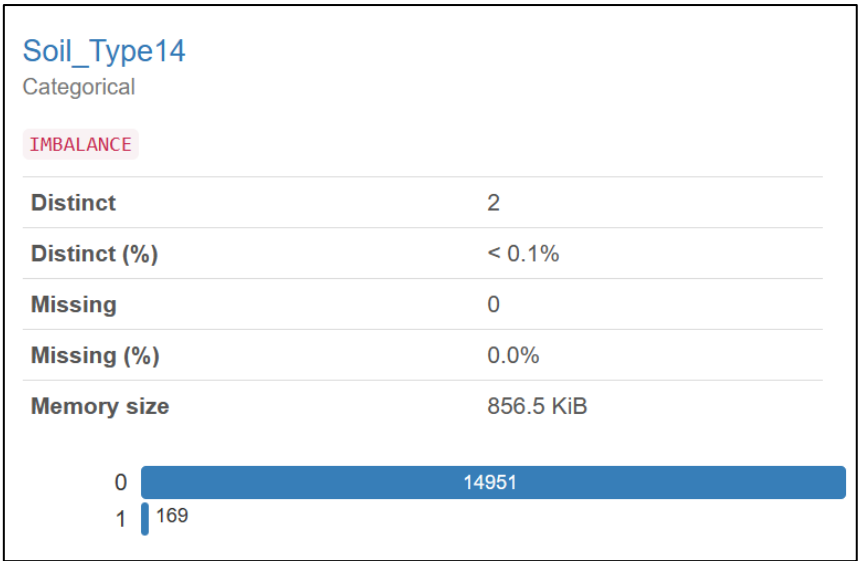
Here the variable ‘Soil_Type12’ has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 98.5% of data is 0 along with 1.5% as 1 which means 1.5% of data contains from ‘Soil_Type12’ and remaining from rest of the soils.

22. Soil Type13:



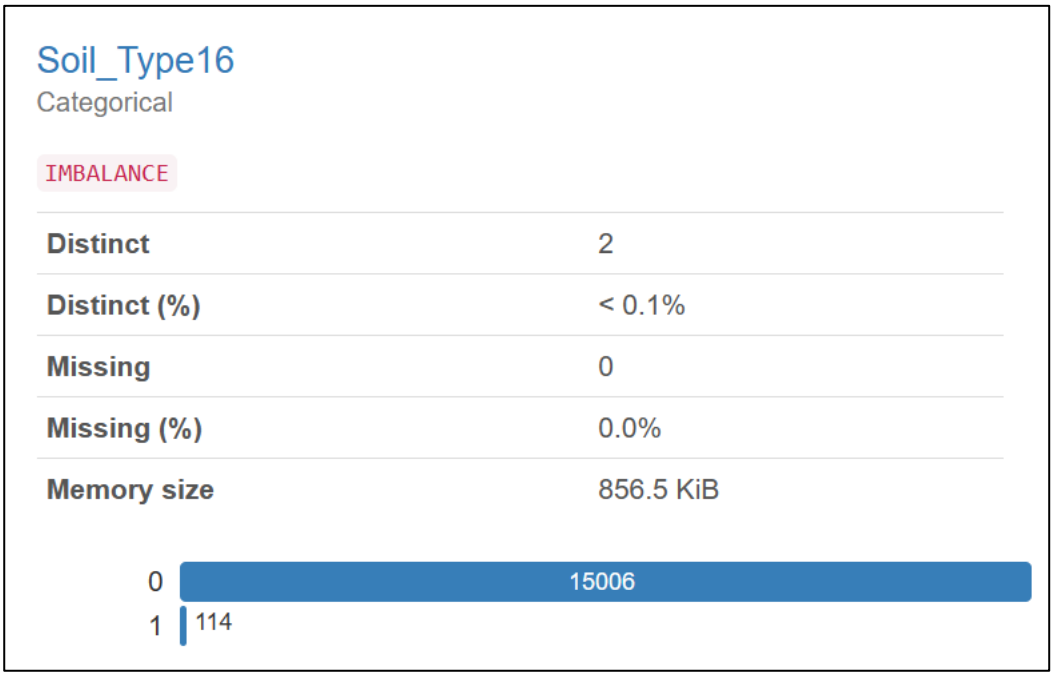
Here the variable ‘Soil_Type13’ has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 96.9% of data is 0 along with 3.1% as 1 which means 3.1% of data contains from ‘Soil_Type13’ and remaining from rest of the soils.

23. Soil Type14:



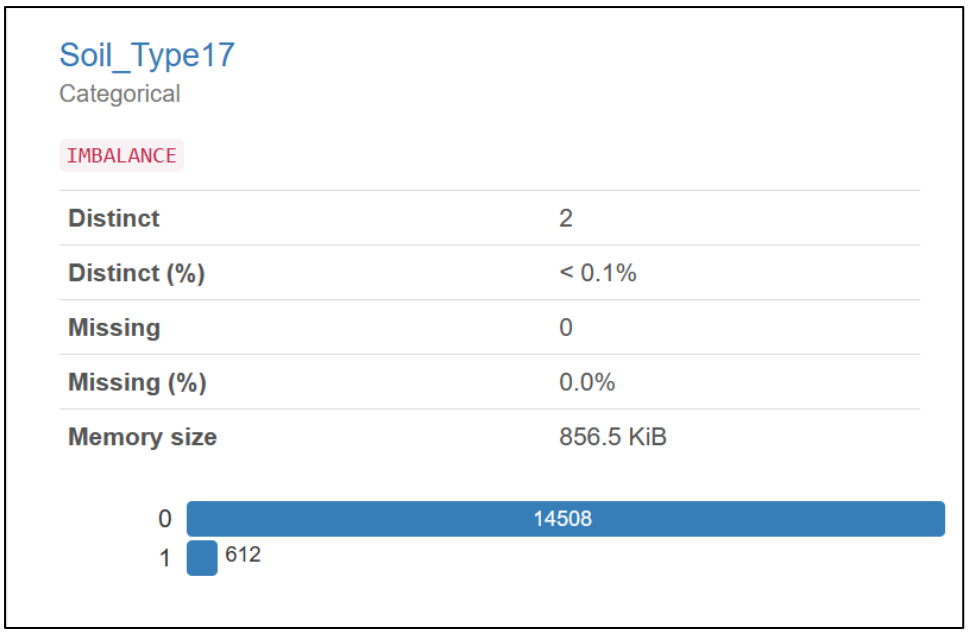
Here the variable ‘Soil_Type14’ has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 98.9% of data is 0 along with 1.1% as 1 which means 1.1% of data contains from ‘Soil_Type14’ and remaining from rest of the soils.

24. Soil Type16:



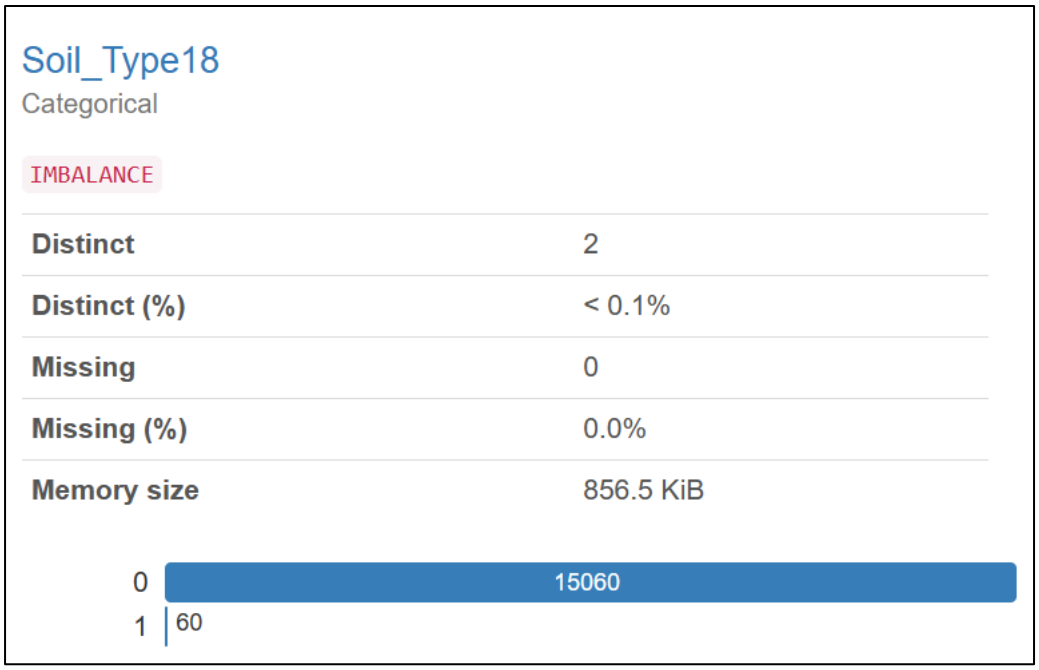
Here the variable ‘Soil_Type16’ has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.2% of data is 0 along with 0.8% as 1 which means 0.8% of data contains from ‘Soil_Type16’ and remaining from rest of the soils.

25. Soil Type17:



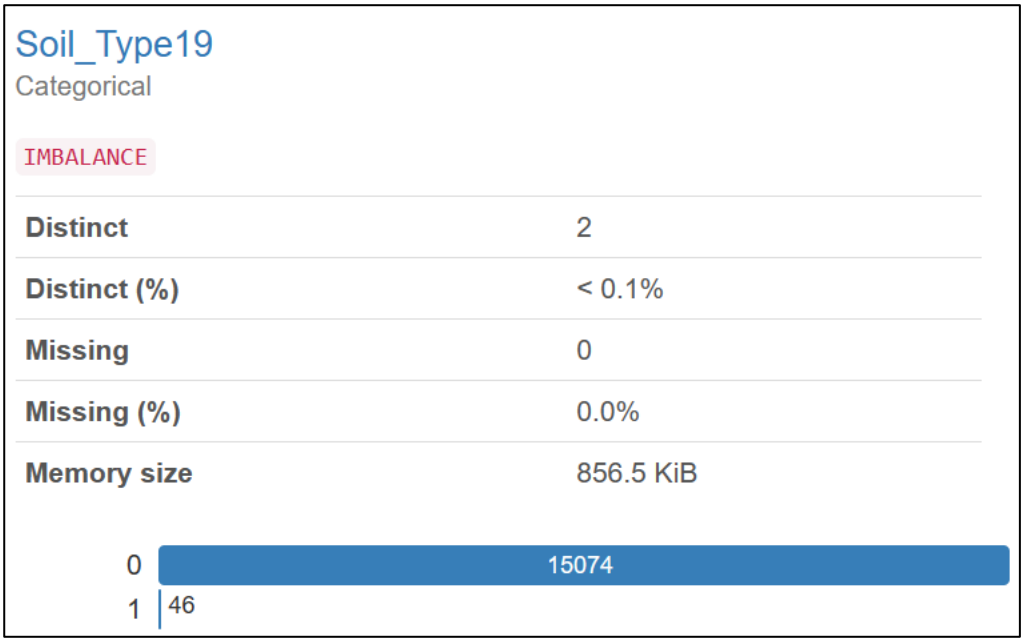
Here the variable 'Soil_Type17' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 96.0% of data is 0 along with 4% as 1 which means 4% of data contains from 'Soil_Type17' and remaining from rest of the soils.

26. Soil Type18:



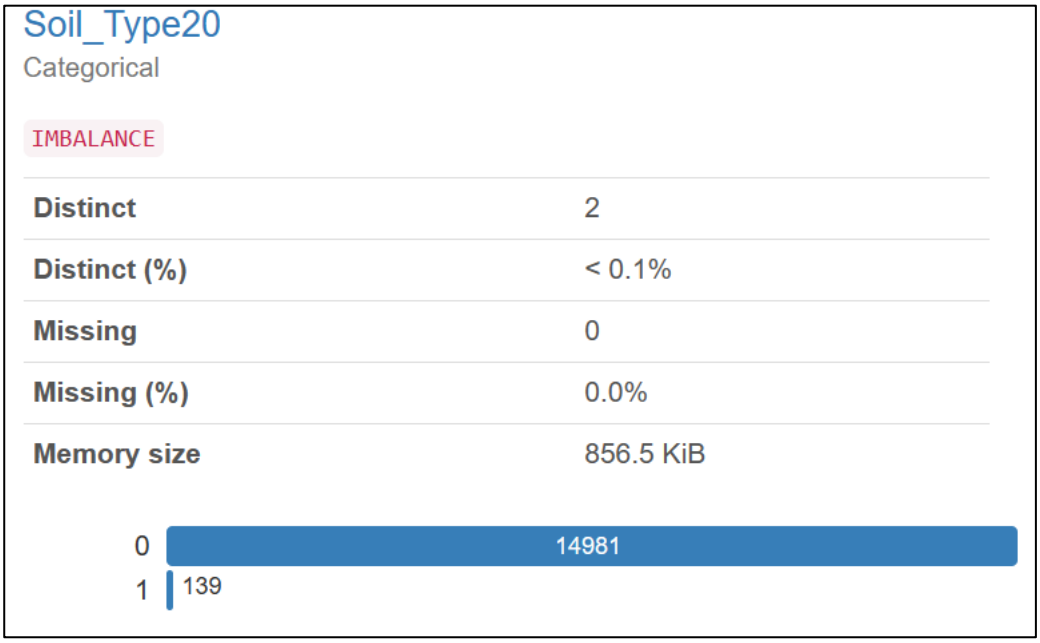
Here the variable 'Soil_type18' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.6% of data is 0 along with 0.4% as 1 which means 0.4% of data contains from 'Soil_Type18' and remaining from rest of the soils.

27. Soil Type19:



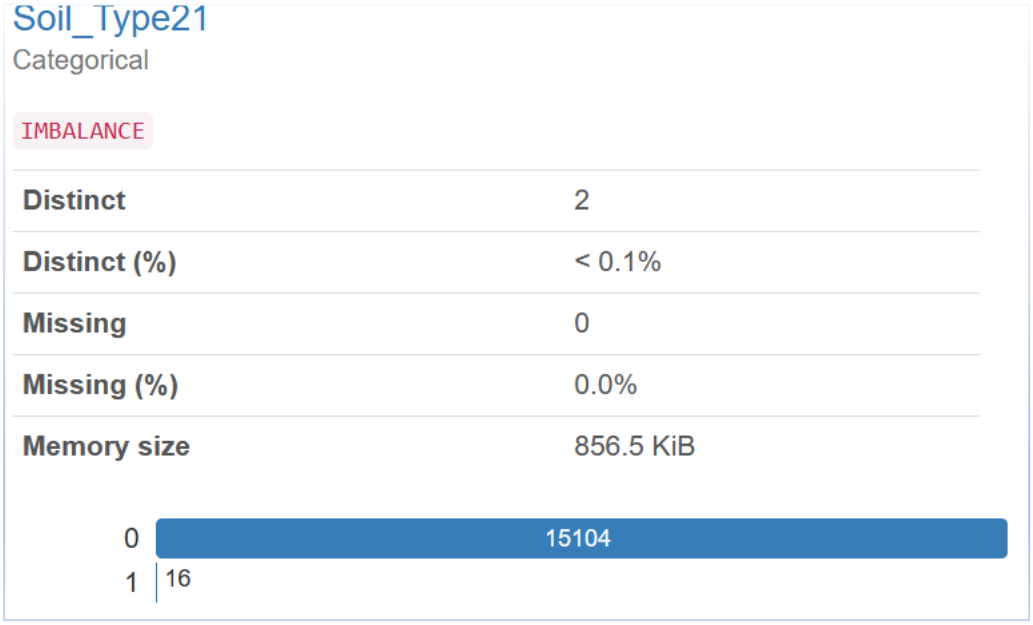
Here the variable 'Soil_type19' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.7% of data is 0 along with 0.3% as 1 which means 0.32% of data contains from 'Soil_Type19' and remaining from rest of the soils.

28. Soil Type20:



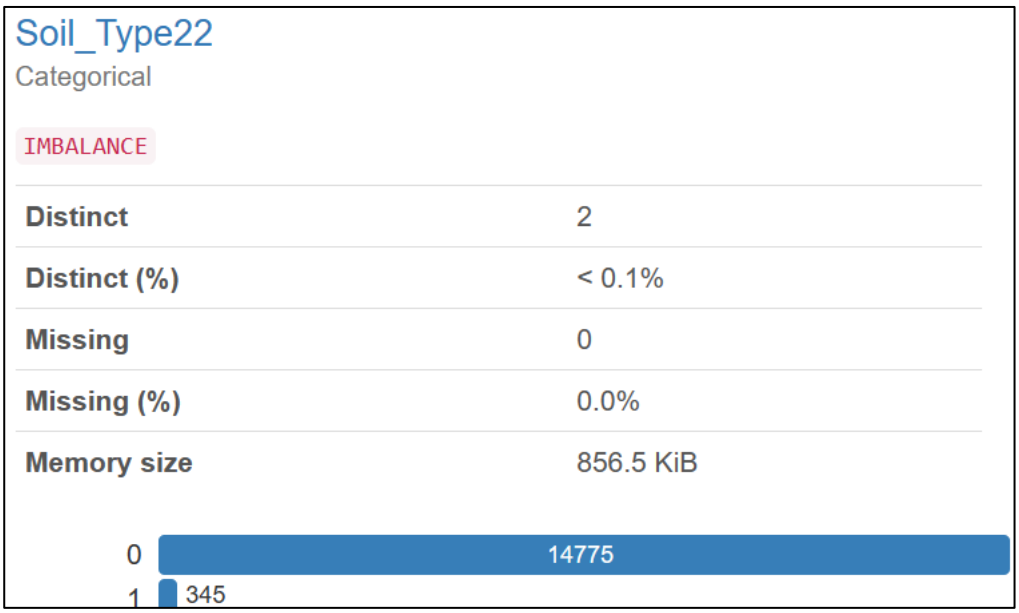
Here the variable 'Soil_Type20' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.1% of data is 0 along with 0.9% as 1 which means 0.9% of data contains from 'Soil_Type20' and remaining from rest of the soils.

29. Soil Type21:



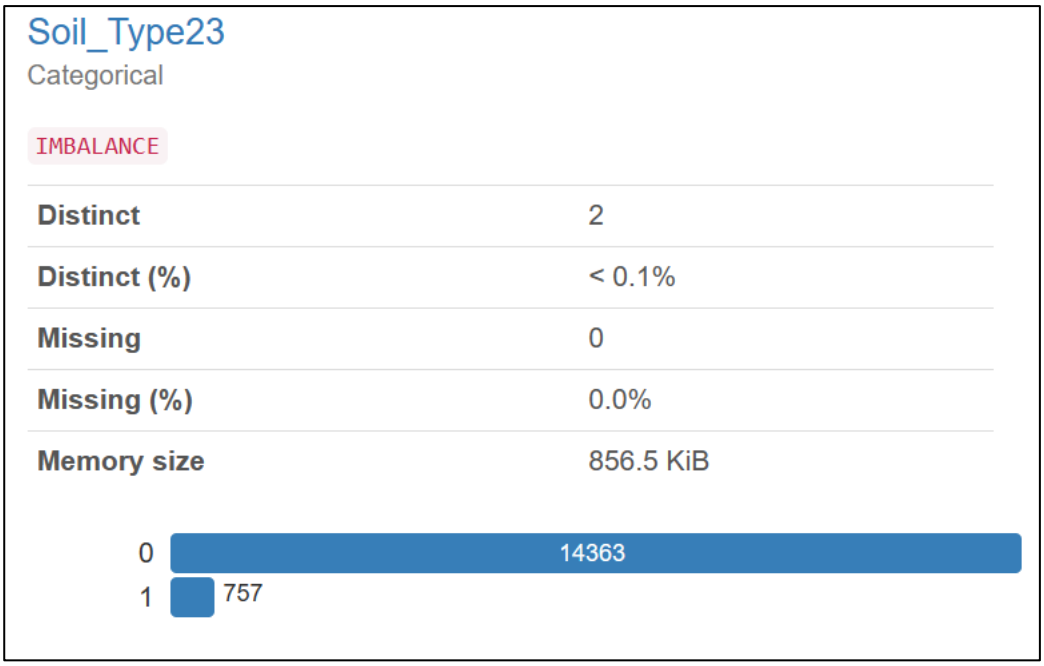
Here the variable 'Soil_Type21' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.9% of data is 0 along with 0.1% as 1 which means 0.1% of data contains from 'Soil_Type21' and remaining from rest of the soils.

30. Soil Type22:



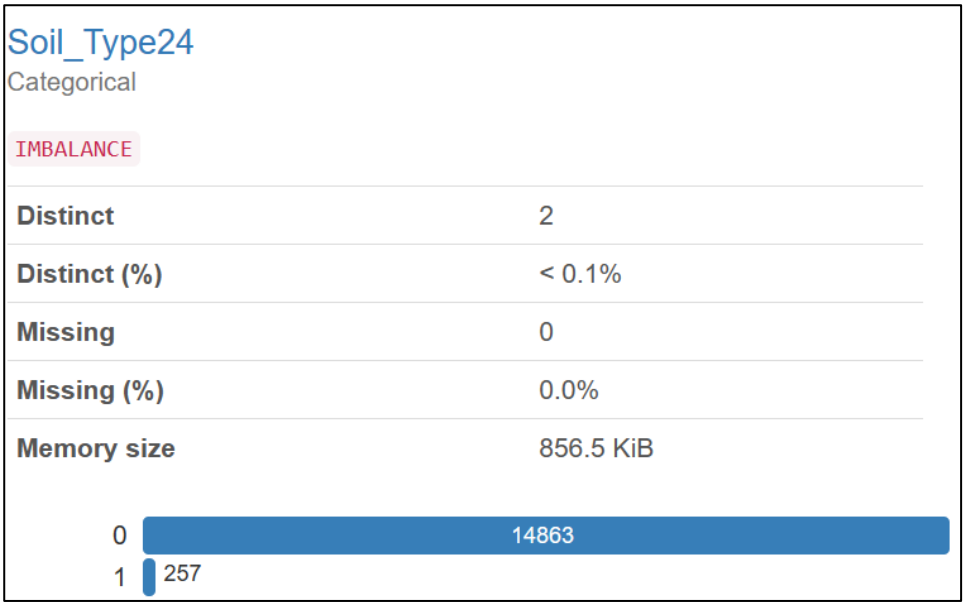
Here the variable 'Soil_Type22' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 97.7% of data is 0 along with 2.3% as 1 which means 2.3% of data contains from 'Soil_Type22' and remaining from rest of the soils.

31. Soil Type23:



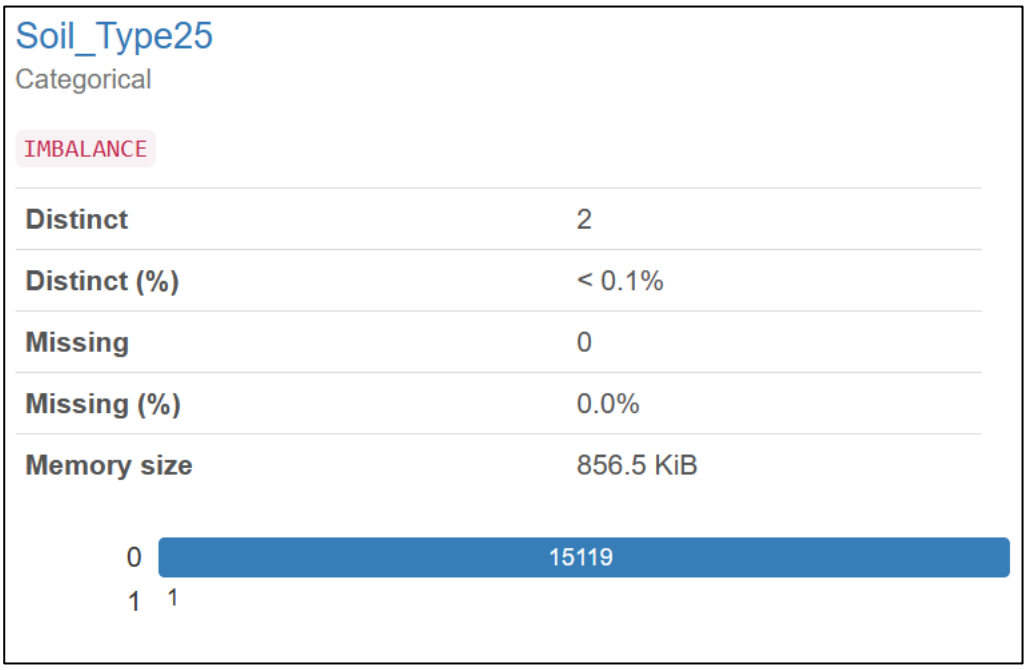
Here the variable 'Soil_Type23' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.0% of data is 0 along with 5% as 1 which means 5% of data contains from 'Soil_Type23' and remaining from rest of the soils.

32. Soil Type24:



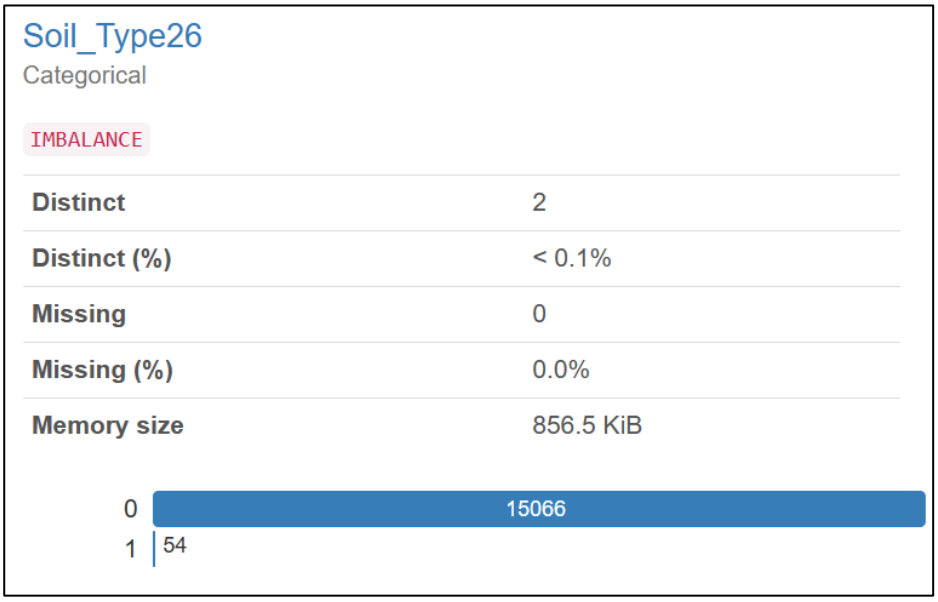
Here the variable 'Soil_Type24' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 98.3% of data is 0 along with 1.7% as 1 which means 1.7% of data contains from 'Soil_Type24' and remaining from rest of the soils.

33. Soil Type25:



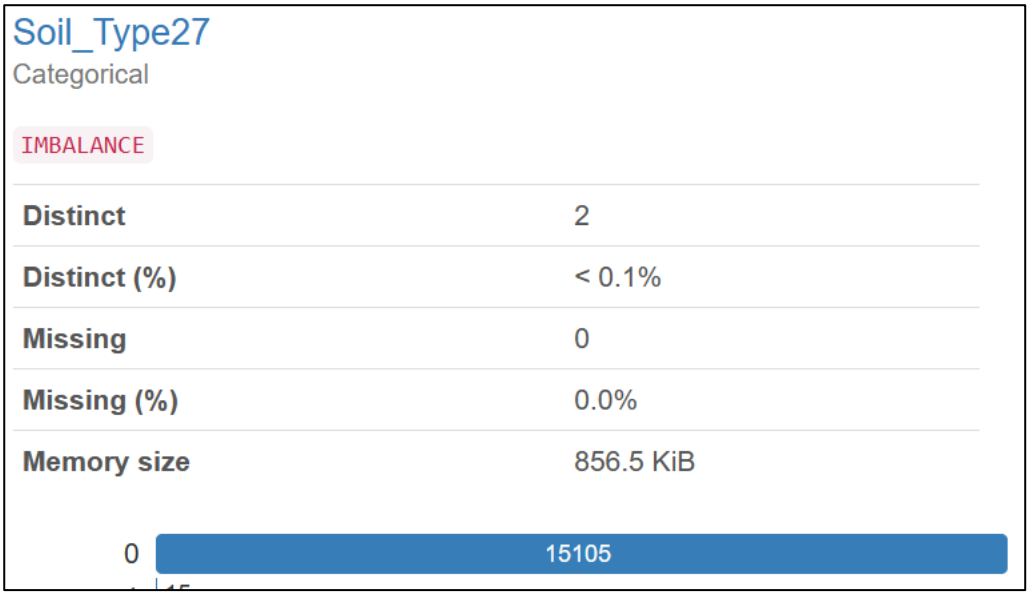
Here the variable 'Soil_Type25' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that greater than 99.9% of data is 0 along with less than 0.1% as 1 which means less than 0.1% of data contains from 'Soil_Type25' and remaining from rest of the soils.

34. Soil Type26:



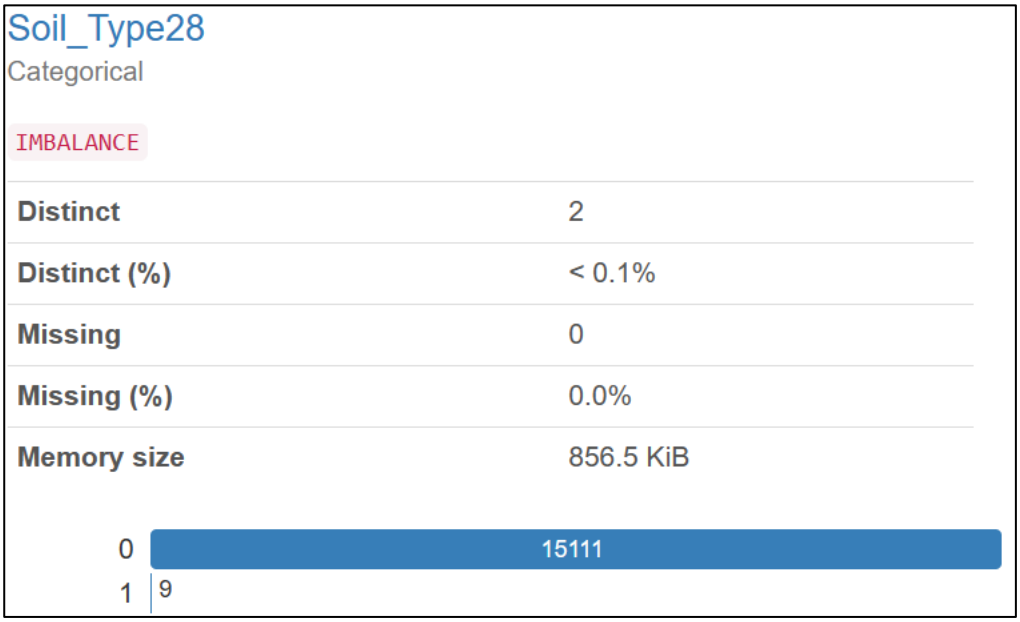
Here the variable 'Soil_Type26' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.6% of data is 0 along with 0.4% as 1 which means 0.4% of data contains from 'Soil_Type26' and remaining from rest of the soils.

35. Soil Type 27:



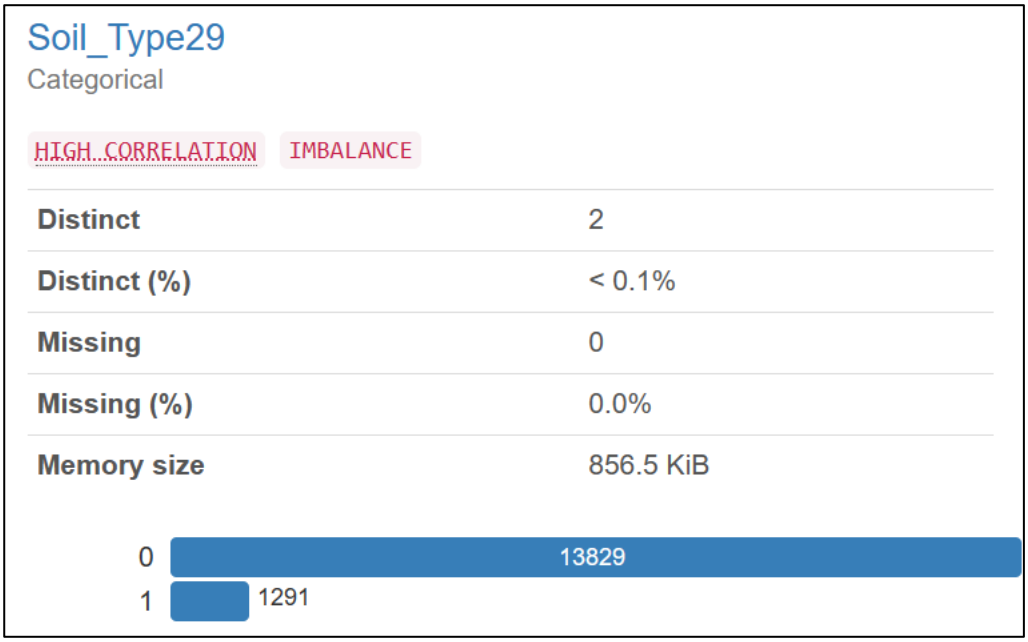
Here the variable 'Soil_Type27' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.9% of data is 0 along with 0.1% as 1 which means 0.1% of data contains from 'Soil_Type27' and remaining from rest of the soils.

36. Soil Type28:



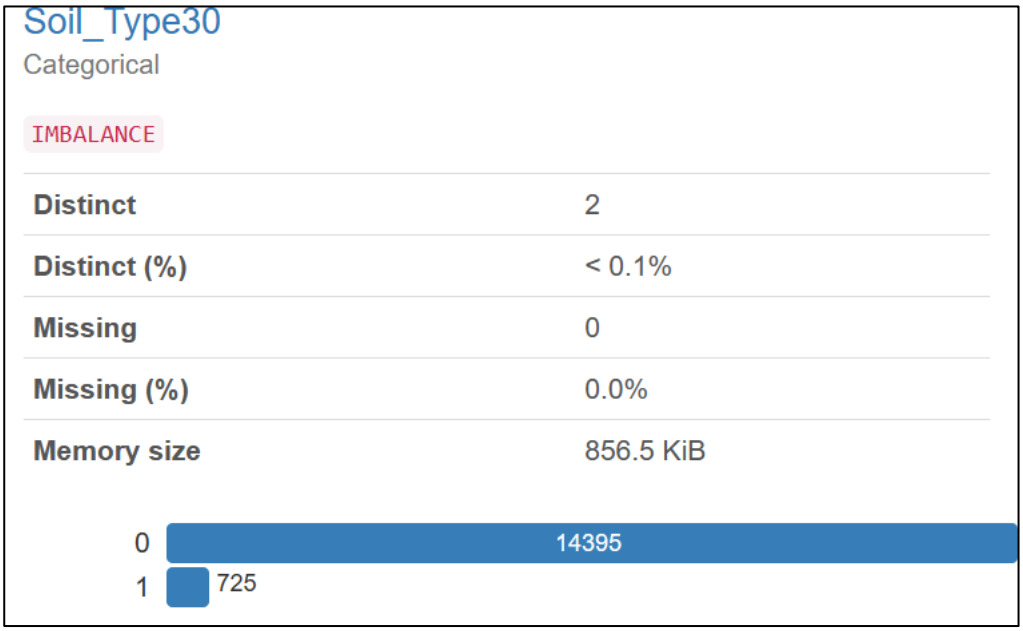
Here the variable 'Soil_Type28' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.9% of data is 0 along with 0.1% as 1 which means 0.1% of data contains from 'Soil_Type28' and remaining from rest of the soils.

37. Soil Type 29:



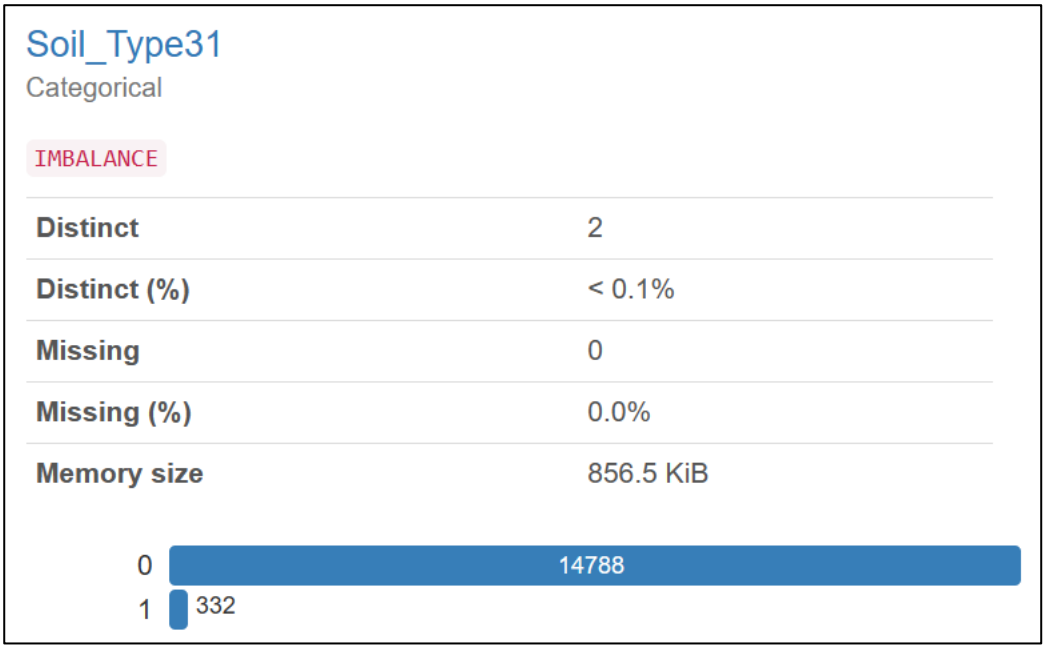
Here the variable 'Soil_Type29' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 91.5% of data is 0 along with 8.5% as 1 which means 8.5% of data contains from 'Soil_Type29' and remaining from rest of the soils.

38. Soil Type 30:



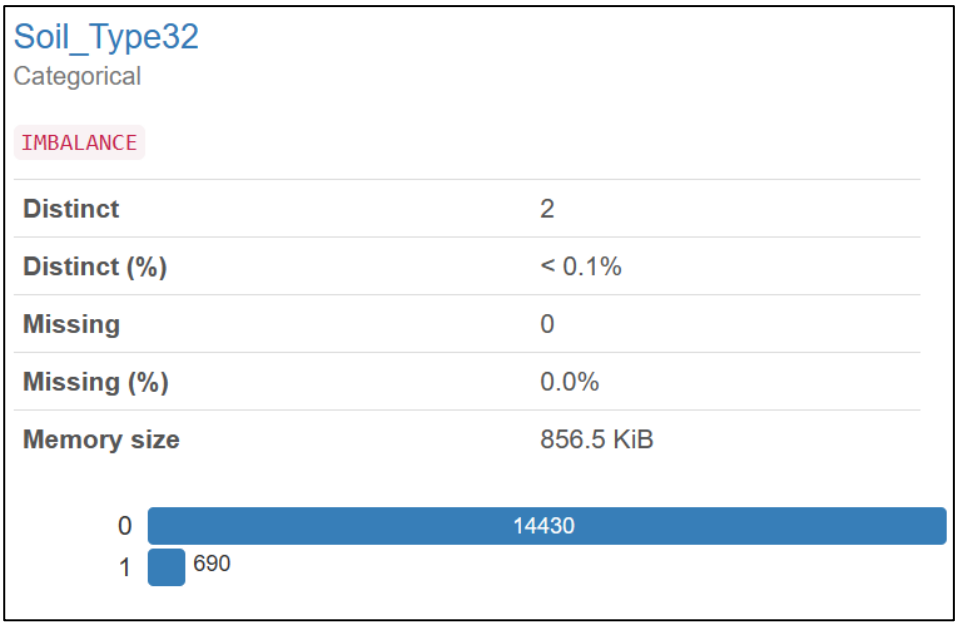
Here the variable 'Soil_Type30' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.2% of data is 0 along with 4.8% as 1 which means 4.8% of data contains from 'Soil_Type30' and remaining from rest of the soils.

39. Soil Type 31:



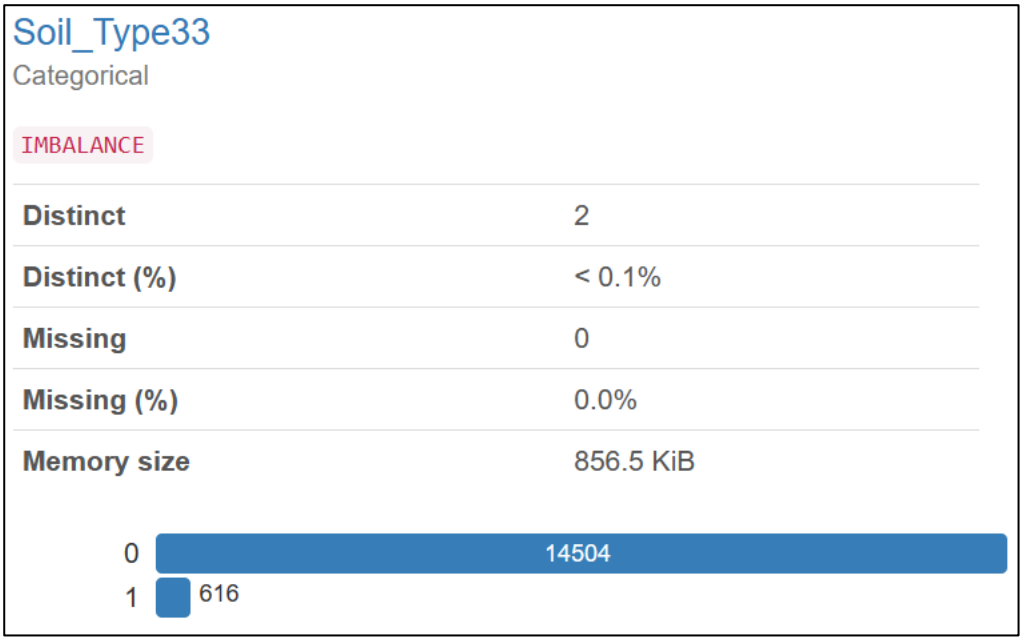
Here the variable 'Soil_Type31' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 97.8% of data is 0 along with 2.2% as 1 which means 2.2% of data contains from 'Soil_Type31' and remaining from rest of the soils.

40. Soil Type32:



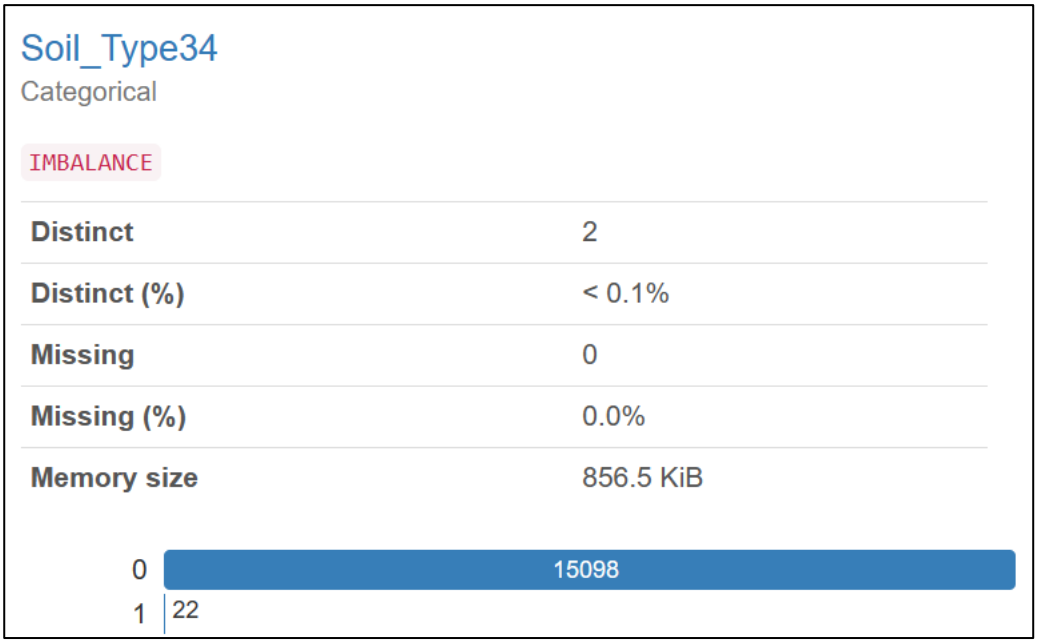
Here the variable 'Soil_Type32' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.4% of data is 0 along with 4.6% as 1 which means 4.6% of data contains from 'Soil_Type32' and remaining from rest of the soils.

41. Soil Type 33:



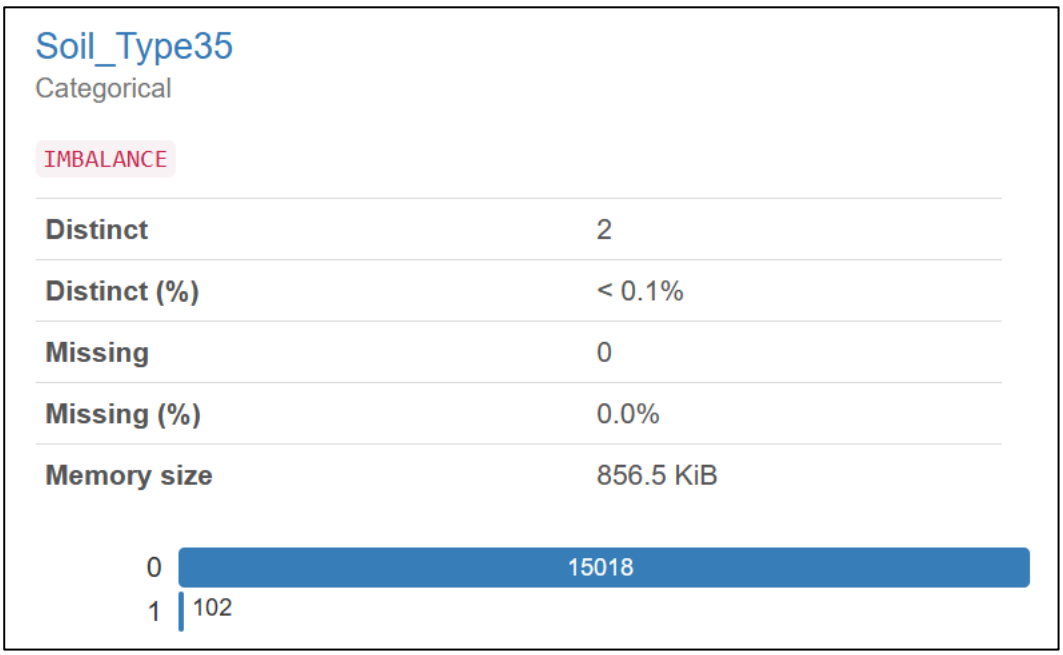
Here the variable 'Soil_Type33' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.9% of data is 0 along with 4.1% as 1 which means 4.1% of data contains from 'Soil_Type33' and remaining from rest of the soils.

42. Soil Type 34:



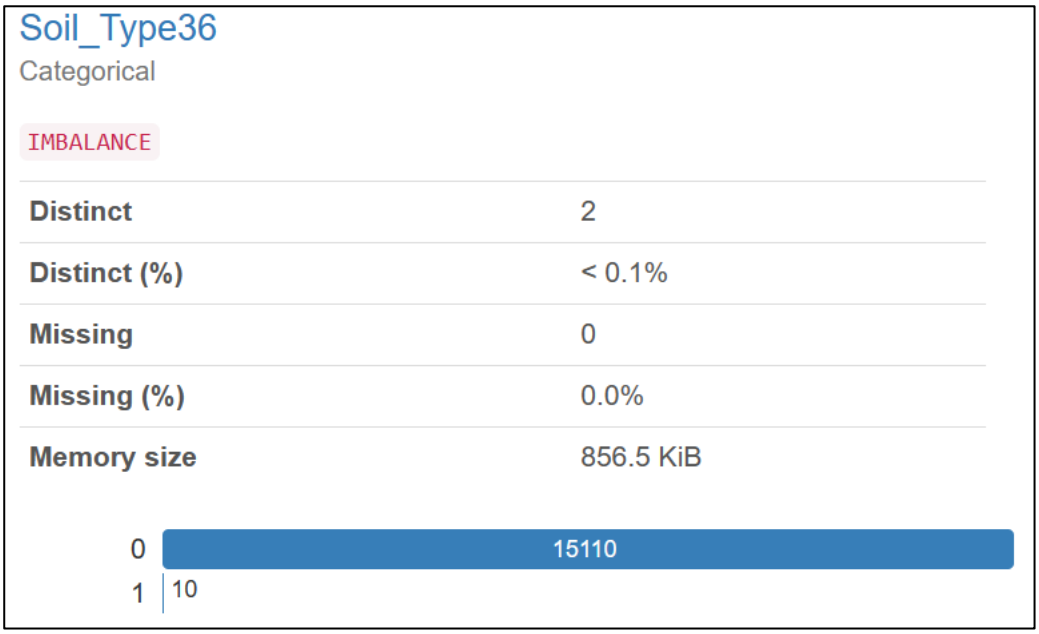
Here the variable 'Soil_Type34' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.9% of data is 0 along with 0.1% as 1 which means 0.1% of data contains from 'Soil_Type34' and remaining from rest of the soils.

43. Soil Type35:



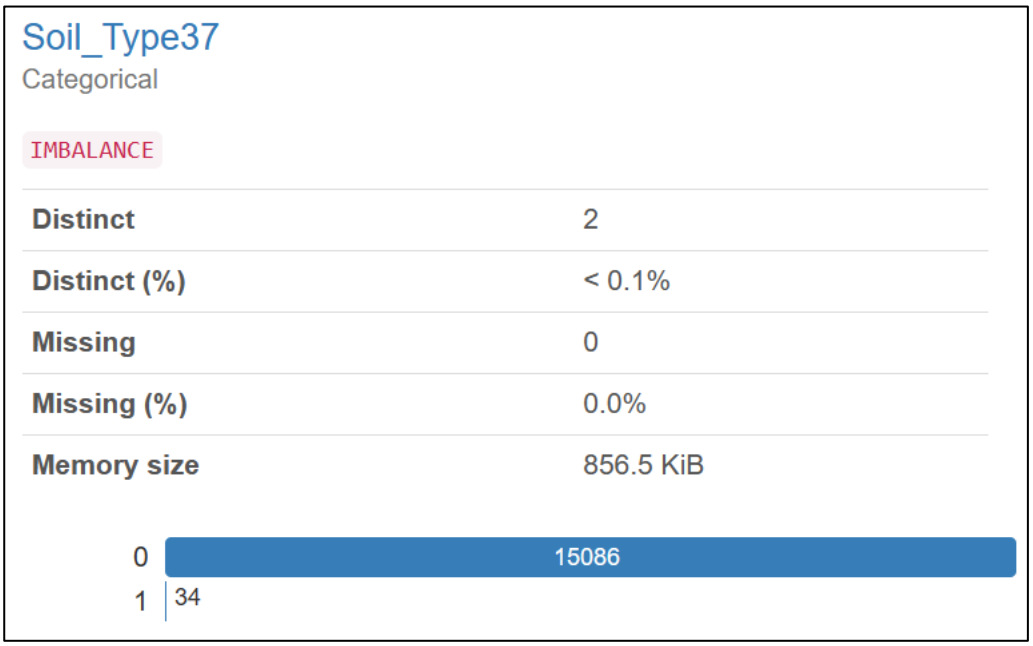
Here the variable 'Soil_Type35' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.3% of data is 0 along with 0.7% as 1 which means 0.7% of data contains from 'Soil_Type35' and remaining from rest of the soils.

44. Soil Type36:



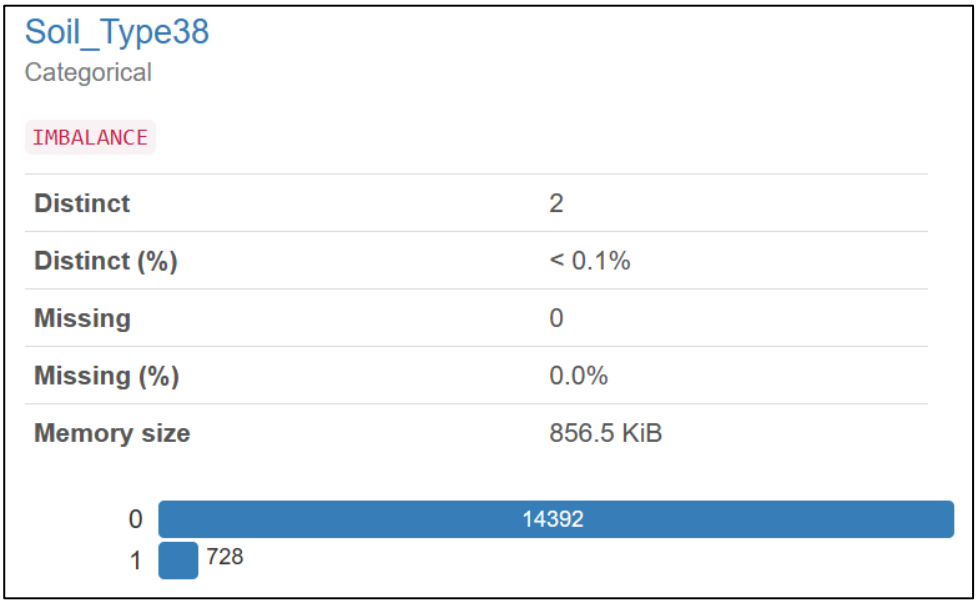
Here the variable 'Soil_Type36' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.9% of data is 0 along with 0.1% as 1 which means 0.1% of data contains from 'Soil_Tpe36' and remaining from rest of the soils.

45. Soil Type37:



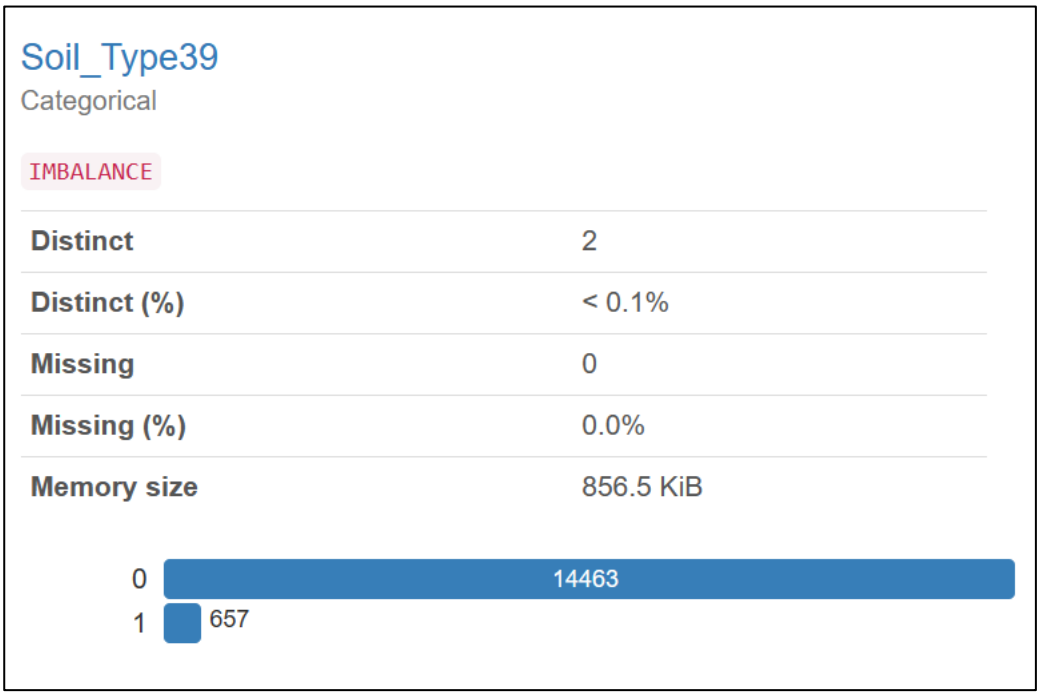
Here the variable 'Soil_Type37 has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 99.8% of data is 0 along with 0.2% as 1 which means 0.2% of data contains from 'Soil_Type37 and remaining from rest of the soils.

46. Soil Type38:



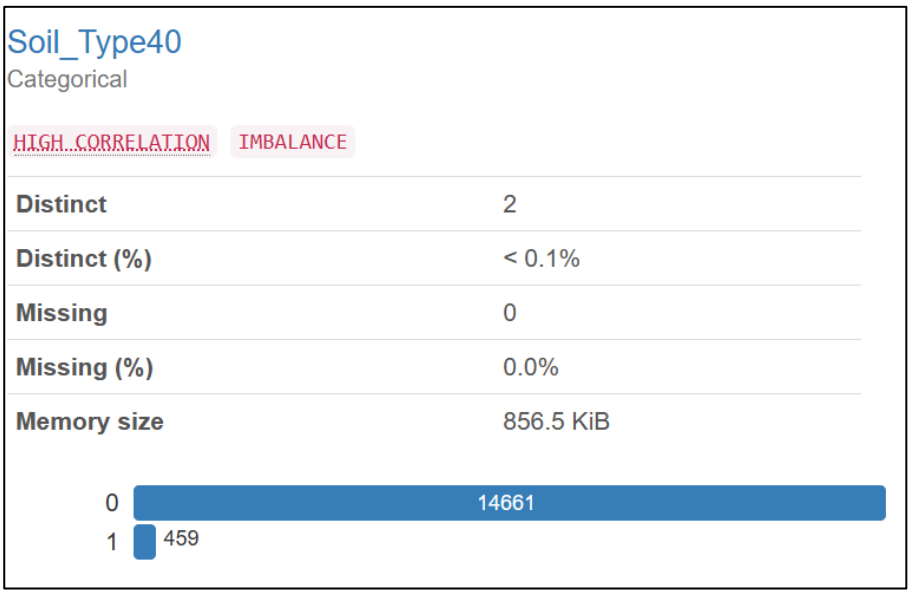
Here the variable 'Soil_Type38' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.2% of data is 0 along with 4.8% as 1 which means 4.8% of data contains from 'Soil_Type38' and remaining from rest of the soils.

47. Soil Type39:



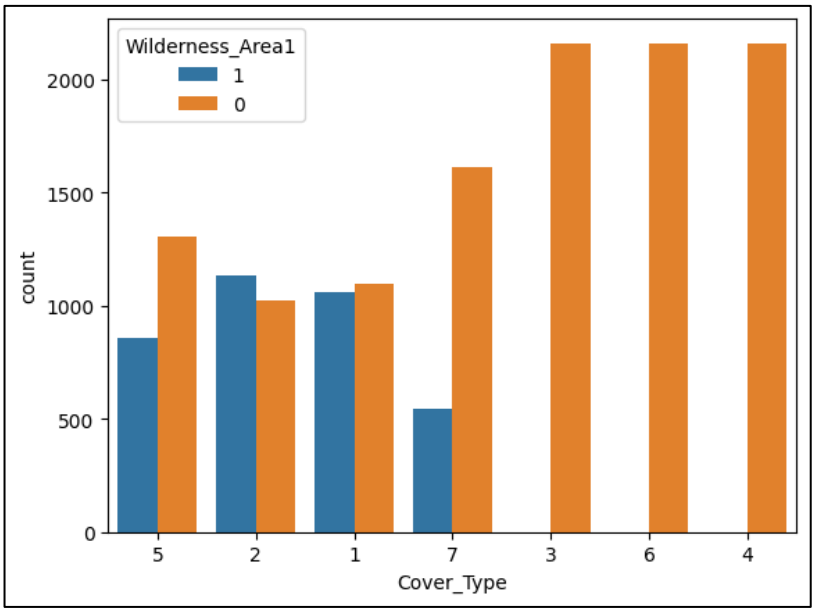
Here the variable 'Soil_Type39' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 95.7% of data is 0 along with 4.3% as 1 which means 4.3% of data contains from 'Soil_Type39'and remaining from rest of the soils.

48. Soil Type 40:



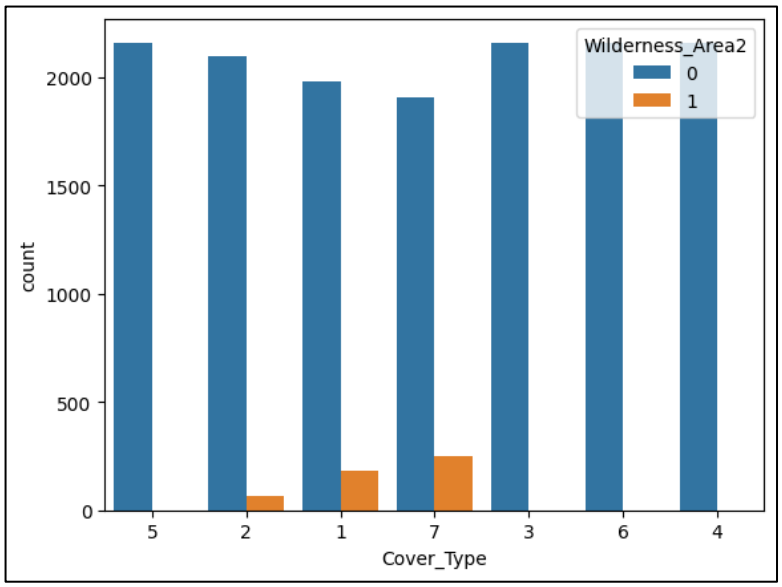
Here the variable 'Soil_Type40' has two categories yes and no in its record which is encoded to 1 and 0 respectively. we can see from the chart that 97.0% of data is 0 along with 3.0% as 1 which means 3.0% of data contains from 'Soil_Type40' and remaining from rest of the soils.

49. Wilderness Area 1:



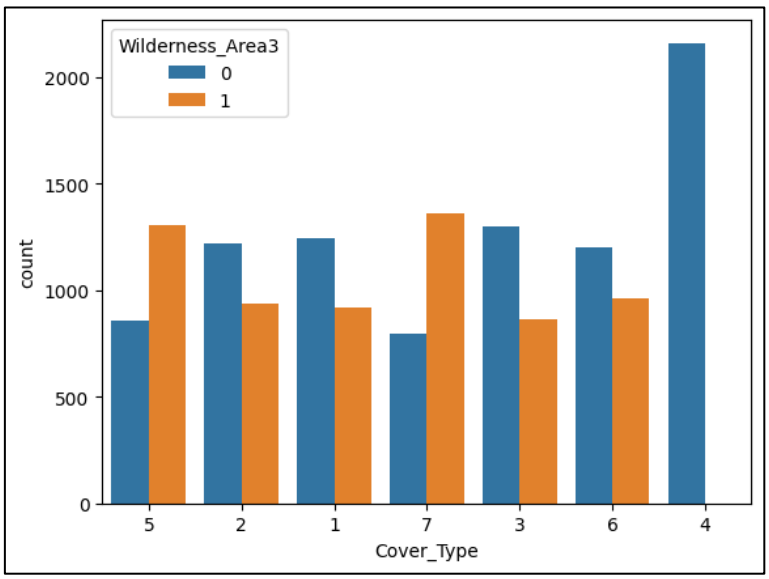
As we can see the plot from wilderness area1 has most forest cover types are Spruce/Fir, Lodgepole Pine, Krummholz and Aspen and remaining forests present in other areas. Here we conclude that moderate amount of data available from Wilderness area 1.

50. Wilderness Area 2:



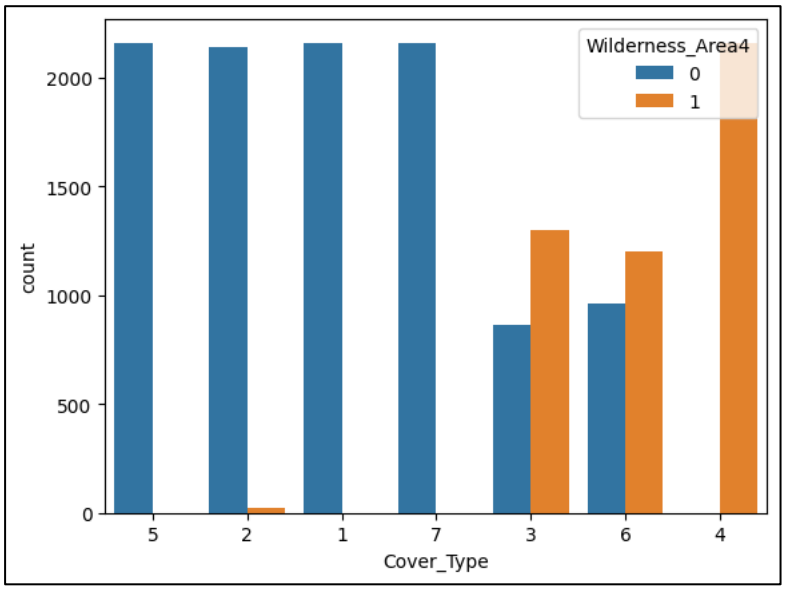
As we can see the plot from wilderness area 2 has most forest cover types are Spruce/Fir, Lodgepole Pine, Krummholz and remaining forests present other areas. Here we can conclude that data contains very less types of forest cover type available from the wilderness Area 2.

51. Wilderness Area 3:



As we can see the plot from wilderness area 3 most cover types are Spruce/Fir, Lodgepole Pine, Ponderosa Pine, Aspen, Douglas-fir, Krummholz and there is no data available from Cottonwood/Willow. Here we conclude that most of the data available from forest cover type contains wilderness Area 3.

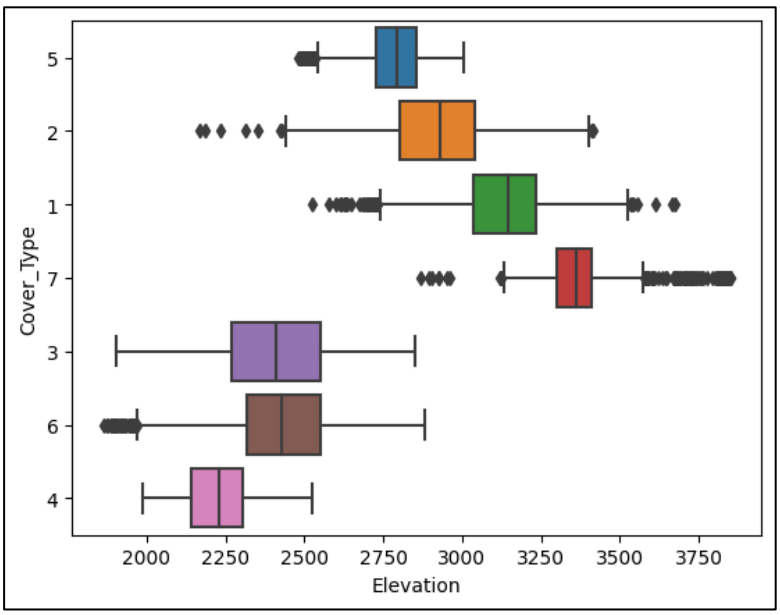
52. Wilderness Area 4:



As we can see the plot from wilderness area 4 has most forest cover type are Lodgepole/Fir, Ponderosa Pine, Douglas-fir, Cottonwood/willow and remaining forests present other areas. Here we conclude that cottonwood/willow forest cover type is very high in Wilderness area 4.

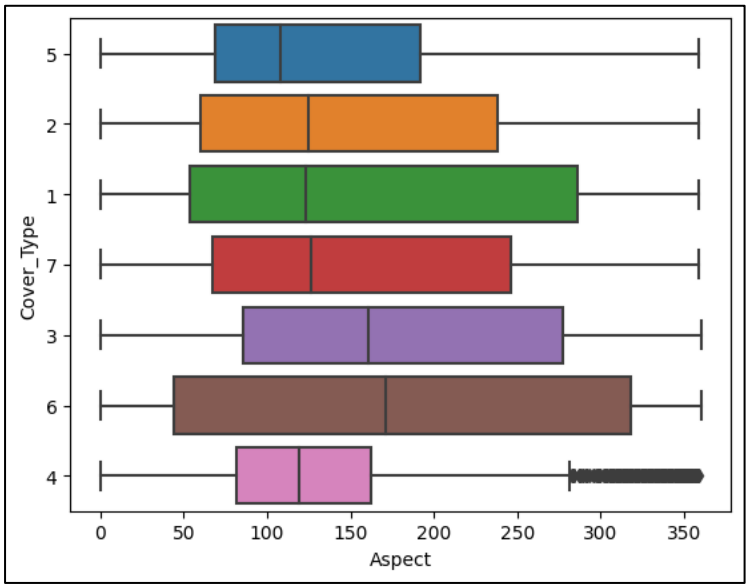
3.2. BIVARIATE ANALYSIS

1. Bivariate Analysis of Elevation and Cover Type:



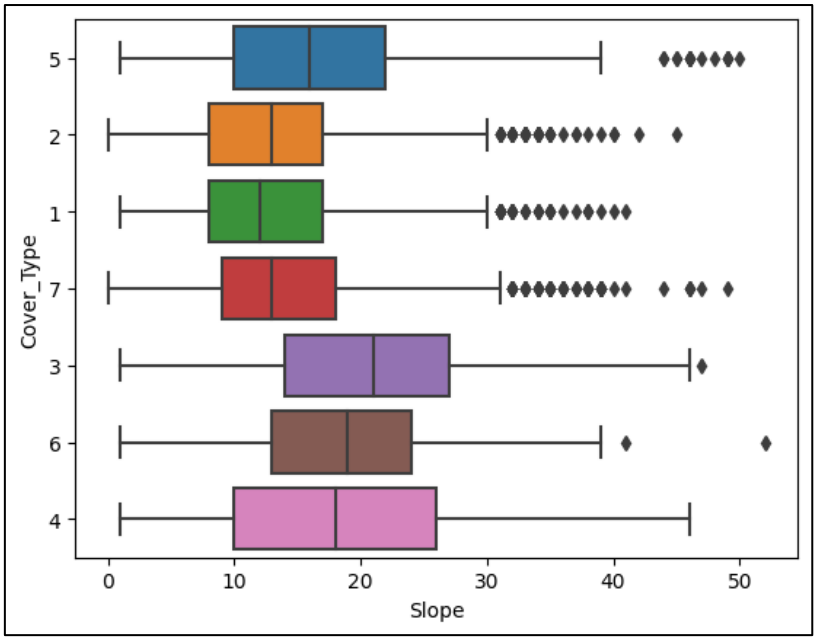
The box plot indicates that for Cover Type 7, Cover Type 2, Cover Type 5, and Cover Type 1, are having high median values when compared to Cover Type 3, Cover Type 6, and Cover Type 4.

2. Bivariate Analysis of Aspect and Cover Type:



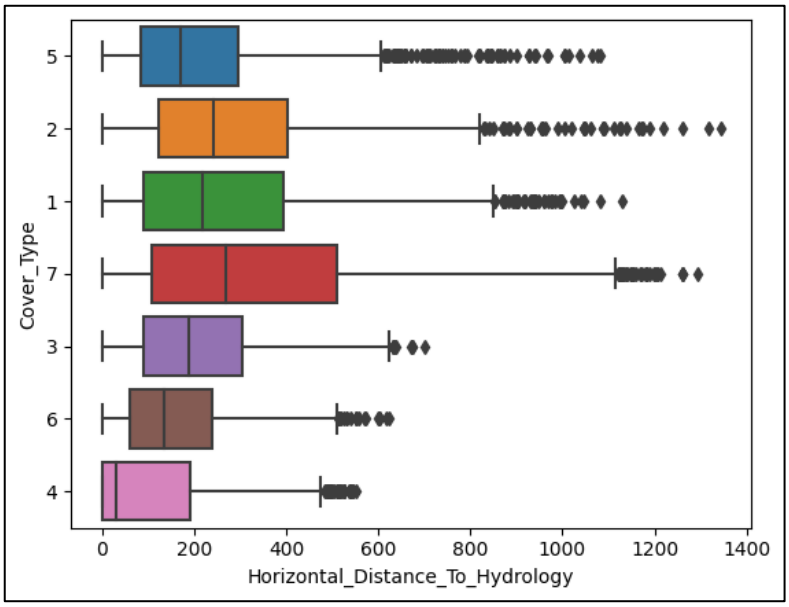
The box plot indicates that for Cover Type 1 and Cover Type 6, Aspect values are widely dispersed, as evidenced by the box size. Additionally, some Cover Type median values overlap, suggesting that Aspect might not be a strong variable for accurate predictions.

3. Bivariate Analysis of Slope and Cover Type:



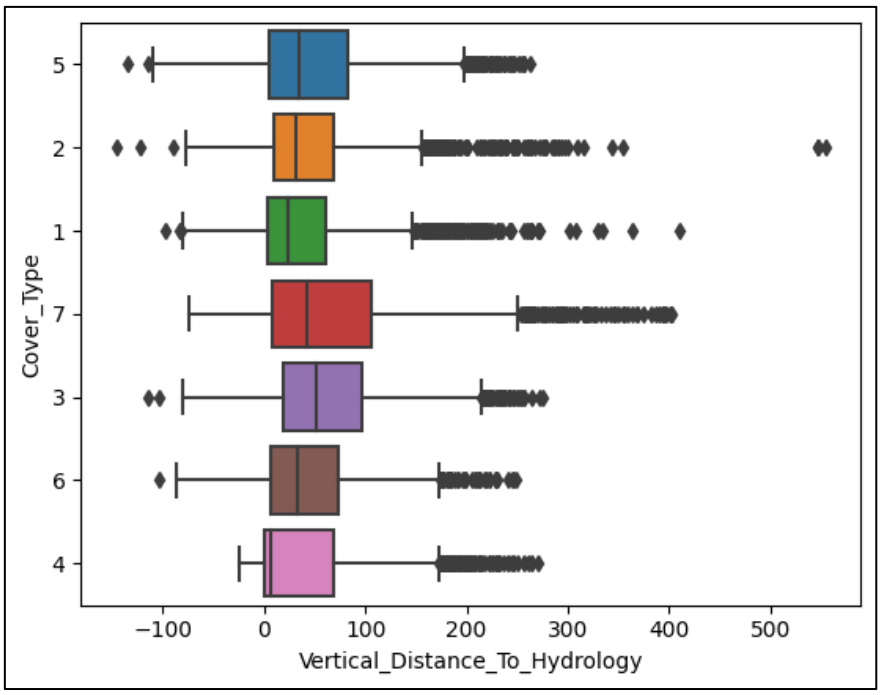
The boxplot reveals that median values are the same for certain slope values across different cover types. Further statistical analysis will be carried out to determine the significance of this observation.

4. Horizontal Distance to Hydrology and Cover Type:



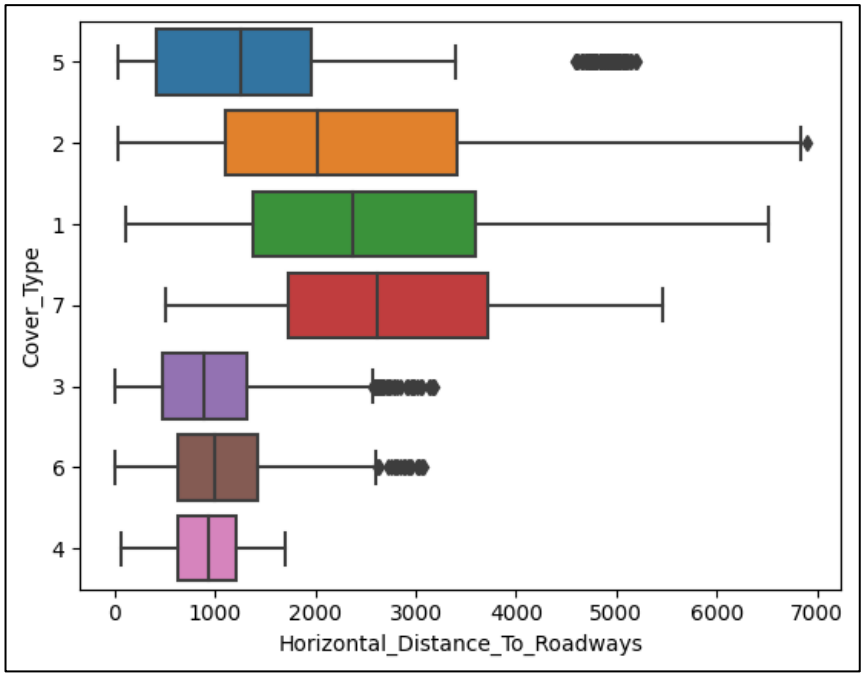
The box plot illustrates the distribution of values for each cover type in horizontal distance to hydrology. The median values show slight differences among cover types, and there are some extreme values presents.

5. Vertical Distance to Hydrology and Cover Type:



The box plot indicates nearly equal median values for vertical distance to hydrology across cover types, suggesting this may not be a significant feature for model building.

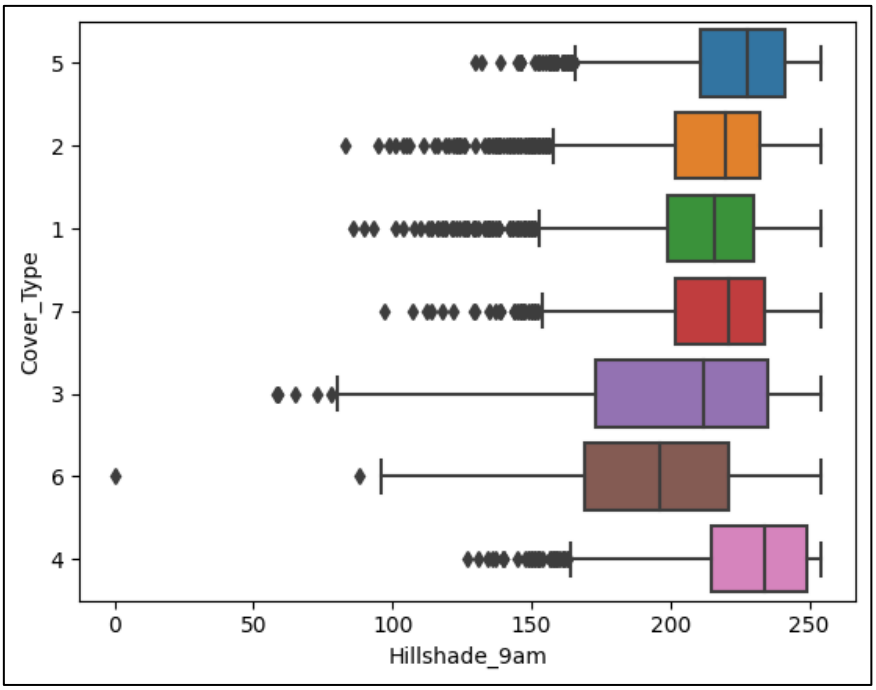
6. Horizontal Distance to Roadways and Cover Type:



The boxplot clearly shows differences in how far the forest areas are from roadways. Cover Type 1 and Cover Type 2, especially Type 2, tend to have higher

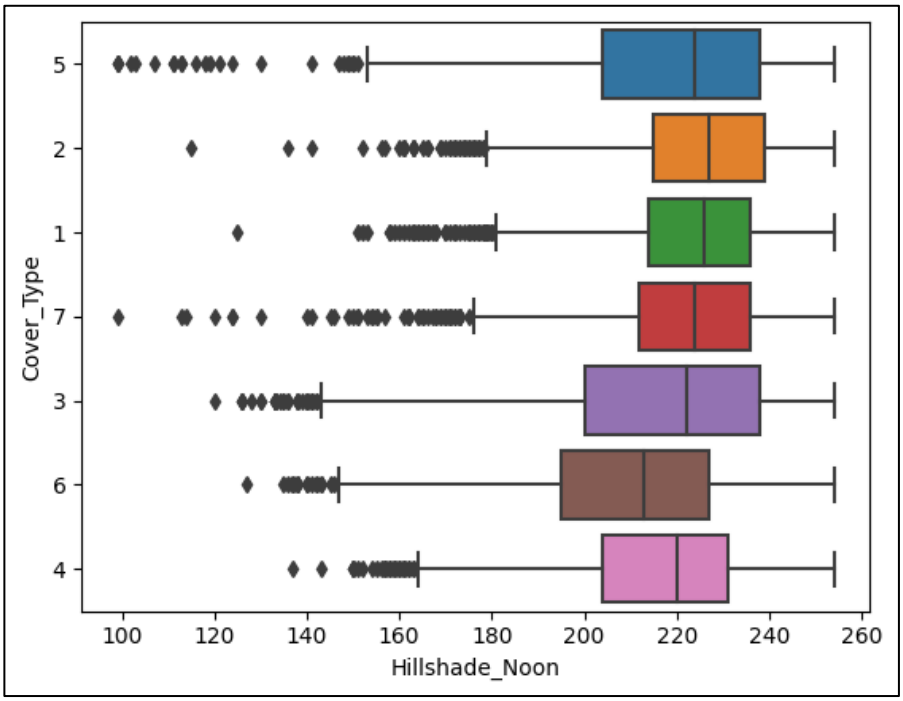
distances. The middle line (median) also varies, indicating distinctions between cover types.

7. Hillshade 9AM and Cover Type:



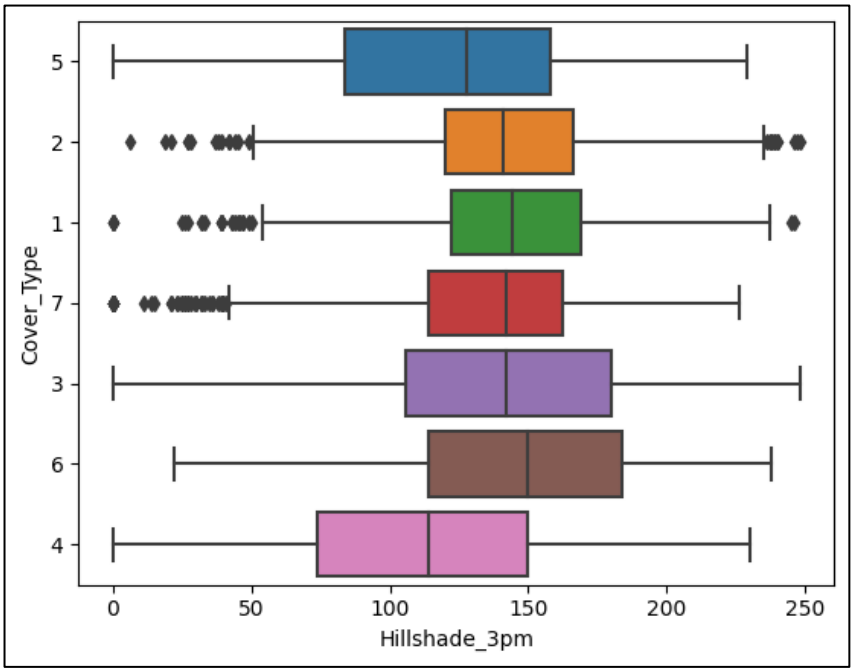
The boxplot indicates how the data is spread for the Hillshade at 9 am. The median values are relatively close, suggesting some similarity in the distribution across different categories.

8. Hillshade Noon and Cover Type:



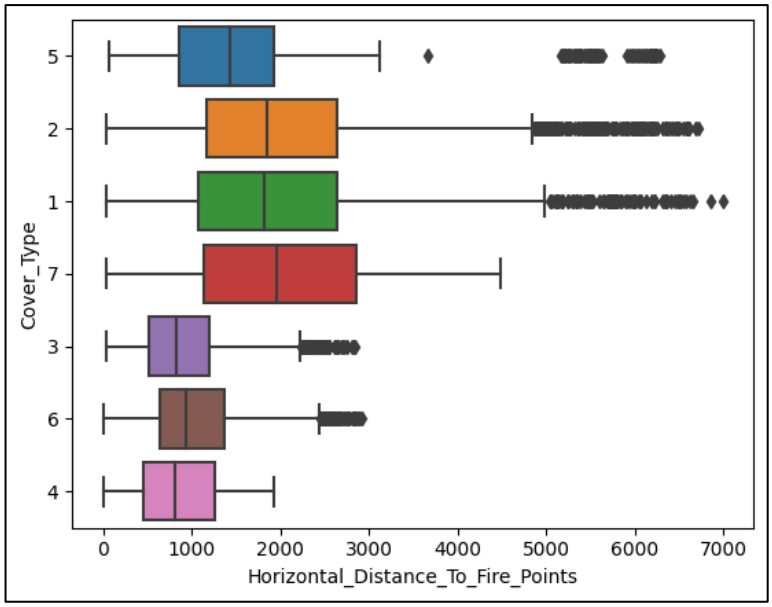
The boxplot indicates that median values are close for Hillshade at Noon, implying similarity. Further statistical analysis is needed to determine its significance as a feature in relation to cover types.

9. Hillshade 3PM and Cover Type:



The boxplot illustrates that median values are relatively close for Hillshade at 3 PM. A detailed statistical analysis will be conducted to assess its significance in relation to different cover types.

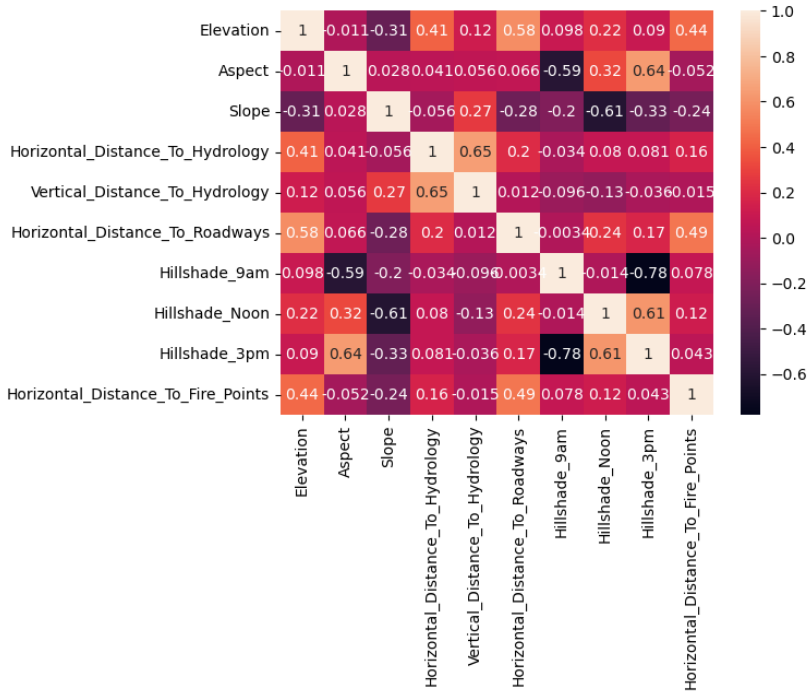
10. Horizontal Distance to Fire Points and Cover Type:



The boxplot indicates noticeable differences in median values among various cover types. However, cover types 1 and 2 show relatively similar median values, suggesting a potential overlap in their distributions.

3.3. Numeric Feature correlation:

```
: sns.heatmap(train.corr(),annot=True)
```



To check the correlation between the variables, Spearman's correlation is used. The Spearman's rank correlation coefficient (ρ) is a measure of monotonic correlation between two variables, and is therefore better in catching nonlinear monotonic

correlations than Pearson's r. Its value lies between -1 and +1, -1 indicating total negative monotonic correlation, 0 indicating no monotonic correlation and 1 indicating total positive monotonic correlation.

4. DATA PREPROCESSING

4.1. DATA CLEANING:

Checking Missing Values:

To understand trees data frame, let us look at the data types and descriptive statistics. With pandas **isnull().sum().sum()** method, we can list the non-null values and data types:

```
: train.isnull().sum().sum()
: 0
```

There are no missing attribute values in the dataset.

4.2. Scaling the Features for Model Optimization:

Here we are using robust scaling to scale the independent variable.

If there are too many outliers in the data, they will influence the mean and the max value or the min value. Thus, even if we scale this data using the above methods, we cannot guarantee a balanced data with a normal distribution.

The Robust Scaler, as the name suggests is not sensitive to outliers. This scaler-

1. removes the median from the data.
2. scales the data by the Interquartile Range (IQR)

It is the difference between the first and third quartile of the variable. The interquartile range can be defined as-

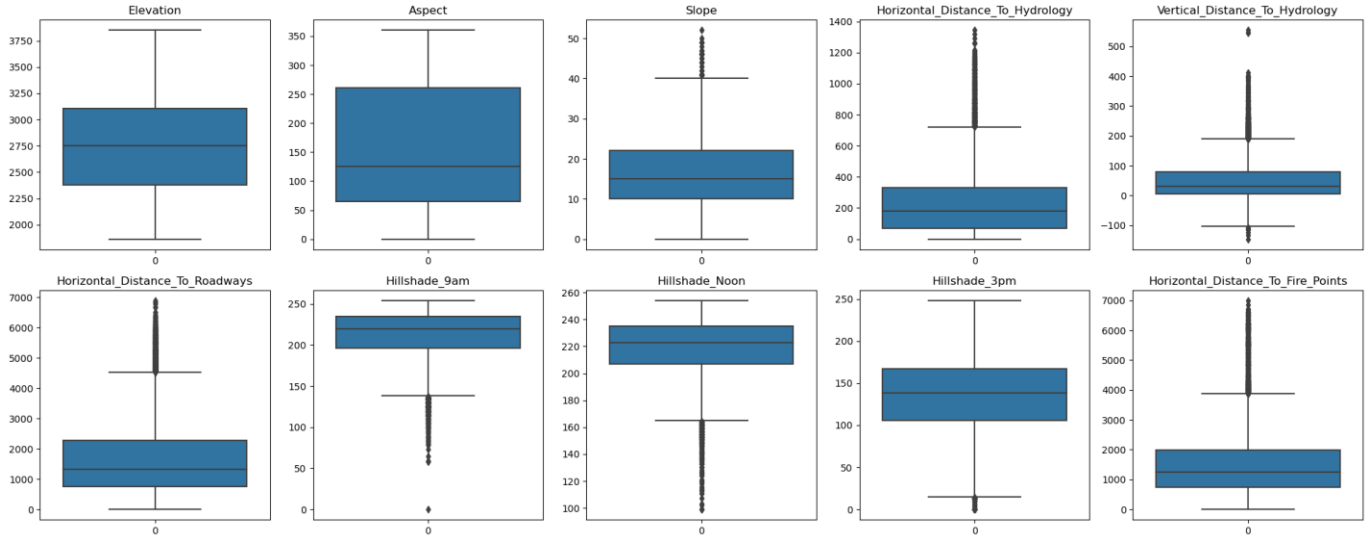
$$IQR = Q3 - Q1$$

Thus, the formula would be:

x_scaled = (x – Q1)/(Q3 – Q1)

4.3. Outlier Treatment Approach:

Exploring Numerical Feature Distributions and Outliers



The boxplots reveal the presence of outliers in specific columns, namely
'Horizontal_Distance_To_Hydrology',
'Vertical_Distance_To_Hydrology,'
'Horizontal_Distance_To_Roadways,'
'Hillshade_9am,'
'Hillshade_Noon,'
'Horizontal_Distance_To_Fire_Points.'

On observation Among these 'Horizontal_Distance_To_Roadways', 'Horizontal_Distance_To_Fire_Points' exhibit a higher frequency of outliers compared to the other variables with a lower frequency of outliers.

Box plots were generated to assess the relationship between independent features and the target variable, revealing the presence of outliers. The boxplots illustrate the existence of outliers in certain columns, including 'Horizontal_Distance_To_Hydrology,' 'Vertical_Distance_To_Hydrology,' 'Horizontal_Distance_To_Roadways,' 'Hillshade 9am,' 'Hillshade_Noon,' and 'Horizontal Distance_To_Fire Points.'

Here we can say that mean and median almost same values are same for the columns - Slope, hillshade_3pm, hillshade_noon, hillshade_9am.we cannot doing any outlier treatment for this columns.

The mean and median values of the columns 'Horizontal_Distance_To_Hydrology', 'Vertical_Distance_To_Hydrology', 'Horizontal_Distance_To_Roadways', and 'Horizontal_Distance_To_Fire_Points' are different, indicating a lack of symmetry in their distributions. To address this issue specifically for these columns, the decision was made to utilize the KNN (K-Nearest Neighbours) imputer.
The KNN imputer is a robust technique employed to handle missing or outlier values by leveraging distances to the k nearest neighbours. By using the characteristics and values of neighboring data points, this method predicts and replaces missing or outlier values more accurately. It helps ensure a more representative depiction of the dataset's distribution, effectively maintaining its integrity while mitigating the impact of outliers on subsequent analysis.

4.4. Addressing Multicollinearity in the Feature:

To mitigate multicollinearity within the feature set, we employ the Variance Inflation Factor (VIF) method. This technique helps us identify and eliminate highly correlated columns to ensure that the predictive variables are independent and do not introduce redundancy into the model. If two columns exhibit similar significance and demonstrate high correlation, we opt to retain only one of them. Following the application of the VIF method, we have identified and removed specific columns, including 'Wilderness_Area1,' 'Soil_Type10,' 'Hillshade_Noon,' 'Hillshade_9am,' 'Hillshade_3pm,' 'Wilderness_Area4,' and 'Slope,' which displayed elevated correlation levels. This process enhances the robustness and accuracy of the model by ensuring that the selected features contribute unique information without introducing collinearity-related issues.

5. MODEL BUILDING

For model building we are Splitting the dataset into an 80-20 ratio for training and testing.

Base Model:

In the base model, we employed Logistic Regression, Decision Tree, and Random Forest to predict the target variable based on cover types. The target variable comprises 7 classes, where each class corresponds to a specific forest cover type:

1. Class 1: Spruce/Fir
2. Class 2: Lodgepole Pine
3. Class 3: Ponderosa Pine
4. Class 4: Cottonwood/Willow
5. Class 5: Aspen

6. Class 6: Douglas-fir
7. Class 7: Krummholz

Decision Tree:

A decision tree is a flow chart structure which consist of internal node which represents a test in attribute and that comes with each branch out i.e. the result of the test and each leaf represents a category label (a decision taken after testing all attributes in the path from the beginning to the leaf). Each path from the source to a leaf can also be interpreted as a sorting rule.

When establishing a supervised classification model, the frequency distribution of attribute values is a potentially significant component in deciding the proportional importance of each attribute at various levels in the model construction procedure.

In data modeling, we can use frequency distributions to compute entropy. We calculate the entropy of multiplying the proportion of cases with each category label by the log of that proportion, and then getting the negative essence of those conditions.

$$\text{Entropy (S)} = -p1\log2\,(p1) - p2\log2\,(p2)$$

Where pi is proportion (relative frequency) of class i within the set S.

A decision tree is constructed algorithm that selects the best attribute, splits the data into subsets based on the values of that attribute present in the dataset and repeats the process on each of these subsets until a stopping condition is met.
Information gain measures the decrease in entropy that results from splitting a set of instances based on an attribute. IG (S, a) = entropy (S) – [p (s1) x entropy (s1) + p (s2) x entropy (s2)… +p (sn) x entropy (sn)].
Where n is the number of distinct values of attribute a, and si is the subset of S where all instances have the ith value of a.

After splitting the dataset into training and testing sets and fitting for Decision Tree model the metrics are:

```
              precision    recall  f1-score   support

           1       0.65      0.60      0.63       516
           2       0.59      0.58      0.59       516
           3       0.73      0.68      0.70       522
           4       0.92      0.92      0.92       509
           5       0.80      0.85      0.83       504
           6       0.72      0.76      0.74       542
           7       0.88      0.90      0.89       520

    accuracy                           0.76      3629
   macro avg       0.75      0.76      0.76      3629
weighted avg       0.75      0.76      0.75      3629

[[312 129   2   0  11   5  57]
 [107 299  13   0  66  23   8]
 [  3  13 357  23  17 109   0]
 [  0   0  25 467   0  17   0]
 [ 11  46   9   0 429   8   1]
 [  3   9  86  20  11 413   0]
 [ 46   7   0   0   0   0 467]]
cohen_kappa_score : 0.7154550345923871
```

- The model exhibits varying levels of performance across different classes, with some classes having higher precision and recall than others.
- Classes 4, 5, and 7 demonstrate strong classification performance, while classes 1 ,3and 6 show moderate performance.
- The overall model accuracy is decent, but it's important to consider class-specific metrics for a more nuanced evaluation.

Random Forest:

Random forest is an ensemble classifier. An ensemble consists of a set of individually trained classifiers whose predictions are combines for classifying new instances.

Random Forest is a classifier consisting of a collection of tree-structured classifiers {h(x, Θk) k=1, 2, ….}, where the {Θk} are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x.

For building a decision tree in random forest the steps have to be followed. If the number of records in the training set is N, then N records are sampled at random but with replacement, from the original data, this is bootstrap sample. This sample will be the training set for growing the tree. If there are M input variables, a number m << M is selected such that at each node, m variables are selected at random out of M and the best split on these m attributes is used to split the node.The value of m is held constant during forest growing. Each tree is grown to the largest extent possible. There is no pruning. In this way, multiple trees are induced in the forest; the number of trees is pre-decided by the parameter Ntree. The number of variables (m) selected at each node is also referred to as mtry or k in the literature. The depth of the tree can be controlled by a parameter node size (i.e. number of instances in the leaf node) which is usually set to one. For random forest python libraries like pandas and numpy are used which consist of set random forest classifier function in that tree estimators must add to get good results.

After splitting the dataset into training and testing sets and fitting for Random Forest model the metrics are:

```
              precision    recall  f1-score   support

           1       0.77      0.70      0.73       516
           2       0.74      0.70      0.72       516
           3       0.82      0.78      0.80       522
           4       0.92      0.97      0.95       509
           5       0.89      0.91      0.90       504
           6       0.82      0.86      0.84       542
           7       0.91      0.97      0.94       520

    accuracy                           0.84      3629
   macro avg       0.84      0.84      0.84      3629
weighted avg       0.84      0.84      0.84      3629

[[360  97   0   0  14   2  43]
 [ 83 359  15   0  37  17   5]
 [  0   5 408  26   5  78   0]
 [  0   0   9 496   0   4   0]
 [  7  21  11   0 461   4   0]
 [  0   1  56  16   2 467   0]
 [ 15   1   0   0   1   0 503]]
cohen_kappa_score : 0.8151260213840456
```

- The model shows a strong ability to differentiate between the classes, with high precision, recall, and F1-scores for most classes.
- Classes 4, 5, and 7 have particularly high precision and recall, indicating that the model is very accurate in predicting these classes.
- The overall model performance, as indicated by the accuracy of 84% and average metrics, is quite good.

Multinomial Logistic Regression:

Multinomial Logistic Regression is a classification algorithm suitable for predicting the probability of multiple classes. It extends binary logistic regression to handle more than two classes.

After splitting the dataset into training and testing sets and fitting for Multinomial Logistic Regression the metrics are:

```
Classifier: Multinomial Logistic Regression
Classification Report:
              precision    recall  f1-score   support

           1       0.64      0.64      0.64       516
           2       0.58      0.53      0.56       516
           3       0.61      0.54      0.57       522
           4       0.80      0.87      0.83       509
           5       0.67      0.74      0.70       504
           6       0.60      0.62      0.61       542
           7       0.86      0.86      0.86       520

    accuracy                           0.68      3629
   macro avg       0.68      0.69      0.68      3629
weighted avg       0.68      0.68      0.68      3629

Confusion Matrix:
[[329 100   1   0  20   5  61]
 [103 273  12   0  91  28   9]
 [  0   6 282  57  35 142   0]
 [  0   0  38 441   0  30   0]
 [ 10  79  18   0 372  25   0]
 [  0   9 108  52  35 338   0]
 [ 70   0   0   0   2   0 448]]
cohen_kappa_score : 0.6315701117107355
```

- The model shows variable performance across different classes, with some classes exhibiting higher precision and recall than others.
- Classes 4 and 7 demonstrate strong classification performance, while classes 1, 5, and 6 show moderate performance.
- The overall accuracy is 68%, indicating the proportion of correctly classified instances in the dataset.

K-nearest Neighbors (KNN) :

K-nearest Neighbors is a simple and versatile algorithm used for both classification and regression tasks. It belongs to the family of instance-based, lazy learning algorithms. It classifies a data point based on how its neighbors are classified.

After splitting the dataset into training and testing sets and fitting for K-nearest Neighbors the metrics are:

```
Classifier: K-nearest Neighbors
Classification Report:
              precision    recall  f1-score   support

           1       0.68      0.61      0.64       516
           2       0.62      0.54      0.58       516
           3       0.69      0.66      0.67       522
           4       0.87      0.93      0.90       509
           5       0.75      0.84      0.80       504
           6       0.70      0.74      0.72       542
           7       0.88      0.94      0.91       520

    accuracy                           0.75      3629
   macro avg       0.74      0.75      0.75      3629
weighted avg       0.74      0.75      0.75      3629

Confusion Matrix:
[[313 119   0   0  29   5  50]
 [107 277  17   2  74  25  14]
 [  0   6 343  44  15 114   0]
 [  0   0  19 474   0  16   0]
 [ 23  32  17   0 424   7   1]
 [  1   4  99  24  15 399   0]
 [ 16   8   0   0   5   0 491]]
cohen_kappa_score : 0.7080781334120738
```

- The K-nearest Neighbors model demonstrates good performance with an accuracy of 75%.
- The confusion matrix illustrates the model's ability to correctly classify instances into different classes.
- Class 4 and Class 7 exhibit particularly high precision, recall, and F1-score values.
- The Cohen's Kappa score of 0.708 indicates substantial agreement beyond chance.

Discriminant Analysis Classifier:

Discriminant Analysis is a statistical technique used for classification and dimensionality reduction. It finds the linear combinations of features that best separate two or more classes.
Discriminant Analysis aims to maximize the distance between the means of different classes while minimizing the spread within each class.

After splitting the dataset into training and testing sets the metrics are:

```
Classifier: Discriminant Analysis
Classification Report:
              precision    recall  f1-score   support

           1       0.59      0.63      0.61       516
           2       0.55      0.53      0.54       516
           3       0.52      0.57      0.55       522
           4       0.84      0.73      0.78       509
           5       0.64      0.66      0.65       504
           6       0.56      0.59      0.57       542
           7       0.87      0.80      0.83       520

    accuracy                           0.64      3629
   macro avg       0.65      0.64      0.65      3629
weighted avg       0.65      0.64      0.65      3629

Confusion Matrix:
[[323 114   1   0  21   2  55]
 [111 274  10   0  83  32   6]
 [  0   2 299  40  26 155   0]
 [  0   0  92 374   0  43   0]
 [ 13  90  47   0 332  22   0]
 [  0  17 123  30  51 321   0]
 [ 99   2   2   0   2   0 415]]
cohen_kappa_score : 0.584882935747236
```

- The model shows moderate performance across different classes.
- Classes 4 and 7 demonstrate relatively higher precision and recall.
- The overall accuracy is decent, but class-specific metrics provide a more nuanced evaluation.

Support Vector Machine (SVM) Classifier:

Support Vector Machine is a powerful supervised machine learning algorithm used for classification and regression tasks. It works by finding the hyperplane that best separates different classes in the feature space.
SVM aims to maximize the margin between different classes, where the margin is the distance between the hyperplane and the nearest data point of each class.

After splitting the dataset into training and testing sets the metrics are:

```
Classifier: Support Vector Machine
Classification Report:
              precision    recall  f1-score   support

           1       0.69      0.68      0.68       516
           2       0.68      0.59      0.63       516
           3       0.71      0.66      0.68       522
           4       0.84      0.96      0.90       509
           5       0.79      0.84      0.81       504
           6       0.71      0.71      0.71       542
           7       0.90      0.90      0.90       520

    accuracy                           0.76      3629
   macro avg       0.76      0.76      0.76      3629
weighted avg       0.76      0.76      0.76      3629

Confusion Matrix:
[[349 102   0   0  17   4  44]
 [ 93 306  13   0  68  27   9]
 [  0   5 344  50  18 105   0]
 [  0   0   9 489   0  11   0]
 [ 14  31  21   0 424  14   0]
 [  0   9  96  40  10 387   0]
 [ 52   0   1   0   1   0 466]]
cohen_kappa_score : 0.7222398510040167
```

- The SVM model demonstrates consistent performance across different classes.
- Classes 4 and 7 show particularly strong classification performance.
- The overall accuracy is impressive, with a high Cohen's Kappa score indicating substantial agreement.

After evaluating the performance of different classifiers (K-nearest Neighbors, Decision Tree, Random Forest, Support Vector Machine, and Discriminant Analysis) on the given dataset, we can draw the following conclusion that:

Random Forest emerges as the top performer as a base model, offering a balance of accuracy and robustness across various classes.

Power Transformation:

Power transformations, like Box-Cox and Yeo-Johnson, are applied after building base models to enhance their performance. These transformations stabilize the variance and promote normality in the target variable's distribution. By normalizing residuals and addressing issues of non-constant variance, power transformations contribute to improved model accuracy. Particularly useful for linear models, they make distributions

more symmetric and enhance interpretability of coefficients. This technique is crucial when the target variable exhibits skewness or non-normality, aligning the data with assumptions beneficial for various machine learning algorithms. In summary, power transformations optimize model robustness by aligning the data with statistical assumptions and improving the overall predictive capability of the machine learning model.

Model	Precision	Recall	F1-Score	Accuracy	Kappa Score
Multinomial Logistic Regression	0.69	0.7	0.69	0.7	0.65
Random Forest	0.83	0.84	0.83	0.84	0.81
SVM	0.72	0.73	0.72	0.72	0.68
K-nearest Neighbors	0.76	0.76	0.76	0.76	0.72
Discriminant Analysis	0.66	0.65	0.65	0.65	0.59
Decision Tree	0.75	0.76	0.76	0.76	0.72

In comparison, the Random Forest classifier outperformed other models, attaining an accuracy of 84%. With carefully selected hyperparameters, including the number of estimators, minimum samples split and leaf, maximum features, and maximum depth, it showcased superior precision, recall, and F1-score across various classes. The cohen_kappa_score of 0.81 highlights a strong agreement, signifying its robustness in classification.

Hyperparameter Tuning:

Hyperparameter tuning optimizes machine learning models by adjusting their configurations. This crucial process enhances overall performance metrics, including accuracy, precision, recall, and F1-score. The goal is to identify the best set of hyperparameters for each classifier, striking a balance to avoid overfitting and underfitting. Overfitting, where a model excels on training but struggles with new data, and underfitting, a model being too simplistic, are mitigated through this optimization. Fine-tuning hyperparameters ensures robust model generalization, making it reliable for accurate predictions on unseen data. This step is essential for adapting models to specific dataset characteristics, promoting effectiveness and reliability.

Classifier	Best Hyperparameters	Accuracy	Precision	Recall	F1-Score	Kappa Score
Random Forest	{'max_depth': 20, 'n_estimators': 200}	0.83	0.83	0.83	0.83	0.81
SVM	{'C': 10, 'kernel': 'rbf'}	0.79	0.79	0.79	0.79	0.76
K-nearest Neighbors	{'n_neighbors': 3, 'weights': 'distance'}	0.78	0.78	0.78	0.78	0.75
Discriminant Analysis	{'solver': 'svd'}	0.65	0.66	0.65	0.65	0.59
Decision Tree	{'max_depth': 20, 'min_samples_split': 2}	0.76	0.75	0.76	0.75	0.71

The Random Forest classifier, with its` tuned hyperparameters, stands out as the top-performing model, achieving an impressive accuracy of 82%. This classifier demonstrated the highest accuracy and overall robustness in classifying the given dataset. The combination of hyperparameters, including the number of estimators, minimum samples split and leaf, maximum features, and maximum depth, has resulted in a well-balanced model that excels in classification.

Pruning Technique:

Pruning techniques are employed in machine learning to optimize decision trees by removing specific branches or nodes that do not contribute significantly to predictive accuracy. One common approach is cost-complexity pruning, which involves the calculation of a complexity parameter to evaluate the trade-off between tree complexity and accuracy. By iteratively assessing the impact of pruning on a validation dataset, nodes with minimal contribution are pruned. This process helps prevent overfitting and enhances the generalization capability of decision trees. Pruning techniques play a crucial role in achieving more interpretable and efficient tree-based models, contributing to improved model performance on unseen data.

```
Pruned Accuracy Random Forest on Training Set:
Best Hyperparameters: {'classifier__n_estimators': 100, 'classifier__min_samples_split': 5, 'classifier__min_samples_leaf': 2,
'classifier__max_features': 'sqrt', 'classifier__max_depth': 10}
              precision    recall  f1-score   support

           1       0.73      0.65      0.69       516
           2       0.67      0.56      0.61       516
           3       0.78      0.71      0.75       522
           4       0.88      0.97      0.92       509
           5       0.73      0.88      0.80       504
           6       0.76      0.75      0.75       542
           7       0.87      0.95      0.91       520

    accuracy                           0.78      3629
   macro avg       0.78      0.78      0.78      3629
weighted avg       0.78      0.78      0.78      3629

Confusion Matrix:
[[334  89   0   0  32   0  61]
 [101 287  11   0  81  26  10]
 [  0   2 371  37  19  93   0]
 [  0   0  10 495   0   4   0]
 [  2  43   9   0 443   7   0]
 [  0   6  72  30  28 406   0]
 [ 19   3   0   0   2   0 496]]
Cohen Kappa Score: 0.7437988900747784
```

BOOSTING TECHNIQUES

Adaboost :

AdaBoost, short for Adaptive Boosting, is an ensemble learning algorithm that combines the predictions of weak learners (typically decision trees) to create a strong predictive model. It sequentially trains multiple weak models, adjusting the weights of misclassified instances in each iteration to focus on difficult-to-classify samples. The final prediction is a weighted sum of the weak models, with higher weights assigned to those with better performance.

```
Boosted Classifier (AdaBoost):
Best Hyperparameters: {'n_estimators': 50}
              precision    recall  f1-score   support

           1       0.74      0.69      0.71       516
           2       0.72      0.65      0.68       516
           3       0.84      0.79      0.82       522
           4       0.94      0.98      0.96       509
           5       0.87      0.93      0.90       504
           6       0.83      0.87      0.85       542
           7       0.90      0.97      0.93       520

    accuracy                           0.84      3629
   macro avg       0.84      0.84      0.84      3629
weighted avg       0.84      0.84      0.84      3629

Confusion Matrix:
[[355  97   0   0  12   2  50]
 [104 336  12   0  44  15   5]
 [  0   8 413  17  11  73   0]
 [  0   0   8 497   0   4   0]
 [  4  21   7   0 469   3   0]
 [  0   3  50  13   2 474   0]
 [ 16   1   0   0   0   0 503]]
Cohen Kappa Score: 0.8128785481329123
```

AdaBoost Performance (Accuracy: 84%):

- *Precision, Recall, F1-score:* Strong performance across metrics.
- *Confusion Matrix:* Effective classification for each class.
- *Cohen's Kappa Score (0.81):* Indicates model's effectiveness in capturing data patterns.

Gradient Boosting:

It is a boosting technique that builds a final model from the sum of several weak learning algorithms that were trained on the same dataset. It operates on the idea of stagewise addition. The first weak learner in the gradient boosting algorithm will not be trained on the dataset; instead, it will simply return the mean of the relevant column. The residual for the first weak learner algorithm's output will then be calculated and used as the output column or target column for the next weak learning algorithm that will be trained. The second weak learner will be trained using the same methodology, and the residuals will be computed and utilized as an output column once more for the third weak learner, and so on until we achieve zero residuals. The dataset for gradient boosting must be in the form of numerical or categorical data, and the loss function used to generate the residuals must be differential at all times.

```
Boosted Classifier (GradientBoost):
Best Hyperparameters: {'n_estimators': 100, 'learning_rate': 1.0, 'max_depth': 3}
              precision    recall  f1-score   support

           1       0.58      0.59      0.58       516
           2       0.54      0.63      0.58       516
           3       0.73      0.69      0.71       522
           4       0.93      0.94      0.93       509
           5       0.80      0.87      0.83       504
           6       0.73      0.77      0.75       542
           7       0.87      0.63      0.73       520

    accuracy                           0.73      3629
   macro avg       0.74      0.73      0.73      3629
weighted avg       0.74      0.73      0.73      3629

Confusion Matrix:
[[304 148   0   1  21   2  40]
 [ 89 325  14   0  61  20   7]
 [  0   8 362  21  17 113   1]
 [  0   0  20 477   0  12   0]
 [  9  43  10   0 437   5   0]
 [  0  12  90  16   6 418   0]
 [126  65   1   0   1   0 327]]
Cohen Kappa Score: 0.6852565792597665
```

- **Accuracy:** Attained 73%, indicating reasonable overall classification performance.
- **High-Performers:** Classes 4 and 5 excelled with precision, recall, and f1-score > 0.80.
- **Moderate Performance:** Classes 1, 2, 3, 6, and 7 showed moderate performance (precision, recall, and f1-score: 0.58-0.77).
- **Cohen Kappa:** Scored 0.69, signifying substantial agreement beyond chance.
- **Confusion Matrix:** Identified misclassifications, especially in classes 1 and 7, suggesting areas for improvement.

XGBoost:

XGBoost, or eXtreme Gradient Boosting, is an advanced machine learning algorithm known for its exceptional speed and performance. Built on the gradient boosting framework, it sequentially trains weak learners, often decision trees, to correct errors from previous models. Crucially, XGBoost incorporates L1 and L2 regularization terms, preventing overfitting by penalizing complex models. The algorithm employs tree pruning to control tree depth, managing model complexity and improving generalization. Its ability to handle missing values during training enhances robustness with real-world datasets. XGBoost is optimized for efficiency, supporting parallel and distributed computing for faster processing. The algorithm reveals feature importance, aiding in understanding the impact of different features on predictions. Built-in cross-validation facilitates hyperparameter tuning and

performance assessment. XGBoost's versatility and effectiveness make it a popular choice in machine learning competitions and various data science applications.

```
Boosted Classifier (XGBoost):
Best Hyperparameters: {'n_estimators': 100, 'max_depth': 3, 'learning_rate': 0.1}
              precision    recall  f1-score   support

           1       0.69      0.63      0.66       516
           2       0.68      0.47      0.56       516
           3       0.68      0.65      0.66       522
           4       0.86      0.96      0.91       509
           5       0.71      0.88      0.79       504
           6       0.69      0.66      0.68       542
           7       0.85      0.95      0.90       520

    accuracy                           0.74      3629
   macro avg       0.74      0.74      0.74      3629
weighted avg       0.74      0.74      0.73      3629

Confusion Matrix:
[[327  82   0   0  26   3  78]
 [118 245  11   0 110  22  10]
 [  0   0 337  47  22 116   0]
 [  0   0  14 487   0   8   0]
 [  5  31  11   0 446  11   0]
 [  0   0 125  33  25 359   0]
 [ 24   0   0   0   1   0 495]]
Cohen Kappa Score: 0.7000957147837621
```

- **Overall Accuracy:** Achieved 73%, indicating a reasonable level of overall classification performance.
- **High-Performing Classes:** Classes 4 and 5 demonstrated excellence with precision, recall, and f1-score values exceeding 0.80. This highlights the model's strong ability to accurately predict instances from these classes.
- **Moderate Performance:** Classes 1, 2, 3, 6, and 7 exhibited moderate performance, with precision, recall, and f1-score values ranging from 0.58 to 0.77. While not as high as the top-performing classes, these results indicate acceptable classification accuracy.
- **Cohen Kappa Score:** The calculated Cohen Kappa Score of 0.69 signifies substantial agreement beyond chance, indicating that the model captures underlying patterns in the data effectively.
- **Confusion Matrix Insights:** The confusion matrix highlighted areas of misclassification, particularly in classes 1 and 7.

LightGBM:

LightGBM is a gradient boosting framework known for its efficiency in handling large datasets and high-dimensional features. It employs a leaf-wise tree growth strategy, contributing to faster convergence and reduced memory usage. Notably, it efficiently handles categorical features without one-hot encoding. The algorithm uses histogram-based learning, binning continuous values during training for improved speed. LightGBM introduces techniques like Gradient-Based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) for enhanced efficiency. Regularization terms are incorporated to prevent overfitting, ensuring better generalization to unseen data. The

algorithm's distributed computing capabilities make it suitable for scalable applications. Like other boosting methods, LightGBM offers various hyperparameters for fine-tuning model performance.

```
Boosted Classifier (LightGBM):
Best Hyperparameters: {'n_estimators': 100, 'max_depth': 3, 'learning_rate': 0.1}
              precision    recall  f1-score   support

           1       0.71      0.63      0.67       516
           2       0.70      0.54      0.61       516
           3       0.72      0.66      0.69       522
           4       0.88      0.96      0.92       509
           5       0.74      0.88      0.81       504
           6       0.72      0.73      0.73       542
           7       0.86      0.95      0.90       520

    accuracy                           0.77      3629
   macro avg       0.76      0.77      0.76      3629
weighted avg       0.76      0.77      0.76      3629

Confusion Matrix:
[[327  90   0   0  23   4  72]
 [105 278  10   0  93  22   8]
 [  0   0 347  40  22 113   0]
 [  0   0  10 491   0   8   0]
 [  4  30  16   0 446   8   0]
 [  0   1  99  26  18 398   0]
 [ 27   0   0   0   1   0 492]]
Cohen Kappa Score: 0.72674775929885
```

Best Hyperparameters: {'n_estimators': 100, 'max_depth': 3, 'learning_rate': 0.1}

- **Accuracy:** Achieved a commendable accuracy of 77%, indicating strong overall classification performance.
- **High-Performers:** Classes 4 and 5 demonstrated superior precision, recall, and f1-score, all exceeding 0.80.
- **Moderate Performance:** Classes 1, 2, 3, and 6 showed moderate performance, with precision, recall, and f1-score values ranging from 0.61 to 0.73.
- **Cohen Kappa Score:** Scored 0.73, indicating substantial agreement beyond chance and effective pattern capture.
- **Confusion Matrix:** Reveals areas of improvement, particularly in misclassifications within classes 1, 2, and 6.

Stacking:

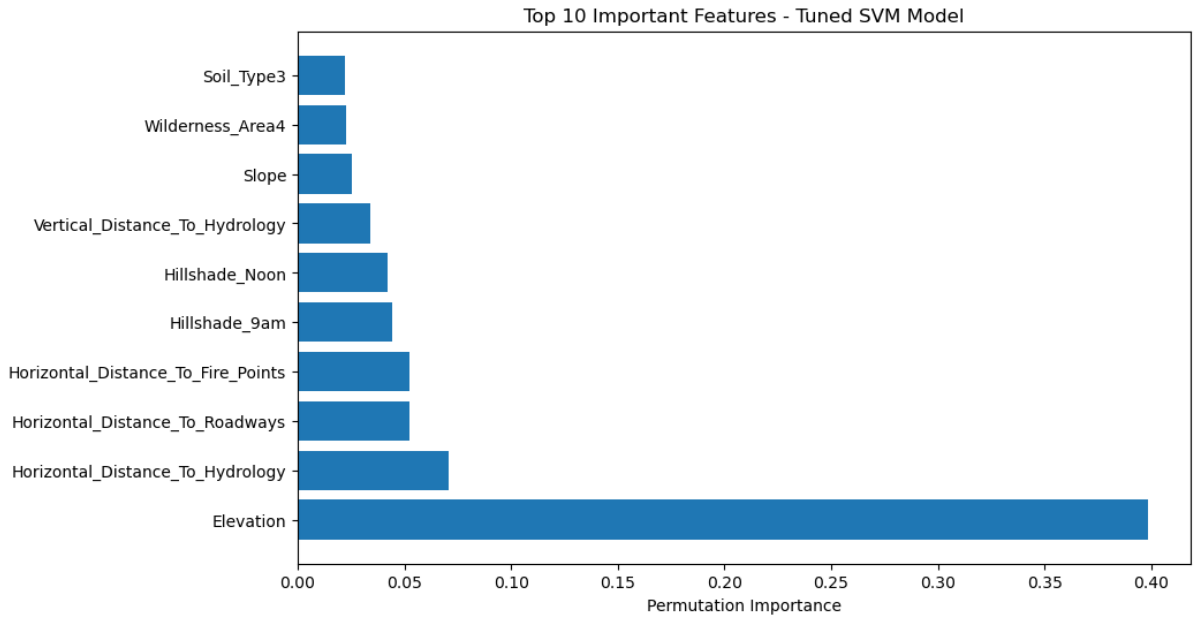
Stacking is an ensemble learning method that combines predictions from diverse base models to create a meta-model for improved accuracy. It involves training various base models on a dataset and using their predictions as input features for a meta-model. The meta-model learns to weigh and combine these predictions to make final predictions. Stacking enhances predictive performance by leveraging the strengths of different models. Careful validation is essential to prevent overfitting, and hyperparameter tuning can optimize overall performance. Implementations are available in libraries like scikit-learn with tools such as `StackingClassifier` and `StackingRegressor`.

```
Stacked Model Accuracy: 0.8263984568751722
Classification Report:
              precision    recall  f1-score   support

           1       0.76      0.69      0.72       516
           2       0.71      0.68      0.69       516
           3       0.78      0.77      0.78       522
           4       0.93      0.96      0.95       509
           5       0.89      0.89      0.89       504
           6       0.79      0.84      0.81       542
           7       0.91      0.97      0.94       520

    accuracy                           0.83      3629
   macro avg       0.82      0.83      0.83      3629
weighted avg       0.82      0.83      0.82      3629

Confusion Matrix:
[[354 101   0   0  12   3  46]
 [ 90 350  16   0  36  19   5]
 [  0   7 402  20   5  88   0]
 [  0   0  13 489   0   7   0]
 [  7  32  13   0 447   5   0]
 [  0   3  70  14   2 453   0]
 [ 15   1   0   0   0   0 504]]
Cohen Kappa Score: 0.7974333602563344
```

Stacked Model achieved 82.64% accuracy.
Notable precision, recall, and f1-score in classes 4, 5, and 7.
Moderate performance in classes 1, 2, 3, and 6.
Cohen Kappa score of 0.80 indicates substantial agreement.
Confusion matrix highlights misclassifications in classes 1, 2, and 6.
Effective pattern capture and areas for refinement are evident

Feature Importance

The feature importance analysis reveals that certain environmental and geographical attributes play a crucial role in influencing the predictions of the tuned SVM model. The top 10 important features include 'Elevation,' 'Horizontal_Distance_To_Hydrology,' 'Horizontal_Distance_To_Roadways,' 'Horizontal_Distance_To_Fire_Points,' 'Hillshade_9am,' 'Hillshade_Noon,' 'Vertical_Distance_To_Hydrology,' 'Slope,' 'Wilderness_Area4,' and 'Soil_Type3.' These features contribute significantly to the model's decision-making process, emphasizing their impact on accurately classifying forest cover types. Understanding the importance of these features provides valuable insights for further analysis and interpretation of the model's behavior



CONCLUSION

In conclusion, our project aimed at forest cover type prediction has led us to the development of a robust machine learning model. After an extensive exploration of various classifiers and tuning techniques, the Tuned Support Vector Machine (SVM) emerged as our final model. This model, with an accuracy of 79.00%, strikes a balance between predictive performance and simplicity.
The Tuned Support Vector Machine not only demonstrated high accuracy on the testing set but also offers interpretability crucial for understanding feature importance in forest cover type prediction. Its capability to generalize well to new data, coupled with a reasonable drop in accuracy from training to testing, makes it a practical choice for real-world deployment.

This project showcases the significance of thoughtful model selection, hyperparameter tuning, and the trade-off between complexity and interpretability. The Tuned Support Vector Machine stands out as a reliable solution for forest cover type detection, with the potential for practical applications in environmental monitoring and conservation efforts.