NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

MACHINE LEARNING IN MOBILE CUBESAT COMMAND AND CONTROL (MC3) GROUND STATIONS

by

Timothy J. Marczewski

June 2021

Thesis Advisor: Giovanni Minelli
Co-Advisor: Noah Weitz

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE June 2021	3. REPORT TYPE AND DATES COVERED Master's thesis

4. TITLE AND SUBTITLE MACHINE LEARNING IN MOBILE CUBESAT COMMAND AND CONTROL (MC3) GROUND STATIONS	5. FUNDING NUMBERS
6. AUTHOR(S) Timothy J. Marczewski	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	12b. DISTRIBUTION CODE A

13. ABSTRACT (maximum 200 words)

The Mobile CubeSat Command and Control (MC3) ground station network is a program designed to enable many organizations to command and control very small satellites (or CubeSats) in low-earth orbit. The MC3 network currently consists of ground stations that are geographically dispersed and utilize non-standard configurations of commercial off-the-shelf equipment. The non-standard configuration of each location poses a challenge for the small staff of MC3 network operators who monitor network and ground station health status. These operators rely on software and automation to ensure the MC3 network is healthy and can support any organization's mission. However, the problem is that a normal state in one location can look different from the normal state at another location in terms of equipment and, therefore, health status. Determining the normal state using machine learning will facilitate further analysis of ground station health and the implementation of near-real-time health status monitoring to augment the MC3 network operators' capabilities. The research focused on using the K-means++ unsupervised machine learning clustering algorithm to model the normal state. This research could not conclusively determine the normal state of the NPS MC3 ground station, but it does establish a launch point for further work

14. SUBJECT TERMS machine learning, MC3, Mobile CubeSat Command and Control, ground station, K-means	15. NUMBER OF PAGES 123
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

MACHINE LEARNING IN MOBILE CUBESAT COMMAND AND CONTROL (MC3) GROUND STATIONS

Timothy J. Marczewski
Major, United States Army
BS, Purdue University, 2008
MS, Missouri University of Science and Technology, 2014

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN SPACE SYSTEMS OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
June 2021

Approved by: Giovanni Minelli
Advisor

Noah Weitz
Co-Advisor

James H. Newman
Chair, Space Systems Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Mobile CubeSat Command and Control (MC3) ground station network is a program designed to enable many organizations to command and control very small satellites (or CubeSats) in low-earth orbit. The MC3 network currently consists of ground stations that are geographically dispersed and utilize non-standard configurations of commercial off-the-shelf equipment. The non-standard configuration of each location poses a challenge for the small staff of MC3 network operators who monitor network and ground station health status. These operators rely on software and automation to ensure the MC3 network is healthy and can support any organization's mission. However, the problem is that a normal state in one location can look different from the normal state at another location in terms of equipment and, therefore, health status. Determining the normal state using machine learning will facilitate further analysis of ground station health and the implementation of near-real-time health status monitoring to augment the MC3 network operators' capabilities. The research focused on using the K-means++ unsupervised machine learning clustering algorithm to model the normal state. This research could not conclusively determine the normal state of the NPS MC3 ground station, but it does establish a launch point for further work.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

API	application programming interface
COTS	commercial off-the-shelf
CSV	comma-separated values
IMS	Inductive Monitoring Systems
IP	internet protocol
LEO	low-earth orbit
MC3	Mobile CubeSat Command and Control
NASA	National Aeronautics and Space Administration
NPS	Naval Postgraduate School
PDU	power distribution unit
R&D	research and design
SATRN	Satellite Agile Transmit Receive Network
SDR	software-defined radio

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PROJECT STATEMENT

The Mobile CubeSat Command and Control (MC3) ground station network is a program designed to enable many organizations to command and control very small satellites (or "CubeSats) in low-earth orbit (LEO). This thesis refers to this class of very small satellites as CubeSats. The MC3 network currently consists of ground stations that are geographically dispersed. The participating organizations include U.S. government agencies, universities, academies, and foreign partners. The program aims to be low-cost, inclusive, and flexible, according to Dr. Minelli at the 2019 SmallSat Conference (Minelli et al., 2019). These goals are supported by utilizing commercial off-the-shelf equipment, open-source libraries, and software-defined radios (SDR) that are not always standardized across individual ground stations.

The non-standard configuration of each location poses a challenge for the small staff of MC3 network operators who monitor ground station health status. Keeping the staff size small is ideal to meet the goal of low-cost but hinders the manual monitoring capability within the MC3 network. These operators rely on software and automation to ensure the MC3 network is healthy and can support any organization's mission. However, the problem is that a normal state in one location can look different from the normal state at another location in terms of equipment and, therefore, health status data.

1. Objective

This thesis aims to use machine learning to determine the normal state of the Naval Postgraduate School (NPS) MC3 ground station. Determining the normal state using machine learning will facilitate further analysis of ground station health and the implementation of near-real-time health status monitoring.

2. Methodology

The datasets collected from various sensors at the NPS MC3 ground station are utilized to explore the feasibility of applying machine learning techniques to identify the

normal state. The data parameters are environmental, power, vibration, and electrical. The machine learning technique was limited to open-source unsupervised machine learning, specifically clustering with K-means++. This approach was based on previous applications in aerospace.

3. Analysis and Results

The analysis focused on using the K-means++ unsupervised machine learning clustering algorithm. This research uses a script written in Python that relies on several data analysis packages, such as Scikit-learn, to format the data and apply the K-means++ clustering algorithm (Pedregosa et al., 2011). Before the analysis with K-means++, the research applied the elbow method and silhouette scores to determine the fit or the ideal number of clusters in the data. 2D and 3D scatter plots illustrate the results for up to three parameters.

B. PROBLEM IDENTIFICATION AND HYPOTHESIS

All ground stations are mostly identical to one another, but each location has unique considerations and various hardware configurations. Therefore, the normal state model for one ground station does not effectively represent that of another ground station. Unsupervised machine learning is a possible solution to obtaining the normal state of individual ground stations with minimal intervention from the MC3 operations team.

The literature review in Chapter 2 introduces the MC3 ground station network, explains supervised and unsupervised machine learning, and reviews machine learning applications in aerospace. Chapter 3 details the methodology used during the data collection and analysis for this research. Chapter 4 presents the results gained from the initial data collection, analysis, and visualization. Chapter 5 concludes the research and discusses possible areas of future studies.

II. LITERATURE REVIEW

A. MOBILE CUBESAT COMMAND & CONTROL (MC3) GROUND STATIONS

The MC3 ground station network is designed to command and control CubeSats in low-earth orbit (LEO). MC3 also provides access for government and non-government entities for research and design (R&D) using CubeSats. The initial fielding of the MC3 ground control network took place in 2011 and, as of 2021, has nine ground station sites, according to the MC3 team (Minelli et al., 2019). The ground station sites as of May 2021 are listed in Table 1. The defining feature of the MC3 network is cost-effectiveness.

Table 1. MC3 Ground Station Locations and Status

Site (Designator)	Location	Capability	Status
Space and Missile Defense Command (SMDC)	Huntsville, AL	S-Band	Active
Naval Information Warfare Center Pacific (NIWC-PAC)	Pearl City, Hawaii	UHF/S-Band/X-Band	Future
Naval Postgraduate School (NPS)	Monterey, CA	UHF/S-Band	Active
Space Dynamics Laboratory (SDL)	Logan, UT	UHF/S-Band	Active
University of New Mexico / Cosmiac (UNM)	Albuquerque, NM	UHF/S-Band	Active
Air Force Institute of Technology (AFIT)	Dayton, OH	UHF/S-Band	Active
U.S. Coast Guard Academy (USCGA)	New London, CT	S-Band	Active
Malabar Transmitter Annex (MLB)	Palm Bay, FL	UHF/S-Band	Active
University of Alaska, Fairbanks (UAF)	Fairbanks, AK	S-Band	Active

The cost-effective MC3 ground station network provides an R&D environment accessible to organizations with limited budgets for small experimental projects with

CubeSats. Several MC3 ground station network features contribute to cost-effectiveness, including COTS hardware and open-source software (Minelli et al., 2019). Shown in Figure 1 is the MC3 ground station located at Malabar Transmitter Annex in Palm Bay, FL, and shows the typical layout of an MC3 ground station. This type of hardware and software allows the MC3 ground network to be flexible and accommodating to users' requirements.

Figure 1. MC3 Ground Station Located at Malabar Transmitter Annex in Palm Bay, FL. Source: Minelli et al. (2019).

The MC3 ground station network is linked together through the Satellite Agile Transmit Receive Network (SATRN) software. SATRN was created by the Space Dynamics Laboratory (SDL) and was designed to support the MC3 network mission and CubeSat operations (Minelli et al., 2019). The basic architecture for SATRN is depicted in Figure 2. Figure 2 includes the client, server, ground site, and CubeSat to visualize how SATRN ties all the pieces of the MC3 network together and provides bent-pipe communications from the user to the CubeSat (Minelli et al., 2019).

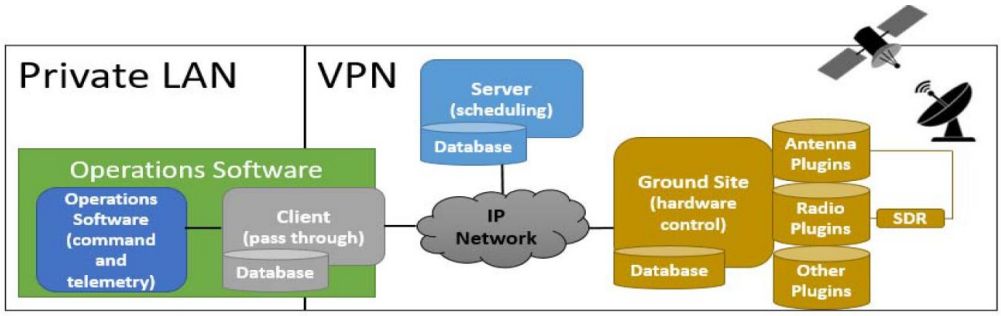


Figure 2. SATRN Architecture Depicting the Client, Server, and Ground site. Source: Minelli et al. (2019).

The MC3 network operations are continually evolving to meet specific user requirements and supporting more users. The requirements are increasing, but the size of the MC3 operations team is remaining the same. Increased automation has been identified as essential to keeping the MC3 network operational 24/7 while not increasing the current manning of the Satellite Operations Center (SOC) at NPS. Currently, the SOC is not staffed by operators 24/7 (Minelli et al., 2019). Maintaining a reliable network to conduct R&D using CubeSats is a primary focus of the MC3 operators and SOC.

B. MACHINE LEARNING

Machine learning algorithms have numerous classifications, but only supervised and unsupervised algorithms will be discussed in this review and thesis. Both supervised and unsupervised machine learning requires dataset inputs from which observation outputs are produced. In the chapter "Introduction to Machine Learning Algorithms" of the book *New Advances in Machine Learning*, Dr. Ayodele describes the difference between supervised and unsupervised machine learning. Dr. Ayodele highlights that supervised machine learning requires a labeled dataset input, while unsupervised machine learning only requires variable inputs from any dataset (Ayodele, 2010). Supervised or unsupervised machine learning is applied to individual situations primarily dependent on the presence of labeled datasets.

1. Supervised Machine Learning

Supervised machine learning requires the use of labeled datasets. It aims to set the conditions for a machine to learn a classification system that has already been created and not make a completely new classification system (Ayodele, 2010). A common example of supervised machine learning is recognizing handwritten letters and checking for correctness against the label. Dr. Kelleher et al. used letters as an example of a classification system that humans made when discussing supervised machine learning applications in his *MIT Press* journal "Fundamentals of Machine Learning for Predictive Data Analytics" (Kelleher et al., 2020). Figure 3 is an example of a labeled dataset used for training during supervised machine learning. The top box is the label, and the bottom box is the handwritten letter that the algorithm is trying to recognize.

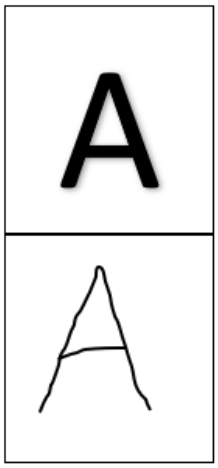


Figure 3. Example of a Labeled Dataset for Supervised Machine Learning. Adapted from Kelleher et al. (2020).

2. Unsupervised Machine Learning

Unsupervised machine learning does not rely on labeled datasets and instead generally uses extensive unlabeled input data. The goal of unsupervised machine learning is to get the machine to learn how to produce the correct outputs without being told the right answer (Ayodele, 2010). Dr. Ayodele discussed two types of approaches for unsupervised machine learning in his chapter "Types of Machine Learning Algorithms."

The first approach noted by Dr. Ayodele uses a reward or punishment system to reinforce the desired classification or deter an undesired classification. This approach did seem to resemble supervised machine learning, but the difference is that this approach is not forcing a predetermined classification. Instead, this approach is rewarding or punishing a decision made by the machine that aims to maximize rewards or minimize punishments (Ayodele, 2010). Dr. Ayodele pointed out that this approach can be time-consuming due to the constant trial-and-error type learning behavior and does require intervention by a human during the learning process. The benefit noted for this approach is that humans can provide decision guidance when working with a limited amount of data that does not cover every possibility in a system (Ayodele, 2010).

The second approach discussed by Dr. Ayodele for unsupervised machine learning uses clustering. The goal of clustering is to cluster data together by similarities (Ayodele, 2010). This approach is more unsupervised than the first approach, but it does assume that the clusters represent relevant classifications. The human is left to figure out what the clusters represent in the analyzed system (Ayodele, 2010). While appearing more unsupervised, the second approach still requires human intervention after the outputs have been produced to assign relevancy to the clustering outputs.

K-means clustering is a widely used clustering algorithm for unsupervised machine learning and is the predecessor to K-means++. To properly understand K-means++, an understanding of K-means is needed. Dr. Ayodele summarized K-means as simply trying to minimize the distance between data points and the center of a cluster (Ayodele, 2010). The simplicity of applying K-means makes the clustering algorithm accessible and a reliable starting point for analysis. The K-means algorithm, described by Dr. Ayodele, aims to minimize the objective function, such as the squared error function shown in Equation 1 (Ayodele, 2010).

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - c_j \right\|^2 \tag{1}$$

In Equation 1 $\left\| x_i^{(j)} - c_j \right\|^2$ is the distance measurement that is being minimized. The distance is between the data point $x_i^{(j)}$ and the cluster center c_j (Ayodele, 2010). The flow of the K-means algorithm is described in the following numbered list and visualized in Figures 4 and 5.

1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid.
3. When all objects have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated. (Ayodele, 2010)

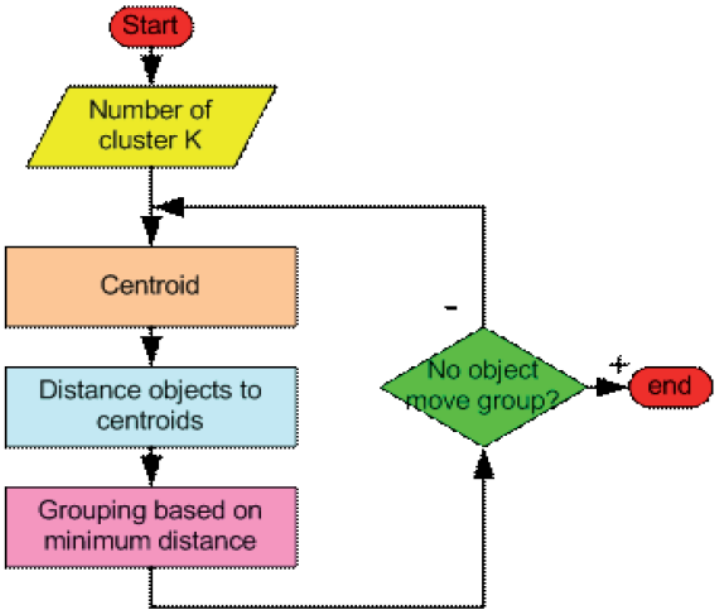


Figure 4. K-means Iteration Flow Chart. Source: Ayodele (2010).

Figure 5 represents the centroids' starting locations and subsequent movements towards the final location at the cluster centers. The movement of the centroid takes place after every iteration in Figure 4.

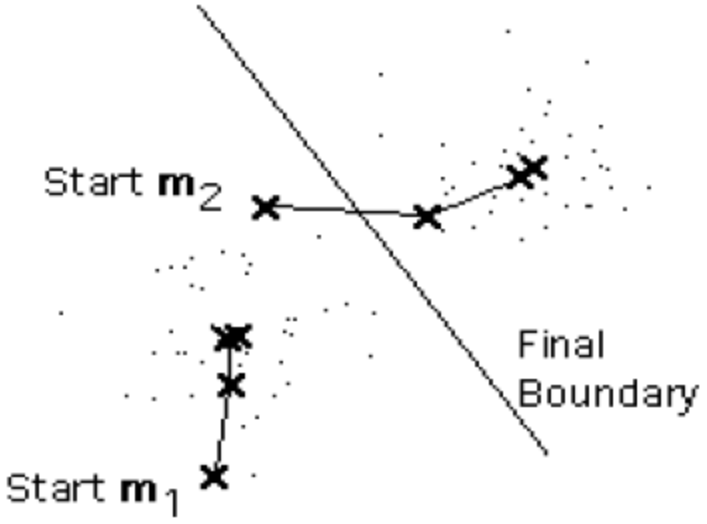


Figure 5. Example of m_1 and m_2 Moving to the Center of Two Clusters. Source: Ayodele (2010).

C. INDUCTIVE MONITORING SYSTEM (IMS)

In 2004 at the International Conference on Artificial Intelligence, David Iverson proposed a monitoring system that backed away from the model-based reasoning that was the current standard for monitoring systems (Iverson, 2004). Iverson's proposal was the Inductive Monitoring System (IMS). He used machine learning to produce the model or nominal dataset, as he called it. The data collected is indexed and organized into a data vector consisting of all the data measurements sampled or derived from various sensors. According to Iverson, a large dataset used for training the machine learning algorithm will likely contain almost all the value combinations required to determine the normal state of the system. Iverson used Table 2 below to demonstrate a visual example of the data vector of sampled measurements.

Table 2. Sample IMS Vector. Source: Iverson (2004).

Pressure A	Valve 1 Position	Pressure B	Valve 2 Position	Pressure C	Temperature 1	Temperature 2
2857.2	86.4%	1218.4	96.2%	1104.1	49.8	37.6

Once the data is organized into data vectors, a clustering algorithm such as K-means can cluster the vectors, as highlighted by Dr. Ayodele in the chapter "Type of Machine Learning Algorithms" (Ayodele, 2010). Iverson used the data in Table 3 to demonstrate a visual example of the cluster structure of the clustered data vectors. Table 3 represents two clusters that have been labeled as "High" and "Low" determined by a clustering algorithm using a large data structure of vectors.

Table 3. Sample IMS Cluster Structure. Source: Iverson (2004).

	Pressure A	Value 1 Position	Pressure B	Valve 2 Position	Pressure C	Temperature 1	Temperature 2
High	2857.6	86.8%	1219.2	96.3%	1105.0	50.1	38.2
Low	2855.8	86.2%	1215.7	95.5%	1103.2	49.6	37.5

The determination of the normal state clusters is the first key to IMS. These individual clusters can characterize the performance of a system as long as the operating conditions are covered in the dataset vectors (Iverson, 2004). The need for many operating conditions in the dataset vectors again stressed the need for an extensive operational dataset.

The next step in IMS is comparing real-time or near-real-time data vectors to the normal state clusters and calculating the deviation of new data from the normal state cluster. The deviation from the normal state will characterize the current system performance (Iverson, 2004). How much deviation from the normal state is acceptable is dependent on the system in which IMS is applied. For example, spacecraft will have less tolerance for deviation than a ground station satellite dish. IMS can be applied to any system in which operating data can be collected and compared against real-time or near-real-time data.

Iverson presented at the International Conference on Artificial Intelligence an analysis using IMS of the STS-107 Columbia mission. The IMS analysis focused on the data from the temperature sensors from the shuttle's wings. The results of the IMS analysis of the wings are visualized in Figure 6. IMS detected a deviation from the baseline almost

immediately after the foam strike on the wing, 17 days before the reentry and detection by mission control. A damaged thermal protection system on the left-wing caused the deviation off baseline due to the foam strike during launch. The baseline was generated using previous Columbia flights' sensor data (Iverson, 2004). Iverson used this example to highlight the usefulness and versatility of IMS.

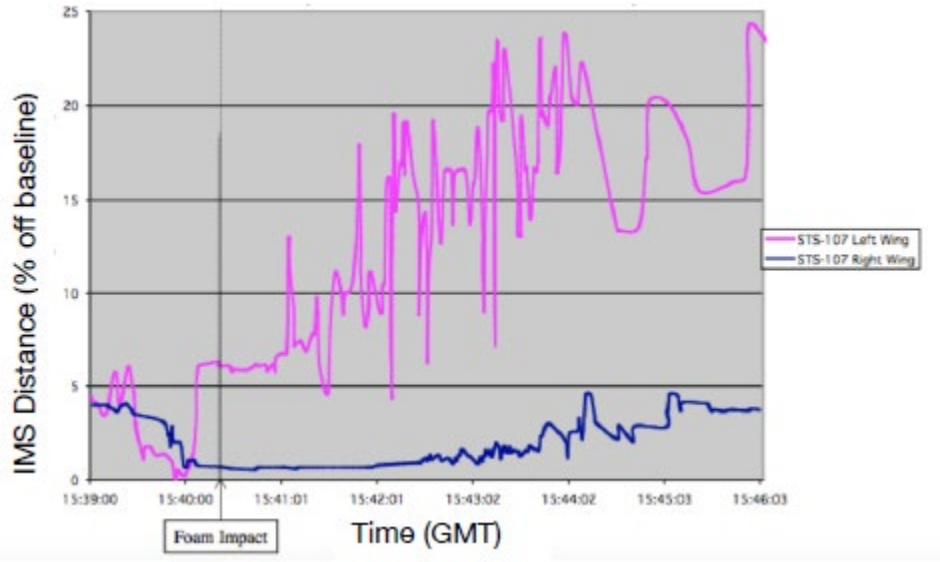


Figure 6. IMS Results for STS-107. Source: Iverson (2004)

D. APPLICATIONS OF IMS IN CUBESATS

Michelle Haddock and Serbinder Singh explored using an IMS-type system deployed in a CubeSat for their respective theses at California Polytechnic State University. Haddock's thesis was titled "Inductive Monitoring Systems: A CubeSat Ground-Based Prototype" and focused on creating a prototype for IMS verification in a ground-based controlled setting (Haddock, 2016). Singh's thesis was titled "A Data-Driven Approach to CubeSat Health Monitoring" and focused on testing adapted IMS approaches for spacecraft health monitoring (Singh, 2018). Both theses had a similar flow for the relationship between the learning and monitoring algorithms, as seen in Figures 7 and 8.

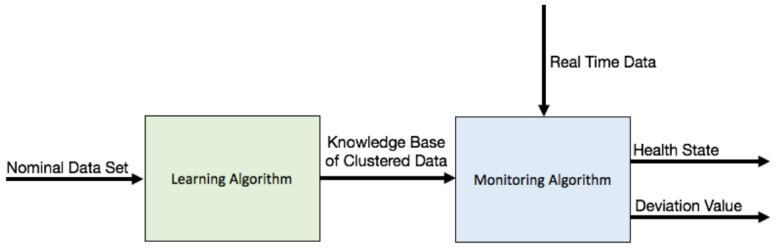


Figure 7. Learning and Monitoring Algorithm Relationship. Source: Haddock (2016)

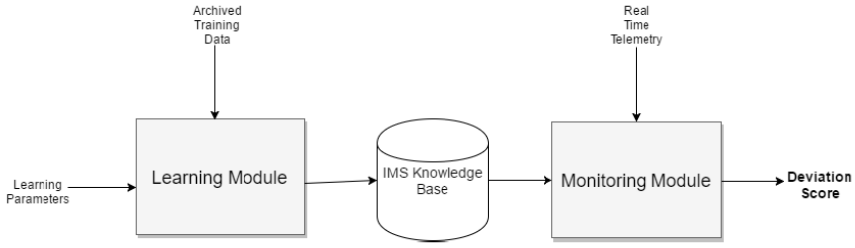


Figure 8. Learning and Monitoring Algorithm Relationship. Source: Singh (2018)

Haddock initially used the same approach as Iverson in the learning algorithm and implemented K-means for testing. This approach using K-means did provide convergence of clusters with the testing data (Haddock, 2016). The test supported the previous success that Iverson demonstrated with K-means (Iverson, 2004). Haddock also implemented and tested K-means++, which provided a noticeable improvement over K-means in the time required for cluster convergence. K-means++ improves on K-means by choosing the starting location of the center of the clusters, or centroids, spaced far enough away from one another to create unique clusters. K-means chooses random locations unless the user specifies the locations for the starting centroids with no regard to how close the cluster centroids may be to one another. K-means++ removes the need for the user to specify the starting locations and improves unique cluster convergence (Haddock, 2016). Haddock's

use of K-means++ in the learning algorithm improved over the original systems proposed by Iverson.

Singh analyzed the original IMS approach by Iverson from a data-driven perspective for implementation in California Polytechnic University's, PolySat lab. Singh noted that there was no need for the human monitoring the system to be an expert on the system when using IMS. Moreover, a human does not need to know what a healthy system should look like because the learning algorithm in IMS determines what a healthy system looks like based on the data collected during the system's normal operation (Singh, 2018). The data-driven analysis by Singh highlighted again that IMS is system agnostic.

THIS PAGE INTENTIONALLY LEFT BLANK

III. METHODOLOGY

The research design for this thesis drew from the previous work of Haddock, Singh, and Iverson. Iverson generally described the use of IMS in different systems with a focus on space vehicles. Both Haddock and Singh implemented IMS for CubeSat health monitoring. The previous success demonstrated by Haddock and Singh with CubeSat health monitoring helped guide the direction of this research towards an unsupervised cluster-based K-means++ algorithm due to its simplicity and effectiveness during the learning portion of a health monitoring system. Applying unsupervised machine learning using the K-mean or K-means++ algorithm is outlined in the approaches of all three previous IMS applications from the review of the literature.

The K-means++ algorithm was implemented over the standard K-means algorithm in this thesis based on the positive results from Haddock's thesis and success with CubeSat health monitoring. With the K-means++ algorithm selected, the data collected at the NPS ground station was the next step for the design. The first data set contained environmental data only. The environmental data included temperature, relative humidity, and dew point from six separate locations on the NPS ground station. The second data set contained environmental, power, and vibration data. The third data set contained environmental, vibration, and pointing data. Datasets were added and removed as the research evolved based on sensor testing, raw data observations, and the addition of new sensors. The reasoning for the inclusion or exclusion of data is discussed in the results.

The analysis algorithm and data set identification allowed for the analysis application design to be initiated. For each of the datasets (one, two, and three), the flow of the application was identical, and only the data input changed. The analysis application's internal script changed to accommodate the various datasets that were each formatted differently from their sensor sources. Comma-separated values (CSV) files were used to output the sensor data and input data to the application. The application flow for analysis is depicted in Figure 9, with the objective of this thesis labeled as "Thesis Objective."

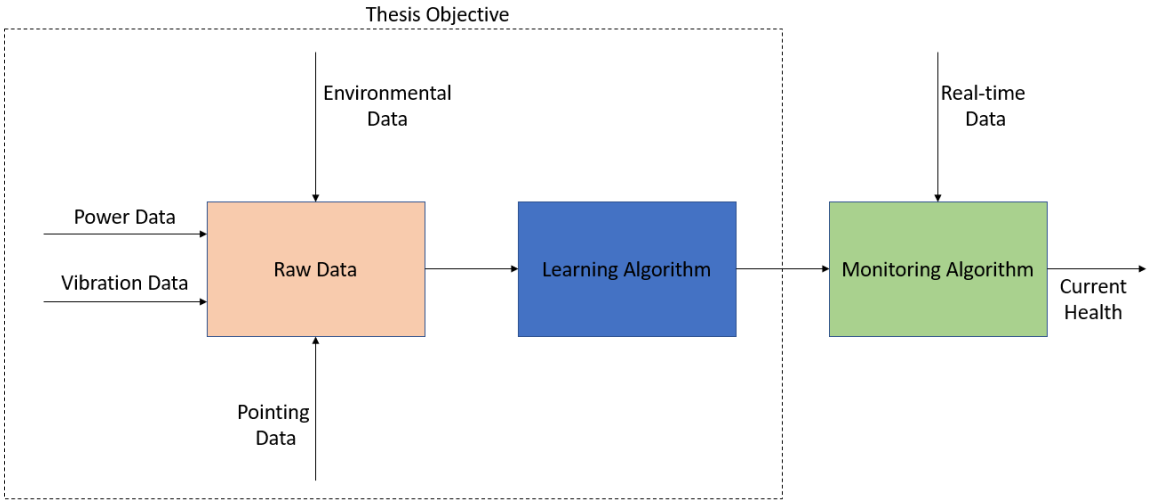


Figure 9. Data and Application Flow Chart. Adapted from Haddock (2016) and Singh (2018).

A. RESEARCH QUESTIONS

The following research questions were proposed during the initial design of the research to create a complete machine learning and monitoring application based on IMS. These proposed research questions covered the entire analysis and application, as this was the proposed scope of this thesis. Some questions were answered throughout the research design and analysis, some were not, and some questions led to more questions that will be discussed in the results and future work.

1. What data are collected by the NPS ground station?
2. How often is the data collected at the NPS ground station?
3. How and where is the data from the NPS ground station stored?
4. Are additional sensors needed to produce a normal state?
5. What other Department of Defense (DOD) systems could this approach be applied?
6. Should a cloud-based solution be explored for flexibility and processing?
7. Can the data from the sensors be analyzed in real-time?

8. What processing capability is needed for real-time machine learning analysis and forecasting?

9. Can anomalies and faults be mitigated autonomously?

10. With improved resiliency and autonomy, can the ground stations become more expeditionary?

11. What are the applications for a more expeditionary ground station?

B. SETTING AND SAMPLE

The data analyzed was generated at the NPS ground station during operational periods. A model was not used to generate the data; only operational ground station data was analyzed. During the duration of this research, the ground station added additional sensor and data output capabilities. Ground station changes are discussed as each data collection tool is introduced and explained.

C. DATA COLLECTION

This section presents how the data was collected and stored before analysis.

1. Environmental Data

Environmental data points were collected using the Network Technologies Incorporated ENVIROMUX environmental monitoring system, as shown in Figure 10 (Network Technologies Inc, 2021).

Figure 10. NTI ENVIROMUX E-5D Medium Enterprise Monitoring System. Source: Network Technologies Inc (2021)

The environmental data collected was timestamped and labeled by the location of the sensor on the ground station. The NPS ground station has six environment sensor locations. The serial number corresponds to the ground station location. All data for this thesis was collected at the NPS ground station site with SERIALNUM equal to 100. The SERIALNUM column is excluded from Table 4. The environmental data collected are the temperature (C), relative humidity (%), and dew point (C). The collection locations are inside the dome, on the high power amplifier (HPA), disk control box, and internal to the Enviromux sensor system. The data is sampled at a 5-minute interval. A sample of the environment data as output into the CSV file is shown in Table 4. Only one sensor was shown as an example in Table 4.

Table 4. Environmental Data Sample

RECORDDATE	UTC	EXTSENS1TMP (C)	EXTSENS1RH (%)	EXTSENS1DEW (C)
9/4/2021 17:00	1617984002	24	36	8.8
9/4/2021 17:05	1617984302	25	35	8.5
9/4/2021 17:10	1617984602	25	35	8.6
9/4/2021 17:15	1617984902	25	35	8.5
9/4/2021 17:20	1617985202	25	35	8.5
9/4/2021 17:25	1617985502	25	35	8.5
9/4/2021 17:30	1617985802	25	35	8.5
9/4/2021 17:35	1617986102	25	34	8.1
9/4/2021 17:40	1617986402	25	34	8.3
9/4/2021 17:45	1617986702	25	34	8.1
9/4/2021 17:50	1617987002	25	34	8.3
9/4/2021 17:55	1617987302	25	34	8.4
9/4/2021 18:00	1617987602	25	33	8

2. Vibration Data

Vibration data points were collected using a Remote Monitoring Solutions Sensaphone 4–20mA type vibration sensor, as shown in Figure 11 (Sensaphone, 2021)

Figure 11. Sensaphone 4–20 mA Type Vibration Sensor. Source: Sensaphone (2021)

The vibration data collected was timestamped and collected at one location on the moving dish of the NPS ground station. The vibration data was collected in mm/sec at an interval of about 0.3 seconds or approximately 3–4 Hz. A sample of the vibration data as output into the CSV file is shown in Table 5.

Table 5. Vibration Data Sample

TIME	RESPONSE (mm/sec)
17:29:25.803	0.11
17:29:26.070	0.11
17:29:26.342	0.12
17:29:26.636	0.12
17:29:26.907	0.12
17:29:27.202	0.09
17:29:27.473	0.09
17:29:27.771	0.12
17:29:28.043	0.12
17:29:28.314	0.13
17:29:28.586	0.10

3. Power Data

Power data points were collected using a CyberPower Switched-by-Outlet Power Distribution Unit (PDU), as shown in Figure 12 (CyberPower, 2021).

Figure 12. CyberPower PDU81001 Switched Metered-by-Outlet PDU. Source: CyberPower (2021).

The power data was timestamped and labeled according to the NPS ground station location and the internet protocol (I.P.) address of the PDU. The power data was collected from the ground station components' main power supply in amps at an interval of 60 seconds. A sample of the power data as output into a CSV is shown in Table 6.

Table 6. Power Data Sample

RECORDDATE	UTC	SITE	IP_ADDRESS	OUTLET5 (AMP)	OUTLET6 (AMP)
20.11.2020, 17:53:35.898	1605894815	NPS	192.168.151.25	0	2.3
20.11.2020, 21:11:56.385	1605906716	NPS	192.168.151.25	0	2.3
20.11.2020, 21:12:00.956	1605906720	NPS	192.168.151.25	0	2.3
20.11.2020, 21:13:00.352	1605906780	NPS	192.168.151.25	0	2.3
20.11.2020, 21:14:00.920	1605906840	NPS	192.168.151.25	0	2.3
20.11.2020, 21:15:00.292	1605906900	NPS	192.168.151.25	0	2.3
20.11.2020, 21:16:00.836	1605906960	NPS	192.168.151.25	0	2.3

4. Dish Pointing Data

Pointing data was collected using SATRN. The network layout of SATRN is shown in Figure 2. SATRN, as discussed in the MC3 section of the literature review, ties the client, server, ground site, and CubeSat together within the MC3 network. The point data is output

from SATRN through an application programming interface (API). The API enables SATRN to output in a few formats, but the CSV output format was used for this thesis as the universal output format for all data collected. The pointing data is timestamped, and the data points collected were the dish pointing azimuth, dish pointing elevation, and the range of the dish to the satellite. The azimuth and elevation were recorded in degrees, and the range was recorded in km. A sample of the pointing data as output into a CSV is shown in Table 7.

Table 7. Pointing Data Sample

Time	Azimuth (Deg)	Elevation (Deg)	Range (K.M.)
2021-04-09T17:30:04.824Z	323.89	0	2378.36
2021-04-09T17:30:05.824Z	323.99	0.01	2372.96
2021-04-09T17:30:06.824Z	324.11	0.06	2367.57
2021-04-09T17:30:07.824Z	324.22	0.11	2362.18
2021-04-09T17:30:08.824Z	324.33	0.16	2356.81
2021-04-09T17:30:09.824Z	324.44	0.20	2351.44
2021-04-09T17:30:10.824Z	324.55	0.25	2346.07
2021-04-09T17:30:11.824Z	324.67	0.30	2340.72

D. DATA ANALYSIS

The data analysis was conducted using the Python programming language with several open-source packages. Python is an open-source programming language licensed by the Python Software Foundation (Python Software Foundation, 2021). The bulk of the analysis was done using Scikit-learn for machine learning and NumPy for data analysis. Both Scikit-learn and NumPy are open-source software packages that work within the Python programming environment (NumPy, 2021). Scikit-learn is also commonly referred to as sklearn. According to "Scikit-learn: Machine Learning in Python" by Pedregosa et al., sklearn does not utilize graphical processing unit (GPU) acceleration to process the machine learning algorithms. Being open-source and not requiring a GPU makes sklearn ideal for application in this thesis because it can run on almost any type of computer without any special hardware (Pedregosa et al., 2011).

E. ASSUMPTIONS

The primary assumption during data collection at the NPS ground site is that the data collected represents normal operating conditions, and the ground site is currently operating normally. The primary assumption during the data analysis is that the software packages such as sklearn and NumPy are functioning correctly.

IV. RESULTS

The results present three datasets. Each of these datasets was an evolution of the data collection and analysis code. The data points were collected at various times from November 2020 to April 2021, and the code varied from each dataset but followed the same methodology. This chapter only presents a sample of the outputs from the results. The complete set of outputs are in Appendix A, and the python code that generated the results is in Appendix B.

A. DATASET ONE

Dataset one focuses solely on environmental data and serves as a partial validation that K-means++ was clustering data. Figure 13 is referred to as an elbow curve. The elbow curve is used to visually determine the optimal number of clusters to use in K-means++. Haddock, in her thesis, described the elbow as the natural bend that occurs right before the rapid decrease in the variance within the clusters as the number of clusters increases. She added that the slope significantly decreases after the bend, resulting in an elbow, and the decrease in slope indicated only marginal improvement with the addition of more clusters (Haddock, 2016). The elbow bend point is the optimal number of clusters based on the elbow curve. Based on Figure 13, five clusters are the optimal number of clusters based on the elbow curve.

Figure 14 is the first set of environmental data points analyzed displayed with K-means++ in a color-coded scatter plot. The different colors visually represent separate clusters. The number of clusters was determined to be five based on the elbow curve analysis in Figure 13. The first observation drawn from Figure 14 is that the ambient temperature and dew point data are consistent. The dew point temperature will always be less than the ambient temperature. If the dew point temperature were higher than the ambient temperature, then the relative humidity would be over 100% which is impossible. The second observation drawn from Figure 14 is the presence of clusters. The clusters appear to be in five continuous groups, which indicates K-means++ successfully analyzed and clustered the input data.

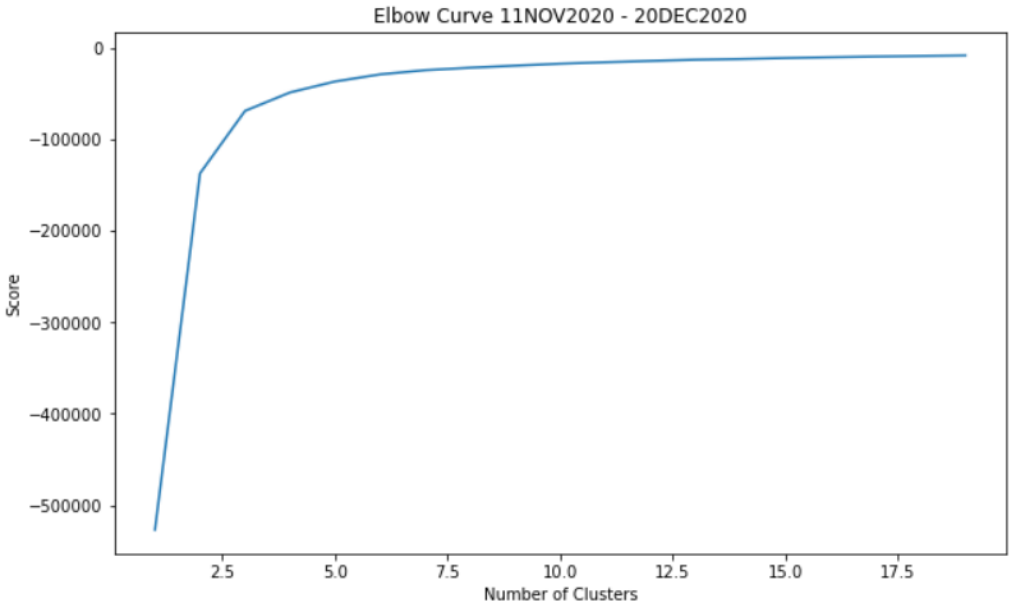


Figure 13. NPS Dome Environmental Data Elbow Curve

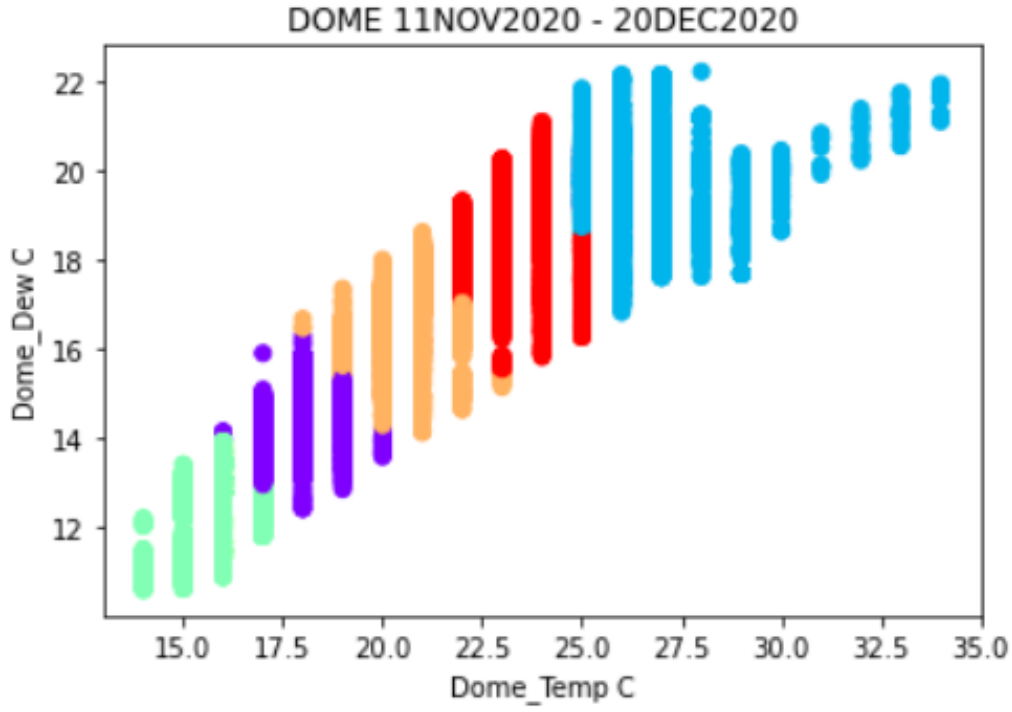


Figure 14. NPS Dome Temperature (C) vs Dew Point (C) with K-means++ Clusters

Figure 15 visualizes the environmental data with five clusters but with a visual representation of relative humidity and temperature instead of temperature and dew point, as in Figure 14. The clusters are grouped into separate blob-like sections, indicating successful analysis and clustering. The appearance of the data looks different visually, but the clusters are present in both figures.

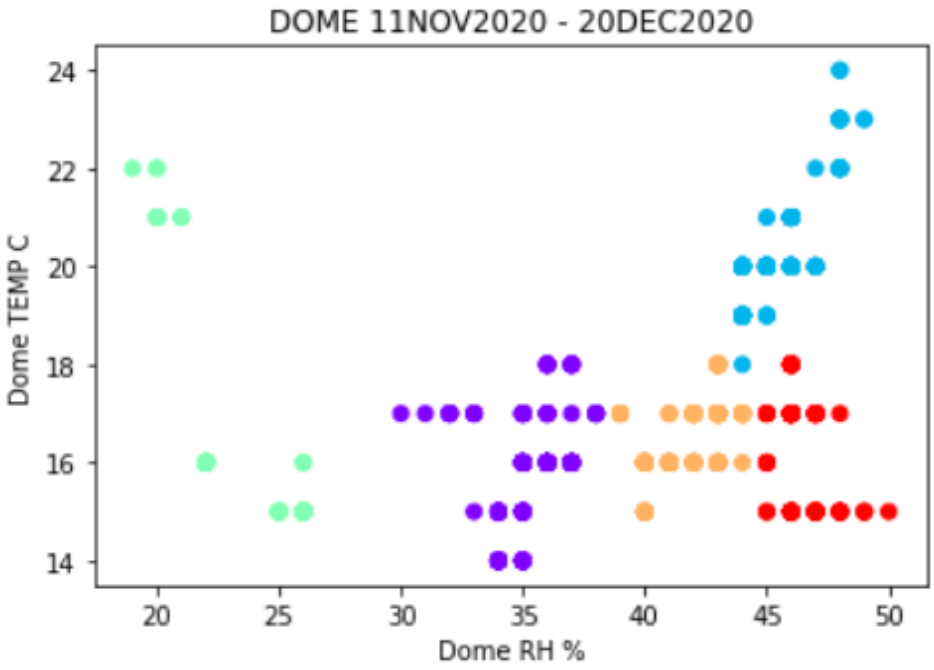


Figure 15. NPS Dome Relative Humidity (C) vs Temperature (C) with K-mean++ Clusters

The hour-of-the-day variable was introduced after the promising analysis of the environmental data. Figure 16 visualizes the analysis of relative humidity, temperature, and hour-of-the-day in three dimensions. The addition of a three-dimensional visualization added complexity to the visual analysis and indicated that three-dimensional visualization would likely be the limit of visual analysis. The data points and clusters are visible in Figure 16, but only confirmation of clustering can be validated from the visualization.

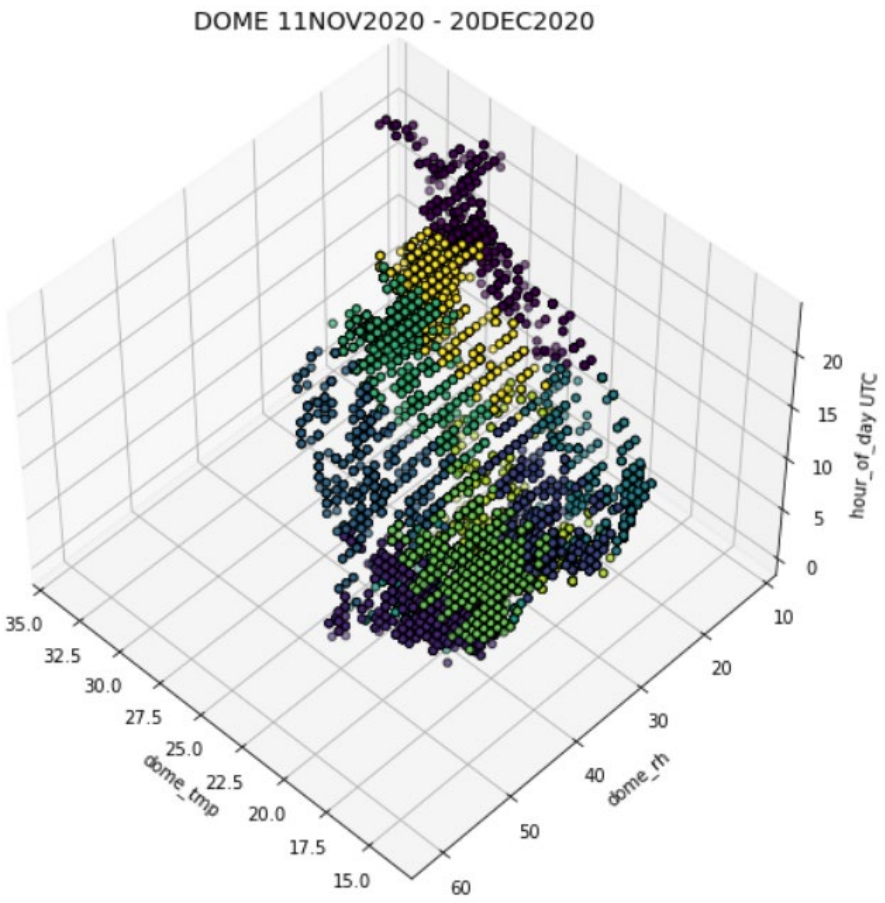


Figure 16. NPS Dome Relative Humidity (C) vs Temperature (C) vs Hour of the Day (UTC)

Dataset one's large set of environmental data collected over months was ideal for validating the methodology and K-means++ implementation. The analysis of dataset one did fall short of confirming a complete normal state of the NPS ground station because the analysis was limited to time and environmental data, which led to adding additional data sources in dataset two.

B. DATASET TWO

Dataset two introduced power and vibration data to the environmental data. The addition of data from two other sources serves two purposes. The first purpose is to test the combination of multiple data sources with data sampled at different rates into a single dataset that can be analyzed. The missing data caused by data points sampled at different

times are filled using a simple linear interpolation. The second purpose of dataset two is to build on the dataset one results and observe the clustering of additional data collected from the NPS ground station.

The data collected from the CyberPower PDU, when visualized, showed minimal fluctuation and low fidelity in current measurements. Figure 17 highlights the lack of change and fidelity over a month-long period. A closer look at the data exported from the PDU to the CSV file confirmed that the fidelity of the current measurements was only to a tenth of an amp. This level of fidelity would likely provide marginal results during analysis.

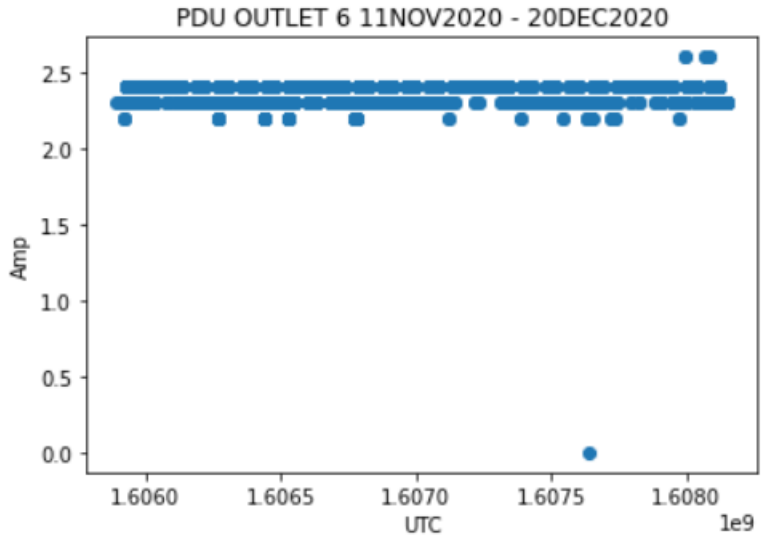


Figure 17. NPS Power PDU Time (UTC) vs Current (Amps)

The marginal results predicted by the visualization of the power data in Figure 17 were confirmed and analyzed of dataset two. As the data is analyzed, Figure 18 confirmed that the optimal number of clusters is five. Figure 19 visualizes the K-means++ analysis of dataset two and clusters generated from outlet1, the outlet for the vibration sensor, and the responses recorded from the vibration sensor while the dish is moving during a satellite pass. Figure 19 visualizes marginal results for the power data and vibration data because the clusters mainly generate off the vibration data.

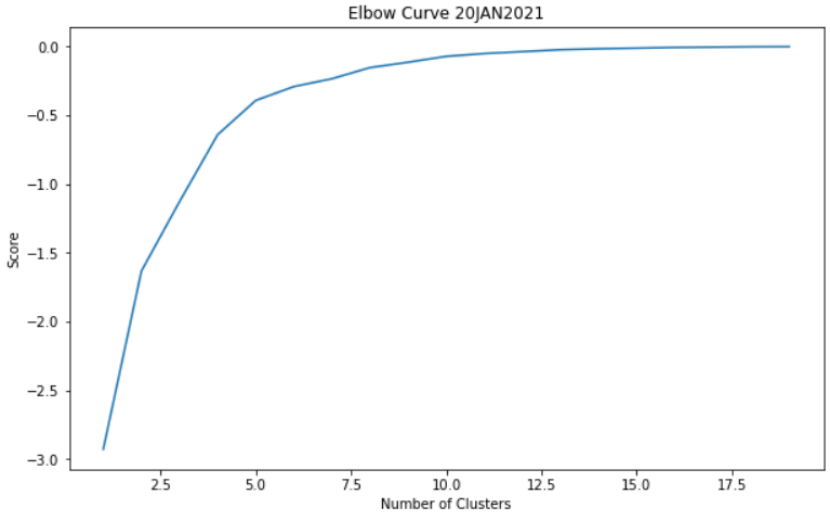


Figure 18. NPS Power, Environmental, and Vibration Data Elbow Curve

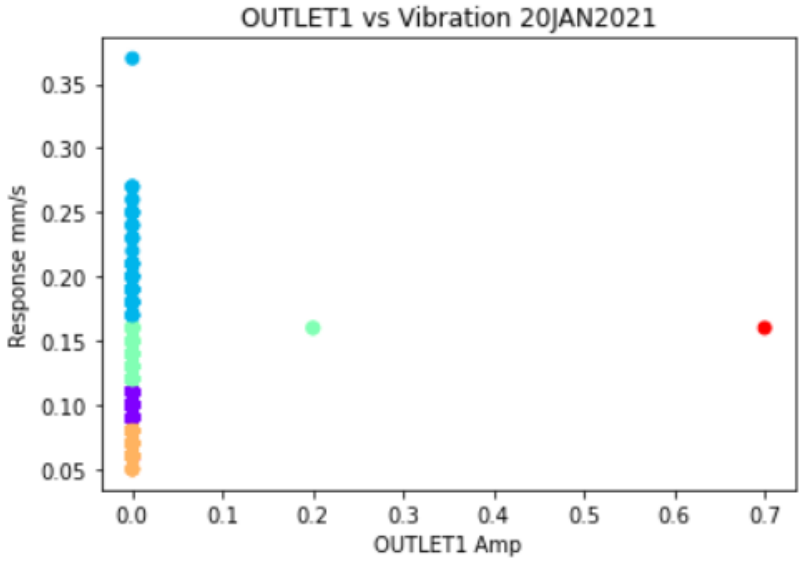


Figure 19. NPS Ground Station Vibration Sensor Outlet (Amps) vs Vibration Response (mm/sec)

Figure 20 builds on Figure 19 and visualizes the additional environmental variable of HPA temperature in three dimensions to confirm the marginal results of the power data. The visualization confirmed the marginal nature of the power data and highlighted that the temperature data exhibited the same marginal results during a pass. The results of dataset two were enlightening because the results showed that some data being analyzed may not

be relevant for short analysis periods, such as a pass length. As a result, dataset three was structured to analyze data that was relevant during a pass length.

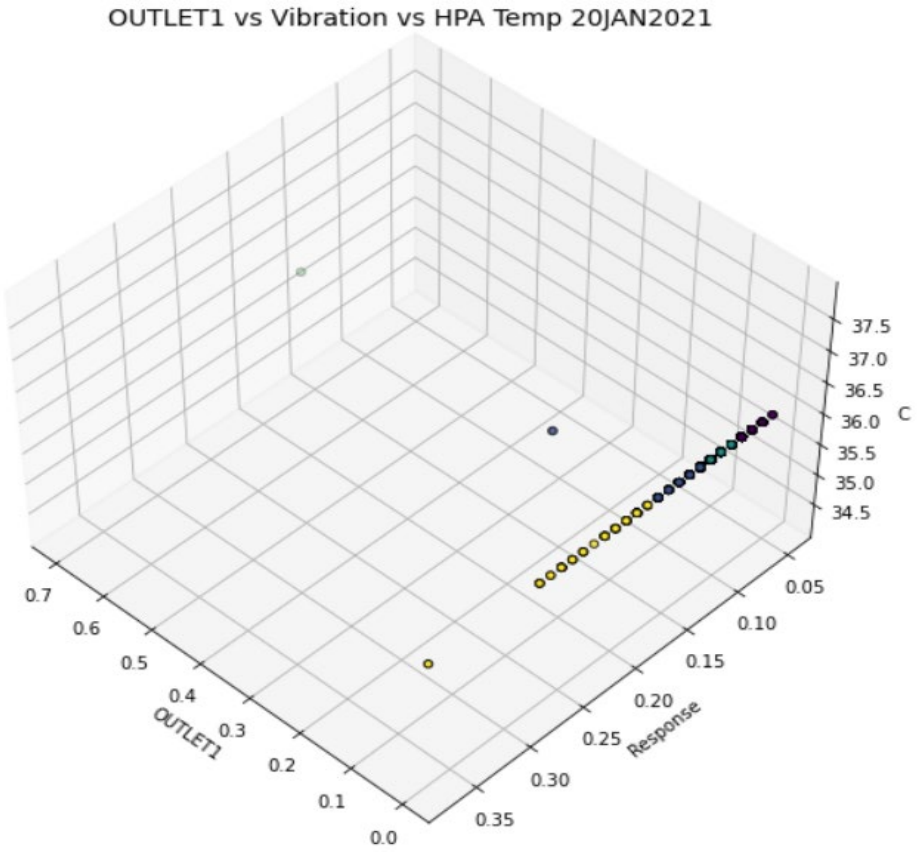


Figure 20. NPS Ground Station Vibration Sensor Outlet (Amps) vs Vibration Response (mm/sec) vs HPA Temperature (C)

C. DATASET THREE

Dataset three consists of the data collected from the vibration sensor, environmental sensors, power sensors, and dish pointing data. However, the analysis using K-means++ only uses the vibration response data and the dish pointing data for analysis. The use of these two sub-datasets is based on the results from dataset two. Dataset three includes ten satellite passes with a mix of high, medium, and low maximum elevation parameters. Three identical low and high maximum elevation paths were conducted, while four medium

maximum elevation paths were conducted. The parameters of the paths are presented in Tables 8, 10, and 12 by starting azimuth, max elevation, and ending azimuth.

Silhouette coefficients were calculated in dataset three as another means to determine the optimal number of clusters and compare against the elbow method used in previous datasets. From the sklearn documentation, the silhouette coefficient is the mean of the intra-cluster distance and the nearest-cluster distance with the best value of one and the worst value of negative one (Pedregosa et al., 2011). The clusters are more compact and spaced apart from one another as the silhouette coefficient approaches one, a characteristic of distinct clusters. The silhouette coefficient method opens up the path to a more autonomous means of determining the optimal number of clusters. A visual interpretation of an elbow curve is not required when using the silhouette coefficient method. Tables 9, 11, 13, 14, and 15 display the silhouette coefficients alongside the elbow curves in Figures 21, 24, 27, 30, 32, but the elbow curve method was favored for dataset three in determining the optimal number of clusters because it was used in the previous datasets.

Each of the paths, low, medium, and high, is presented separately before all the paths are combined to visualize the difference in vibration responses at different max elevations. Separating the visualizations aided in analyzing and understanding what data the K-means++ algorithm is possibly favoring for determining clusters. The determination of specific data being favored is helpful in future scaling and normalization of datasets.

1. Low Max Elevation Path

The low max elevation path had the least vibration response with most data points between 0.04 and 0.1 mm/sec, as seen in Figures 22 and 23.

Table 8. Low Max Elevation Path Parameters

Starting Azimuth (degrees)	Max Elevation (degrees)	Ending Azimuth (Degrees)
323.89	10.61	74.06

Table 9. Low Max Elevation Path Silhouette Coefficients

Number of Clusters	Silhouette Coefficient (Max = 1)
2	0.66
3	0.62
4	0.60
5	0.59
6	0.58
7	0.57
8	0.57
9	0.56

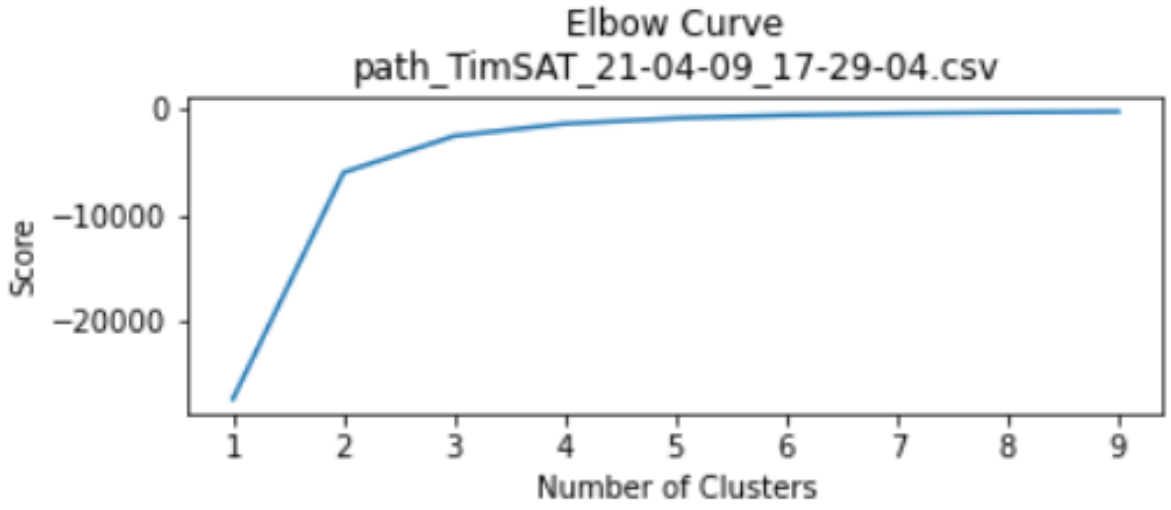


Figure 21. NPS Low Dish Path and Vibration Data Elbow Curve

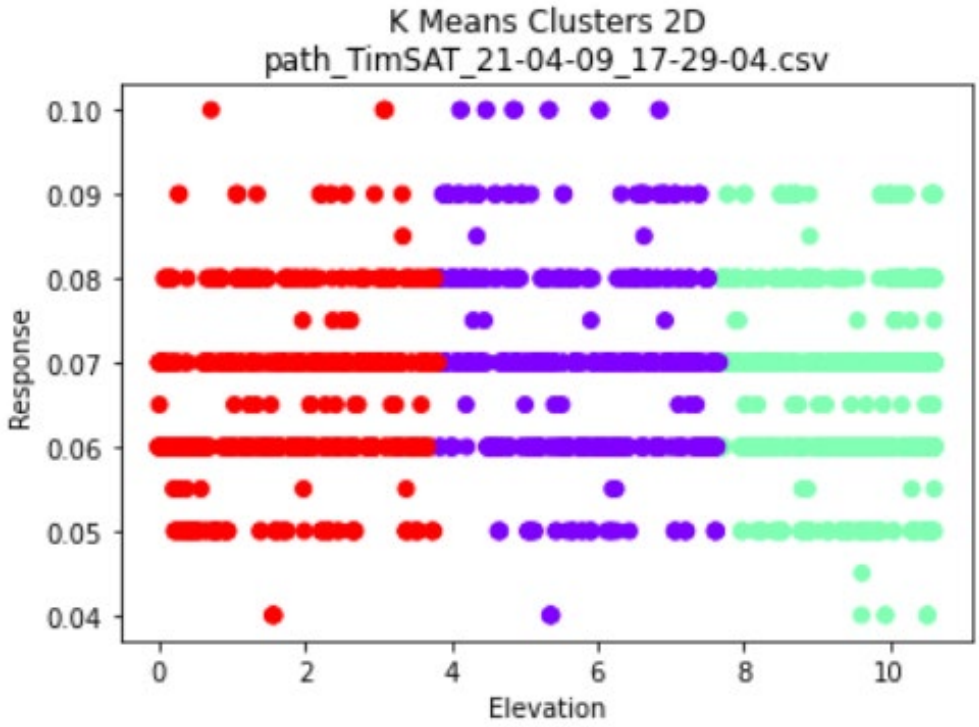


Figure 22. NPS Ground Station Low Elevation (deg) vs Vibration Response (mm/sec)

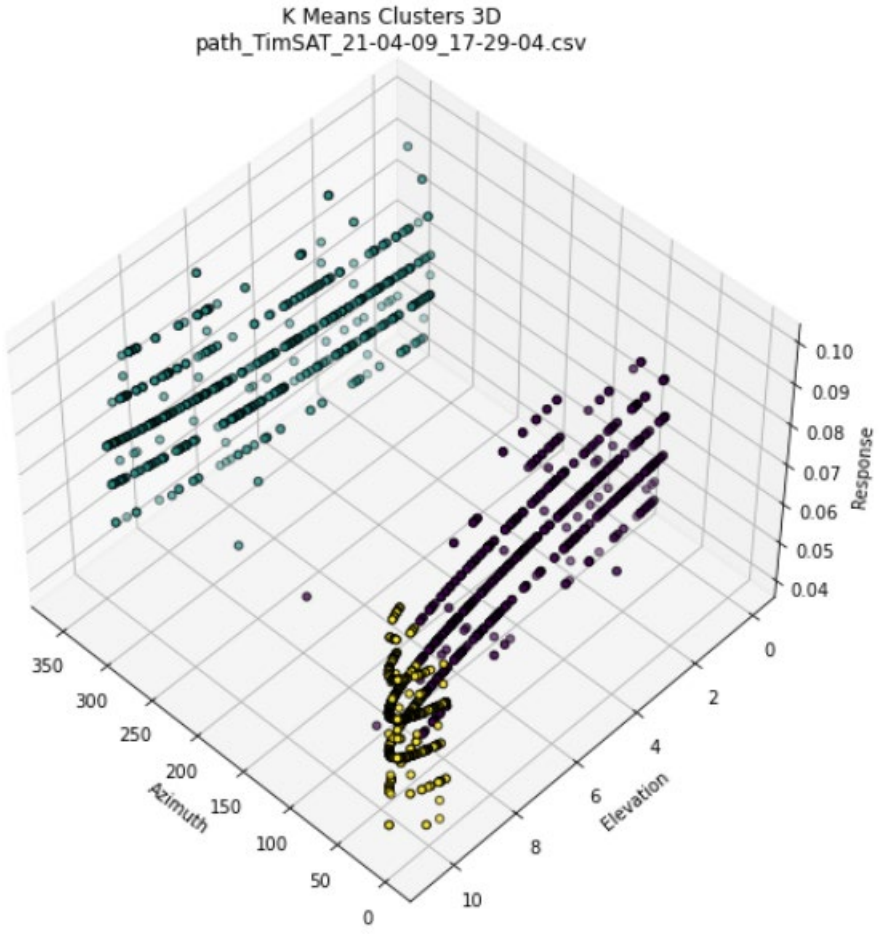


Figure 23. NPS Ground Station Low Elevation (deg) vs Azimuth (deg) vs Vibration Response (mm/sec)

2. Medium Max Elevation Path

The medium max elevation path had the middle vibration response with most data points between 0.04 and 0.12 mm/sec, as seen in Figures 25 and 26.

Table 10. Medium Max Elevation Path Parameters

Starting Azimuth (degrees)	Max Elevation (degrees)	Ending Azimuth (degrees)
243.80	40.90	43.99

Table 11. Medium Max Elevation Path Silhouette Coefficients

Number of Clusters	Silhouette Coefficient (Max = 1)
2	0.66
3	0.62
4	0.59
5	0.58
6	0.57
7	0.56
8	0.56
9	0.56

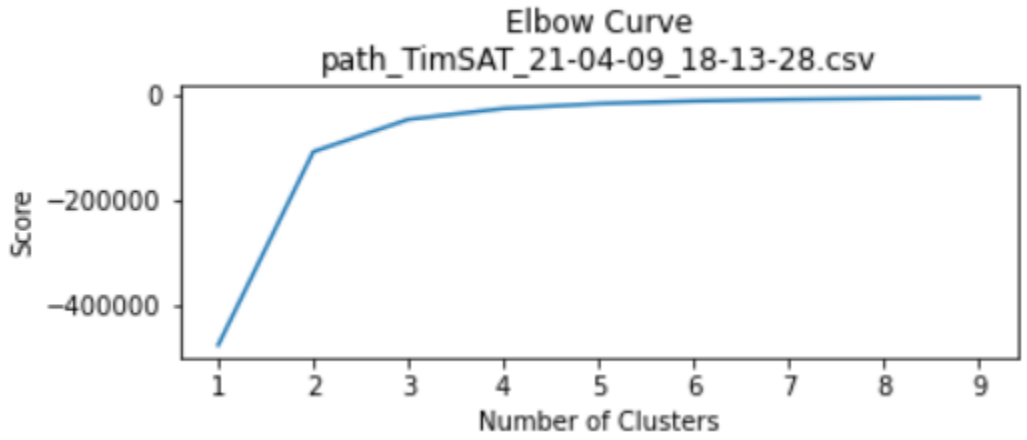


Figure 24. NPS Medium Dish Path and Vibration Data Elbow Curve

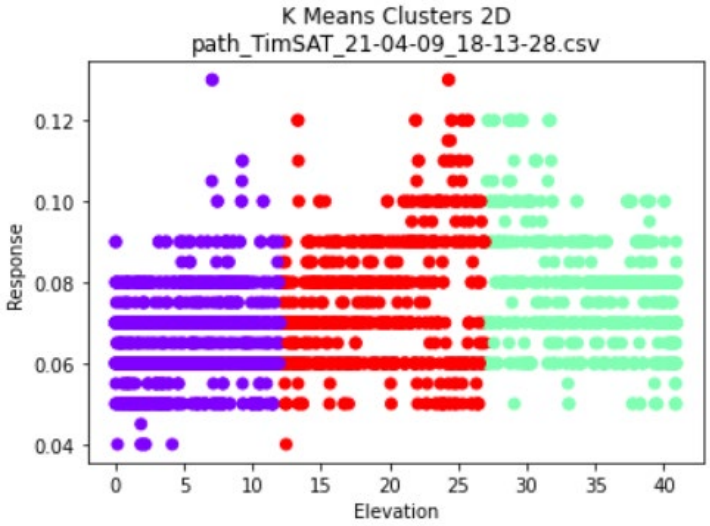


Figure 25. NPS Ground Station Medium Elevation (deg) vs Vibration Response (mm/sec)

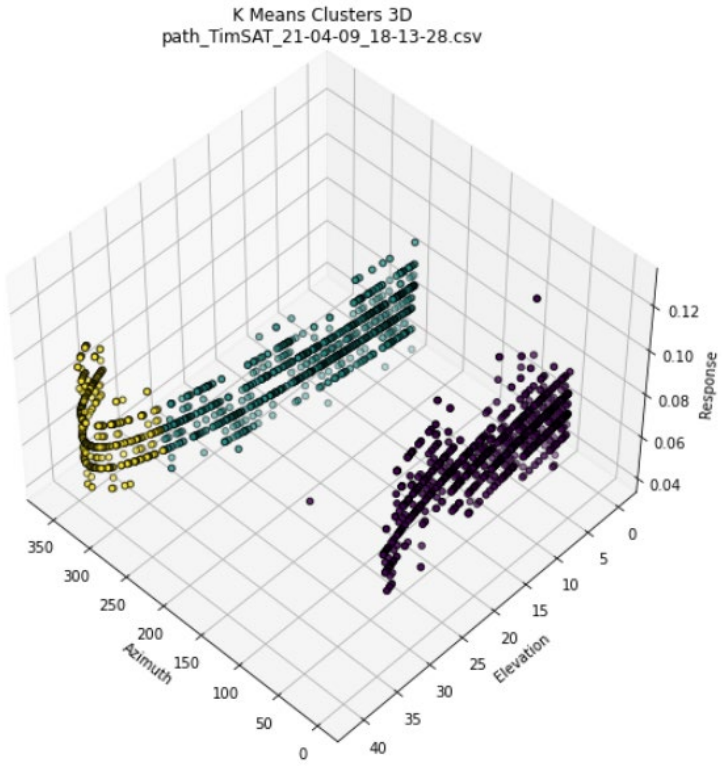


Figure 26. NPS Ground Station Medium Elevation (deg) vs Azimuth (deg) vs Vibration Response (mm/sec

3. High Max Elevation Path

The high max elevation path had the most vibration response with most data points between 0.04 and 0.2 mm/sec, as seen in Figures 28 and 29.

Table 12. High Max Elevation Path Parameters

Starting Azimuth (degrees)	Max Elevation (degrees)	Ending Azimuth (degrees)
220.57	62.20	135.47

Table 13. High Max Elevation Path Silhouette Coefficients

Number of Clusters	Silhouette Coefficient (Max = 1)
2	0.68
3	0.63
4	0.60
5	0.58
6	0.57
7	0.568
8	0.56
9	0.55

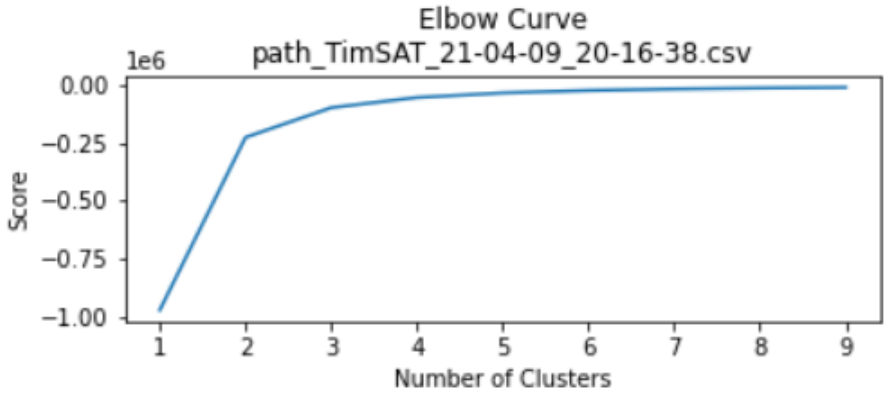


Figure 27. NPS High Dish Path and Vibration Data Elbow Curve

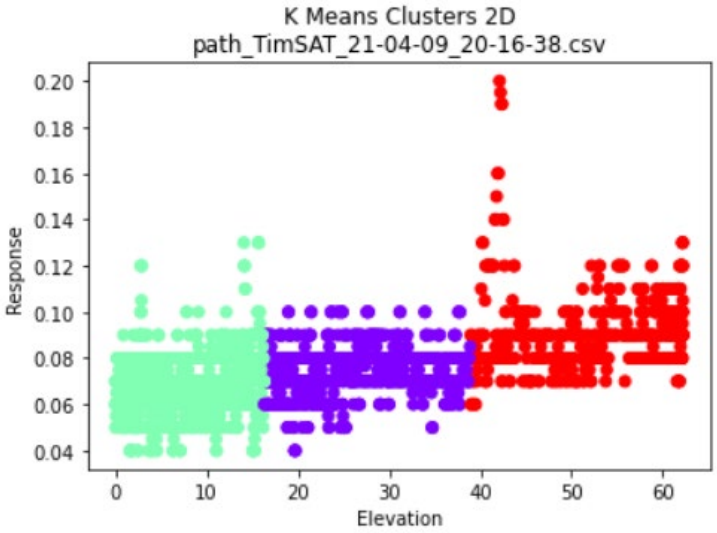


Figure 28. NPS Ground Station High Elevation (deg) vs Vibration Response (mm/sec)

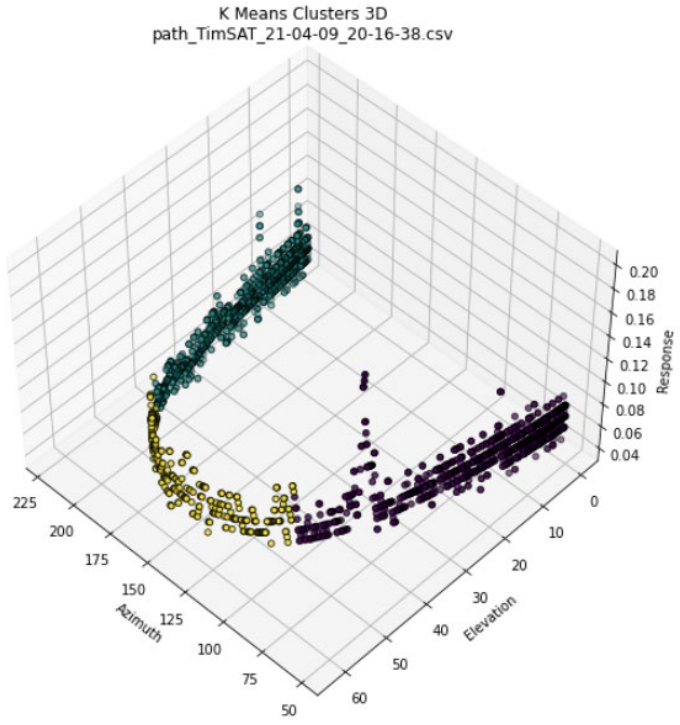


Figure 29. NPS Ground Station High Elevation (deg) vs Azimuth (deg) vs Vibration Response (mm/sec

4. Combined Paths

The combined paths analyzed all ten paths of data as a single dataset. The two-dimensional visualization is represented in Figure 31, and the three-dimensional visualization is represented in Figure 33. Figure 31 looks similar to the two-dimensional visualization in Figures 22, 25, and 28 but with an increase in density. The three-dimensional visualization for the combined dataset differs from the visualization in Figures 23, 26, and 29 because it is beginning to fill in the normal state. The variation of the path parameters demonstrates that the algorithm is learning what the normal state looks like, and it simply needs more data. Figure 33 has the appearance of a pseudo-noise floor for the normal state.

Table 14. Combined Elevation Path Silhouette Coefficients

Number of Clusters	Silhouette Coefficient (Max = 1)
2	0.71
3	0.64
4	0.62
5	0.59
6	0.59
7	0.58
8	0.56
9	0.57

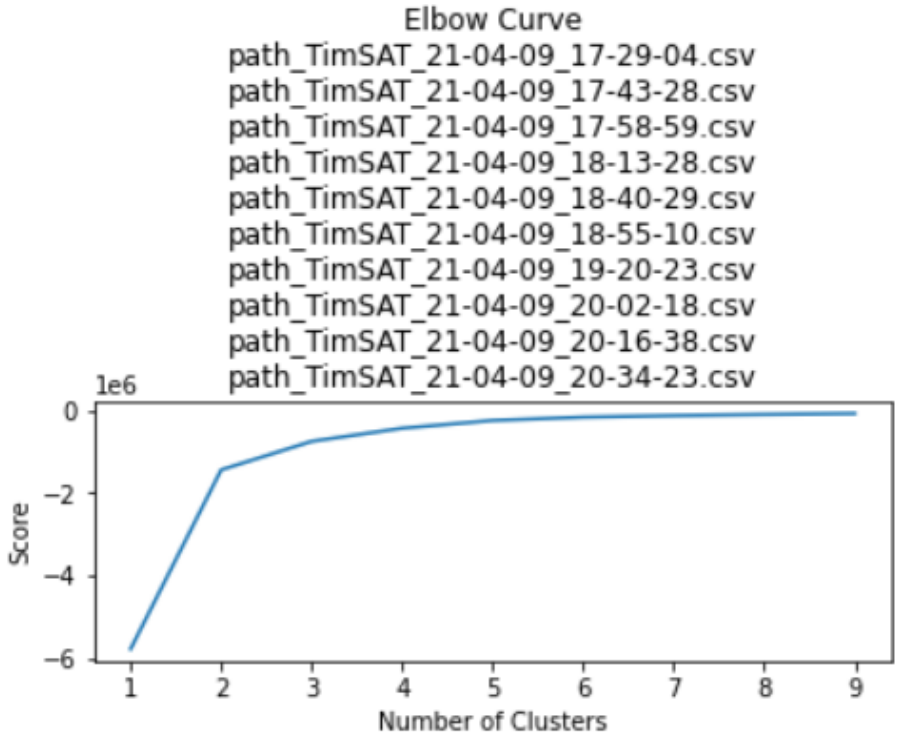


Figure 30. NPS Dish Path and Vibration Data Elbow Curve

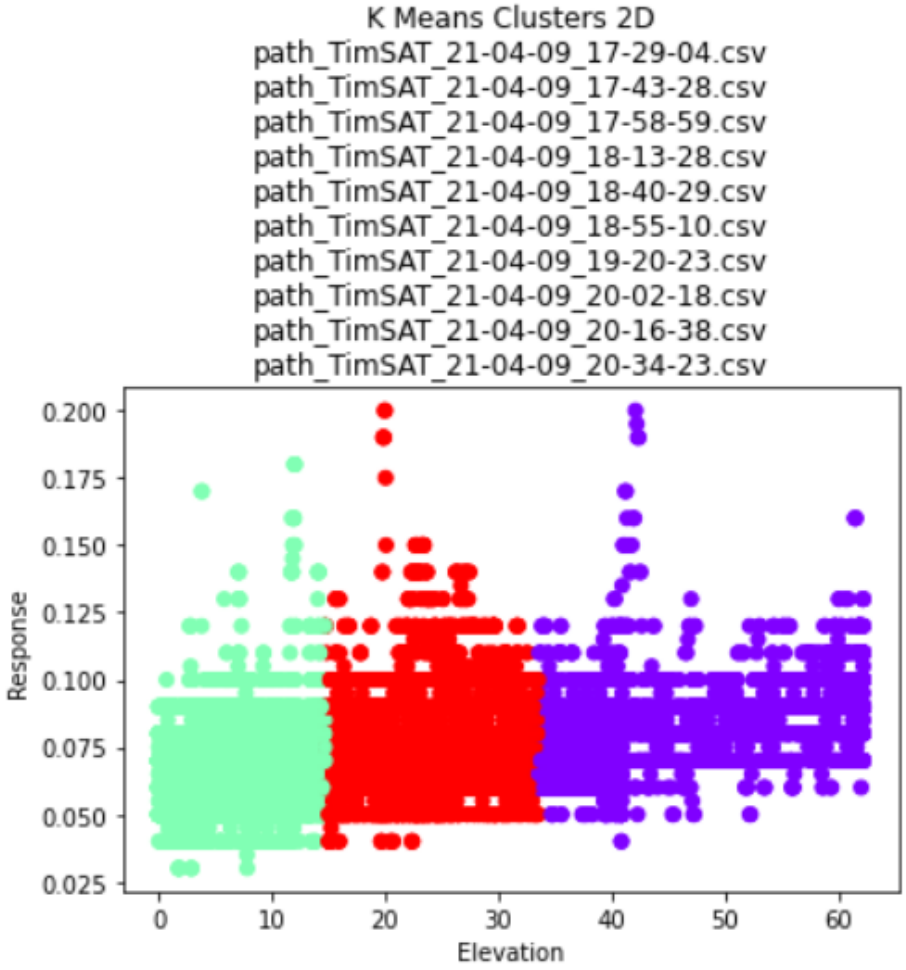


Figure 31. NPS Ground Station Elevation (deg) vs Vibration Response (mm/ sec)

Table 15. Combined Elevation and Azimuth Path Silhouette Coefficients

Number of Clusters	Silhouette Coefficient (Max = 1)
2	0.78
3	0.74
4	0.70
5	0.52
6	0.55
7	0.54
8	0.568
9	0.57

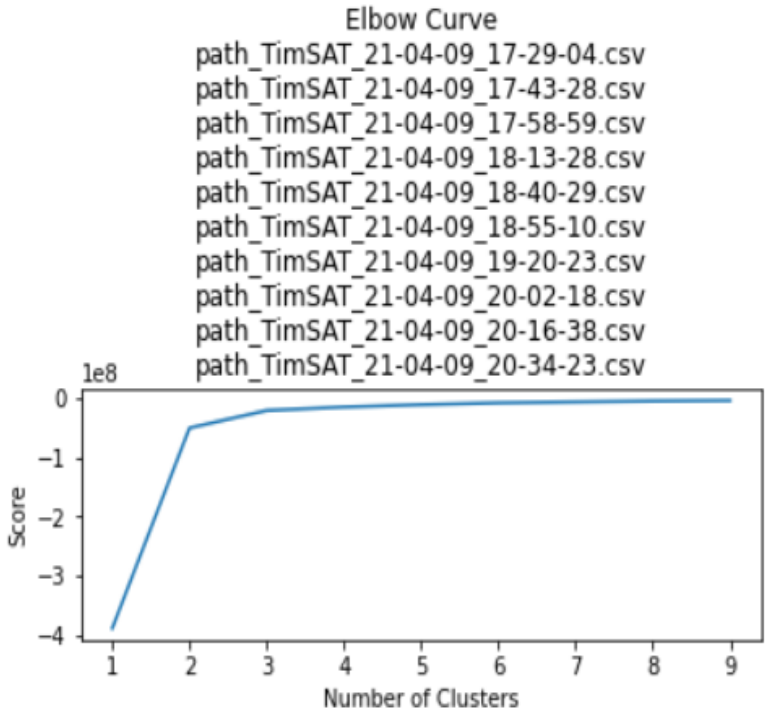


Figure 32. NPS Combined Dish Path and Vibration Data Elbow Curve

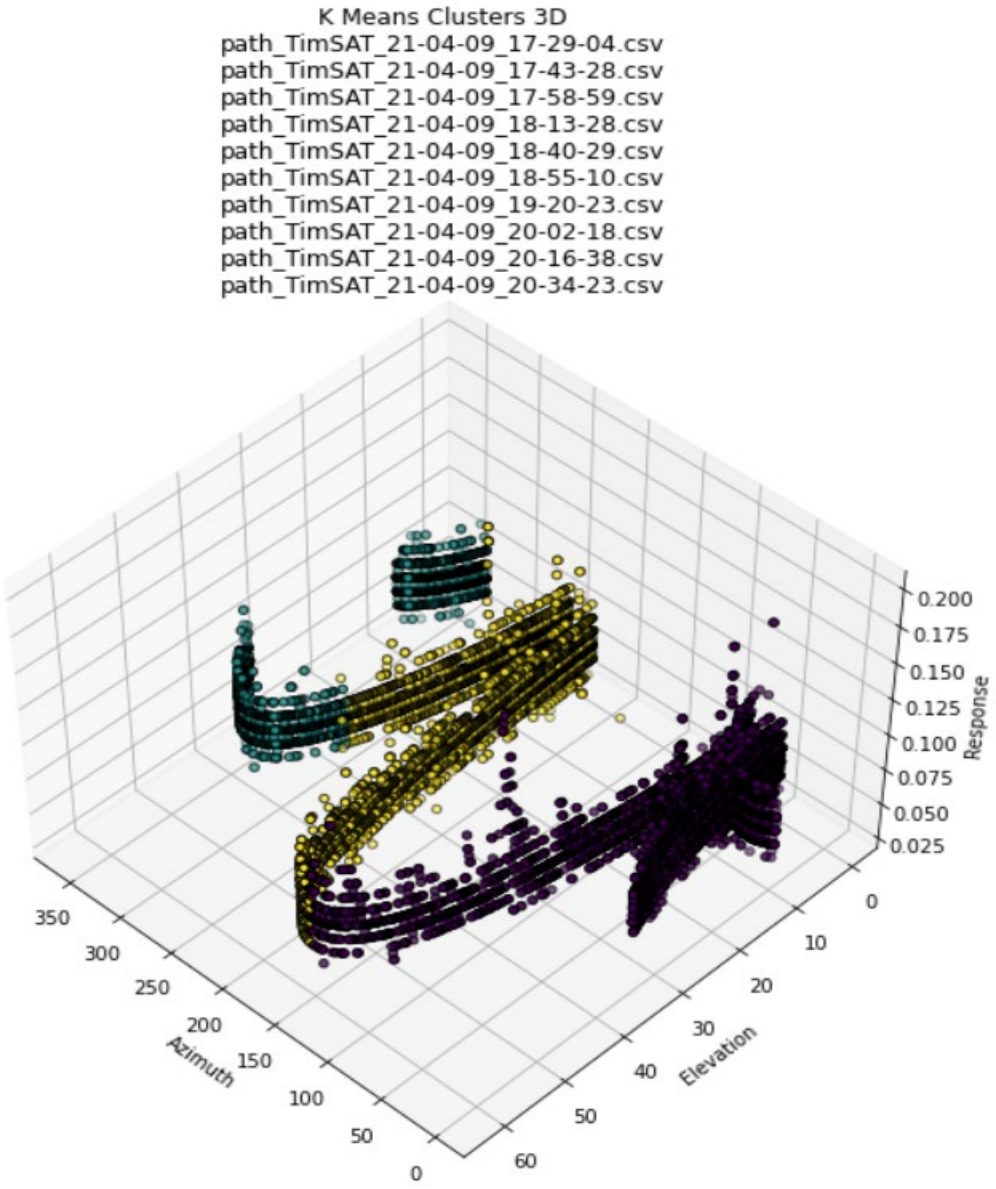


Figure 33. NPS Ground Station Combined Elevation (deg) vs Azimuth (deg) vs Vibration Response (mm/sec)

V. CONCLUSION, DISCUSSION, AND SUGGESTIONS FOR FUTURE WORK

This thesis could not conclusively determine the normal state of the NPS MC3 ground station, but it does establish a launch point for further work. This research answered questions one through five of the initially proposed research questions, but questions six through eleven remain unanswered. However, the answers to questions one through five are not final and require further research.

1. What data are collected by the NPS ground station?
2. How often is the data collected at the NPS ground station?
3. How and where is the data from the NPS ground station stored?
4. Are additional sensors needed to produce a normal state?
5. What other Department of Defense (DOD) systems could this approach be applied?
6. Should a cloud-based solution be explored for flexibility and processing?
7. Can the data from the sensors be analyzed in real-time?
8. What processing capability is needed for real-time machine learning analysis and forecasting?
9. Can anomalies and faults be mitigated autonomously?
10. With improved resiliency and autonomy, can the ground stations become more expeditionary?
11. What are the applications for a more expeditionary ground station?

A. CONCLUSION AND DISCUSSION

The environmental, power, vibration and pointing data are collected at the NPS MC3 ground station at varying levels of fidelity. However, based on the results, more and higher fidelity sensors are required to reliably determine the normal state and any deviation

from that normal state. The power data fidelity level identified in dataset two highlighted that sensor selection criteria are needed to produce significant data. Along with the fidelity, the sample rate of data points needs further examination to promote significant data but limit marginal data that only expends storage space and processing power during analysis. The placement of the sensors on the ground station can drastically change the data collected, especially for the temperature and vibration sensors. A vibration sensor located on the moving dish will likely produce a more significant response than one placed on the inside dome wall. However, that vibration sensor on the dome wall may indicate a systemic problem in the ground station when analyzed with the other collected data points. The placement location of sensors is another aspect of data collection that needs further research and testing.

The early testing of the Python analysis code revealed a data storage and formatting challenge. The data used for analysis was exported from the respective data collection source to CSV files and then manually transferred via physical media to a separate computer for formatting and analysis. The use of physical media to transfer data was time-consuming and did not allow for rapid acquisition of new data for analysis, especially in a constrained COVID-19 environment. A network-based data export capability would greatly benefit further research.

Varying data and time formats from each data source required additional functions to standardize the data before analysis. Standardization of the data format should be applied before entry into a network-based database. Offloading the data formatting from the analysis and future monitoring applications would free up processing power for analysis.

This methodology can be applied to any system on which data samples from multiple sensors can be collected for a specified period of time. The longer the period of time and the more data samples collected, the more likely the normal state model generated by the learning algorithm will be accurate. Iverson stated that IMS could be used in any system with these data collection capabilities (Iverson, 2004). With the decreasing cost of computing power and availability of low-cost, high-quality sensors, monitoring using unsupervised machine learning has wide-ranging potential.

B. FUTURE WORK

Multiple topics for future work are proposed throughout the conclusion and discussion. Figure 34 illustrates a roadmap for suggested future work. The green arrows and text represent the most promising work for the advancement of unsupervised machine learning in the MC3 network, determining the normal state of the NPS ground station, and implementing a near-real-time health monitoring system. Cloud/network-based integration would provide versatility and responsiveness to the research and monitoring. An increase in the number of data sensors and data collected will help build a broad base to support the normal state determination. Once the normal state is determined and validated, the implementation of the monitoring application will close the loop and provide that near-real-time health monitoring system.

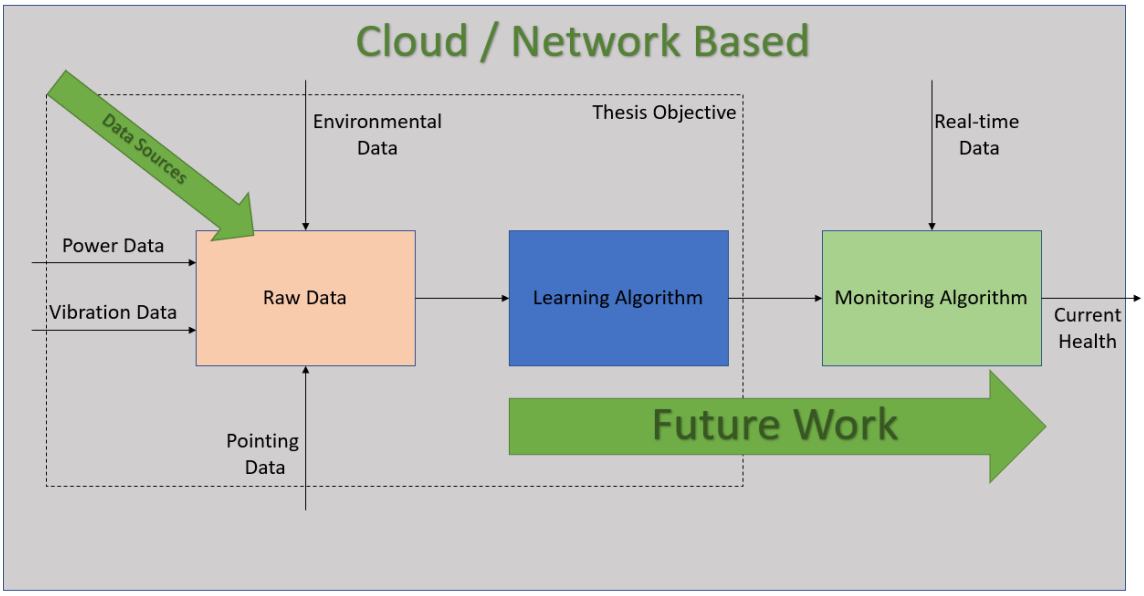


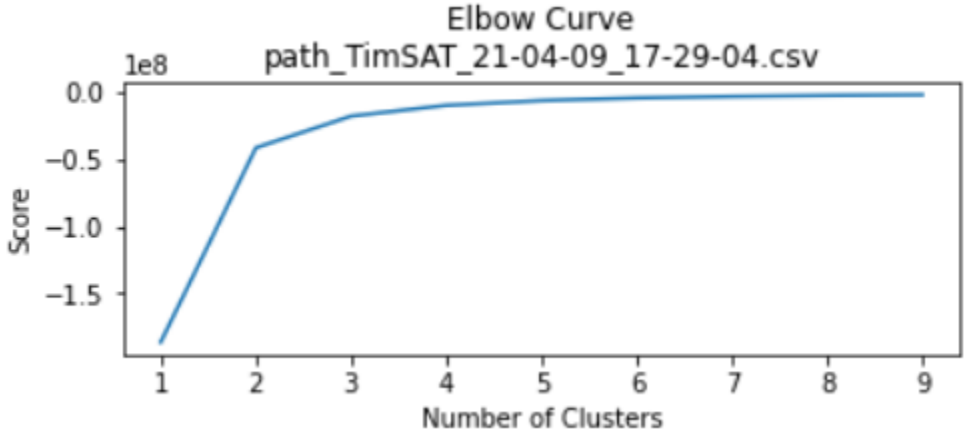
Figure 34. Future Work Roadmap

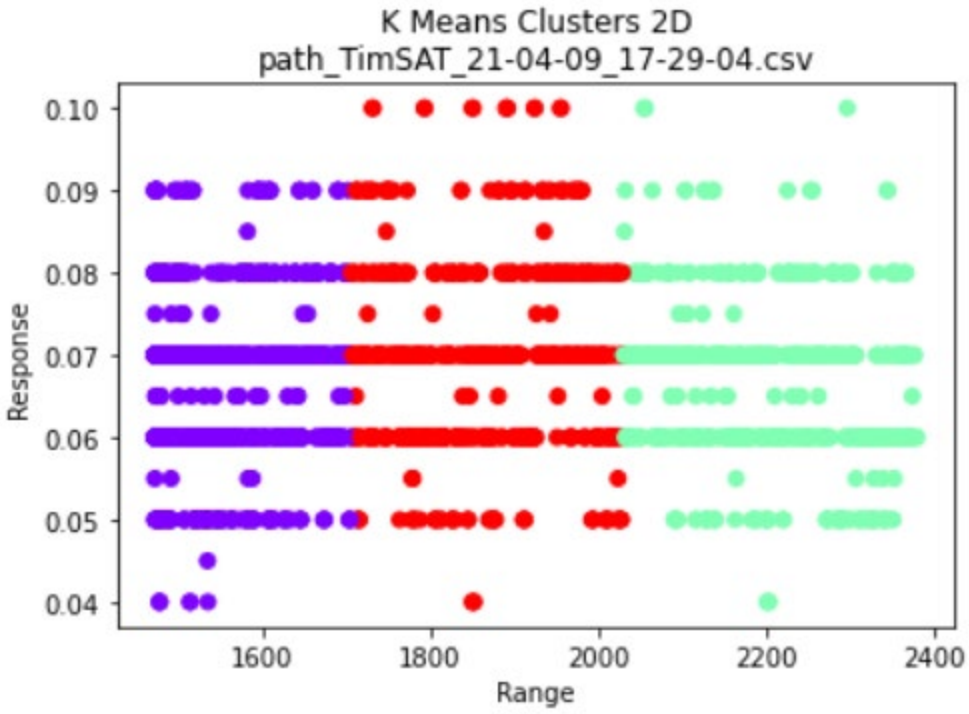
THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. RESULTS

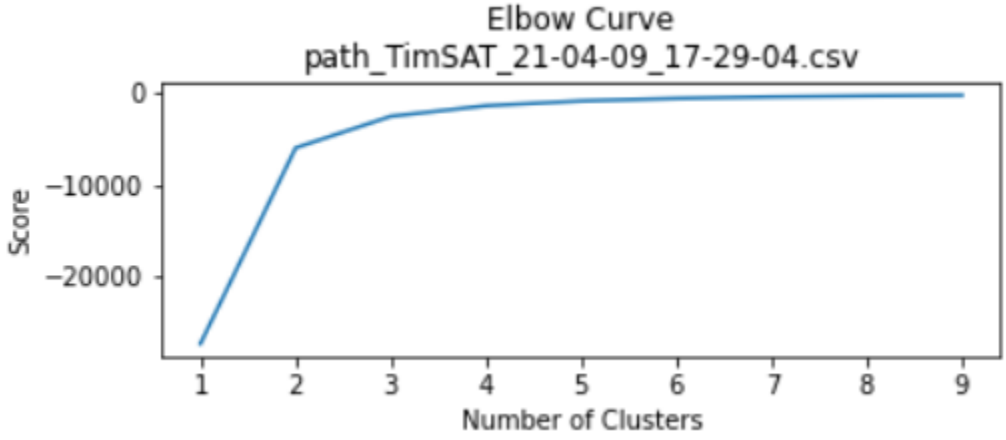
Appendix A includes all the raw result outputs from dataset three. Additional variations of these results can be generated by modifying the python analysis code in Appendix B.

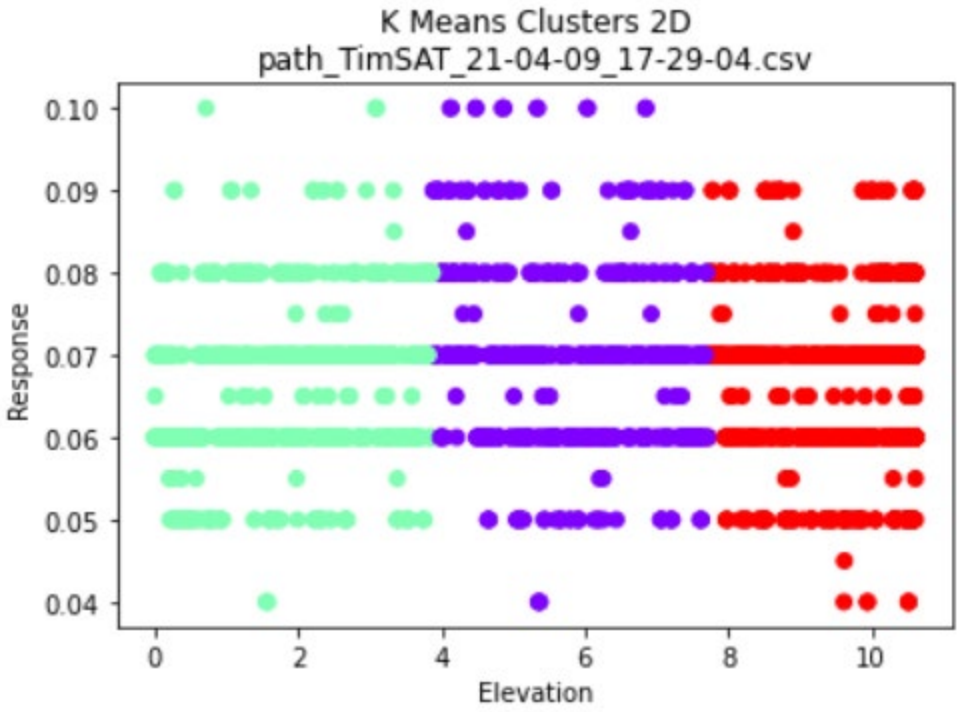
For n_clusters=2, The Silhouette Coefficient is 0.6660936911262124
For n_clusters=3, The Silhouette Coefficient is 0.6279133470182224
For n_clusters=4, The Silhouette Coefficient is 0.6091073585013047
For n_clusters=5, The Silhouette Coefficient is 0.5940793265895957
For n_clusters=6, The Silhouette Coefficient is 0.5872856662005118
For n_clusters=7, The Silhouette Coefficient is 0.5786171048411128
For n_clusters=8, The Silhouette Coefficient is 0.5730788725975419
For n_clusters=9, The Silhouette Coefficient is 0.5671866795844146



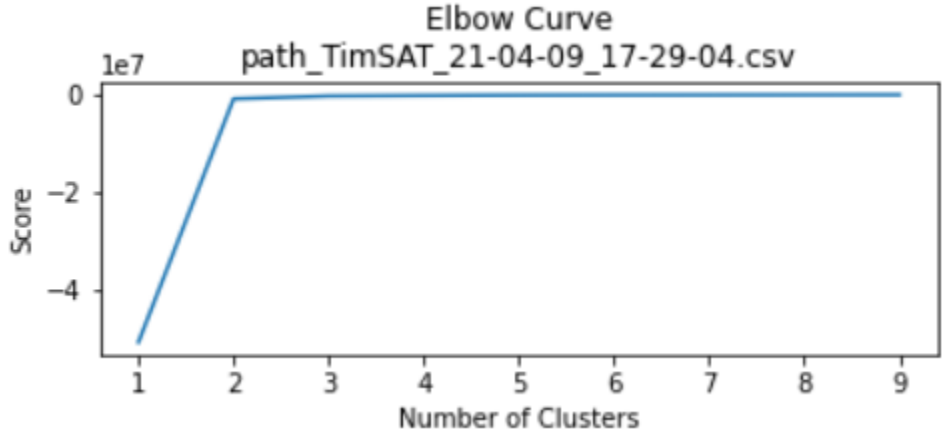


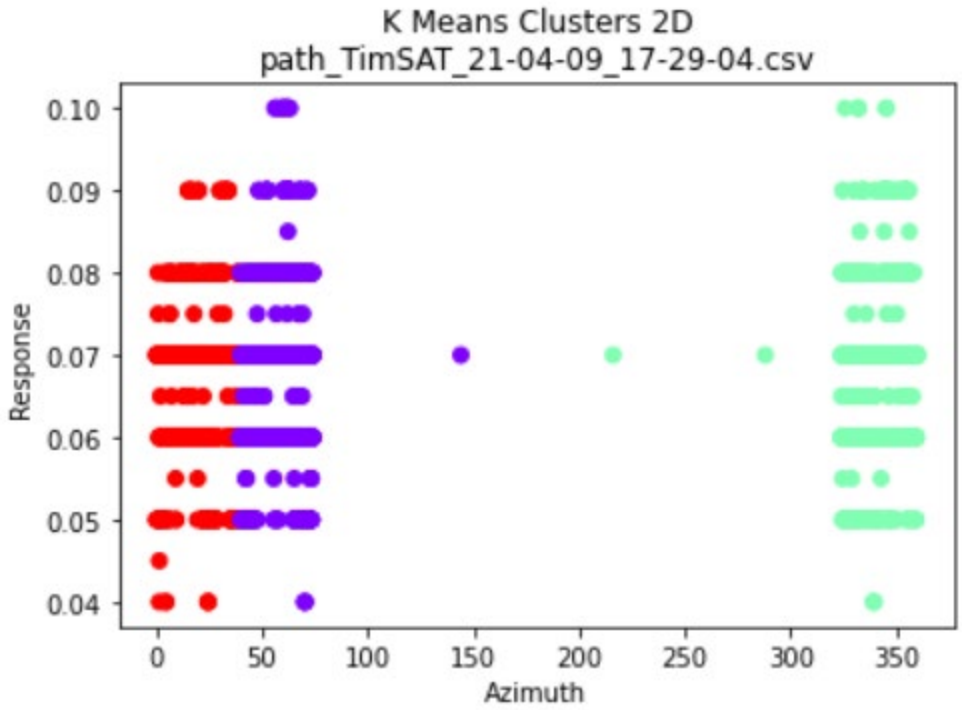
For n_clusters=2, The Silhouette Coefficient is 0.6600310822405209
For n_clusters=3, The Silhouette Coefficient is 0.6231704736783132
For n_clusters=4, The Silhouette Coefficient is 0.6033003251427791
For n_clusters=5, The Silhouette Coefficient is 0.5901506341670502
For n_clusters=6, The Silhouette Coefficient is 0.5802265075087698
For n_clusters=7, The Silhouette Coefficient is 0.574861637273819
For n_clusters=8, The Silhouette Coefficient is 0.5686836146525064
For n_clusters=9, The Silhouette Coefficient is 0.5643451683871253



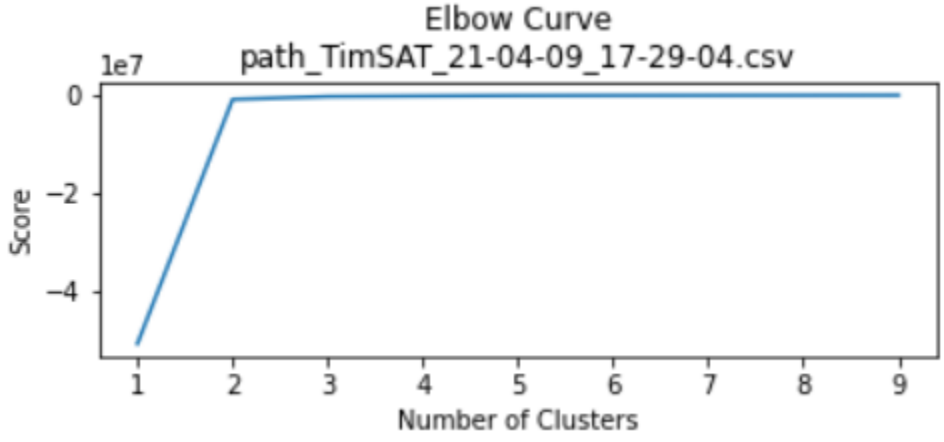


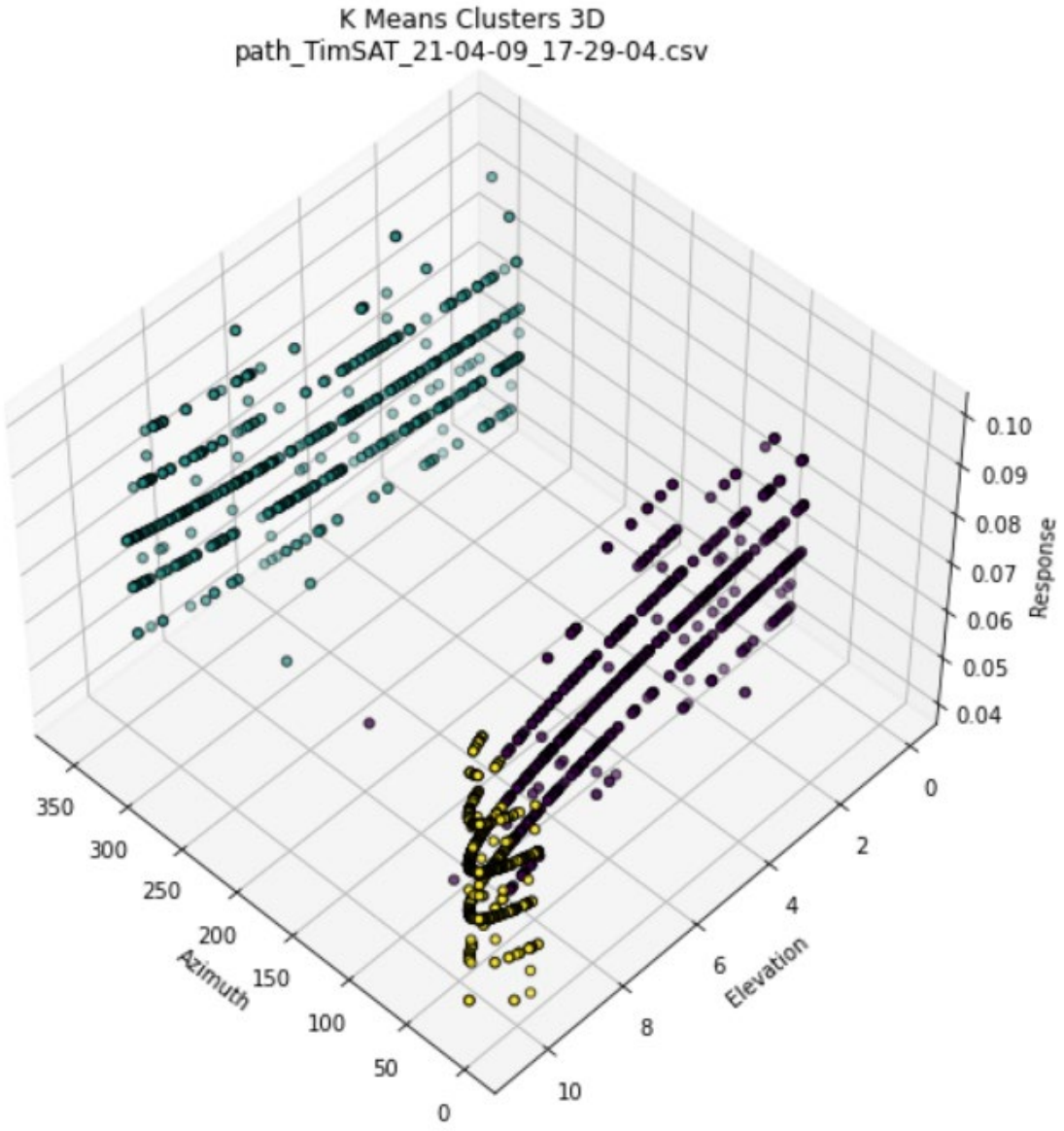
For n_clusters=2, The Silhouette Coefficient is 0.9305612273953091
For n_clusters=3, The Silhouette Coefficient is 0.7639762604274862
For n_clusters=4, The Silhouette Coefficient is 0.7363590454594308
For n_clusters=5, The Silhouette Coefficient is 0.6138137458251015
For n_clusters=6, The Silhouette Coefficient is 0.5989778680726051
For n_clusters=7, The Silhouette Coefficient is 0.5884904794049439
For n_clusters=8, The Silhouette Coefficient is 0.5907709404151086
For n_clusters=9, The Silhouette Coefficient is 0.5755226230599041



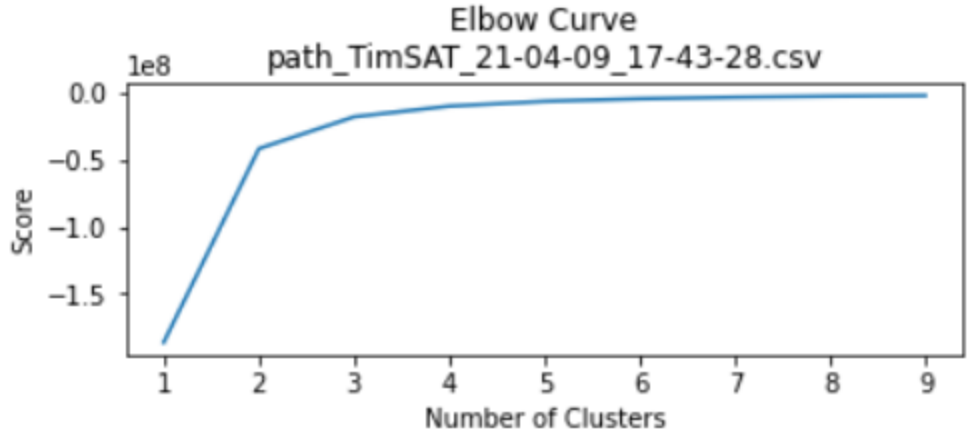


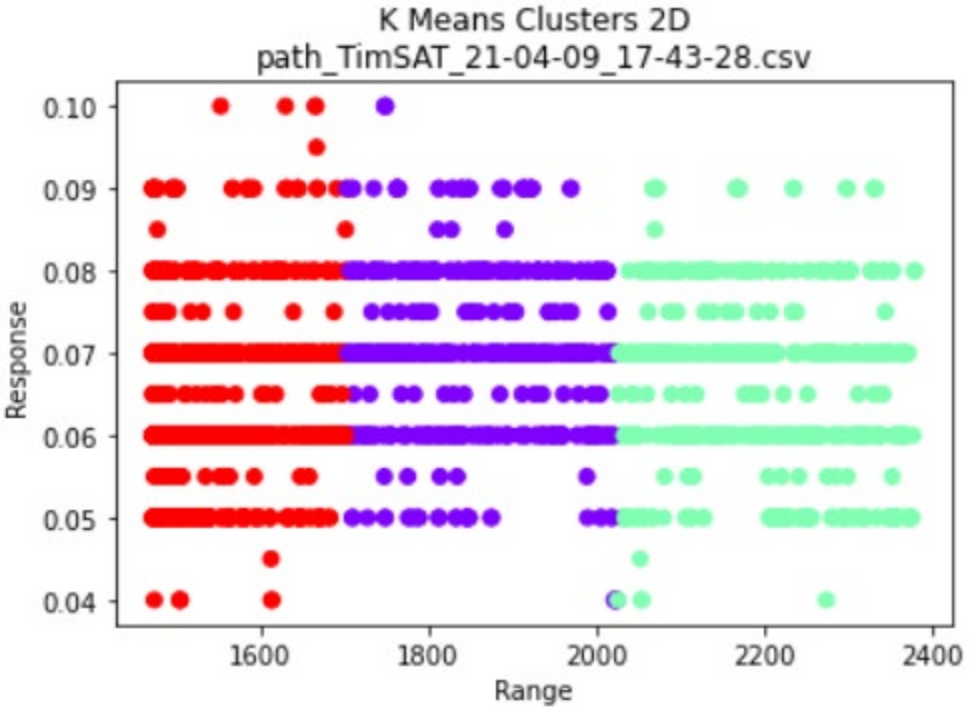
For n_clusters=2, The Silhouette Coefficient is 0.9292253959073893
For n_clusters=3, The Silhouette Coefficient is 0.7612353006210434
For n_clusters=4, The Silhouette Coefficient is 0.7334715614067241
For n_clusters=5, The Silhouette Coefficient is 0.6122348063841742
For n_clusters=6, The Silhouette Coefficient is 0.597802012810985
For n_clusters=7, The Silhouette Coefficient is 0.5882641785256938
For n_clusters=8, The Silhouette Coefficient is 0.5917733115653747
For n_clusters=9, The Silhouette Coefficient is 0.5748919562881005



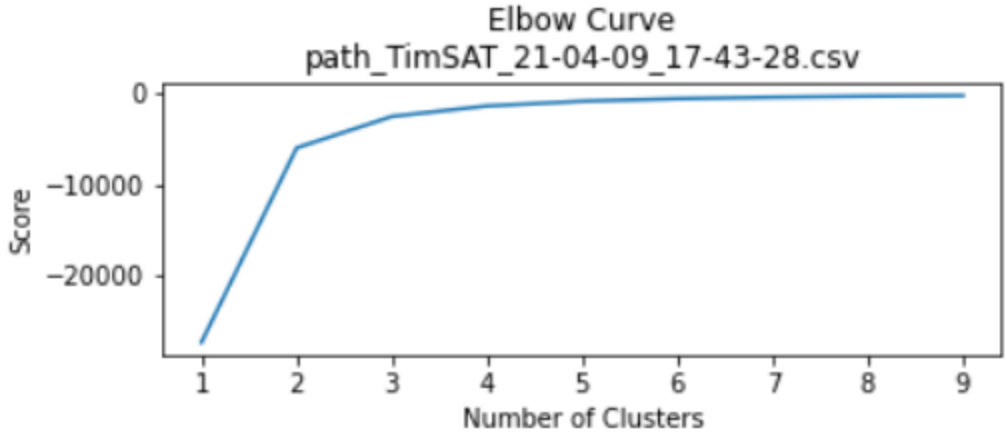


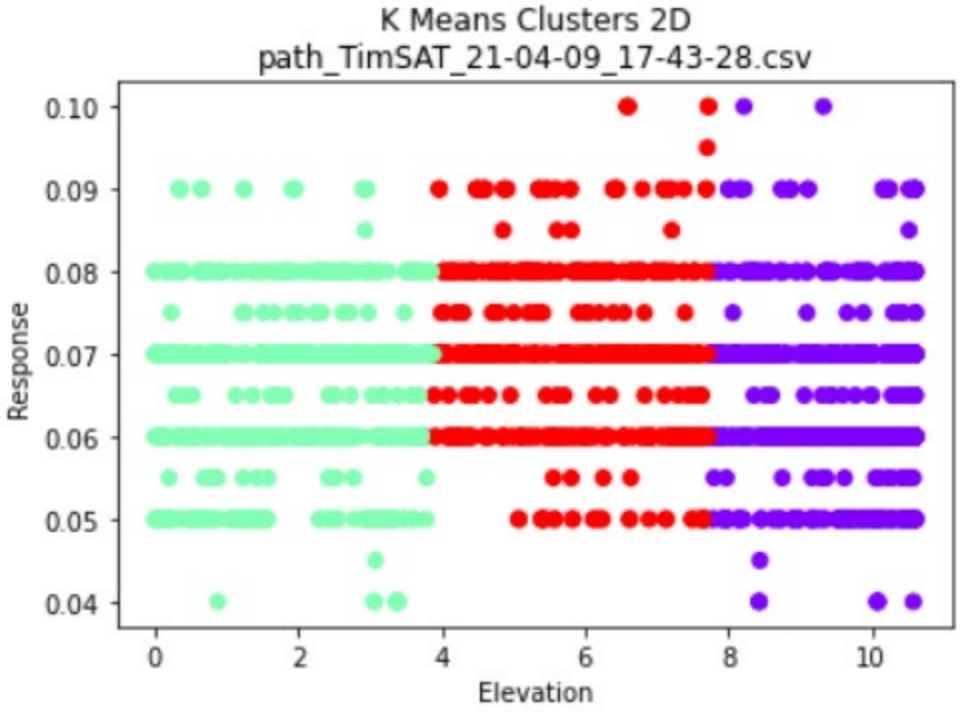
For n_clusters=2, The Silhouette Coefficient is 0.665028441965702
For n_clusters=3, The Silhouette Coefficient is 0.6281452510944467
For n_clusters=4, The Silhouette Coefficient is 0.6077864381443278
For n_clusters=5, The Silhouette Coefficient is 0.5941933357864291
For n_clusters=6, The Silhouette Coefficient is 0.5836274148678351
For n_clusters=7, The Silhouette Coefficient is 0.5777091932687017
For n_clusters=8, The Silhouette Coefficient is 0.5726587713244643
For n_clusters=9, The Silhouette Coefficient is 0.5671256414343041



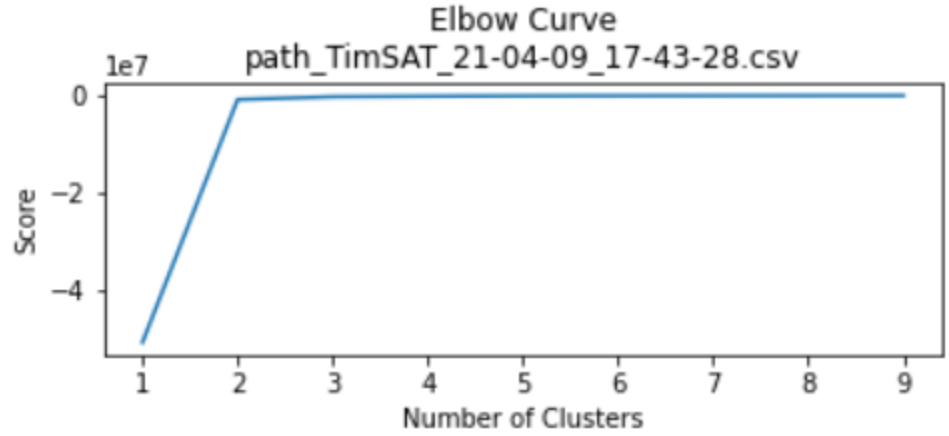


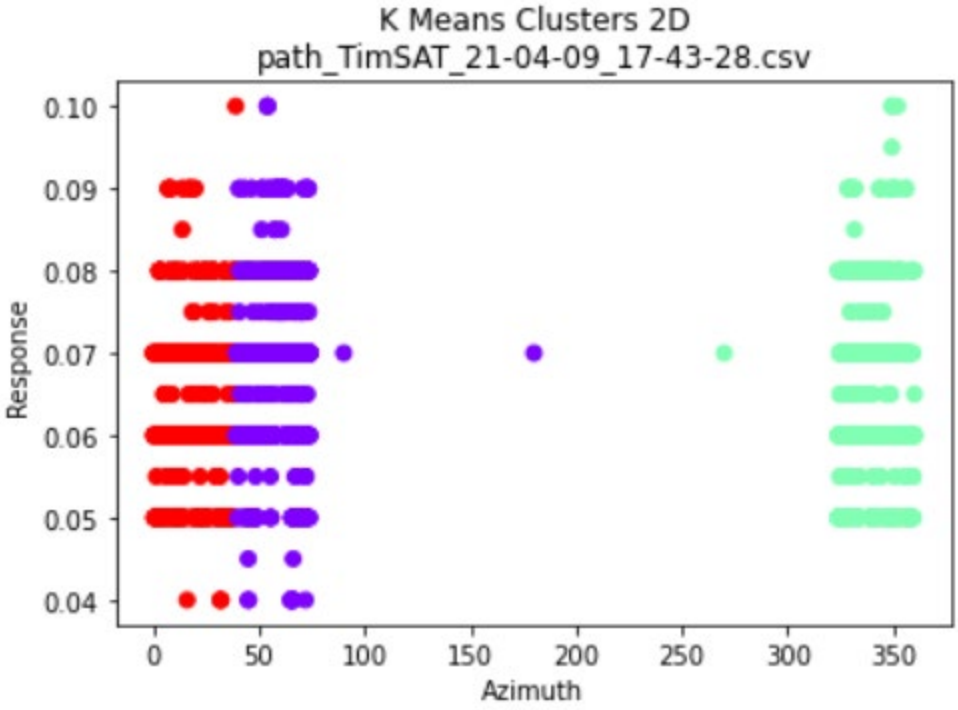
For n_clusters=2, The Silhouette Coefficient is 0.6596284793088887
For n_clusters=3, The Silhouette Coefficient is 0.623280784536207
For n_clusters=4, The Silhouette Coefficient is 0.6032380862657333
For n_clusters=5, The Silhouette Coefficient is 0.5902736022226094
For n_clusters=6, The Silhouette Coefficient is 0.5812722515568808
For n_clusters=7, The Silhouette Coefficient is 0.5725771628347588
For n_clusters=8, The Silhouette Coefficient is 0.569184564832598
For n_clusters=9, The Silhouette Coefficient is 0.5630194590714188



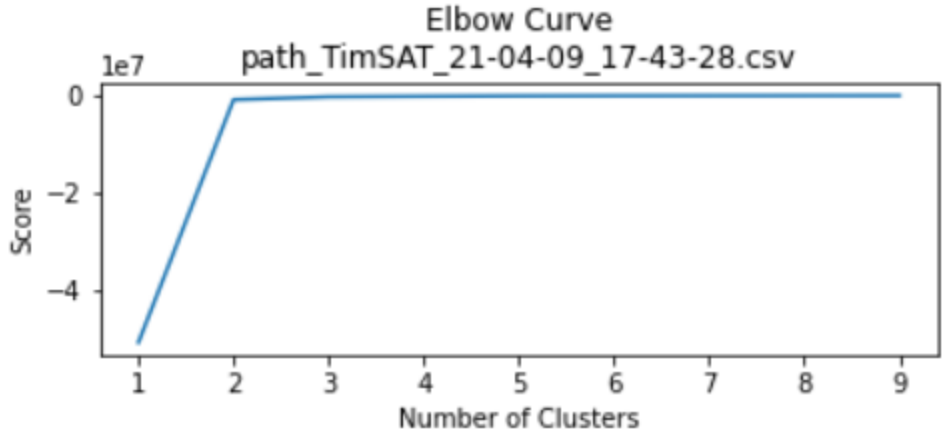


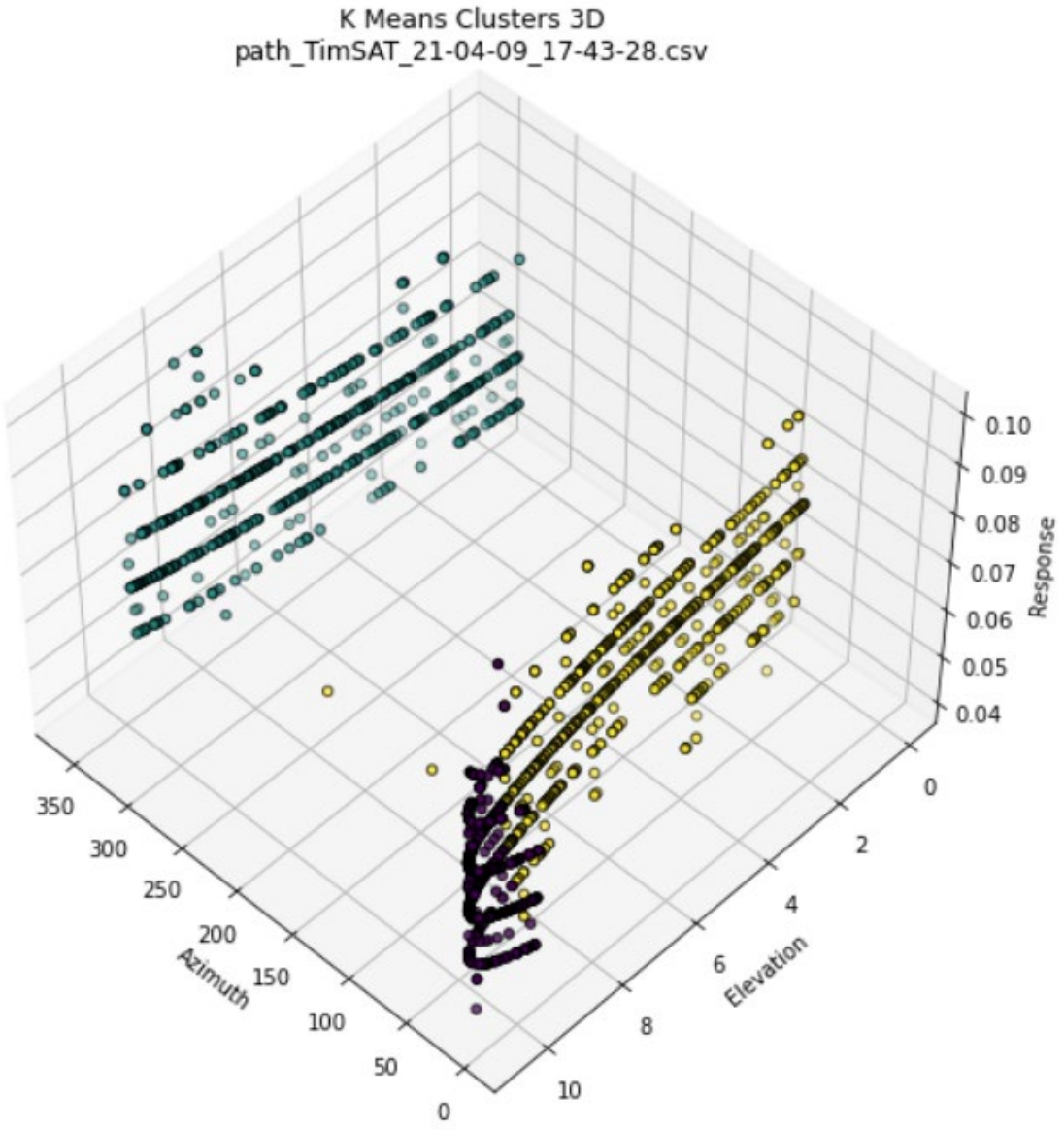
For n_clusters=2, The Silhouette Coefficient is 0.9307734061636478
For n_clusters=3, The Silhouette Coefficient is 0.7636280259325146
For n_clusters=4, The Silhouette Coefficient is 0.7366169306001592
For n_clusters=5, The Silhouette Coefficient is 0.6133865493748198
For n_clusters=6, The Silhouette Coefficient is 0.5985322248404447
For n_clusters=7, The Silhouette Coefficient is 0.5898694940613574
For n_clusters=8, The Silhouette Coefficient is 0.5736916946964169
For n_clusters=9, The Silhouette Coefficient is 0.576196209390065



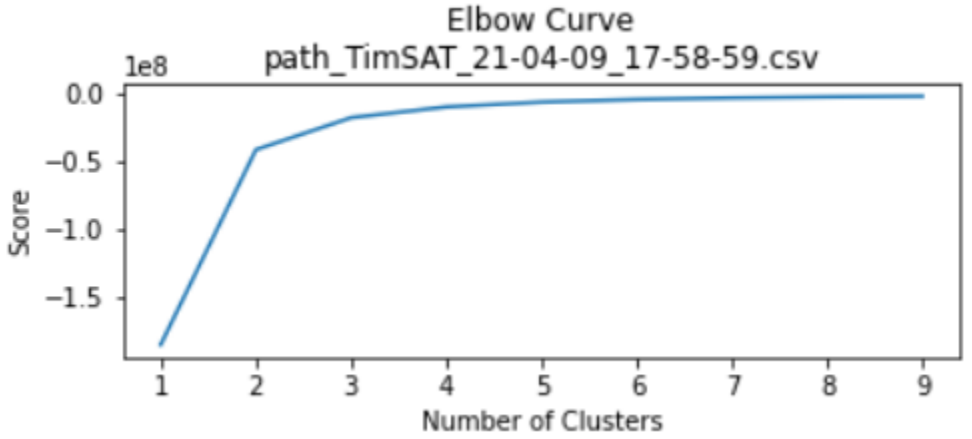


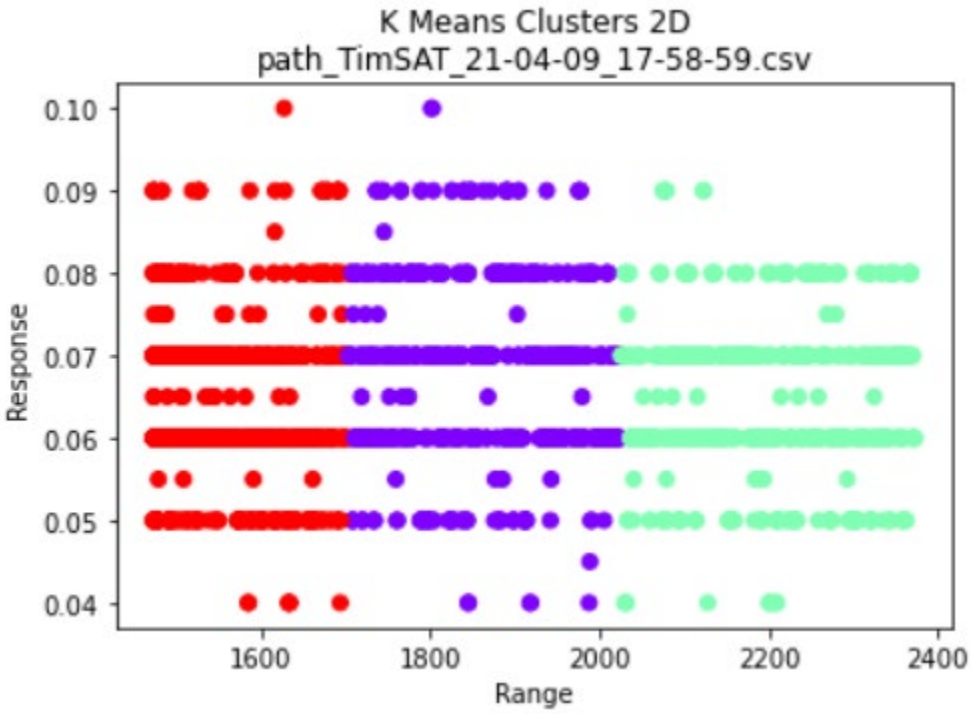
For n_clusters=2, The Silhouette Coefficient is 0.9294416164914073
For n_clusters=3, The Silhouette Coefficient is 0.7611579943345965
For n_clusters=4, The Silhouette Coefficient is 0.7343787305856271
For n_clusters=5, The Silhouette Coefficient is 0.6128892375727621
For n_clusters=6, The Silhouette Coefficient is 0.5967672535898704
For n_clusters=7, The Silhouette Coefficient is 0.5889706358231183
For n_clusters=8, The Silhouette Coefficient is 0.5708569873711213
For n_clusters=9, The Silhouette Coefficient is 0.5739781681125123


Elbow Curve
path_TimSAT_21-04-09_17-43-28.csv
1e7
0
−2
−4
Score
1
2
3
4
5
6
7
8
9
Number of Clusters


K Means Clusters 3D
path_TimSAT_21-04-09_17-43-28.csv
0.10
0.09
0.08
0.07
0.06
0.05
0.04
Response
350
300
250
200
150
100
50
0
Azimuth
0
2
4
6
8
10
Elevation

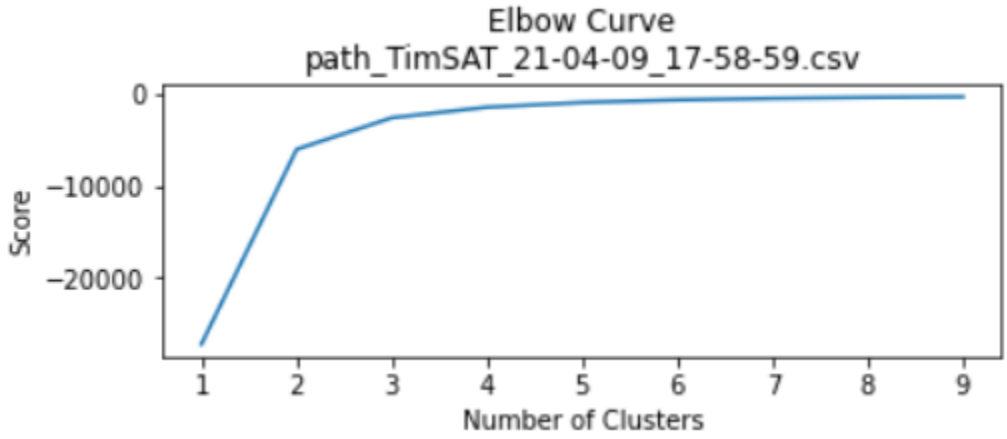
For n_clusters=2, The Silhouette Coefficient is 0.6649594268946978
For n_clusters=3, The Silhouette Coefficient is 0.6279594312467356
For n_clusters=4, The Silhouette Coefficient is 0.6078085054792793
For n_clusters=5, The Silhouette Coefficient is 0.5946445657771765
For n_clusters=6, The Silhouette Coefficient is 0.5853252364681679
For n_clusters=7, The Silhouette Coefficient is 0.5794543874787849
For n_clusters=8, The Silhouette Coefficient is 0.5713582739641703
For n_clusters=9, The Silhouette Coefficient is 0.566692552264133

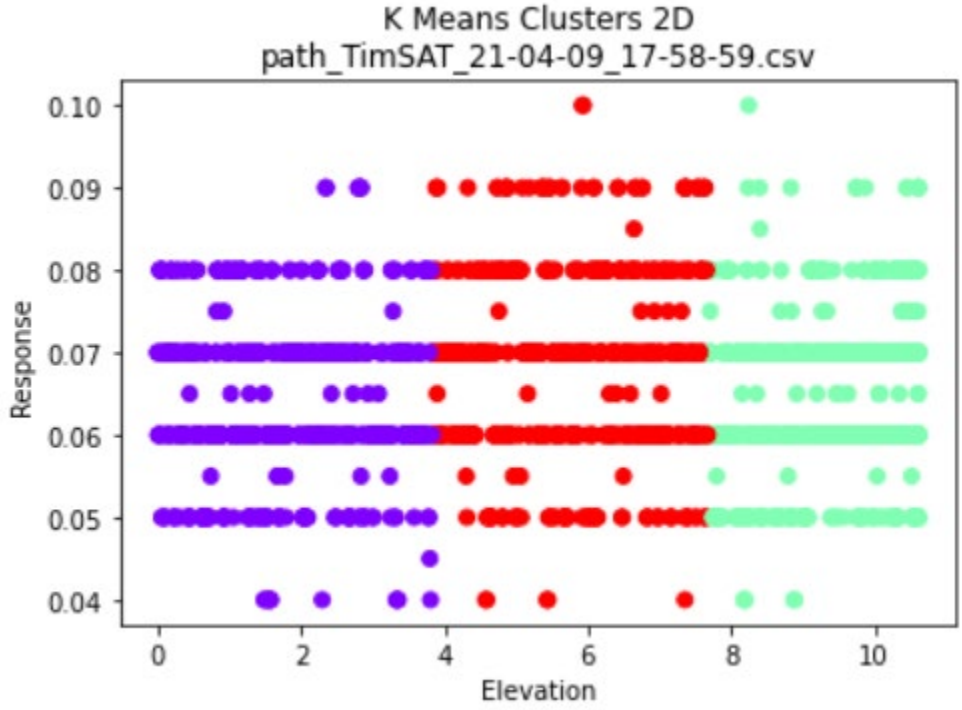




For n_clusters=2, The Silhouette Coefficient is 0.6595099476259819
For n_clusters=3, The Silhouette Coefficient is 0.6232536032014647
For n_clusters=4, The Silhouette Coefficient is 0.6030704428552417

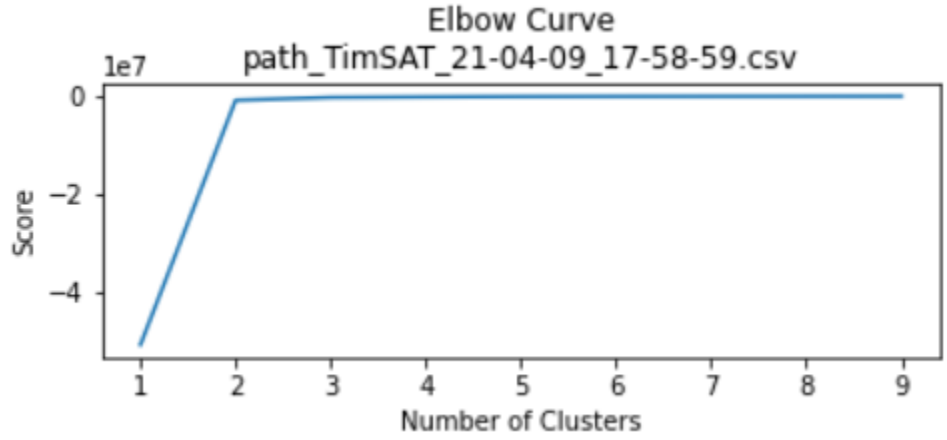
For n_clusters=5, The Silhouette Coefficient is 0.5898353059426649
For n_clusters=6, The Silhouette Coefficient is 0.5810052111207067
For n_clusters=7, The Silhouette Coefficient is 0.5732257879724694
For n_clusters=8, The Silhouette Coefficient is 0.5674194019493447
For n_clusters=9, The Silhouette Coefficient is 0.5651870756035587

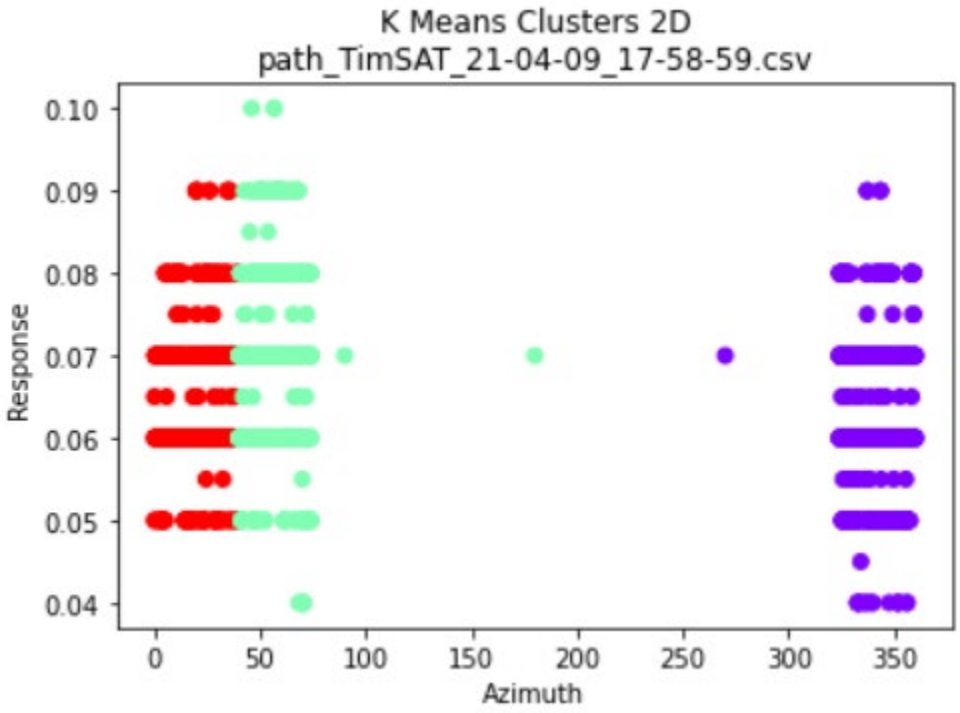




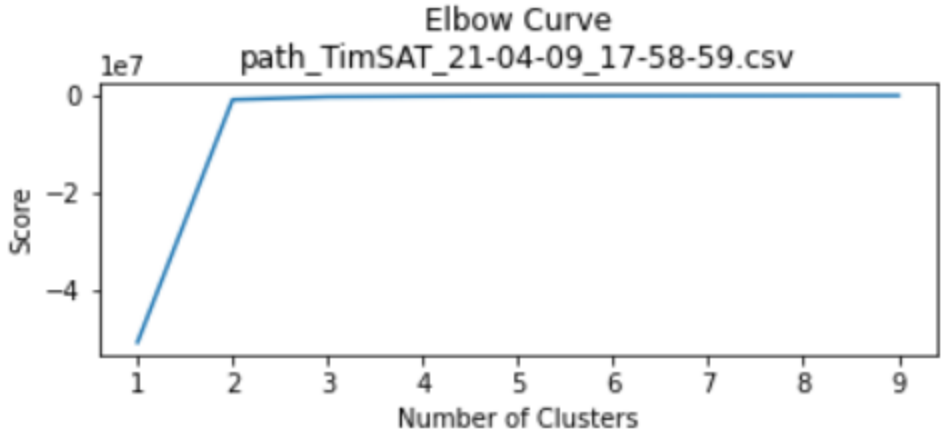
For n_clusters=2, The Silhouette Coefficient is 0.9307818786297365
For n_clusters=3, The Silhouette Coefficient is 0.7638584193202802
For n_clusters=4, The Silhouette Coefficient is 0.7357828499278102
For n_clusters=5, The Silhouette Coefficient is 0.6146221590927551
For n_clusters=6, The Silhouette Coefficient is 0.5994233768565284
For n_clusters=7, The Silhouette Coefficient is 0.5876787913378563
For n_clusters=8, The Silhouette Coefficient is 0.5721038565525433

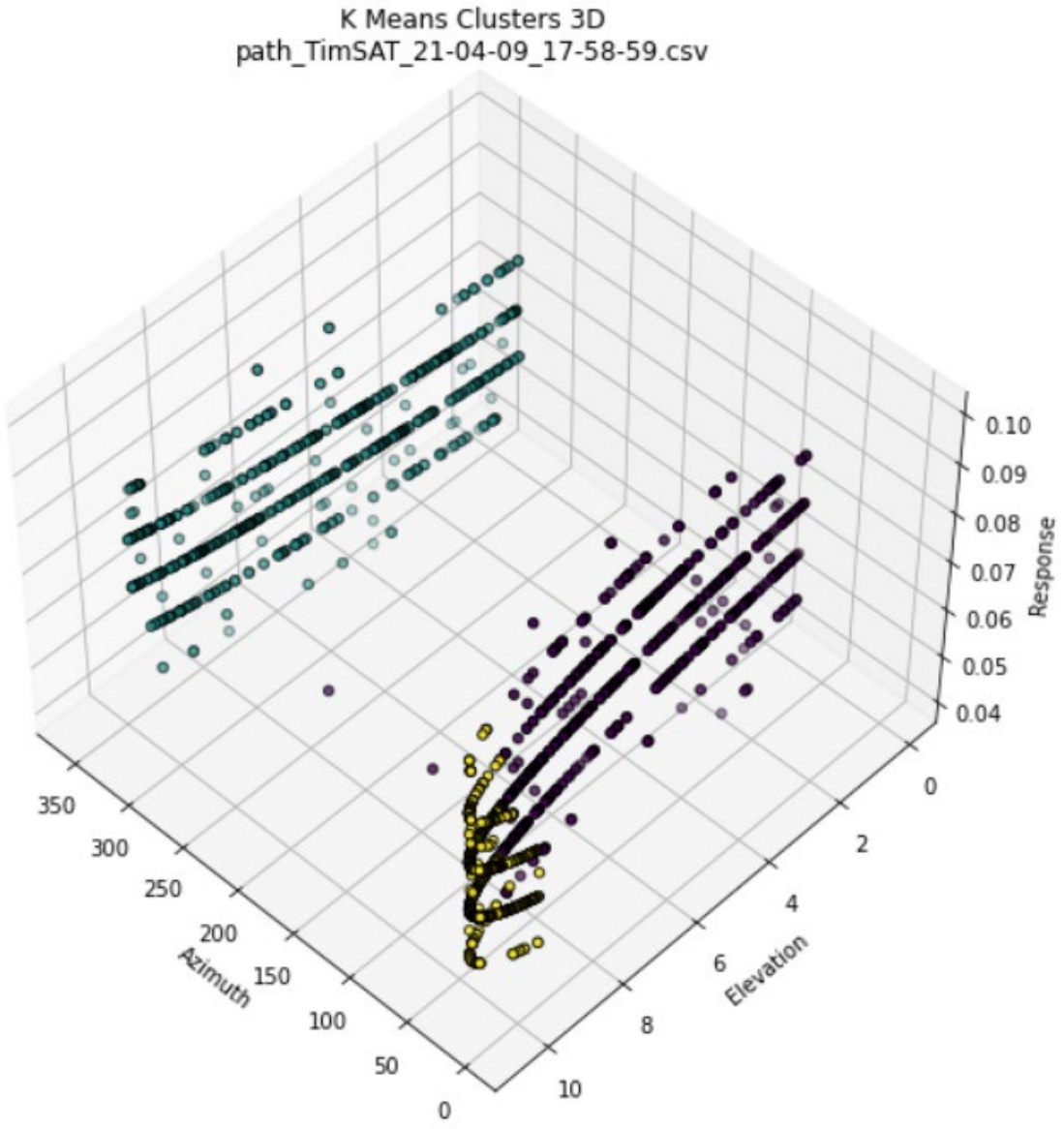
For n_clusters=9, The Silhouette Coefficient is 0.5756819478461006



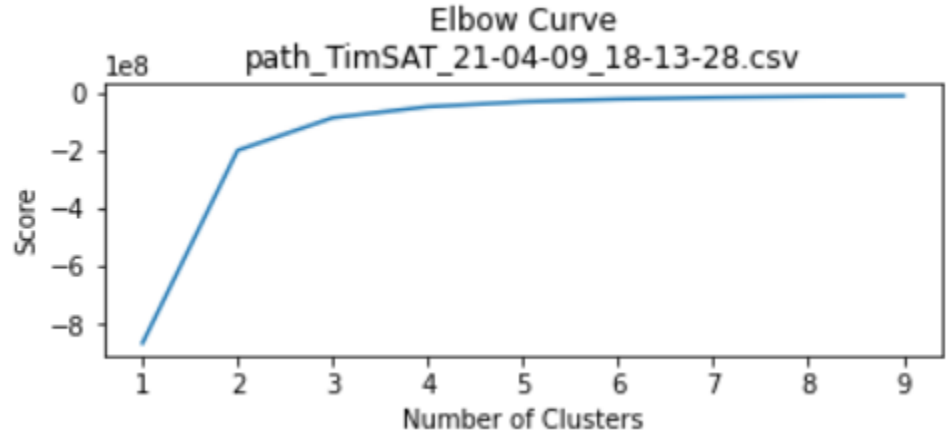


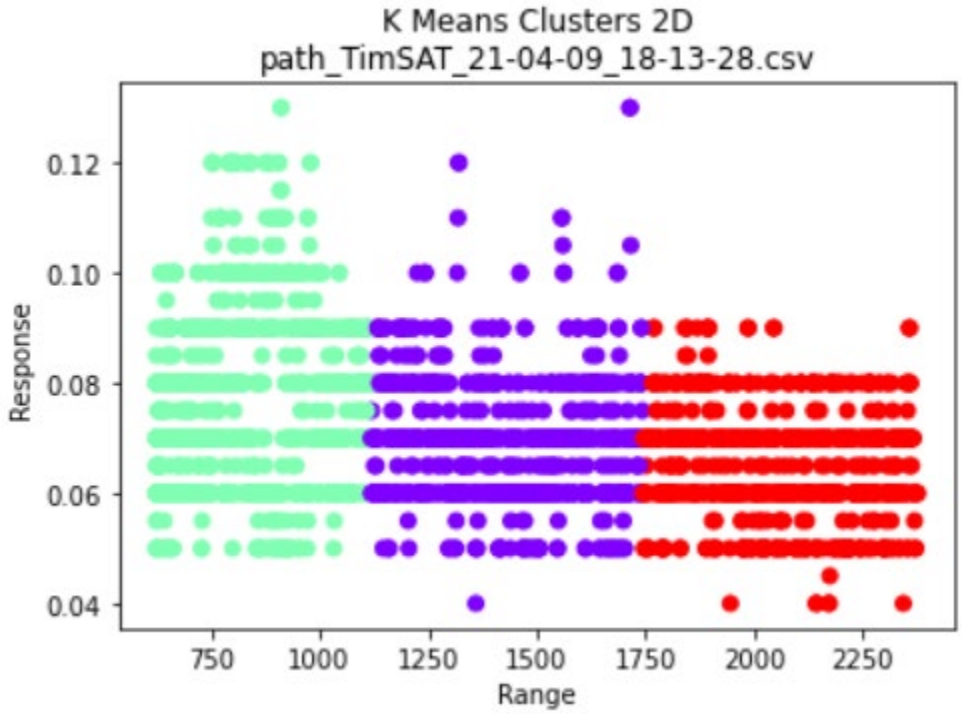
For n_clusters=2, The Silhouette Coefficient is 0.9294536411322791
For n_clusters=3, The Silhouette Coefficient is 0.7609474651598498
For n_clusters=4, The Silhouette Coefficient is 0.7343080863577438
For n_clusters=5, The Silhouette Coefficient is 0.6118570336789476
For n_clusters=6, The Silhouette Coefficient is 0.5969445380392611
For n_clusters=7, The Silhouette Coefficient is 0.5879064903076265
For n_clusters=8, The Silhouette Coefficient is 0.5721213907969821
For n_clusters=9, The Silhouette Coefficient is 0.5744153774946328


Elbow Curve
path_TimSAT_21-04-09_17-58-59.csv
1e7
0
−2
−4
Score
1
2
3
4
5
6
7
8
9
Number of Clusters


K Means Clusters 3D
path_TimSAT_21-04-09_17-58-59.csv
0.10
0.09
0.08
0.07
0.06
0.05
0.04
Response
350
300
250
200
150
100
50
0
Azimuth
0
2
4
6
8
10
Elevation

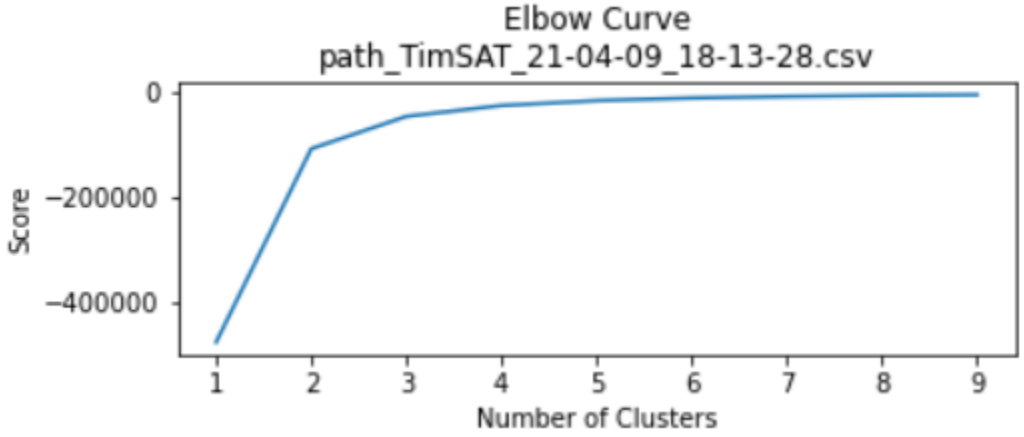
For n_clusters=2, The Silhouette Coefficient is 0.6495626814364096
For n_clusters=3, The Silhouette Coefficient is 0.614257702594421
For n_clusters=4, The Silhouette Coefficient is 0.595823867750173
For n_clusters=5, The Silhouette Coefficient is 0.583889718474442
For n_clusters=6, The Silhouette Coefficient is 0.575988943347529
For n_clusters=7, The Silhouette Coefficient is 0.5685641288837039
For n_clusters=8, The Silhouette Coefficient is 0.5635029379754702
For n_clusters=9, The Silhouette Coefficient is 0.5595565001622446

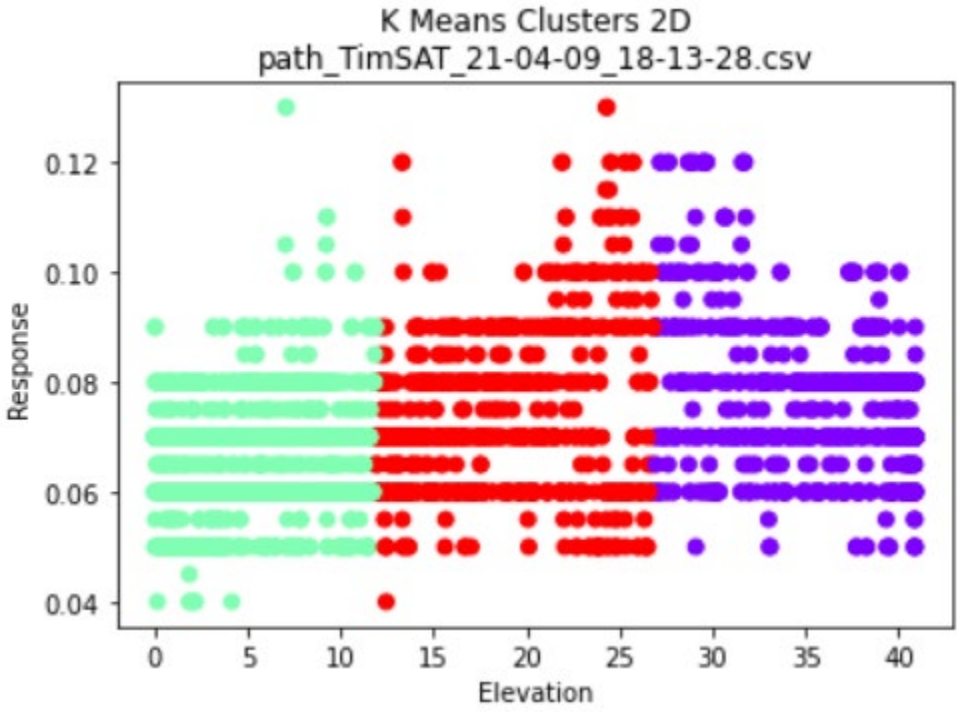




For n_clusters=2, The Silhouette Coefficient is 0.6640797396548737
For n_clusters=3, The Silhouette Coefficient is 0.6177225986192904
For n_clusters=4, The Silhouette Coefficient is 0.5951418553494483

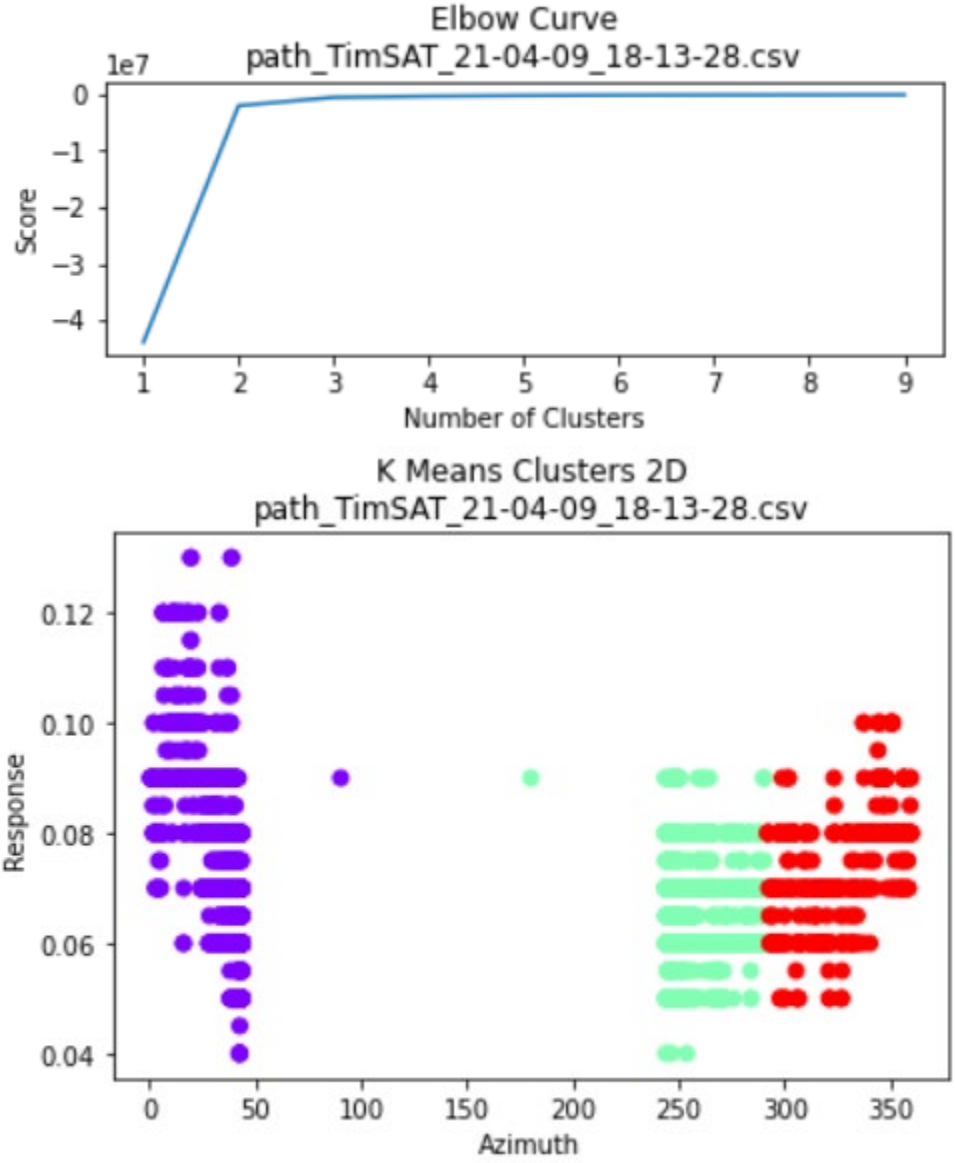
For n_clusters=5, The Silhouette Coefficient is 0.5804941743616523
For n_clusters=6, The Silhouette Coefficient is 0.5711272313928686
For n_clusters=7, The Silhouette Coefficient is 0.5658945014956323
For n_clusters=8, The Silhouette Coefficient is 0.5586135062711107
For n_clusters=9, The Silhouette Coefficient is 0.5567313586776889



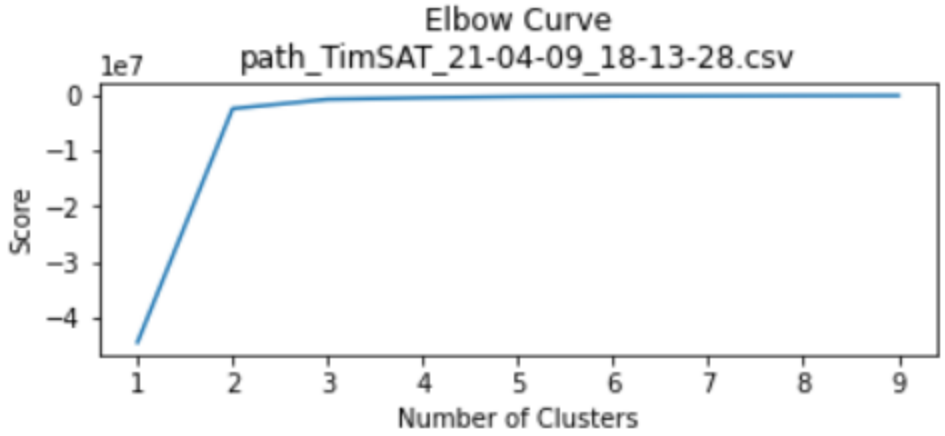


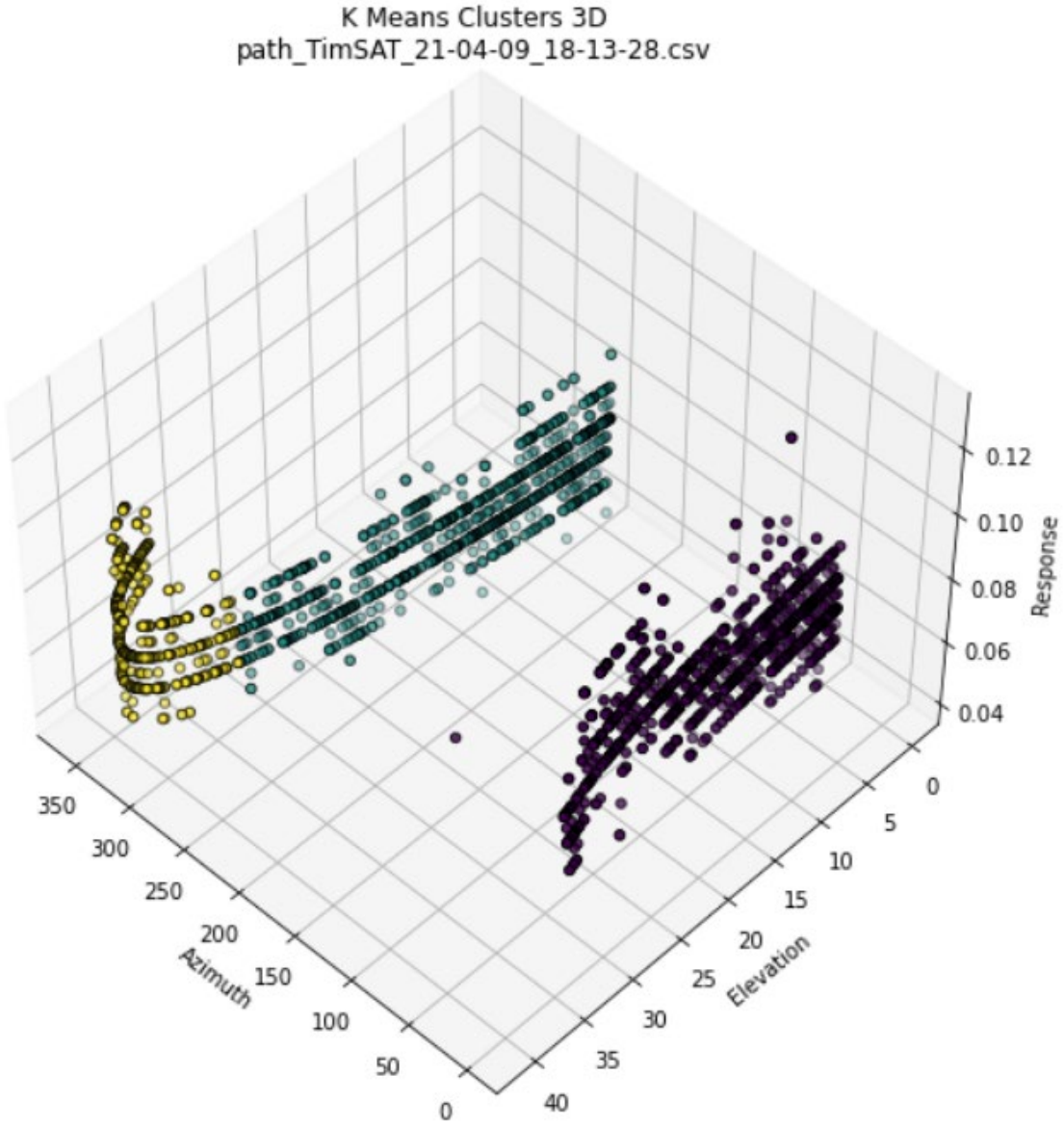
For n_clusters=2, The Silhouette Coefficient is 0.9003138184413342
For n_clusters=3, The Silhouette Coefficient is 0.8308655148767751
For n_clusters=4, The Silhouette Coefficient is 0.792953980067198
For n_clusters=5, The Silhouette Coefficient is 0.6757742993104305
For n_clusters=6, The Silhouette Coefficient is 0.6552933891438999
For n_clusters=7, The Silhouette Coefficient is 0.6370957788651562
For n_clusters=8, The Silhouette Coefficient is 0.6166135755404532

For n_clusters=9, The Silhouette Coefficient is 0.6082867718584035

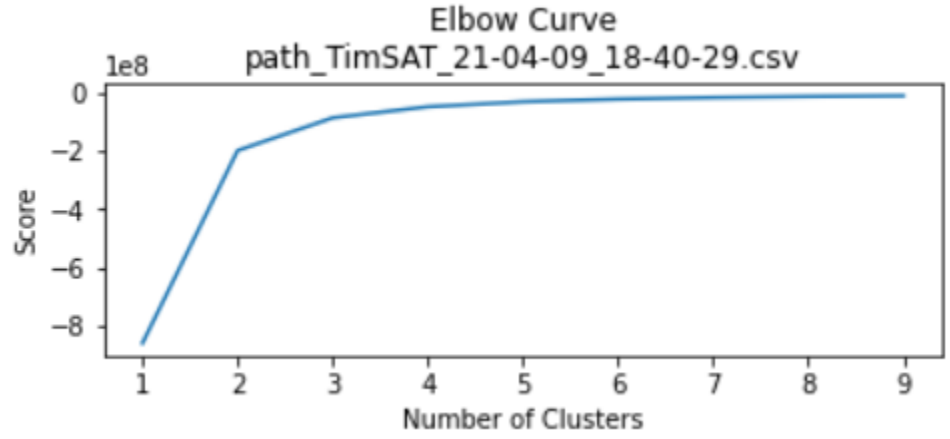


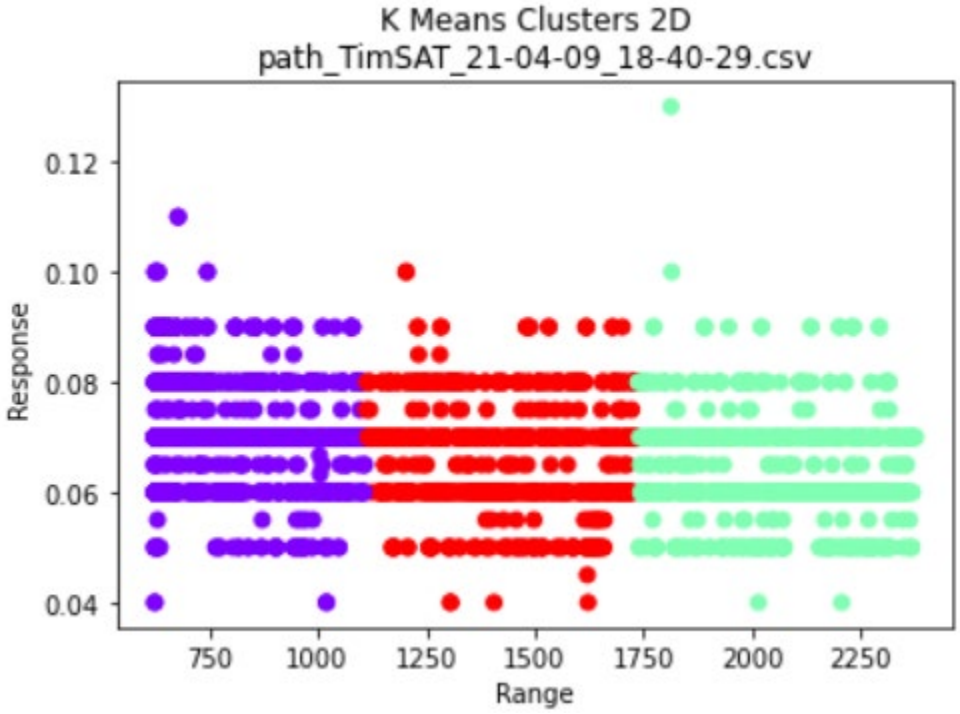
For n_clusters=2, The Silhouette Coefficient is 0.8823687373176661
For n_clusters=3, The Silhouette Coefficient is 0.803348567678576
For n_clusters=4, The Silhouette Coefficient is 0.7679895543774335
For n_clusters=5, The Silhouette Coefficient is 0.6504169000154896
For n_clusters=6, The Silhouette Coefficient is 0.630328760762678
For n_clusters=7, The Silhouette Coefficient is 0.6199485744732294
For n_clusters=8, The Silhouette Coefficient is 0.5985116796974531
For n_clusters=9, The Silhouette Coefficient is 0.5887183134011512


Elbow Curve
path_TimSAT_21-04-09_18-13-28.csv
1e7
0
−1
−2
−3
−4
Score
1
2
3
4
5
6
7
8
9
Number of Clusters


K Means Clusters 3D
path_TimSAT_21-04-09_18-13-28.csv
0.12
0.10
0.08
0.06
0.04
Response
350
300
250
200
150
100
50
0
Azimuth
0
5
10
15
20
25
30
35
40
Elevation

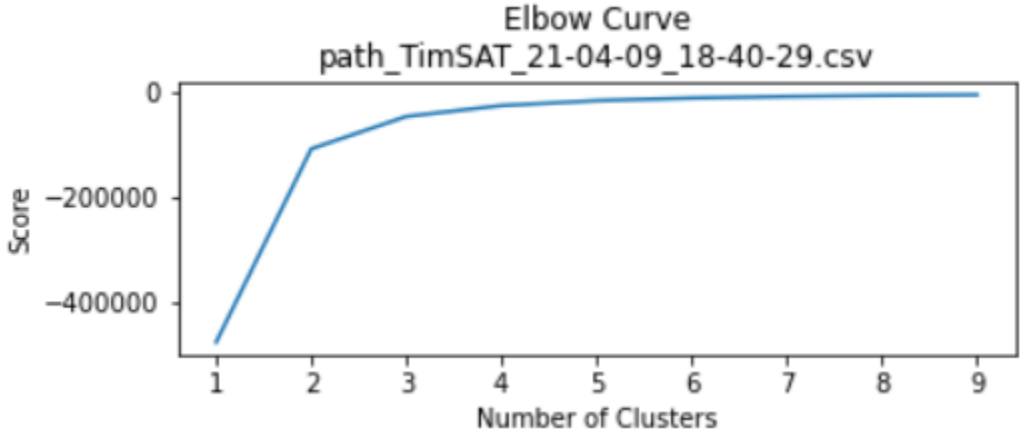
For n_clusters=2, The Silhouette Coefficient is 0.6488043069985987
For n_clusters=3, The Silhouette Coefficient is 0.6140406199550499
For n_clusters=4, The Silhouette Coefficient is 0.5952100288273509
For n_clusters=5, The Silhouette Coefficient is 0.5852398313915677
For n_clusters=6, The Silhouette Coefficient is 0.575785298915239
For n_clusters=7, The Silhouette Coefficient is 0.5685082013760426
For n_clusters=8, The Silhouette Coefficient is 0.5628196186366358
For n_clusters=9, The Silhouette Coefficient is 0.5597525925963832

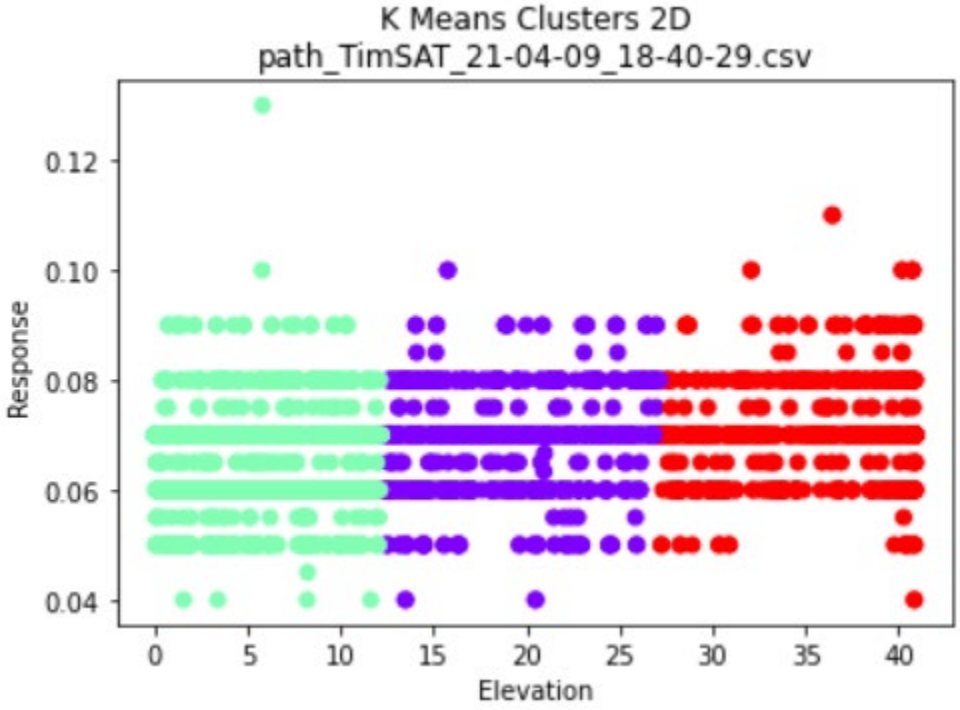




For n_clusters=2, The Silhouette Coefficient is 0.6639769292140553
For n_clusters=3, The Silhouette Coefficient is 0.6175632714098288
For n_clusters=4, The Silhouette Coefficient is 0.5943567656387757

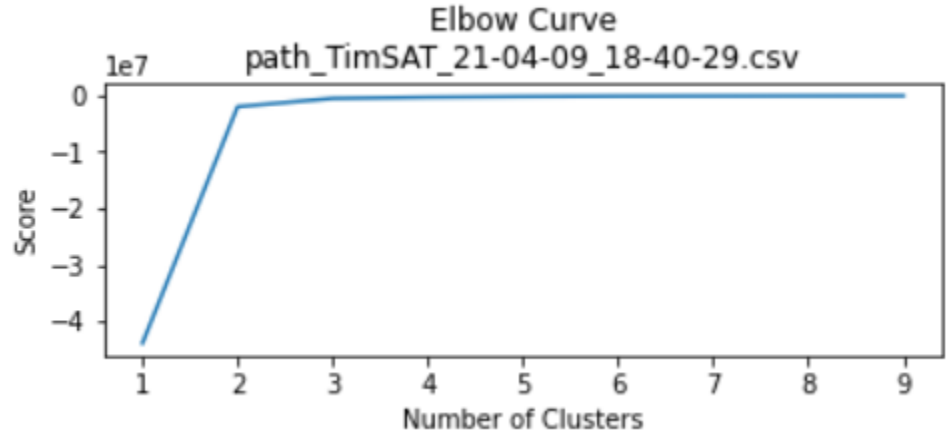
For n_clusters=5, The Silhouette Coefficient is 0.5802535049328704
For n_clusters=6, The Silhouette Coefficient is 0.5706734228759832
For n_clusters=7, The Silhouette Coefficient is 0.564802266875791
For n_clusters=8, The Silhouette Coefficient is 0.5605338137027868
For n_clusters=9, The Silhouette Coefficient is 0.5557747488521222

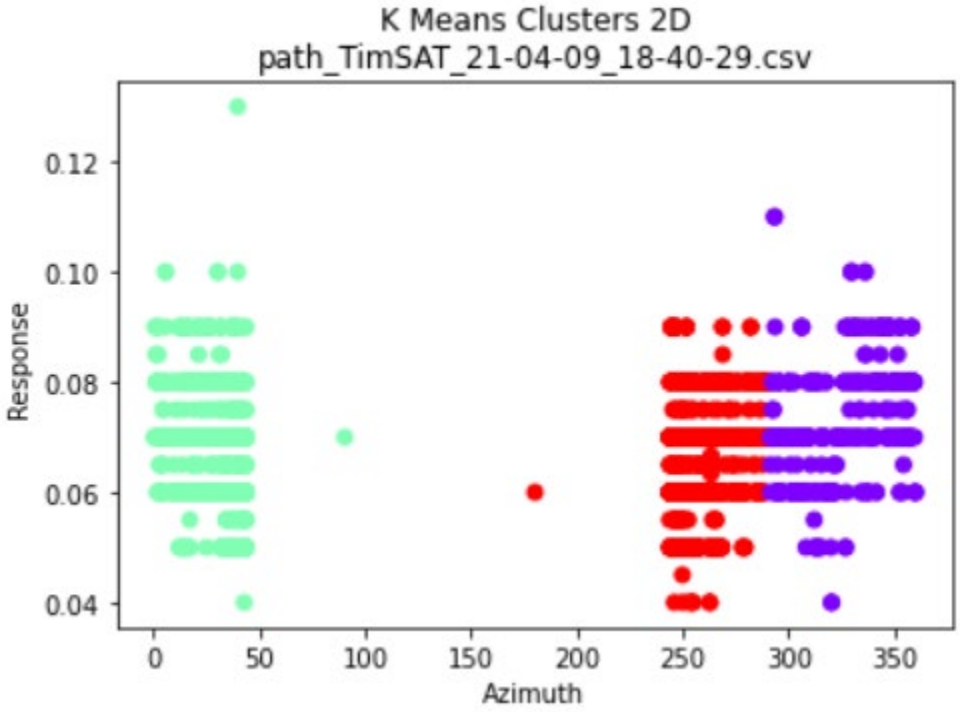




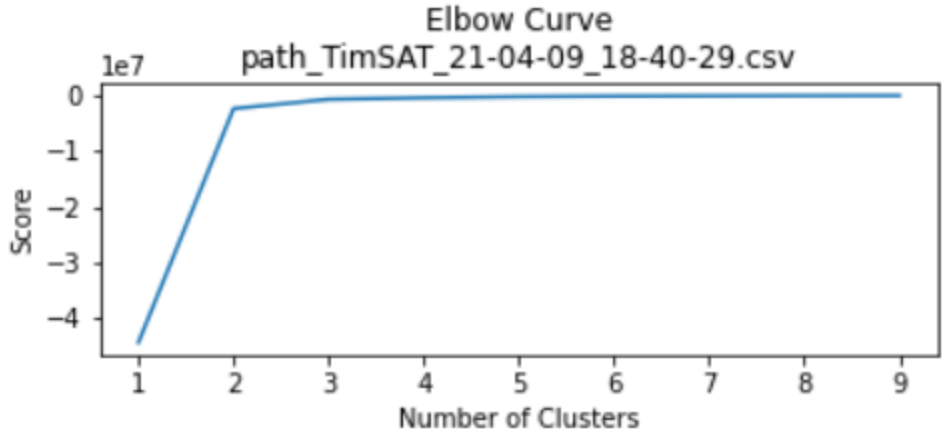
For n_clusters=2, The Silhouette Coefficient is 0.9003997993008455
For n_clusters=3, The Silhouette Coefficient is 0.8304061798592213
For n_clusters=4, The Silhouette Coefficient is 0.7938152183648608
For n_clusters=5, The Silhouette Coefficient is 0.6787934170281836
For n_clusters=6, The Silhouette Coefficient is 0.6525501279000385
For n_clusters=7, The Silhouette Coefficient is 0.6381188151964549
For n_clusters=8, The Silhouette Coefficient is 0.620551219396405

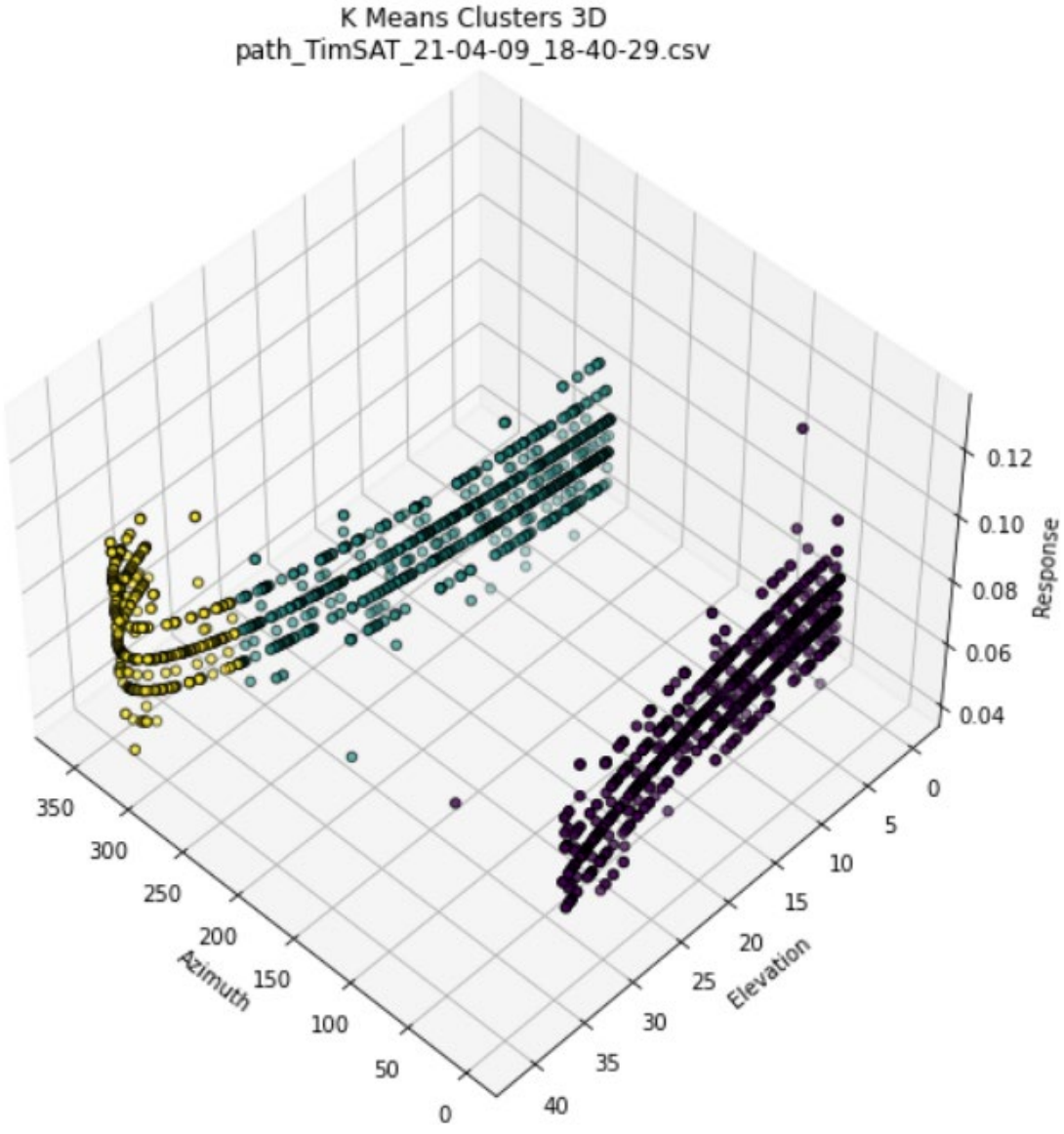
For n_clusters=9, The Silhouette Coefficient is 0.6032410664111615





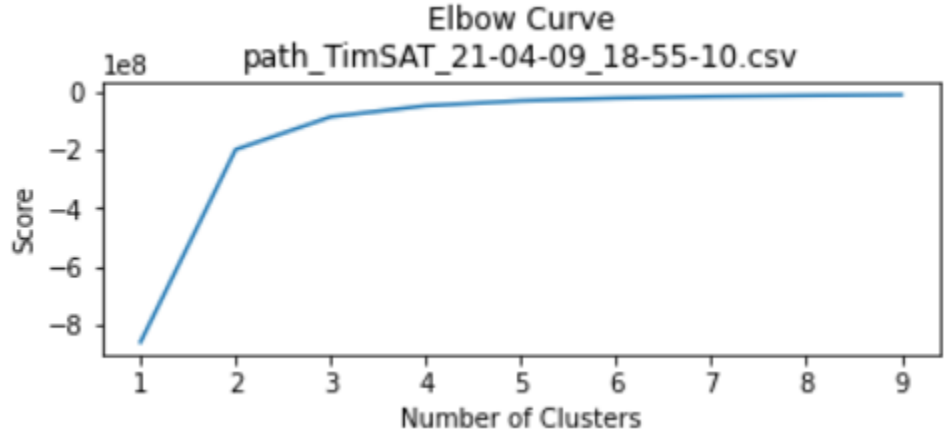
For n_clusters=2, The Silhouette Coefficient is 0.8824668100542858
For n_clusters=3, The Silhouette Coefficient is 0.8027863586083128
For n_clusters=4, The Silhouette Coefficient is 0.766817239278739
For n_clusters=5, The Silhouette Coefficient is 0.6490984206492102
For n_clusters=6, The Silhouette Coefficient is 0.6293102309384982
For n_clusters=7, The Silhouette Coefficient is 0.6164148800705778
For n_clusters=8, The Silhouette Coefficient is 0.5969166199984697
For n_clusters=9, The Silhouette Coefficient is 0.5868887498916382

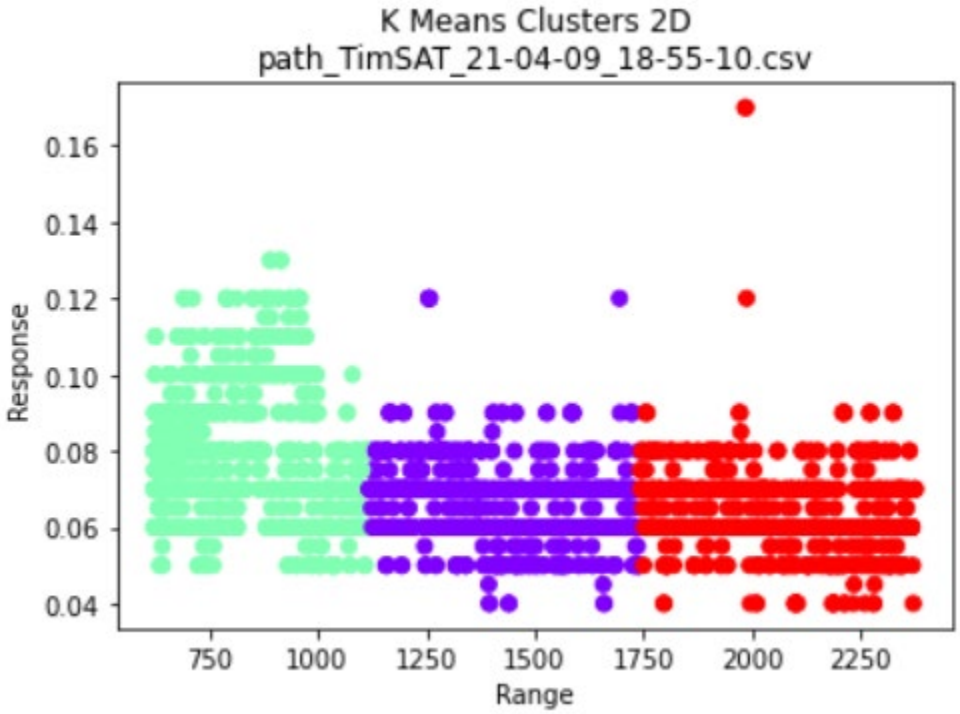




For n_clusters=2, The Silhouette Coefficient is 0.6485990007913757

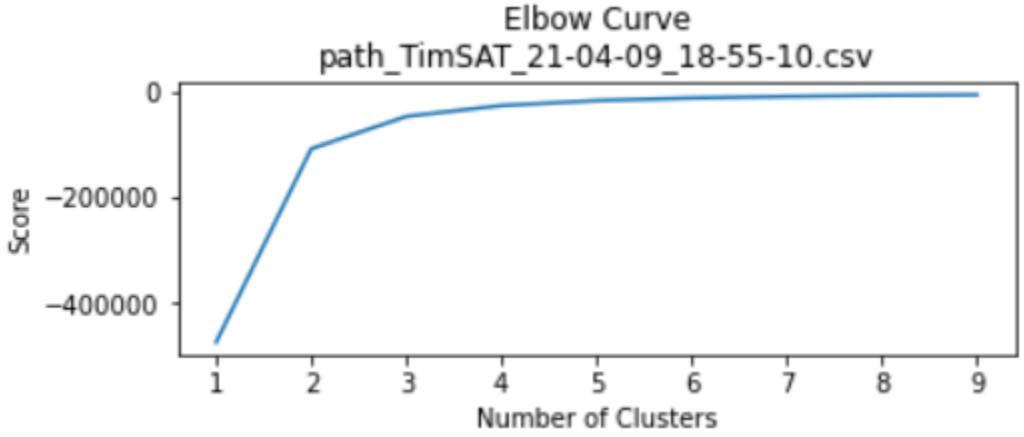
For n_clusters=3, The Silhouette Coefficient is 0.6142309326436637
For n_clusters=4, The Silhouette Coefficient is 0.5962657109006916
For n_clusters=5, The Silhouette Coefficient is 0.5833251578485438
For n_clusters=6, The Silhouette Coefficient is 0.5748793955227559
For n_clusters=7, The Silhouette Coefficient is 0.5696498384393258
For n_clusters=8, The Silhouette Coefficient is 0.5649080721306031
For n_clusters=9, The Silhouette Coefficient is 0.5595977001007452

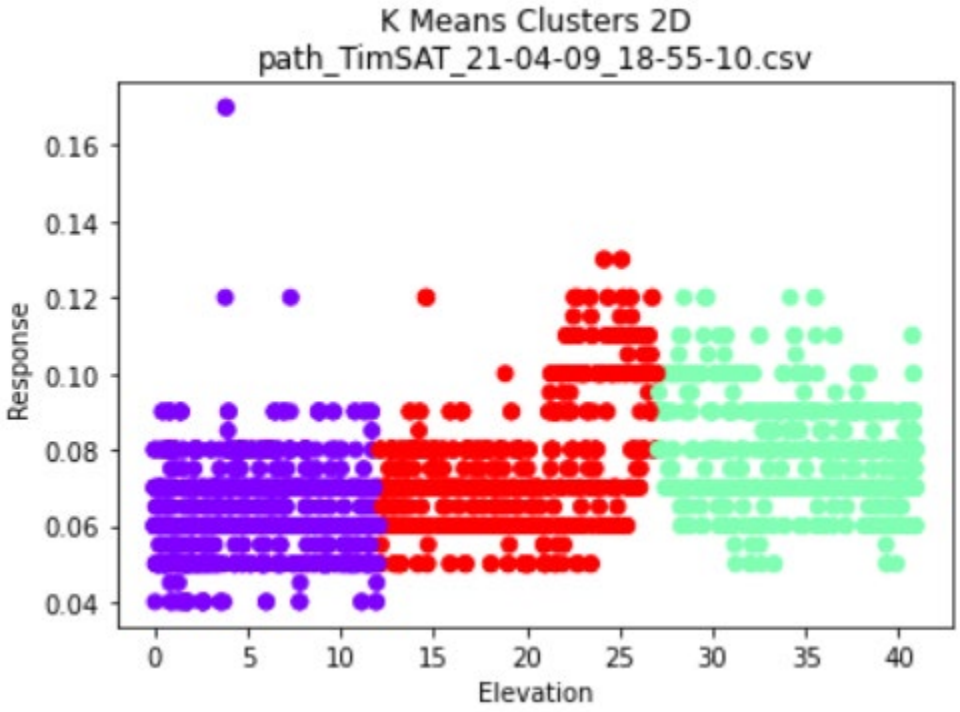




For n_clusters=2, The Silhouette Coefficient is 0.6640951538782159
For n_clusters=3, The Silhouette Coefficient is 0.6173721748574945
For n_clusters=4, The Silhouette Coefficient is 0.5950083981595532

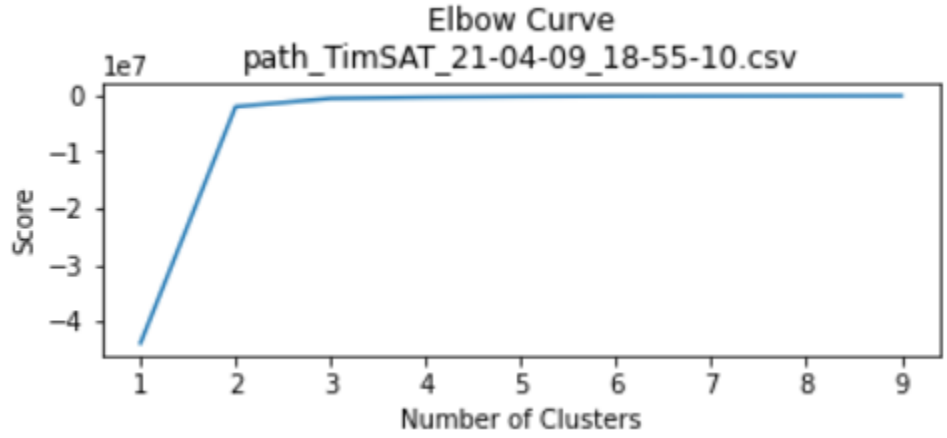
For n_clusters=5, The Silhouette Coefficient is 0.5805722044524045
For n_clusters=6, The Silhouette Coefficient is 0.5718158458393817
For n_clusters=7, The Silhouette Coefficient is 0.5656287562520947
For n_clusters=8, The Silhouette Coefficient is 0.5590510870797014
For n_clusters=9, The Silhouette Coefficient is 0.5545526554654774

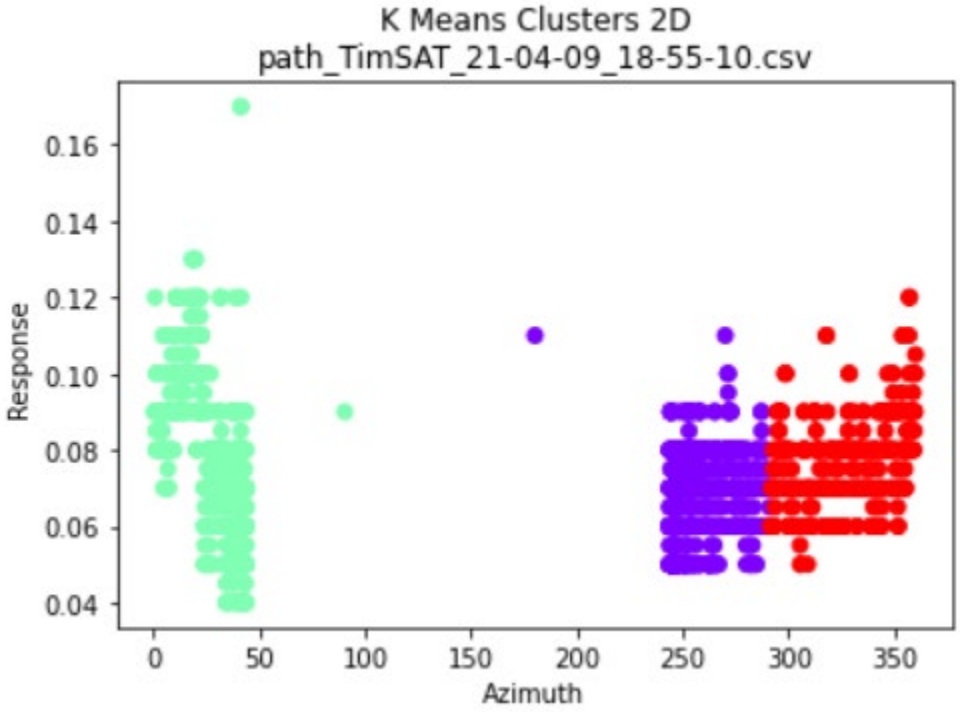




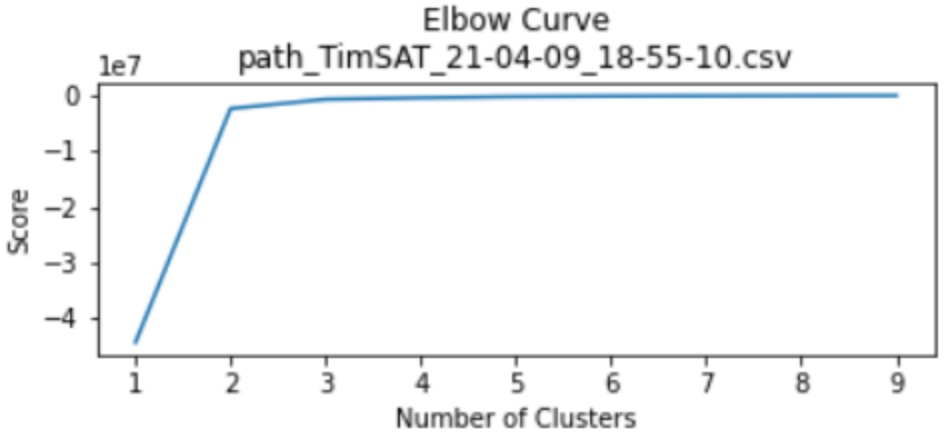
For n_clusters=2, The Silhouette Coefficient is 0.9003937577081761
For n_clusters=3, The Silhouette Coefficient is 0.830527753286595
For n_clusters=4, The Silhouette Coefficient is 0.7934384188726147
For n_clusters=5, The Silhouette Coefficient is 0.6770735748625119
For n_clusters=6, The Silhouette Coefficient is 0.6541820132512696
For n_clusters=7, The Silhouette Coefficient is 0.6388515175388225
For n_clusters=8, The Silhouette Coefficient is 0.6141146795557548

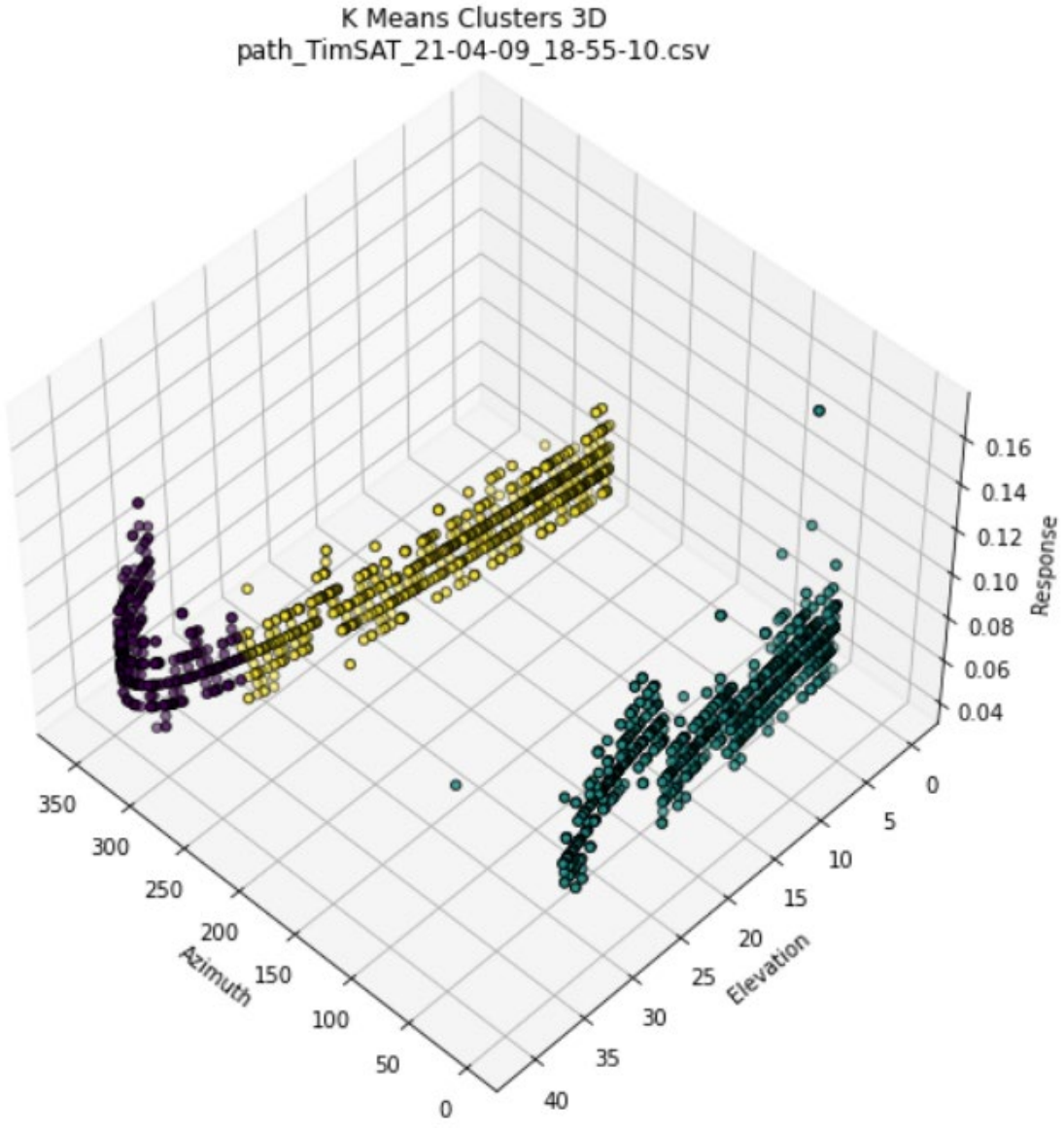
For n_clusters=9, The Silhouette Coefficient is 0.6035343486559936





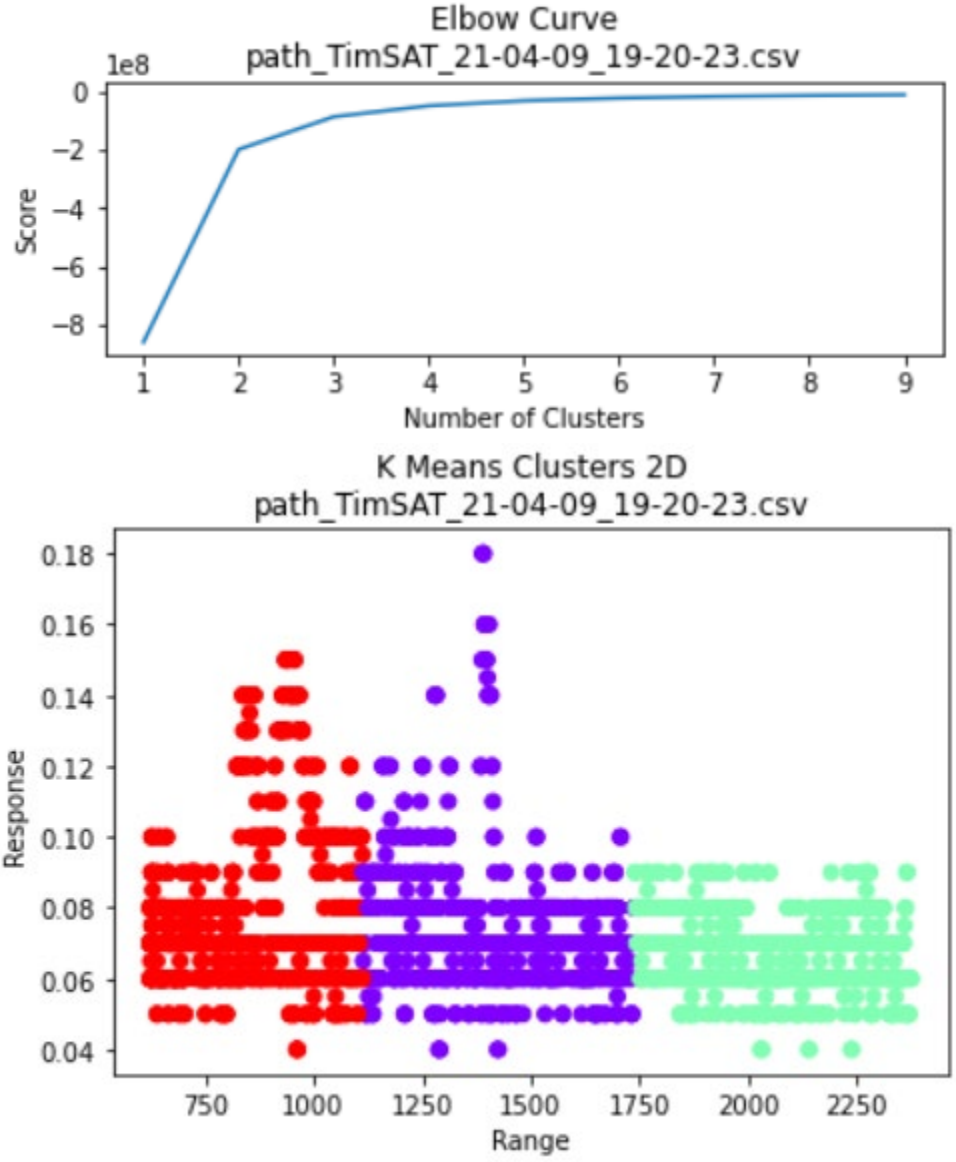
For n_clusters=2, The Silhouette Coefficient is 0.8824528223871805
For n_clusters=3, The Silhouette Coefficient is 0.8033785674699083
For n_clusters=4, The Silhouette Coefficient is 0.7695829596948284
For n_clusters=5, The Silhouette Coefficient is 0.6513127385113583
For n_clusters=6, The Silhouette Coefficient is 0.6298166996834618
For n_clusters=7, The Silhouette Coefficient is 0.6149118486830031
For n_clusters=8, The Silhouette Coefficient is 0.5959638878815015
For n_clusters=9, The Silhouette Coefficient is 0.5861714130145079





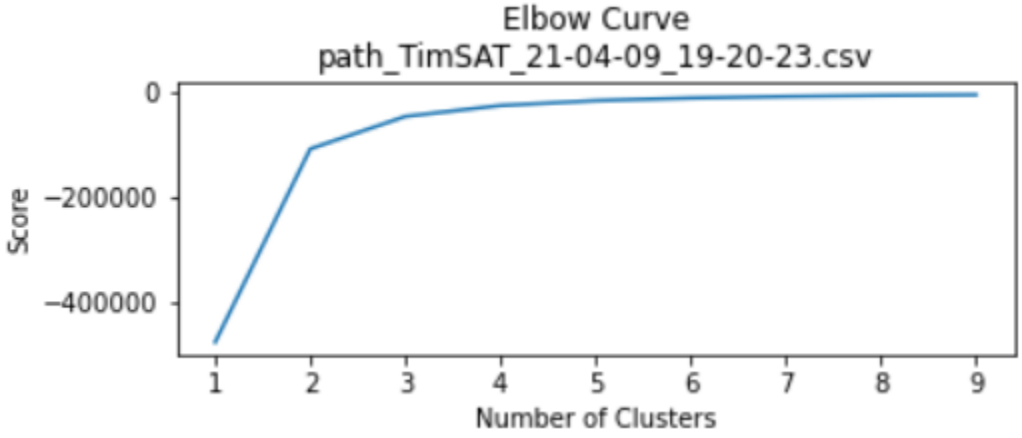
For n_clusters=2, The Silhouette Coefficient is 0.6491518595101199

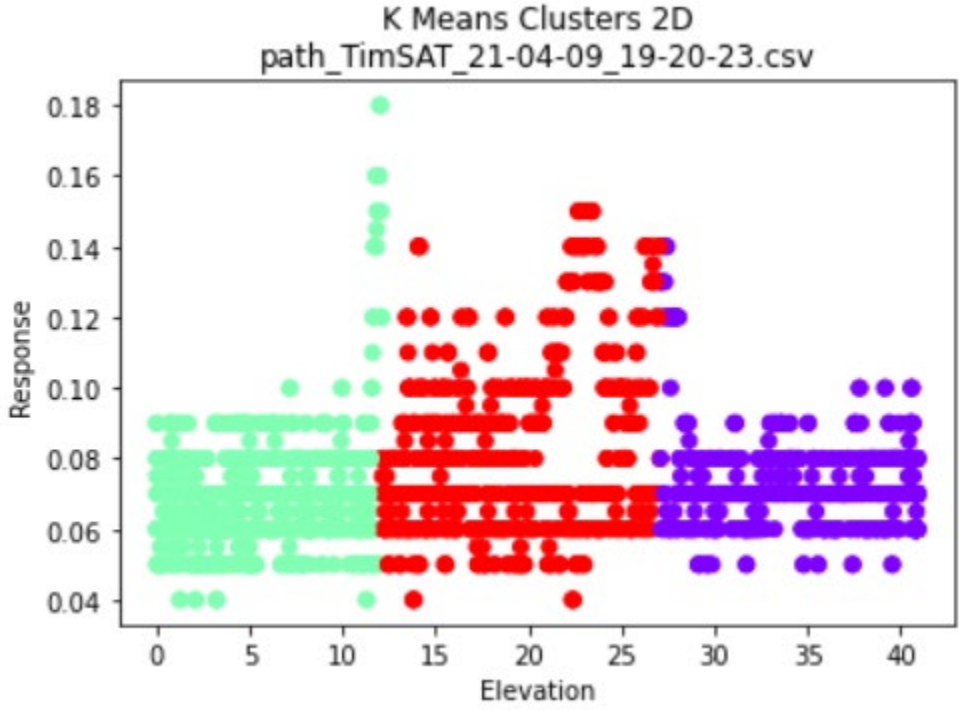
For n_clusters=3, The Silhouette Coefficient is 0.6141673817785293
For n_clusters=4, The Silhouette Coefficient is 0.5946195753272223
For n_clusters=5, The Silhouette Coefficient is 0.5841127186971257
For n_clusters=6, The Silhouette Coefficient is 0.5750617930542855
For n_clusters=7, The Silhouette Coefficient is 0.5694457709094166
For n_clusters=8, The Silhouette Coefficient is 0.5648567936290979
For n_clusters=9, The Silhouette Coefficient is 0.5586912115028386



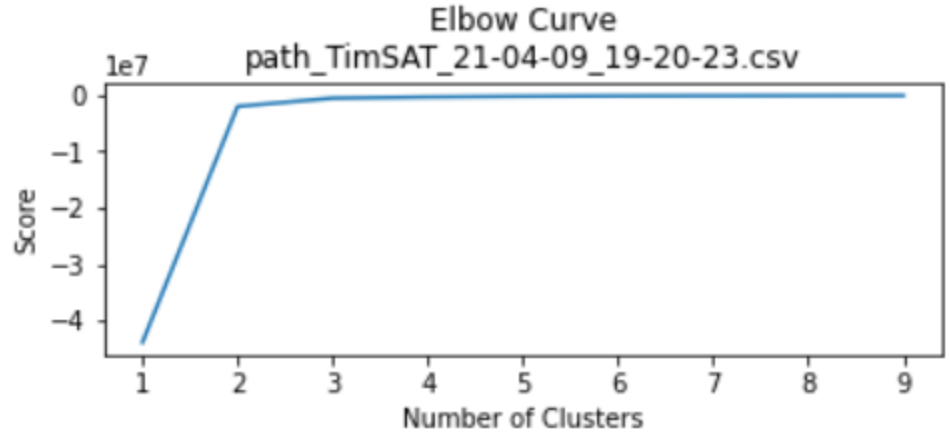
For n_clusters=2, The Silhouette Coefficient is 0.6636481624410627
For n_clusters=3, The Silhouette Coefficient is 0.6173610916639006
For n_clusters=4, The Silhouette Coefficient is 0.5952084573719343
For n_clusters=5, The Silhouette Coefficient is 0.5810679357950641

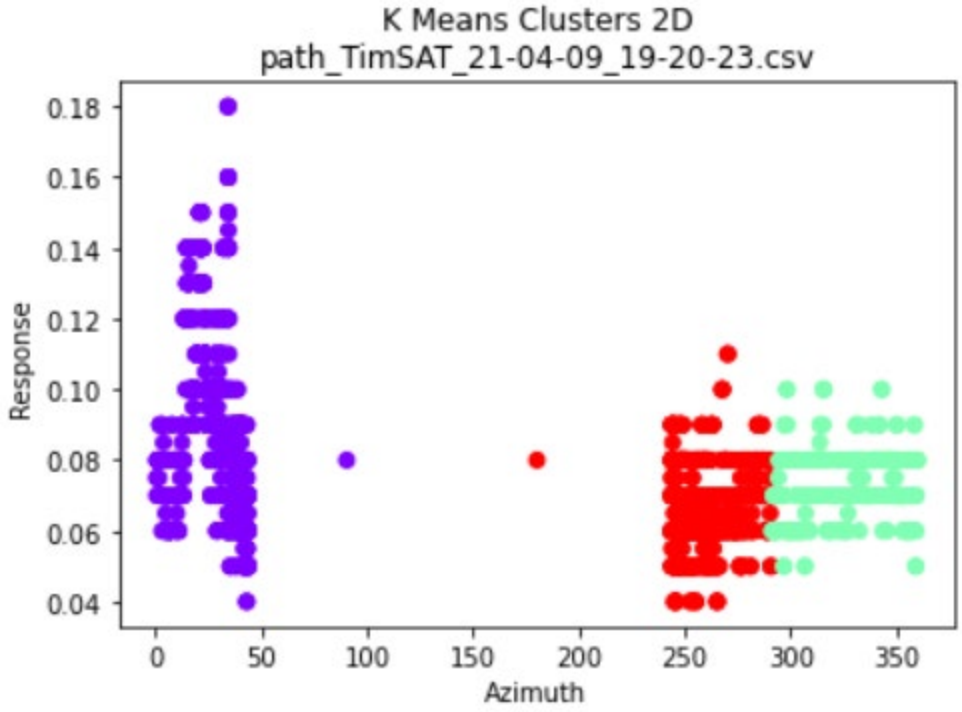
For n_clusters=6, The Silhouette Coefficient is 0.5707425885137803
For n_clusters=7, The Silhouette Coefficient is 0.5645996894252385
For n_clusters=8, The Silhouette Coefficient is 0.5582674179844994
For n_clusters=9, The Silhouette Coefficient is 0.5554028223910805



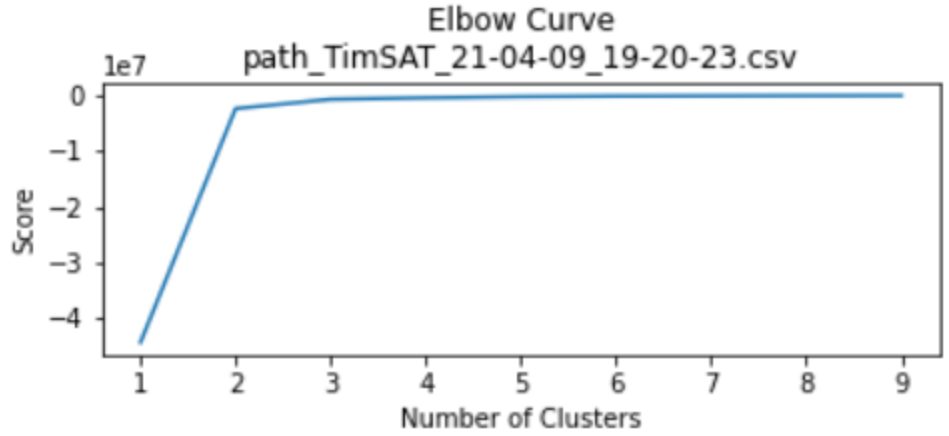


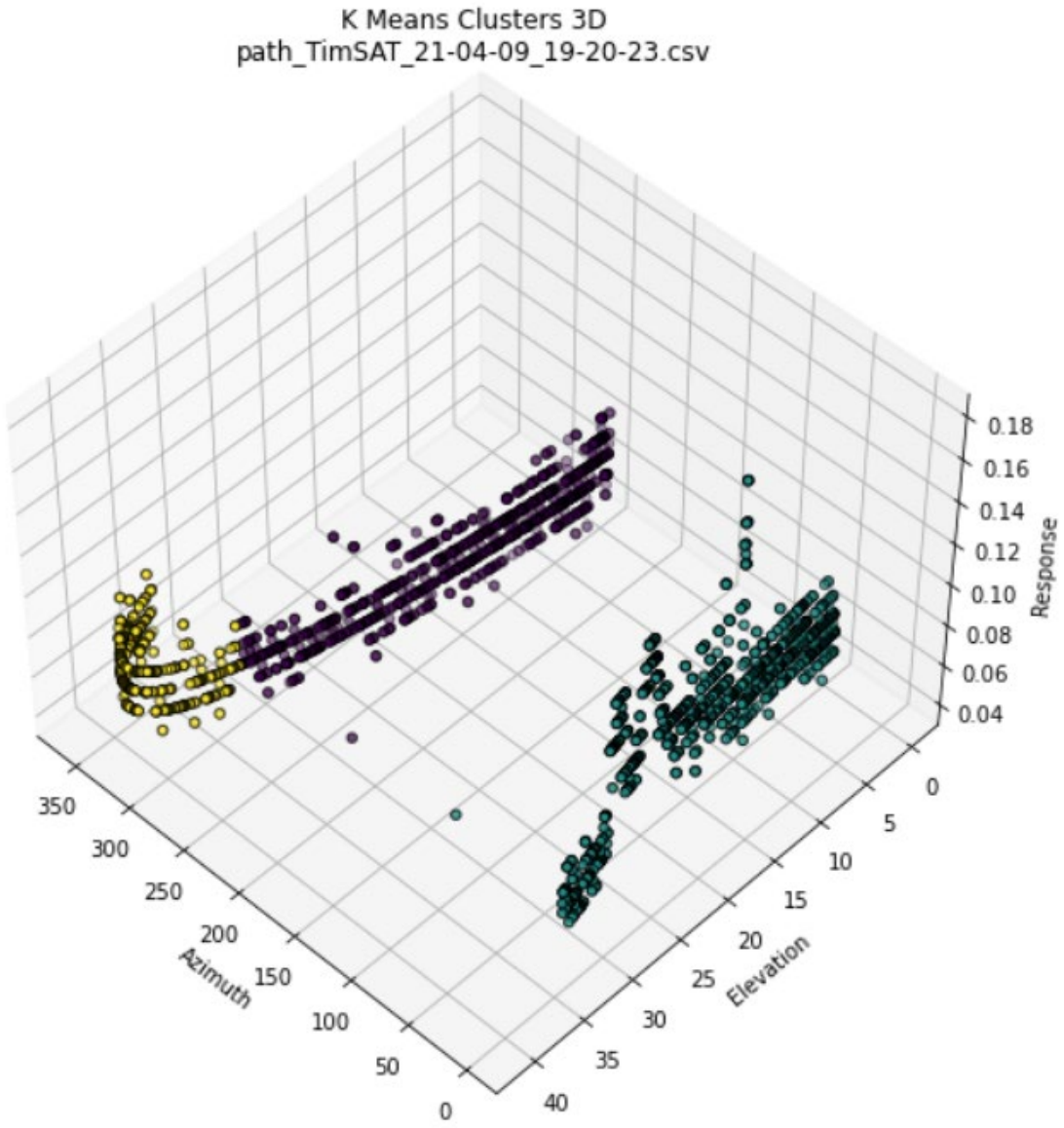
For n_clusters=2, The Silhouette Coefficient is 0.9004335111780254
For n_clusters=3, The Silhouette Coefficient is 0.8304199591694837
For n_clusters=4, The Silhouette Coefficient is 0.7934394497000533
For n_clusters=5, The Silhouette Coefficient is 0.67696894093557
For n_clusters=6, The Silhouette Coefficient is 0.6546727114945686
For n_clusters=7, The Silhouette Coefficient is 0.6411289912329604
For n_clusters=8, The Silhouette Coefficient is 0.6155923390955069
For n_clusters=9, The Silhouette Coefficient is 0.6048800460509356



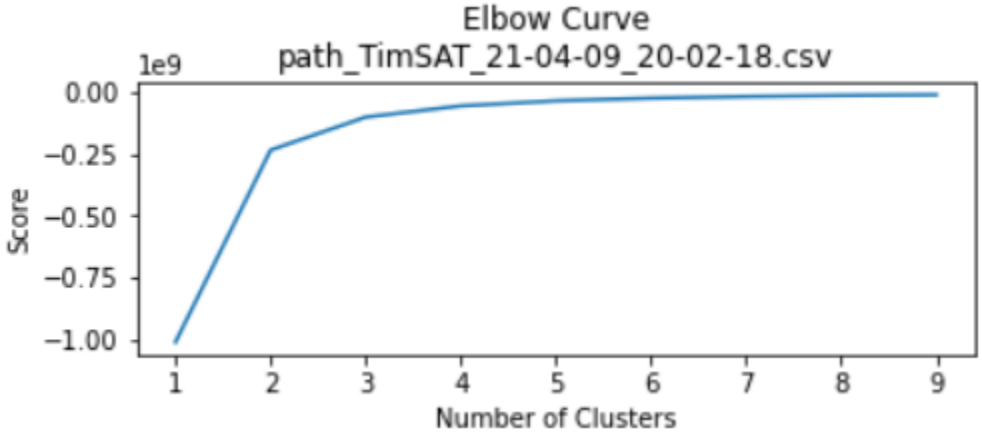


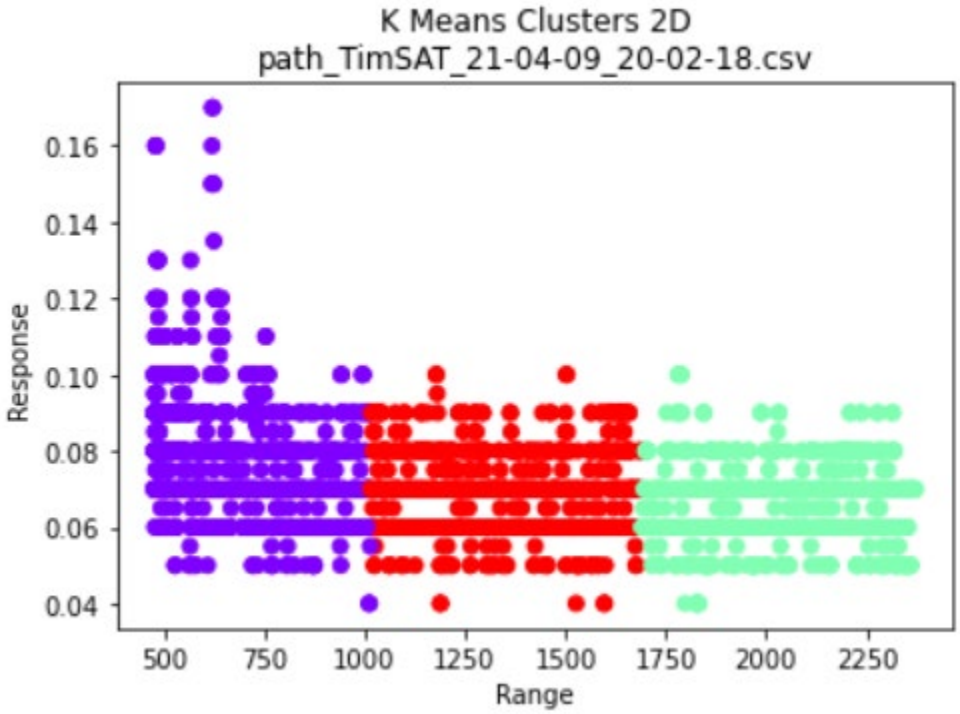
For n_clusters=2, The Silhouette Coefficient is 0.8824949128623985
For n_clusters=3, The Silhouette Coefficient is 0.8027628920244758
For n_clusters=4, The Silhouette Coefficient is 0.7682506849762937
For n_clusters=5, The Silhouette Coefficient is 0.6487703822768061
For n_clusters=6, The Silhouette Coefficient is 0.6296264989358301
For n_clusters=7, The Silhouette Coefficient is 0.6166266311973361
For n_clusters=8, The Silhouette Coefficient is 0.5960129453791122
For n_clusters=9, The Silhouette Coefficient is 0.5877810789781595


Elbow Curve
path_TimSAT_21-04-09_19-20-23.csv
1e7
0
−1
−2
−3
−4
Score
1
2
3
4
5
6
7
8
9
Number of Clusters


K Means Clusters 3D
path_TimSAT_21-04-09_19-20-23.csv
0.18
0.16
0.14
0.12
0.10
0.08
0.06
0.04
Response
350
300
250
200
150
100
50
0
Azimuth
0
5
10
15
20
25
30
35
40
Elevation

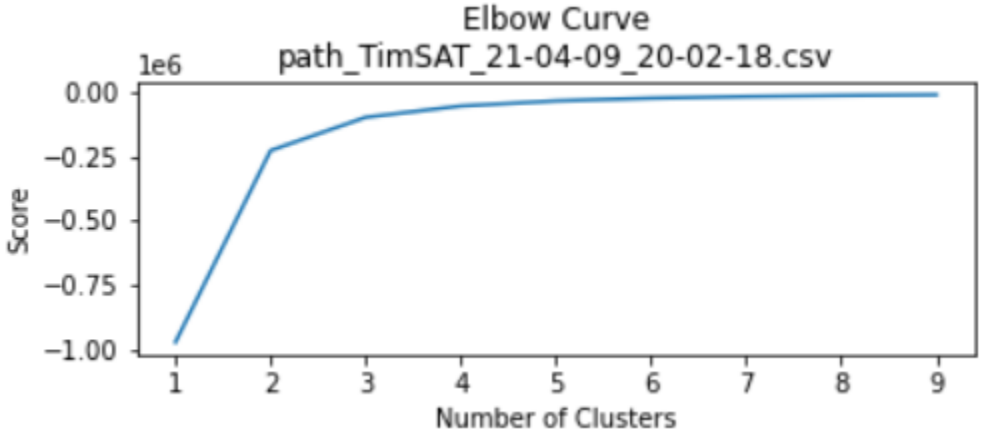
For n_clusters=2, The Silhouette Coefficient is 0.6450738216938539
For n_clusters=3, The Silhouette Coefficient is 0.6097909803076133
For n_clusters=4, The Silhouette Coefficient is 0.5915435615245994
For n_clusters=5, The Silhouette Coefficient is 0.5807925128586988
For n_clusters=6, The Silhouette Coefficient is 0.5714826674733159
For n_clusters=7, The Silhouette Coefficient is 0.5652444917454542
For n_clusters=8, The Silhouette Coefficient is 0.5610147453976717
For n_clusters=9, The Silhouette Coefficient is 0.5588149686577855

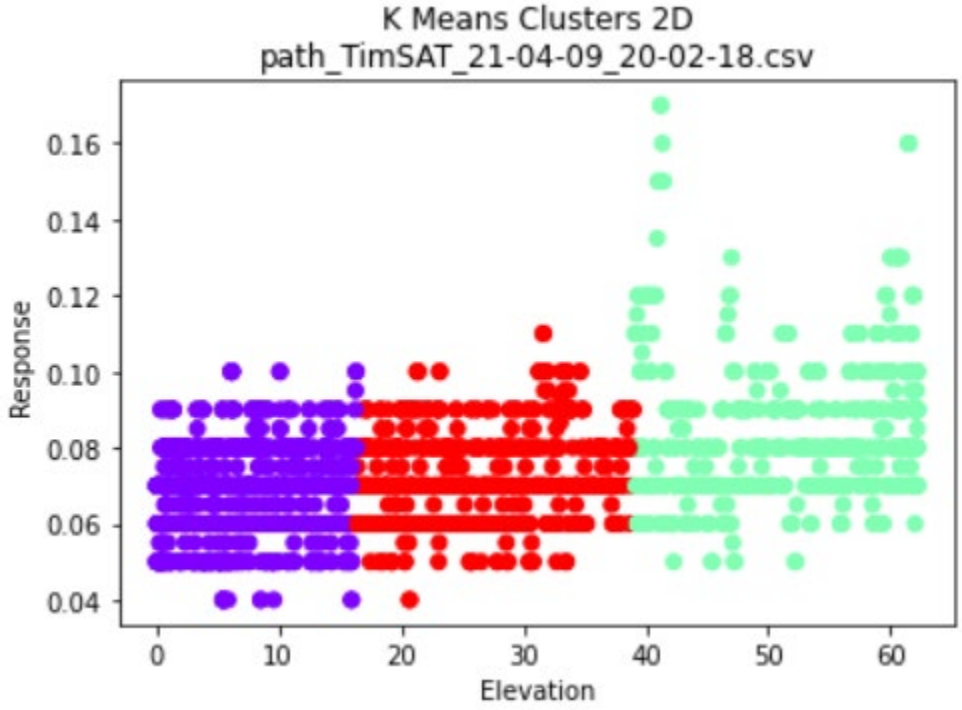




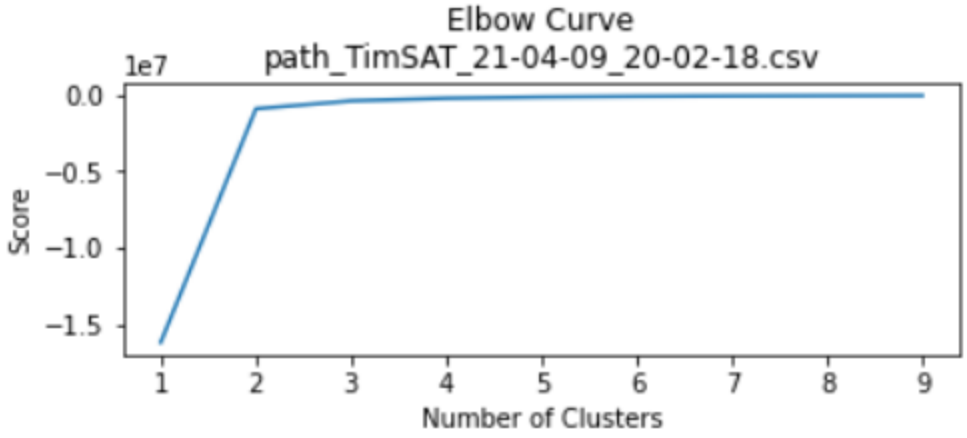
For n_clusters=2, The Silhouette Coefficient is 0.6833677904739591
For n_clusters=3, The Silhouette Coefficient is 0.6264021921933344
For n_clusters=4, The Silhouette Coefficient is 0.5991387114098354
For n_clusters=5, The Silhouette Coefficient is 0.5839802423188313

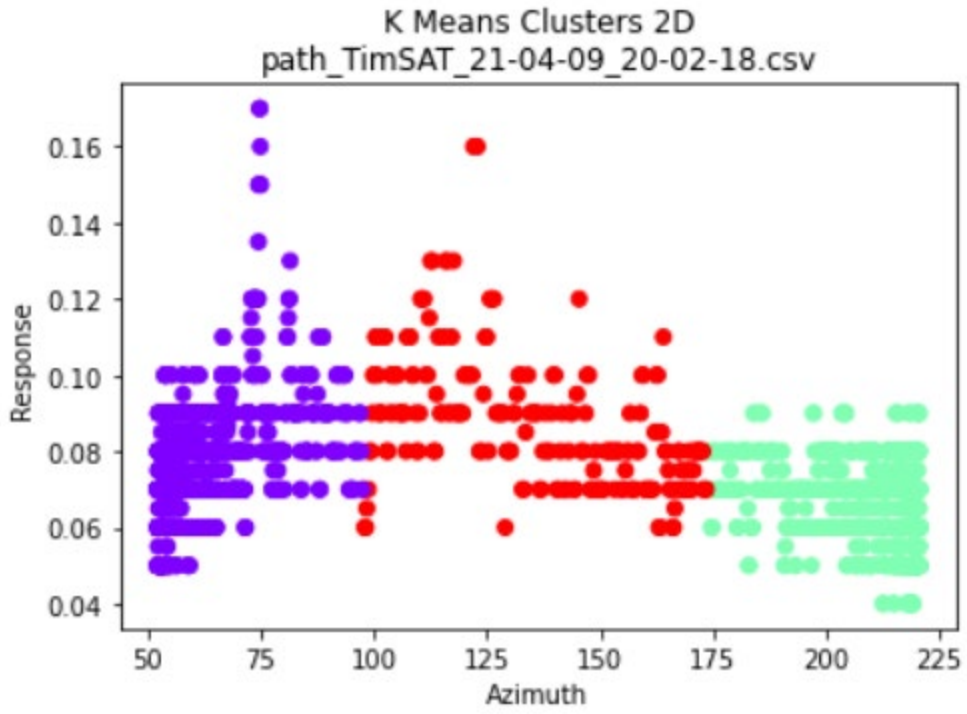
For n_clusters=6, The Silhouette Coefficient is 0.5718968066177139
For n_clusters=7, The Silhouette Coefficient is 0.5640481067627112
For n_clusters=8, The Silhouette Coefficient is 0.5598547988278584
For n_clusters=9, The Silhouette Coefficient is 0.5549042539387564



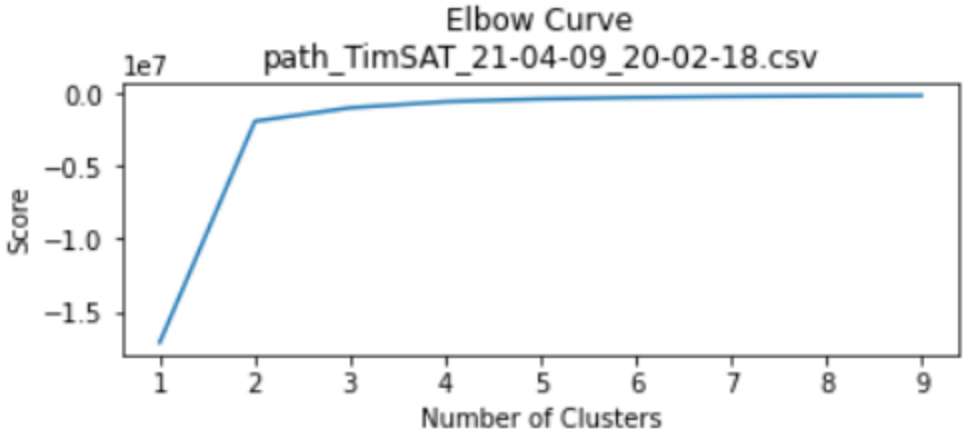


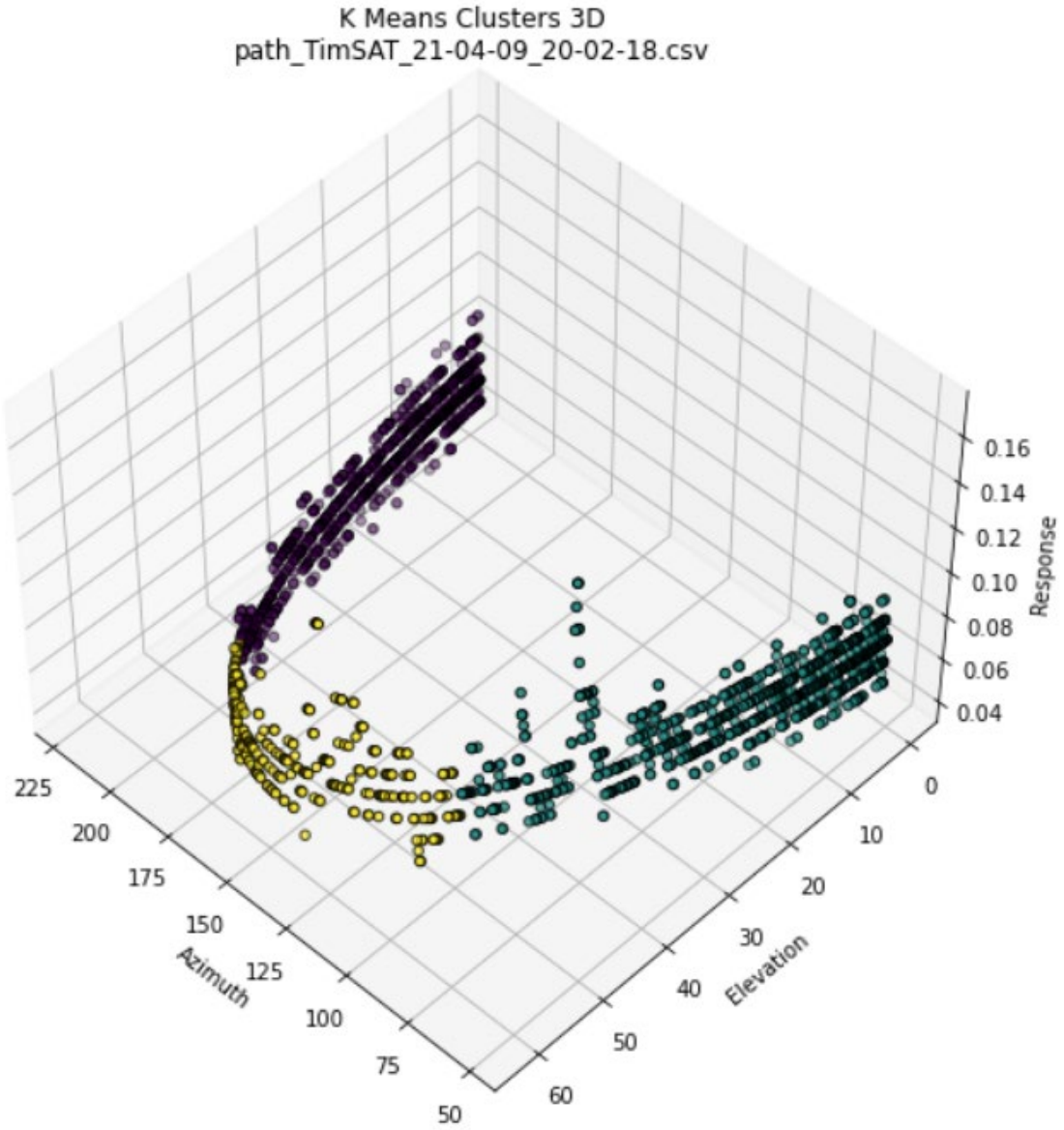
For n_clusters=2, The Silhouette Coefficient is 0.8724029250525156
For n_clusters=3, The Silhouette Coefficient is 0.8251846609753427
For n_clusters=4, The Silhouette Coefficient is 0.7783448849827457
For n_clusters=5, The Silhouette Coefficient is 0.7413294636598955
For n_clusters=6, The Silhouette Coefficient is 0.7071175002476782
For n_clusters=7, The Silhouette Coefficient is 0.6829281112095771
For n_clusters=8, The Silhouette Coefficient is 0.6703891886260314
For n_clusters=9, The Silhouette Coefficient is 0.6467353136570996



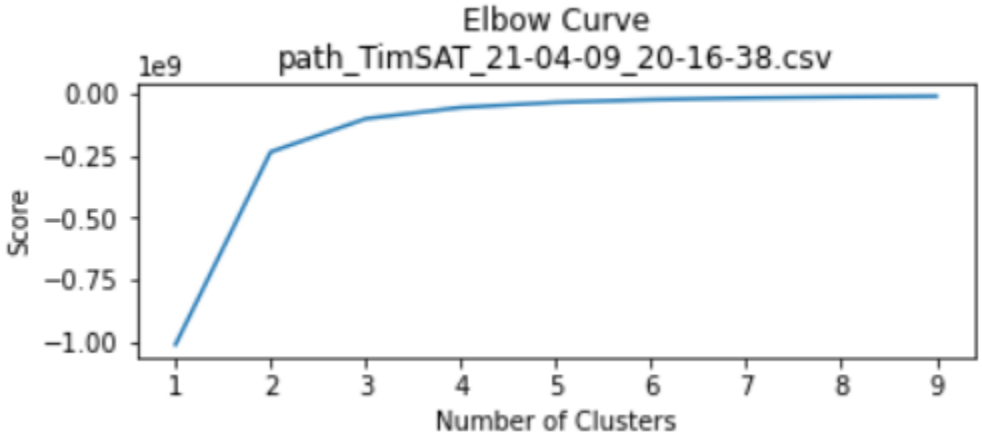


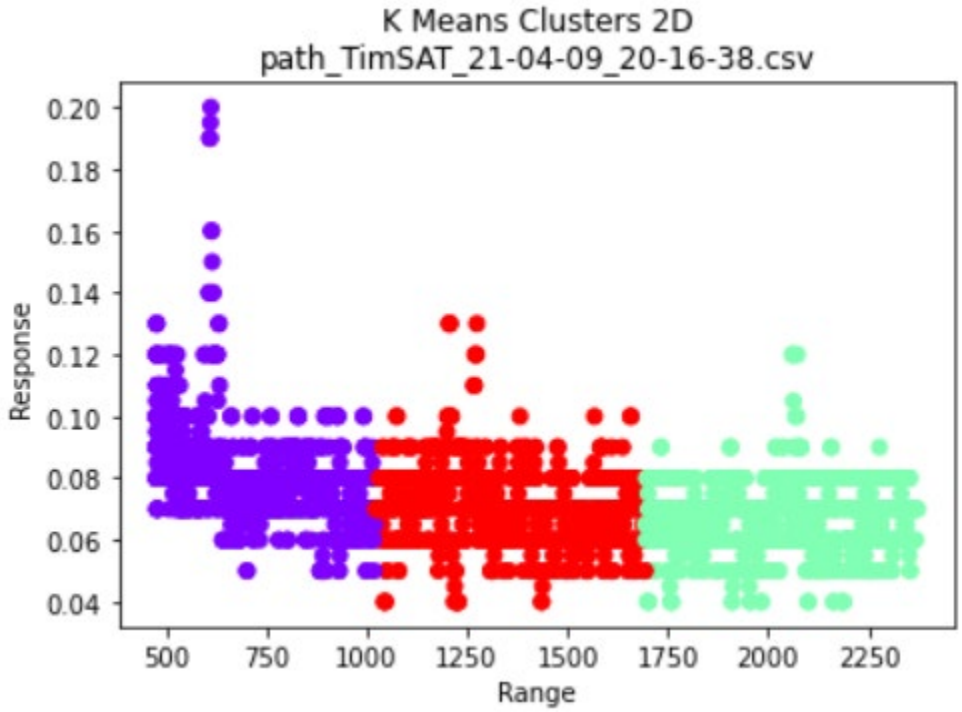
For n_clusters=2, The Silhouette Coefficient is 0.8060027681313787
For n_clusters=3, The Silhouette Coefficient is 0.7486494661519573
For n_clusters=4, The Silhouette Coefficient is 0.6913533374253987
For n_clusters=5, The Silhouette Coefficient is 0.6592561447095443
For n_clusters=6, The Silhouette Coefficient is 0.631083959712522
For n_clusters=7, The Silhouette Coefficient is 0.6130689733957352
For n_clusters=8, The Silhouette Coefficient is 0.5990568275907581
For n_clusters=9, The Silhouette Coefficient is 0.5909370898363774


Elbow Curve
path_TimSAT_21-04-09_20-02-18.csv
1e7
0.0
−0.5
−1.0
−1.5
Score
1
2
3
4
5
6
7
8
9
Number of Clusters


K Means Clusters 3D
path_TimSAT_21-04-09_20-02-18.csv
0.16
0.14
0.12
0.10
0.08
0.06
0.04
Response
225
200
175
150
125
100
75
50
Azimuth
0
10
20
30
40
50
60
Elevation

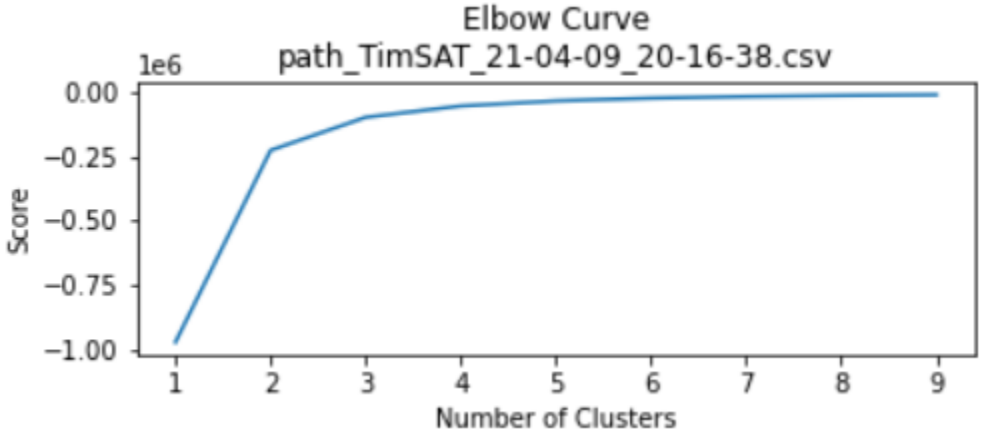
For n_clusters=2, The Silhouette Coefficient is 0.6442623864702967
For n_clusters=3, The Silhouette Coefficient is 0.6096968437420853
For n_clusters=4, The Silhouette Coefficient is 0.5920357946871688
For n_clusters=5, The Silhouette Coefficient is 0.5804527177178417
For n_clusters=6, The Silhouette Coefficient is 0.5713698346558206
For n_clusters=7, The Silhouette Coefficient is 0.5661968084875927
For n_clusters=8, The Silhouette Coefficient is 0.5615742371061955
For n_clusters=9, The Silhouette Coefficient is 0.5581910157477895

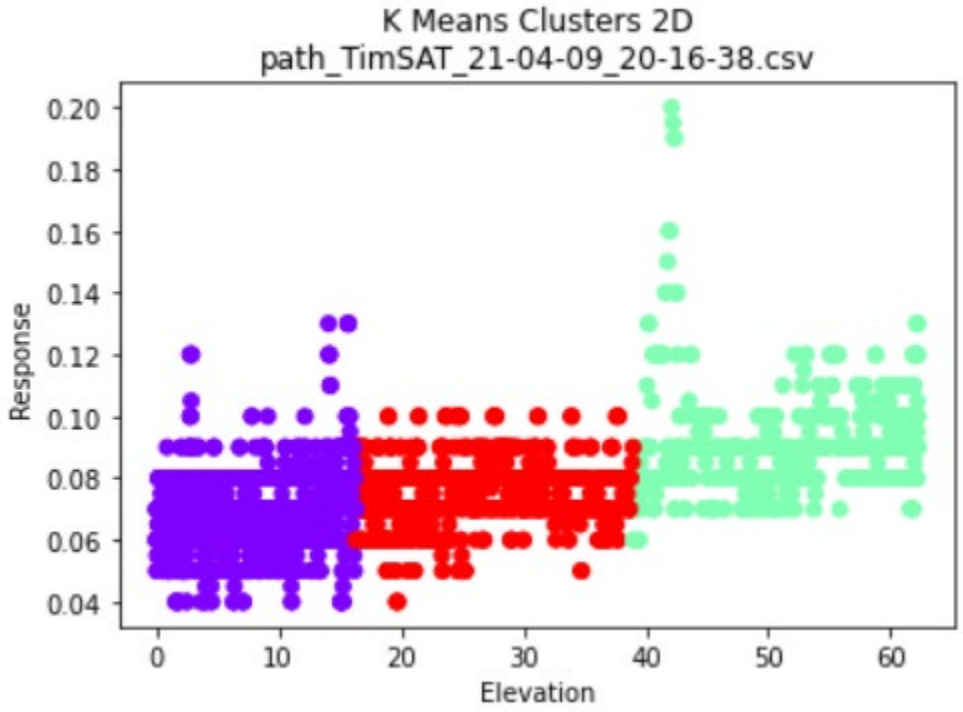




For n_clusters=2, The Silhouette Coefficient is 0.6845728904763417
For n_clusters=3, The Silhouette Coefficient is 0.6262633191164899
For n_clusters=4, The Silhouette Coefficient is 0.5996638911349859

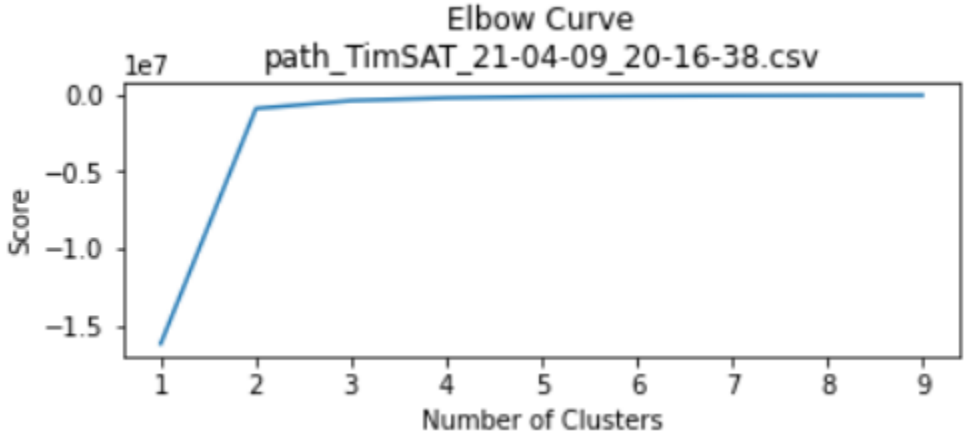
For n_clusters=5, The Silhouette Coefficient is 0.5835838906199832
For n_clusters=6, The Silhouette Coefficient is 0.5728106508281139
For n_clusters=7, The Silhouette Coefficient is 0.5646806697319662
For n_clusters=8, The Silhouette Coefficient is 0.5595296011364533
For n_clusters=9, The Silhouette Coefficient is 0.5541184108764984

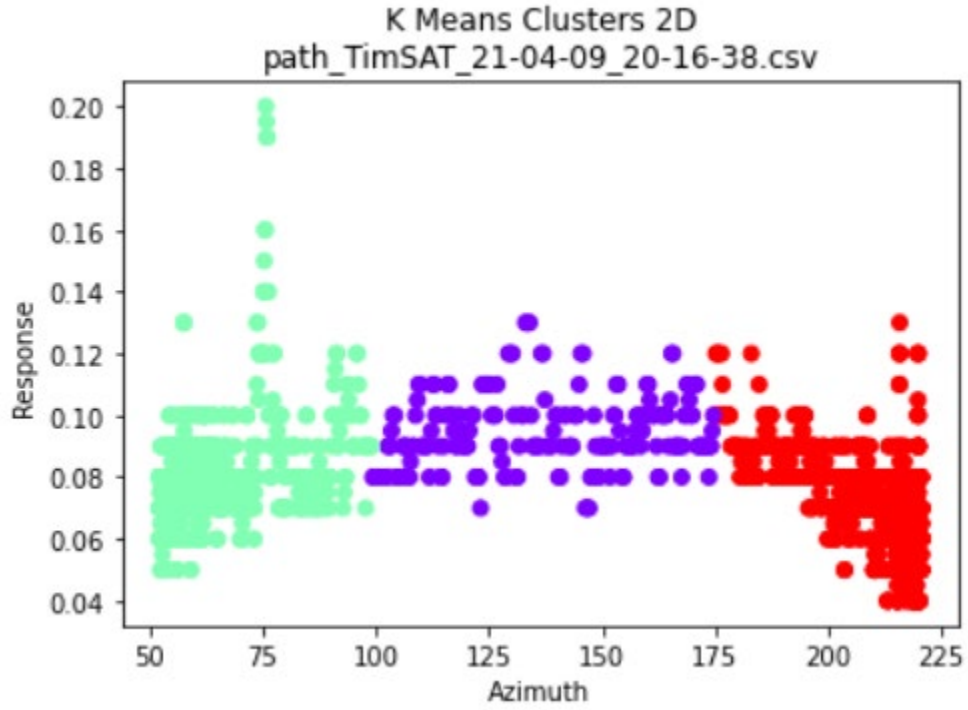




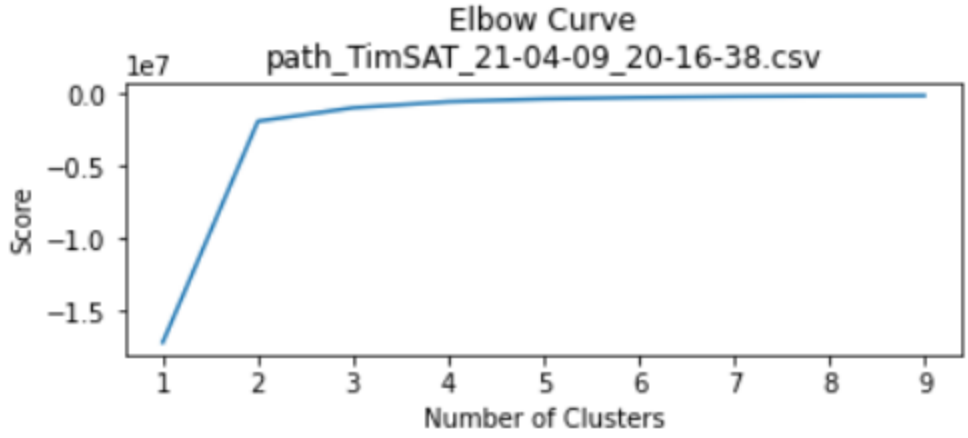
For n_clusters=2, The Silhouette Coefficient is 0.872432595198676
For n_clusters=3, The Silhouette Coefficient is 0.8253674919285936
For n_clusters=4, The Silhouette Coefficient is 0.7791029862082302
For n_clusters=5, The Silhouette Coefficient is 0.7421311913375119
For n_clusters=6, The Silhouette Coefficient is 0.7130538256407758
For n_clusters=7, The Silhouette Coefficient is 0.6887241872149181
For n_clusters=8, The Silhouette Coefficient is 0.6664913397855695

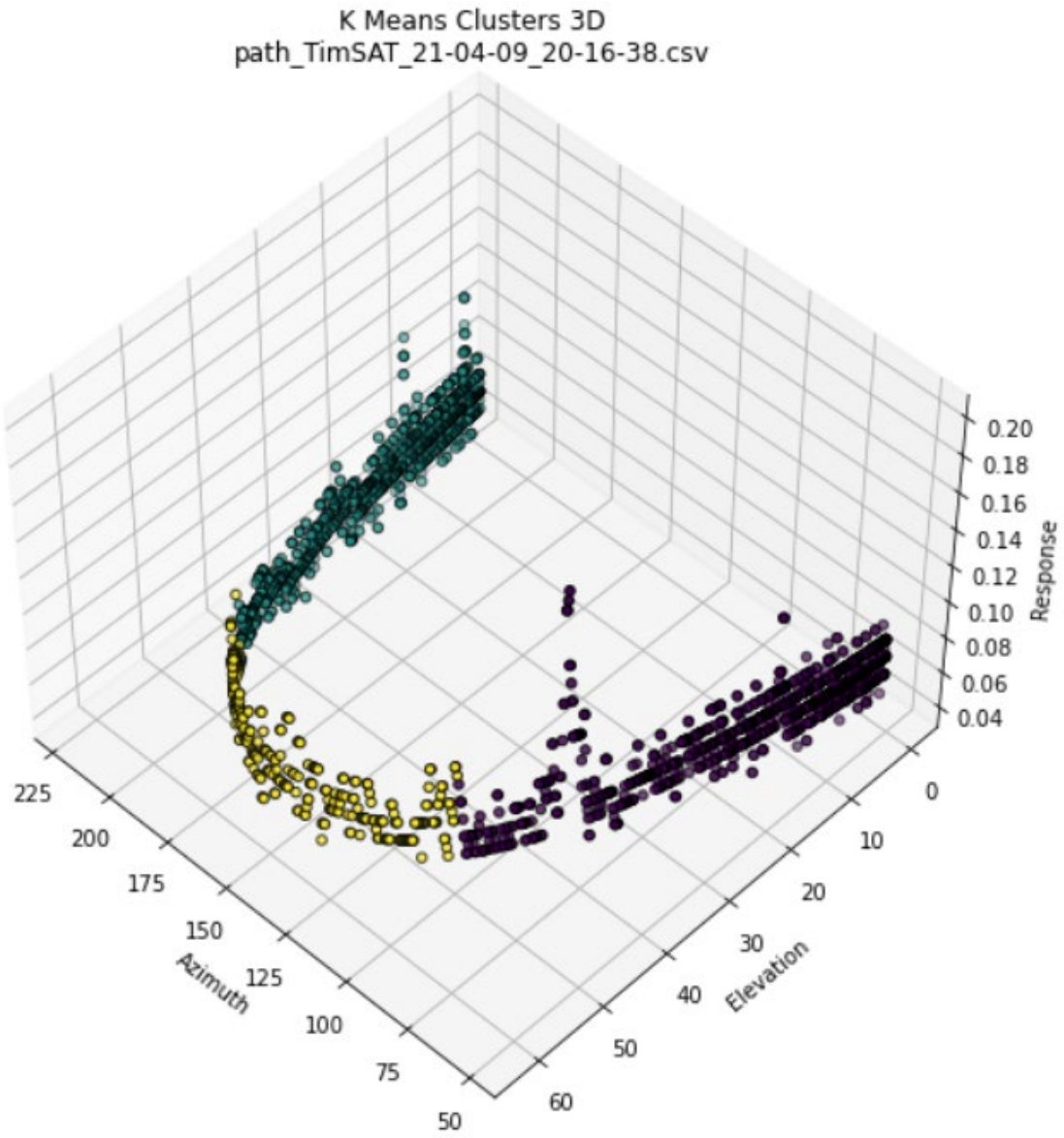
For n_clusters=9, The Silhouette Coefficient is 0.6506565972440226



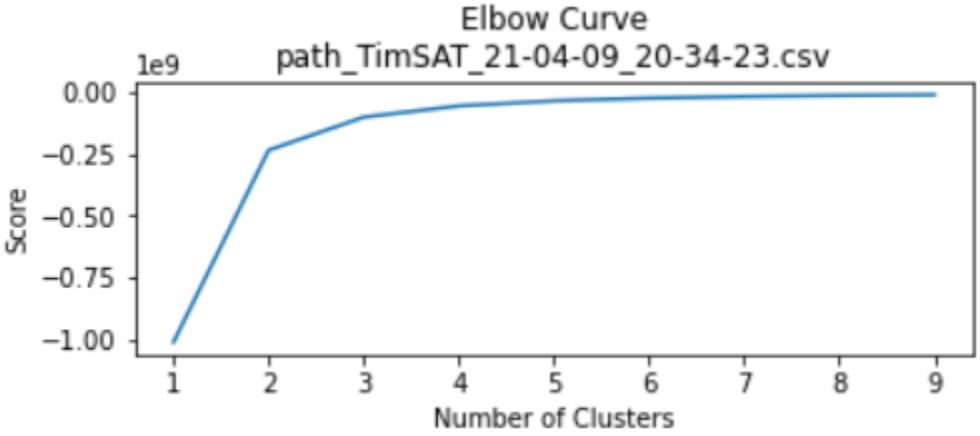


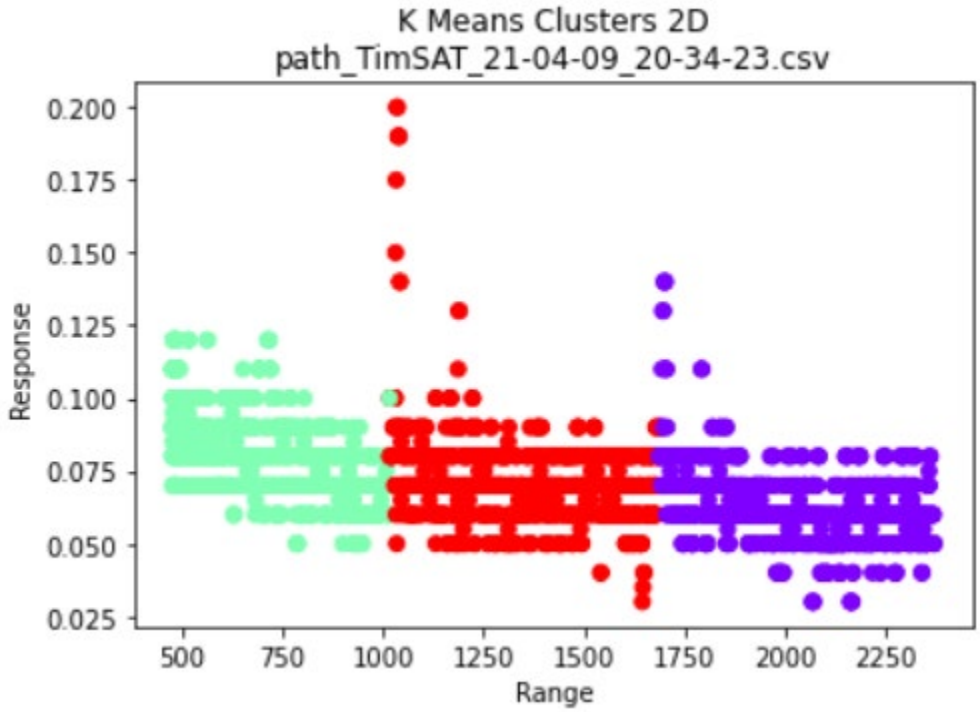
For n_clusters=2, The Silhouette Coefficient is 0.8060959427577198
For n_clusters=3, The Silhouette Coefficient is 0.7489549977250487
For n_clusters=4, The Silhouette Coefficient is 0.6916734266026439
For n_clusters=5, The Silhouette Coefficient is 0.6587260310283632
For n_clusters=6, The Silhouette Coefficient is 0.6317231449829467
For n_clusters=7, The Silhouette Coefficient is 0.6139526881632444
For n_clusters=8, The Silhouette Coefficient is 0.5976935738139929
For n_clusters=9, The Silhouette Coefficient is 0.5866231463292421


Elbow Curve
path_TimSAT_21-04-09_20-16-38.csv
1e7
0.0
−0.5
−1.0
−1.5
Score
1
2
3
4
5
6
7
8
9
Number of Clusters


K Means Clusters 3D
path_TimSAT_21-04-09_20-16-38.csv
0.20
0.18
0.16
0.14
0.12
0.10
0.08
0.06
0.04
Response
225
200
175
150
125
100
75
50
Azimuth
0
10
20
30
40
50
60
Elevation

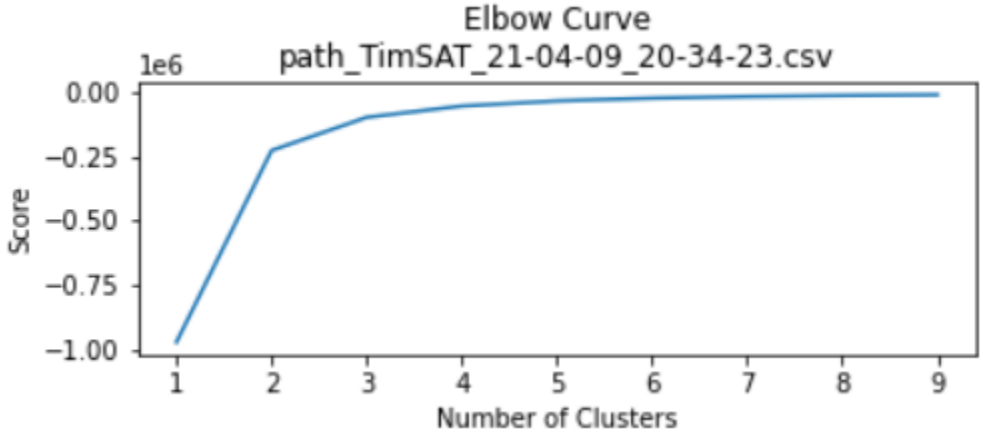
For n_clusters=2, The Silhouette Coefficient is 0.644698377953107
For n_clusters=3, The Silhouette Coefficient is 0.6097280860298011
For n_clusters=4, The Silhouette Coefficient is 0.5913042248123969
For n_clusters=5, The Silhouette Coefficient is 0.5804238144463842
For n_clusters=6, The Silhouette Coefficient is 0.5731311465823171
For n_clusters=7, The Silhouette Coefficient is 0.566812128513393
For n_clusters=8, The Silhouette Coefficient is 0.5617535094861289
For n_clusters=9, The Silhouette Coefficient is 0.5580138417679226

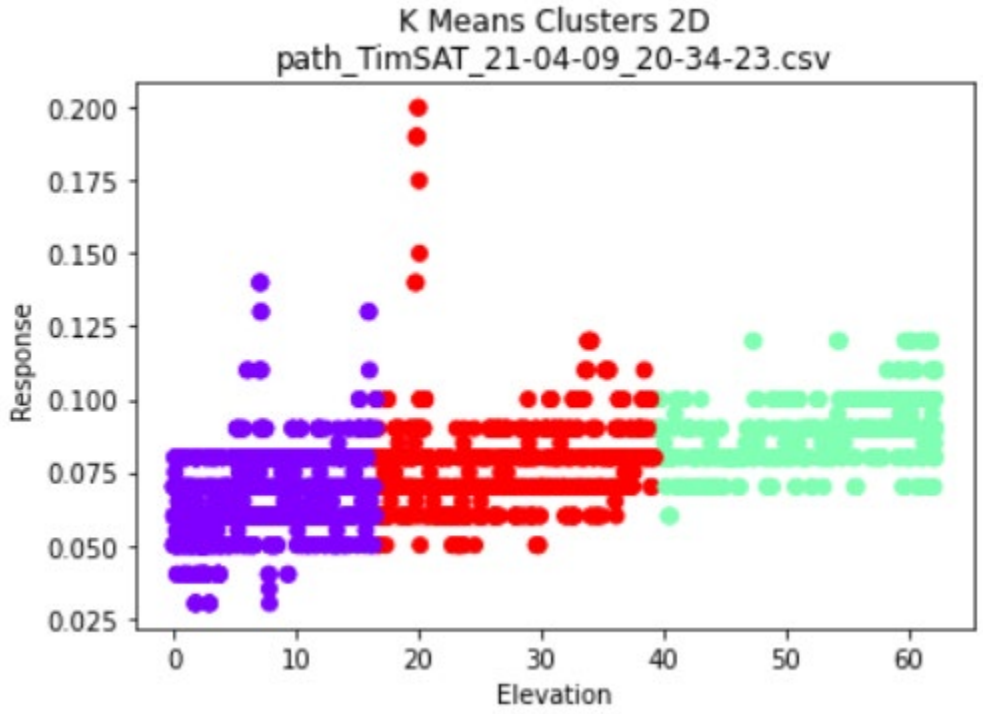




For n_clusters=2, The Silhouette Coefficient is 0.6837009672894909
For n_clusters=3, The Silhouette Coefficient is 0.6257277913803952
For n_clusters=4, The Silhouette Coefficient is 0.598724508870682

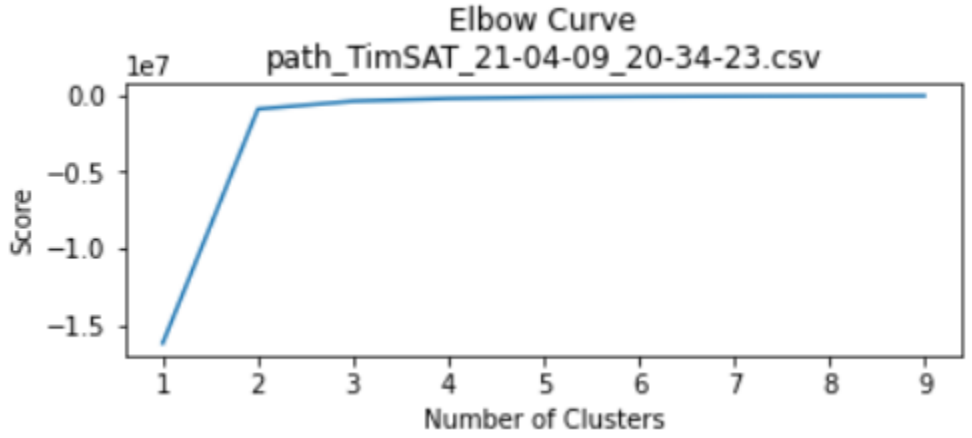
For n_clusters=5, The Silhouette Coefficient is 0.5833338256839933
For n_clusters=6, The Silhouette Coefficient is 0.5722572063121593
For n_clusters=7, The Silhouette Coefficient is 0.5655389257478809
For n_clusters=8, The Silhouette Coefficient is 0.5576799040777747
For n_clusters=9, The Silhouette Coefficient is 0.5541749006087021

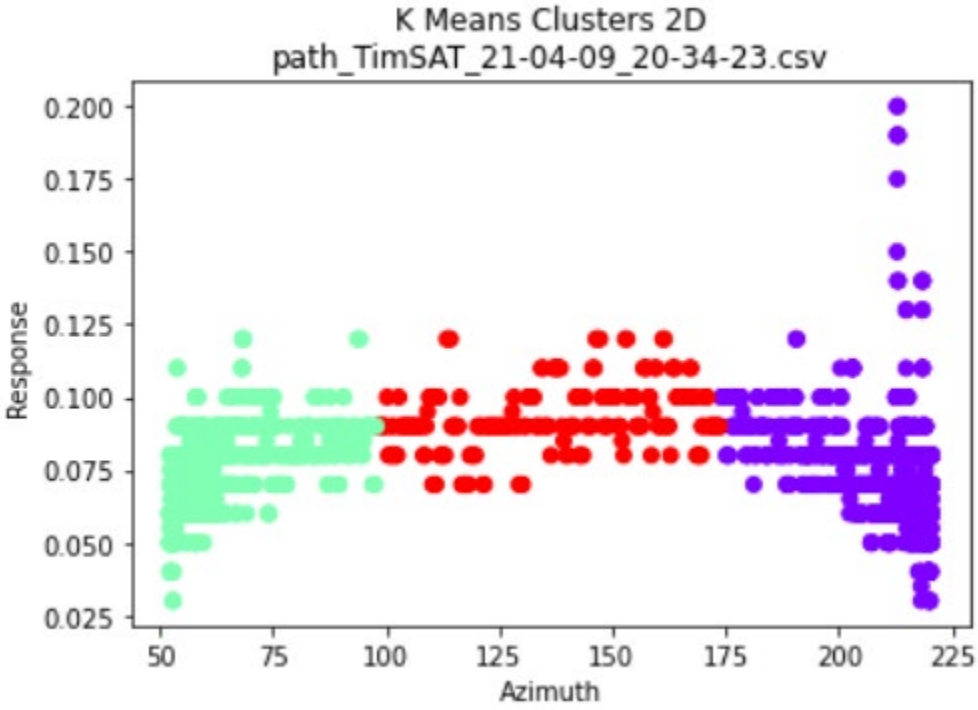




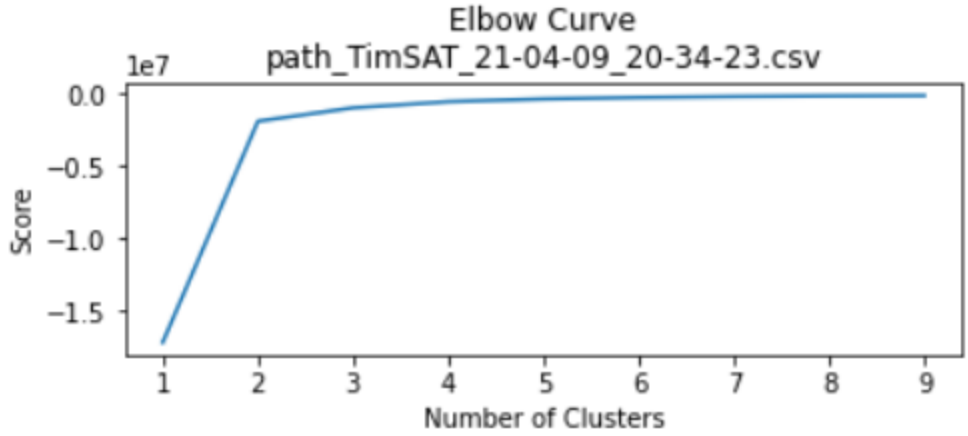
For n_clusters=2, The Silhouette Coefficient is 0.8723867634887597
For n_clusters=3, The Silhouette Coefficient is 0.8250555299954209
For n_clusters=4, The Silhouette Coefficient is 0.777585823614723
For n_clusters=5, The Silhouette Coefficient is 0.740399279802734
For n_clusters=6, The Silhouette Coefficient is 0.710362478002766
For n_clusters=7, The Silhouette Coefficient is 0.6847571773098295
For n_clusters=8, The Silhouette Coefficient is 0.6707186282385402

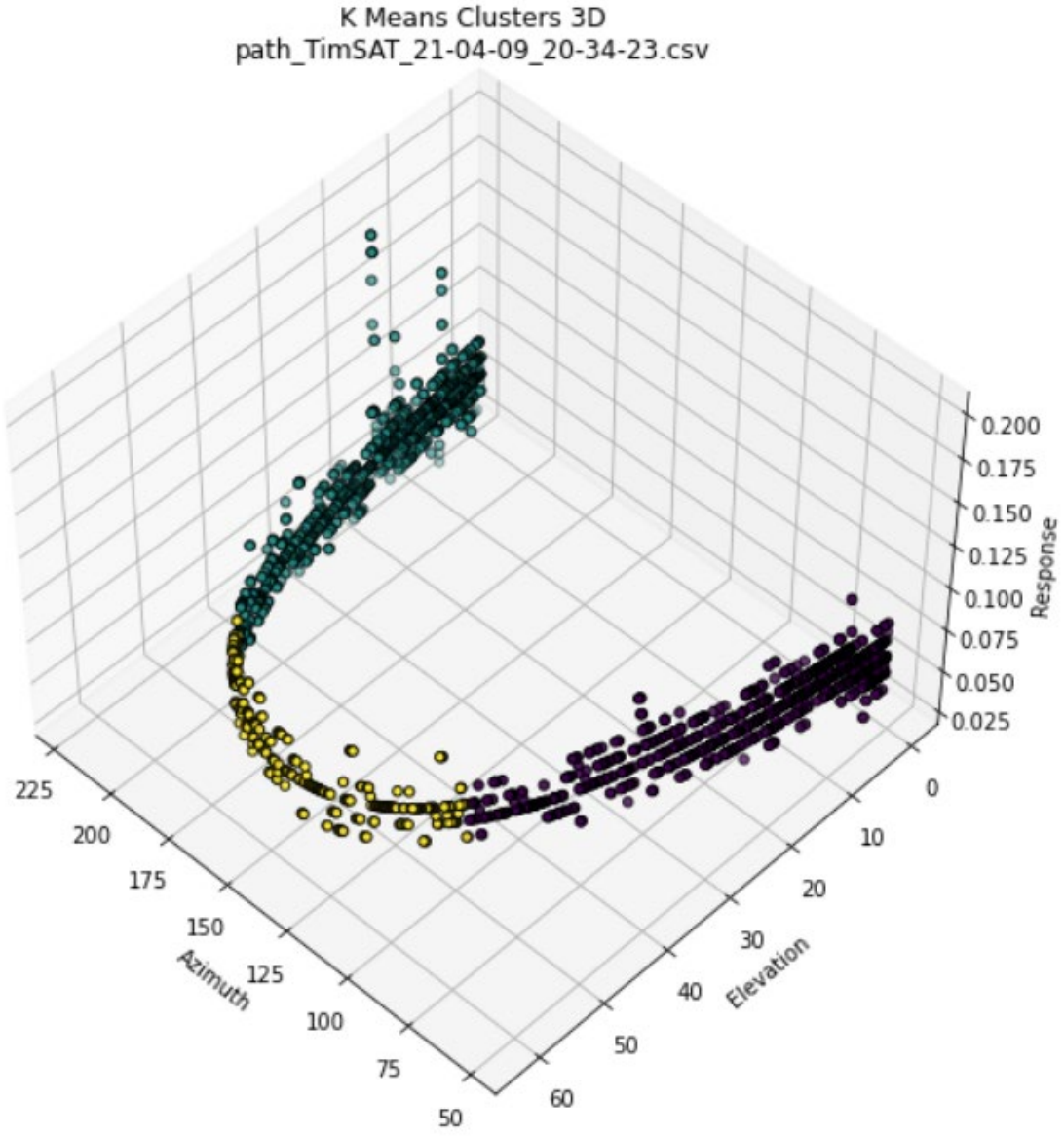
For n_clusters=9, The Silhouette Coefficient is 0.6533137597110338



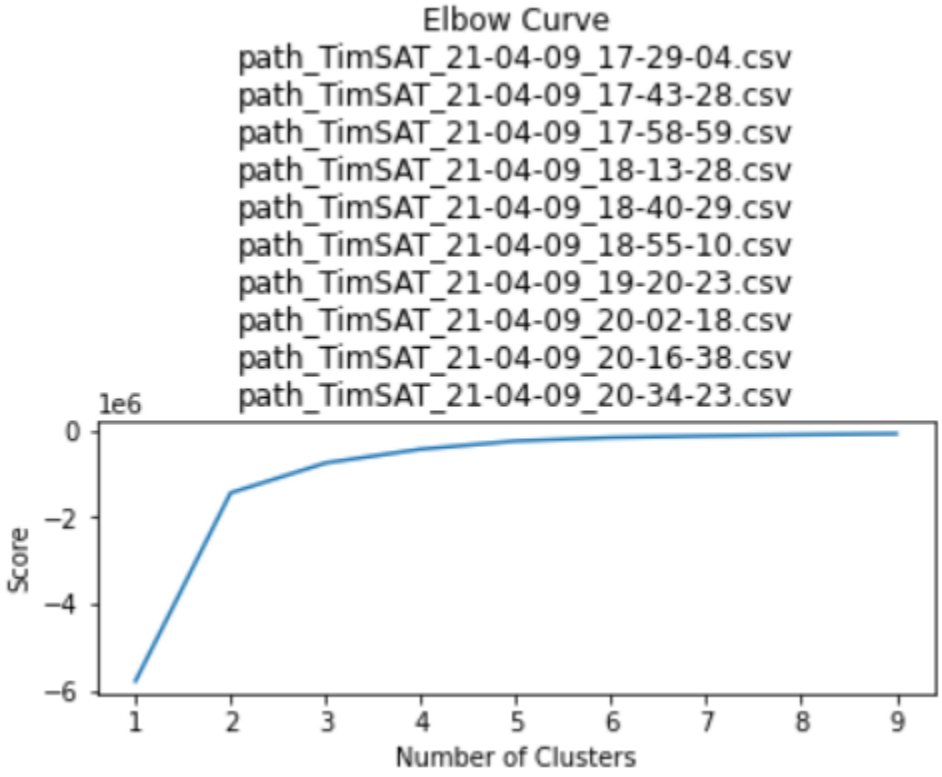


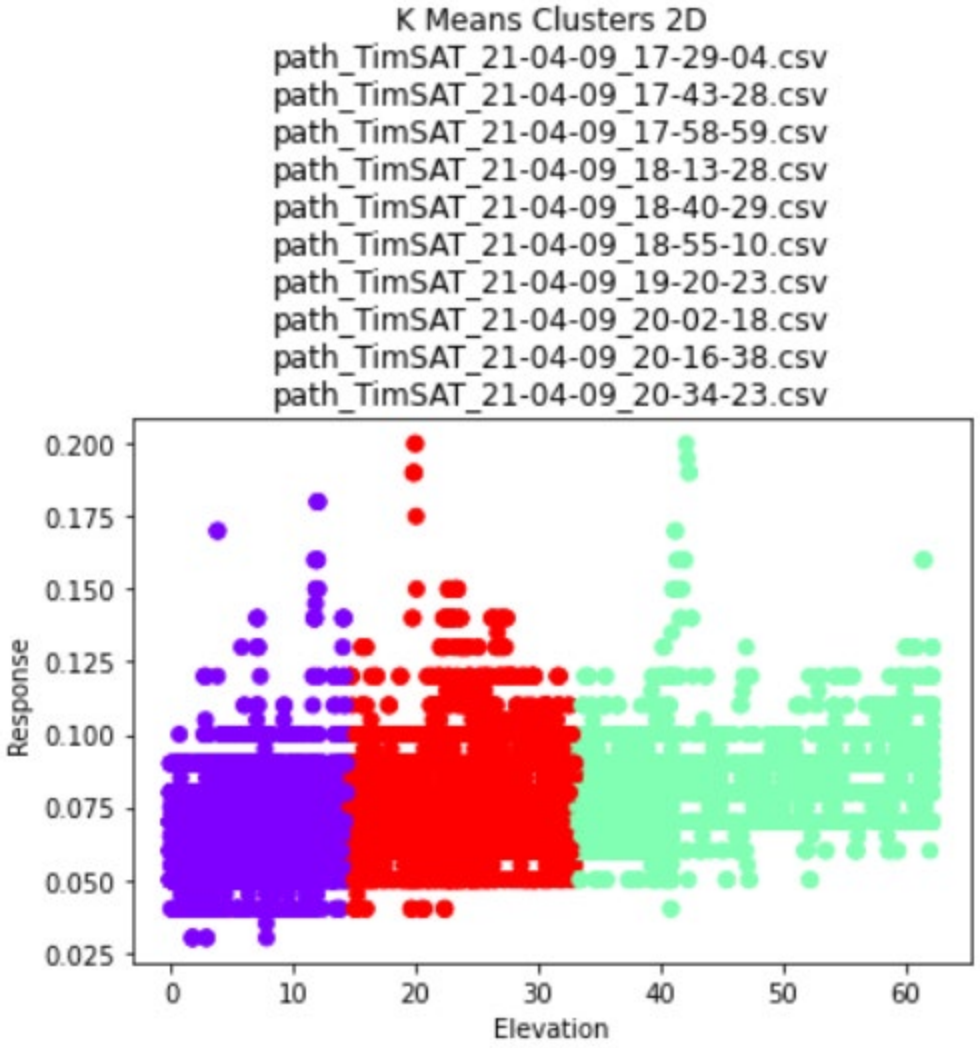
For n_clusters=2, The Silhouette Coefficient is 0.8059714041045408
For n_clusters=3, The Silhouette Coefficient is 0.7483666061416031
For n_clusters=4, The Silhouette Coefficient is 0.6911417933717611
For n_clusters=5, The Silhouette Coefficient is 0.658736978206778
For n_clusters=6, The Silhouette Coefficient is 0.6313289644724451
For n_clusters=7, The Silhouette Coefficient is 0.6133978015452647
For n_clusters=8, The Silhouette Coefficient is 0.598981373950408
For n_clusters=9, The Silhouette Coefficient is 0.5878982437159738


Elbow Curve
path_TimSAT_21-04-09_20-34-23.csv
1e7
0.0
−0.5
−1.0
−1.5
Score
1
2
3
4
5
6
7
8
9
Number of Clusters

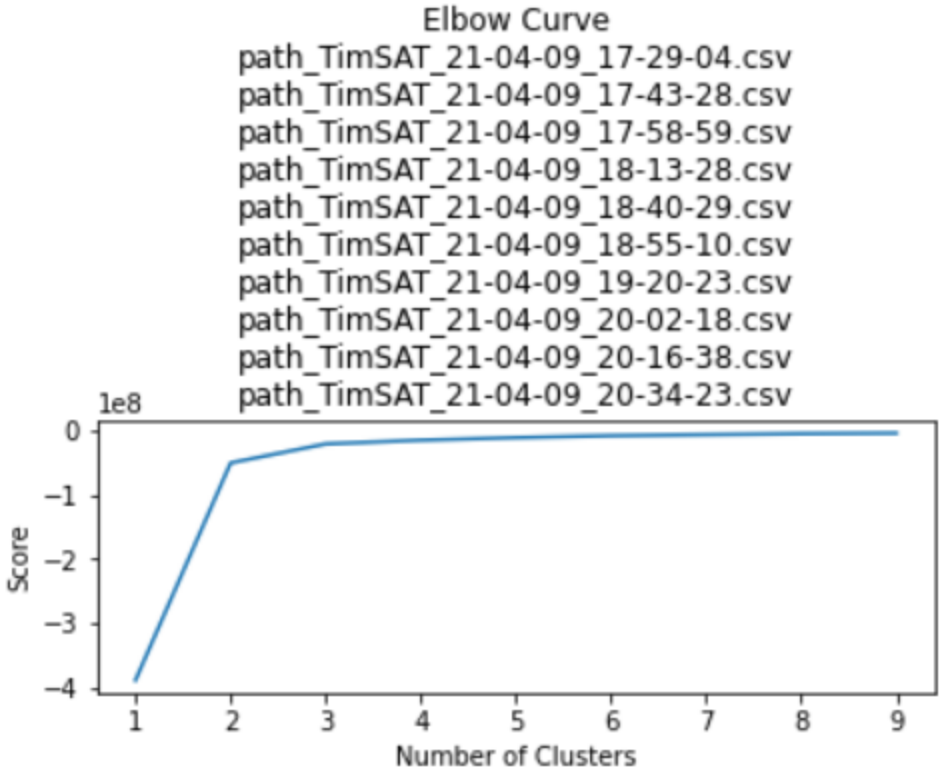

K Means Clusters 3D
path_TimSAT_21-04-09_20-34-23.csv
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
Response
225
200
175
150
125
100
75
50
Azimuth
0
10
20
30
40
50
60
Elevation

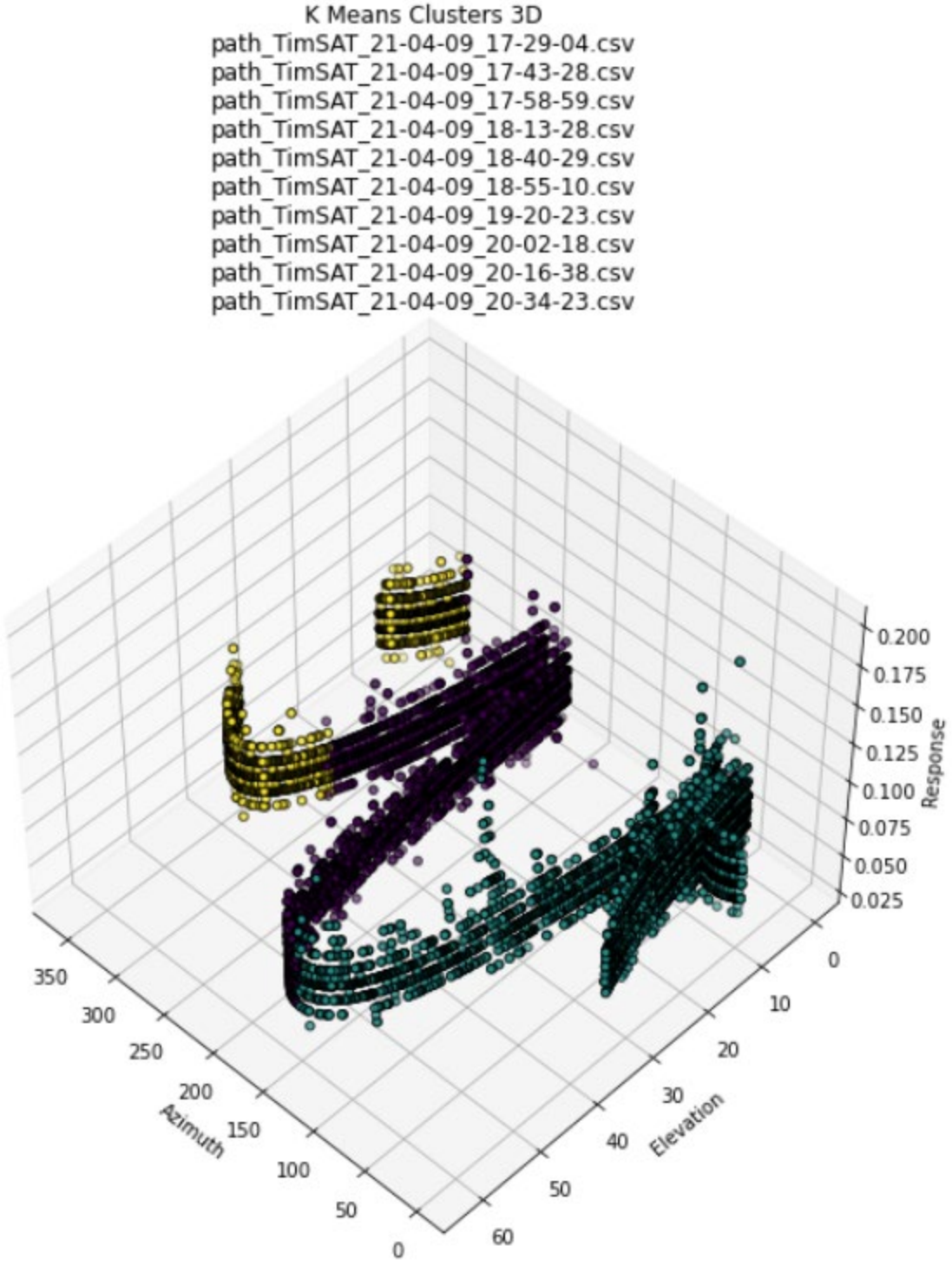
For n_clusters=2, The Silhouette Coefficient is 0.7090767334437937
For n_clusters=3, The Silhouette Coefficient is 0.637482951423615
For n_clusters=4, The Silhouette Coefficient is 0.6195271193574252
For n_clusters=5, The Silhouette Coefficient is 0.5882943661748177
For n_clusters=6, The Silhouette Coefficient is 0.5927358139148533
For n_clusters=7, The Silhouette Coefficient is 0.5829674194232175
For n_clusters=8, The Silhouette Coefficient is 0.5669226080330054
For n_clusters=9, The Silhouette Coefficient is 0.5700264543613299





For n_clusters=2, The Silhouette Coefficient is 0.7804090158381637
For n_clusters=3, The Silhouette Coefficient is 0.7409086011043108
For n_clusters=4, The Silhouette Coefficient is 0.704986760372301
For n_clusters=5, The Silhouette Coefficient is 0.5257009177799129
For n_clusters=6, The Silhouette Coefficient is 0.5491451985980098
For n_clusters=7, The Silhouette Coefficient is 0.5430333840033923
For n_clusters=8, The Silhouette Coefficient is 0.5635235203801531
For n_clusters=9, The Silhouette Coefficient is 0.5686712533888169


Elbow Curve
path_TimSAT_21-04-09_17-29-04.csv
path_TimSAT_21-04-09_17-43-28.csv
path_TimSAT_21-04-09_17-58-59.csv
path_TimSAT_21-04-09_18-13-28.csv
path_TimSAT_21-04-09_18-40-29.csv
path_TimSAT_21-04-09_18-55-10.csv
path_TimSAT_21-04-09_19-20-23.csv
path_TimSAT_21-04-09_20-02-18.csv
path_TimSAT_21-04-09_20-16-38.csv
path_TimSAT_21-04-09_20-34-23.csv
1e8
0
−1
−2
−3
−4
Score
1
2
3
4
5
6
7
8
9
Number of Clusters


K Means Clusters 3D
path_TimSAT_21-04-09_17-29-04.csv
path_TimSAT_21-04-09_17-43-28.csv
path_TimSAT_21-04-09_17-58-59.csv
path_TimSAT_21-04-09_18-13-28.csv
path_TimSAT_21-04-09_18-40-29.csv
path_TimSAT_21-04-09_18-55-10.csv
path_TimSAT_21-04-09_19-20-23.csv
path_TimSAT_21-04-09_20-02-18.csv
path_TimSAT_21-04-09_20-16-38.csv
path_TimSAT_21-04-09_20-34-23.csv
Response
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
Azimuth
350
300
250
200
150
100
50
0
Elevation
0
10
20
30
40
50
60

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. PYTHON CODE

Appendix B includes the Python analysis code used in dataset three. This code evolved with the datasets.

```
# thesis_ml_nps.py
# Grabs, combines, normalizes, and runs unsupervised machine learning (K-means++)
# Created by: Tim Marczewski (TJM)


#Imports - libraries and toolkits
from mpl_toolkits.mplot3d import Axes3D
#mplot3d
#Matplotlib mplot3d toolkit
#The mplot3d toolkit adds simple 3D plotting capabilities to matplotlib by supplying an axes object that can create a 2D projection of a 3D scene. The resulting graph will have the same look and feel as regular 2D plots.
#https://matplotlib.org/2.2.2/mpl_toolkits/mplot3d/index.html
#Copyright 2002 - 2012 John Hunter, Darren Dale, Eric Firing, Michael Droettboom and the Matplotlib development team; 2012 - 2018 The Matplotlib development team.
import datetime as dt
#The datetime module supplies classes for manipulating dates and times.
#While date and time arithmetic is supported, the focus of the implementation is on efficient attribute extraction for output formatting and manipulation.
#https://docs.python.org/3/library/datetime.html
#Copyright 2001-2021, Python Software Foundation.
import pandas as pd
#pandas
#pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool,built on top of the Python programming language.
#https://pandas.pydata.org/
#Fiscally sponsored project of NumFOCUS.
#The mission of NumFOCUS is to promote open practices in research, data, and scientific computing by serving as a
#fiscal sponsor for open source projects and organizing community-driven educational programs.
import matplotlib.pyplot as plt
#Matplotlib: Visualization with Python
#Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
#https://matplotlib.org/
#Copyright 2002 - 2012 John Hunter, Darren Dale, Eric Firing, Michael Droettboom and the Matplotlib development team; 2012 - 2021 The Matplotlib development team.
#Fiscally sponsored project of NumFOCUS.
```

```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
#scikit-learn - Machine Learning in Python
#Simple and efficient tools for predictive data analysis
#Accessible to everybody, and reusable in various contexts
#Built on NumPy, SciPy, and matplotlib
#Open source, commercially usable - BSD license
#https://scikit-learn.org/stable/
import numpy as np
#Nearly every scientist working in Python draws on the power of NumPy.
#NumPy brings the computational power of languages like C and Fortran to Python, a language much easier to learn and use. With this power comes simplicity: a solution in NumPy is often clear and elegant.
#2019-2020 NumPy. All rights reserved.
#https://numpy.org/
import os

#Functions

#cat_data() - Uses the RECORDDATE in UTC to categorize the hour of the day a sample was collected
#example 2021-02-23 22:01:43.34 will get a cat value of 22
#this can be used to further analyze data based on the time of day it is collected
def cat_data(df):
    index = pd.DatetimeIndex(df['RECORDDATE']) # the column of RECORDDATE is used to index
    df.iloc[index.indexer_between_time('0:00', '0:59:59', include_start=True, include_end=True),1] = 0

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('1:00', '1:59:59', include_start=True, include_end=True),1] = 1

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('2:00', '2:59:59', include_start=True, include_end=True),1] = 2

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('3:00', '3:59:59', include_start=True, include_end=True),1] = 3

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('4:00', '4:59:59', include_start=True, include_end=True),1] = 4
```

```
    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('5:00',          '5:59:59',          include_start=True,
include_end=True),1] = 5

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('6:00',          '6:59:59',          include_start=True,
include_end=True),1] = 6

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('7:00',          '7:59:59',          include_start=True,
include_end=True),1] = 7

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('8:00',          '8:59:59',          include_start=True,
include_end=True),1] = 8

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('9:00',          '9:59:59',          include_start=True,
include_end=True),1] = 9

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('10:00',         '10:59:59',         include_start=True,
include_end=True),1] = 10

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('11:00',         '11:59:59',         include_start=True,
include_end=True),1] = 11

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('12:00',         '12:59:59',         include_start=True,
include_end=True),1] = 12

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('13:00',         '13:59:59',         include_start=True,
include_end=True),1] = 13

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('14:00',         '14:59:59',         include_start=True,
include_end=True),1] = 14

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('15:00',         '15:59:59',         include_start=True,
include_end=True),1] = 15

    index = pd.DatetimeIndex(df['RECORDDATE'])
```

```
    df.iloc[index.indexer_between_time('16:00', '16:59:59', include_start=True,
include_end=True),1] = 16

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('17:00', '17:59:59', include_start=True,
include_end=True),1] = 17

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('18:00', '18:59:59', include_start=True,
include_end=True),1] = 18

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('19:00', '19:59:59', include_start=True,
include_end=True),1] = 19

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('20:00', '20:59:59', include_start=True,
include_end=True),1] = 20

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('21:00', '21:59:59', include_start=True,
include_end=True),1] = 21

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('22:00', '22:59:59', include_start=True,
include_end=True),1] = 22

    index = pd.DatetimeIndex(df['RECORDDATE'])
    df.iloc[index.indexer_between_time('23:00', '23:59:59', include_start=True,
include_end=True),1] = 23

    return df


def import_env_data(data): # the import environmental data function works to import the
environmental data from the ENVIROMUX 5D sensors. The data is imported as a CSV
file
    data_env = pd.read_csv(data, sep='\t') # the data for this CSV file is tab dilinitated
    df = pd.DataFrame(data_env)
    df = df.drop(columns=['UTC'])
    df['RECORDDATE']= pd.to_datetime(df['RECORDDATE'], format="%d.%m.%Y,
%H:%M:%S.%f")

    df.rename(columns = {'EXTSENS1TMP':'dome_tmp',
                'EXTSENS1RH':'dome_rh',
```

```
            'EXTSENS1DEW':'dome_dew',
            'EXTSENS2TMP':'dome_box_tmp',
            'EXTSENS2RH':'dome_box_rh',
            'EXTSENS2DEW':'dome_box_dew',
            'EXTSENS3TMP':'hpa_pwr_tmp',
            'EXTSENS3RH':'hpa_pwr_rh',
            'EXTSENS3DEW':'hpa_pwr_dew',
            'EXTSENS4TMP':'hpa_tmp',
            'EXTSENS4RH':'hpa_rh',
            'EXTSENS4DEW':'hpa_dew',
            'INTTMP':'envm_tmp',
            'INTRH':'envm_rh',
            'INTDEW':'envm_dew',
            'SERIALNUM':'cat'},
        inplace=True)

    df = cat_data(df) #Call the categorize data function

    #df = df.set_index('RECORDDATE')

    return df # The data file is formatted and turned into a data frame which can then be exported to the main function

def import_vibe_data(data): # Import vibration data function works at importing the vibration data collected on the NPS satellite dish. That is important as a CSV file.
    data_vibe = pd.read_csv(data, sep='\t') # To CSV file is tab diliniated
    df = pd.DataFrame(data_vibe)

#########################################################################
####
    data_date = '2021-04-09' ############################UPDATE WITH DATA DATE!

#########################################################################
####
    df.columns = ['RECORDDATE','Response']

    df['RECORDDATE'] = data_date + ' ' + df['RECORDDATE'].astype(str) # The date format collected on the path data was not consistent with the other data collected. This required re-formatting of the datetime

    df['RECORDDATE']= pd.to_datetime(df['RECORDDATE']) # converted from string to datetime object

    df.insert(1, 'cat', 100) # the category column is inserted in the data frame
```

```
    df = cat_data(df) #Call the categorize data function

    return df

def import_path_data(data): # the import tathata function format the past data collected during a satellite pass on the NPS satellite dish. Data is imported as a CSV file
    data_path = pd.read_csv(data, sep=',') #The CSV file is, delineated
    df = pd.DataFrame(data_path)
    df.rename(columns = {'Time':'RECORDDATE'}, inplace=True)

    df["RECORDDATE"] = pd.to_datetime(df["RECORDDATE"], format="%Y-%m-%dT%H:%M:%S.%fZ") # The date format collected on the path data was not consistent with the other data collected. This required re-formatting of the datetime
    df.insert(1, 'cat', 100) # the category column is inserted in the data frame

    df = cat_data(df) # call the categorize data function

    return df

def cluster_2d(df,n,x,y,csv): #2 dimensional clustering using K means unsupervised machine learning

    g = df.copy()
    g = g[[x,y]]

    n_cluster = range(1, 10)
    kmeans = [KMeans(n_clusters=i).fit(g) for i in n_cluster]
    scores = [kmeans[i].score(g) for i in range(len(kmeans))]
    for i in range(2,10):
        kmeans = KMeans(n_clusters=i).fit(g)
        label = kmeans.labels_
        sil_coeff = silhouette_score(g, label, metric='euclidean')
        print("For n_clusters={}, The Silhouette Coefficient is {}".format(i, sil_coeff))

    fig, ax = plt.subplots(figsize=(6,2))
    ax.plot(n_cluster, scores)
    plt.xlabel('Number of Clusters')
    plt.ylabel('Score')
    plt.title('Elbow Curve\n' + csv)
    plt.show();

    kmeans = KMeans(n_clusters=n) #Add n to define number of clusters#####################################
    kmeans.fit(g)
```

```
    clusters=g.copy()
    clusters['cluster_pred']=kmeans.fit_predict(g)

    plt.scatter(clusters[x],clusters[y], c=clusters['cluster_pred'],cmap='rainbow')
    plt.xlabel(x)
    plt.ylabel(y)
    plt.title('K Means Clusters 2D\n' + csv)
    plt.show()

def cluster_3d(df,n,x,y,z,csv): # 3 dimensional clustering using K means unsupervised machine learning

    h = df.copy()
    h = h[[x,y,z]]

    n_cluster = range(1, 10)
    kmeans = [KMeans(n_clusters=i).fit(h) for i in n_cluster]
    scores = [kmeans[i].score(h) for i in range(len(kmeans))]
    for i in range(2,10):
        kmeans = KMeans(n_clusters=i).fit(h)
        label = kmeans.labels_
        sil_coeff = silhouette_score(h, label, metric='euclidean')
        print("For n_clusters={}, The Silhouette Coefficient is {}".format(i, sil_coeff))


    fig, bx = plt.subplots(figsize=(6,2))
    bx.plot(n_cluster, scores)
    plt.xlabel('Number of Clusters')
    plt.ylabel('Score')
    plt.title('Elbow Curve\n' + csv)
    plt.show();

    km = KMeans(n_clusters=n)
    km.fit(h)
    km.predict(h)
    labels = km.labels_
    #Plotting
    fig = plt.figure(1, figsize=(7,7))
    ax = Axes3D(fig, rect=[0, 0, 0.95, 1], elev=48, azim=134)
    ax.scatter(h.iloc[:,0], h.iloc[:,1], h.iloc[:,2],
               c=labels.astype(np.float), edgecolor="k")
    ax.set_xlabel(x)
    ax.set_ylabel(y)
    ax.set_zlabel(z)
```

```
    plt.title('K Means Clusters 3D\n' + csv);



#Possible future work
#Saving data by sampling higher only during passes
#Vibe data noise floor, sampling, Highlighting the spikes
#Nominal State vs Pass State

# MAIN - used as the main code to call other functions
def main():

    csv_path = './data_9apr/path'
    csv_vibe = 'data_9apr/vibe/combined_vibe.csv'
    csv_env = 'data_9apr/env/env_data.csv'

    path_csv = os.listdir(csv_path) # individual CSV files called for analysis
    vibe_csv = csv_vibe # individual CSV files called for analysis
    env_csv = csv_env # individual CSV files called for analysis


    #path_csv = ['23FEB21/path_TimSAT_21-02-23_21-43-24.csv', '23FEB21/
path_TimSAT_21-02-23_22-00-43.csv','23FEB21/path_TimSAT_21-02-23_22-16-
54.csv'] # individual CSV files called for analysis
    #vibe_csv = ''
    #env_csv = '23FEB21/Tim_env_data_02_23_2021.csv' # individual CSV files called for
analysis

    df_env = import_env_data(env_csv) # calling the import environmental data function to
create the enviornment dataframe



    df_path_vibe_all = pd.DataFrame() #Creating an empty data frame that will later be used
to aggregate all the data frames from each individual pass

    length_path = len(path_csv) # set number of passes to be analyzed, based off of number
of CSV files called

    df_vibe = import_vibe_data(vibe_csv) # select the individual vibration CSV that
corresponds to the path data
    df_env = import_env_data(env_csv) # calling the import environmental data function to
create the enviornment dataframe

    df_vibe = df_vibe.set_index('RECORDDATE') # set the index to the record date
    df_env = df_env.set_index('RECORDDATE')
```

```
    #print(df_vibe)
    #print(df_env)
    for itr in range(length_path): # iterate across the number of CSV files  called

        df_path = import_path_data(csv_path + '/' + path_csv[itr]) #select an individual path CSV
        min_path = df_path['RECORDDATE'].min() # find the earliest record date in the pass data
        max_path = df_path['RECORDDATE'].max() # find the latest record date in the pass data


        #df_vibe = import_vibe_data(vibe_csv) # select the individual vibration CSV that corresponds to the path data

        #print(df_vibe)


        df_path = df_path.set_index('RECORDDATE') # set the index to the record date
        #df_vibe = df_vibe.set_index('RECORDDATE') # set the index to the record date
        #df_env = df_env.set_index('RECORDDATE')
        #print(df_vibe)


        df_vibe_hold = df_vibe.loc[df_path.index.min():df_path.index.max()] # Size the vibration dataframe to within the time frame of the pass
        #df_vibe = df_vibe.loc[min_path:max_path]

        df_env_hold = df_env.loc[df_path.index.min():df_path.index.max()] # Size the environmental dataframe to within the time frame of the pass
        #df_env = df_env.loc[min_path:max_path]

        #print(df_env)

        #print(df_vibe)

        df_path_vibe = [df_path,df_vibe_hold,df_env_hold] # create a series of data frames
        df_path_vibe = pd.concat(df_path_vibe) # imput series of data frames into concatenate function
        df_path_vibe = df_path_vibe.sort_index() #Sort the concatenated dataframe by RECORDDATE index
```

```
    df_path_vibe = df_path_vibe.loc[df_path_vibe['cat'] != 100] #Remove error category data
    df_path_vibe = df_path_vibe.interpolate(limit_direction='both') # interpolate to fill in not a number (NaN) values where linearly interpolated values
    #df_path_vibe = df_path_vibe.fillna(method='bfill')
    #df_path_vibe = df_path_vibe.fillna(method='ffill') # fill in backwards to fill any NaN values left after interpolation

    if df_path_vibe_all.empty: # check to see if the path 5 all data frame used to aggregate all the passes is empty. This indicates the first run through the loop

        df_path_vibe_all = df_path_vibe # set the path vibol data frame to the value of the first iteration through the for loop

    else:

        df_path_vibe_holder = [df_path_vibe_all,df_path_vibe]
        df_path_vibe_all = pd.concat(df_path_vibe_holder) # adds the current iteration data to last pass data to aggregate the values together


    #print(df_path_vibe_all)

    x = 'Range'
    y = 'Response'
    n = 3
    cluster_2d(df_path_vibe,n,x,y,path_csv[itr]) # run the 2D cluster and plot using the 2d cluster function

    x = 'Elevation'
    y = 'Response'
    n = 3
    cluster_2d(df_path_vibe,n,x,y,path_csv[itr]) # run the 2D cluster and plot using the 2d cluster function

    x = 'Azimuth'
    y = 'Response'
    n = 3
    cluster_2d(df_path_vibe,n,x,y,path_csv[itr]) # run the 2D cluster and plot using the 2d cluster function

    x = 'Azimuth'
    y = 'Elevation'
    z = 'Response'
    n = 3
```

```
        cluster_3d(df_path_vibe,n,x,y,z,path_csv[itr]) # run the 3D cluster and plot using the 3d cluster function

    x = 'Azimuth'
    y = 'Elevation'
    z = 'Response'
    n = 3

    joined_path_csv= "\n".join(path_csv)

    cluster_2d(df_path_vibe_all,n,y,z,joined_path_csv)

    cluster_3d(df_path_vibe_all,n,x,y,z,joined_path_csv) # run the 3D cluster and plot the aggregated data from all selected passes using the 3d cluster function


    #df_path_vibe_all.to_csv(r'test2.csv', index = True)
    #print(df_path_vibe_all)

if __name__ == "__main__":
    main()
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Ayodele, T. O. (2010, February 1). *Types of machine learning algorithms.* In IntechOpen. https://doi.org/10.5772/9385.

CyberPower. (2021). PDU81001 - Switched Metered-by-Outlet PDU Series. CyberPower - PDU81001. https://www.cyberpowersystems.com/product/pdu/switched-mbo/pdu81001.

Haddock, M. (2016, February 2). Inductive Monitoring Systems: A CubeSat Ground-Based Prototype. DigitalCommons@CalPoly. https://doi.org/10.15368/theses.2015.164.

Hunter, J., Droettboom, M., Eric Firing, & Dale, D. (2021). Matplotlib: Visualization with Python. Matplotlib. https://matplotlib.org.

Iverson, D. L. (2004, June). 2004 International Conference on Artificial Intelligence (IC-AI'04). In *CSREA Press* (pp. 605–611). Las Vegas. Retrieved June 15, 2020, from https://ti.arc.nasa.gov/m/groups/intelligent-data-understanding/ICAI2004-Iverson.pdf.

Kelleher, J. D., MacNamee, B., D'Arcy, A. (2020). *Fundamentals of machine learning for predictive data analytics: Algorithms, worked examples, and case studies*. MIT Press.

Minelli, G., Magallanes, L., Weitz, N., Rigmaiden, D., Horning, J., Newman, J., … Oakden, R. (2019, July 16). The Mobile CubeSat Command and Control (MC3) Ground Station Network: An Overview and Look Ahead. DigitalCommons@USU. https://digitalcommons.usu.edu/smallsat/2019/all2019/85/.

Network Technologies Inc. (2021). Network Technologies Inc. Environment Monitoring System Server Room Temperature I.P. Sensor Alert. Retrieved January 20, 2021, from https://www.networktechinc.com/environment-monitor-5d.html.

NumPy. (2021). *NumPy*. https://numpy.org/.

pandas. (2021). *Pandas.* https://pandas.pydata.org/.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, É. (2011, November 1). Scikit-learn: Machine Learning in Python. The Journal of Machine Learning Research. https://dl.acm.org/doi/10.5555/1953048.2078195.

Python Software Foundation. (2021). Python.org. https://www.python.org/.

Sensaphone. (2021). 4–20mA Type Vibration Sensor. Sensaphone - 4–20mA Type Vibration Sensor. Retrieved January 20, 2021, from https://www.sensaphone.com/products/4-20ma-type-vibration-sensor.php.

Singh, S. (2018, January 3). A Data-Driven Approach to Cubesat Health Monitoring. DigitalCommons@CalPoly. https://doi.org/10.15368/theses.2017.100.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California