EPIDEMIC OUTBREAK PREDICTION

Team Members
Joinal Ahmed (Team Lead)
Zahoor Ansari
Soham Shinde
Ashlesha Somavanshi

Table of content

Abstract

Reliable predictions of the dynamics of infectious diseases are extremely valuable to public health organizations planning interventions to reduce or prevent the spread of disease. With the growth of big data in the fields of health and biomedicine, accurate analysis of such data helps early detection of diseases and better patient care. With enormous computing power, it is now highly feasible to use to predict and manage outbreaks. Our idea is to analyze and determine the spread of epidemics in villages and suburbs, where medical care may not be available. We want to build a machine learning model that can predict the dynamics of an epidemic and tell us where the next epidemic is most likely to break out. Our method takes into account the geography, climate, and population distribution of the affected area because these are relevant characteristics and subtly contribute to the dynamics of the disease epidemic. Our model will help health authorities take appropriate measures to ensure that there are sufficient resources to meet demand and, where possible, to curb the emergence of such epidemics.

Introduction

Environmental change, human demography, international travel, microbial evolution and the breakdown of public health facilities have all contributed to the changing spectrum of infectious diseases with which the global community is challenged. Existing mechanisms for infectious disease surveillance and response are inadequate to meet the increasing needs for prevention, detection, reporting and response. The ability to predict epidemics will provide a mechanism for governments and health-care services to respond to outbreaks in a timely fashion, enabling the impact to be minimized and limited resources to be saved. For many infectious diseases, particularly those transmitted by arthropod vectors, advanced surveillance and modelling technologies incorporating environmental data create the potential to predict the temporal and spatial risk of epidemics. When combined with communication technologies, these techniques can provide important tools that are both cost-effective and timely.

As disease boundaries shift and expand to threaten new populations, there is increasing need to develop operational models with predictive capacity: ‘As more experience is gained in linking changes detected by global imaging with changes in disease patterns, geographical information systems are likely to play an increasingly important role in forecasting outbreaks, especially those of vector-borne diseases such as zika virus’.

Advances in disease surveillance systems, epidemiological modelling and information technology have generated the expectation that early warning systems are not only feasible but necessary tools to combat the re-emergence and spread of infectious diseases. While much of the environmental data used in these systems are available free or at low cost, the quality and availability of epidemiological data vary enormously. The length and spatial extent of the epidemiological data series are particularly important for investigating annual and inter-annual patterns of disease.

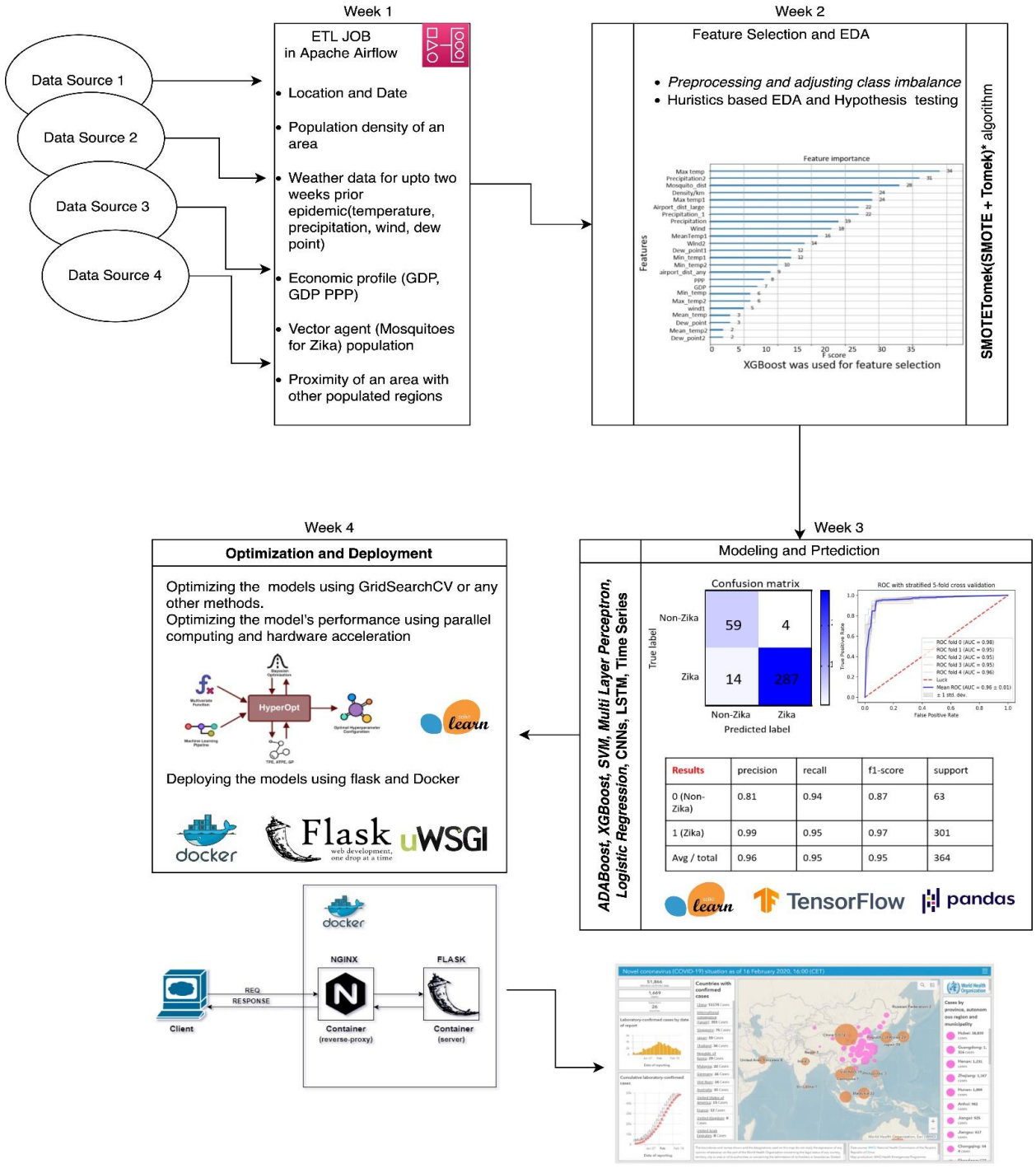
Problem Statement

COVID19 has shown the real face of our healthcare infrastructure, planning, and readiness to address a pandemic with limited resources, unprepared staff, and broken supply chains. Government agencies, if provided with this information well in advance, can plan and execute them in a well-orchestrated manner optimizing the use of staff, resources at their disposal. In this project, we aim is to analyze and build a multimodal model to predict the likelihood of an area having an outbreak.

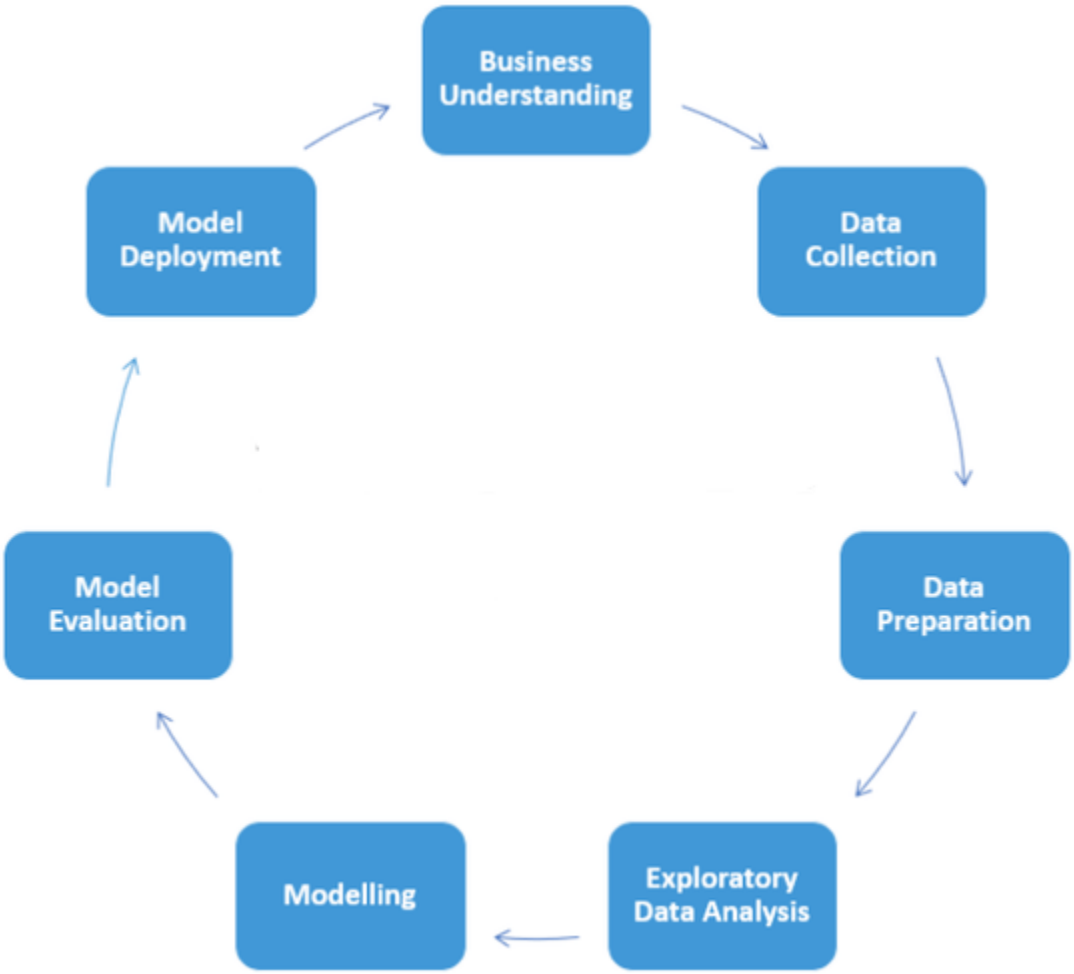
Objectives

- Curbing the preventable disease-related suffering
- Minimize financial burden on governments and health care systems by providing them first-hand information about outbreak prone areas and causative agents for the spread of epidemic
- Given an area where an epidemic outbreak has occurred, our ML model should be able to identify next outbreak prone areas and identify features which contribute significantly in the spread of the outbreak
- Epidemics of infectious disease are generally caused by several factors including a change in the ecology of the host population,
- Change in the pathogen reservoir or the introduction of an emerging pathogen to a host population.
- The feature vectors in our model are general enough to be adapted with a slight change to study any epidemic disease.

Project Architecture



Data Science Lifecycle



Technology Stack

Following technologies are used to develop this project

About Zika Virus

History

Zika virus is a mosquito-borne flavivirus that was first identified in Uganda in 1947 in monkeys. It was later identified in humans in 1952 in Uganda and the United Republic of Tanzania.

Outbreaks of Zika virus disease have been recorded in Africa, the Americas, Asia and the Pacific. From the 1960s to 1980s, rare sporadic cases of human infections were found across Africa and Asia, typically accompanied by mild illness.

The first recorded outbreak of Zika virus disease was reported from the Island of Yap (Federated States of Micronesia) in 2007. This was followed by a large outbreak of Zika virus infection in French Polynesia in 2013 and other countries and territories in the Pacific. In March 2015, Brazil reported a large outbreak of rash illness, soon identified as Zika virus infection, and in July 2015, found to be associated with Guillain-Barré syndrome.

In October 2015, Brazil reported an association between Zika virus infection and microcephaly. Outbreaks and evidence of transmission soon appeared throughout the Americas, Africa, and other regions of the world. To date, a total of 86 countries and territories have reported evidence of mosquito-transmitted Zika infection.

Transmission

Zika virus is primarily transmitted by the bite of an infected mosquito from the *Aedes* genus, mainly *Aedes aegypti*, in tropical and subtropical regions. *Aedes* mosquitoes usually bite during the day, peaking during early morning and late afternoon/evening. This is the same mosquito that transmits dengue, chikungunya and yellow fever.

Zika virus is also transmitted from mother to fetus during pregnancy, through sexual contact, transfusion of blood and blood products, and organ transplantation.

Signs and symptoms

The incubation period (the time from exposure to symptoms) of Zika virus disease is estimated to be 3–14 days. The majority of people infected with Zika virus do not develop symptoms. Symptoms are generally mild including fever, rash, conjunctivitis, muscle and joint pain, malaise, and headache, and usually last for 2–7 days.

Complications of Zika virus disease

Zika virus infection during pregnancy is a cause of microcephaly and other congenital abnormalities in the developing fetus and newborn. Zika infection in pregnancy also results in pregnancy complications such as fetal loss, stillbirth, and preterm birth.

Zika virus infection is also a trigger of Guillain-Barré syndrome, neuropathy and myelitis, particularly in adults and older children.

Research is ongoing to investigate the effects of Zika virus infection on pregnancy outcomes, strategies for prevention and control, and effects of infection on other neurological disorders in children and adults.

There is no treatment available for Zika virus infection or its associated diseases.

Dataset Collection

Apache Airflow

Apache Airflow is a platform for programmatically authoring, scheduling, and monitoring workflows. It is completely open source and is especially useful in architecting and orchestrating complex data pipelines. Airflow was originally created to solve the issues that come with long-running cron tasks and hefty scripts, but it's since grown to become one of the most powerful open source data pipeline platforms out there.

Airflow has a couple of key benefits, namely:

- It's dynamic: Anything you can do in Python, you can do in Airflow.
- It's extensible: Airflow has readily available plugins for interacting with most common external systems. You can also create your own plugins as needed.
- It's scalable: Teams use Airflow to run thousands of different tasks per day.

With Airflow, workflows are architected and expressed as Directed Acyclic Graphs (DAGs), with each node of the DAG representing a specific task. Airflow is designed with the belief that all data pipelines are best expressed as code, and as such is a code-first platform where you can quickly iterate on workflows.

Key concepts used in the project:

DAG

A Directed Acyclic Graph, or DAG, is a data pipeline defined in Python code. Each DAG represents a collection of tasks you want to run and is organized to show relationships between tasks in Airflow's UI.

Tasks

Tasks represent each node of a defined DAG. They are visual representations of the work being done at each step of the workflow, with the actual work that they represent being defined by operators.

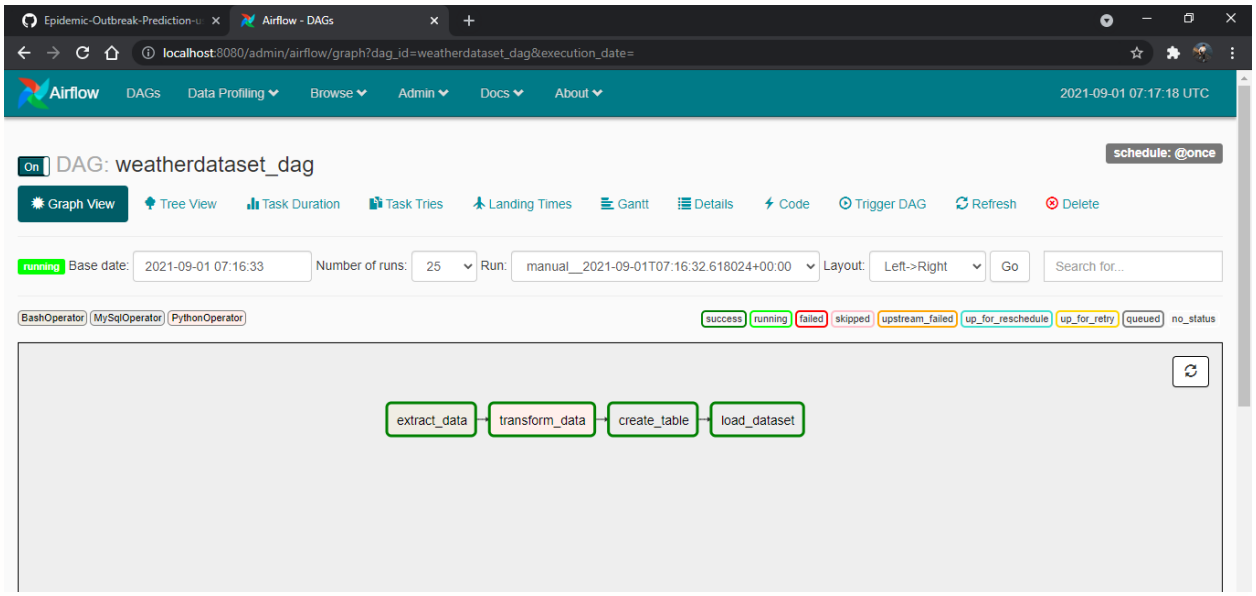
Operators

Operators are the building blocks of Airflow, and determine the actual work that gets done. They can be thought of as a wrapper around a single task, or node of a DAG, that defines how that task will be run. DAGs make sure that operators get scheduled and run in a certain order, while operators define the work that must be done at each step of the process.

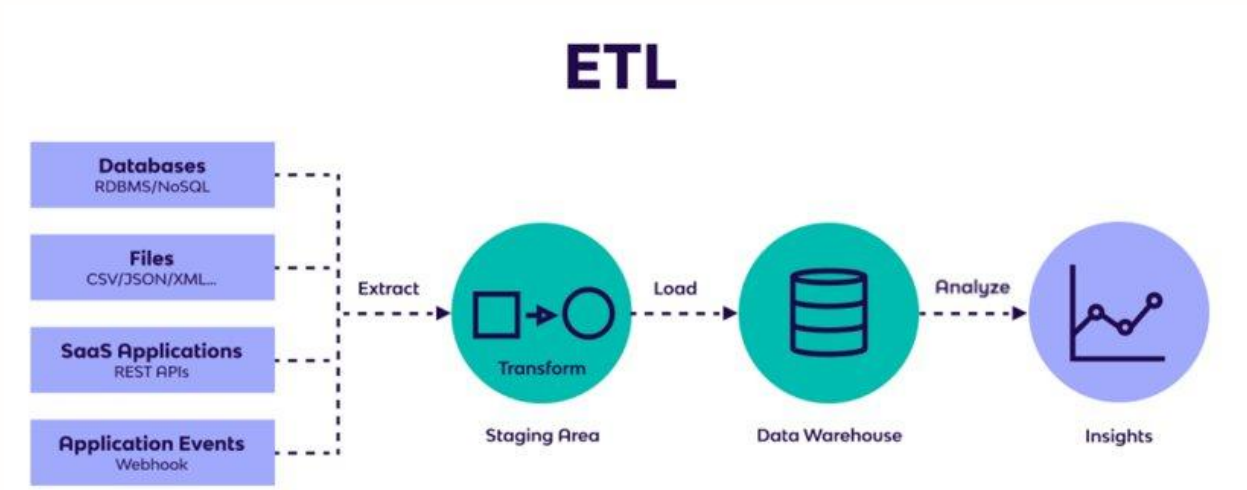
Hooks

Hooks are Airflow's way of interfacing with third-party systems. They allow you to connect to external APIs and databases like Hive, S3, GCS, MySQL, Postgres, etc. They act as building blocks for operators.

A DAG to store weather dataset into MySQL database:



Following is the flow of the DAG:



Zika Virus Dataset

Zika Data Repository maintained by the Center for Disease Control and Prevention contains publicly available data for Zika epidemic. It had enough data for building and testing our model.

https://github.com/cdcepi/zika

Features in the dataset:

- **report_date:** The report date is the date that the report was published.
  report_date: 2016-01-30
- **location:** The location at which case was reported.
  location: Brazil-Bahia
- **location_type**: A location code should also be included indicating: city, district, municipality, county, state, province, or country.
  location_type: state
- **data_field:** The data field is a short description of what data is represented in the row and is related to a specific definition defined by the report from which it comes.
  data_field: microcephaly_under_investigation
- **data_field_code:** This code is defined in the country data guide.
  data_field_code: BR0001
- **value:** The observation indicated for the specific 'report_date', 'location', 'data_field'
  value: 10

➔ **unit:** The unit of measurement for the 'data_field'.
unit: cases

	A	B	C	D	E	F	G
1	report_date	location	location_type	data_field	data_field_code	value	unit
2	1/12/2017	Argentina-Buenos_Aires	province	cumulative_confirmed_imported_cases	AR0003	0	cases
3	1/12/2017	Argentina-CABA	province	cumulative_confirmed_imported_cases	AR0003	1	cases
4	1/12/2017	Argentina-Cordoba	province	cumulative_confirmed_imported_cases	AR0003	2	cases
5	1/12/2017	Argentina-Entre_Rios	province	cumulative_confirmed_imported_cases	AR0003	0	cases
6	1/12/2017	Argentina-Santa_Fe	province	cumulative_confirmed_imported_cases	AR0003	2	cases
7	1/12/2017	Argentina-Mendoza	province	cumulative_confirmed_imported_cases	AR0003	1	cases
8	1/12/2017	Argentina-San_Juan	province	cumulative_confirmed_imported_cases	AR0003	0	cases
9	1/12/2017	Argentina-San_Luis	province	cumulative_confirmed_imported_cases	AR0003	0	cases
10	1/12/2017	Argentina-Chaco	province	cumulative_confirmed_imported_cases	AR0003	0	cases
11	1/12/2017	Argentina-Corrientes	province	cumulative_confirmed_imported_cases	AR0003	0	cases
12	1/12/2017	Argentina-Formosa	province	cumulative_confirmed_imported_cases	AR0003	0	cases
13	1/12/2017	Argentina-Misiones	province	cumulative_confirmed_imported_cases	AR0003	0	cases
14	1/12/2017	Argentina-Catamarca	province	cumulative_confirmed_imported_cases	AR0003	0	cases
15	1/12/2017	Argentina-Jujuy	province	cumulative_confirmed_imported_cases	AR0003	0	cases
16	1/12/2017	Argentina-La_Rioja	province	cumulative_confirmed_imported_cases	AR0003	0	cases
17	1/12/2017	Argentina-Salta	province	cumulative_confirmed_imported_cases	AR0003	0	cases
18	1/12/2017	Argentina-Sgo_Del_Estero	province	cumulative_confirmed_imported_cases	AR0003	0	cases
19	1/12/2017	Argentina-Tucuman	province	cumulative_confirmed_imported_cases	AR0003	4	cases
20	1/12/2017	Argentina-Chubut	province	cumulative_confirmed_imported_cases	AR0003	0	cases
21	1/12/2017	Argentina-La_Pampa	province	cumulative_confirmed_imported_cases	AR0003	0	cases

zikadataset

Weather Dataset:

Source: https://www.worldweatheronline.com/developer/api/historical-weather-api.aspx

Features in the dataset:

- date_time
- maxtempC
- mintempC
- totalSnow_cm
- sunHour
- uvIndex
- moon_illumination
- moonrise
- moonset
- sunrise
- sunset
- DewPointC
- FeelsLikeC
- HeatIndexC
- WindChillC
- WindGustKmph
- cloudcover
- humidity
- precipMM
- pressure
- tempC
- visibility
- winddirDegree
- windspeedKmph
- location

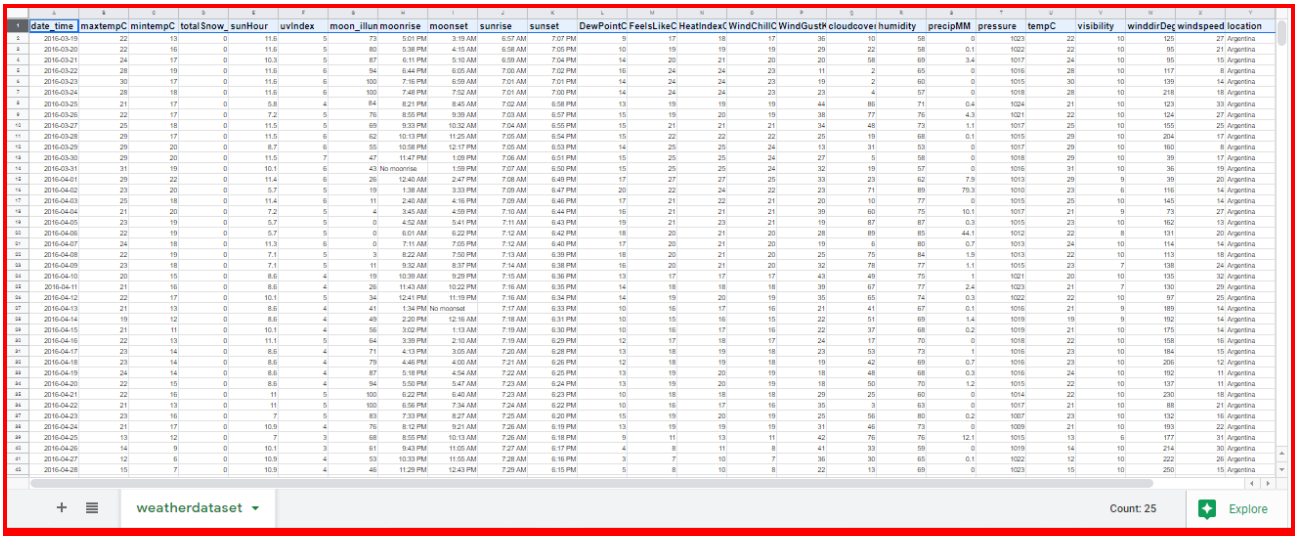
Method of extraction:

```
from wwo_hist import retrieve_hist_data

frequency = 24
api_key = '092a2ffeb04645c5ba855736210909'


start_date = '2016-02-04'
end_date = '2016-03-04'

location_list = ['Argentina']
hist_weather_data = retrieve_hist_data(api_key,
                                location_list,
                                start_date,
                                end_date,
                                frequency,
                                location_label = False,
                                export_csv=True)
```



date_time	maxtempC	mintempC	totalSnow_	sunHour	uvIndex	moon_illun	moonrise	moonset	sunrise	sunset	DewPointC	FeelsLikeC	HeatIndexC	WindChillC	WindGustK	cloudcover	humidity	precipMM	pressure	tempC	visibility	winddirDeg	windspeed	location
2016-03-19	22	13	0	11.6	5	73	5:01 PM	3:19 AM	6:57 AM	7:07 PM	9	17	18	17	36	10	58	0	1023	22	10	125	27	Argentina
2016-03-20	22	16	0	11.6	5	80	5:38 PM	4:15 AM	6:58 AM	7:05 PM	10	19	19	19	29	22	58	0.1	1022	22	10	95	21	Argentina
2016-03-21	24	17	0	10.3	5	87	6:11 PM	5:10 AM	6:59 AM	7:04 PM	14	20	21	20	20	58	69	3.4	1017	24	10	95	15	Argentina
2016-03-22	28	19	0	11.6	6	94	6:44 PM	6:05 AM	7:00 AM	7:02 PM	16	24	24	23	11	2	65	0	1016	28	10	117	8	Argentina
2016-03-23	30	17	0	11.6	6	100	7:16 PM	6:59 AM	7:01 AM	7:01 PM	14	24	24	23	19	2	60	0	1015	30	10	139	14	Argentina
2016-03-24	28	18	0	11.6	6	100	7:48 PM	7:52 AM	7:01 AM	7:00 PM	14	24	24	23	23	4	57	0	1018	28	10	218	18	Argentina
2016-03-25	21	17	0	5.8	4	84	8:21 PM	8:45 AM	7:02 AM	6:58 PM	13	19	19	19	44	86	71	0.4	1024	21	10	123	33	Argentina
2016-03-26	22	17	0	7.2	5	76	8:55 PM	9:39 AM	7:03 AM	6:57 PM	15	19	20	19	38	77	76	4.3	1021	22	10	124	27	Argentina
2016-03-27	25	18	0	11.5	5	69	9:33 PM	10:32 AM	7:04 AM	6:55 PM	15	21	21	21	34	48	73	1.1	1017	25	10	155	25	Argentina
2016-03-28	29	17	0	11.5	6	62	10:13 PM	11:25 AM	7:05 AM	6:54 PM	15	22	22	22	25	19	68	0.1	1015	29	10	204	17	Argentina
2016-03-29	29	20	0	8.7	6	55	10:58 PM	12:17 PM	7:05 AM	6:53 PM	14	25	25	24	13	31	53	0	1017	29	10	160	8	Argentina
2016-03-30	29	20	0	11.5	7	47	11:47 PM	1:09 PM	7:06 AM	6:51 PM	15	25	25	24	27	5	58	0	1018	29	10	39	17	Argentina
2016-03-31	31	19	0	10.1	6	43	No moonrise	1:59 PM	7:07 AM	6:50 PM	15	25	25	24	32	19	57	0	1016	31	10	36	19	Argentina
2016-04-01	29	22	0	11.4	6	35	12:40 AM	2:47 PM	7:08 AM	6:49 PM	17	27	27	25	33	23	62	7.9	1013	29	9	39	20	Argentina
2016-04-02	23	20	0	5.7	5	19	1:38 AM	3:33 PM	7:09 AM	6:47 PM	20	22	24	22	23	71	89	79.3	1010	23	6	116	14	Argentina
2016-04-03	25	18	0	11.4	6	11	2:40 AM	4:16 PM	7:09 AM	6:46 PM	17	21	22	21	20	10	77	0	1015	25	10	145	14	Argentina
2016-04-04	21	20	0	7.2	5	4	3:45 AM	4:59 PM	7:10 AM	6:44 PM	16	21	21	21	39	60	75	10.1	1017	21	9	73	27	Argentina
2016-04-05	23	19	0	5.7	5	0	4:52 AM	5:41 PM	7:11 AM	6:43 PM	19	21	23	21	19	87	87	0.3	1015	23	10	162	13	Argentina
2016-04-06	22	19	0	5.7	5	0	6:01 AM	6:22 PM	7:12 AM	6:42 PM	18	20	21	20	28	89	85	44.1	1012	22	8	131	20	Argentina
2016-04-07	24	18	0	11.3	6	0	7:11 AM	7:05 PM	7:12 AM	6:40 PM	17	20	21	20	19	6	80	0.7	1013	24	10	114	14	Argentina
2016-04-08	22	19	0	7.1	5	3	8:22 AM	7:50 PM	7:13 AM	6:39 PM	18	20	21	20	25	75	84	1.9	1013	22	10	113	18	Argentina
2016-04-09	23	18	0	7.1	5	11	9:32 AM	8:37 PM	7:14 AM	6:38 PM	16	20	21	20	32	78	77	1.1	1015	23	7	138	24	Argentina
2016-04-10	20	15	0	8.6	4	19	10:39 AM	9:29 PM	7:15 AM	6:36 PM	13	17	17	17	43	49	75	1	1021	20	10	135	32	Argentina
2016-04-11	21	16	0	8.6	4	26	11:43 AM	10:22 PM	7:16 AM	6:35 PM	14	18	18	18	39	67	77	2.4	1023	21	7	130	29	Argentina
2016-04-12	22	17	0	10.1	5	34	12:41 PM	11:19 PM	7:16 AM	6:34 PM	14	19	20	19	35	65	74	0.3	1022	22	10	97	25	Argentina
2016-04-13	21	13	0	8.6	4	41	1:34 PM	No moonset	7:17 AM	6:33 PM	10	16	17	16	21	41	67	0.1	1016	21	9	189	14	Argentina
2016-04-14	19	12	0	8.6	4	49	2:26 PM	12:16 AM	7:18 AM	6:31 PM	10	15	16	15	22	51	69	1.4	1019	19	9	192	14	Argentina
2016-04-15	21	11	0	10.1	4	56	3:02 PM	1:13 AM	7:19 AM	6:30 PM	10	16	17	16	22	37	68	0.2	1019	21	10	175	14	Argentina
2016-04-16	22	13	0	11.1	5	64	3:39 PM	2:10 AM	7:19 AM	6:29 PM	12	17	18	17	24	17	70	0	1018	22	10	158	16	Argentina
2016-04-17	23	14	0	8.6	4	71	4:13 PM	3:05 AM	7:20 AM	6:28 PM	13	18	19	18	23	53	73	1	1016	23	10	184	15	Argentina
2016-04-18	23	14	0	8.6	4	79	4:46 PM	4:00 AM	7:21 AM	6:26 PM	12	18	19	18	19	42	69	0.7	1016	23	10	206	12	Argentina
2016-04-19	24	14	0	8.6	4	87	5:18 PM	4:54 AM	7:22 AM	6:25 PM	13	19	20	19	18	48	68	0.3	1016	24	10	192	11	Argentina
2016-04-20	22	15	0	8.6	4	94	5:50 PM	5:47 AM	7:23 AM	6:24 PM	13	19	20	19	18	50	70	1.2	1015	22	10	137	11	Argentina
2016-04-21	22	16	0	11	5	100	6:22 PM	6:40 AM	7:23 AM	6:23 PM	10	18	18	18	29	25	60	0	1014	22	10	230	18	Argentina
2016-04-22	21	13	0	11	5	100	6:56 PM	7:34 AM	7:24 AM	6:22 PM	10	16	17	16	35	3	63	0	1017	21	10	88	21	Argentina
2016-04-23	23	16	0	7	5	83	7:33 PM	8:27 AM	7:25 AM	6:20 PM	15	19	20	19	25	56	80	0.2	1017	23	10	132	16	Argentina
2016-04-24	21	17	0	10.9	4	76	8:12 PM	9:21 AM	7:26 AM	6:19 PM	13	19	19	19	31	46	73	0	1019	21	10	193	22	Argentina
2016-04-25	13	12	0	7	3	68	8:55 PM	10:13 AM	7:26 AM	6:18 PM	9	11	13	11	42	76	76	12.1	1015	13	6	177	31	Argentina
2016-04-26	14	9	0	10.1	3	61	9:43 PM	11:05 AM	7:27 AM	6:17 PM	4	8	11	8	41	33	59	0	1019	14	10	214	30	Argentina
2016-04-27	12	6	0	10.9	4	53	10:33 PM	11:55 AM	7:28 AM	6:16 PM	3	7	10	7	36	30	65	0.1	1022	12	10	222	26	Argentina
2016-04-28	15	7	0	10.9	4	46	11:29 PM	12:43 PM	7:29 AM	6:15 PM	5	8	10	8	22	13	69	0	1023	15	10	250	15	Argentina

Population dataset

Features in the dataset:

- **location:** Location based on the Zika virus dataset.
- **density_per_km**: Number of people living in a square kilometer area.

	A	B
1	location	density_per_km
2	Argentina-Buenos_Aires	12625.80078
3	Argentina-CABA	12625.80078
4	Argentina-Cordoba	2404.108887
5	Argentina-Entre_Rios	72.49529266
6	Argentina-Santa_Fe	208.0922852
7	Argentina-Chaco	121.3316498
8	Argentina-Corrientes	837.7285156
9	Argentina-Formosa	41.44081116
10	Argentina-Misiones	414.0570068
11	Argentina-Catamarca	460.153595
12	Argentina-Jujuy	146.1084595
13	Argentina-Salta	347.3858643
14	Argentina-Sgo_Del_Estero	138.0267029
15	Argentina-Tucuman	6299.477539
16	Argentina-La_Rioja	17.69784164
17	Argentina-San_Luis	19.51153183
18	Argentina-Mendoza	2292.493164
19	Argentina-San_Juan	3898.334717
20	Argentina-Chubut	37.09564209
21	Argentina-La_Pampa	46.05672455

06_population_density

Latitude and longitude dataset:

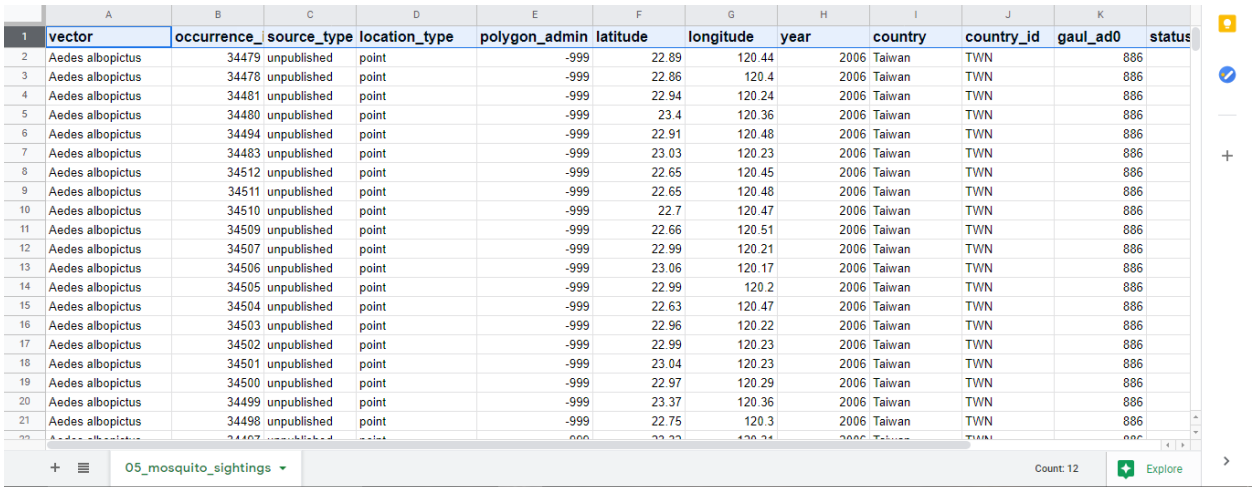
	A	B	C	D	E	F	G	H
1	location	location_type	country	province	county	city	latitude	longitude
2	Argentina-Buenos_Aires	province	Argentina	Buenos Aires			-34.603684	-58.3815591
3	Argentina-CABA	province	Argentina	Ciudad de Buenos Aires			-34.603684	-58.3815591
4	Argentina-Cordoba	province	Argentina	Cordoba			-31.420083	-64.1887761
5	Argentina-Entre_Rios	province	Argentina	Entre Rios			-31.774665	-60.4956461
6	Argentina-Santa_Fe	province	Argentina	Santa Fe			-31.610658	-60.697294
7	Argentina-Chaco	province	Argentina	Chaco			-27.425718	-59.0243784
8	Argentina-Corrientes	province	Argentina	Corrientes			-27.469213	-58.8306349
9	Argentina-Formosa	province	Argentina	Formosa			-26.185777	-58.1755669
10	Argentina-Misiones	province	Argentina	Misiones			-27.426926	-55.9467076
11	Argentina-Catamarca	province	Argentina	Catamarca			-28.469581	-65.7795441
12	Argentina-Jujuy	province	Argentina	Jujuy			-24.18434	-65.302177
13	Argentina-Salta	province	Argentina	Salta			-24.782127	-65.4231976
14	Argentina-Sgo_Del_Estero	province	Argentina	Santiago Del Estero			-27.783357	-64.264167
15	Argentina-Tucuman	province	Argentina	Tucuman			-26.808285	-65.2175903
16	Argentina-La_Rioja	province	Argentina	La Rioja			-29.413454	-66.8564579
17	Argentina-San_Luis	province	Argentina	San Luis			-33.301727	-66.3377522
18	Argentina-Mendoza	province	Argentina	Mendoza			-32.889459	-68.8458386
19	Argentina-San_Juan	province	Argentina	San Juan			-31.535107	-68.5385941
20	Argentina-Chubut	province	Argentina	Chubut			-43.293425	-65.1114818
21	Argentina-La_Pampa	province	Argentina	La Pampa			-36.614757	-64.2839209

01_latitude_longitude_google

Mosquito Dataset:

Main features of the dataset:

- **Vector:** Mosquito species- Aedes aegypti and Aedes albopictus
- **occurence_id:** An ID of the mosquito sightings
- **latitude:** latitude of the location
- **longitude:** longitude of the location
- **year:** year of observation
- **country:** country in which the mosquitoes were sighted



	A	B	C	D	E	F	G	H	I	J	K	
1	vector	occurrence_	source_type	location_type	polygon_admin	latitude	longitude	year	country	country_id	gaul_ad0	status
2	Aedes albopictus	34479	unpublished	point	-999	22.89	120.44	2006	Taiwan	TWN	886	
3	Aedes albopictus	34478	unpublished	point	-999	22.86	120.4	2006	Taiwan	TWN	886	
4	Aedes albopictus	34481	unpublished	point	-999	22.94	120.24	2006	Taiwan	TWN	886	
5	Aedes albopictus	34480	unpublished	point	-999	23.4	120.36	2006	Taiwan	TWN	886	
6	Aedes albopictus	34494	unpublished	point	-999	22.91	120.48	2006	Taiwan	TWN	886	
7	Aedes albopictus	34483	unpublished	point	-999	23.03	120.23	2006	Taiwan	TWN	886	
8	Aedes albopictus	34512	unpublished	point	-999	22.65	120.45	2006	Taiwan	TWN	886	
9	Aedes albopictus	34511	unpublished	point	-999	22.65	120.48	2006	Taiwan	TWN	886	
10	Aedes albopictus	34510	unpublished	point	-999	22.7	120.47	2006	Taiwan	TWN	886	
11	Aedes albopictus	34509	unpublished	point	-999	22.66	120.51	2006	Taiwan	TWN	886	
12	Aedes albopictus	34507	unpublished	point	-999	22.99	120.21	2006	Taiwan	TWN	886	
13	Aedes albopictus	34506	unpublished	point	-999	23.06	120.17	2006	Taiwan	TWN	886	
14	Aedes albopictus	34505	unpublished	point	-999	22.99	120.2	2006	Taiwan	TWN	886	
15	Aedes albopictus	34504	unpublished	point	-999	22.63	120.47	2006	Taiwan	TWN	886	
16	Aedes albopictus	34503	unpublished	point	-999	22.96	120.22	2006	Taiwan	TWN	886	
17	Aedes albopictus	34502	unpublished	point	-999	22.99	120.23	2006	Taiwan	TWN	886	
18	Aedes albopictus	34501	unpublished	point	-999	23.04	120.23	2006	Taiwan	TWN	886	
19	Aedes albopictus	34500	unpublished	point	-999	22.97	120.29	2006	Taiwan	TWN	886	
20	Aedes albopictus	34499	unpublished	point	-999	23.37	120.36	2006	Taiwan	TWN	886	
21	Aedes albopictus	34498	unpublished	point	-999	22.75	120.3	2006	Taiwan	TWN	886	

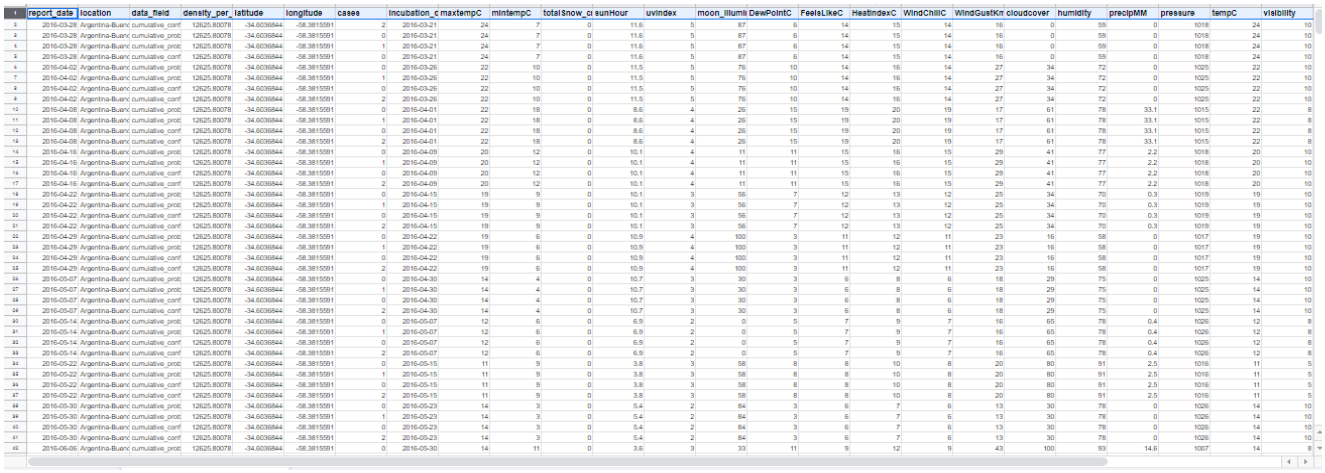
05_mosquito_sightings

Count: 12

Explore

Dataset Preparation

- We handled all the missing values from the dataset by removing them with feasible imputation methods.
- We later merged all the datasets on the basis of the ‘location’ column.
- Furthermore we added an ‘incubation_date’ column and we extracted the weather data based on the incubation date.
- Incubation date is the date 7 days prior to the report date, so as to get an idea of the effect of weather on the number of cases.
- We then created two datasets for model building- classification dataset and regression dataset.
- In the classification dataset the target variable was binary i.e. 0 for no case reported and 1 for case reported.
- We also checked for duplicate values in both the datasets and dropped all of them to maintain consistency in the dataset.



	report_date	location	data_field	density_per_	latitude	longitude	cases	incubation_d	maxtempC	mintempC	total Snow_ci	sunHour	uvindex	moon_illumi	DewPointC	FeelsLikeC	HeatIndexC	WindChillC	WindGustKm	cloudcover	humidity	precipMM	pressure	tempC	visibility
1	2016-03-28	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-03-21	24	7	0	11.6	5	87	6	14	15	14	16	0	59	0	1018	24	10
2	2016-03-28	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-03-21	24	7	0	11.6	5	87	6	14	15	14	16	0	59	0	1018	24	10
3	2016-03-28	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-03-21	24	7	0	11.6	5	87	6	14	15	14	16	0	59	0	1018	24	10
4	2016-03-28	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-03-21	24	7	0	11.6	5	87	6	14	15	14	16	0	59	0	1018	24	10
5	2016-04-02	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-03-26	22	10	0	11.5	5	76	10	14	16	14	27	34	72	0	1025	22	10
6	2016-04-02	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-03-26	22	10	0	11.5	5	76	10	14	16	14	27	34	72	0	1025	22	10
7	2016-04-02	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-03-26	22	10	0	11.5	5	76	10	14	16	14	27	34	72	0	1025	22	10
8	2016-04-02	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	2	2016-03-26	22	10	0	11.5	5	76	10	14	16	14	27	34	72	0	1025	22	10
9	2016-04-08	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-04-01	22	18	0	8.6	4	26	15	19	20	19	17	61	78	33.1	1015	22	8
10	2016-04-08	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	1	2016-04-01	22	18	0	8.6	4	26	15	19	20	19	17	61	78	33.1	1015	22	8
11	2016-04-08	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-04-01	22	18	0	8.6	4	26	15	19	20	19	17	61	78	33.1	1015	22	8
12	2016-04-08	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	2	2016-04-01	22	18	0	8.6	4	26	15	19	20	19	17	61	78	33.1	1015	22	8
13	2016-04-16	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-04-09	20	12	0	10.1	4	11	11	15	16	15	29	41	77	2.2	1018	20	10
14	2016-04-16	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	1	2016-04-09	20	12	0	10.1	4	11	11	15	16	15	29	41	77	2.2	1018	20	10
15	2016-04-16	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-04-09	20	12	0	10.1	4	11	11	15	16	15	29	41	77	2.2	1018	20	10
16	2016-04-16	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-04-09	20	12	0	10.1	4	11	11	15	16	15	29	41	77	2.2	1018	20	10
17	2016-04-22	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-04-15	19	9	0	10.1	3	56	7	12	13	12	25	34	70	0.3	1019	19	10
18	2016-04-22	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-04-15	19	9	0	10.1	3	56	7	12	13	12	25	34	70	0.3	1019	19	10
19	2016-04-22	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-04-15	19	9	0	10.1	3	56	7	12	13	12	25	34	70	0.3	1019	19	10
20	2016-04-22	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-04-15	19	9	0	10.1	3	56	7	12	13	12	25	34	70	0.3	1019	19	10
21	2016-04-29	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-04-22	19	6	0	10.9	4	100	3	11	12	11	23	16	58	0	1017	19	10
22	2016-04-29	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-04-22	19	6	0	10.9	4	100	3	11	12	11	23	16	58	0	1017	19	10
23	2016-04-29	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-04-22	19	6	0	10.9	4	100	3	11	12	11	23	16	58	0	1017	19	10
24	2016-04-29	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-04-22	19	6	0	10.9	4	100	3	11	12	11	23	16	58	0	1017	19	10
25	2016-05-07	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-04-30	14	4	0	10.7	3	30	3	6	8	6	18	29	75	0	1025	14	10
26	2016-05-07	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-04-30	14	4	0	10.7	3	30	3	6	8	6	18	29	75	0	1025	14	10
27	2016-05-07	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-04-30	14	4	0	10.7	3	30	3	6	8	6	18	29	75	0	1025	14	10
28	2016-05-07	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-04-30	14	4	0	10.7	3	30	3	6	8	6	18	29	75	0	1025	14	10
29	2016-05-14	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-05-07	12	6	0	6.9	2	0	5	7	9	7	16	65	78	0.4	1026	12	8
30	2016-05-14	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-05-07	12	6	0	6.9	2	0	5	7	9	7	16	65	78	0.4	1026	12	8
31	2016-05-14	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-05-07	12	6	0	6.9	2	0	5	7	9	7	16	65	78	0.4	1026	12	8
32	2016-05-14	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-05-07	12	6	0	6.9	2	0	5	7	9	7	16	65	78	0.4	1026	12	8
33	2016-05-22	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-05-15	11	9	0	3.8	3	58	8	8	10	8	20	80	91	2.5	1016	11	5
34	2016-05-22	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-05-15	11	9	0	3.8	3	58	8	8	10	8	20	80	91	2.5	1016	11	5
35	2016-05-22	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-05-15	11	9	0	3.8	3	58	8	8	10	8	20	80	91	2.5	1016	11	5
36	2016-05-22	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-05-15	11	9	0	3.8	3	58	8	8	10	8	20	80	91	2.5	1016	11	5
37	2016-05-30	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-05-23	14	3	0	5.4	2	84	3	6	7	6	13	30	78	0	1026	14	10
38	2016-05-30	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	1	2016-05-23	14	3	0	5.4	2	84	3	6	7	6	13	30	78	0	1026	14	10
39	2016-05-30	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	0	2016-05-23	14	3	0	5.4	2	84	3	6	7	6	13	30	78	0	1026	14	10
40	2016-05-30	Argentina-Buen	cumulative_conf	12625.80078	-34.6036844	-58.3815591	2	2016-05-23	14	3	0	5.4	2	84	3	6	7	6	13	30	78	0	1026	14	10
41	2016-06-06	Argentina-Buen	cumulative_prob	12625.80078	-34.6036844	-58.3815591	0	2016-05-30	14	11	0	3.6	3	33	11	9	12	9	43	100	93	14.6	1007	14	8

Final dataset

Shape of the dataset:

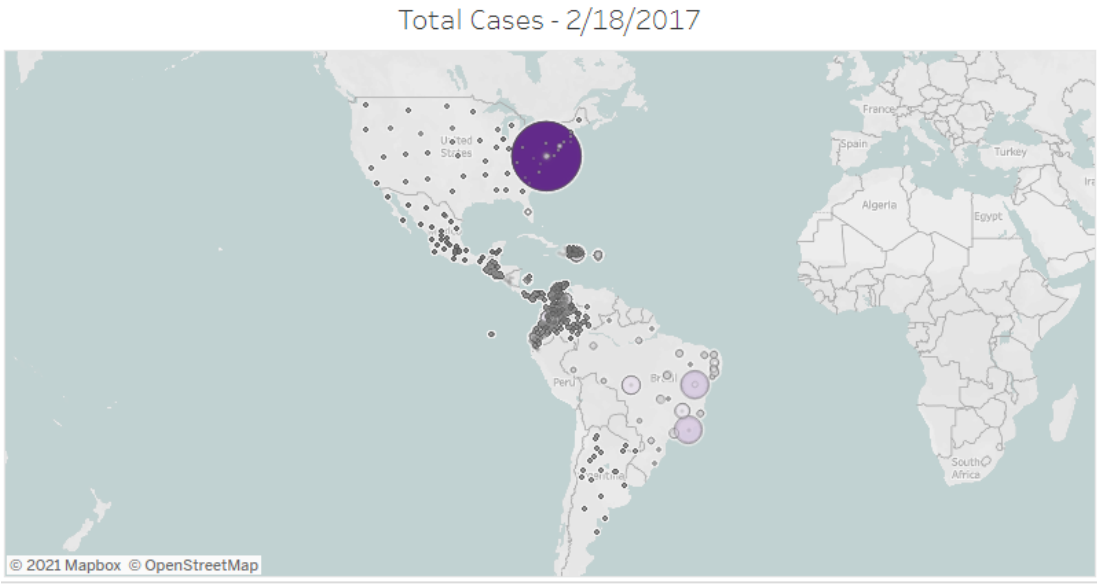
- For classification: 75345 rows X 25 columns
- For regression: 103045 rows X 25 columns

Exploratory Data Analysis

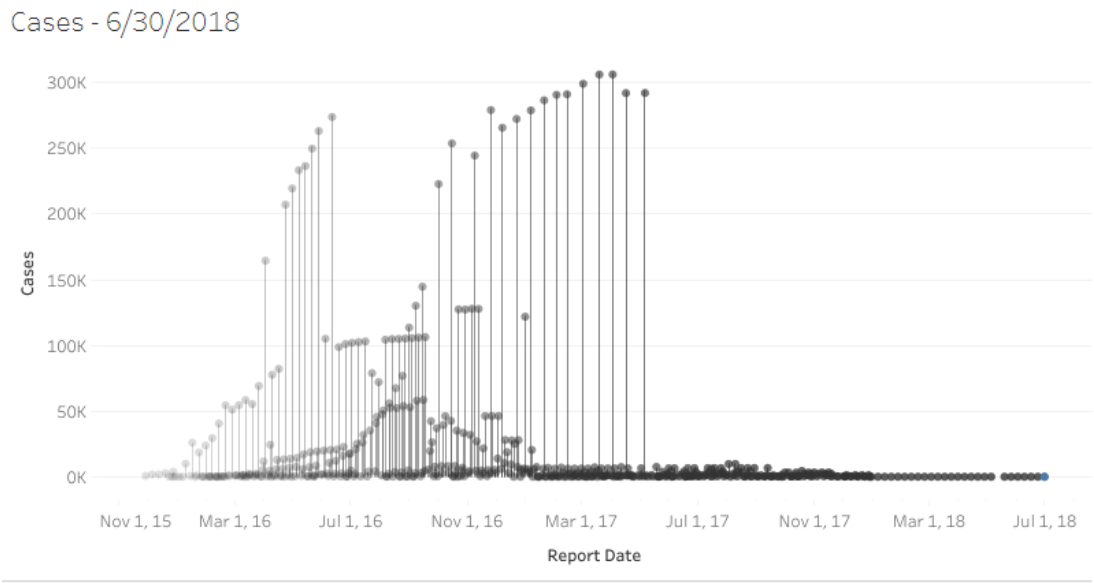
Zika virus dataset analysis

The zika virus dataset consists of following 14 countries:

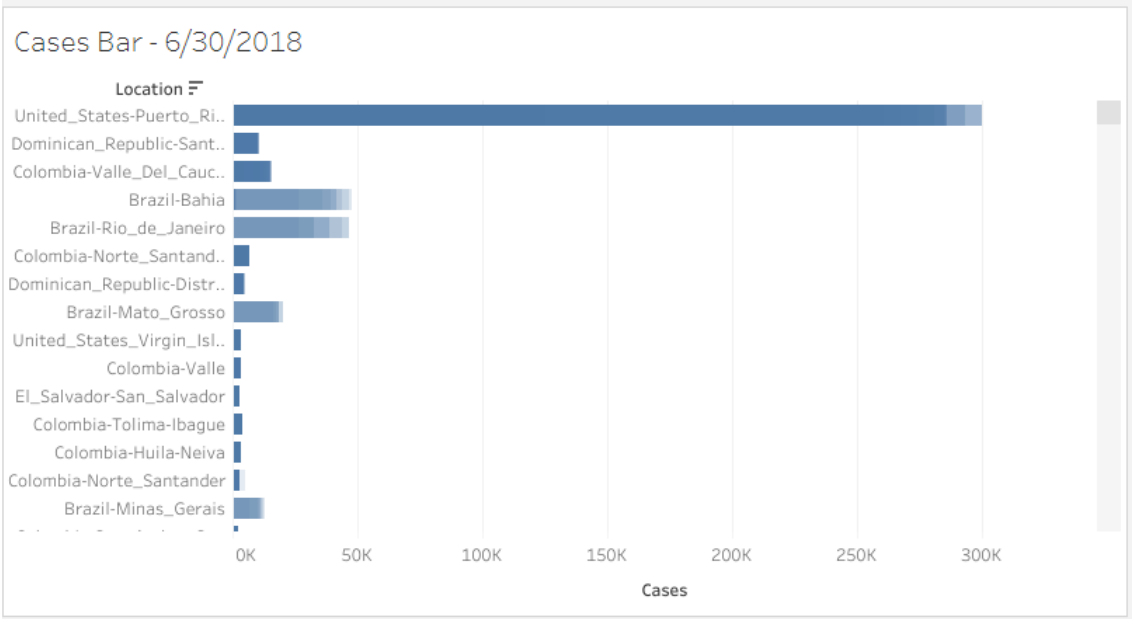
1. Argentina
2. United States
3. Brazil
4. Colombia
5. Dominican Republic
6. Ecuador
7. El Salvador
8. Guatemala
9. Haiti
10. Mexico
11. Nicaragua
12. Panama
13. Puerto Rico
14. U.S. Virgin Islands.



Zika virus cases in the 14 observed countries



Trend of Zika virus cases with time from November 2015 to July 2018



Regions with the most number of observed cases.

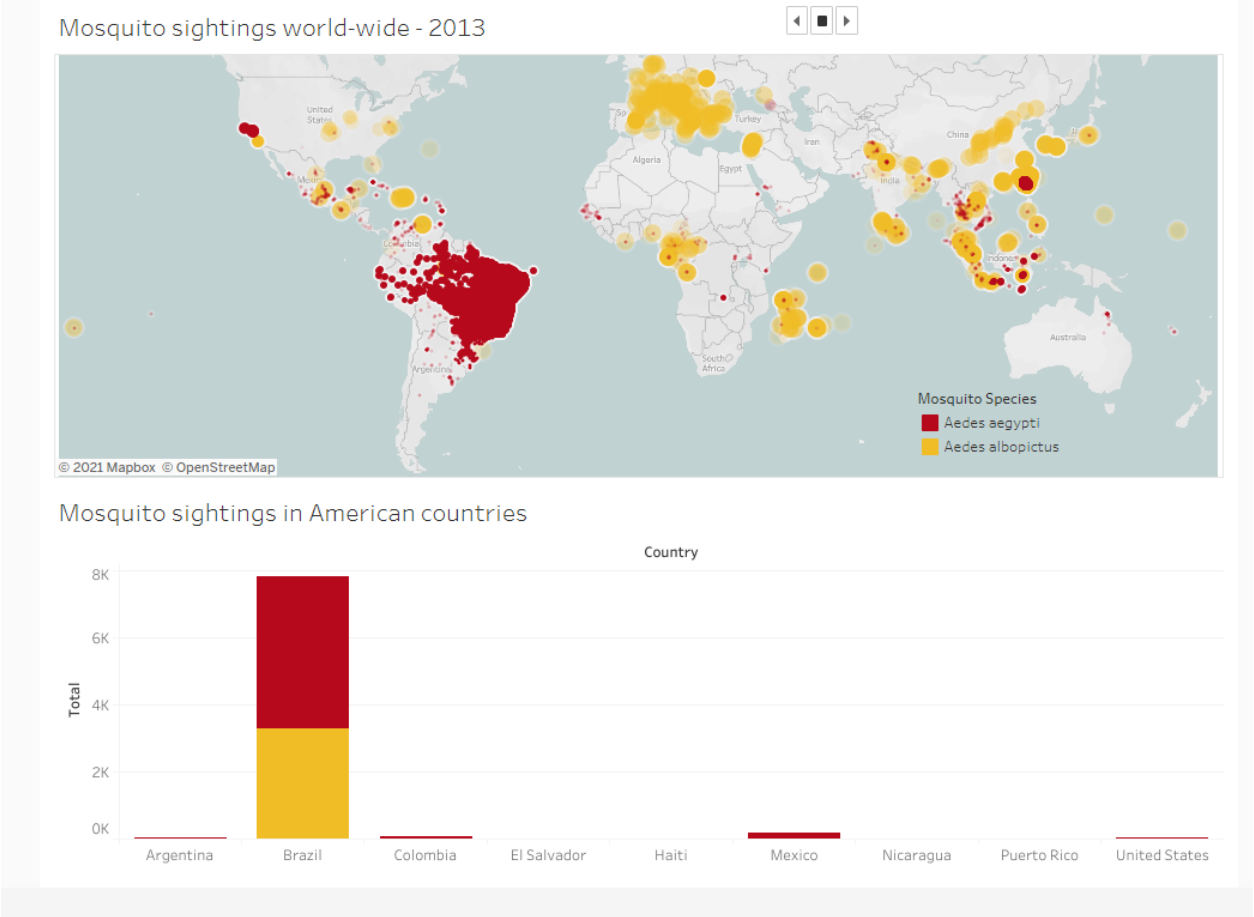
Dashboard-

https://public.tableau.com/app/profile/soham.shinde4264/viz/ZikaVirus_16327423649330/Dashboard1

Observations from Mosquito Dataset

Known facts:

- ➔ Zika Virus is a mosquito-borne disease generally caused by the Aedes vector family.
- ➔ Ae. aegypti is native to Africa while Ae. albopictus also known as asian tiger mosquito originates from Asia; both species now have a worldwide distribution.
- ➔ Ae. aegypti invaded in Americas back in 1600 during slave trade from Africa while Ae. albopictus was recently found out in Texas in 1985.



https://public.tableau.com/app/profile/soham.shinde4264/viz/ZikaVirus_16327423649330/Dashboard2

Observations:

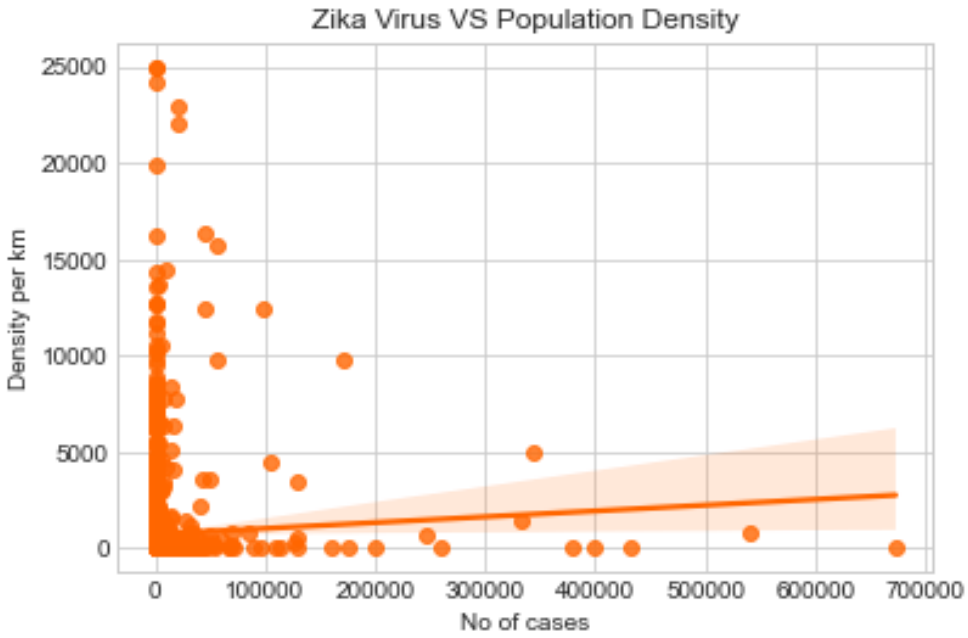
- Two main types of mosquito vectors are sighted throughout the world: Aedes aegypti and Aedes albopictus.
- Aedes aegypti has been observed more in American countries.
- Ae. aegypti mosquitoes are responsible for the spread of Zika virus in the countries under observation.

Effect of population density on zika virus cases:

Country	No. of cases	Density
Panamá Metro Las Garzas	9389432	1966.26
United States Puerto Rico	19	24970.13

Country with the most number of cases and the country with the least number of cases.

We can see that, for some regions, the number of cases are high and population density is low while for other regions, the number of cases are low and population density is high.



Scatter Plot to show the relationship between number of cases and population density.

We can see that there is a slight positive correlation between the two. But, as observed earlier in the table we can say that there is no certain relation between number of cases and population density of a certain area.

Effect of weather on Zika virus

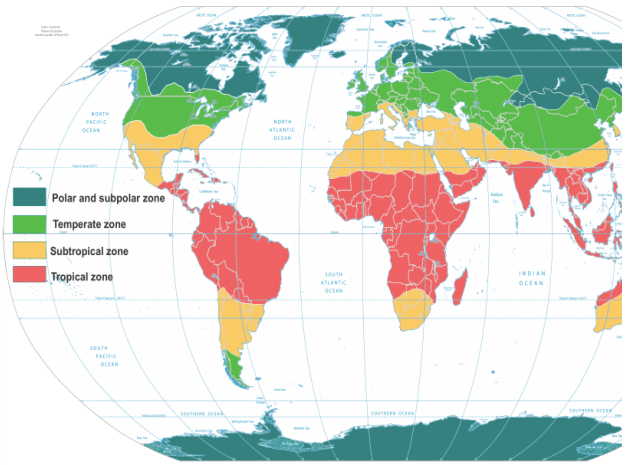
For our analysis we have divided our country list into two zones: Tropical and subtropical.

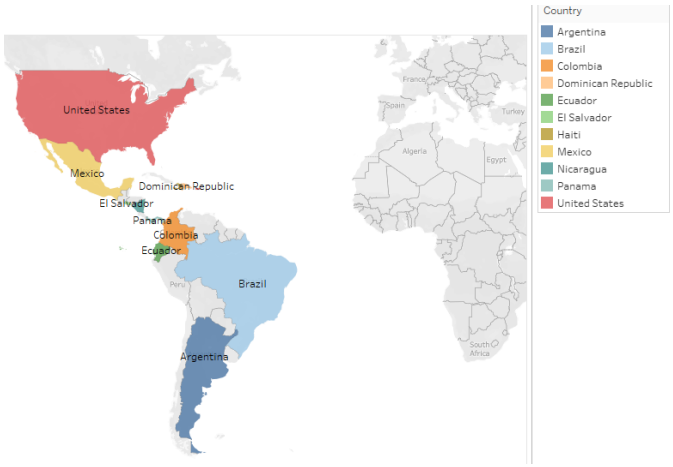
Tropical zone

- These are the regions where fixed months of any particular season are observed.
- Summer season temperatures average about 77 degrees
- Dry season temperatures average about 68 degrees
- Precipitation only falls during the summer months, usually from may-august with June and July having the heaviest rain
- Countries in the tropical zone: Brazil, Colombia, Dominican Republic, Ecuador, El Salvador, Guatemala, Haiti, Nicaragua, Panama.

Subtropical zone

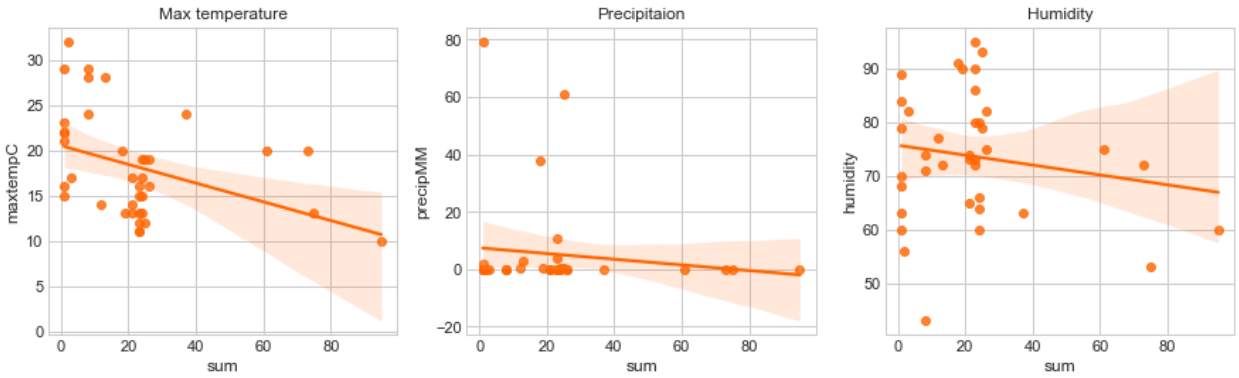
- These are the regions where there is no fixed period of a particular season.
- Humid summers and mild winters
- Summer, the average temperature is between 70 and 80 degrees
- Coldest month usually averages 45-50 degrees
- Rain falls throughout the year.
- There are 9/10 days with rainfall per month from October to April.
- Countries in the subtropical zone: United States, Argentina, Mexico.





As we know, the zika virus is a mosquito-borne disease. Mosquitoes are most active in humid, moisturized weather conditions i.e mostly in the rainy season. So, for our weather analysis we have considered three main factors that can help us understand the rain effect on zika virus: Precipitation, humidity and temperature in that area.

Weather in subtropical zone: Argentina

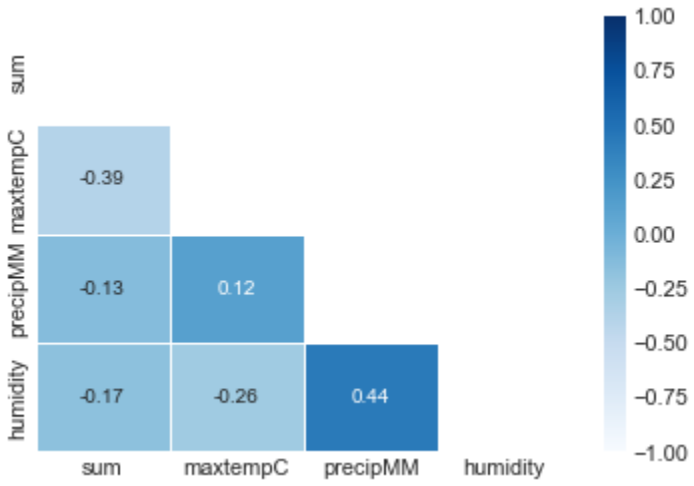


Scatterplots to show the relationship between the number of cases and temperature, precipitation, humidity in Argentina.

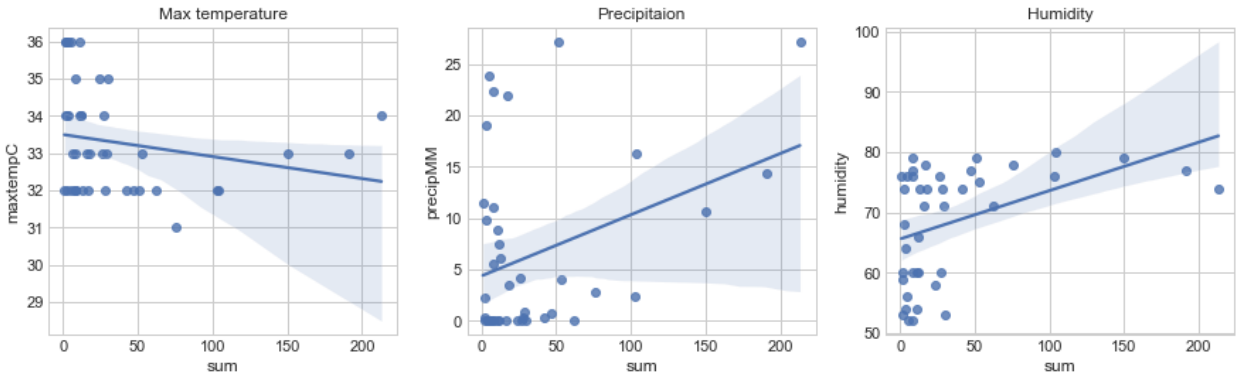
We can see that,

- In subtropical countries like Argentina there is negative correlation between the number of cases and weather.
- Thus we can say that, in subtropical countries, weather doesn't affect the number of cases much. Cases are affected when there are sudden changes in weather.

We can also see the correlativity from the following heatmap:



Weather in tropical countries: Nicaragua

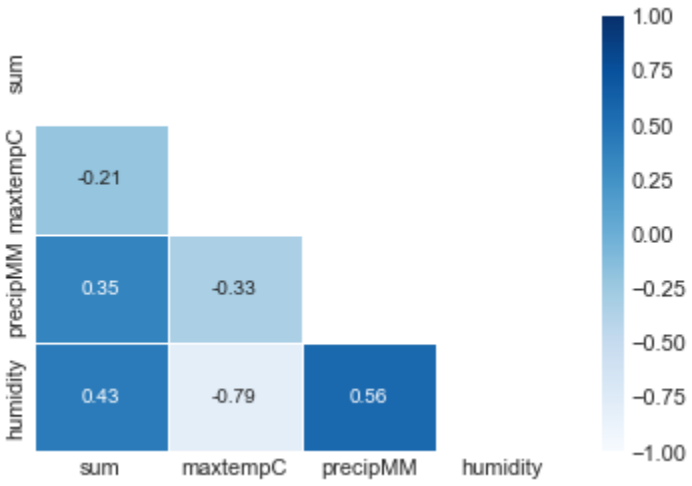


Scatterplots to show the relationship between the number of cases and temperature, precipitation, humidity in Nicaragua.

We can see that,

- In tropical countries like Nicaragua there is a positive correlation between the number of cases and weather. This is because of the rainy season which is observed in June-August.
- Thus we can say that, in tropical countries, cases are affected by the commencement of the rainy season.

We can also see the correlativity from the following heatmap:



Observations of weather with the incubation period of zika virus.

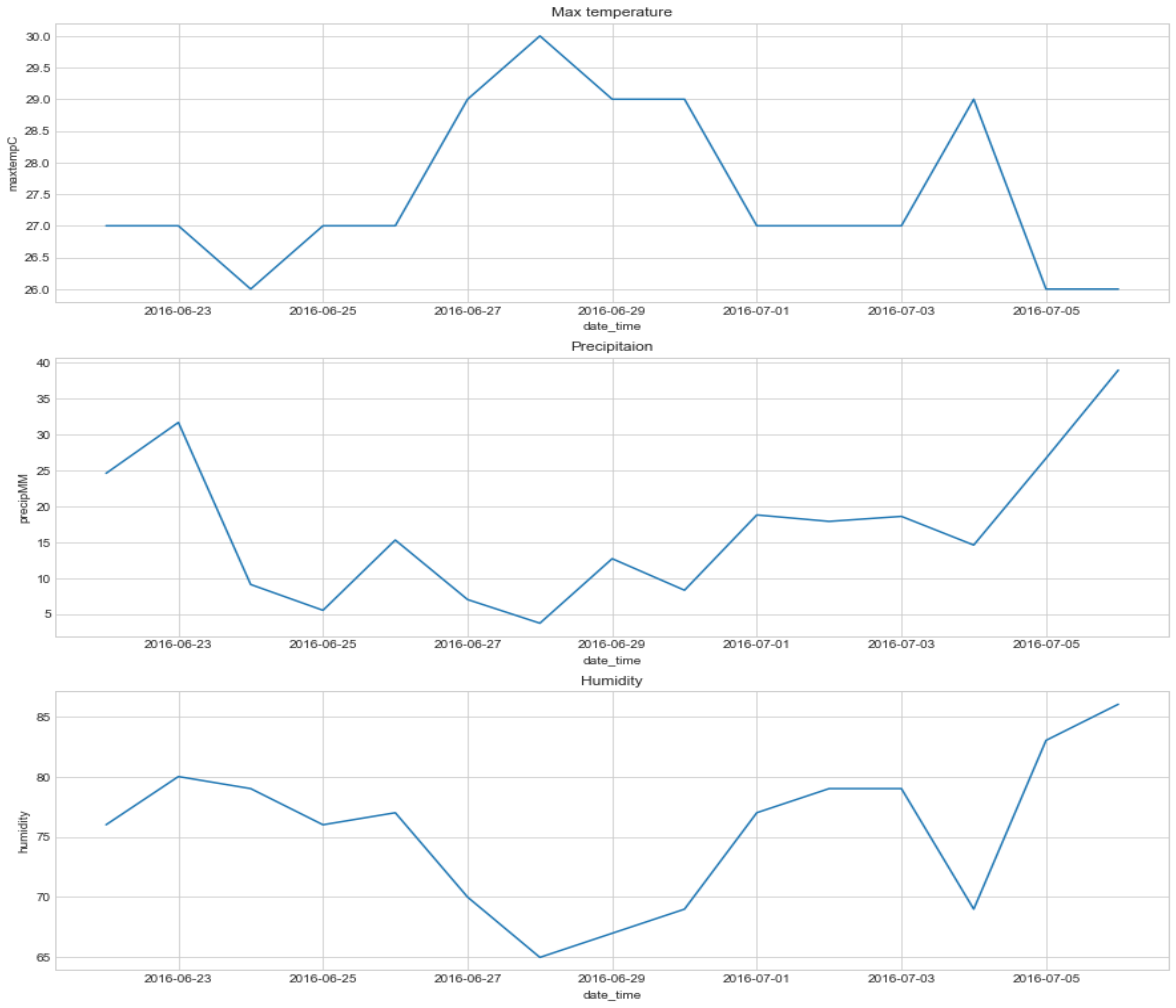
The incubation period of Zika virus is known to be 3-14 days. In our study we are considering the incubation period of 7 days. Hence, we are observing the weather conditions 7 days prior to when the case was reported in the state of Florida.

For this analysis, we have considered four case studies:

Case Study 1

When the number of cases on a single day is 0.

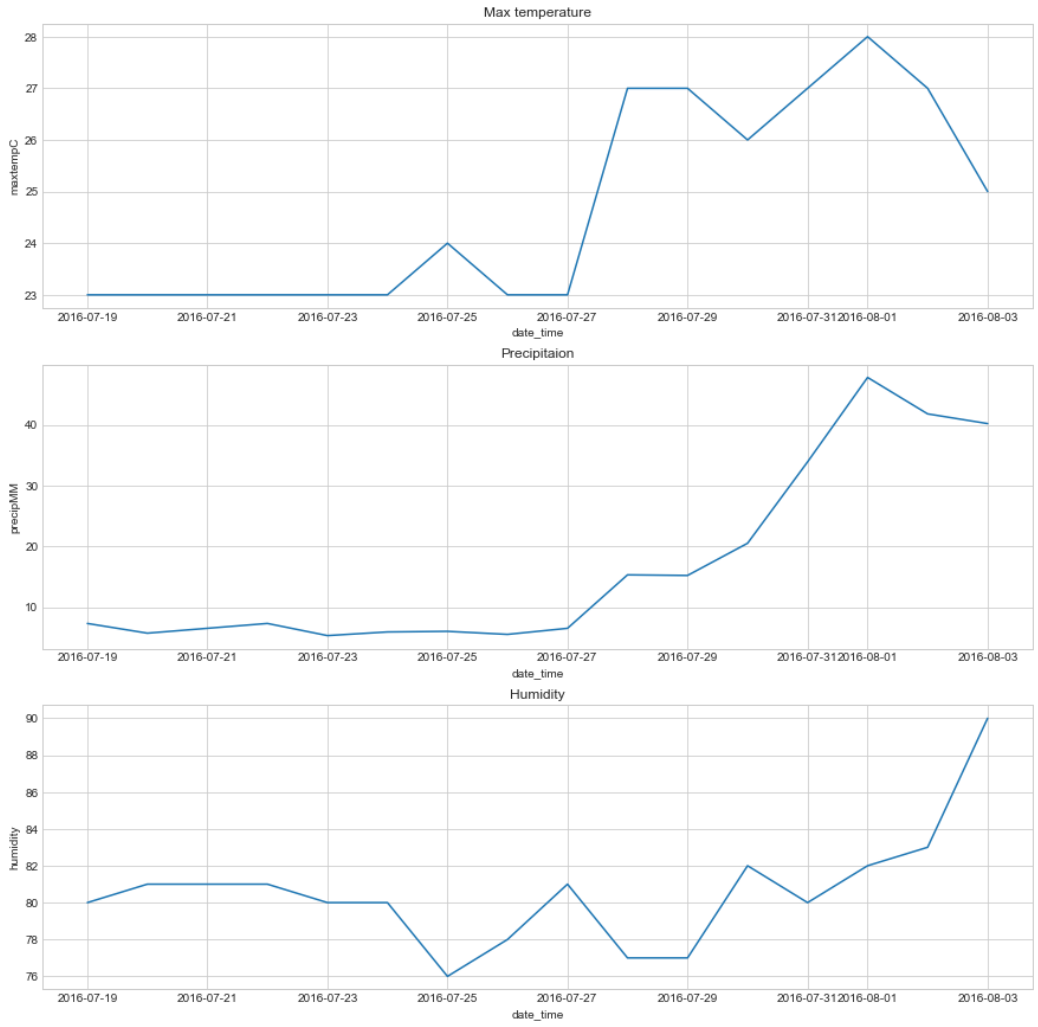
- Report date - 2016-07-06 = 0
- Weather date - 2016-06-22 to 2016-07-06



Case Study 2

When the number of cases are increasing.

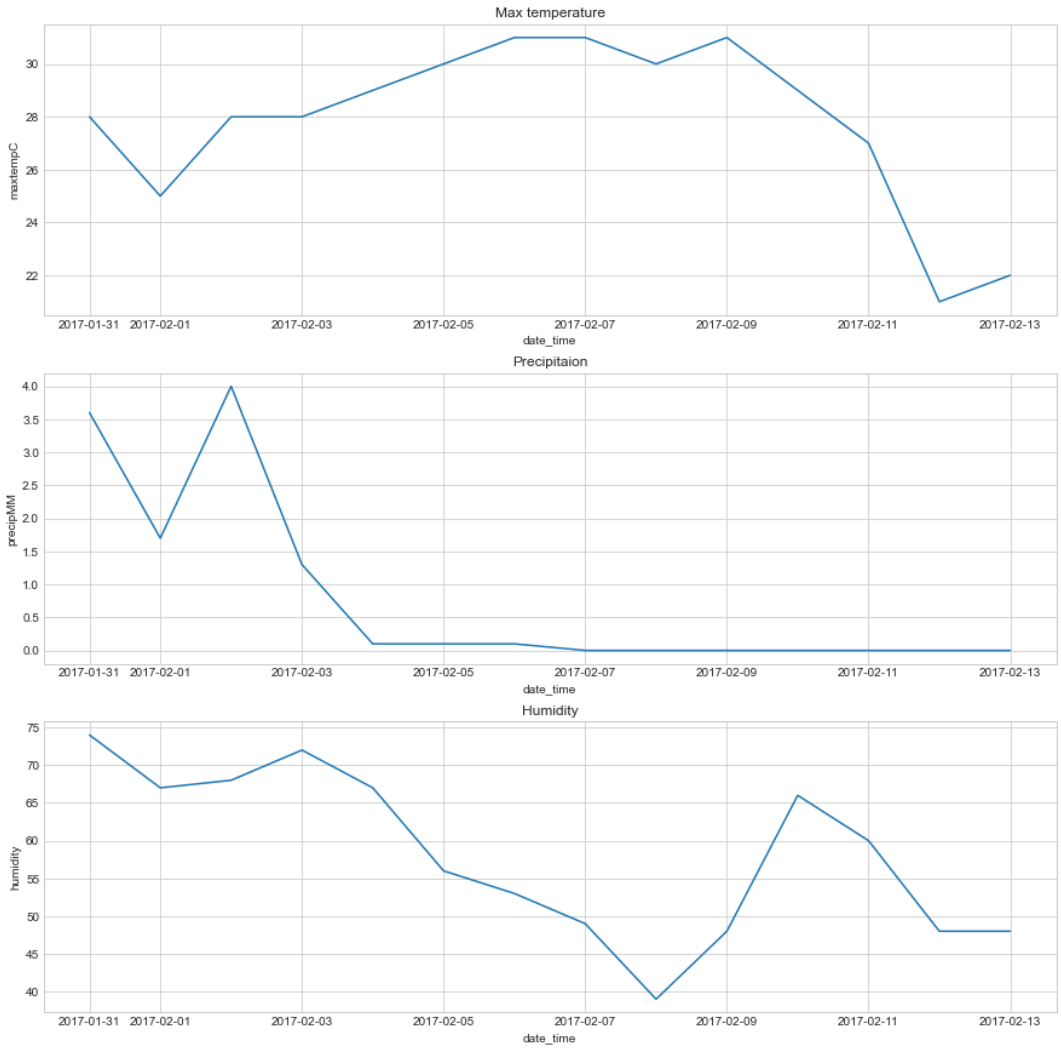
- Report date - 2016-08-03 = 6
- weather date - 2016-07-19 to 2016-08-03



Case Study 3

When the number of cases reported on a single day is 214.

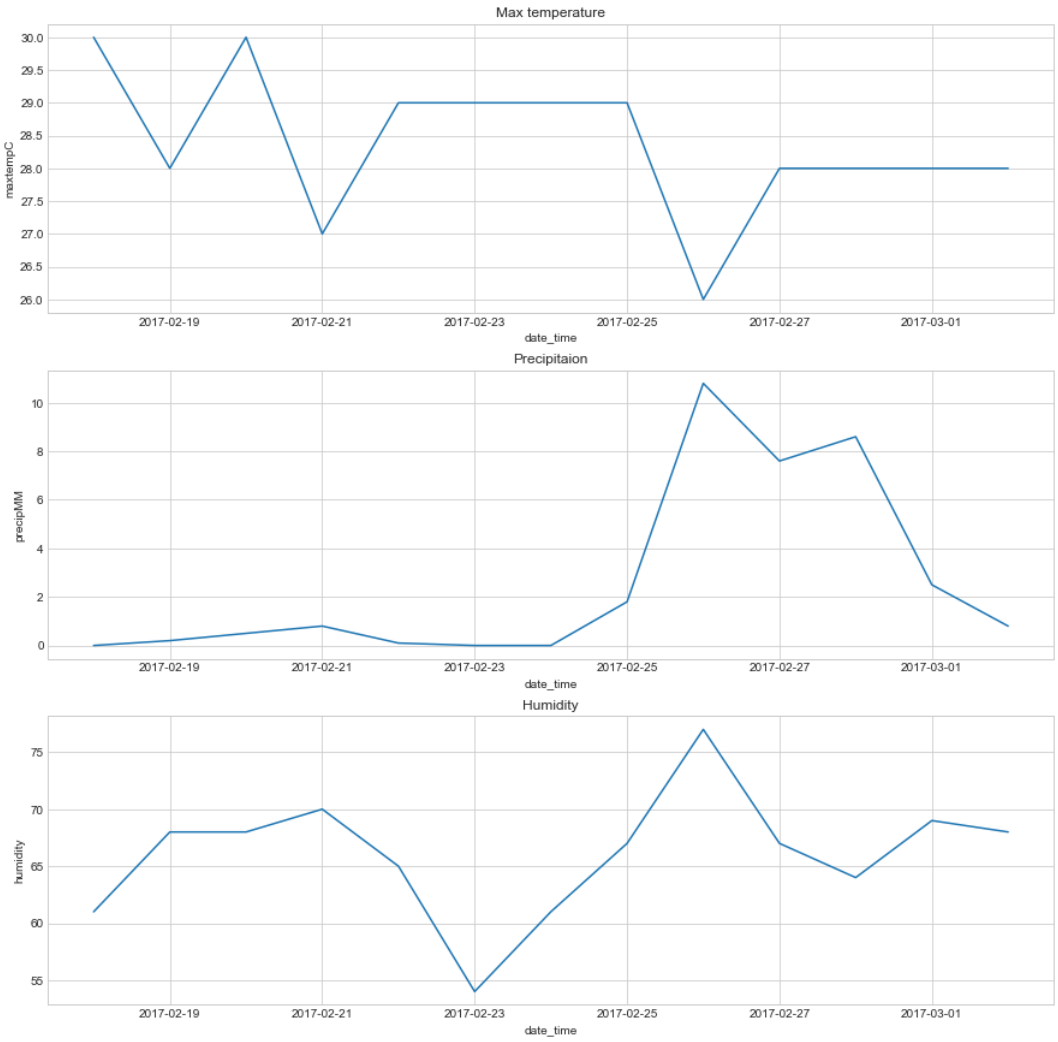
- Report date - 2017-02-13 = 214
- weather date - 2017-01-31 to 2017-02-13



Case Study 4

When the number of cases are decreasing.

- Report date - 2017-03-02 = 24
- weather date - 2017-02-18 to 2017-03-02



Complete observation of the number of cases and weather for the state of Florida.

From the incubation analysis we can observe that:

- ➔ When the number of cases are increasing, there is an increase in the precipitation and humidity.
- ➔ When the number of cases are decreasing, the precipitation is almost equal to zero and less humidity.
- ➔ Hence, for a sudden rise in the number of cases there was the presence of rain in its previous 7 days.
- ➔ Thus, we say that rain is responsible for the increase in zika virus cases.

Feature Selection

Feature selection is the process of reducing the number of input variables when developing a predictive model. It is desirable to reduce the number of input variables to both reduce the computational cost of modeling and, in some cases, to improve the performance of the model.

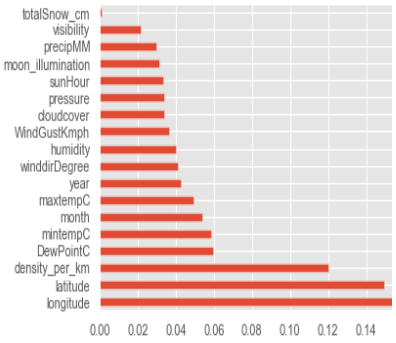
Feature Scaling:

We applied three feature scaling methods- Minmax scaler, Standard scaler and Robust scaler for minimizing the continuous features in our dataset viz. ‘density_per_km', 'precipMM’.

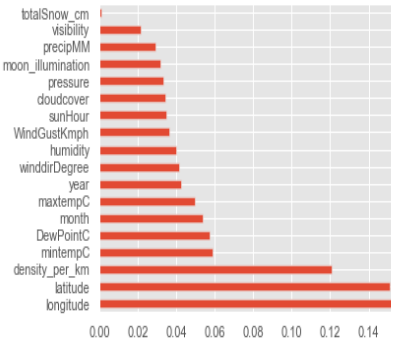
Methods used for feature selection:

Extra trees classifier

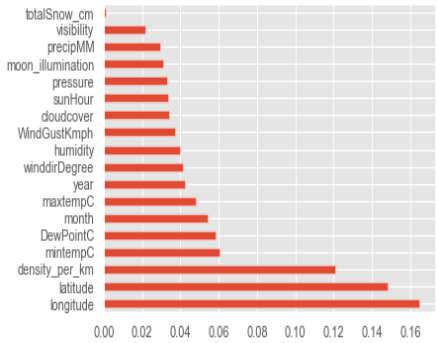
Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique which aggregates the results of multiple de-correlated decision trees collected in a “forest” to output it’s classification result.



Minmax scaler



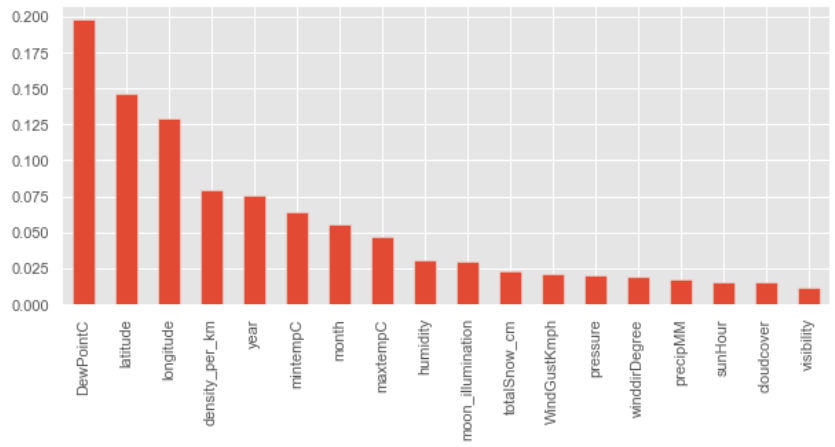
Robust scaler



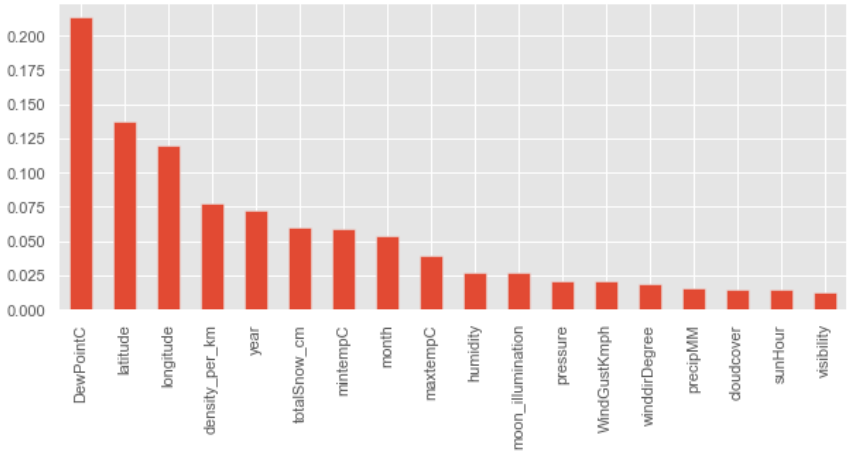
Standard scaler

XGBoost

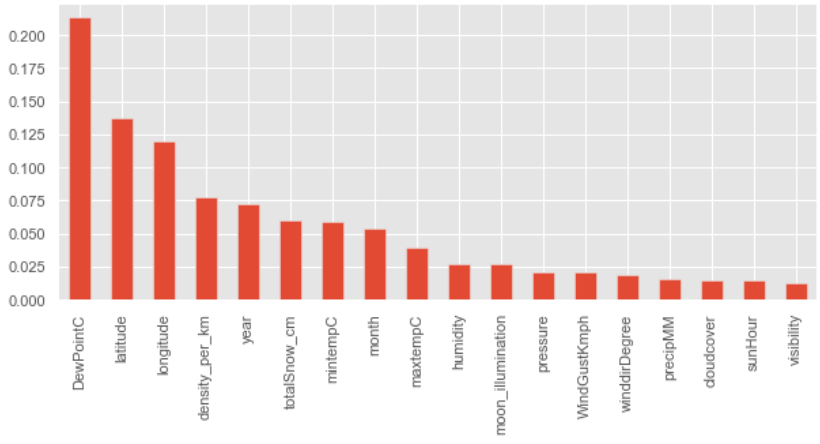
A benefit of using gradient boosting is that after the boosted trees are constructed, it is relatively straightforward to retrieve importance scores for each attribute. Generally, importance provides a score that indicates how useful or valuable each feature was in the construction of the boosted decision trees within the model. The more an attribute is used to make key decisions with decision trees, the higher its relative importance.



MinMax Scaler



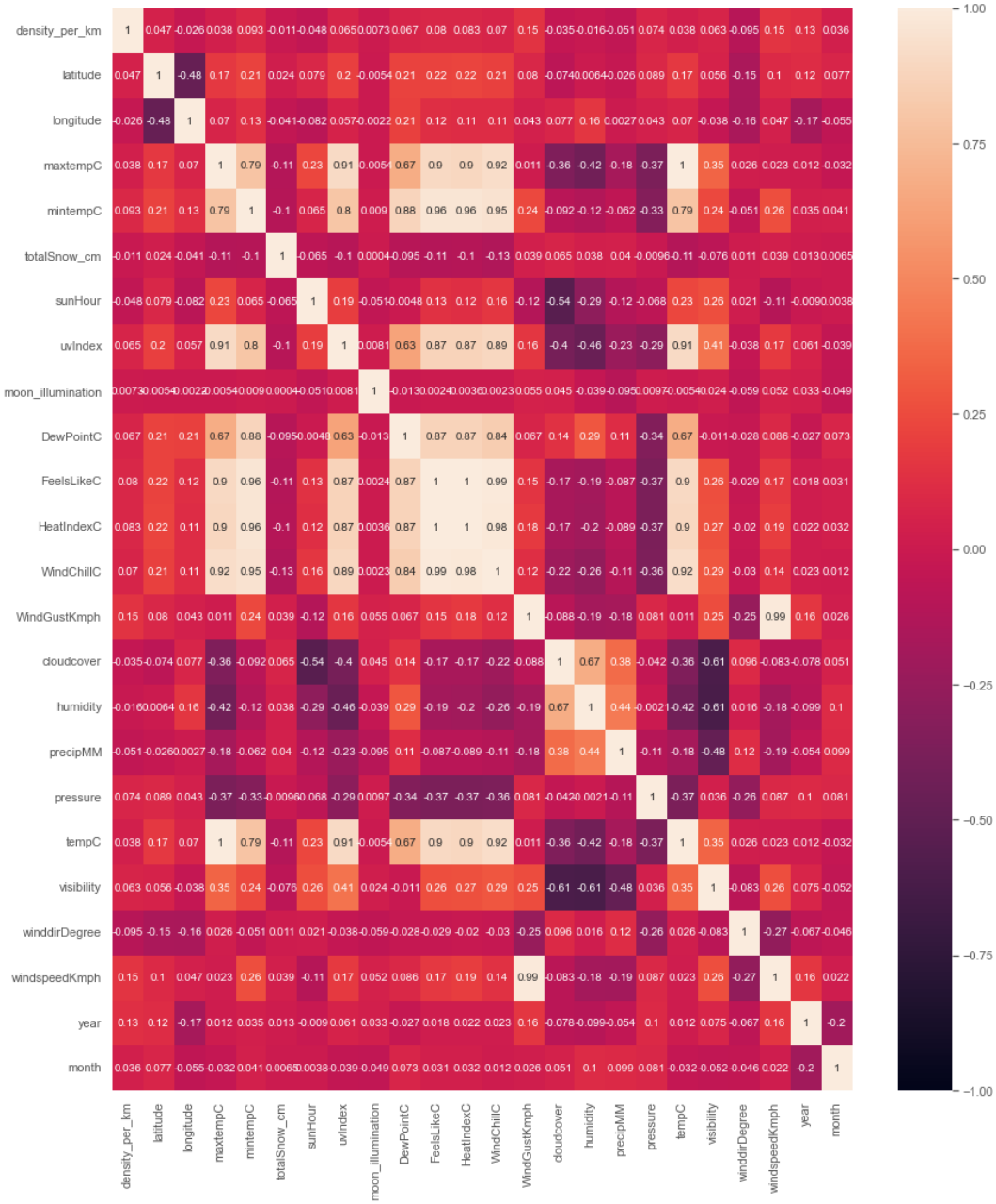
Robust scaler



Standard scaler

Since, both the methods gave different feature importance, we plotted a correlation matrix to remove those independent features which were highly correlated with each other.
We set the threshold value to 0.9

Feature selection based on correlation matrix



Heatmap consisting of all the independent variables

Final features selected for model building:

Independent Features: 'density_per_km', 'latitude', 'longitude', 'maxtempC', 'mintempC', 'totalSnow_cm', 'sunHour', 'moon_illumination', 'DewPointC', 'WindGustKmph', 'cloudcover', 'humidity', 'precipMM', 'pressure', 'visibility', 'winddirDegree', 'year', 'month'

Dependent Features : Target (0,1) – Classification
Cases - Regression

Model Building

For Classification

Following baseline models were trained for classification

AdaBoost

AdaBoost algorithm, short for Adaptive Boosting, is a Boosting technique used as an Ensemble Method in Machine Learning. It is called Adaptive Boosting as the weights are re-assigned to each instance, with higher weights assigned to incorrectly classified instances. Boosting is used to reduce bias as well as variance for supervised learning. It works on the principle of learners growing sequentially. Except for the first, each subsequent learner is grown from previously grown learners. In simple words, weak learners are converted into strong ones.

CatBoost

“CatBoost” name comes from two words “Category” and “Boosting”. CatBoost builds upon the theory of decision trees and gradient boosting. The main idea of boosting is to sequentially combine many weak models (a model performing slightly better than random chance) and thus through greedy search create a strong competitive predictive model. Because gradient boosting fits the decision trees sequentially, the fitted trees will learn from the mistakes of former trees and hence reduce the errors. This process of adding a new function to existing ones is continued until the selected loss function is no longer minimized.

XGBoost

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. Optimized Gradient Boosting algorithm through parallel processing, tree-pruning, handling missing values and regularization to avoid overfitting.

Random Forest

Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Bagging based algorithm where only a subset of features is selected at random to build a forest or collection of decision trees.

Decision Tree

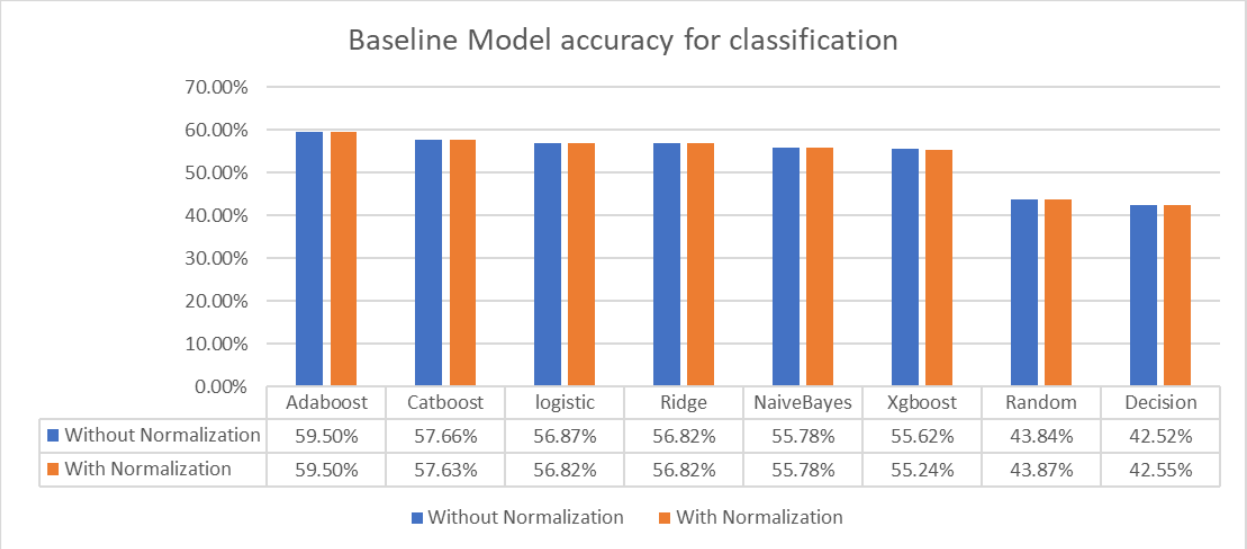
Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.

Logistic Regression

Logistic Regression is a Machine Learning algorithm which is used for classification problems, it is a predictive analysis algorithm and based on the concept of probability. We can call a Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the 'Sigmoid function' or also known as the 'logistic function' instead of a linear function. The hypothesis of logistic regression tends to limit the cost function between 0 and 1. Therefore linear functions fail to represent it as it can have a value greater than 1 or less than 0 which is not possible as per the hypothesis of logistic regression.

Naive Bayes

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. The Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.



	Adaboost	Catboost	logistic	Ridge	NaiveBayes	Xgboost	Random	Decision
Without Normalization	59.50%	57.66%	56.87%	56.82%	55.78%	55.62%	43.84%	42.52%
With Normalization	59.50%	57.63%	56.82%	56.82%	55.78%	55.24%	43.87%	42.55%

Performance evaluation of classification models

For Regression

Following baseline models were trained for regression

Linear Regression

Linear regression is a quiet and simple statistical regression method used for predictive analysis and shows the relationship between the continuous variables. Linear regression shows the linear relationship between the independent variable (X-axis) and the dependent variable (Y-axis), consequently called linear regression. If there is a single input variable (x), such linear regression is called simple linear regression. And if there is more than one input variable, such linear regression is called multiple linear regression. The linear regression model gives a sloped straight line describing the relationship within the variables.

Ridge Regression

Ridge regression is almost identical to linear regression (sum of squares) except we introduce a small amount of bias. In return, we get a significant drop in variance. In other words, by starting with a slightly worse fit, Ridge Regression can provide better long-term predictions.

Lasso Regression

Lasso Regression is a popular type of regularized linear regression that includes an L1 penalty. This has the effect of shrinking the coefficients for those input variables that do not contribute much to the prediction task. This penalty allows some coefficient values to go to the value of zero, allowing input variables to be effectively removed from the model, providing a type of automatic feature selection.

XGBoost Regressor

XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. Optimized Gradient Boosting algorithm through parallel processing, tree-pruning, handling missing values and regularization to avoid overfitting.

Random Forest Regressor

Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Bagging based algorithm where only a subset of features is selected at random to build a forest or collection of decision trees.

Decision Tree Regressor

Decision trees can be used for classification as well as regression problems. The name itself suggests that it uses a flowchart like a tree structure to show the predictions that result from a series of feature-based splits. It starts with a root node and ends with a decision made by leaves.

Model	Mean Squared Error	Coefficient of Determination (R^2 value)
Linear Regression	482209.82	0.00673
Lasso	482521.69	0.00624
Ridge	482442.85	0.00673
Decision Tree Regressor	902975.40	-0.859
Random Forest Regressor	669564.42	-0.37
XGBoost Regressor	512600.67	-0.005

Performance evaluation of regression models

Hyperparameter Tuning

For Classification models

AdaBoost

Number of trees effect on performance

- n_estimators=10 accuracy=0.587
- n_estimators=50 accuracy=0.595
- n_estimators=100 accuracy=0.597
- n_estimators=500 accuracy=0.597
- n_estimators=1000 accuracy=0.599
- n_estimators=5000 accuracy=0.601

Tree depth effect on performance

If max_depth = 1 to 10, then accuracy is =

- max_depth=1 accuracy=0.595
- max_depth=2 accuracy=0.600
- max_depth=3 accuracy=0.598
- max_depth=4 accuracy=0.583
- max_depth=5 accuracy=0.569
- max_depth=6 accuracy=0.545
- max_depth=7 accuracy=0.514
- max_depth=8 accuracy=0.479
- max_depth=9 accuracy=0.448
- max_depth=10 accuracy=0.435

learning rate effect on performance

learning rates from 0.1 to 2

- best accuracy is for learning rate 0.60.

Tuning using GridSearchCV – Cross validation with 10 splits

- grid['n_estimators'] = [10, 50, 100, 500]
- grid['learning_rate'] = [0.0001, 0.001, 0.01, 0.1, 1.0]

Best parameters for model:

base = DecisionTreeClassifier(max_depth=2)

learning_rate=0.1,n_estimators=500,base_estimator=base

	Accuracy
Before tuning (Default parameter)	59.88%
After tuning	60.27%
	+0.39%

Code:
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/Adaboostmodel.ipynb

CatBoost

Manual Tuning with 5 CV

- max_depth =3
- max_depth =5

Grid Search with 5 CV

Define parameters 1

parameters = {'depth': [4,5,6,7,8,9, 10],

'learning_rate': [0.01,0.02,0.03,0.04],

'iterations': [10, 20,30,40,50,60,70,80,90, 100]}

Define parameters 2

parameters = {'max_depth': [3,4,5],'n_estimators': [100, 200, 300]}

Define parameters 3

parameters = {'learning_rate': [0.03, 0.1],
'depth': [4, 6, 10],

'l2_leaf_reg': [1, 3, 5],
'iterations': [50, 100, 150]}

Best parameters for model:

{'depth': 3, 'iterations': 1000, 'l2_leaf_reg': 10,'learning_rate': 0.02}

	Accuracy
Before tuning (Default parameter)	57.70%
After tuning	60.40%
	+2.7%

Code:
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/CatBoostModel.ipynb

Logistic Regression

Tuning using GridSearchCV – Cross validation with 10 splits

Define parameters

- solvers = ['newton-cg', 'lbfgs', 'liblinear']
- penalty = ['l1', 'l2']
- c_values = [100, 10, 1.0, 0.1, 0.01]

Best parameters: {'C': 1.0, 'penalty': 'l2', 'solver': 'liblinear'}

	Accuracy
Before tuning (Default parameter)	56.81%
After tuning	56.83%
	+0.02%

Code:
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/LogisticModel_Normalize.ipynb

Ridge Classifier

Tuning using GridSearchCV – Cross validation with 10 splits

Define parameters

- alpha = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0]

Best parameter: {'alpha': 0.1}

	Accuracy
Before tuning (Default parameter)	56.97%
After tuning	57.09%
	+0.12%

Code:
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/RidgeClassifierModel_Normalize.ipynb

Naive Bayes

Tuning using GridSearchCV – Cross validation with 10 splits

Define parameters

- {'var_smoothing': np.logspace(0,-9, num=100)}

Calculation Stability to Widen (or Smooth) the Curve

Best parameter: {'var_smoothing': 0.43287612810830584}

	Accuracy
Before tuning (Default parameter)	55.78%
After tuning	57.08%
	+1.3%

Code:
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/NaiveBayesModel_Normalize.ipynb

XGBoost

Tuning using GridSearchCV – Cross validation with 5 splits

Define parameters

- "learning_rate" : [0.05, 0.10, 0.15, 0.20, 0.25, 0.30]
- "max_depth" : [3, 4, 5, 6, 8, 10, 12, 15]
- "min_child_weight" : [1, 3, 5, 7]
- "gamma" : [0.0, 0.1, 0.2 , 0.3, 0.4]
- "colsample_bytree" : [0.3, 0.4, 0.5 , 0.7]

Best parameter for Model: {'colsample_bytree': 0.7,
'gamma': 0.2,
'learning_rate': 0.3,
'max_depth': 3,
'min_child_weight': 5}

	Accuracy
Before tuning (Default parameter)	55.24%
After tuning	60.20%
	4.96+%

Code:
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/XGBoostModel.ipynb

Decision Tree

Tuning using GridSearchCV – Cross validation with 10 splits

Define parameters

"splitter":["best","random"],
'criterion':['entropy','gini']
"max_depth": [1,3,5,7,9,11,12]
"min_samples_leaf":[1,2,3,4,5,6,7,8,9,10]
'min_samples_split': [2, 5, 10, 14]
"max_features":["auto","log2","sqrt",None]

Best parameters for model:{'criterion': 'gini', 'max_depth': 5, 'max_features': None, 'min_samples_leaf': 10, 'min_samples_split': 5, 'splitter': 'best'}

	Accuracy
Before tuning (Default parameter)	42.10%
After tuning	59.28%
	+17.18%

Code: https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/DecisionTreeModel.ipynb

Random Forest

Randomized Search CV

Define parameters

- 'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]
- 'max_features': ['auto', 'sqrt', 'log2'],
- 'max_depth': [10, 120, 230, 340, 450, 560, 670, 780, 890, 1000]
- 'min_samples_split': [2, 5, 10, 14]
- 'min_samples_leaf': [1, 2, 4, 6, 8]
- 'criterion': ['entropy', 'gini']

Best: {'criterion': 'entropy',
'max_depth': 10,
'max_features': 'auto',
'min_samples_leaf': 2,
'min_samples_split': 14,
'n_estimators': 2000}

GridSearchCV – Cross validation with 10 splits

Define parameters

- n_estimators = [10, 100, 500]
- max_features = ['sqrt', 'log2']
- criterion = ['entropy','gini']

Best: 0.452936 using {'criterion': 'entropy',

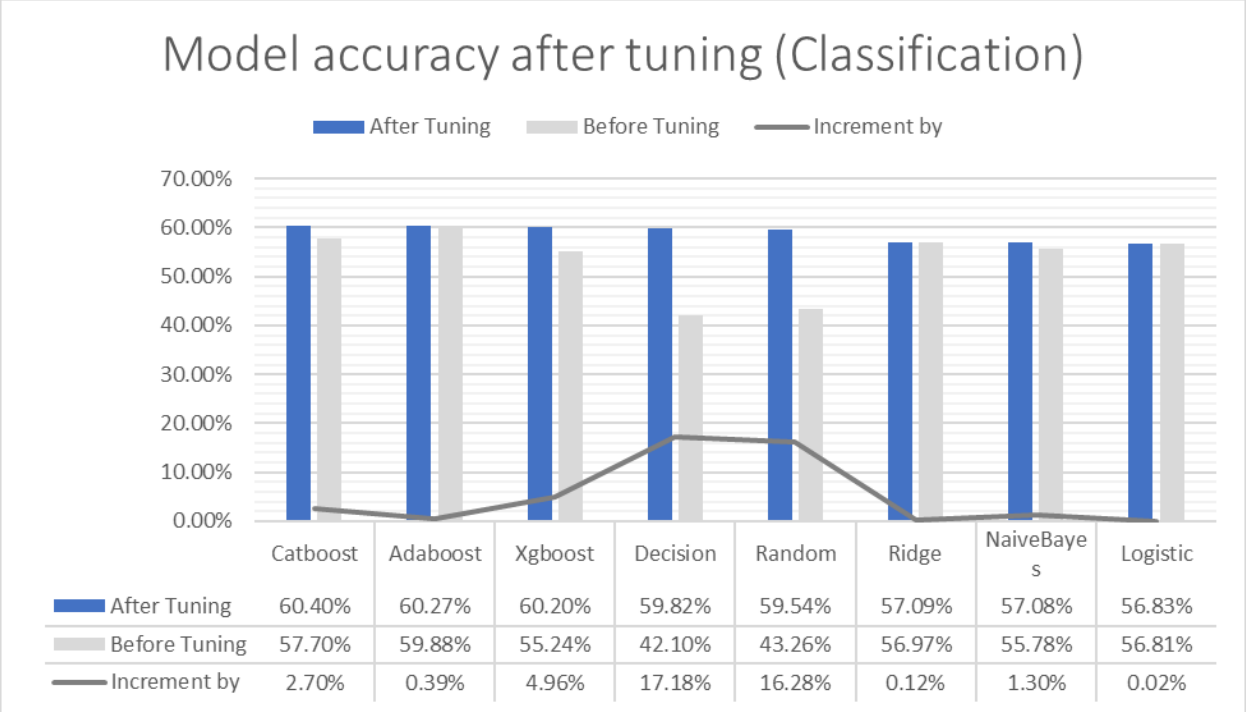
'max_features': 'log2',
'n_estimators': 500}

Best parameter for model:

{'criterion': 'entropy', 'max_features': 'sqrt', max_depth=8, 'n_estimators': 500}

	Accuracy
Before tuning (Default parameter)	43.26%
After tuning	59.54%
	+16.28%

Code:
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/RandomForestModel.ipynb



	Catboost	Adaboost	Xgboost	Decision	Random	Ridge	NaiveBayes	Logistic
After Tuning	60.40%	60.27%	60.20%	59.82%	59.54%	57.09%	57.08%	56.83%
Before Tuning	57.70%	59.88%	55.24%	42.10%	43.26%	56.97%	55.78%	56.81%
Increment by	2.70%	0.39%	4.96%	17.18%	16.28%	0.12%	1.30%	0.02%

For Regression models:

Lasso Regression

Tuning using GridSearchCV – Cross validation with 10 splits

Define parameters
{'alpha': (np.logspace(-8, 8, 100))} *# It will check from 1e-08 to 1e+08*

Best parameter for Model: 'alpha': 0.013848863713938746

	R2 score
Before tuning (Default parameter)	0.00624
After tuning	0.0067
	7.46+%

Ridge Regression

Tuning using GridSearchCV – Cross validation with 10 splits

Define parameters
{'alpha': (np.logspace(-8, 8, 100))} *# It will check from 1e-08 to 1e+08*

Best parameter for Model: 'alpha': 0.27185882427329455

	R2 score
Before tuning (Default parameter)	0.0067
After tuning	0.0066
	1.49+%

Code-
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/Regression%20Models/LinLasRidge.ipynb

Random Forest

GridSearchCV – Cross validation with 3 splits

Define parameters

```
parameters = {'bootstrap': [True, False],
 'max_depth': [50, 60, 70, 80, 90, 100],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
'min_samples_split': [2, 5, 10],
'n_estimators': [200, 400, 600, 800, 1000]}
```

Best parameter for model:

```
 max_depth=300, max_features='sqrt',
 min_samples_leaf=16, min_samples_split=30,
n_estimators=3000, random_state=10
```

	R2 score
Before tuning (Default parameter)	-0.37
After tuning	0.09
	+124.32%

Code-
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/Regression%20Models/RandomForestRegressor.ipynb

Decision Tree

Tuning using GridSearchCV – Cross validation with 10 splits

Define parameters

```
{"splitter":["best","random"],
  "max_depth" : [1,3,5,7,9,11,12],
  "min_samples_leaf":[1,2,3,4,5,6,7,8,9,10],
  "min_weight_fraction_leaf":[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9],
  "max_features":["auto","log2","sqrt",None],
   "max_leaf_nodes":[None,10,20,30,40,50,60,70,80,90] }
```

Best parameters for model:

```
{'max_depth': 9,
 'max_features': 'log2',
 'max_leaf_nodes': 50,
 'min_samples_leaf': 10,
 'min_weight_fraction_leaf': 0.1,
 'splitter': 'best'}
```

	R2 score
Before tuning (Default parameter)	-0.85
After tuning	0.0060
	+101.02%

Code-
https://github.com/TeamEpicProjects/Epidemic-Outbreak-Prediction-using-Artificial-Intelligence/blob/Day-17/Regression%20Models/DecisionTreeRegressor.ipynb

Model	Mean Squared Error	R^2 value	% increase
Lasso	546999.53	0.0067	7.46
Ridge	514096.20	0.0066	1.49
Decision Tree Regressor	481223.73	0.0060	101.02
Random Forest Regressor	441754.63	0.090	124.32

Performance evaluation after tuning

AutoML

Automated Machine Learning (AutoML) refers to techniques for automatically discovering well-performing models for predictive modeling tasks with very little user involvement. We implemented three methods of AutoML for getting the best model for prediction.

Auto-Sklearn

Auto-Sklearn is an open-source library for performing AutoML in Python. It makes use of the popular Scikit-Learn machine learning library for data transforms and machine learning algorithms and uses a Bayesian Optimization search procedure to efficiently discover a top-performing model pipeline for a given dataset.

The benefit of Auto-Sklearn is that, in addition to discovering the data preparation and model that performs for a dataset, it also is able to learn from models that performed well on similar datasets and is able to automatically create an ensemble of top-performing models discovered as part of the optimization process.

Following are the results after implementation of Auto-Sklearn:

	Classification	**Regression**
Best Model	Gradient Boost	Random Forest
Performance	60% accuracy	0.07 R^2 value

Autogluon

Autogluon is an open-source python library that automates the whole process of machine learning and helps in achieving high accuracy. It automatically trains and predicts the models in just a single line of code. It works on different types of datasets i.e. Tabular, Image, Text, etc.

AutoGluon-Tabular currently supports the following algorithms and trains all of them if no time

limit is imposed:

- Random Forests
- Extremely Randomized trees
- k-nearest neighbors
- LightGBM boosted trees
- CatBoost boosted trees
- AutoGluon-Tabular deep neural networks

Following are the results after implementation of AutoGluon

	Classification	**Regression**
Best Model	Weighted Ensemble	Light Gradient Boost
Performance	83% accuracy	-0.27 R^2 value

Tree-based Pipeline Optimization Tool (TPOT)

TPOT is an open-source library for performing AutoML in Python. It makes use of the popular Scikit-Learn machine learning library for data transforms and machine learning algorithms and uses a Genetic Programming stochastic global search procedure to efficiently discover a top-performing model pipeline for a given dataset.

TPOT uses a tree-based structure to represent a model pipeline for a predictive modeling problem, including data preparation and modeling algorithms and model hyperparameters. An optimization procedure is then performed to find a tree structure that performs best for a given dataset. Specifically, a genetic programming algorithm, designed to perform a stochastic global optimization on programs represented as trees.

Following are the results after implementation of TPOT:

	Classification
Best Model	Gradient Boost
Performance	60% accuracy

Model Selection

After building the classification and regression models, we selected the best model based on the AutoML methods and further hypertuned it.

For Classification

AutoML method: TPOT

Best Model: XGBoost Classifier

Hyperparameter Tuning:

Best Parameters: (base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
importance_type='gain', interaction_constraints='',
learning_rate=0.1, max_delta_step=0, max_depth=2,
min_child_weight=5, missing=nan, monotone_constraints='()',
n_estimators=100, n_jobs=1, num_parallel_tree=1, random_state=10,
reg_alpha=0, reg_lambda=1, scale_pos_weight=1,
subsample=0.9500000000000001, tree_method='exact',
validate_parameters=1, verbosity=0)

Performance Evaluation: Accuracy = 60.373%

For Regression:

AutoML method: Auto Sklearn

Best Model: Random Forest Regressor

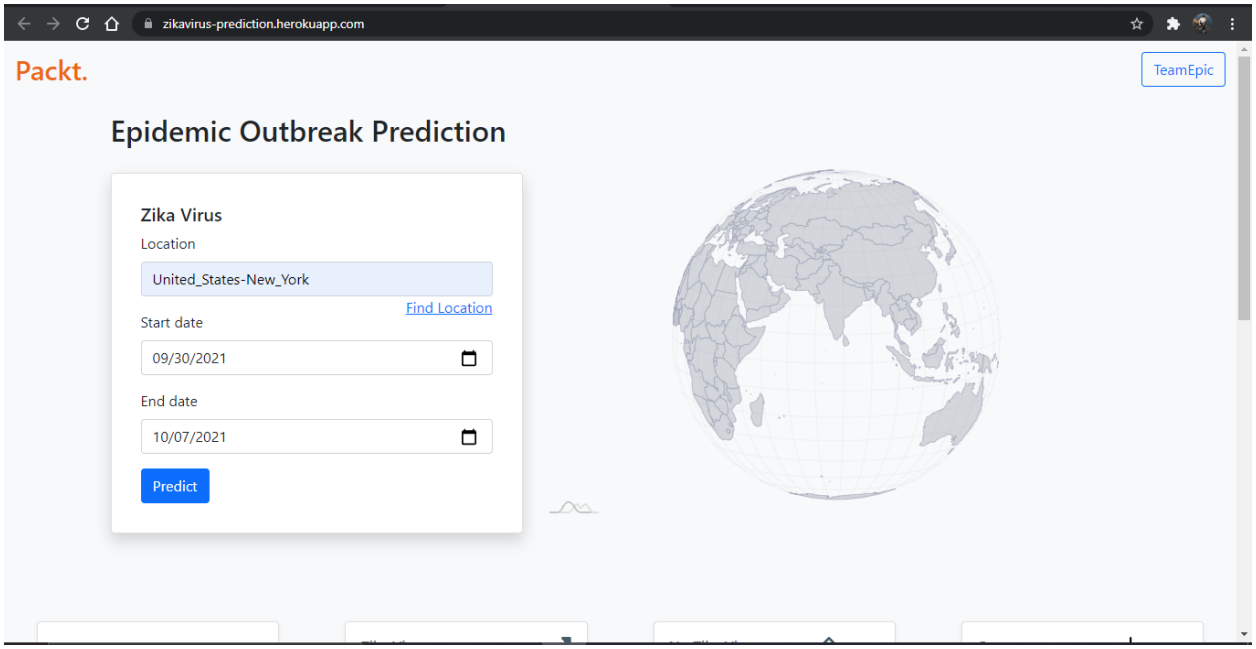
Hyperparameter Tuning:

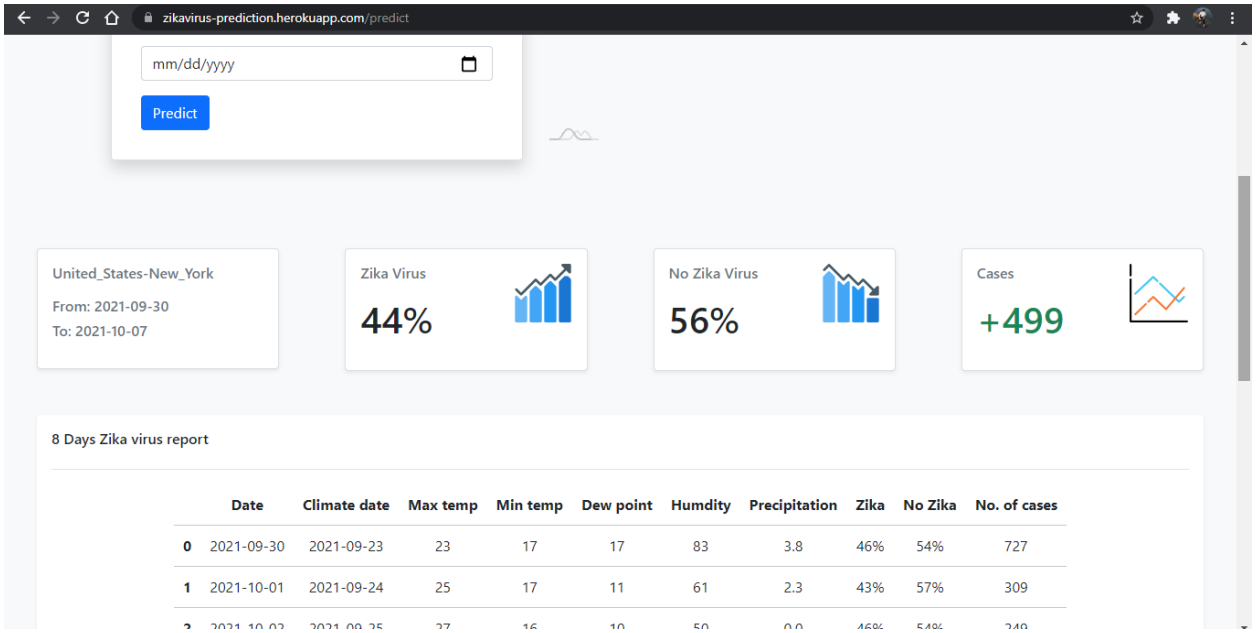
Best Parameters: max_depth=300, max_features='sqrt',
min_samples_leaf=16, min_samples_split=30,
n_estimators=3000, random_state=10

Performance Evaluation:
Mean Squared Error = 441754.63
Coefficient of determination (R2 score) = 0.09

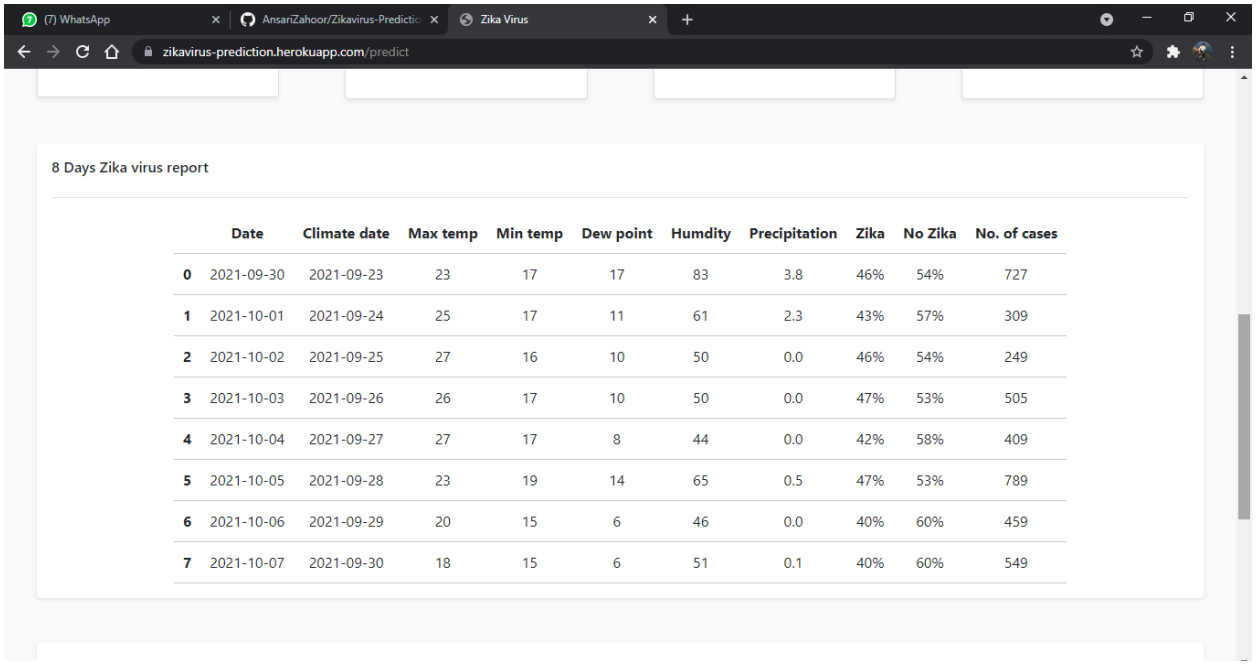
Model Deployment

- In frontend we have created a website with the help of html, CSS and bootstrap framework.
- Bootstrap is a powerful toolkit - a collection of HTML, CSS, and JavaScript tools for creating and building web pages and web applications.
- In backend: Flask API was used to deploy the machine learning model in backend. It is used to manage HTTP requests and uses API functions to get the data and display the result to the end user in the front end UI.
- Users have to enter country name and start date to end date for prediction on a webpage, model will predict the likelihood of an outbreak with number of cases.

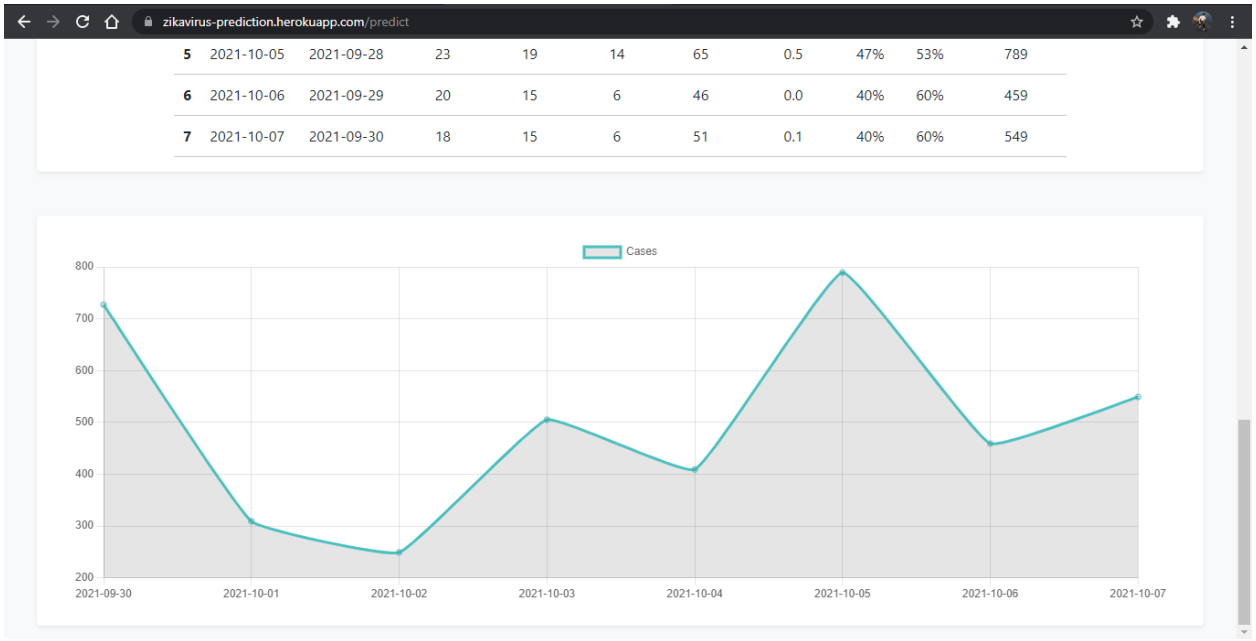




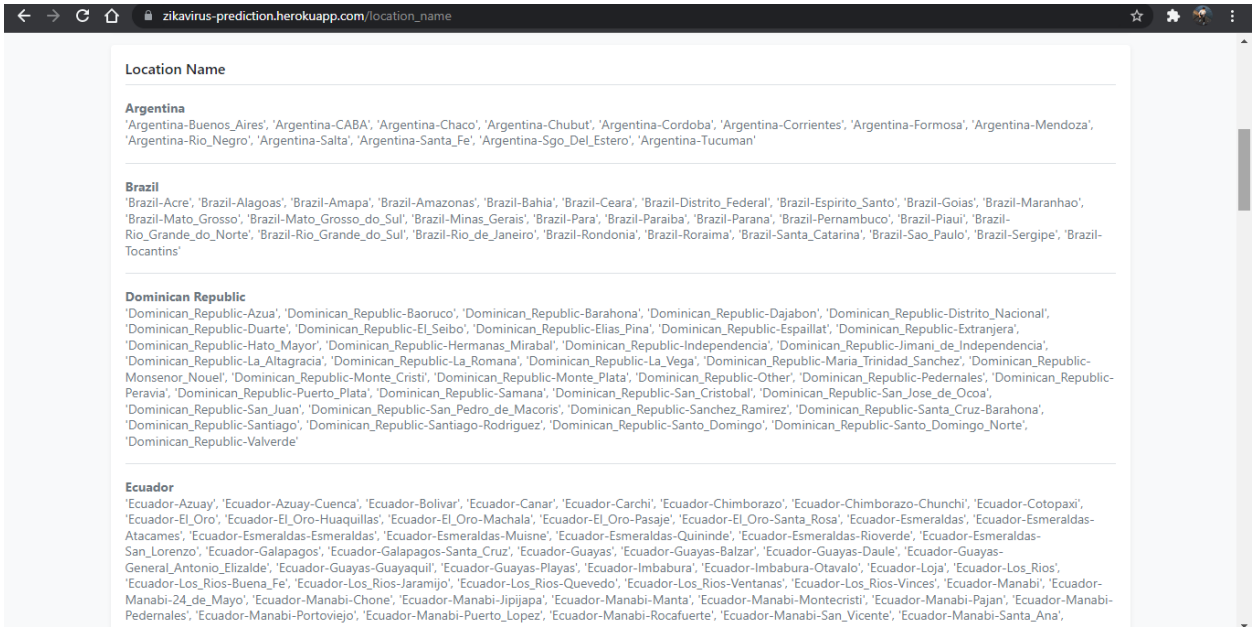
	Date	Climate date	Max temp	Min temp	Dew point	Humdity	Precipitation	Zika	No Zika	No. of cases
0	2021-09-30	2021-09-23	23	17	17	83	3.8	46%	54%	727
1	2021-10-01	2021-09-24	25	17	11	61	2.3	43%	57%	309



	Date	Climate date	Max temp	Min temp	Dew point	Humdity	Precipitation	Zika	No Zika	No. of cases
0	2021-09-30	2021-09-23	23	17	17	83	3.8	46%	54%	727
1	2021-10-01	2021-09-24	25	17	11	61	2.3	43%	57%	309
2	2021-10-02	2021-09-25	27	16	10	50	0.0	46%	54%	249
3	2021-10-03	2021-09-26	26	17	10	50	0.0	47%	53%	505
4	2021-10-04	2021-09-27	27	17	8	44	0.0	42%	58%	409
5	2021-10-05	2021-09-28	23	19	14	65	0.5	47%	53%	789
6	2021-10-06	2021-09-29	20	15	6	46	0.0	40%	60%	459
7	2021-10-07	2021-09-30	18	15	6	51	0.1	40%	60%	549


zikavirus-prediction.herokuapp.com/predict
5 2021-10-05 2021-09-28 23 19 14 65 0.5 47% 53% 789
6 2021-10-06 2021-09-29 20 15 6 46 0.0 40% 60% 459
7 2021-10-07 2021-09-30 18 15 6 51 0.1 40% 60% 549
Cases
800
700
600
500
400
300
200
2021-09-30
2021-10-01
2021-10-02
2021-10-03
2021-10-04
2021-10-05
2021-10-06
2021-10-07

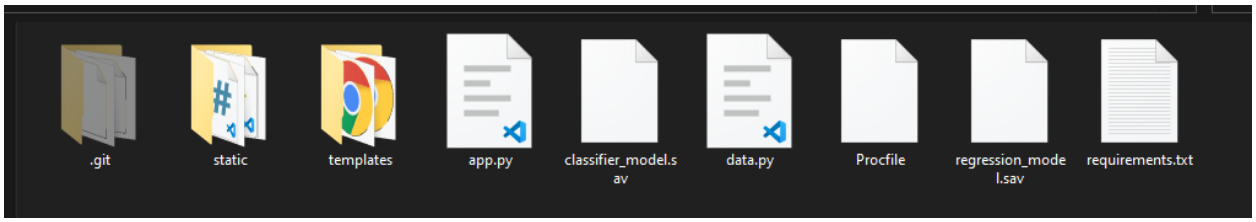

zikavirus-prediction.herokuapp.com/location_name
Packt.
TeamEpic
Back



Website Deployment

Heroku is a container-based cloud Platform as a Service (PaaS). Developers use Heroku to deploy, manage, and scale modern apps. Our platform is elegant, flexible, and easy to use, offering developers the simplest path to getting their apps to market.

Heroku is fully managed, giving developers the freedom to focus on their core product without the distraction of maintaining servers, hardware, or infrastructure. The Heroku experience provides services, tools, workflows, and polyglot support—all designed to enhance developer productivity.



Website link: https://zikavirus-prediction.herokuapp.com/

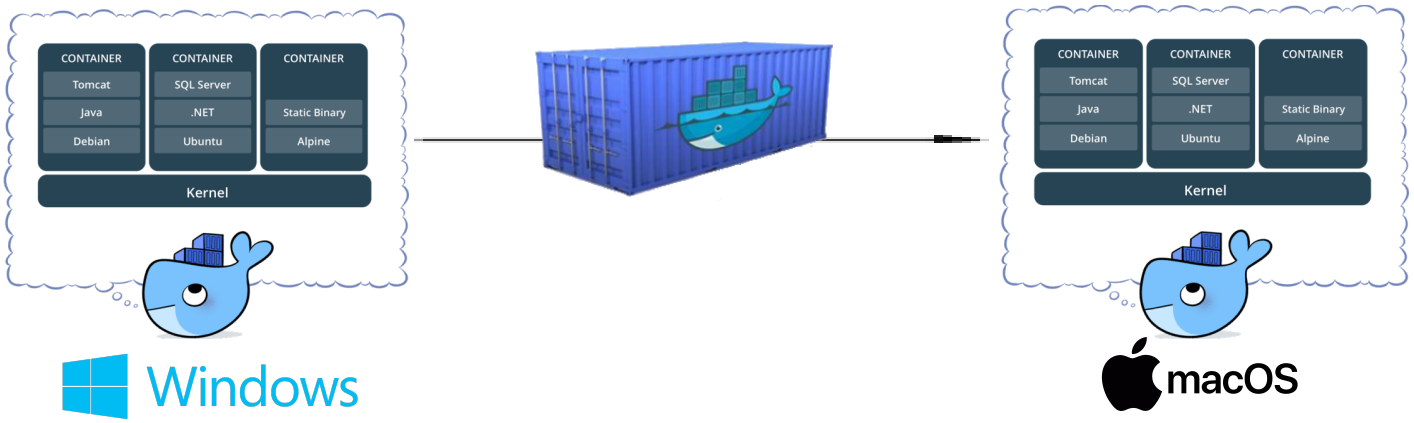
Docker

What is the problem statement that docker is trying to solve?

- Whenever a developer develops any product there are certain issues which probably occur almost every time, well that problem is whenever you are designing a project it works absolutely fine in your machine the developer machine.
- But as soon as that project is being moved on to the production state maybe on server or somebody's else computer in that work with the same level of working, when that project is moved one place to another place
- Docker is designed to specifically address this exact problem.

What is Docker?

- Docker is an open-source platform for building, deploying, and managing containerized applications.
- Docker is compatible with almost any programming language or any project.
- It allows you to have absolutely sealed air tight containers. These containers wrap up the entire code and these containers are absolutely portable.
- It also allows us to have social containers.



- Docker File creates a Docker Image using build command
- A Docker Image contains all the project's code
- Using docker image, any user can run the code in order to create docker containers
- Once a Docker Image is built, It's uploaded in a registry or a Docker Hub

Conclusion

- We were able to load the dataset using Airflow and perform relevant analysis and data manipulation for deriving further insights.
- Exploratory Data Analysis was performed in order to obtain visual information on how weather and density was impacting the number of cases.
- We also looked at which mosquito species was responsible for the cause of zika virus.
- Dashboard creation for visualizing the insights and zika virus trend in these countries.
- Post EDA, we implemented dimensionality reduction techniques and did the feature selection.
- Various classification and regression algorithms were implemented with hyper parameter tuning to understand which model performed well.
- AutoML methods such as AutoSklearn, AutoGluon, and TPOT were implemented to check the best model automatically.
- UI development using HTML, FLASK API was done in order to create a suitable and interactive UI for prediction.
- We further dockerized the whole application for making it suitable for different operating systems.

Future Scope

- We can further extend this project to predict the zika virus cases worldwide.
- With the availability of weather forecasting data we can predict the possibility of an epidemic for more days.
- Given that our proof of concept worked well, we can augment it further by adding new parameters like social media symptomatic data, lifestyle, population dynamics, etc.
- We can study and predict various other epidemics as well using the same approach.
- We can additionally request government and healthcare institutions for more data that's not currently available in the public domain.
- A crucial but often ignored factor in ML models is lack of sufficient structured data. Particularly, in the healthcare sector, the plethora of data that is available is often 'unstructured' or plain text data and not amenable to ML.