Assignment 1- SEVIR Dataset

Summary	In this codelab, you will download and analyze the SEVIR dataset from aws using boto 3 module. You will analyze the Catalog: A CSV file with rows describing the metadata of an event and Data Files: A set of HDF5 files containing events for a certain sensor type. The set of storm events files from NWS registry is visualized using GCP bucket, big query and Google data studio. The reports have been generated for various attributes contributing to the storm events.
URL	https://codelabs-preview.appspot.com/?file_id=1a15lLYAxzjThC4sFioo2zqqEymSbLbNCMd3vF4aSHlM#0
Category	Web
Environment	Jupyter notebook, Google data studio, Big query, GCP bucket
Status	Published
Feedback Link	
Author	Mayuri More, Nishanth Reddy Kogilathota, Romita Thally
Author LDAP	Mayuri
Analytics Account	

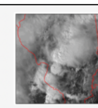
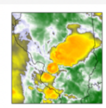
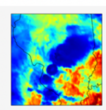
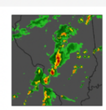
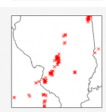
Introduction to SEVIR Dataset

The Storm EVent ImagRy (SEVIR) dataset is a collection of temporally and spatially aligned images containing weather events captured by satellite and radar. This dataset was created using publically available datasets distributed by NOAA, including the GOES-16 geostationary satellite, and data derived from NEXRAD weather radars, both of which are available on the Registry of Open Data on AWS. This tutorial provides information about the SEVIR dataset as well as code samples that can be used to access the data.

There is no shortage of weather data available to the public: Weather radar, GEO & LEO satellite, surface observations, numerical weather prediction models, balloons, ocean buoys, aircraft measurements, lightning sensing, radio occultation, and others are being measured every second of every day to aid in our ability to analyze and forecast the weather. As a result, both the size and complexity of all of this data can make it difficult to work with. Many datasets, such as radar and satellite, are too large to process in bulk for those without access to massive compute and storage. Some datasets also require a level of domain expertise to decode, understand and process. These are some of the barriers for scientists and researchers who want to understand and use this data for their applications.

To help alleviate these issues, the SEVIR dataset was constructed to enable faster R&D in weather sensing, avoidance, short-term forecasting and other related applications. SEVIR combines and aligns multiple weather sensing modalities into a single, accessible dataset accessible for free on the cloud that can be used by meteorologists, data scientists and other researchers.

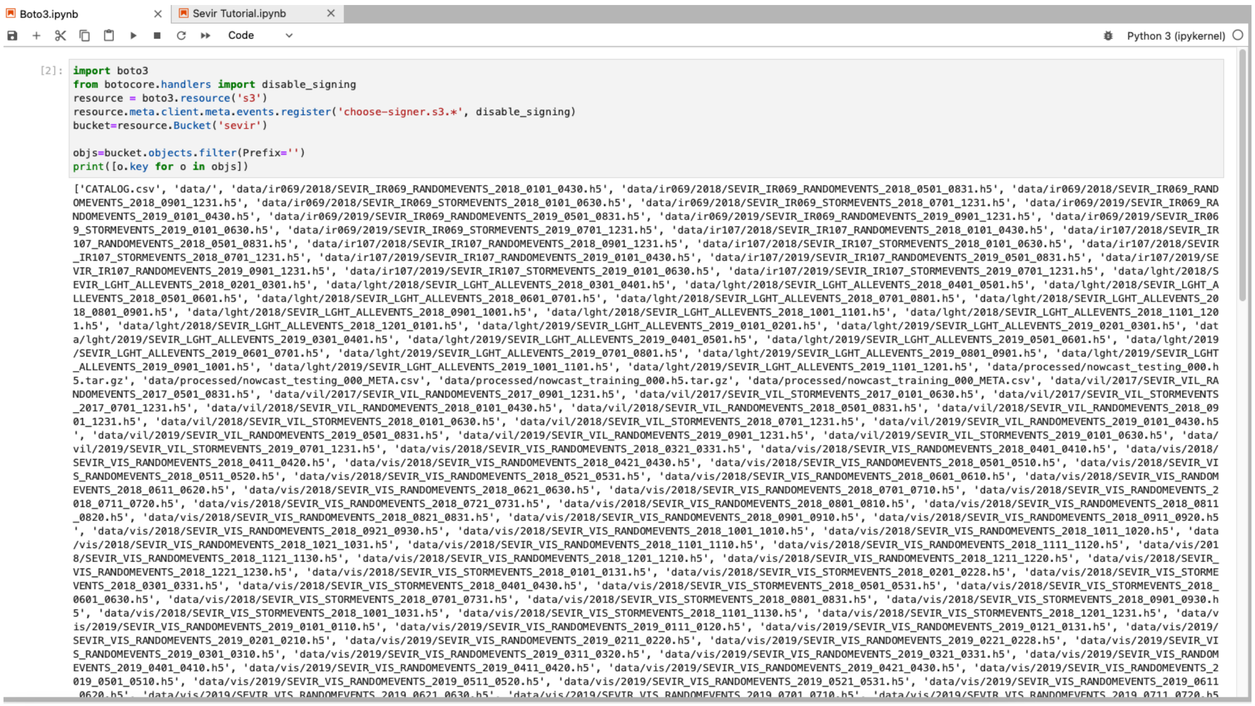
SEVIR is a collection of thousands of "events", which are 4-hour sequences of weather captured by up to 5 sensors. These 5 sensing modalities are summarized in the table below:

Sensor	Data key	Description	Spatial Resolution	Patch Size	Time step	Sample Colorized Image
GOES-16 C02 0.64 μm	`vis`	Visible satellite imagery	0.5 km	768x768	5 minute	
GOES-16 C09 6.9 μm	`ir069`	Infrared Satellite imagery (Water Vapor)	2 km	192 x 192	5 minutes	
GOES-16 C13 10.7 μm	`ir107`	Infrared Satellite imagery (Window)	2 km	192 x 192	5 minutes	
Vertically Integrated Liquid (VIL)	`vil`	NEXRAD radar mosaic of VIL	1 km	384 x 384	5 minutes	
GOES-16 GLM flashes	`lght`	Detections of inter cloud and cloud to ground lightning events	8 km	N/A	Continuous	

Downloading SEVIR

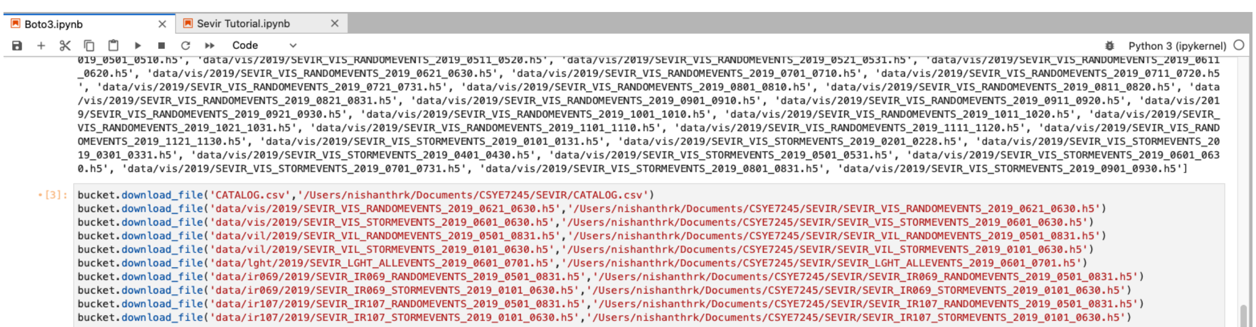
Using `boto3` modules :

Downloading CSV and HDF5 files from aws s3 using boto 3 modules




```python
import boto3
from botocore.handlers import disable_signing
resource = boto3.resource('s3')
resource.meta.client.meta.events.register('choose-signer.s3.*', disable_signing)
bucket=resource.Bucket('sevir')

objs=bucket.objects.filter(Prefix='')
print([o.key for o in objs])
```


```
['CATALOG.csv', 'data/', 'data/ir069/2018/SEVIR_IR069_RANDOMEVENTS_2018_0101_0430.h5', 'data/ir069/2018/SEVIR_IR069_RANDOMEVENTS_2018_0501_0831.h5', 'data/ir069/2018/SEVIR_IR069_RAND
OMEVENTS_2018_0901_1231.h5', 'data/ir069/2018/SEVIR_IR069_STORMEVENTS_2018_0101_0630.h5', 'data/ir069/2018/SEVIR_IR069_STORMEVENTS_2018_0701_1231.h5', 'data/ir069/2019/SEVIR_IR069_RA
NDOMEVENTS_2019_0101_0430.h5', 'data/ir069/2019/SEVIR_IR069_RANDOMEVENTS_2019_0501_0831.h5', 'data/ir069/2019/SEVIR_IR069_RANDOMEVENTS_2019_0901_1231.h5', 'data/ir069/2019/SEVIR_IR06
9_STORMEVENTS_2019_0101_0630.h5', 'data/ir069/2019/SEVIR_IR069_STORMEVENTS_2019_0701_1231.h5', 'data/ir107/2018/SEVIR_IR107_RANDOMEVENTS_2018_0101_0430.h5', 'data/ir107/2018/SEVIR_IR
107_RANDOMEVENTS_2018_0501_0831.h5', 'data/ir107/2018/SEVIR_IR107_RANDOMEVENTS_2018_0901_1231.h5', 'data/ir107/2018/SEVIR_IR107_STORMEVENTS_2018_0101_0630.h5', 'data/ir107/2018/SEVIR
_IR107_STORMEVENTS_2018_0701_1231.h5', 'data/ir107/2019/SEVIR_IR107_RANDOMEVENTS_2019_0101_0430.h5', 'data/ir107/2019/SEVIR_IR107_RANDOMEVENTS_2019_0501_0831.h5', 'data/ir107/2019/SE
VIR_IR107_RANDOMEVENTS_2019_0901_1231.h5', 'data/ir107/2019/SEVIR_IR107_STORMEVENTS_2019_0101_0630.h5', 'data/ir107/2019/SEVIR_IR107_STORMEVENTS_2019_0701_1231.h5', 'data/lght/2018/S
EVIR_LGHT_ALLEVENTS_2018_0201_0301.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_0301_0401.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_0401_0501.h5', 'data/lght/2018/SEVIR_LGHT_A
LLEVENTS_2018_0501_0601.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_0601_0701.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_0701_0801.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_20
18_0801_0901.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_0901_1001.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_1001_1101.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_1101_120
1.h5', 'data/lght/2018/SEVIR_LGHT_ALLEVENTS_2018_1201_0101.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0101_0201.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0201_0301.h5', 'dat
a/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0301_0401.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0401_0501.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0501_0601.h5', 'data/lght/2019
/SEVIR_LGHT_ALLEVENTS_2019_0601_0701.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0701_0801.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0801_0901.h5', 'data/lght/2019/SEVIR_LGHT
_ALLEVENTS_2019_0901_1001.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_1001_1101.h5', 'data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_1101_1201.h5', 'data/processed/nowcast_testing_000.h
5.tar.gz', 'data/processed/nowcast_testing_000_META.csv', 'data/processed/nowcast_training_000.h5.tar.gz', 'data/processed/nowcast_training_000_META.csv', 'data/vil/2017/SEVIR_VIL_RA
NDOMEVENTS_2017_0501_0831.h5', 'data/vil/2017/SEVIR_VIL_RANDOMEVENTS_2017_0901_1231.h5', 'data/vil/2017/SEVIR_VIL_STORMEVENTS_2017_0101_0630.h5', 'data/vil/2017/SEVIR_VIL_STORMEVENTS
_2017_0701_1231.h5', 'data/vil/2018/SEVIR_VIL_RANDOMEVENTS_2018_0101_0430.h5', 'data/vil/2018/SEVIR_VIL_RANDOMEVENTS_2018_0501_0831.h5', 'data/vil/2018/SEVIR_VIL_RANDOMEVENTS_2018_09
01_1231.h5', 'data/vil/2018/SEVIR_VIL_STORMEVENTS_2018_0101_0630.h5', 'data/vil/2018/SEVIR_VIL_STORMEVENTS_2018_0701_1231.h5', 'data/vil/2019/SEVIR_VIL_RANDOMEVENTS_2019_0101_0430.h5
', 'data/vil/2019/SEVIR_VIL_RANDOMEVENTS_2019_0501_0831.h5', 'data/vil/2019/SEVIR_VIL_RANDOMEVENTS_2019_0901_1231.h5', 'data/vil/2019/SEVIR_VIL_STORMEVENTS_2019_0101_0630.h5', 'data/
vil/2019/SEVIR_VIL_STORMEVENTS_2019_0701_1231.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0321_0331.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0401_0410.h5', 'data/vis/2018/
SEVIR_VIS_RANDOMEVENTS_2018_0411_0420.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0421_0430.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0501_0510.h5', 'data/vis/2018/SEVIR_VI
S_RANDOMEVENTS_2018_0511_0520.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0521_0531.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0601_0610.h5', 'data/vis/2018/SEVIR_VIS_RANDOM
EVENTS_2018_0611_0620.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0621_0630.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0701_0710.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2
018_0711_0720.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0721_0731.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0801_0810.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0811
_0820.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0821_0831.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0901_0910.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0911_0920.h5
', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_0921_0930.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_1001_1010.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_1011_1020.h5', 'data
/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_1021_1031.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_1101_1110.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_1111_1120.h5', 'data/vis/201
8/SEVIR_VIS_RANDOMEVENTS_2018_1121_1130.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_1201_1210.h5', 'data/vis/2018/SEVIR_VIS_RANDOMEVENTS_2018_1211_1220.h5', 'data/vis/2018/SEVIR_
VIS_RANDOMEVENTS_2018_1221_1230.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_0101_0131.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_0201_0228.h5', 'data/vis/2018/SEVIR_VIS_STORME
VENTS_2018_0301_0331.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_0401_0430.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_0501_0531.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_
0601_0630.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_0701_0731.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_0801_0831.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_0901_0930.h
5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_1001_1031.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_1101_1130.h5', 'data/vis/2018/SEVIR_VIS_STORMEVENTS_2018_1201_1231.h5', 'data/v
is/2019/SEVIR_VIS_RANDOMEVENTS_2019_0101_0110.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0111_0120.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0121_0131.h5', 'data/vis/2019/
SEVIR_VIS_RANDOMEVENTS_2019_0201_0210.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0211_0220.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0221_0228.h5', 'data/vis/2019/SEVIR_VI
S_RANDOMEVENTS_2019_0301_0310.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0311_0320.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0321_0331.h5', 'data/vis/2019/SEVIR_VIS_RANDOM
EVENTS_2019_0401_0410.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0411_0420.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0421_0430.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2
019_0501_0510.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0511_0520.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0521_0531.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0611
_0620.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0621_0630.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0701_0710.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0711_0720.h5
```

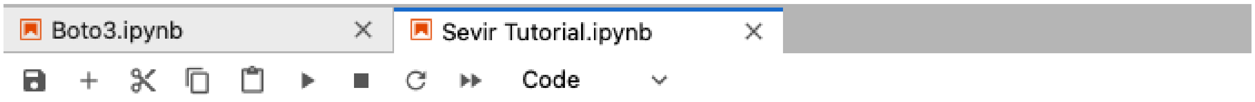

```
019_0501_0510.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0511_0520.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0521_0531.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0611
_0620.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0621_0630.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0701_0710.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0711_0720.h5
', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0721_0731.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0801_0810.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0811_0820.h5', 'data
/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0821_0831.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0901_0910.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0911_0920.h5', 'data/vis/201
9/SEVIR_VIS_RANDOMEVENTS_2019_0921_0930.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_1001_1010.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_1011_1020.h5', 'data/vis/2019/SEVIR_
VIS_RANDOMEVENTS_2019_1021_1031.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_1101_1110.h5', 'data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_1111_1120.h5', 'data/vis/2019/SEVIR_VIS_RAND
OMEVENTS_2019_1121_1130.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0101_0131.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0201_0228.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_20
19_0301_0331.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0401_0430.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0501_0531.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0601_063
0.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0701_0731.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0801_0831.h5', 'data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0901_0930.h5']
```


[3]:

```python
bucket.download_file('CATALOG.csv','/Users/nishanthrk/Documents/CSYE7245/SEVIR/CATALOG.csv')
bucket.download_file('data/vis/2019/SEVIR_VIS_RANDOMEVENTS_2019_0621_0630.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_VIS_RANDOMEVENTS_2019_0621_0630.h5')
bucket.download_file('data/vis/2019/SEVIR_VIS_STORMEVENTS_2019_0601_0630.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_VIS_STORMEVENTS_2019_0601_0630.h5')
bucket.download_file('data/vil/2019/SEVIR_VIL_RANDOMEVENTS_2019_0501_0831.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_VIL_RANDOMEVENTS_2019_0501_0831.h5')
bucket.download_file('data/vil/2019/SEVIR_VIL_STORMEVENTS_2019_0101_0630.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_VIL_STORMEVENTS_2019_0101_0630.h5')
bucket.download_file('data/lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0601_0701.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_LGHT_ALLEVENTS_2019_0601_0701.h5')
bucket.download_file('data/ir069/2019/SEVIR_IR069_RANDOMEVENTS_2019_0501_0831.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_IR069_RANDOMEVENTS_2019_0501_0831.h5')
bucket.download_file('data/ir069/2019/SEVIR_IR069_STORMEVENTS_2019_0101_0630.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_IR069_STORMEVENTS_2019_0101_0630.h5')
bucket.download_file('data/ir107/2019/SEVIR_IR107_RANDOMEVENTS_2019_0501_0831.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_IR107_RANDOMEVENTS_2019_0501_0831.h5')
bucket.download_file('data/ir107/2019/SEVIR_IR107_STORMEVENTS_2019_0101_0630.h5','/Users/nishanthrk/Documents/CSYE7245/SEVIR/SEVIR_IR107_STORMEVENTS_2019_0101_0630.h5')
```

SEVIR Tutorial

Accessing the SEVIR data from local storage :



[1]:

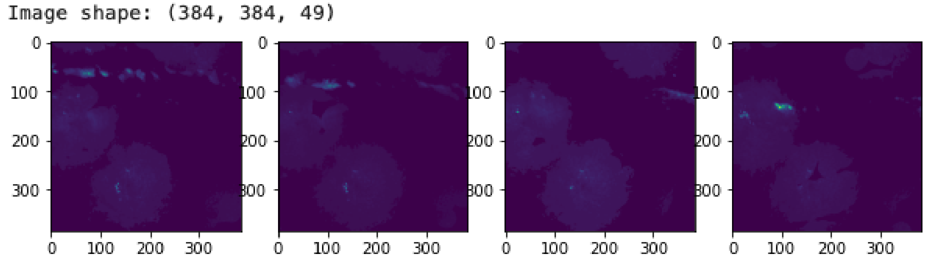
```
DATA_PATH    = '/Users/nishanthrk/Documents/CSYE7245/SEVIR'
CATALOG_PATH = '/Users/nishanthrk/Documents/CSYE7245/SEVIR/CATALOG.csv'

import os
os.environ["HDF5_USE_FILE_LOCKING"]='FALSE'
```

Accessing a specific event with event ID by opening data file directly without using the catalog

[14]:

```
import os
import h5py
import matplotlib.pyplot as plt
id = 835047

file_index = 0
with h5py.File('%s/SEVIR_VIL_STORMEVENTS_2019_0101_0630.h5' % DATA_PATH,'r') as hf:
    event_id = hf['id'][file_index]
    vil      = hf['vil'][file_index]

print('Event ID:',event_id)
print('Image shape:',vil.shape)

fig,axs=plt.subplots(1,4,figsize=(10,5))
axs[0].imshow(vil[:,:,10])
axs[1].imshow(vil[:,:,20])
axs[2].imshow(vil[:,:,30])
axs[3].imshow(vil[:,:,40])
plt.show()
```

```
Event ID: b'S834603'
Image shape: (384, 384, 49)
```



Accessing all available sensor data for a particular event

Boto3.ipynb | Sevir Tutorial.ipynb

Code | Python 3 (ipykernel)

```python
catalog = pd.read_csv(CATALOG_PATH,parse_dates=['time_utc'],low_memory=False)

# Desired image types
img_types = set(['vis','ir069','ir107','vil'])

# Group by event id, and filter to only events that have all desired img_types
events = catalog.groupby('id').filter(lambda x: img_types.issubset(set(x['img_type']))).groupby('id')
event_ids = list(events.groups.keys())
print('Found %d events matching' % len(event_ids),img_types)

# Grab a sample event and view catalog entries
sample_event = events.get_group(event_ids[-500])
print('Sample Event:',event_ids[-500])
sample_event
```

```
Found 12739 events matching {'ir107', 'ir069', 'vil', 'vis'}
Sample Event: S835047
```

[29]:

	id	file_name	file_index	img_type	time_utc	minute_offsets	episode_id	event_id	event_type	llcrnrlat	...	urcrnrlat	urcrnrlon
19299	S835047	vil/2019/SEVIR_VIL_STORMEVENTS_2019_0101_0630.h5	14	vil	2019-06-26 21:32:00	-122:-117:-112:-107:-102:-97:-92:-87:-82:-77:-...	138836.0	835047.0	Hail	35.015742	...	38.311287	-91.574426
22412	S835047	ir107/2019/SEVIR_IR107_STORMEVENTS_2019_0101_0...	12	ir107	2019-06-26 21:32:00	-124:-119:-114:-109:-104:-99:-94:-89:-84:-79:-...	138836.0	835047.0	Hail	35.015742	...	38.311287	-91.574426
29236	S835047	ir069/2019/SEVIR_IR069_STORMEVENTS_2019_0101_0...	12	ir069	2019-06-26 21:32:00	-124:-119:-114:-109:-104:-99:-94:-89:-84:-79:-...	138836.0	835047.0	Hail	35.015742	...	38.311287	-91.574426
41182	S835047	vis/2019/SEVIR_VIS_STORMEVENTS_2019_0601_0630.h5	34	vis	2019-06-26 21:32:00	-124:-119:-114:-109:-104:-99:-94:-89:-84:-79:-...	138836.0	835047.0	Hail	35.015742	...	38.311287	-91.574426
72402	S835047	lght/2019/SEVIR_LGHT_ALLEVENTS_2019_0601_0701.h5	0	lght	2019-06-26 21:32:00	NaN	138836.0	835047.0	Hail	35.015742	...	38.311287	-91.574426

5 rows × 21 columns

Using rows of sample_event to extract image data for each type

Boto3.ipynb | Sevir Tutorial.ipynb

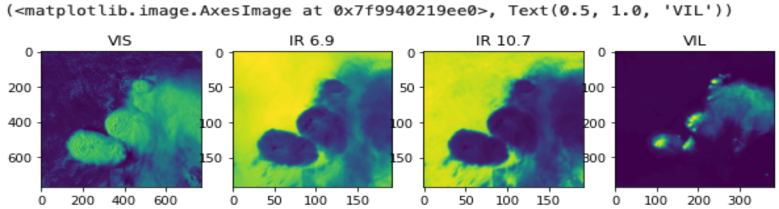
Code

[77]:

```python
def read_data(sample_event, img_type, data_path=DATA_PATH):

    fn = sample_event[sample_event.img_type==img_type].squeeze().file_name
    fi = sample_event[sample_event.img_type==img_type].squeeze().file_index
    with h5py.File(data_path + '/' + fn,'r') as hf:
        data=hf[img_type][fi]
    return data

vis = read_data(sample_event, 'vis')
ir069 = read_data(sample_event, 'ir069')
ir107 = read_data(sample_event, 'ir107')
vil = read_data(sample_event, 'vil')

# plot a frame from each img_type
fig,axs = plt.subplots(1,4,figsize=(10,5))
frame_idx = 30
axs[0].imshow(vis[:,:,frame_idx]), axs[0].set_title('VIS')
axs[1].imshow(ir069[:,:,frame_idx]), axs[1].set_title('IR 6.9')
axs[2].imshow(ir107[:,:,frame_idx]), axs[2].set_title('IR 10.7')
axs[3].imshow(vil[:,:,frame_idx]), axs[3].set_title('VIL')
```

[77]:

```
(<matplotlib.image.AxesImage at 0x7f9940219ee0>, Text(0.5, 1.0, 'VIL'))
```



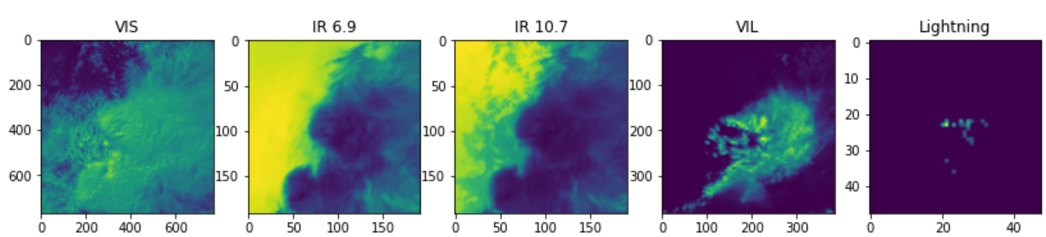
Including Lightning by converting the flash data into flash counts per pixel per 5 minute frame

Boto3.ipynb | Sevir Tutorial.ipynb

Code

```
[40]: import numpy as np
      def lght_to_grid(data):

          FRAME_TIMES = np.arange(-120.0,125.0,5) * 60 # in seconds
          out_size = (48,48,len(FRAME_TIMES))
          if data.shape[0]==0:
              return np.zeros(out_size,dtype=np.float32)

          x,y=data[:,3],data[:,4]
          m=np.logical_and.reduce( [x>=0,x<out_size[0],y>=0,y<out_size[1]] )
          data=data[m,:]
          if data.shape[0]==0:
              return np.zeros(out_size,dtype=np.float32)

          t=data[:,0]
          z=np.digitize(t,FRAME_TIMES)-1
          z[z==-1]=0 # special case:  frame 0 uses lght from frame 1

          x=data[:,3].astype(np.int64)
          y=data[:,4].astype(np.int64)

          k=np.ravel_multi_index(np.array([y,x,z]),out_size)
          n = np.bincount(k,minlength=np.prod(out_size))
          return np.reshape(n,out_size).astype(np.float32)

      def read_lght_data( sample_event, data_path=DATA_PATH ):

          fn = sample_event[sample_event.img_type=='lght'].squeeze().file_name
          id = sample_event[sample_event.img_type=='lght'].squeeze().id
          with h5py.File(data_path + '/' + fn,'r') as hf:
              data      = hf[id][:]
          return lght_to_grid(data)

      lght = read_lght_data(sample_event)
      fig,axs = plt.subplots(1,5,figsize=(14,5))
      frame_idx = 30
      axs[0].imshow(vis[:,:,frame_idx]), axs[0].set_title('VIS')
      axs[1].imshow(ir069[:,:,frame_idx]), axs[1].set_title('IR 6.9')
      axs[2].imshow(ir107[:,:,frame_idx]), axs[2].set_title('IR 10.7')
      axs[3].imshow(vil[:,:,frame_idx]), axs[3].set_title('VIL')
      axs[4].imshow(lght[:,:,frame_idx]), axs[4].set_title('Lightning')

[40]: (<matplotlib.image.AxesImage at 0x7f99791e8b80>, Text(0.5, 1.0, 'Lightning'))
```

```
Out[7]: (<matplotlib.image.AxesImage at 0x7f69e00c8470>, Text(0.5, 1.0, 'Lightning'))
```



Adding Color using colormaps

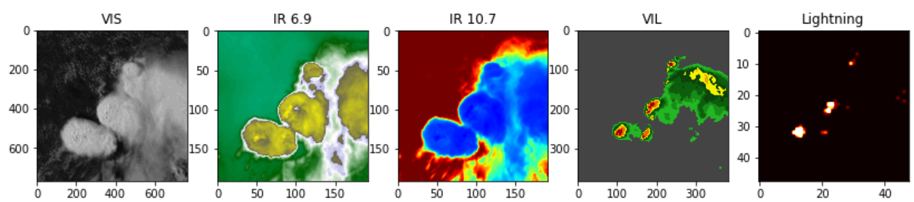
Boto3.ipynb | Sevir Tutorial.ipynb

Code

[44]:
```python
import sys
sys.path.append('/Users/nishanthrk/Downloads/eie-sevir-master') # add sevir module to path
from sevir.display import get_cmap
# Get colormaps for encoded types
vis_cmap,vis_norm,vis_vmin,vis_vmax = get_cmap('vis',encoded=True)
ir069_cmap,ir069_norm,ir069_vmin,ir069_vmax = get_cmap('ir069',encoded=True)
ir107_cmap,ir107_norm,ir107_vmin,ir107_vmax = get_cmap('ir107',encoded=True)
vil_cmap,vil_norm,vil_vmin,vil_vmax = get_cmap('vil',encoded=True)
lght_cmap,lght_norm,lght_vmin,lght_vmax = get_cmap('lght',encoded=True)

fig,axs = plt.subplots(1,5,figsize=(14,5))
frame_idx = 30
axs[0].imshow(vis[:,:,frame_idx],cmap=vis_cmap,norm=vis_norm,vmin=vis_vmin,vmax=vis_vmax), axs[0].set_title('VIS')
axs[1].imshow(ir069[:,:,frame_idx],cmap=ir069_cmap,norm=ir069_norm,vmin=ir069_vmin,vmax=ir069_vmax), axs[1].set_title('IR 6.9')
axs[2].imshow(ir107[:,:,frame_idx],cmap=ir107_cmap,norm=ir107_norm,vmin=ir107_vmin,vmax=ir107_vmax), axs[2].set_title('IR 10.7')
axs[3].imshow(vil[:,:,frame_idx],cmap=vil_cmap,norm=vil_norm,vmin=vil_vmin,vmax=vil_vmax), axs[3].set_title('VIL')
axs[4].imshow(lght[:,:,frame_idx],cmap=lght_cmap,norm=lght_norm,vmin=lght_vmin,vmax=lght_vmax), axs[4].set_title('Lightning')
```

```
/var/folders/_8/112gzk_92_n32czpf637ft9r0000gn/T/ipykernel_1674/3501669142.py:13: MatplotlibDeprecationWarning: Passing parameters 
ce 3.3 and will become an error two minor releases later. Please pass vmin/vmax directly to the norm when creating it.
  axs[0].imshow(vis[:,:,frame_idx],cmap=vis_cmap,norm=vis_norm,vmin=vis_vmin,vmax=vis_vmax), axs[0].set_title('VIS')
```

[44]: `(<matplotlib.image.AxesImage at 0x7f99784085b0>, Text(0.5, 1.0, 'Lightning'))`



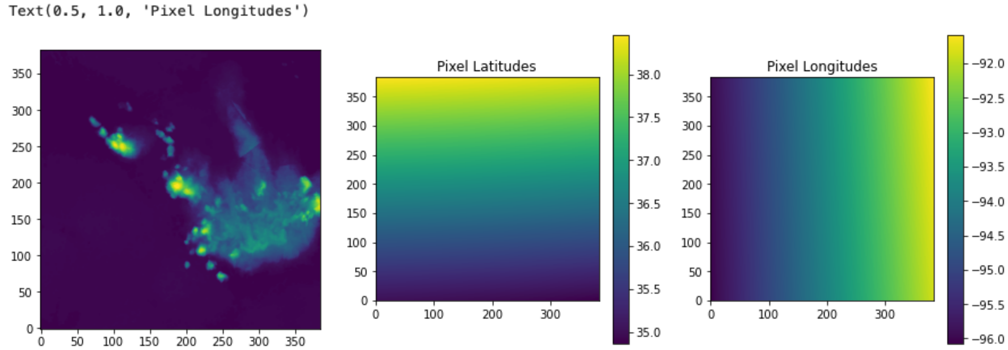
Georeferencing an Event

Boto3.ipynb | Sevir Tutorial.ipynb

Code

[86]:
```python
proj = LaeaProjection( sample_event[sample_event.img_type=='vil'].squeeze() )
X,Y = np.meshgrid(np.arange(vil.shape[0]),np.arange(vil.shape[0]))
lons,lats = proj.inverse(X,Y)

# Plot with origin='lower' so up corresponds to north.
fig,ax=plt.subplots(1,3,figsize=(15,5))
ax[0].imshow(vil[:,:,20],origin='lower')
la = ax[1].imshow(lats,origin='lower')
fig.colorbar(la,ax=ax[1])
ax[1].set_title('Pixel Latitudes')
lo = ax[2].imshow(lons,origin='lower')
fig.colorbar(lo,ax=ax[2])
ax[2].set_title('Pixel Longitudes')
```

[86]: `Text(0.5, 1.0, 'Pixel Longitudes')`



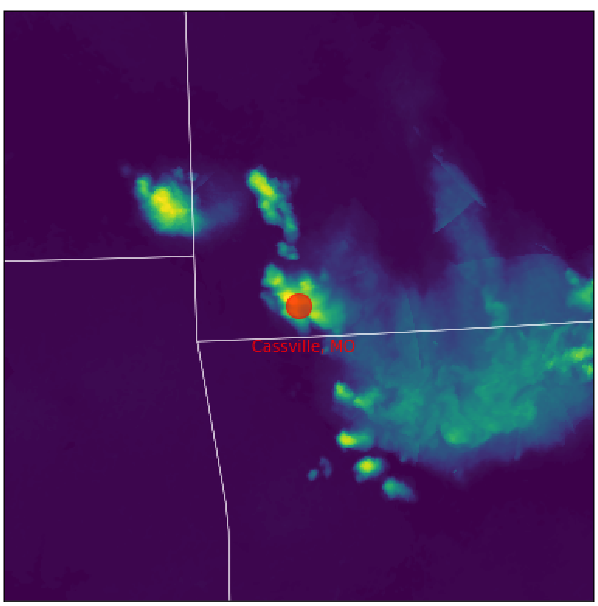
Adding markers for particular locations by converting from lat/lon coordinates into the image pixel coordinates

Boto3.ipynb | Sevir Tutorial.ipynb

Code

```
import os
os.environ["PROJ_LIB"] = os.path.join(os.environ["CONDA_PREFIX"], "share", "proj")
import numpy
warnings.filterwarnings('ignore')
from mpl_toolkits.basemap import Basemap

s=sample_event[sample_event.img_type=='vil'].squeeze()
fig,ax=plt.subplots(1,1,figsize=(7,7))
m = Basemap(llcrnrlat=s.llcrnrlat, llcrnrlon=s.llcrnrlon,
            urcrnrlat=s.urcrnrlat,urcrnrlon=s.urcrnrlon,
            width=s.width_m, height=s.height_m,
            lat_0=38, lon_0=-98,
            projection='laea',
            resolution='i',
            ax=ax)
m.drawstates(color='w')
try:
    m.drawcoastlines(color='w')
except:
    pass
m.drawcountries(color='w')
m.imshow(vil[:,:,24])

lat,lon = 36.69, -93.87
x,y=m(lon,lat) # will be in projection coordinates
m.plot(x,y,linestyle='none', marker="o", markersize=16, alpha=0.6, c="red")
ax.text(x-30e3,y-30e3,'Cassville, MO',color='r')
```

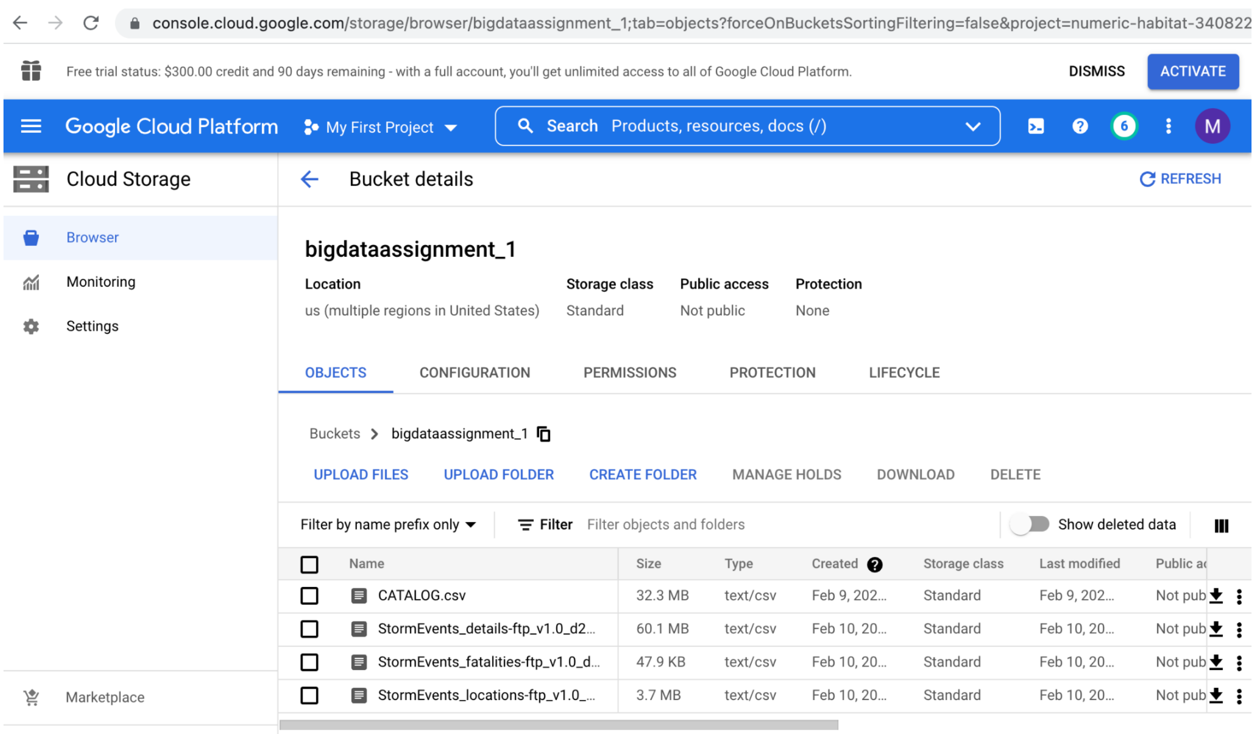
```
[84]: Text(162107.48498892388, 162449.59602166762, 'Cassville, MO')
```



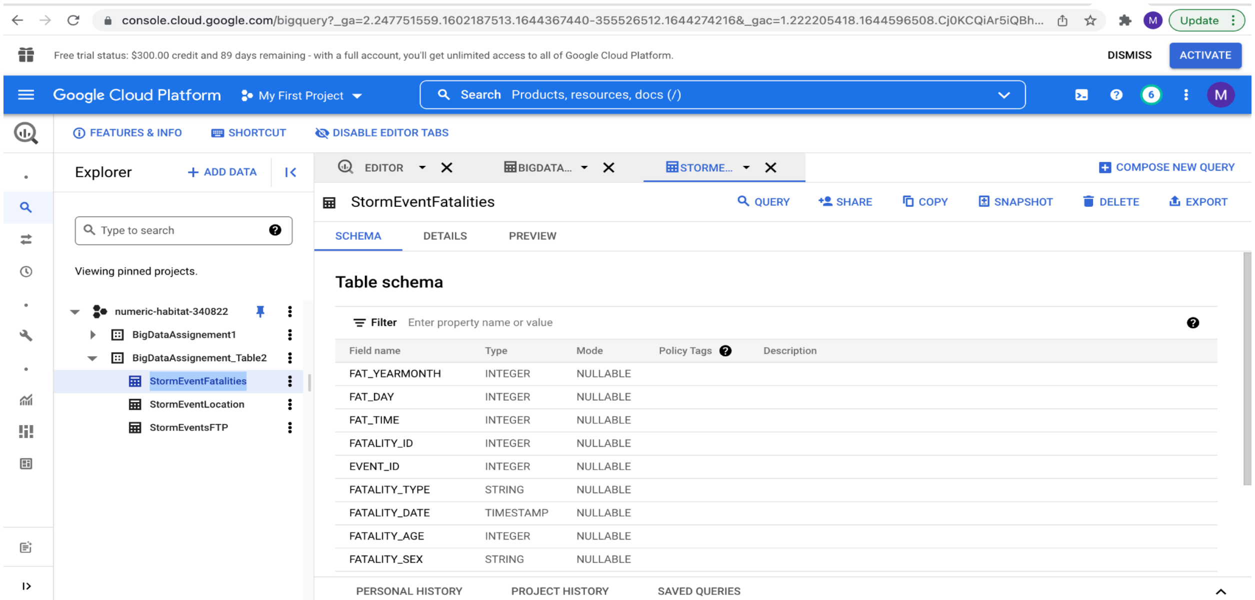
GCP and Bigquery

Adding datasets to GCP bucket

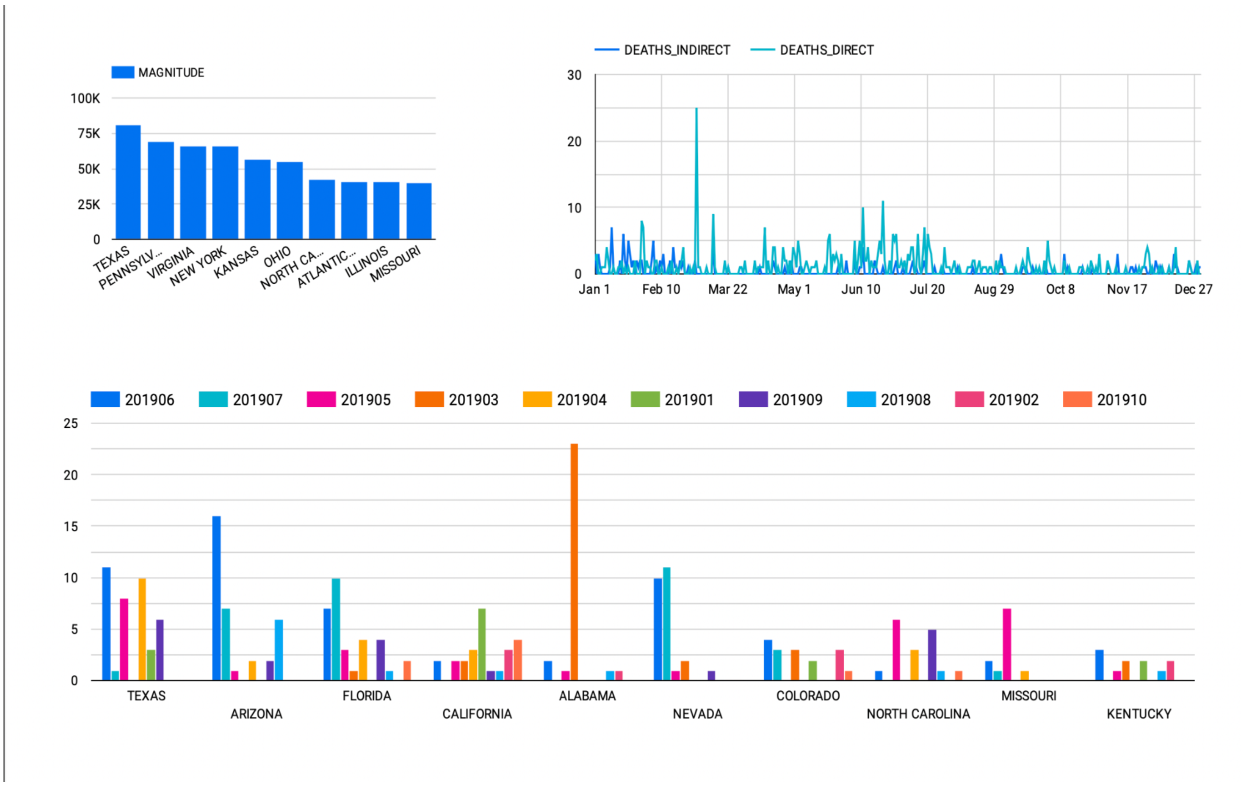
Catalog.csv and 2019 storm events files are added to the GCP bucket.



Creating tables using Big Query.

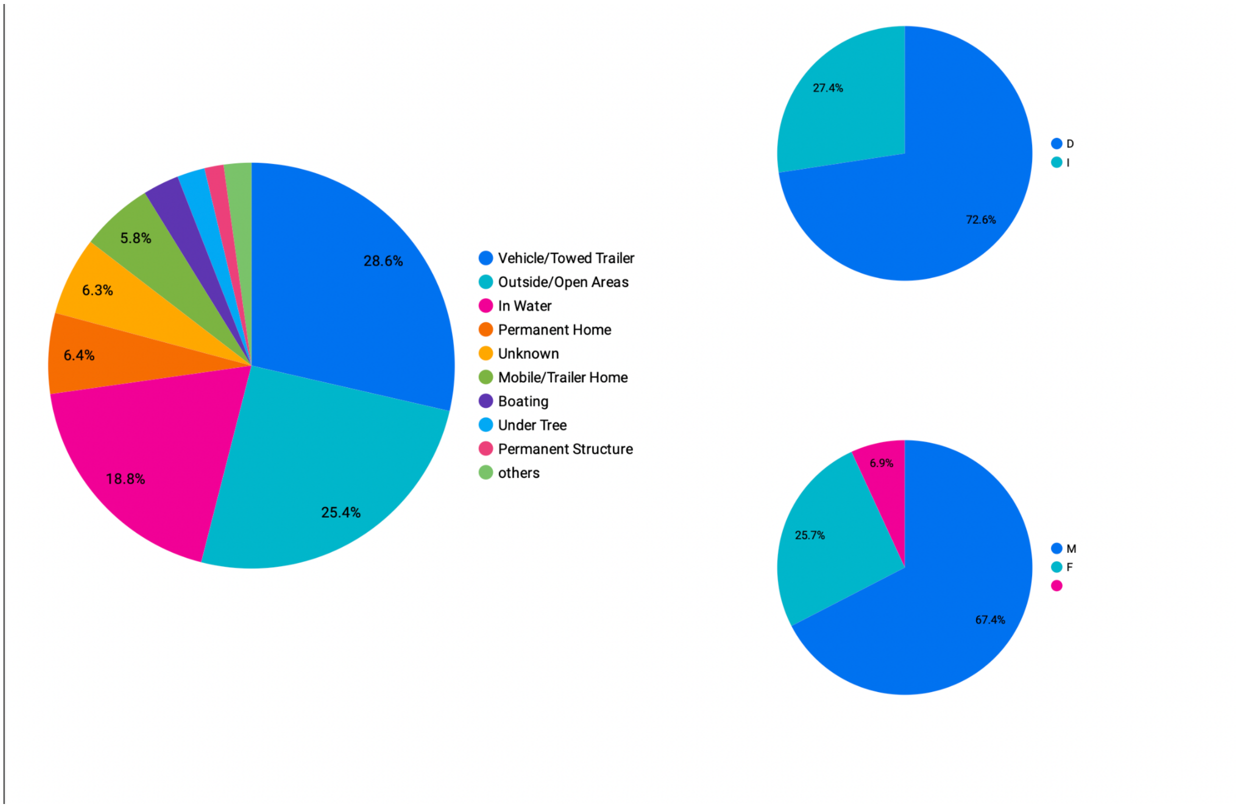


Connecting the Google data studio via big query to generate reports and visualize the data.



These graphs are generated for the StormEventsFTP file.

1. The first bar chart represents the magnitude of storm events in each state.
2. The second time series chart represents the count of deaths directly caused by storms and indirectly caused by storms over a time period of a year.
3. The third combo chart represents the death count in each state for every month in the year 2019.



These graphs are generated for the StormEventsfatalities file.

1. The first pie chart represents the area where the fatalities have happened. We can see that apporx 28.6% of the population which was using vehicles or towed trailers were injured during storm events.
2. The second Pie chart represents the direct fatality caused by storm and the indirect fatality caused by storm. We can see that the direct fatality is larger in the number.
3. The third pie chart represents the gender of the people affected during the storm.