Scatterometer L1B Data Product Flag Verification Using ML Technique

A PROJECT REPORT

Submitted by

Dhairvi Shah

180320132051

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

In

Information & Communication Technology Department

L. J. Institute of Engineering & Technology

Ahmedabad

Gujarat Technological University, Ahmedabad

May, 2022

L. J. Institute of Engineering & Technology

Ahmedabad

CERTIFICATE

This is to certify that the project report submitted along with the project entitled **Scatterometer L1B Data Product Flag Verification Using Ml Technique** has been carried out by **Dhairvi Shah** under my guidance in partial fulfillment for the degree of Bachelor of Engineering in **Information & Communication Technology**, 8th Semester of Gujarat Technological University, Ahmedabad during the academic year 2021-22.

Prof. Ankita Tiwari	Prof. Mitesh Thakkar
Internal Guide	**Head of the Department**

L. J. Institute of Engineering & Technology
Ahmedabad

DECLARATION

We hereby declare that the Internship report submitted along with the Internship entitled **Scatterometer L1B Data Product Flag Verification Using ML Technique** is in partial fulfillment for the degree of Bachelor of Engineering in **Information & Communication Technology** to Gujarat Technological University, Ahmedabad. It is a bonafide record of original project work carried out by me at SAC-ISRO under the supervision of Scientist. Anuja Sharma and that no part of this report has been directly copied from any students' reports or taken from any other source, without providing due reference.

Name of the Student | Sign of Student

Dhairvi Shah

I. ACKNOWLEDGEMENT

It gives me great joy to be able to present my gratitude to all those who guided, helped and encouraged me throughout the project. Because of them, I was able to uphold my practical skills in solving real world problems.

To start with, I would like to thank my family for supporting me throughout the period and providing with encouragement, enthusiasm and assistance. Without them I might have not be able to complete this efficiently.

Furthermore, I would like to present my gratitude towards Prof. Mitesh Thakkar, H.O.D of Information and Communication Technology for their immense support and for giving us this excellent opportunity to pursue internship program as well as Prof. Ankita Tiwari who played a major role in supervising and mentoring my work throughout the period.

In addition to this, I would like to present my sincere gratitude to Dr. B Kartikeyan, Head of Image analysis and Quality evaluation Division (IAQD) at Space Application Centre (SAC) – Indian Space Research Organization (ISRO) for giving me this opportunity to work in his department and pushed me to give my best. Scientist. Anuja Sharma, my external guide who guided, encouraged, helped and mentored me throughout this internship. I thank her for walking me through this entire successful period without whom I might not have presented this work. She also uplifted my other skills that will prove to be invaluable and significant throughout my life.

I would like to give my sincere regards to Ms. Alka Saini, a colleague of mine for all the help and guidance she provided to me. She helped me throughout my journey on every step.

Finally, I would like thank all those unnamed people who have helped me in an unknown way in accomplishing my tasks.

II. ABSTRACT

Scatsat-1 Scatterometer is a microwave non - imaging radar aimed to measure backscattered energy over the ocean surface. Backscattered energy helps in deriving wind vector – speed and direction at the spatial resolution of nearly 25 X 25 Km cell. Data received from satellite goes through a set of data processing steps and Level-1B data product is generated in HDF format. This data product includes Sigma-0 values, brightness temperature with precise location, Meta information, geometry related parameters and Sigma-0 flag for each footprint covered on Earth surface.

Data Quality Evaluation System is a system responsible to evaluate the quality of data product independently and generates quality metrics for the same. Apart from quality metrics, we can depict Sigma-0 flag that is an important parameter packed in data product from which node, beam, scan-direction, sigma-0 quality, sigma0 validity, sea-land boundary details, sea-ice information etc. are received. During data processing, this information is tagged with each footprint using various methods, criteria and references.

Under data quality evaluation, we need to verify these flags using machine-learning techniques. The proposed model in this report uses historic data from Scatsat-1, and after applying ML techniques on this data, the model can verify existing flags and predict them for future missions.

III. LIST OF FIGURES

IV. LIST OF TABLES

V. ABBREVIATIONS

SAC	Space Application Centre
ISRO	Indian Space Research Organization
SIPG	Signal and Image Processing Group
IAQD	Image Analysis and Quality evaluation Division
DQE	Data Quality Evaluation
HH	Horizontally transmitted and Horizontally received (Inner Beam)
VV	Vertically transmitted and Vertically received (Outer Beam)
L1B	Level – 1B
HDF/.h5	Hierarchical Data Format

VI. TABLE OF CONTENETS

Chapter 1 OVERVIEW OF THE COMPANY

1.1 HISTORY

Space Application Centre (SAC) – Indian Space Research Organization (ISRO)

Indian Space Research Organization, formed in 1969, Vikram Sarabhai, having identified the role and importance of space technology in a Nation's development, provided ISRO the necessary direction to function as an agent of development. ISRO then embarked on its mission to provide the Nation space based services and to develop the technologies to achieve the same independently. Throughout the years, ISRO has upheld its mission of bringing space to the service of the common person, to the service of the Nation. In the process, it has become one of the six largest space agencies in the world.

The genesis of the Centre dates back to 1966, with establishment of the Experimental Satellite Communication Earth Station (ESCES), by late Dr. Vikram Sarabhai in Ahmedabad. In 1972, the different units of ISRO in Ahmedabad pursuing research in applications of space technology were merged to form Space Applications Centre (SAC). Eminent Scientist Prof. Yashpal was the first Director of SAC. The success journey of SAC started with a unique experiment called the Satellite Instructional Television Experiment (SITE), which was conducted by SAC/ISRO during 1975-76. Hailed as 'the largest techno-social experiment in the world', SITE demonstrated the potential of satellite technology as an effective mass communication media, aimed at socio-economic development of rural India.

Fig 1.1.1 Company Logo

Space Applications Centre (SAC) is a major research and development Centre of the Indian Space Research Organization (ISRO). SAC has expertise in the design of space-borne and air-borne instruments for various ISRO missions and development & operationalization of applications of space technologies ranging from communication, broadcasting, navigation, disaster monitoring, meteorology, oceanography, environment monitoring to natural resources management. Located at Ahmedabad, SAC is spread across three campuses having multi-disciplinary activities apart from Delhi Earth Station (DES), which is located in New Delhi.

1.2 ORGANIZATION CHART

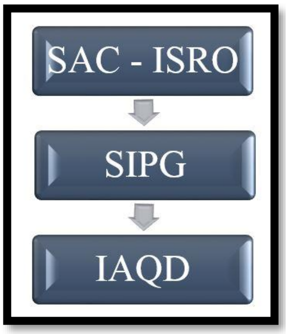


Fig 1.2.2 Organization Chart

SAC – ISRO has in total 13 departments all working according to their role. I am currently pursuing my internship at Signal and Image Processing Group (SIPG) department in Image Analysis and Quality Evaluation Division (IAQD), one of its subdivisions.

The SIPG – IAQD department is responsible to carry out three types of activities related to DQE (Data Quality Evaluation). They are:

a) Design and development of techniques and software for DQE
b) Participate in sensor calibration and process qualification activities at initial phase
c) Participate in detailed analysis of quality parameters if any anomaly occurs in operational phase

DQE is a broad activity carried out with an objective of monitoring the in-orbit performance of sensor, quantification of data quality, raising alarm in case of anomaly and providing feedback to payload, data products and other teams for ensuring end – quality of data products.

Chapter 2 INTRODUCTION TO INTERNSHIP AND INTERNSHIP MANAGEMENT

2.1 INTERNSHIP SUMMARY

I pursued my last semester internship from SAC – ISRO from 1st Dec 2021 to May 2022. I worked at IAQD/SIPG department. I spent my first couple of weeks understanding the working of the department, concepts related to the work and other standard procedures. I learned about

- Remote Sensing and its types
- Sensors
- Working principle of Scatterometer
- Data Products
- Type of files
- Formats
- Coding Standards

After having the important, basic and preliminary knowledge, I optimized some preexisting database codes in order to make them generic so that it can work for both the Oracle and PostgreSQL and tried to acquire skills like:

- Database Connections
- Advance Python
- Plot Making in Python
- LaTeX
- Java
- SQL
- PostgreSQL

Lastly, I started with my project with my enhanced skills and completed that within a given period.

2.2 CONCEPTS TO KNOW

Scatterometer

It is a non-imaging, active, microwave radar. It is Ku band (13.515GHz) Scatterometer. It uses pencil beam. Carries a parabolic reflector antenna that measures. Backscatter strength using dual beams HH (inner beam) and VV (outer beam), covers 1400 km and 1800 km swaths respectively. It considers four points Inner after, Inner fore, Outer after, Outer fore. It is used to obtain wind velocity.

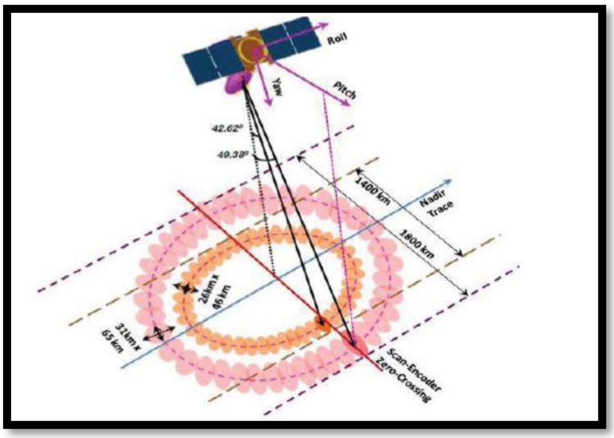


Fig 2.2.3 Scatterometer

Geometrically and radio metrically corrected and packed, this data can be defined as Data Products which are synchronized for time and then data is separated ½ revolution wise. After necessary calculations with the raw data different levels of data product is obtained. The levels of data products are as shown in the following Table:

Table 2.2.1 Levels Of Data Products

Name of Levels	**Attribute**
Level – 0	Raw Instrumental Data
Level – 1B	Scan Mode Data
Level – 2A	Swath Grid Data
Level – 3W	Global Grid
Level – 3S	Global Sigma0

The project is based on the Level – 1B Data Product of Scatterometer. It is obtained in HDF format (Hierarchical Data Format). L1B file provides with Sigma0 parameter. Sigma-0 informs us about **the backscattered strength of the target**. It provides us with knowledge about wind vector and weather conditions. This HDF file also contains physical parameters like:

- Attributes
- Science Data
- OAT Data

In addition, Sigma-0 flag is an important parameter packed in Level – 1B data product with some flag bits whose validation with reference data and prediction for fresh data is the moto of the proposed model.

2.3 PURPOSE

Model aims to validate and predict the sigma0 flag bits on the basis of historic data obtained from Scatsat-1 Scatterometer. Application of Machine Learning techniques on the data extracted from Level-1B Data Product helps in fulfilling this purpose. The flag bits predicted by this model are:

i. South-Pole to North-Pole i.e. Ascending node (S-N)
ii. North-Pole to South-Pole i.e. Descending node (N-S)
iii. Aft scan-direction
iv. Fore scan-direction
v. Good/Poor sigma0 or brightness temperature bit
vi. Valid/Invalid sigma0 or brightness temperature bit
vii. Land-Sea boundary

These Sigma0 flag bits inform about the quality of sigma0 for specific slices and footprints on the earth surface where “sigma0” is the backscattered co-efficient that is the measure of backscattered energy from the target towards Scatterometer. Thus, the information related to its quality holds the great importance. This model uses the historic data from ScatSat-1 for validating the flag bits that will also help in predicting those bits for future missions.

2.4 OBJECTIVE

The proposed model has two objectives:

1. It uses the post-calibrated data from Scatsat-1 in order to train the model that is used to validate the flag bits.
2. It uses this trained model to predict the flag bits for the pre calibrated data

Sigma0 parameter that is obtained from Level-1B is the backscattered co-efficient that represents the strength of Scatterometer signal reflected by the distributed scatterer. This Sigma0 can be imperfect as its estimation can be noisy due to the instrument thermal noise and radar signal fading effects. Sigma0 flag bits tell about the quality of this Sigma0 for specific slices and footprints on earth. As a result, this bit information is very important because when this Data Product goes to the user, he should be informed to use it cautiously as there can be a low quality data for land, sea-ice and heavy rain conditions that has to be rejected.

To summarize the model it can be inferred that the model can validate the flag bits for pre-existing data along with predicting them for upcoming missions.

2.5 SCOPE

The HDF file is the basic input file required here. Model obtains data like OAT Data, Scale & Offset values and science data from this file. The original data is in 4-byte (Float) form that utilizes more space. As a result they are compressed to short type that is 2-byte using certain scale and offset value. So, in order to achieve original data again, they are recomputed with those respective values.

Flag bits are extracted from this files and CSV file is generated on which Machine Learning algorithm is applied. As a result, a model is made which can predict those flag values for any other similar like data set.

2.6 INTERNSHIP PLANNING

2.6.1 Internship Development Approach

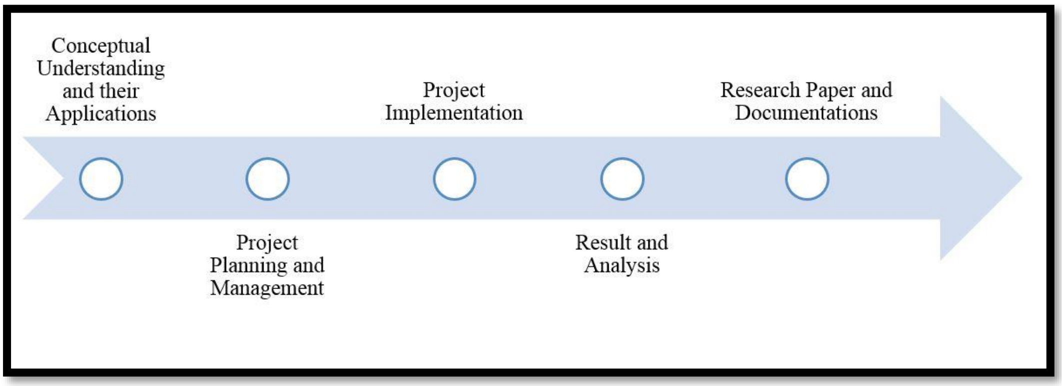


Fig 2.6.1.4 Internship Development Approach

The Internship Development Approach is as shown in Fig 2.6.1.4 given above. As you can see, the approach is divided into 5 parts.

To start with, I tried to understand the concept of my project, usage of data, data types and format of those data. Then I tried to implement small functions on those data, which later on might prove to be useful for project. Hence, the first major part of the internship was to get comfortable with the working environment.

Once I was comfortable, project planning was done by having discussions with guide and preparation of road map which helped me in staying on the correct path. This formed the second part in my internship but base for my project.

The third and fourth phase of my internship was crucial as it involved project implementation and getting optimized and accurate results. I applied new techniques and different ways to get accurate and precise results with lower time and space complexities.

The Last thing I did was to prepare a Research Paper as well as formation of reports and presentations. I followed proper standards for each documentation procedures. Thus, the internship development approach played a major role in helping me to complete my work within the given period.

2.6.2 Roles & Responsibilities

As the name suggests, during my internship I pursued the role of a trainee. I had few responsibilities to carry out which were:

- To learn about the way things were done there
- To know about the system
- To acquire knowledge regarding the type of work I was going to perform
- To work on guidelines provided to me by my external guide
- To work sincerely, enthusiastically and passionately and to make sure to complete the allotted work in the given period.

I made sure to fulfil every responsibility during my time there.

2.7 INTERNSHIP SCHEDULING

The given below Fig 2.7.5 of Gantt Chart depicts my internship scheduling.

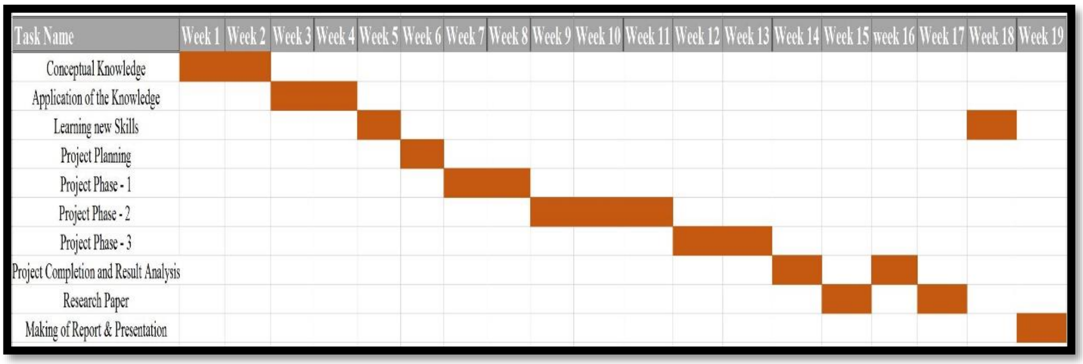


Fig 2.7.5 Gantt Chart

Chapter 3 SYSTEM ANALYSIS

3.1 CURRENT SYSTEM

To start with, the current system did not use machine-learning techniques for estimating flag bits.

In the current system, they manually do the flag bit check using old reference data. Bit value in the flag depicting ascending or descending orbit (Nodes) was set using OAT information. In addition, bit value corresponding to the look direction (Scan – direction Aft-Fore) was set using antenna azimuth angle. Lastly, bit value for beam (VV/HH) is indicated in accordance with the polarization of the footprint within a given composite. For assigning validity of sigma0, the sigma0 value obtained after composition is checked for specified thresholds. If it falls beyond these, the corresponding composite is marked as poor sigma0. Similar treatment is given to computation of brightness temperature information for composite. If all slices of a composite from a footprint are invalid, the composite is flagged with invalid sigma0 and brightness temperature. As they were estimated based on old references and that too manually, it was little tedious as well as time consuming. Thus, the new system was developed in order to overcome those limitations.

Automation is introduced in the new system as the proposed system uses machine learning algorithms where flag bits are estimated independent of OAT parameter, azimuth angle and polarization information. This model uses parameters like Latitude, Longitude, Sigma0, SNR, X-Factor, Incidence Angle and Brightness Temperature for the same purpose.

3.2 SYSTEM FEASIBILITY

3.2.1 Does The System Contribute To The Overall Objectives Of The Organization?

The new system (proposed model) contributes to the overall objectives of the organization. Using Ml techniques, flag bits can be predicted with the accuracy of 99% and thus helps us in estimating or acquiring knowledge related to Weather conditions.
In addition, it is no longer dependent on just Sigma0 but many more parameters, which are much more reliable.

3.2.2 Can The System Be Implemented Using The Current Technology And Within The Given Cost And Schedule Constraints?

The system can be implemented using the current technology. In addition, it is also cost effective. It do fir perfectly within the cost constraint. However, The time complexity has increased in this technique. ML models take great amount of time in training as well as testing the bits.

3.2.3 Can The System Be Integrated With Other Systems That Are Already In Place?

During the whole procedure of implementation of the new system, not a single new module has been installed. It uses the inbuilt modules and functions and hence, it can readily integrate with other system that are already there.

3.3 PROPOSED SYSTEM

Fig 3.3.6 given below shows the overview of the proposed System.

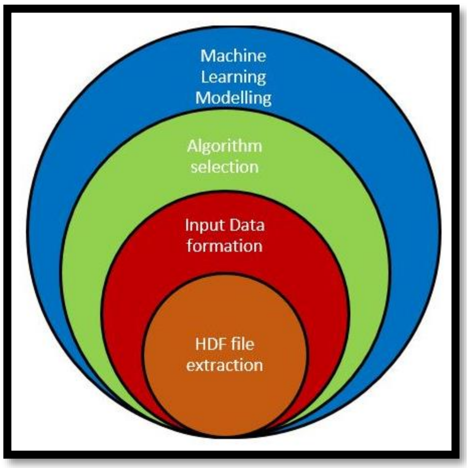


Fig 3.3.6 Proposed Model

Phase1: HDF File Extraction

Extraction of this file provides us with needed physical parameters along with Scale and offset values or say information on attributes are obtained from this file.

Phase2: Input Data Formation

CSV file having bit wise data from the flag file along with some related physical parameter acts as the input data for further steps.

Phase3: Algorithm Selection

Machine learning is the process of meta-learning. Its main aim is to find the algorithm, which will have minimum error rate for the present dataset.

Phase4: Machine-Learning Modelling

For execution of any machine learning technique, there are two main requirements,
Data and Algorithm that applies on that data. Need of the data is fulfilled by step 2 and that of algorithm by step 3. As a result, application of ML technique is possible. Model is trained with 80-90% of the data from the dataset while tested with remaining 10-20% of the dataset.

3.4 APPROACH

3.4.1 Hardware

The hardware used in this proposed model are:

- Central Processor : Intel E5 2630
- Version : 3
- Memory : 768 GB
- Multi core
- 64 bit

3.4.2 Software

The software used in this proposed model are:

- OS : RED hat Linux
- Python
- Spyder
- HDF Viewer
- Microsoft Word

3.4.3 Techniques

Techniques used in this overall approach are:

- Data Collection
- Observation
- Data Analyzing
- Input Data Preparation
- Machine Learning Application
- Accuracy Score
- Result Analysis

Chapter 4 IMPLEMENTATION

4.1 PROCESS

4.1.1 Phase1: HDF File Extraction

As our proposed model is based on Level-1B data and it is present in HDF format, the first and foremost step is to extract those data. As stated earlier 4byte, float data is converted to 2byte short to form Hdf file. As a result, while extracting the same data they should be calculated again using scale and offset values to get original values.

*(Scale * Values) + Offset*

The following figure 4.1.1.7, throws some light on how the HDF file is extracted.

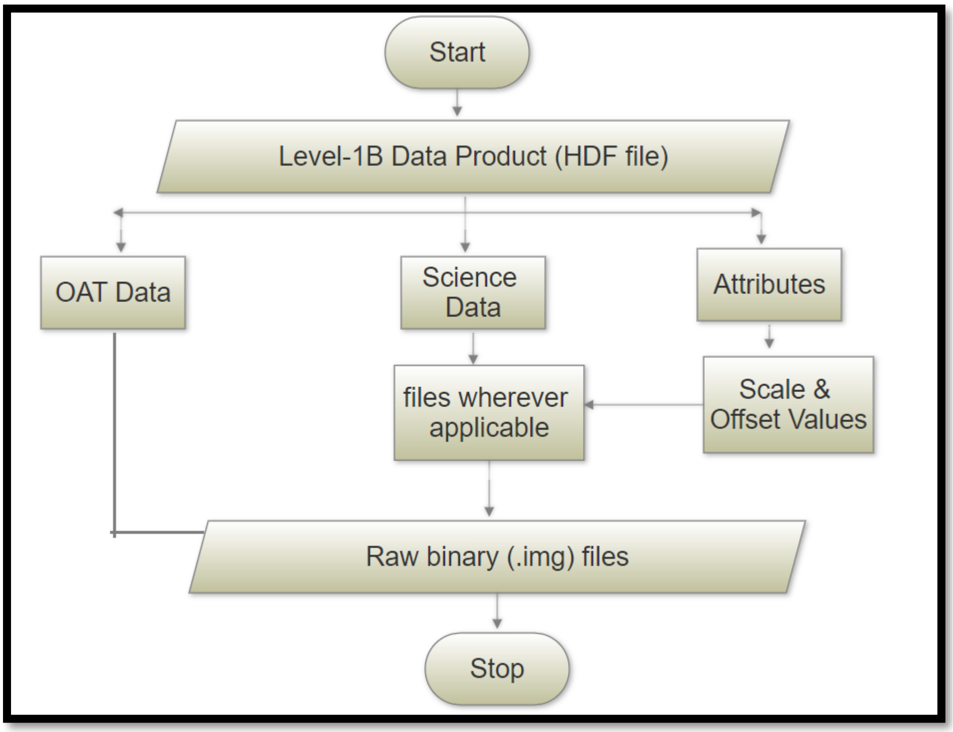


Fig 4.1.1.7 HDF File Extraction

HDF files provides with three type of data. They are:

- OAT Data (Orbit and Attitude data)
    1. Yaw
    2. Pitch
    3. Roll
    4. Position x, y & z
    5. Velocity x, y & z
    6. Altitude

- Science Data
    1. Latitude
    2. Longitude
    3. Sigma0 / Brightness Temperature
    4. Kp
    5. SNR
    6. X-factor
    7. Incidence angle

- Attributes
    1. Scale values
    2. Offset values

The OAT data are simply stored as .img files while respective physical parameters are computed with scale and offset values and then stored as .img file. These binary files acting as output of phase 1 work as input file for phase 2.

4.1.2 Phase2: Input Data Formation

Binary output files of phase one acts as an input here. Those binary files are read here. Sigma0 flag file is the only parameter that is converted to binary form and from which some of the least significant bits are extracted which are used in further phases. Other parameters are kept as it is. The given below Fig 4.1.2.8 Shows some insights of the second Phase.

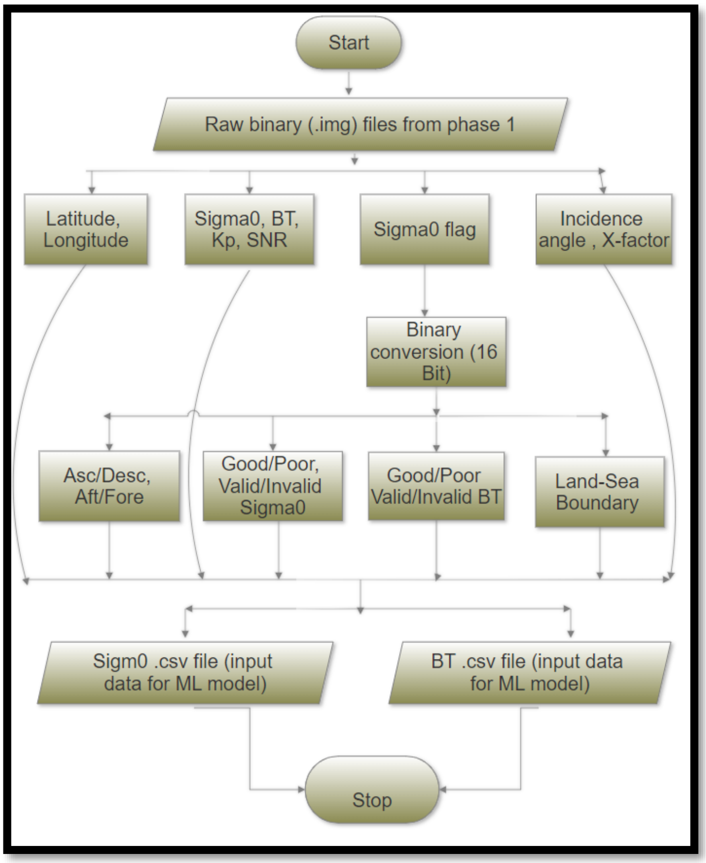


Fig 4.1.2.8 Input Data Formation

The bits sigma0 flag read are [15, 13, 12, 11, 8] for sigma0 and [15, 13, 10, 9, 8] bits for brightness temperature where this bits represent following characteristic:

- 15 – Node
- 13 – Scan - Direction
- 12 – Valid /Invalid Sigm0
- 11 – Goop/Poor Sigma0
- 10 – Valid/Invalid BT
- 9 – Good/Poor BT
- 8 – Sea-Land Boundary

Other than the parameters mentioned before, other parameters like latitude, longitude, Kp, SNR, X-factor and incidence angle are used from science data. CSV file is formed using all those parameters and passed further for machine learning modelling.

In total eight files are formed at the end of phase 2. They are inner/outer slice sigma0 files, inner/outer footprints sigma0, inner/outer slice BT and lastly inner/outer footprints BT.

4.1.3 Phase3: Algorithm Selection

Algorithms are well-stated set of rules for carrying out computational tasks. Our model uses machine-learning techniques, as a result we can choose algorithm from main three types of categories. They are:

- Supervised Learning – We can define it by its use of labelled datasets to train algorithms and to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weight until the model has been fitted accurately.
- Unsupervised Learning – Uses ML to analyze and cluster unlabeled dataset. The algorithm discovers hidden patterns or data group without human intervention. It finds similarity as well as differences in datasets.
- Reinforcement Learning - Ml training method based on rewarding desired behavior and/or punishing undesired ones. It learns through trial and error method.

Supervised Learning proves to be useful here as the input data used for the proposed model is a labelled data set. Here we can have two types of variables:-

- Independent Variables: Variables on which values to be predicted are dependent on.
- Dependent Variables: Variables to be predicted.

The table 4.1.3.2 given below presents list of dependent and independent variables for the proposed model.

Table 4.1.3.2 List of Variables

Dependent variables	**Independent Variables**
Ascending/Descending (Node)	Latitude, Longitude
After/Fore (Scan-Direction)	Kp, SNR
Valid/Invalid Sigma0/BT	X-Factor
Good/Poor Sigma0/BT	Incidence Angle
Land-Sea Boundary	Sigma0 / BT

4.1.4 Phase4: Machine Learning

There are two basic requirements that are prerequisite for applying machine-learning techniques. They are data and algorithm, which will be applied on that data. We obtain input data from phase 2 and algorithmic information from phase 3. Having all the required information, now we can apply ML techniques. Three scenarios are crosschecked to get most accurate results. They are:

- 80% data is considered as training data and 20% data as test data
- 90% data is considered as training data and 10% data as test data
- Node and Scan-Direction are considered independent variable instead of dependent (A/D – A/F)

For all the above three scenarios, two different procedures were considered. They are

1. Multiple Input Single Output

   Here only one dependent variable is predicted at a time. The ML techniques applied here are:

   - K-Neighbors
   - Decision Tree
   - Random Forest
   - Naïve Bayes
   - Logistic Regression

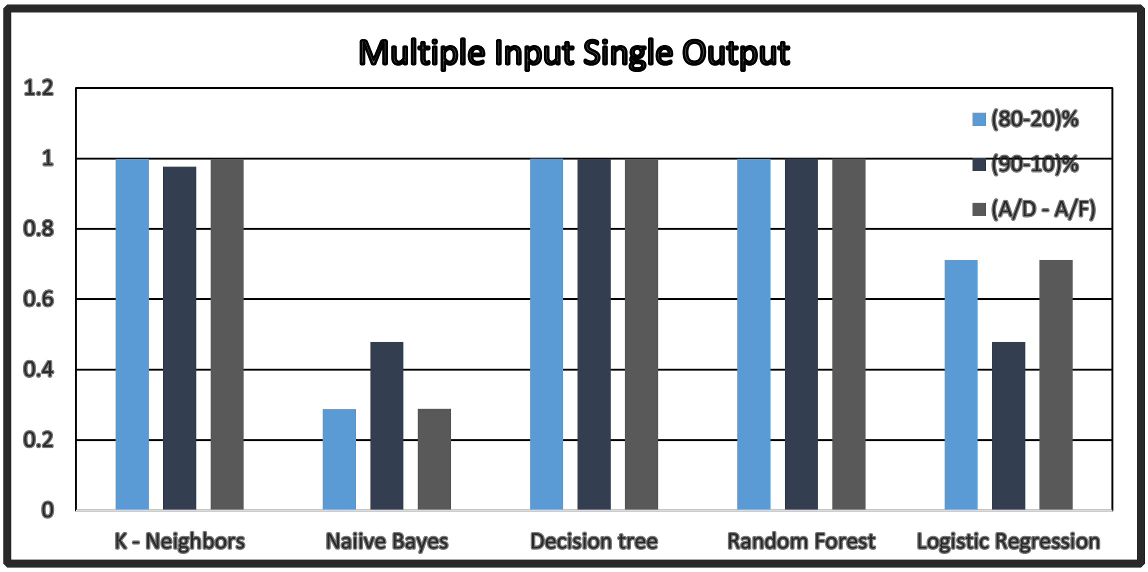


Fig 4.1.4.9 Multiple Input Single Output

2. Multiple Input Multiple Output

   Here every dependent variable is predicted simultaneously. The ML techniques applied here are:

   - K-Neighbors
   - Decision Tree
   - Random Forest

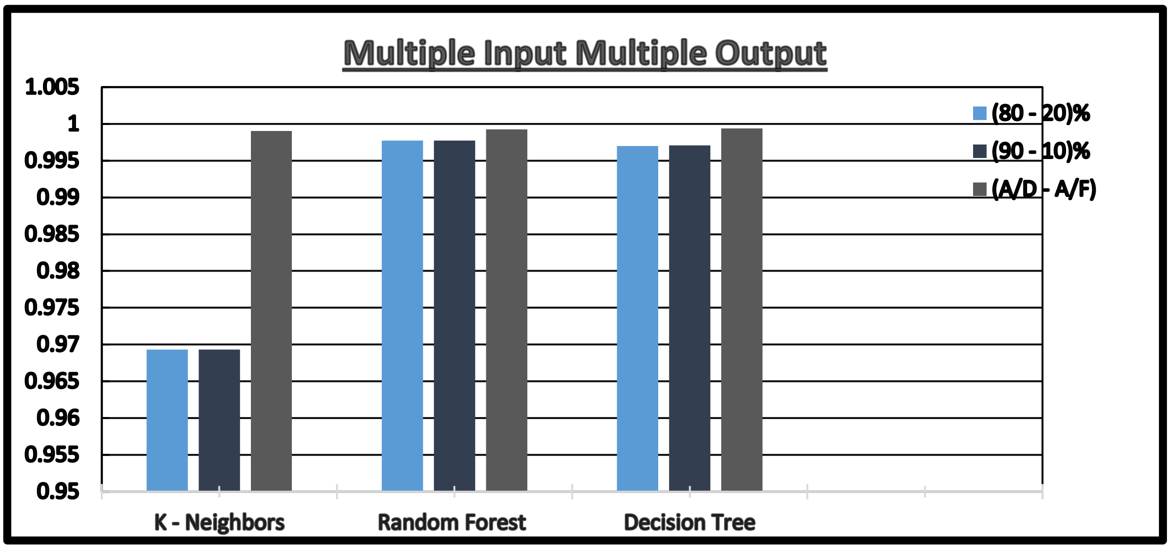


Fig 4.1.4.10 Multiple Input Multiple Output

Naïve Bayes and logistic Regression does not support multiple output.

4.2 RESULT ANALYSIS

Fig 4.2.11 given below provides us with overall analysis for the proposed model.

	K - Neighbors	Random Forest	Decision Tree	Logistic Regression	Naïve Bayes
Multiple Input Single Output	99.17%	99.92%	99.86%	63.43%	36.58%
Multiple Input Multiple Output	97.92%	99.82%	99.78%	N/A	N/A

Fig 4.2.11 Result Analysis

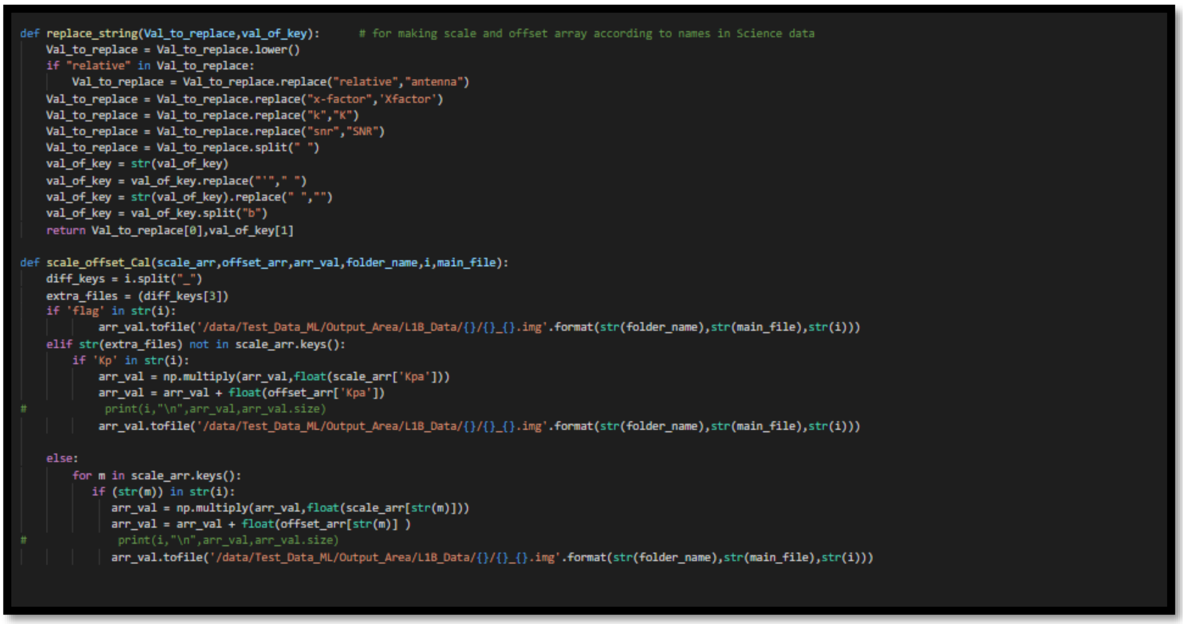
From the above figure, we can say that Random Forest is the best technique for the given dataset. Based on the accuracy score of techniques the above result is accurate. However, if we consider the time taken by each technique for training the model the best technique is K-neighbors. Consider the following table 4.2.3 to support above statement.

Table 4.2.3 Time Complexity of ML Techniques

Techniques	**Time to Train the Model**
Random Forest	Approx. 22 – 23 Hrs.
Decision Tree	Approx. 17 – 18 Hrs.
Naïve Bayes	Approx. 4 – 5 Hrs.
K-Neighbors	Approx. 30 – 35 Min
Logistic Regression	Approx. 10 – 15 Min

4.3 CODE SNIPPETS

Fig 4.3.12 and Fig 4.3.13 are the code snippets for phase 1 HDF File Extraction.

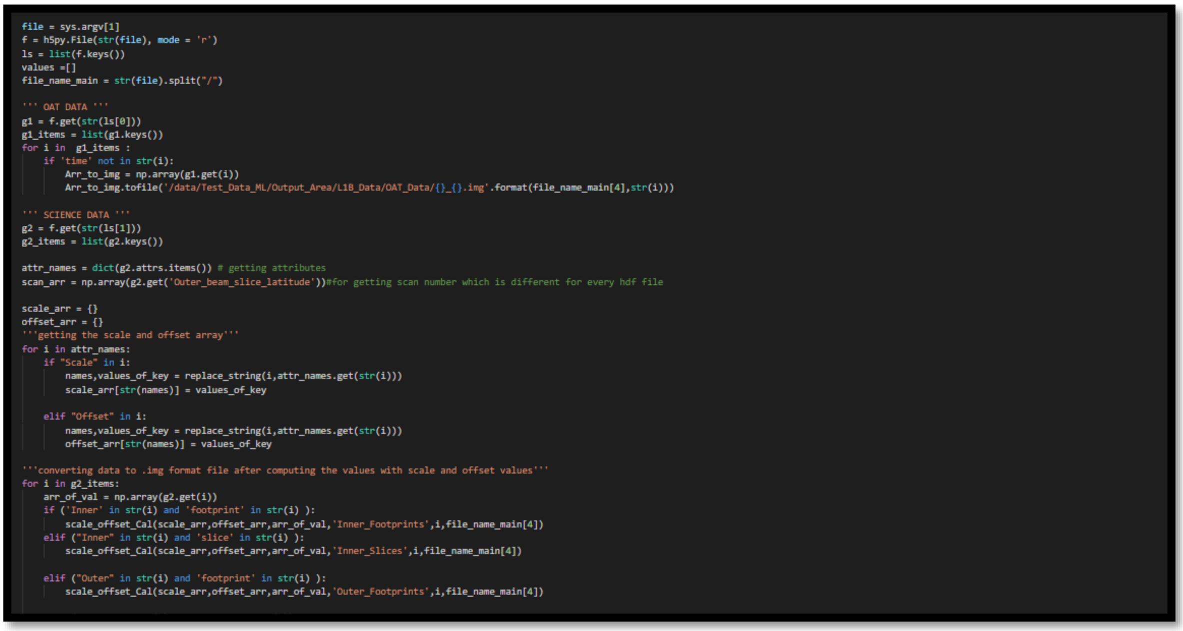


```python
def replace_string(Val_to_replace,val_of_key):      # for making scale and offset array according to names in Science data
    Val_to_replace = Val_to_replace.lower()
    if "relative" in Val_to_replace:
        Val_to_replace = Val_to_replace.replace("relative","antenna")
    Val_to_replace = Val_to_replace.replace("x-factor",'Xfactor')
    Val_to_replace = Val_to_replace.replace("k","K")
    Val_to_replace = Val_to_replace.replace("snr","SNR")
    Val_to_replace = Val_to_replace.split(" ")
    val_of_key = str(val_of_key)
    val_of_key = val_of_key.replace("'"," ")
    val_of_key = str(val_of_key).replace(" ","")
    val_of_key = val_of_key.split("b")
    return Val_to_replace[0],val_of_key[1]

def scale_offset_Cal(scale_arr,offset_arr,arr_val,folder_name,i,main_file):
    diff_keys = i.split("_")
    extra_files = (diff_keys[3])
    if 'flag' in str(i):
            arr_val.tofile('/data/Test_Data_ML/Output_Area/L1B_Data/{}/{}_{}.img'.format(str(folder_name),str(main_file),str(i)))
    elif str(extra_files) not in scale_arr.keys():
        if 'Kp' in str(i):
            arr_val = np.multiply(arr_val,float(scale_arr['Kpa']))
            arr_val = arr_val + float(offset_arr['Kpa'])
#            print(i,"\n",arr_val,arr_val.size)
            arr_val.tofile('/data/Test_Data_ML/Output_Area/L1B_Data/{}/{}_{}.img'.format(str(folder_name),str(main_file),str(i)))

    else:
        for m in scale_arr.keys():
           if (str(m)) in str(i):
              arr_val = np.multiply(arr_val,float(scale_arr[str(m)]))
              arr_val = arr_val + float(offset_arr[str(m)] )
#              print(i,"\n",arr_val,arr_val.size)
              arr_val.tofile('/data/Test_Data_ML/Output_Area/L1B_Data/{}/{}_{}.img'.format(str(folder_name),str(main_file),str(i)))
```

Fig 4.3.12 Phase 1 Code Snippet (1)



```python
file = sys.argv[1]
f = h5py.File(str(file), mode = 'r')
ls = list(f.keys())
values =[]
file_name_main = str(file).split("/")

''' OAT DATA '''
g1 = f.get(str(ls[0]))
g1_items = list(g1.keys())
for i in  g1_items :
    if 'time' not in str(i):
        Arr_to_img = np.array(g1.get(i))
        Arr_to_img.tofile('/data/Test_Data_ML/Output_Area/L1B_Data/OAT_Data/{}_{}.img'.format(file_name_main[4],str(i)))

''' SCIENCE DATA '''
g2 = f.get(str(ls[1]))
g2_items = list(g2.keys())

attr_names = dict(g2.attrs.items()) # getting attributes
scan_arr = np.array(g2.get('Outer_beam_slice_latitude'))#for getting scan number which is different for every hdf file

scale_arr = {}
offset_arr = {}
'''getting the scale and offset array'''
for i in attr_names:
    if "Scale" in i:
        names,values_of_key = replace_string(i,attr_names.get(str(i)))
        scale_arr[str(names)] = values_of_key

    elif "Offset" in i:
        names,values_of_key = replace_string(i,attr_names.get(str(i)))
        offset_arr[str(names)] = values_of_key

'''converting data to .img format file after computing the values with scale and offset values'''
for i in g2_items:
    arr_of_val = np.array(g2.get(i))
    if ('Inner' in str(i) and 'footprint' in str(i) ):
        scale_offset_Cal(scale_arr,offset_arr,arr_of_val,'Inner_Footprints',i,file_name_main[4])
    elif ("Inner" in str(i) and 'slice' in str(i) ):
        scale_offset_Cal(scale_arr,offset_arr,arr_of_val,'Inner_Slices',i,file_name_main[4])

    elif ("Outer" in str(i) and 'footprint' in str(i) ):
        scale_offset_Cal(scale_arr,offset_arr,arr_of_val,'Outer_Footprints',i,file_name_main[4])
```

Fig 4.3.13 Phase 1 Code Snippet (2)

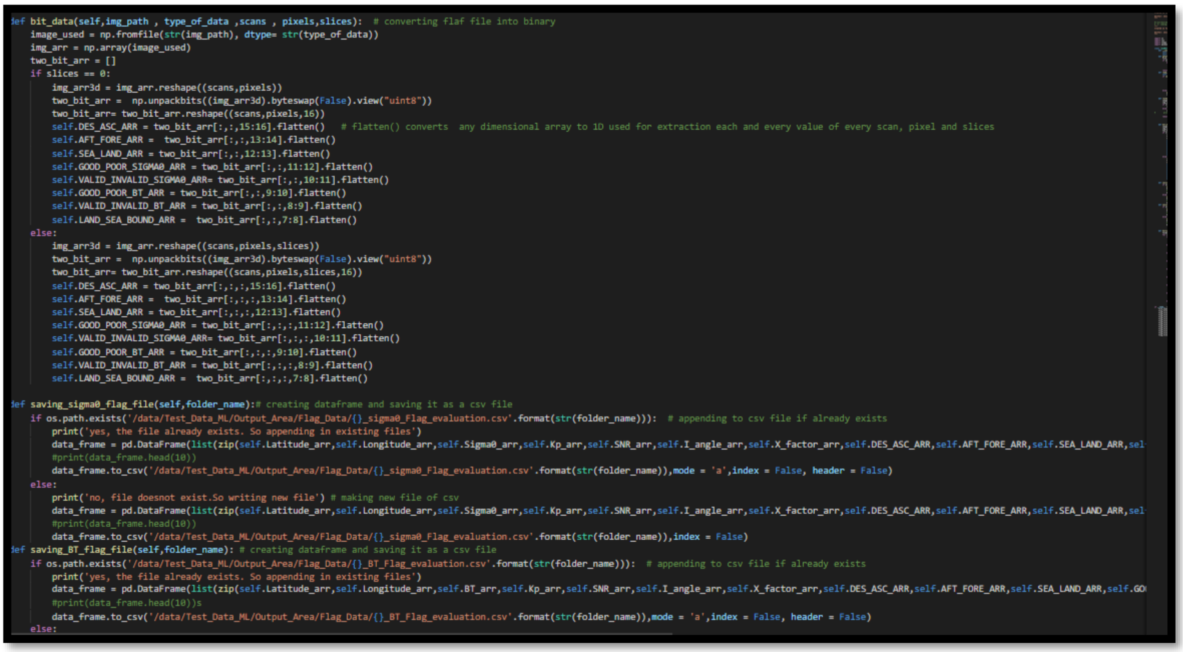
Fig 4.3.14 and Fig 4.3.15 Shows the Code Snippets for phase 2 Input Data Formation

```python
class readMetafile:
    def __init__(self,Metafile):
        self.Metafile = Metafile
        self.read_Metaf(Metafile)
    def getproperties(self, file):
        file_val = {}
        if os.path.exists(file):
            with open(file) as open_file:
                for line in open_file:
                    key,value = line.rstrip("\n").partition("=")[::2]
                    file_val[key] = value
        return file_val
    def read_Metaf(self,file_name):  # reading Scale_Offset_Meta.txt file and get values
        self.Meta_value = self.getproperties(file_name)
        if len(self.Meta_value.keys())>0:
                    self.Scans = int(self.Meta_value["Scan_number"])
                    self.Inner_FP = int(self.Meta_value["Inner_footprint"])
                    self.Inner_Slice = int(self.Meta_value["Inner_slices"])
                    self.Outer_FP = int(self.Meta_value["Outer_footprint"])
                    self.Outer_Slice = int(self.Meta_value["Outer_slices"])
                    self.dqestring = (self.Meta_value["DqeString"])
        else:
            print("work file is not present")
            sys.exit(128)

    def reshape(self,img_path,type_of_data,scans,pixels,slices): #reshaping all the arrays obtained from .img except flag files
        image_used = np.fromfile(img_path, dtype= type_of_data)
        img_arr = np.array(image_used)
        if slices == 0:
            img_arr3d = img_arr.reshape((int(scans),int(pixels)))
        else:
            img_arr3d = img_arr.reshape((int(scans),int(pixels),int(slices)))
#        print( img_arr3d,"\n" ,img_arr3d .shape , "\n", img_path)

        return img_arr3d
```

Fig 4.3.14 Phase 2 Code Snippet (1)



```python
def bit_data(self,img_path , type_of_data ,scans , pixels,slices):  # converting flaf file into binary
    image_used = np.fromfile(str(img_path), dtype= str(type_of_data))
    img_arr = np.array(image_used)
    two_bit_arr = []
    if slices == 0:
        img_arr3d = img_arr.reshape((scans,pixels))
        two_bit_arr =  np.unpackbits((img_arr3d).byteswap(False).view("uint8"))
        two_bit_arr= two_bit_arr.reshape((scans,pixels,16))
        self.DES_ASC_ARR = two_bit_arr[:,:,15:16].flatten()   # flatten() converts  any dimensional array to 1D used for extraction each and every value of every scan, pixel and slices
        self.AFT_FORE_ARR =  two_bit_arr[:,:,13:14].flatten()
        self.SEA_LAND_ARR = two_bit_arr[:,:,12:13].flatten()
        self.GOOD_POOR_SIGMA0_ARR = two_bit_arr[:,:,11:12].flatten()
        self.VALID_INVALID_SIGMA0_ARR= two_bit_arr[:,:,10:11].flatten()
        self.GOOD_POOR_BT_ARR = two_bit_arr[:,:,9:10].flatten()
        self.VALID_INVALID_BT_ARR = two_bit_arr[:,:,8:9].flatten()
        self.LAND_SEA_BOUND_ARR =  two_bit_arr[:,:,7:8].flatten()
    else:
        img_arr3d = img_arr.reshape((scans,pixels,slices))
        two_bit_arr =  np.unpackbits((img_arr3d).byteswap(False).view("uint8"))
        two_bit_arr= two_bit_arr.reshape((scans,pixels,slices,16))
        self.DES_ASC_ARR = two_bit_arr[:,:,:,15:16].flatten()
        self.AFT_FORE_ARR =  two_bit_arr[:,:,:,13:14].flatten()
        self.SEA_LAND_ARR = two_bit_arr[:,:,:,12:13].flatten()
        self.GOOD_POOR_SIGMA0_ARR = two_bit_arr[:,:,:,11:12].flatten()
        self.VALID_INVALID_SIGMA0_ARR= two_bit_arr[:,:,:,10:11].flatten()
        self.GOOD_POOR_BT_ARR = two_bit_arr[:,:,:,9:10].flatten()
        self.VALID_INVALID_BT_ARR = two_bit_arr[:,:,:,8:9].flatten()
        self.LAND_SEA_BOUND_ARR =  two_bit_arr[:,:,:,7:8].flatten()

def saving_sigma0_flag_file(self,folder_name):# creating dataframe and saving it as a csv file
    if os.path.exists('/data/Test_Data_ML/Output_Area/Flag_Data/{}_sigma0_Flag_evaluation.csv'.format(str(folder_name))):  # appending to csv file if already exists
        print('yes, the file already exists. So appending in existing files')
        data_frame = pd.DataFrame(list(zip(self.Latitude_arr,self.Longitude_arr,self.Sigma0_arr,self.Kp_arr,self.SNR_arr,self.I_angle_arr,self.X_factor_arr,self.DES_ASC_ARR,self.AFT_FORE_ARR,self.SEA_LAND_ARR,sel
        #print(data_frame.head(10))
        data_frame.to_csv('/data/Test_Data_ML/Output_Area/Flag_Data/{}_sigma0_Flag_evaluation.csv'.format(str(folder_name)),mode = 'a',index = False, header = False)
    else:
        print('no, file doesnot exist.So writing new file') # making new file of csv
        data_frame = pd.DataFrame(list(zip(self.Latitude_arr,self.Longitude_arr,self.Sigma0_arr,self.Kp_arr,self.SNR_arr,self.I_angle_arr,self.X_factor_arr,self.DES_ASC_ARR,self.AFT_FORE_ARR,self.SEA_LAND_ARR,sel
        #print(data_frame.head(10))
        data_frame.to_csv('/data/Test_Data_ML/Output_Area/Flag_Data/{}_sigma0_Flag_evaluation.csv'.format(str(folder_name)),index = False)
def saving_BT_flag_file(self,folder_name): # creating dataframe and saving it as a csv file
    if os.path.exists('/data/Test_Data_ML/Output_Area/Flag_Data/{}_BT_Flag_evaluation.csv'.format(str(folder_name))):  # appending to csv file if already exists
        print('yes, the file already exists. So appending in existing files')
        data_frame = pd.DataFrame(list(zip(self.Latitude_arr,self.Longitude_arr,self.BT_arr,self.Kp_arr,self.SNR_arr,self.I_angle_arr,self.X_factor_arr,self.DES_ASC_ARR,self.AFT_FORE_ARR,self.SEA_LAND_ARR,self.GO
        #print(data_frame.head(10))s
        data_frame.to_csv('/data/Test_Data_ML/Output_Area/Flag_Data/{}_BT_Flag_evaluation.csv'.format(str(folder_name)),mode = 'a',index = False, header = False)
    else:
```

Fig 4.3.15 Phase 2 Code Snippet (2)

Fig 4.3.16 and Fig 4.3.17 shows the last two phases of machine learning applications.

```python
def train_test_data(csv_file):
    df= pd.read_csv('/data/Test_Data_ML/Output_Area/Flag_Data/{}'.format(csv_file))
    x = df.iloc[:,[0,1,2,3,4,5,6]]
    y = df.iloc[:,7]
```

```python
def kneighbors_method(x_train,x_test,y_train):
    kN_Model = KNeighborsClassifier(n_neighbors = 1)
    kN_Model.fit(x_train,y_train)
    y_pred = kN_Model.predict(x_test)
    return y_pred

def naiive_bayes_method(x_train,x_test,y_train):
    naiivemodel = GaussianNB()
    naiivemodel.fit(x_train,y_train)
    y_pred = naiivemodel.predict(x_test)
    return y_pred

def Decision_tree_method(x_train,x_test,y_train):
    DTModel = DecisionTreeClassifier()
    DTModel.fit(x_train,y_train)
    y_pred = DTModel.predict(x_test)
    return y_pred

def SVM_Classifier(x_train,x_test,y_train):
    svm_model = SVC(kernel='linear')
    svm_model.fit(x_train,y_train)
    y_pred = svm_model.predict(x_test)
    return y_pred

def random_forest_method(x_train,x_test,y_train):
    rfc = RandomForestClassifier(n_estimators=100)
    rfc.fit(x_train, y_train)
    y_pred = rfc.predict(x_test)
    return y_pred

def Logistic_Regression_Method(x_train,x_test,y_train):
    logreg = LogisticRegression()
    logreg.fit(x_train,y_train)
    y_pred=logreg.predict(x_test)
    return y_pred

def confusion_matrix_accuracy(y_test,y_pred,methodname):
    cmat=confusion_matrix(y_test,y_pred)
    print(cmat)
    print("accuracy of {} algorithm : ".format(str(methodname)),accuracy_score(y_test,y_pred))
```

Fig 4.3.16 Multiple Input Single Output Code Snippet

```python
def confusion_matrix_accuracy(y_test,y_pred,methodname):
    cmat=multilabel_confusion_matrix(y_test,y_pred)
    print("confusion matrix\n",cmat)
    print("accuracy of {} algorithm : ".format(str(methodname)),accuracy_score(y_test,y_pred))

def kneighbors_method(x_train,x_test,y_train):
    kN_Model = KNeighborsClassifier(n_neighbors = 1)
    kN_Model.fit(x_train,y_train)
    y_pred = kN_Model.predict(x_test)
    return y_pred
def naiive_bayes_method(x_train,x_test,y_train):
    naiivemodel = GaussianNB()
    naiivemodel.fit(x_train,y_train)
    y_pred = naiivemodel.predict(x_test)
    return y_pred

def Decision_tree_method(x_train,x_test,y_train):
    DTModel = DecisionTreeClassifier()
    DTModel.fit(x_train,y_train)
    y_pred = DTModel.predict(x_test)
    return y_pred

def random_forest_method(x_train,x_test,y_train):
    rfc = RandomForestClassifier(n_estimators=100)
    rfc.fit(x_train, y_train)
    y_pred = rfc.predict(x_test)
    return y_pred


df= pd.read_csv('/data/Test_Data_ML/Output_Area/Flag_Data/Inner_Footprints_sigma0_Flag_evaluation.csv')
#print(df.head(5))
x = df.iloc[:,[0,1,2,3,4,5,6]]
y = df.iloc[:,[7,8,9,10,11]]
```

Fig 4.3.17 Multiple Input Multiple Output

Chapter 5 CONCLUSION

5.1 SUMMARY OF WORK DONE

I started my internship on 1st Dec 2021, and since then I have successfully completed all the tasks handed to me. The starting couple of weeks were about getting knowledge about the system, the work place and environment and basic concepts.

After having some basic knowledge, I started applying them by optimizing some old codes, which helped me in brushing up my skills as well as enhance them. I also learned about the database connection over Oracle as well as PostgreSQL using general code of Java using which I was able to make the reports using python and Latex.

Now, as I was acquainted with the methods, techniques, and data, I started working on my project, which was allocated to me by my supervisor. I also worked on Research Paper for the project.

Therefore, to summarize my work of 19 weeks, I can say that I worked enthusiastically and sincerely to complete every task handed to me and learned a lot throughout that process.

5.2 LIMITATIONS AND FUTURE ENHANCEMENTS

The proposed model has 2 main limitations:

1. Firstly, as the size of dataset increases the time to train the model also increases.
2. The dataset needs to be properly formed. It should not have null values or wrong values which may hamper the results.

More parameters can be considered in future for better results and better algorithmic techniques can be developed to decrease time complexity.

5.3 EXPERIENCE

I would like to end this report by sharing my wonderful experience of these 5 months. Firstly, I would like to thank SAC-ISRO, Dr. B Kartikeyan and Scientist. Anuja Sharma for providing me with such a great opportunity of working with the most reputed and honorable organization of India.

I had a great time working with those intellectual people and I learned a lot from them. My colleagues and supervisor were very helpful and humble and they guided me throughout the journey.

I was able to acquire a great deal of knowledge and skills and I am thankful to my guide here who pushed me to give my best. She guided as well as inspired me to work efficiently. She also made me work harder to get best-optimized results, I am sure knowledge, skills, and values that I learnt from her are going to be very useful in my future.

My learnings from here has definitely made me ready for the future challenges.

5.4 DATES OF COMMON EVALUATION

My college guide Prof. Ankita Tiwari conducted continuous to make sure of our work being up to the mark and to be encourage, assist and guide us through this process.

CE 1 = 12^{th} April 2022

CE 2 = 4^{th} May 2022

REFERENCES

- Batta Mahesh (2020) “Machine Learning Algorithms – A Review” – IJSR Vol.9 Issue: 01
- Charles Elachi, Jakob Vanzy (2006) “Introduction to Physics and Techniques of Remote Sensing” – John_Willey_sons,Inc, Inc. Second Edition
  Devang Mankand, Rajesh Sikhakolli, Puja Kakkar, Qamer Saquib, Krishna Murari Jain, V.M. Ramanujan and Pradeep Thapliyal (2019) “ScatSat-1 Scatterometer Data Processing” – Current Science Vol.117 Issue: 06
- George Joseph (2003) “Fundamentals of Remote Sensing” - ISRO
  Maneesha Gupta, Dr. B Kartikeyan (2011) “Initial Phase Study of Oceansat2 Scatterometer” – TN-57 IAQD/SIPG/SAC (ISRO)
- Maneesha Gupta*, Anuja Sharma, B. Kartikeyan (2019) “Evaluation and Quality Monitoring of ScatSat-1 Scan Mode Data ” – ISG Vol. 13 Issue:02
- Sartar Vij Singh, Reet Kamal Tiwari, Hemendra Singh Gussain, Vishakha Sood (2020) “Potential Applications of ScatSat-1 Satellite Sensor: A Systematic Review” – IEEE Vol.20 Issue: 21
- ScatSat-1 SAC Team (2014) “ScatSat-1 Scatterometer Data Product System SAC Elements - Preparatory Design Review Document” - IAQD/SIPG/SAC (ISRO)
- ScatSat-1 SAC Team (2016) “ScatSat-1 Scatterometer Data Product Detailed Design Review Document” - MPDP/SIPG/SAC (ISRO)
- https://scikit-learn.org
- Suchandra A. Bhowmick, James Cotton, Alexander Fore, Raj Kumar, Christophe Payan, Ernesto Rodríguez, Anuja Sharma, Bryan Stiles, Ad Stoffelen and Anton Verhoef (2019) “An Assessment of the Performance of ISROs ScatSat-1 Scatterometer” – Current Science Vol.117 Issue: 06