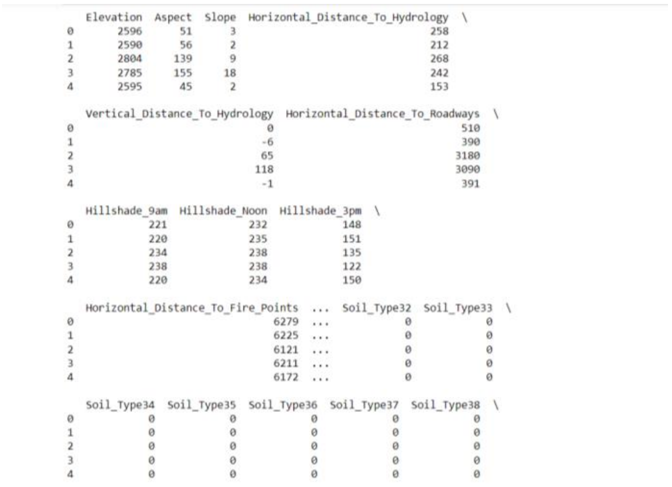
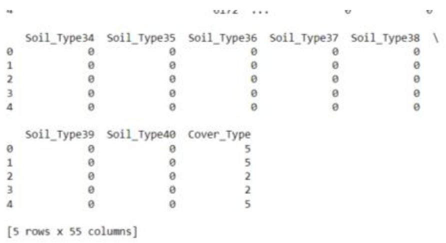
The initial step in designing any model is to understand the data, including its shape, values, and any categorical features. When we first encounter the data, it appears in the following way.



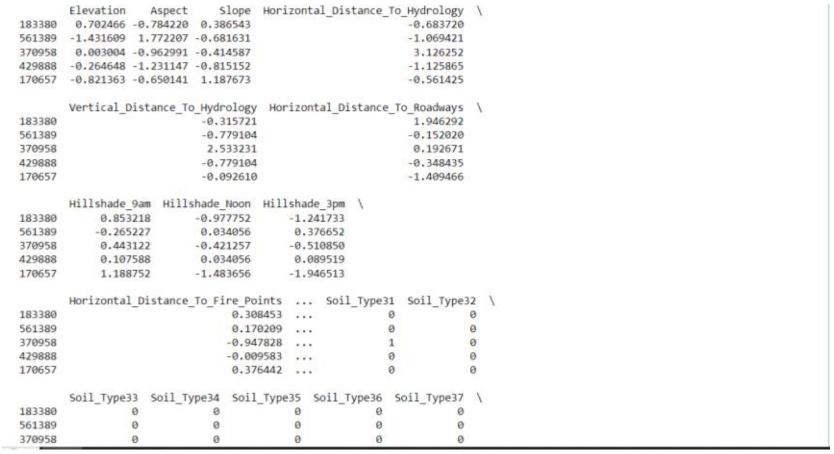
```
   Elevation  Aspect  Slope  Horizontal_Distance_To_Hydrology  \
0       2596      51      3                               258
1       2590      56      2                               212
2       2804     139      9                               268
3       2785     155     18                               242
4       2595      45      2                               153

   Vertical_Distance_To_Hydrology  Horizontal_Distance_To_Roadways  \
0                               0                              510
1                              -6                              390
2                              65                             3180
3                             118                             3090
4                              -1                              391

   Hillshade_9am  Hillshade_Noon  Hillshade_3pm  \
0            221             232            148
1            220             235            151
2            234             238            135
3            238             238            122
4            220             234            150

   Horizontal_Distance_To_Fire_Points  ...  Soil_Type32  Soil_Type33  \
0                                6279  ...            0            0
1                                6225  ...            0            0
2                                6121  ...            0            0
3                                6211  ...            0            0
4                                6172  ...            0            0

   Soil_Type34  Soil_Type35  Soil_Type36  Soil_Type37  Soil_Type38  \
0            0            0            0            0            0
1            0            0            0            0            0
2            0            0            0            0            0
3            0            0            0            0            0
4            0            0            0            0            0
```



```
   Soil_Type34  Soil_Type35  Soil_Type36  Soil_Type37  Soil_Type38  \
0            0            0            0            0            0
1            0            0            0            0            0
2            0            0            0            0            0
3            0            0            0            0            0
4            0            0            0            0            0

   Soil_Type39  Soil_Type40  Cover_Type
0            0            0           5
1            0            0           5
2            0            0           2
3            0            0           2
4            0            0           5

[5 rows x 55 columns]
```

Our dataset consists of 55 columns, representing 54 features for each item, with the last column being our target variable. It is apparent that our data does not contain any categorical features. However, it is worth noting that certain columns have significantly larger values compared to others, indicating the need for scaling in future steps.

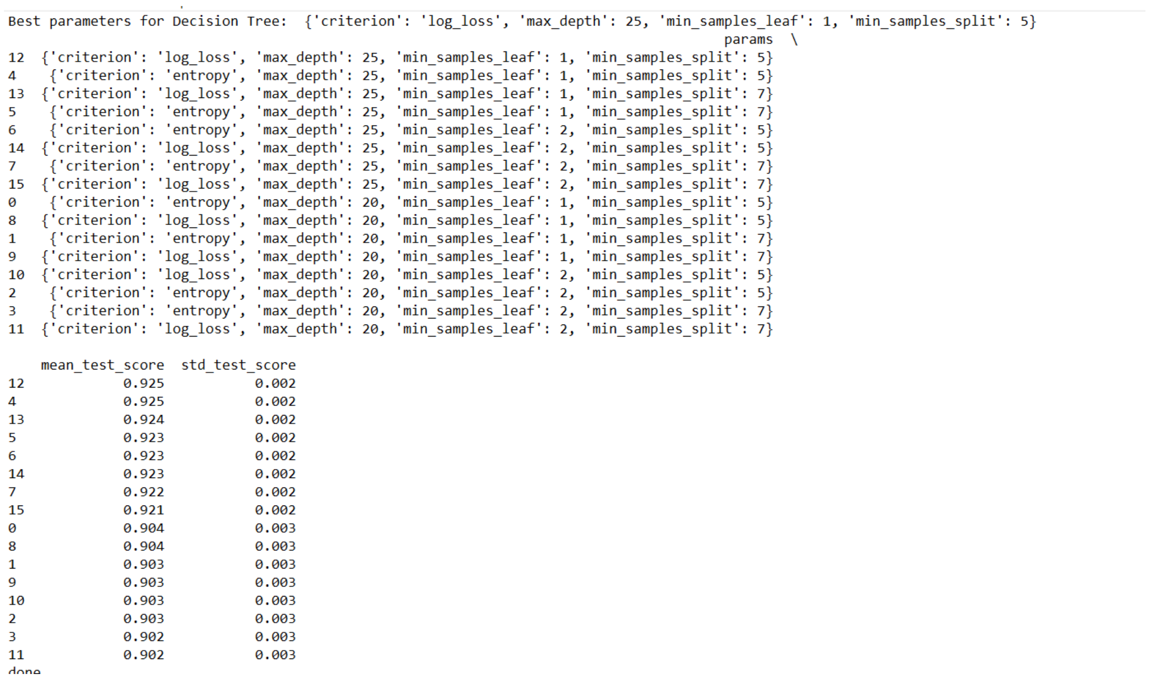
After dividing our data into train and test sets, we examined them for null and duplicate values, but found none. For preprocessing, we only applied scaling to our first column, which produced the following results.



```
        Elevation    Aspect     Slope  Horizontal_Distance_To_Hydrology  \
183380   0.702466 -0.784220  0.386543                         -0.683720
561389  -1.431609  1.772207 -0.681631                         -1.069421
370958   0.003004 -0.962991 -0.414587                          3.126252
429888  -0.264648 -1.231147 -0.815152                         -1.125865
170657  -0.821363 -0.650141  1.187673                         -0.561425

        Vertical_Distance_To_Hydrology  Horizontal_Distance_To_Roadways  \
183380                       -0.315721                         1.946292
561389                       -0.779104                        -0.152020
370958                        2.533231                         0.192671
429888                       -0.779104                        -0.348435
170657                       -0.092610                        -1.409466

        Hillshade_9am  Hillshade_Noon  Hillshade_3pm  \
183380       0.853218       -0.977752      -1.241733
561389      -0.265227        0.034056       0.376652
370958       0.443122       -0.421257      -0.510850
429888       0.107588        0.034056       0.089519
170657       1.188752       -1.483656      -1.946513

        Horizontal_Distance_To_Fire_Points  ...  Soil_Type31  Soil_Type32  \
183380                            0.308453  ...            0            0
561389                            0.170209  ...            0            0
370958                           -0.947828  ...            1            0
429888                           -0.009583  ...            0            0
170657                            0.376442  ...            0            0

        Soil_Type33  Soil_Type34  Soil_Type35  Soil_Type36  Soil_Type37  \
183380            0            0            0            0            0
561389            0            0            0            0            0
370958            0            0            0            0            0
```

The columns are now more uniform in their values, as can be observed. It's worth noting that the varying IDs are due to the shuffling that occurred during the data splitting process.

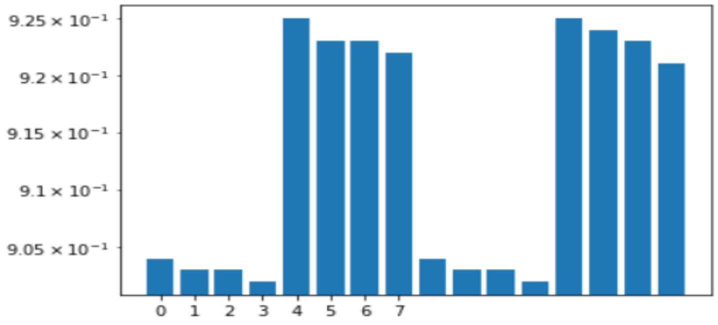
The different parameters we used for the decision tree models were 'max_depth': [20, 25], 'criterion': ['entropy', 'log_loss'], 'min_samples_split': [5, 7], and 'min_samples_leaf': [1, 2], resulting in a total of 16 models. After testing various combinations, we found that leaving the max depth at its default value resulted in a better model. Additionally, we discovered that the gini criterion typically performed worse for this particular dataset, so we excluded it from our analysis. The results of our analysis are presented below.



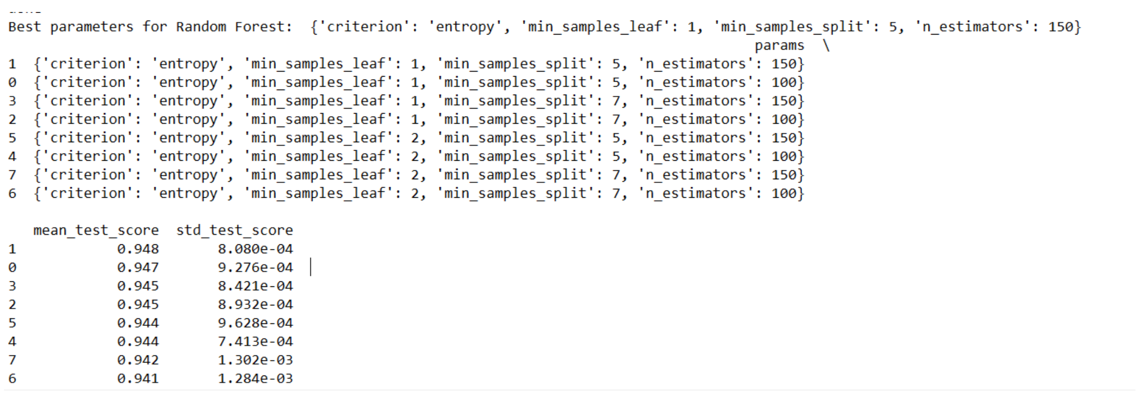
```
Best parameters for Decision Tree:  {'criterion': 'log_loss', 'max_depth': 25, 'min_samples_leaf': 1, 'min_samples_split': 5}
                                                                                   params  \
12  {'criterion': 'log_loss', 'max_depth': 25, 'min_samples_leaf': 1, 'min_samples_split': 5}
4    {'criterion': 'entropy', 'max_depth': 25, 'min_samples_leaf': 1, 'min_samples_split': 5}
13  {'criterion': 'log_loss', 'max_depth': 25, 'min_samples_leaf': 1, 'min_samples_split': 7}
5    {'criterion': 'entropy', 'max_depth': 25, 'min_samples_leaf': 1, 'min_samples_split': 7}
6    {'criterion': 'entropy', 'max_depth': 25, 'min_samples_leaf': 2, 'min_samples_split': 5}
14  {'criterion': 'log_loss', 'max_depth': 25, 'min_samples_leaf': 2, 'min_samples_split': 5}
7    {'criterion': 'entropy', 'max_depth': 25, 'min_samples_leaf': 2, 'min_samples_split': 7}
15  {'criterion': 'log_loss', 'max_depth': 25, 'min_samples_leaf': 2, 'min_samples_split': 7}
0    {'criterion': 'entropy', 'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 5}
8   {'criterion': 'log_loss', 'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 5}
1    {'criterion': 'entropy', 'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 7}
9   {'criterion': 'log_loss', 'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 7}
10  {'criterion': 'log_loss', 'max_depth': 20, 'min_samples_leaf': 2, 'min_samples_split': 5}
2    {'criterion': 'entropy', 'max_depth': 20, 'min_samples_leaf': 2, 'min_samples_split': 5}
3    {'criterion': 'entropy', 'max_depth': 20, 'min_samples_leaf': 2, 'min_samples_split': 7}
11  {'criterion': 'log_loss', 'max_depth': 20, 'min_samples_leaf': 2, 'min_samples_split': 7}

    mean_test_score  std_test_score
12            0.925           0.002
4             0.925           0.002
13            0.924           0.002
5             0.923           0.002
6             0.923           0.002
14            0.923           0.002
7             0.922           0.002
15            0.921           0.002
0             0.904           0.003
8             0.904           0.003
1             0.903           0.003
9             0.903           0.003
10            0.903           0.003
2             0.903           0.003
3             0.902           0.003
11            0.902           0.003
done
```

The optimal parameter set for decision tree is {'criterion': 'log_loss', 'min_samples_leaf': 1, 'min_samples_split': 5}. The result implies that log_loss and entropy typically perform similarly, so we used log_loss for the parameters in our random forest.

The parameter range we considered for random forest was {'n_estimators': [100, 150], 'criterion': ['entropy'], 'min_samples_split': [5, 7], 'min_samples_leaf': [1, 2]}.



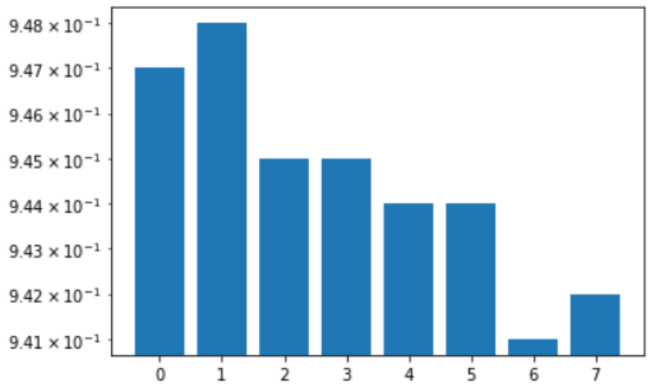
Which resulted:



```
Best parameters for Random Forest:  {'criterion': 'entropy', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 150}
                                                                                              params  \
1  {'criterion': 'entropy', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 150}
0  {'criterion': 'entropy', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 100}
3  {'criterion': 'entropy', 'min_samples_leaf': 1, 'min_samples_split': 7, 'n_estimators': 150}
2  {'criterion': 'entropy', 'min_samples_leaf': 1, 'min_samples_split': 7, 'n_estimators': 100}
5  {'criterion': 'entropy', 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 150}
4  {'criterion': 'entropy', 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 100}
7  {'criterion': 'entropy', 'min_samples_leaf': 2, 'min_samples_split': 7, 'n_estimators': 150}
6  {'criterion': 'entropy', 'min_samples_leaf': 2, 'min_samples_split': 7, 'n_estimators': 100}

   mean_test_score  std_test_score
1            0.948       8.080e-04
0            0.947       9.276e-04
3            0.945       8.421e-04
2            0.945       8.932e-04
5            0.944       9.628e-04
4            0.944       7.413e-04
7            0.942       1.302e-03
6            0.941       1.284e-03
```

The accuracy obtained by the random forest model was better, which is in line with what we were hoping to achieve.

The optimal set of parameters for this model were {'criterion': 'entropy', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 150}.

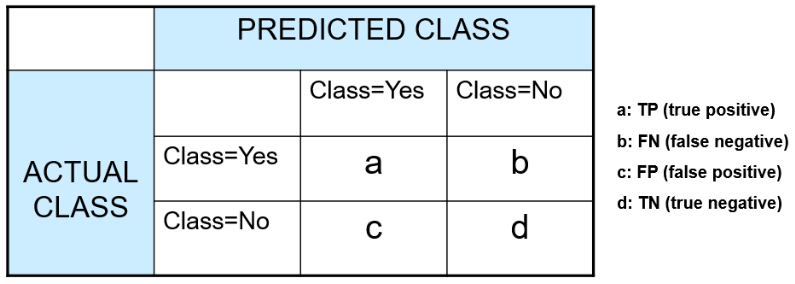


The X shows the combination of parameters used, and the values are represented in logarithmic form to highlight the differences.

After fitting our model, it is time to evaluate it using different metrics. Let's begin by describing the metrics we will be discussing:

- Accuracy: Accuracy measures the overall correctness of the model's predictions. It is calculated as the number of correct predictions divided by the total number of predictions.
- Precision: Precision measures how many of the predicted positive cases were actually positive. It is calculated as the number of true positives divided by the sum of true positives and false positives.
- Recall: Recall measures how many of the actual positive cases were correctly predicted as positive by the model. It is calculated as the number of true positives divided by the sum of true positives and false negatives.
- F1 score: F1 score is a weighted average of precision and recall, and it balances between the two metrics. It is calculated as 2 times the product of precision and recall, divided by the sum of precision and recall.

Confusion matrix: which in our case is 7*7 because it's not a binary classification.

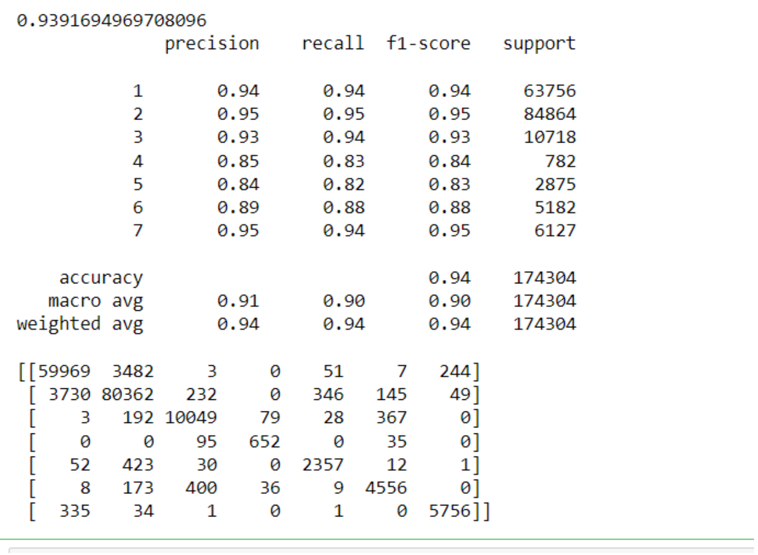


		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)
b: FN (false negative)
c: FP (false positive)
d: TN (true negative)

Now we should analyze this metrices for our predicted values.

Decision Tree:

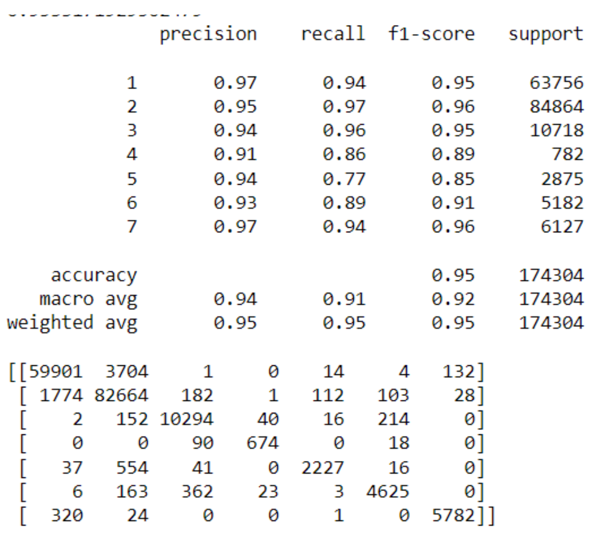


```
0.9391694969708096
              precision    recall  f1-score   support

           1       0.94      0.94      0.94     63756
           2       0.95      0.95      0.95     84864
           3       0.93      0.94      0.93     10718
           4       0.85      0.83      0.84       782
           5       0.84      0.82      0.83      2875
           6       0.89      0.88      0.88      5182
           7       0.95      0.94      0.95      6127

    accuracy                           0.94    174304
   macro avg       0.91      0.90      0.90    174304
weighted avg       0.94      0.94      0.94    174304

[[59969  3482     3     0    51     7   244]
 [ 3730 80362   232     0   346   145    49]
 [    3   192 10049    79    28   367     0]
 [    0     0    95   652     0    35     0]
 [   52   423    30     0  2357    12     1]
 [    8   173   400    36     9  4556     0]
 [  335    34     1     0     1     0  5756]]
```

As you can see, our accuracy is 94%. For the other metrics, let's consider the 4th class. Its precision is 85%, which means 85% of the items we predicted as the 4th class truly belong to the 4th class. Its recall is 83%, which means 83% of the 4th class items were predicted correctly. Finally, the F1 measure can be computed as (2 x 0.85 x 0.83)/(0.85 + 0.83).

Other classes can be computed in the same way using the confusion matrix. All the data on the diagonal are the ones that were predicted correctly, and precision is calculated as the data on the diagonal divided by the sum of its column. Recall is also computed by dividing the desired diagonal data by its row.

Random Forest:



```
[illegible]
              precision    recall  f1-score   support

           1       0.97      0.94      0.95     63756
           2       0.95      0.97      0.96     84864
           3       0.94      0.96      0.95     10718
           4       0.91      0.86      0.89       782
           5       0.94      0.77      0.85      2875
           6       0.93      0.89      0.91      5182
           7       0.97      0.94      0.96      6127

    accuracy                           0.95    174304
   macro avg       0.94      0.91      0.92    174304
weighted avg       0.95      0.95      0.95    174304

[[59901  3704     1     0    14     4   132]
 [ 1774 82664   182     1   112   103    28]
 [    2   152 10294    40    16   214     0]
 [    0     0    90   674     0    18     0]
 [   37   554    41     0  2227    16     0]
 [    6   163   362    23     3  4625     0]
 [  320    24     0     0     1     0  5782]]
```

The accuracy of our model is 0.95%. For the precision of the 4th class, we found that 0.91% of the items we predicted as the 4th class are actually a part of the 4th class. The recall of the 4th class is 0.96, meaning that 0.96% of the 4th class items were predicted correctly. To calculate the F1 score, we can use the formula (2 x 0.91 x 0.96) / (0.96 + 0.91). We can use the confusion matrix to calculate these metrics for other classes as well. The diagonal entries in the matrix represent the items that were predicted correctly, and precision is calculated by dividing the diagonal entry by the sum of its column. Recall is calculated by dividing the diagonal entry by the sum of its row.