Analytics Early Talent Case Study - Interview

Data Source: https://archive.ics.uci.edu/ml/datasets/Covertype

Introduction

In this classic dataset, over 50 attributes are made available. The goal of this dataset is to help predict the Cover Type. In other words, we want to know what type of tree grows in an area based on the attributes provided. Your tasks are to complete an exploratory data analysis, develop and execute a modeling strategy, and provide results & conclusions. Please note that this case study is not designed to have you develop the best performing model. Rather, the key focus is on understanding the approach.

Exploratory Data Analysis (EDA)

1. **Describe the dataset and any interesting variables (feel free to include response).**

    Hint: It may be useful to provide a high-level summary (i.e. descriptive statistics) on the variables

    The dataset is heterogeneous (binary and continuous variables)

    The dataset is high-dimensional with high number of samples

    The problem is a 7-way (supervised) classification task (baseline = 14.29% accuracy)

    6 general quantitative (elevation, aspect, slope, hori/vert distances to water features, hori distance to roadways, hori distance to fire points), 3 quantitative features on hillshade index at 3 different times of the day (9am, noon, 3pm), 4 wilderness areas, 40 soil types (non-independent), 7 cover types (discrete response)

    54 predictors and 1 response

2. **Provide up to 5 key insights that you learned from this dataset**

    Hint: Do not focus on statistical summaries. Consider focusing on groups and clusters and describing them

    A lot of the variables were Normally distribution, but many were uniformly distributed as well.

    vertical_distance_to_hydrology, horizontal_distance_to_fire_points, slope, horizontal_distance_to_hydrology, hillshade_9am (all from random forest), and wilderness_area_3 (from decision tree) are the most important features of this dataset.

3. **Highlight challenges in the dataset and the plans to mitigate those challenges**

   Hint: If there are missing data, how would you address this?

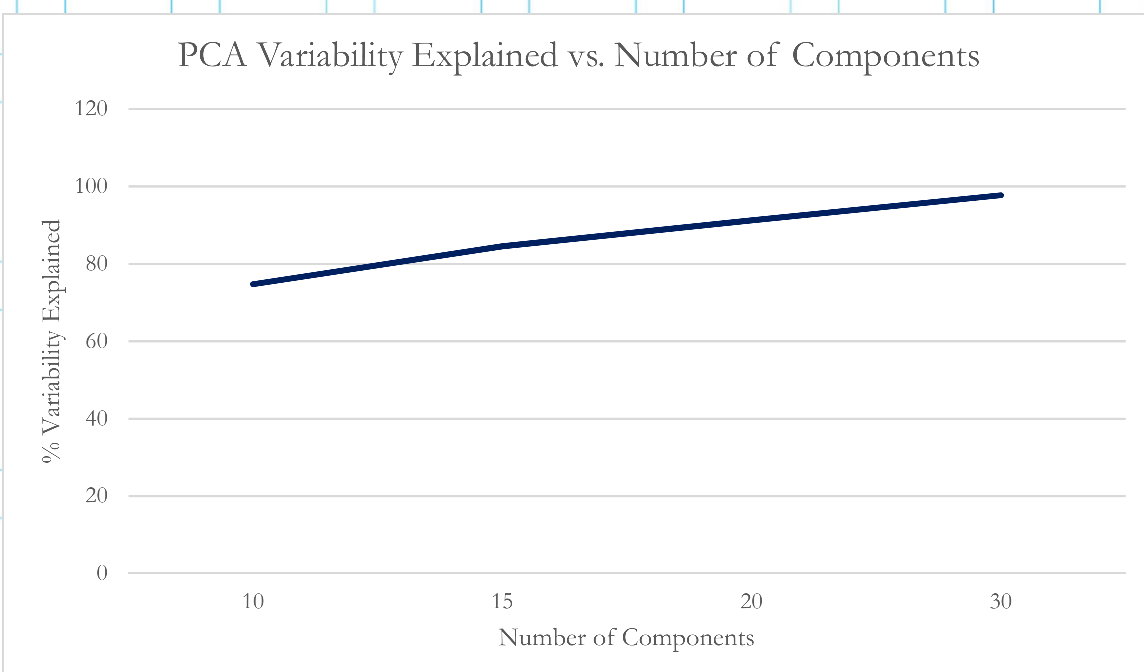
   There were no missing values or weird values in the dataset.

   The class imbalance was pretty bad. So, we take the smallest class size and random sample from the larger classes so they all have the same number of samples.

   The data isn't normalized. We need to prevent the network from learning bias towards specific features, so we have to normalize the features with respect to their distribution.

Modeling Strategy

1. What model(s) are you using and why?
   a. Normalization of all features (first used minmax, then used gaussian)
   b. PCA analysis for dimensionality reduction
      i. Before class balancing:
      ii. N_components = 30 => 97.71% variability
      iii. N_components = 20 => 91.21% variability
      iv. N_components = 15 => 84.57% variability
      v. N_components = 10 => 74.75% variability



      vi. After class balancing:
      vii. N_components = 30 => 97.29% variability
      viii. N_components = 20 => 89.04% variability
      ix. N_components = 15 => 81.38% variability

x. N_components = 10 => 70.13% variability
xi. After switching to gaussian normalization:
xii. N_components = 30 => 92.60% variability
xiii. N_components = 20 => 83.95% variability
xiv. N_components = 15 => 75.73% variability
xv. N_components = 10 => 62.50% variability

c. Lasso for the feature selection, compared to Ridge regression
d. SVM due to its strength in previous works, also helps with inference
e. Logistic regression due to its inference abilities
f. Decision tree due to binary nature of some datapoints
g. Random forests to produce low variance models of decision trees
h. DBScan to find outliers

2. Describe (in detail) how each model was developed

Results

1. Describe your model results

Hint: It may be helpful to include tables or graphs

Training DBScan with eps = 0.2 and min_samples = 3...

Epsilon: 0.2, Min samples: 3, Num outliers: 9909, Score: -0.5594504635496482

Training DBScan with eps = 0.2 and min_samples = 5...

Epsilon: 0.2, Min samples: 5, Num outliers: 11995, Score: -0.6832048304002939

Training DBScan with eps = 0.2 and min_samples = 10...

Epsilon: 0.2, Min samples: 10, Num outliers: 13223, Score: -0.4304529893288533

Training DBScan with eps = 0.5 and min_samples = 3...

Epsilon: 0.5, Min samples: 3, Num outliers: 1606, Score: -0.1326926021099541

Training DBScan with eps = 0.5 and min_samples = 5...

Epsilon: 0.5, Min samples: 5, Num outliers: 2454, Score: -0.04738586686913065

Training DBScan with eps = 0.5 and min_samples = 10...

Epsilon: 0.5, Min samples: 10, Num outliers: 4223, Score: -0.03814675306936041

Training DBScan with eps = 1.0 and min_samples = 3...

Epsilon: 1.0, Min samples: 3, Num outliers: 110, Score: 0.29439934959530417

Training DBScan with eps = 1.0 and min_samples = 5...

Epsilon: 1.0, Min samples: 5, Num outliers: 212, Score: 0.3038236139340124

Training DBScan with eps = 1.0 and min_samples = 10...

Epsilon: 1.0, Min samples: 10, Num outliers: 558, Score: 0.2999014259663512

Conclusion: Not a ton of outliers

Used silhouette score as scoring metric (ranges from -1 to +1, where values near 0 indicate overlapping clusters, -1 means wrong cluster assignments, and 1 is perfect separation of clusters)

Train on 0.5% of 581,012 samples (2905 samples)

Linear kernels

One-vs-rest classification scheme

Using 20 PCA components

Uniform scaling

Before class balancing:

SVM (C = 0.1) = 0.6229

[9015 9707 1377 133 478 842 1037]

SVM (C = 0.1) = 0.6399

[8501 9046 1175 133 478 842 928]

SVM (C = 0.5) = 0.6410

[8486 9112 1197 131 478 842 856]

SVM (C = 1.0) = 0.6410

[8470 9126 1192 129 478 842 827]

SVM (C = 2.0) = 0.6100

[8487 9215 1210 128 478 842 820]

With dimensionality reduction, still training on 0.5% of data, accuracies drop by about 4% on all models

After class balancing:

Linear Kernel

Using 0.5% of data for training

SVM (C = 0.1) = 0.1425

SVM (C = 0.1) = 0.4048

SVM (C = 0.5) = 0.5291

SVM (C = 1.0) = 0.5373

Using 50% of data for training

SVM (C = 0.1) = 0.5572

SVM (C = 0.1) = 0.5688

SVM (C = 0.5) = 0.5759

SVM (C = 1.0) = 0.5762

Using 70% of data for training

SVM (C = 0.1) = 0.5517

SVM (C = 0.1) = 0.5654

SVM (C = 0.5) = 0.5720

SVM (C = 1.0) = 0.5725

Using 70% of data for training (ovr instead of ovo classification)

SVM (C = 0.1) = 0.5623

SVM (C = 0.1) = 0.5814

SVM (C = 0.5) = 0.5900

SVM (C = 1.0) = 0.5935

RBF Kernel

ovr classification

SVM (C = 0.1) = 0.3956

SVM (C = 0.1) = 0.5542

SVM (C = 0.5) = 0.5679

SVM (C = 1.0) = 0.5706

ovo classification

SVM (C = 0.1) = 0.4082

SVM (C = 0.1) = 0.5614

SVM (C = 0.5) = 0.5783

SVM (C = 1.0) = 0.5831

Polynomial Kernel (degree = 3)

SVM (C = 0.1) = 0.1375

SVM (C = 0.1) = 0.1375

SVM (C = 0.5) = 0.1375

SVM (C = 1.0) = 0.1375

ovo classification

SVM (C = 0.1) = 0.1392

SVM (C = 0.1) = 0.1392

SVM (C = 0.5) = 0.1392

SVM (C = 1.0) = 0.1392

We will continue with a linear kernel:

ovr classification

Gaussian scaling

SVM (C = 0.1) = 0.5781

SVM (C = 0.1) = 0.5828

SVM (C = 0.5) = 0.5864

SVM (C = 1.0) = 0.5946

Decision tree score (5-fold CV): 65.96%

Random forest score: 69.79%

Logistic regression (L1 regularization): 59.68%

Logistic regression (L2 regularization): 58.71%

Analyzed all coefficients of logistic regression, random forest, and decision tree models.

I found all principal components are pretty useful for the project, none of them are "more important" or have a higher coefficient than the other, for the most part.

Random forest importances of features:

[0.08410482 0.07925606 0.05358839 0.0548627 0.04280521 0.03941313 0.04313247 0.04117022 0.04581392 0.03151285 0.02887122 0.02878627 0.06509489 0.03706813 0.047941
0.051307 0.05468086 0.0890564 0.04454073 0.03699371]

Then switched to using all 53 features:

Decision tree score (5-fold CV): 76.91%

Random forest score: 79.58%

Logistic regression (L1 regularization): 64.06%

Logistic regression (L2 regularization): 63.82%

Decision tree importances of features:

[0.04607704 0.02957654 0.07665138 0.0476782 0.13072399
0.07199059 0.04267652 0.03540508 0.12260777 0.00424185
0.00069534 0.0104947 0.11041007 0.00128324 0.01508742
0.02114243 0.0187542 0.0013094 0.00327764 0. 0. 0.
0.0246174 0.00536415 0.0069637 0.00353211 0.000482 0.
0.00170454 0.00937221 0.00326218 0.00016469 0.00081057
0.00015855 0.004709 0.00579501 0.00268675 0. 0.00183433
0.00053593 0.00035953 0.00682012 0.01335041 0.00258122
0.00427237 0.00265049 0.00050071 0.00832232 0. 0.0014669
0.03549884 0.03572728 0.02637332]

Random forest importances of features:

[5.95545201e-02 5.14750378e-02 8.42776220e-02 6.27941605e-02
1.30460529e-01 6.03394109e-02 5.40961502e-02 5.93826923e-02
9.99611566e-02 2.71591485e-02 5.40864429e-03 2.33267559e-02
4.83928874e-02 2.21526632e-03 1.15220754e-02 2.55244186e-02
1.53659934e-02 9.57661120e-04 4.30932538e-03 3.50305755e-07
3.53643144e-05 8.66064435e-05 2.57969312e-02 5.57970369e-03
5.71382869e-03 8.03090532e-03 7.74375189e-04 9.24805073e-06
1.19101615e-03 6.37213977e-03 1.22254076e-03 7.35430238e-04
1.82115337e-03 3.90860739e-04 8.20948104e-03 7.10097252e-03
2.92214549e-03 1.27142598e-05 1.09652844e-03 1.89528940e-04
1.31690298e-04 8.03623413e-03 1.04023751e-02 3.76175746e-03
5.00791446e-03 5.50516634e-03 3.17568812e-04 3.34171388e-03
1.27052418e-04 9.20002609e-04 2.35699435e-02 2.26149645e-02
1.24483354e-02]

2. (OPTIONAL) If more than 1 model was developed, please explain which model should be chosen and why.
    a. Random forests due to low variance, high accuracy, and possibility of inference.
3. If more time were provided, what other strategies would you pursue? Why?
    a. Honestly, I would try Neural Networks and Decision Trees. This problem is very clearly strongly supervised, which would make it perfect for a NN. I also would’ve done more hyperparameter tuning on the random forests.

Supplemental

['elevation', 'aspect', 'slope', 'horizontal_distance_to_hydrology',
'vertical_distance_to_hydrology', 'hillshade_9am', 'hillshade_noon',
'hillshade_3pm', 'horizontal_distance_to_fire_points',
'wilderness_area_0', 'wilderness_area_1', 'wilderness_area_2',
'wilderness_area_3', 'soil_type_0', 'soil_type_1', 'soil_type_2',
'soil_type_3', 'soil_type_4', 'soil_type_5', 'soil_type_6',
'soil_type_7', 'soil_type_8', 'soil_type_9', 'soil_type_10',
'soil_type_11', 'soil_type_12', 'soil_type_13', 'soil_type_14',
'soil_type_15', 'soil_type_16', 'soil_type_17', 'soil_type_18',
'soil_type_19', 'soil_type_20', 'soil_type_21', 'soil_type_22',
'soil_type_23', 'soil_type_24', 'soil_type_25', 'soil_type_26',
'soil_type_27', 'soil_type_28', 'soil_type_29', 'soil_type_30',
'soil_type_31', 'soil_type_32', 'soil_type_33', 'soil_type_34',
'soil_type_35', 'soil_type_36', 'soil_type_37', 'soil_type_38',
'soil_type_39', 'cover_type']

Notes

Used 70/30 train/test split because some variables don't have a ton of positive samples

Would've used 60/40 split otherwise due to low dimensionality with high number of samples

Shuffled data before train/test split

No validation set was used

"Comparative Accuracies of Artificial Neural Networks and Discriminant Analysis in Predicting Forest Cover Types from Cartographic Variables" (Denis Joseph Dean) was the paper from 2000

- Achieved 70.52% accuracy with ANN
- Achieved 58.38% accuracy with LDA
- Mean classification accuracy was the evaluation metric

Normally distributed

Aspect

Cover_type

Hillshade_3pm

Hillshade_9am

Hillshade_noon

Horizontal_distance_to_fire_points
Horizontal_distance_to_hydrology

Uniformly distributed

Elevation
Vertical_distance_to_hydrology
Wilderness_area_0
Wilderness_area_2

Other distribution

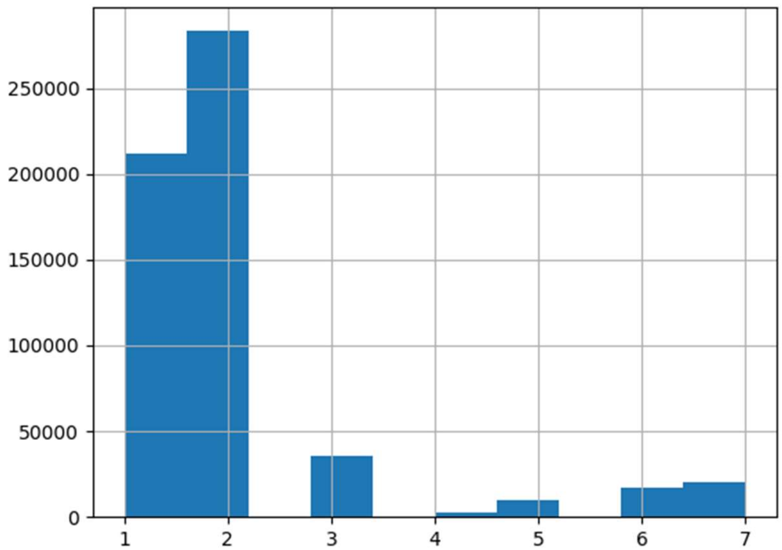
Slope
Soil_type_28

Nothing special

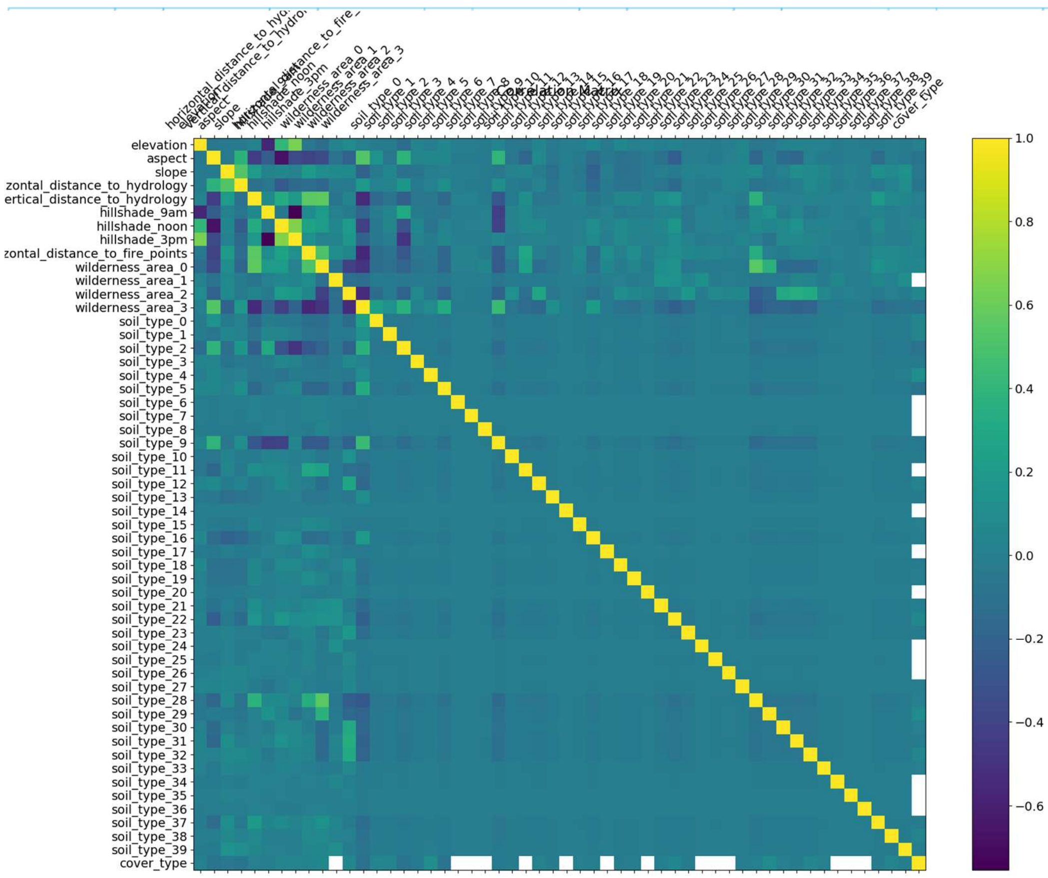
Soil_type_X for all X from 0 to 39, except for 28
Wilderness_area_1
Wilderness_area_3

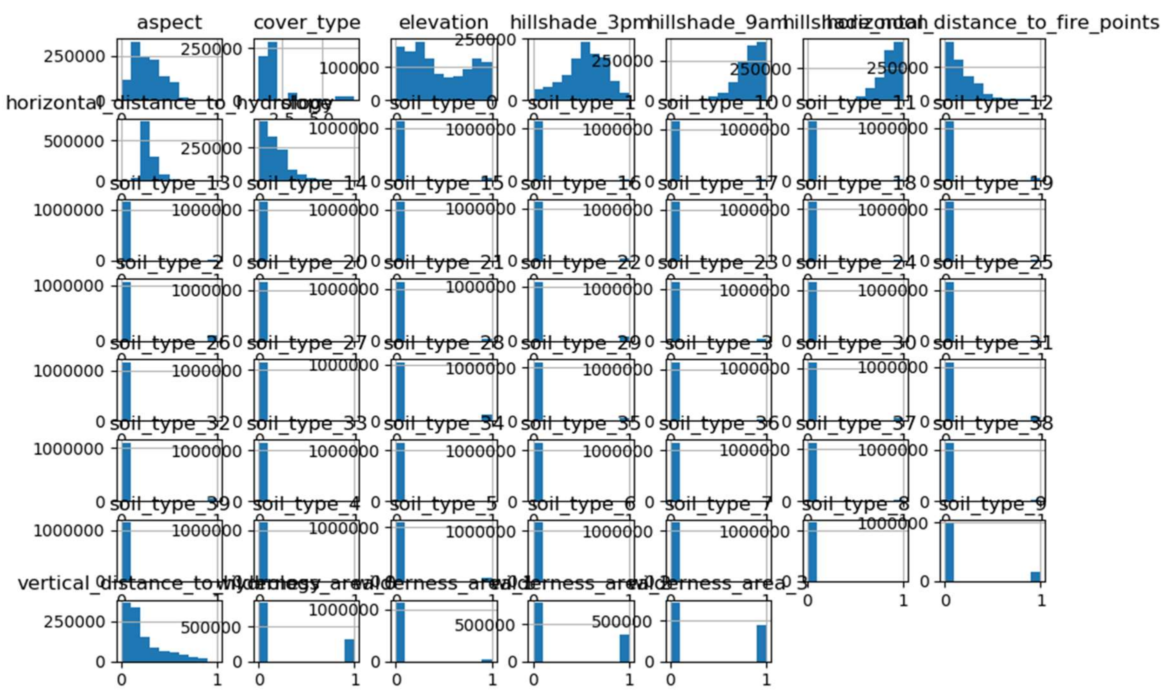
Technologies used

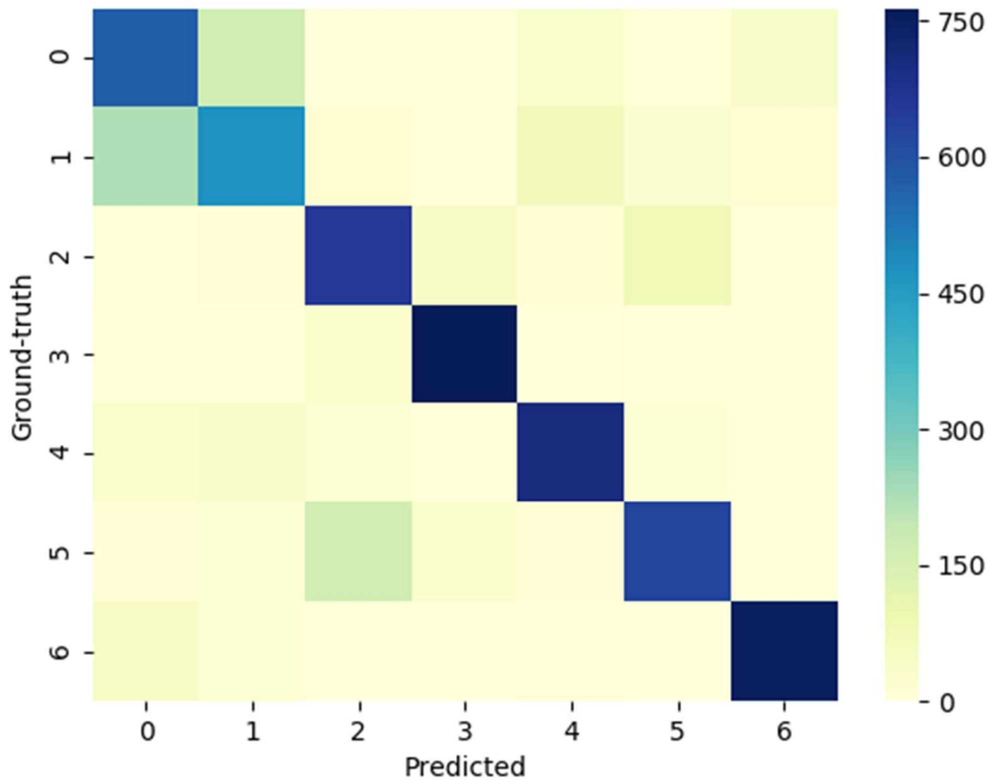
Python3
Numpy
Pandas
Sklearn
Matplotlib
Seaborn



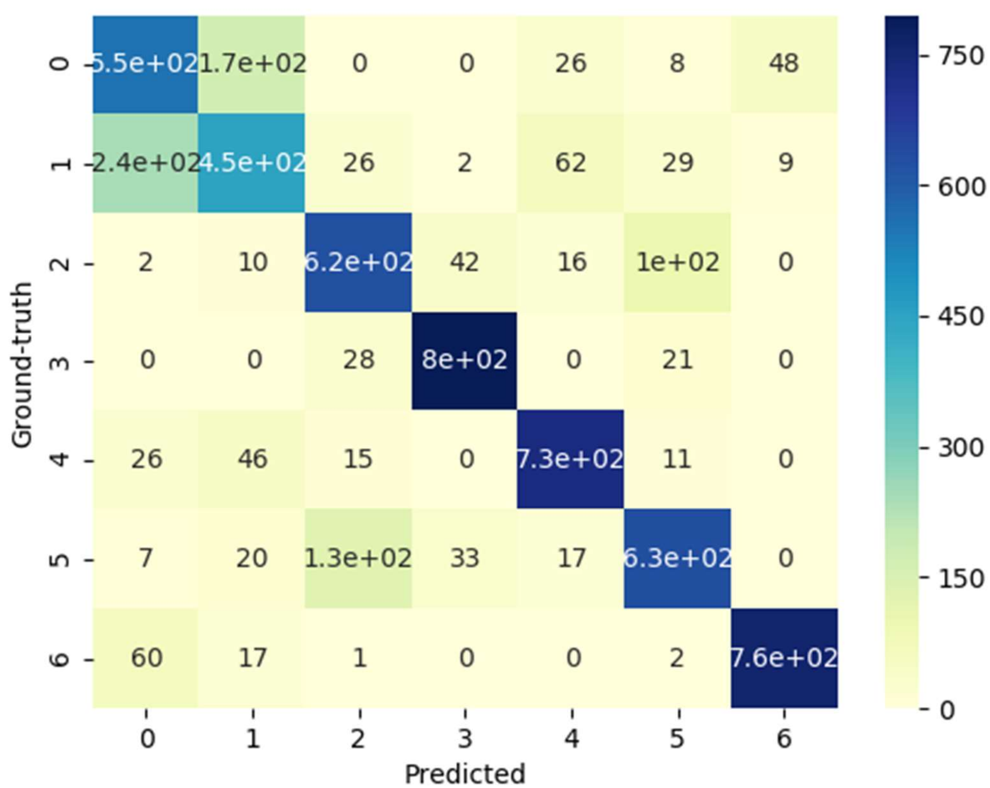
<= **Class Distrib.**


Correlation Matrix
elevation
aspect
slope
zontal_distance_to_hydrology
ertical_distance_to_hydrology
hillshade_9am
hillshade_noon
hillshade_3pm
zontal_distance_to_fire_points
wilderness_area_0
wilderness_area_1
wilderness_area_2
wilderness_area_3
soil_type_0
soil_type_1
soil_type_2
soil_type_3
soil_type_4
soil_type_5
soil_type_6
soil_type_7
soil_type_8
soil_type_9
soil_type_10
soil_type_11
soil_type_12
soil_type_13
soil_type_14
soil_type_15
soil_type_16
soil_type_17
soil_type_18
soil_type_19
soil_type_20
soil_type_21
soil_type_22
soil_type_23
soil_type_24
soil_type_25
soil_type_26
soil_type_27
soil_type_28
soil_type_29
soil_type_30
soil_type_31
soil_type_32
soil_type_33
soil_type_34
soil_type_35
soil_type_36
soil_type_37
soil_type_38
soil_type_39
cover_type
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6


aspect
cover_type
elevation
hillshade_3pm
soil_type_0
soil_type_1
soil_type_10
soil_type_11
soil_type_12
wilderness_area_3



Ground-truth
Predicted
0
1
2
3
4
5
6
750
600
450
300
150
0



Ground-truth
Predicted
5.5e+02 1.7e+02 0 0 26 8 48
2.4e+02 4.5e+02 26 2 62 29 9
2 10 6.2e+02 42 16 1e+02 0
0 0 28 8e+02 0 21 0
26 46 15 0 7.3e+02 11 0
7 20 1.3e+02 33 17 6.3e+02 0
60 17 1 0 0 2 7.6e+02
750
600
450
300
150
0