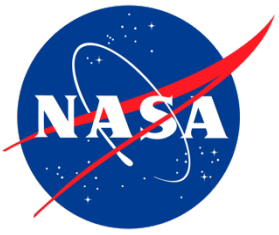

2024

K-Means Clustering and Random Forest protocols for lunar data products

USER GUIDE

VERSION 1.0.0, 08/14/2024

1. *Introduction:*

This document is meant to serve as a brief guide to the K-Means clustering and Random Forest protocols created for lunar data products. These notebooks are tailored for data cubes constructed from data preprocessed using Francesco Civilini's data processing pipeline.

2. *Data and methods:*

The data used in these notebooks corresponds to JMARS and Kaguya data products for the Apollo 17 landing site. The specific products used for the initial tests are:

Data Product	Instrument	Description	Source	Date
Kaguya TC Stereoscopic Uncontrolled Observations	KAGUYA's Terrain Camera	Stereo images with 10 m/pixel resolution	Astrogeology catalog	06/18/24
Clementine Optical Maturity Parameter	Clementine color cameras	Characterizes the overall maturity of lunar soil and crater ejecta by changes in reflectance spectra	JMARS	07/10/24
Diviner Thermal Inertia	LRO's Diviner Lunar Radiometer	Shows reflected solar and emitted IR radiation in nine spectral channels ranging from 0.3-400 microns	JMARS	07/10/24
Diviner Rock Abundance	LRO's Diviner Lunar Radiometer	Map of lunar rock abundance derived from Diviner thermal data	JMARS	07/10/24
Circular Polarization Ratio (CPR)	Mini-RF	Used as an indicator of surface and subsurface roughness	JMARS	07/10/24

The two notebooks created use K-Means Clustering and Random Forest algorithms to automatically classify the data cube provided into clusters or units.

K-means clustering is a popular unsupervised machine learning technique that groups data points together into a predefined number (k) of clusters. This algorithm allows researchers to find groups in an unlabeled dataset, and is oftentimes a good first step to explore patterns and variations in data. K-means clustering is sometimes employed as a tool to create labels for unlabeled datasets. These labels can later be used to train more complex algorithms, such as Neural Networks.

Random forest is a supervised machine learning algorithm consisting of an ensamble of decision trees. These trees learn simple decision rules inferred from the data to predict the value of a target variable. Random Forest is a popular technique amongst data scientists, and is regarded as one of the most easily implemented and powerful machine learning classifiers.

3. Files:

The folder “apollo_data_exploration” has the following files:

- KMeansClustering_FCiviliniPipeline.ipynb: Jupyter notebook with k-means clustering algorithm
- RandomForest_FCiviliniPipeline.ipynb: Jupyter notebook with random forest algorithm
- Kmeans
    - A17 (Area of interest)
        - combined_clustered_map_30Clusters_Protocol1_08-08.png: Cluster map created using protocol 1
        - combined_clustered_map_30Clusters_Protocol2_Part1_08-08.png: Cluster map of the Kaguya TC imagery, corresponding to part 1 of protocol 2.
        - combined_clustered_map_30Clusters_Protocol2_Part2_08-08.png: Cluster map of the JMARS data cube+ the cluster map of the Kaguya TC imagery, corresponding to part 2 of protocol 2.
- RF

- Results (results folder)
  - A17 (Area of interest)
    - predictedlabels_08-13.png: Labels predicted (based on the geologic map of the Apollo 17 landing site).

4. *Notebooks:*

4.1. K-means:

This K-means clustering notebook (KMeansClustering_FCiviliniPipeline.ipynb) is tailored for data acquired and preprocessed using Francesco Civilini's pipeline. The notebook is subdivided into two different protocols, based on the order in which files are clustered.

- ***Protocol 1:***
  - *Preprocessing:* This protocol reads in the Kaguya and JMARS data files, making sure to flip and rotate the JMARS data to match the orientation of the Kaguya imagery. Data is then resized to match the largest image and added to a data cube. This data cube is then flattened and NaN values are replaced with 0s. The flattened data is standardized using a a standard scaler.
  - *K-means:* Once the JMARS and Kaguya data have been preprocessed (as outlined above), k-means clustering is performed. The number of clusters and random state are defined to initialize the model. The resulting map is then plotted and saved to the designated directory.
- ***Protocol 2:***
  - ***Part 1:***
    - *Preprocessing 1:* This protocol first reads, flattens, and scales the Kaguya data.
    - *K-means 1:* k-means clustering is performed on the Kaguya data alone, making sure to define the number of clusters and random state. The resulting map and labels are saved and plotted.
  - ***Part 2:***

- *Preprocessing 2:* The JMARS data files are read, flipped, and rotated to match the orientation of the Kaguya imagery. The cluster map from part 1 is also read. Data is then resized to match the largest image and added to a data cube. Just as in protocol 1, the data cube is flattened, NaN values are replaced with 0s, and standardized using a standard scaler.
- *K-means:* We perform K-means clustering again, this time using the resuting cluster map from Part 1 as one of the data layers. The resulting map is then plotted and saved to the designated directory.

4.2. Random Forest:

This Random Forest notebook (RandomForest_FCiviliniPipeline.ipynb) is tailored for data acquired and preprocessed using Francesco Civilini's pipeline.

- ***Protocol:***
  - *Preprocessing:* This protocol reads in the Kaguya and JMARS data files, making sure to flip and rotate the JMARS data to match the orientation of the Kaguya imagery. Data is then resized to match the largest image and added to a data cube. Given that Random Forest is a supervised ML technique, our notebook makes use of prelabeled data for the Apollo 17 site (Garry, 1972) for training. This labeled map is loaded and reshaped to match the dimensions of the data cube layers.
  - *Set-up for Training:* Before applying the Random Forest to the data cube, training data must be generated. For this, we define a number of points on the data cube and assign each a label based on the geologic map layer. This is done by defining the number of points to be extracted from each label and allowing a function to randomly pick and save their indices. The location of the randomly selected points is then plotted, overlaying the geologic map. We devide the selected points into training and validation subsets, using 80% for training and setting 20% aside for validation. As a final step in the

training set-up, NaNs are removed from the data cube and replaced by the mean of the column, using a simple imputer.

- *Random Forest:* We define the model's hyperparameters and train it on the data reserved for training. Accuracy metrics are then calculated with respect to the validation data and displayed. We calculate precision, recall, and f1 score for each of the labels individually, as well as the overall accuracy, macro and weighted averages. Finally, the resulting labeled map is saved and plotted alongside the actual labels.