Forest Cover Type Clustering

Pedro Carrion Castagnola
Department of Computer Science
University of Missouri - Columbia
pjc322@mail.missouri.edu

Dewi Kharismawati
Department of Computer Science
University of Missouri - Columbia
dek8v5@mail.missouri.edu

Abstract

On this project we will try different clustering techniques for a dataset that contains samples with cartographic variables from sections for the Roosevelt National Forest, Colorado. We propose methods to find which areas of the park and soil types are more easy to cluster and generate a new dataset that hopefully have samples easier to cluster. We apply different clustering algorithms and validate our results by calculating the NMIs and accuracies of our clustering with the clustering produced by the labels of the dataset. Finally, we compare our result. Spectral clustering method obtained a higher NMI measure for this dataset.

1. Introduction

Natural resources inventory information is important to any private, state, or federal land management. Forest cover type is one basic example of natural resources inventory. This type of inventory is usually retrieved directly from expert or estimated from remote sensing technology. Both techniques are costly and time consuming. Therefore, predictive models is one alternative method to obtain this kind of inventory.

This project analyzes inventory information from natural resources inventory located in Roosevelt National Forest, Colorado. This forest has minimum human-caused disturbance; thus, forest cover types are the result of ecological processes instead of human management practice. We want to study the relationship of forest cover to the surroundings characteristics. With this information we can also know what kind of trees can grow in different areas and soil types.

Previous research on cover type predictions are focused on neural network and supervised learning, where it still depend on dataset labels. Discriminant Analysis (DA) and Artificial Neural Networks (ANNs) was implemented in [1]. The accuracy from using ANN is 70.58%, whereas DA is 58.38[1]. In addition, this dataset was used in Kaggle challenge [2], the winner of the challenge implemented K-nearest neighbor, a supervised learning technique that used feature similarity based on how close the out-of-sample features resemble from the training set. With KNN the winner got an accuracy of 96.51% [2]. However, these techniques required labels to solve the problem. Whereas, labeling natural resources inventory is really expensive and time consuming, which bring such disadvantages if we want to predict all natural resources on earth. Therefore, in this project, we want to investigate how accurate can clustering algorithms group the same types of trees given the surroundings characteristics. By achieving these goals, we hope to help researchers to provide an automation system to predict tree cover. Therefore, we can give a real contribution to protect and manage natural ecology and wildlife.

This study examined the ability of clustering algorithm, including BSAS, k-means, spectral, gaussian mixture model, agglomerative, and combination between BSAS-kmeans to predict forest cover type classes in the area of interest. The prediction from all the algorithms were evaluated with Normalized Mutual Information (NMI) and purity (cluster accuracy) compared to the ground truth cover provided in the dataset. Finally, we want to investigate and compare each algorithm result and find which algorithm performs best in this dataset.

2. Data

This dataset was retrieved from Kaggle and UCI Machine Learning Repositories [2] [3]. Credits of the dataset correspond to the original authors. Each row of the dataset represent cartographic measurements from a 30 by 30 meters section of the Roosevelt National Forest Park, Colorado. This dataset originally contains 54 features and labels:

- 10 measurements (cartographic variables): Elevation, Aspect, Slope, Horizontal Distance To Hydrology, Vertical Distance To Hydrology, Horizontal Distance To Roadways, Hillshade 9am, Hillshade Noon, Hillshade 3pm, Horizontal Distance To Fire Points.
- 4 types of wilderness area (each type is a boolean nominal feature).
- 40 Soil Type (each type is a boolean nominal feature).
- Label: Cover type

This is a very large dataset, it has 581 012 samples. Also, as table 1 shows, it is not balanced. Given the different types of features (measurements and nominal) and the quantity of samples, a method to reduce the dimensionality and sample size was needed. Table 1 shows the class distribution of the labels.

Tree species	**Number of samples**
Spurce	211 840
Lodgepole Pine	283 301
Ponderosa Pine	35 754
Cottonwood	2 747
Aspen	9 493
Douglas-fir	17 367
Krummholz	20 510
Total	**581 012**

Table 1. Class distribution of the cover types (labels)

3. Methodology

3.1 Dataset reshaping

For this project we decided to work only with the first 10 measures since they were not nominal. The 4 area types and 40 soil types were converted to two nominals features, with values ranging from 1 to 4 and from 1 to 40 respectively.

After this reshaping we wanted to know which combination of area and soil type combination were more easy to cluster. Therefore, we divided the 581 012 samples of the original dataset into 160 subsets (for all possible combinations of 4 area types and 40 soil types). After this, we found the following information for each possible combination of area and soil type:

- 86 combinations did not have any sample.
- 7 had all samples belonging to the same class.
- 67 had more than 1 class.

Finally, to decide which combination of area and soil type contains samples more easy to cluster, we tried a clustering algorithm for each of the 67 subsets from combinations that contained more than 1 class. We had to use a fast algorithm because the number of samples for some combinations were still very large. The selected method was Agglomerative UPGMA (group average distance). With the clustering results we calculated the NMI for each combination. Table 2 shows 4 samples of the results obtained:

Area type	Soil type	Number of samples	Number of classes	NMI
2	22	1067	2	0.0215
2	23	1615	2	0.0029
2	24	416	2	0.2904
2	25	107	2	0.0352

Table 2. Information and NMI of four combinations of area and soil types clustered using Agglomerative UPGMA

With this information we were able to know which combinations of area and soil type were more easy to cluster based on the NMI result. With a threshold value for NMI of 0.2 we found that 10 combinations were easy to cluster and 57 were difficult to cluster.

Finally, to overcome the problems of processing large amounts of datasets, we decided to create a new dataset that contains the information of all the samples that belong to an easy to cluster combination of area and soil type. We called this new dataset "easy dataset". In the end, the results showed that the easy dataset were more easy to cluster than taking random samples from each class of the original dataset. The size of the easy dataset is 9 536 samples. It also included the samples were the combination of area and soil type contain only one type of class. The pipeline for the easy dataset creation is on figure 1.

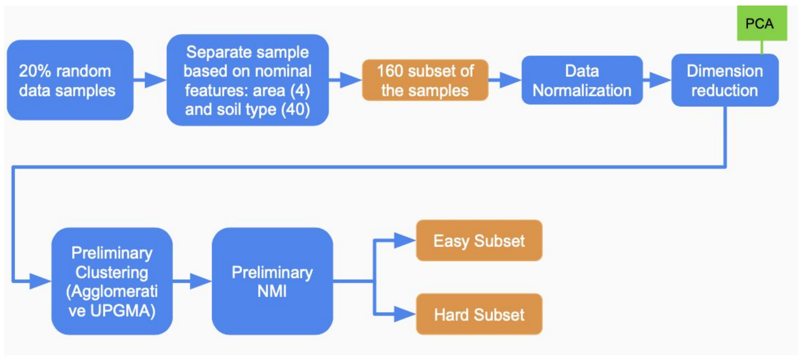

PCA
20% random data samples
Separate sample based on nominal features: area (4) and soil type (40)
160 subset of the samples
Data Normalization
Dimension reduction
Preliminary Clustering (Agglomerative UPGMA)
Preliminary NMI
Easy Subset
Hard Subset

Figure 1. Pipeline of the easy dataset creation

3.2 Clustering Algorithms

Based on figure 2 flow chart, the easy dataset we extracted from data preprocessing is reduced with PCA. The reduced dimension then clustered into 7 classes. We are going to implement some clustering algorithms to cluster the cover type, including BSAS, k-means, spectral clustering, Gaussian Mixture Model, Agglomerative Algorithm, and combination of BSAS-kmeans. We want to compare the ability of these algorithm to cluster the cover type, and which algorithm gives the best clustering.

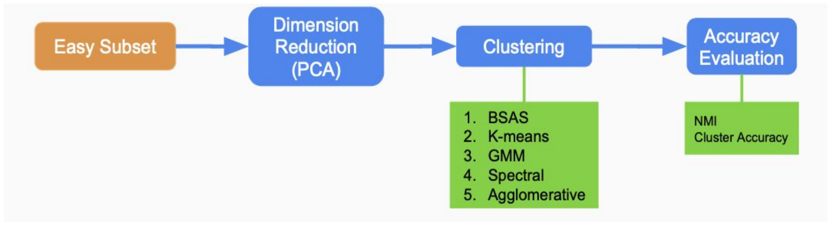


Figure 2. Pipeline for evaluating different clustering algorithms

3.2.1 Hierarchical Clustering - Basic Sequential Algorithmic Scheme (BSAS)

Basic Sequential Algorithmic Scheme (BSAS) is one of hierarchical clustering method. It is a top-down, all data belong to one cluster, then splits recursively as one moves down the hierarchy. The main idea of BSAS is to assign each data to existing cluster or to newly created cluster based on its distance from the existing cluster. We implemented BSAS in matlab by ourselves. Following is the pseudocode of BSAS we implemented:

```
m=1             %initialize cluster 1
Cm = {x1}       %initialize cluster center
For i = 2 to number of dataset
    Find Ck : d(xi,Ck) = min d(xi,Cj)1≤j ≤m  %find the min distance from data to c center
    If(d(xi,Ck)>Θ)AND(m<q) then              %if the d > threshold and # of cluster is < max cluster
        m=m+1                  %create new cluster
        Cm ={xi}
    Else
        Ck =Ck ∪ {xi} %else assigned the data xi to the cluster with nearest distance to the center
        Update cluster center
    End {if}
End {For}
```

3.2.2 K-means

K-means is a type of clustering algorithm that subdivide data points of dataset into cluster based on nearest mean values. Distance between points in each cluster is minimized to get the optimum division. K in k-means denotes the number of cluster in the data, and we required to determine k. Determining the right k is important because the algorithm is heavily dependent on the correctness of k value. We implemented k-means algorithm using matlab available function.

3.2.3 Spectral Clustering

Spectral clustering is a popular clustering method that uses information from the eigendecomposition of a similarity graph to obtain the similarity information of the data. For this project we used the program "Fast and efficient spectral clustering" [5] for Spectral clustering. This program let you try different methods for the creation of the similarity graph and for the clustering technique. The best results were achieved using a k-Nearest Neighbors Similarity Graph which considers that two samples are similar if one of them is within the k nearest neighbours of the other. Then it computes the unnormalized graph laplacian and the eigen analysis. Finally, a matrix containing the eigen vectors as columns is created and the final clustering is achieved using k-means on the rows of this matrix for a given k value. This clustering technique is based on Shi, J., & Malik [6].

3.2.4 Gaussian Mixture Model (GMM)

Gaussian Mixture Model uses a probabilistic approach to cluster the data. In this approach, we describe each cluster by its mean centroid, covariance and weight of the cluster. After calculate the parameter of each data point, GMM can calculate probabilities of it belonging to each cluster. We implemented GMM build in function from Matlab to get the distribution and the cluster them using matlab cluster build in function.

3.2.5 Agglomerative Clustering - UPGMA

Agglomerative UPGMA (Unweighted Pair Group Method with Arithmetic Mean) is a hierarchical and agglomerative clustering method that uses the group average position of the data in the cluster to calculate the distances between clusters.

3.2.6 Combinations of BSAS - Kmeans

In this algorithm, we combined the results of BSAS with K-means. First, we process the dataset on BSAS to get the cluster centers. Then, use those cluster centers to initialize the cluster centers of the k-means. This algorithm was implemented using our own BSAS code and build in K-means function.

3.3 NMI and accuracy

All the proposed clustering algorithms were compared using NMI and accuracy (purity) measurements. For the NMI our own method was implemented. For the accuracy the method from Mathworks file exchange was used [4]

Based on figure 2 flow chart, the easy dataset we extracted from data preprocessing is reduced with PCA. The reduced d

4. Results and discussion

After the easy data extraction, we did the dimension reduction from 10 dimension to 5 dimension. We tried to use both linear (PCA) and non-linear (t-sne) dimension reduction. Then, we applied k-means on both result under the same environment. NMI result from PCA is higher compare to t-sne. In addition, dimension reduction with PCA is much faster compare to t-sne. Therefore, we decided to use the reduced dimension from PCA for the clustering purpose.

In each of the experiment, we tried different parameters with purpose to get the best NMI and cluster accuracy, such as sigma in BSAS, k in k-means, and k in spectral for knn similarity graph. We also used, on some experiments, dataset randomization and data normalization.

4.1 BSAS

In BSAS, we experiment several parameters combinations of normalization, randomization, and sigma. Sigma is a threshold we defined for cluster creation condition. First of all, we tried easy dataset with data randomization, normalization, and PCA to 5 dimension,. Then, tried to find the best sigma, and it was sigma=1. The NMI is 0.1395 and accuracy of 30.15%. The next experiment was using randomized easy dataset and applying PCA to 5 dimension without normalization. After some trial and error, best sigma that give us the best result was sigma = 700. The NMI from this experiment is 0.2891 with the accuracy of 38.07%. The last experiment in BSAS was using easy no randomized dataset, apply PCA to 5 dimension with no normalization, with sigma = 700. This experiment gives the best result, the NMI is 0.3091 and the accuracy is 42.09%.

The challenge in BSAS in to define sigma. Picking the best sigma is crucial for creating new cluster. When sigma is too small, the NMI is too low. When the sigma is too large, it will cluster less than what it should be, which resulting in undefined NMI.

4.2 K-means

In k-means, we did experiments with different parameters as well. With easy dataset randomization, normalization, PCA, and k=7, the NMI is 0.1887 and the accuracy is 31.25%. With dataset randomization, no normalization, PCA, and k=7, the NMI is 0.2428 with cluster accuracy 37.1644%. The NMI and cluster accuracy is exactly the same with no randomization, no normalization, PCA and k=7.

In k-means, we pick k=7 because we know in advance what k is. Another problem using k-means is that it started with a random point as the cluster center initialization. Therefore, in each execution it cluster differently each time. Therefore, we set the random seed to control the random number generator using rng() function in Matlab. In addition, we apply replicates 15 times in each k-means execution. K-means will replicate the whole algorithm 15 times and get the minimum sum distance to centroid center. This solve the problem, and it give us the same cluster everytime we execute.

4.3 Spectral Clustering

For our implementation we tried different k values for the k nearest neighbor similarity graph until the final clustering gave us the best results. Figure 3 shows the k value tested for the construction of the similarity graph (horizontal coordinate) versus the NMI after the final clustering (vertical coordinate). For the experiment we tried the dataset with no normalization or PCA. We obtained the best results when k was equal to 5 with an NMI of 0.42.

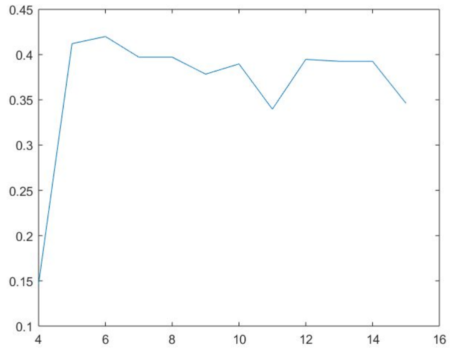


Figure 3. Results of NMI (vertical coordinate) versus value of k used for the construction of the similarity graph (horizontal coordinate).

4.4 GMM

In GMM, we did the same parameter experiment as in BSAS and K-means. First experiment was with randomized easy dataset, normalization, then apply PCA to 5 dimension. The NMI result is 0.2038 with accuracy of 32.2%. The second experiment was with randomized dataset, no normalization, and PCA to 5 dimension. The NMI is 0.3264 and cluster accuracy of 40.29%. The third experiment was with no randomization, no normalization, and PCA to 5 dimension. The NMI result is 0.2931 and the cluster accuracy of 38.6%. Surprisingly, the second experiment give us the best result of GMM. This is because dataset randomization affecting the mean center initialization.

In GMM, we encountered similar problem in k-means. The algorithm pick random points for parameter initialization. Therefore, we applied random number generator rng() as well to control the random seed each time the algorithm is executed.

4.5 Agglomerative

Agglomerative experiments was conducted in the easy dataset experiments. We tried all agglomerative algorithms to find the best NMI to extract the easy dataset. UPGMA was performing the best. Then, we applied PCA to 5 dimensions with normalization on easy dataset. Applying UPGMA in the result dataset gives NMI result of 0.2112 and the cluster accuracy of 42.8%.

4.6 Combination between BSAS then Kmeans

Only one experiment was conducted in this part. We run BSAS algorithm in point 4.1 with parameters that give us the best result, no randomization, no normalization, PCA to 5 dimension, sigma=700, and cluster=7. We extracted each cluster center, and use them to initialize the cluster center in k-means with random number generator for random number control. The NMI result is 0.1829 and the cluster accuracy is 39.25%.

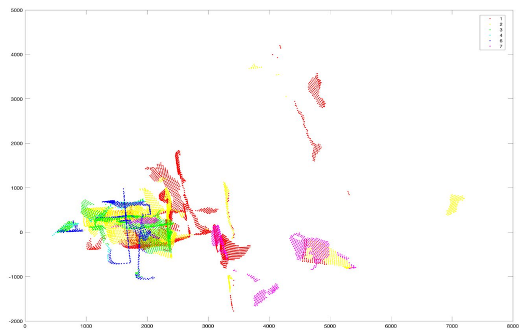
4.7 Comparison

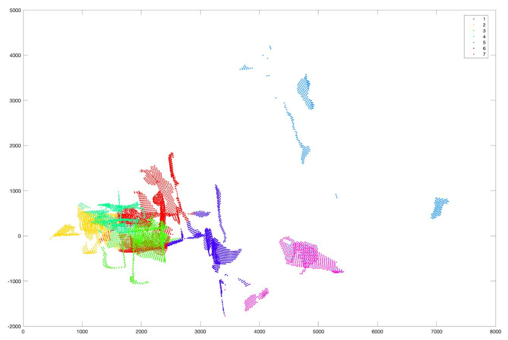
Based on point 4.1-4.5, we have discussed all experiments conducted to solve this cover type predictions. We summarized the best result from each algorithm on table 3.

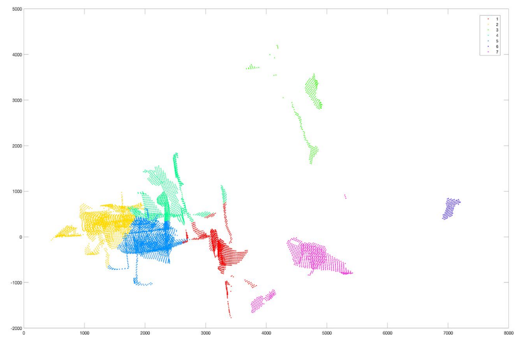
Clustering	**NMI**	**Accuracy**
Hierarchical	0.3091	42.1%
K-means	0.2428	37.2%
GMM	0.2931	38.6%
Spectral	**0.4200**	**51.1%**
Agglomerative	0.2112	42.8%
BSAS then Kmeans	0.1829	39.25%

Table 3. NMI and accuracies for each tested clustering algorithm.

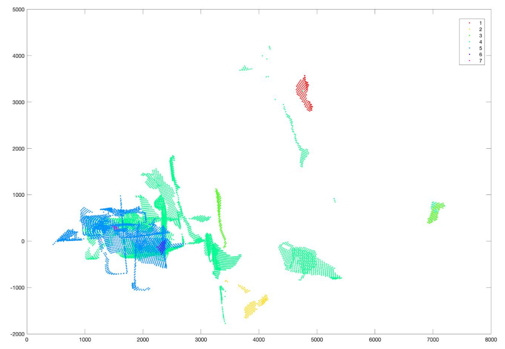
Figure 4 shows a visualization of the clustering results for each algorithm. These visualizations shows the first two dimensions of the samples as coordinates. Samples with the same color belong to the same cluster.


(a)

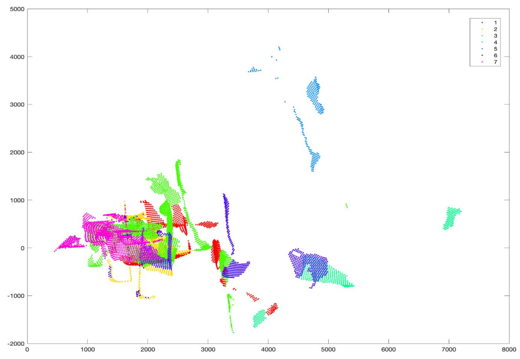

(b)



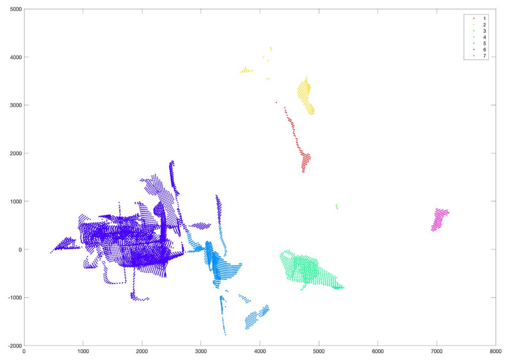
(c)



(d)



(e)



(f)

Figure 4. Plots above document the first 2 dimension dataset after PCA. **(a)**cluster representation from original label, **(b)**BSAS, **(c)**k-means, **(d)**spectral clustering, **(e)**GMM, **(f)**UPGMA

5. Conclusions

The cover type dataset is problematic to analyze. It has half a million data samples, which is too much data to analyze. In addition, there are 44 nominal categories features that impossible to manipulate using mathematical operations. Lastly, the class distribution of this dataset is not balanced, and many samples are mixed on a dimensional space, making it very difficult to cluster. Techniques like normalization, or techniques to reduce the dimension of the dataset like PCA and Isomap were tested but did not improve the results significatively. The analysis of the easy and hard combinations of area type were useful to find a more easy to cluster dataset with fewer number of samples.

By the results, we can conclude that spectral clustering is the best unsupervised learning approach to solve this forest cover type dataset. This means that the similarity graph constructed for spectral clustering (k nearest neighbours similarity graph) and the eigen analysis were useful to extract information than can separate the different cover types. However, the result is still far compared to using neural network or supervised algorithm.

References

[1] Blackard, J., and Denis J. Dean. 2000. "Comparative Accuracies of Artificial Neural Networks and Discriminant Analysis in Predicting Forest Cover Types from Cartographic Variables." Computers and Electronics in Agriculture 24(3):131-151. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.128.2475&rep=rep1&type=pdf

[2] Kaggle. (2016). Forest Cover Type Dataset. https://www.kaggle.com/uciml/forest-cover-type-dataset/home

[3] UCI Machine Learning Repository. Covertype Data Set https://archive.ics.uci.edu/ml/datasets/Covertype

[4] Praisan Padungweang. MathWorks file exchange. https://www.mathworks.com/matlabcentral/fileexchange/32197-clustering-results-measurement

[5] Ingo Burk. Bachelor Thesis. University of Stuttgart 2012. MathWorks file exchange. https://www.mathworks.com/matlabcentral/fileexchange/34412-fast-and-efficient-spectral-clustering

[6] Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. IEEE Transactions on pattern analysis and machine intelligence, 22(8), 888-905.