ProjectA- Working with National Weather Service Data

Deepak Khirey

# Introduction

National Centers for Environmental Information stores and maintains database of Storm events in USA from 1950 to 2019. The data is stored a zipped CSV files. It is a very reliable source of information to study patterns of events and damage caused by it.

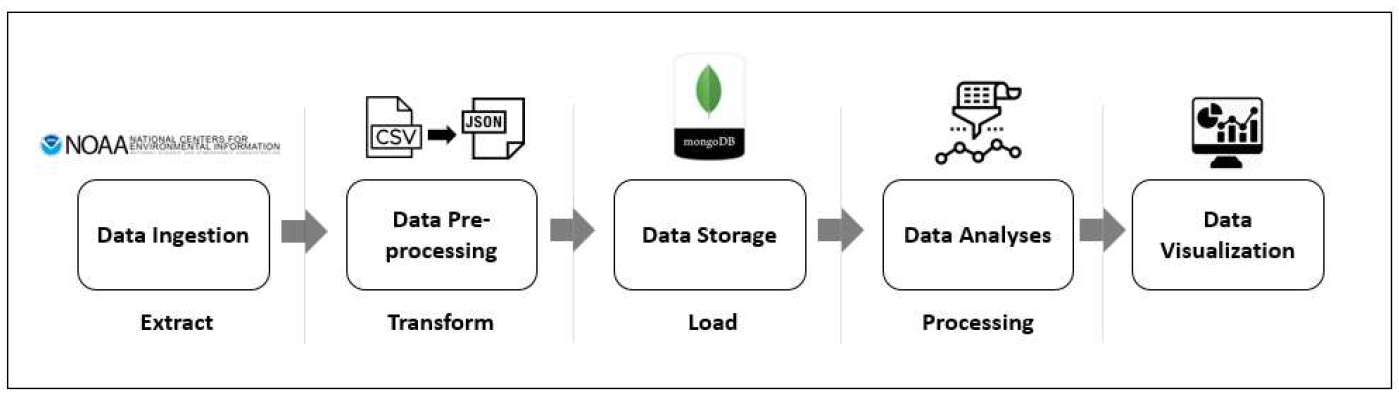
I will use a subset of this dataset by filtering Tornado events from 2008 to 2018. This data will be subjected to pipeline stages which will progressively make it suitable for data analytics and then finding useful insights from the data through queries and visualizations. Finally, a conclusion around visible patterns will be outcome of this exercise.

# Pipeline Description

The Data Pipeline used in this project consists of following stages - **Download, Extract, Transform, Load, Analyze and Visualize**. This pipeline closely follows the steps for **USGS Data Lifecycle** model. Fortunately, NOAA has provided the data in structured CSV format, hence we can perform batch operations throughout the pipeline. All actions to be performed are scripted in python in a utility code project\_utilities.py.

## USGS Data Lifecycle

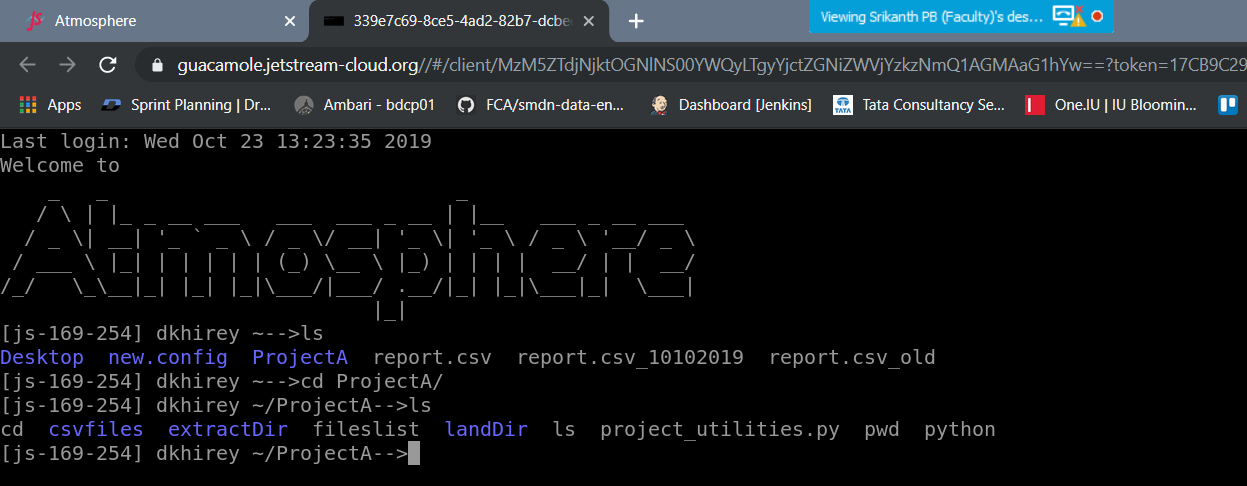
## Data Pipeline



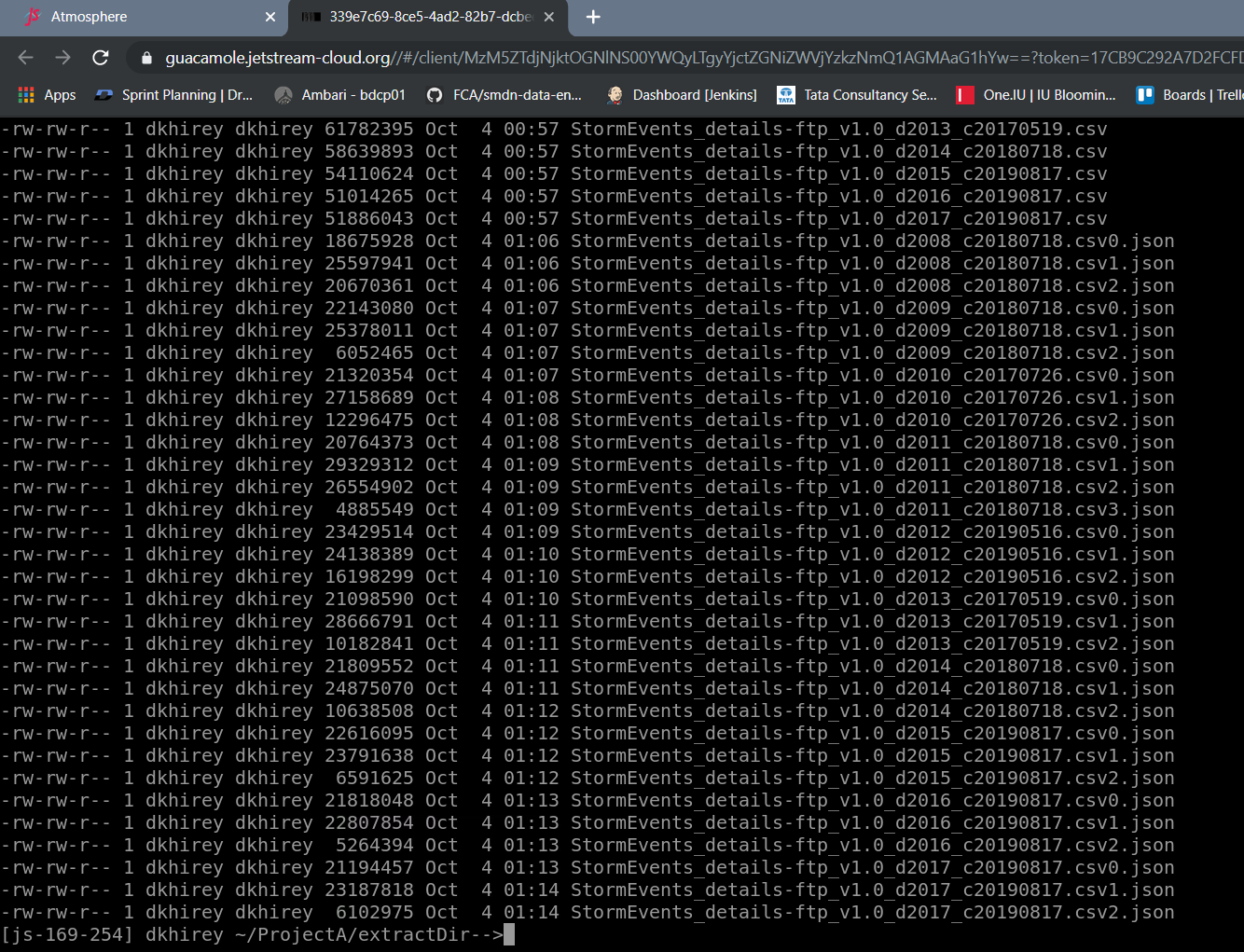
* Download - The input data is to be downloaded over the internet. For this purpose, a filelist is provided so that each of the file can be retrieved from URL as required. Files will be downloaded to landing directory on the virtual environment.
* Extract - Once the files downloaded, they are in zipped format. Hence they need to be unzipped and extracted to a directory. This step will create a corpus of CSV data for further storage and analysis.
* Transform - We are going to store extracted data in MongoDB. We have chosen this database as our choice because it is a NoSQL database and it is quite flexible on the schema requirements. Also it is suitable for large amount of data. However, it needs data to be in JSON format. Hence we have to transform CSV data into JSON such that each row of CSV becomes separate JSON file. We also make sure to filter out empty records in this step.
* Load - Once the data is prepared for load, we use MongoDB load API provided by pymongo package. This will insert records into database.
* Analyze - Using MongoDB queries, we can perform analysis on data to identify Top 10 events. For further analysis, we are exporting the data out of database using mongoexport utility. This provides a CSV output with required field.
* Visualize - I am using **PowerBI** as my choice of Visualization tool. I have created a dashboard to show analysis at a glance. We are also showing geospatial patterns using plotly.

# Pipeline Execution

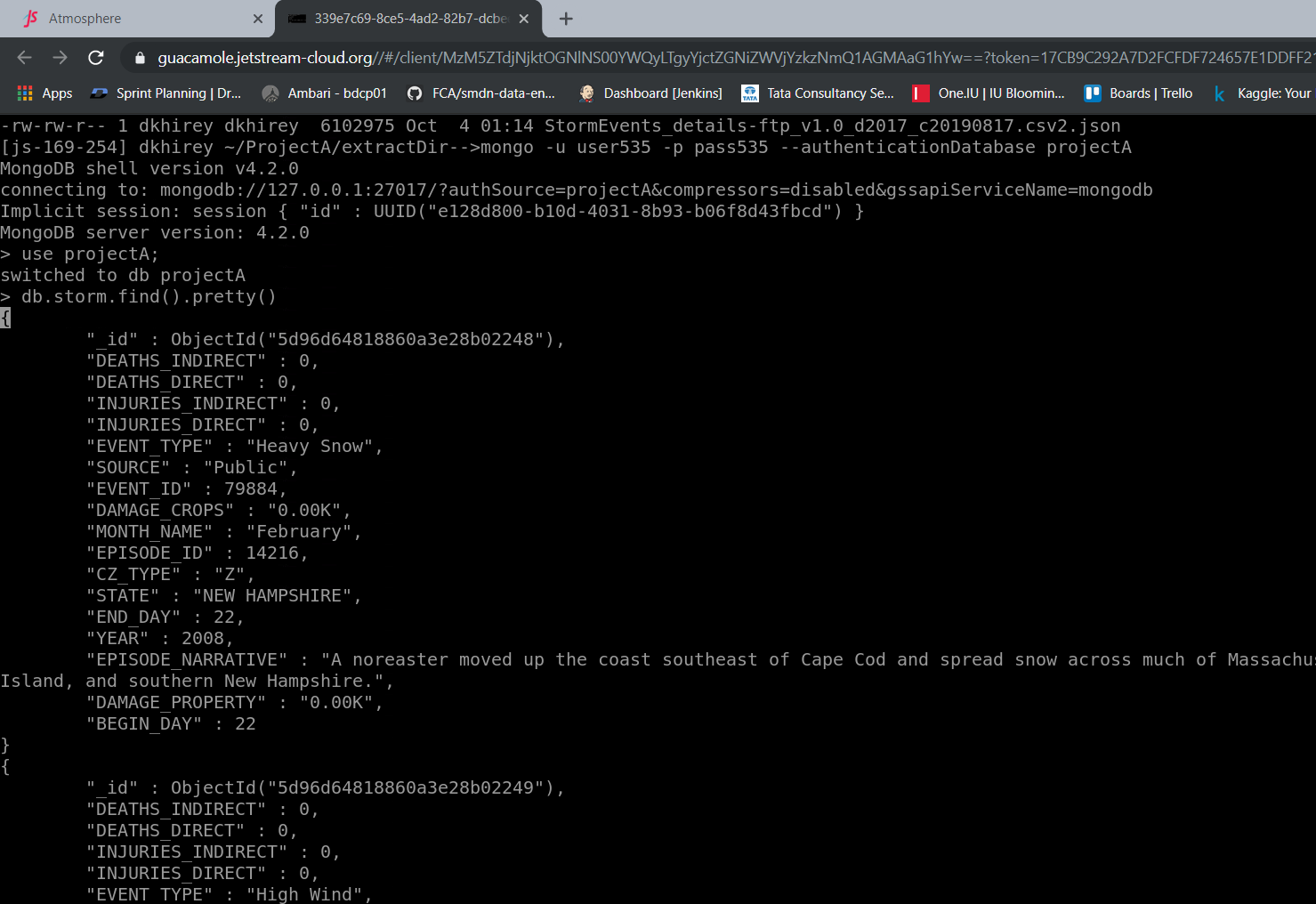
1. To perform pipeline execution, I followed steps provided in the instructions. It was pretty much elaborated description and I did not come across any issues while performing these steps.
2. First step of execution is to get Virtual environment ready. This was done using JetStream by choosing I535-MGMT-DATA-FA19 image. This image comes with python, MongoDB and python libraries installed with it.

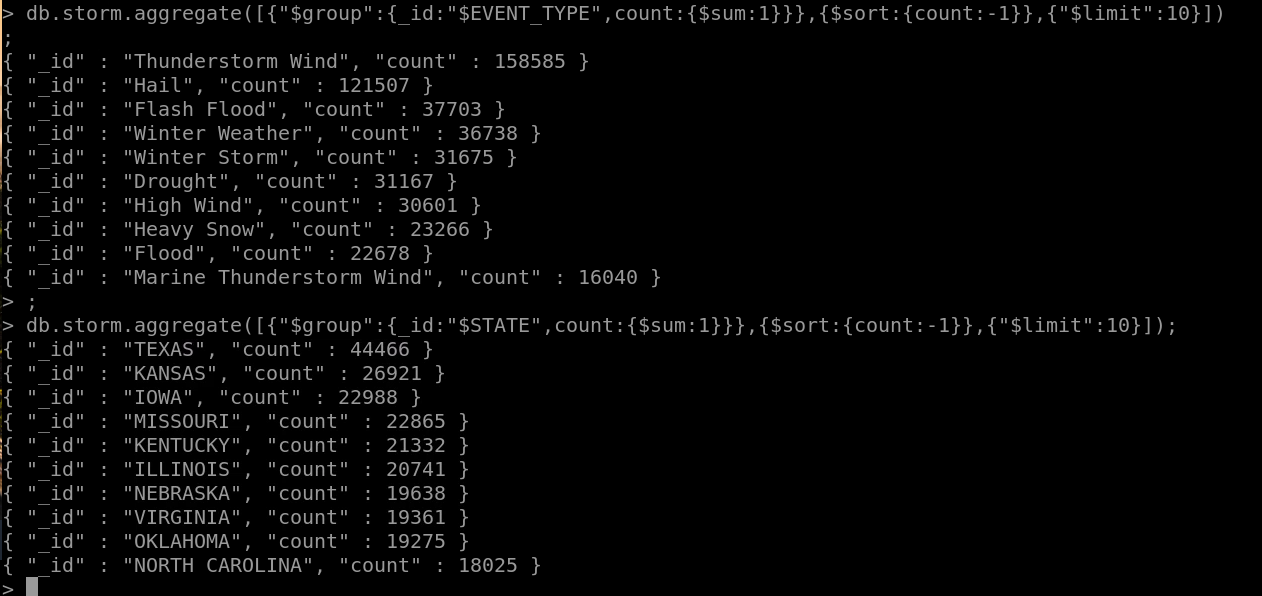


1. Next step was to deploy MongoDB database. We already have access to projectA database so I created "storm" collection in it to hold the data.
2. Then I performed steps of Download, Extract and Transform using project\_utilities.py. While downloading data, I provided year range of **2008 to 2018**. Then the zip files were downloaded to landDir. Extract step unzipped those files into extractDir directory. For transform step, I used chunksize of 10000. It created multiple JSON files in extractDir directory.

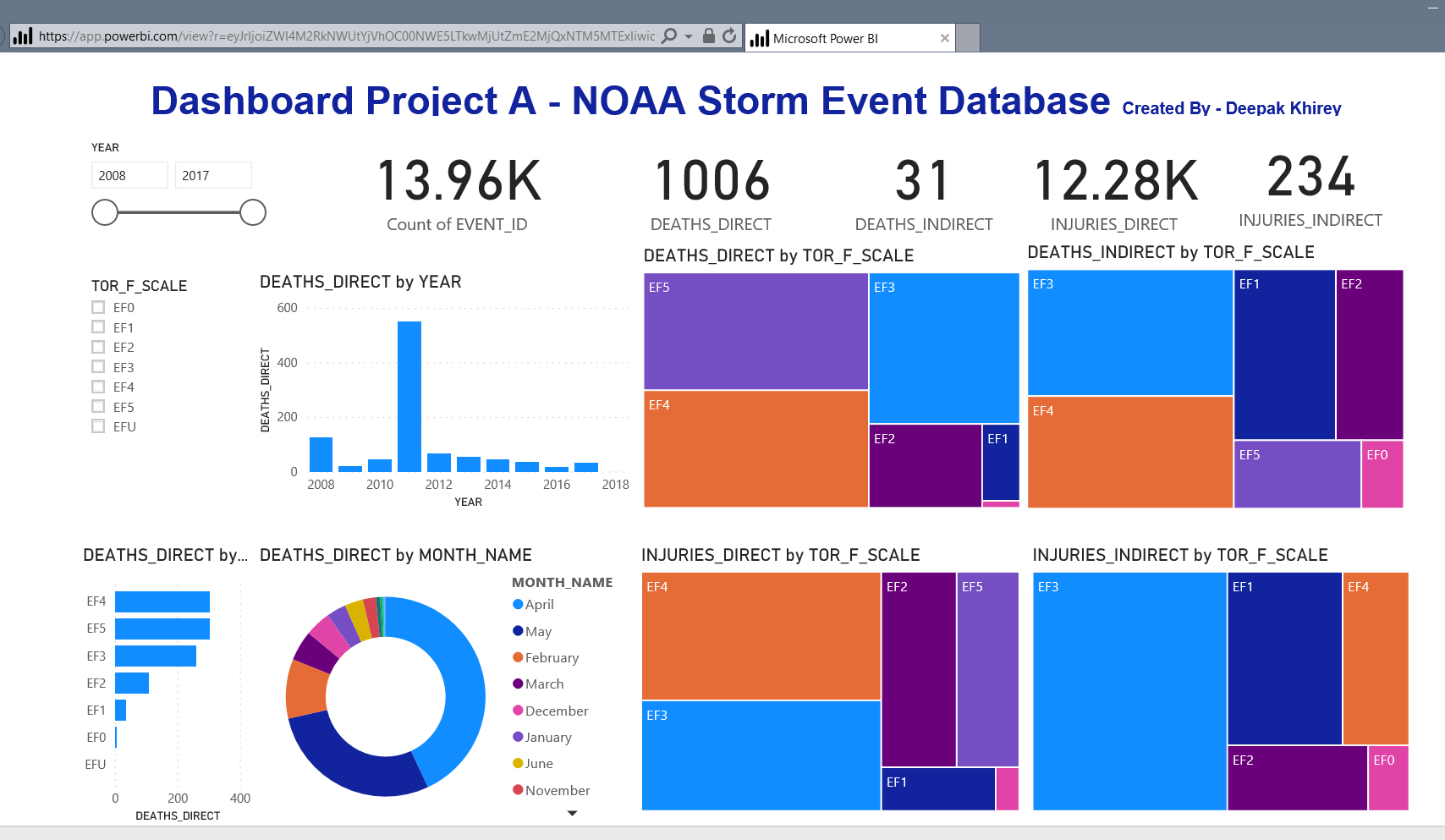


1. Load step was executed to load data into storm collection. And then performed basic operations like insert, update, delete record etc. Also tried aggregation query for Top 10 events. Finally exported data out in CSV format.





1. For visualization, using plotly library, Geospatial analysis was carried out. This was possible because the exported data had State field which can be mapped to USA geography. Also for deeper understanding, **I created a dashboard in PowerBI.** This provides a very good interactive way for user to get insights. Screenshot of dashboard is as below**. I would highly recommend to visit published dashboard at** [**https://app.powerbi.com/view?r=eyJrIjoiZWI4M2RkNWUtYjVhOC00NWE5LTkwMjUtZmE2MjQxNTM5MTExIiwidCI6ImMxYWVlMGIxLTgzZjQtNDUxOC1hMjI2LWU3NTA4YWVjNGMyZCIsImMiOjh9**](https://app.powerbi.com/view?r=eyJrIjoiZWI4M2RkNWUtYjVhOC00NWE5LTkwMjUtZmE2MjQxNTM5MTExIiwidCI6ImMxYWVlMGIxLTgzZjQtNDUxOC1hMjI2LWU3NTA4YWVjNGMyZCIsImMiOjh9)



# Conclusions

1. We see that we can establish a robust Data Pipeline which runs the dataset through different Lifecycle states. Execution of such pipeline helps us to analyze data in a very articulated way. We can run this pipeline on new data or schedule it, so as to get incremental insights about the data. It can also help to develop predictive analysis using Machine learning algorithms.
2. In a way we can consider a Data Pipeline as a shell where all components of data analysis are orchestrated.
3. From the storm dataset, we see that there have been ~14000 Tornado events since 20087 which has caused 1000 loss of lives.
4. 2011 was the deadliest year because it caused most direct deaths.
5. Tornado of type EF4 and EF5 are most frequent to occur.
6. April and May are most likely months for Tornados. It coincides with common knowledge that Tornados is a Summer phenomenon.
7. Southern states of Alabama, Texas, Missouri are most affected areas due to Tornados.
8. This data analysis helps us prepare better for Tornados and helps government plan for rescue operations, supplies of medicines etc.

# Suggestions

I have a couple of suggestions to improve overall efficiency of the pipeline.

1. **Load CSV files directly into MongoDB**

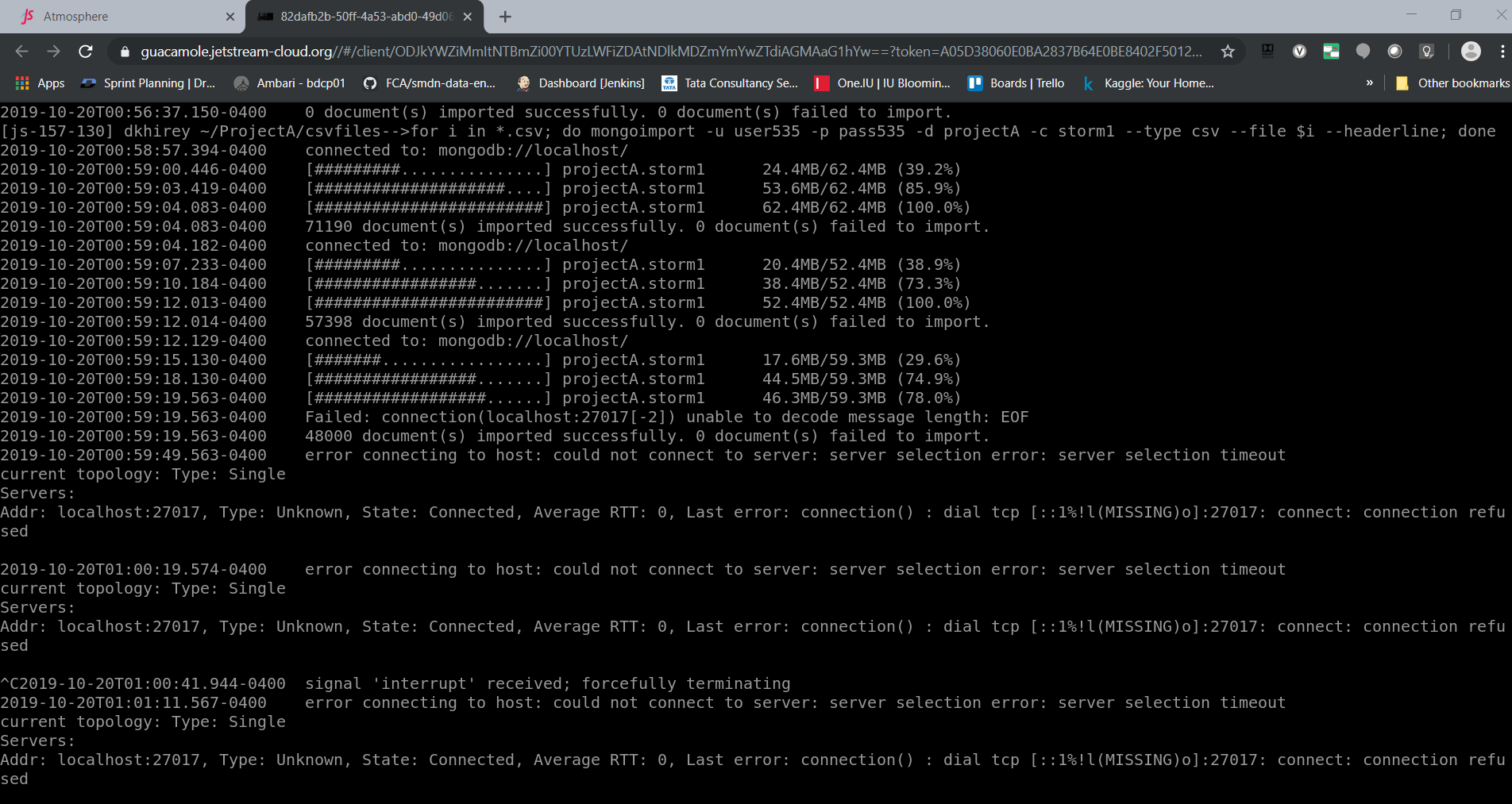
Currently in our pipeline, we are using transform step which converts CSV input to JSON files and then we load those JSON files to MongoDB. Since MongoDB 4.2, CSV import is supported which can reduce this overhead and make data ingestion faster.

I checked MongoDB documentation and tried to run mongoimport untility as below –

**for i in \*.csv; do mongoimport -u user535 -p pass535 -d projectA -c storm1 --type csv --file $i --headerline; done**

**Ref -** <https://docs.mongodb.com/manual/reference/program/mongoimport/>

This script went well for some initial CSV files and then suddenly I got error of connection refused. Please refer screenshot below for error. I believe this error could be due to heavy writing job which can be dealt with either providing chunksize or by increasing memory.



1. **Data Correction during Transform action**

INJURIES\_DIRECT field has values like 0.30K which actually should be transformed as integer 300. Integer values will help in analysis using aggregate functions like min, max, average etc. This can be achieved by calling a custom method in project\_utilities.py transform method while preparing data\_json object. The method could be like below-

import re

value\_dict = {'[kK]': '\*1e3', '[mM]': '\*1e6', '[bB]': '\*1e9'}

def find\_replace\_multi(string, dictionary):

for item in dictionary.keys():

string = re.sub(item, dictionary[item], string)

return string

and it can be called like

find\_replace\_multi(data\_json[“INJURIES\_DIRECT”], value\_dict)

# List of Resources

1. NOAA Storm Event Dataset
2. I535-MGMT-DATA-FA19 image on Jetstream Atmosphere
   * MongoDB 4.2
   * Python 2.7
   * python libraries pandas, pymongo
3. python script project\_utilities.py
4. Jupyter Notebook
5. python libraries pandas, plotly
6. Microsoft PowerBI Desktop
7. WinSCP
8. Putty