General Purpose of the Notebook

This notebook is designed to process and analyze data related to radio bursts. It includes steps for reading data, labeling images, and exploring the dataset for insights. The documentation will provide a detailed explanation of each cell to ensure the code is understandable for future reference and collaborative work.

Cell-by-Cell Explanation

Cell 1
Purpose: Importing libraries.

```
from astropy.io import fits
import os
import pandas as pd
import numpy as np
```

This cell imports essential libraries:

- astropy.io.fits for handling FITS files, which are commonly used in astronomy.
- os for file and directory operations.
- pandas for data manipulation and analysis.
- numpy for numerical operations.

These libraries enable the reading of data files and performing numerical operations.

Cell 2
Purpose: Listing files in the directory.

```
path = 'files'
os.listdir(path)
len(os.listdir(path))
```

This cell defines the path to the directory containing the data files (files) and lists all files within it. The len() function is used to count the total number of files.

Cell 3
Purpose: Extracting specific time and file information.

```
f ='BIR_20120101_185959_03.fit.gz'
f.split('_')[2][:2]+':'+f.split('_')[2][2:4]+':'+f.split('_')[2][4:6]
f.index('.')
f[:f.index('.')]
```

This cell demonstrates string manipulation to extract specific information from a file name:

- Extracting the time from the filename (18:59:59).
- Finding the position of the first period (.) in the string.
- Extracting the substring before the first period.

Cell 4
Purpose: Checking conditions for time and station entries.

```
ts = ['18:59:59','20:00:00']
ss = ['bir','bir']

t = '18:59:59'
s = 'bir'
if t in ts and s not in ss[ts.index(t)]:
   print('t')
else: print('f')

ss[ts.index(t)] +=  ', '+str(s)
ss[ts.index(t)]
ss
```

This cell initializes lists (ts and ss) for times and station names. It checks if a time exists in the list and whether a station is missing at that time. If the station is not listed, it appends it.

Cell 5
Purpose: Extracting metadata from files.

```
stations =[]
dates = []
time =[]
name=[]

for file in os.listdir(path):
   stations.append(file.split('_')[0])
   dates.append(file.split('_')[1])
   time.append(file.split('_')[2])
   name.append(file)

for f in range(len(time)):
   time[f]=time[f][:2]+':'+time[f][2:4]+':'+time[f][4:6]
```

This cell iterates through the files in the directory, extracting metadata:

- stations: The station name.
- dates: The observation date.
- time: The observation time, formatted as HH:MM:SS.

- name: The full file name.

Cell 6
Purpose: Creating and populating a DataFrame.

```
df = pd.DataFrame(columns=['Type'])
df ['date'] = np.array(dates)
df ['stations'] = np.array(stations)
df ['start time'] = np.array(time)
df['name'] = np.array(name)

df['date'] = pd.to_datetime(df['date'], format='%Y%m%d')
df['stations'] = df['stations'].apply(lambda x: x.split(', '))
df['start time'] = pd.to_datetime(df['start time'], format='%H:%M:%S').dt.time
df['name'] = df['name'].astype('str')
df.head()
```

This cell creates a DataFrame to organize metadata and converts columns to appropriate formats:

- date: Converted to datetime.
- stations: Split into lists.
- start time: Converted to datetime.time.

Cell 7
Purpose: Handling time-based data deduplication.

```
df = df.sort_values(by=['date', 'start time']).reset_index(drop=True)

for i in range(1, len(df)):
    ...
    # Deduplication logic
```

This cell sorts the DataFrame by date and time, then deduplicates rows based on specific conditions (e.g., time within a certain threshold).

Cell 8
Purpose: Reading FITS files and processing images.

```
from tqdm import tqdm
import read_Data
import cv2

images = []
times = []
freqs = []
names=[]
```

```
first_image_shape = None

for file in tqdm(list(df['name'])):
    ...
```

This cell reads FITS files using `read_Data.read_fits()` and processes images:

- Resizing images to match the shape of the first image.
- Appending metadata (times, freqs) to lists.

Cell 9
Purpose: Checkpoints and classification.

```
Type = [...]
checkpoint = pd.DataFrame()
checkpoint['Type'] = np.array(Type)
checkpoint['Name'] = info['name'][:checkpoint.shape[0]]
checkpoint.to_csv('checkpoint1.csv')
```

This cell creates labeled checkpoints for images, saving them to a CSV file for future reference.
