DATA SCIENCE PRACTICE

Collaborative Development of Data Explorer Web App

Liam Huang | Jason Nguyen | Darren Li | Nick Drage

Data Explorer Tool Overview

The Data Explorer Tool is a web-based application that allows a user to generate some preliminary EDA (Exploratory Data Analysis) from a given dataset with minimum effort. It is hosted in a docker container.

Once the container is running, the app can be accessed by a web browser via the link http://localhost:8501/

Github Repository

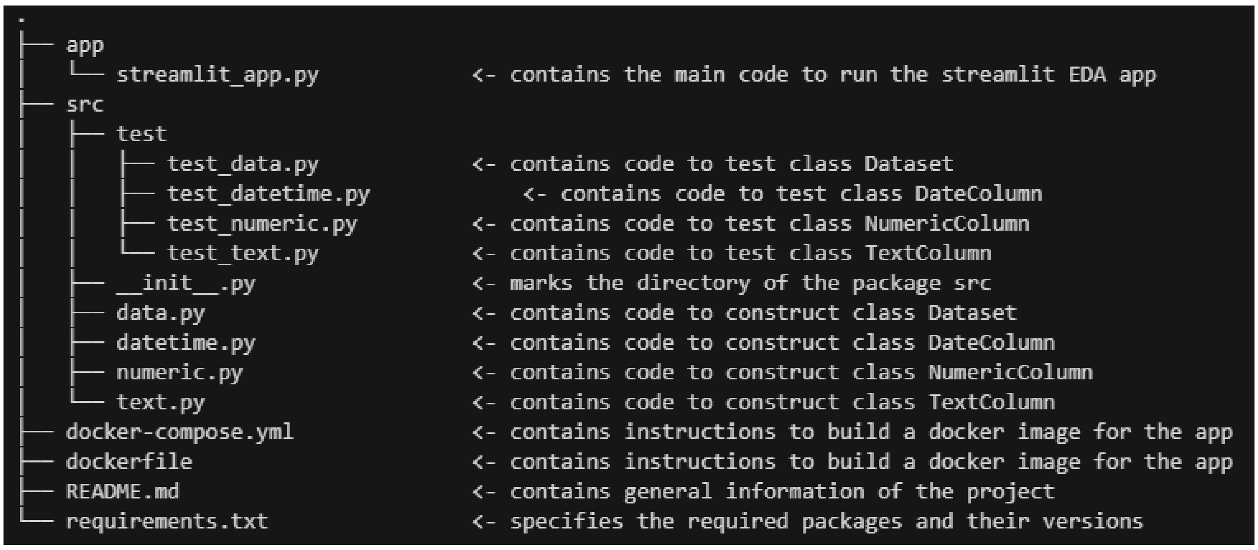
https://github.com/jaseuts/dsp-assignment3

The Design Of The Application

The app requires a docker container for hosting. The Docker container is built with the following architecture:

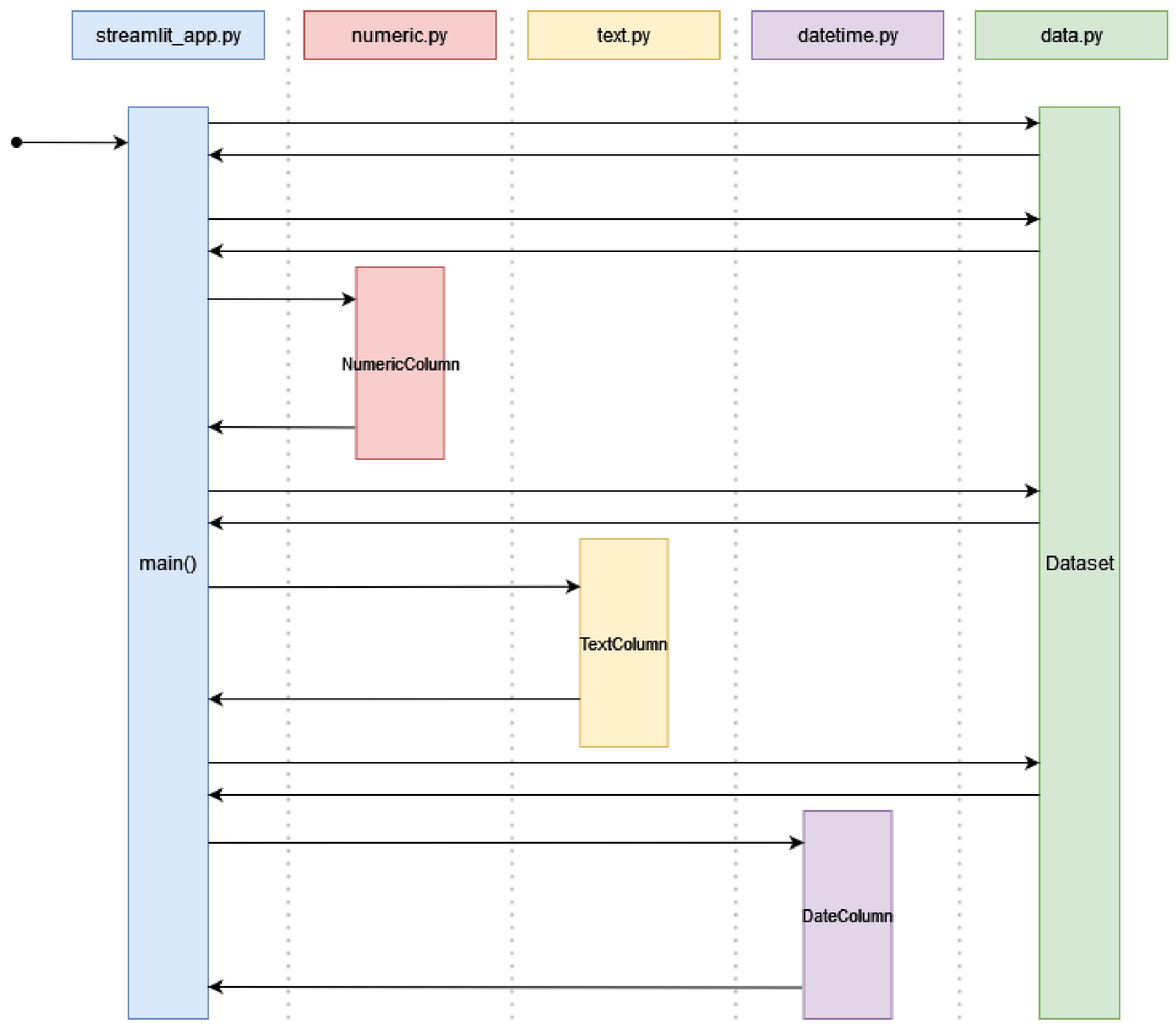
- A Docker image (python:3.8.2-slim-buster)
- Streamlit 0.89.0
- Pandas 1.3.3

The app's folder has the following structure



```
.
├── app
│   └── streamlit_app.py          <- contains the main code to run the streamlit EDA app
├── src
│   ├── test
│   │   ├── test_data.py          <- contains code to test class Dataset
│   │   ├── test_datetime.py          <- contains code to test class DateColumn
│   │   ├── test_numeric.py       <- contains code to test class NumericColumn
│   │   └── test_text.py          <- contains code to test class TextColumn
│   ├── __init__.py               <- marks the directory of the package src
│   ├── data.py                   <- contains code to construct class Dataset
│   ├── datetime.py               <- contains code to construct class DateColumn
│   ├── numeric.py                <- contains code to construct class NumericColumn
│   └── text.py                   <- contains code to construct class TextColumn
├── docker-compose.yml            <- contains instructions to build a docker image for the app
├── dockerfile                    <- contains instructions to build a docker image for the app
├── README.md                     <- contains general information of the project
└── requirements.txt              <- specifies the required packages and their versions
```

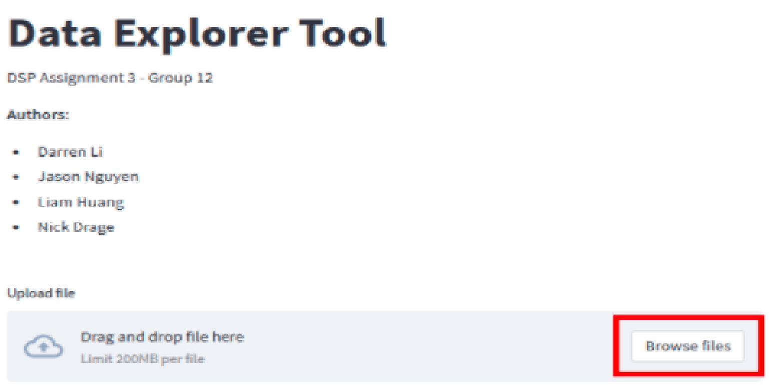
The function flow between the script and modules is depicted as below



Installation & Launch Instructions

1. Ensure that Docker is installed
2. Once Docker is installed, download and extract the repository as found here.
3. Using the command line navigate to the appropriate directory. If done correctly, you should be able to access Dockerfile.
4. Run ‘docker build -t streamlit_assignment03:latest .’ (remember the period at the end!). This should create a docker image named *streamlit_assigment_03* which will be used as the basis of the container.
5. Next run ‘docker run -dit --rm --name eda_app -p 8501:8501 -v "${PWD}":/app streamlit_assignment03:latest’. This should create a container named *eda_app* from the image created in step 2.
6. If no errors appear, the the app should be accessible by a web browser via the link http://localhost:8501/

Guide to Use



Data Explorer Tool

DSP Assignment 3 - Group 12

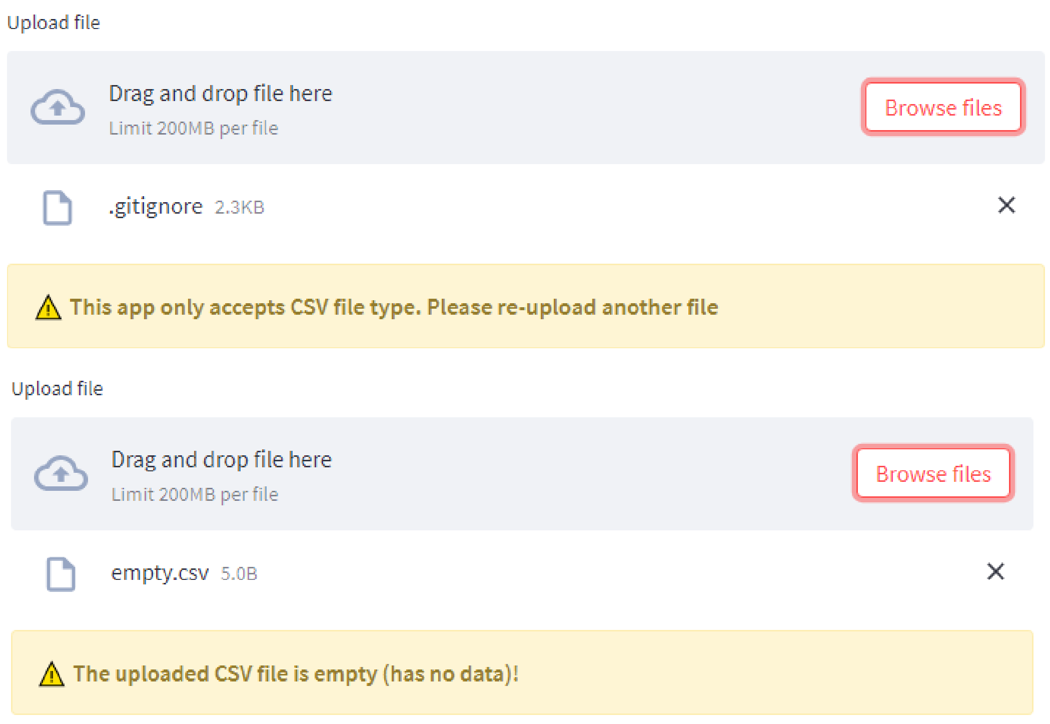
Authors:

- Darren Li
- Jason Nguyen
- Liam Huang
- Nick Drage

Upload file

Once the app is launched, an upload file section appears. A user can simply drag and drop a file into the designated area or click on the Browse files button to upload it.

The Data Explorer Tool only accepts a file of CSV type, and will display a warning message when other file types are uploaded. Additionally, if an empty CSV file is uploaded, the app will display a warning message alerting the user to the fact that the uploaded file has no data.



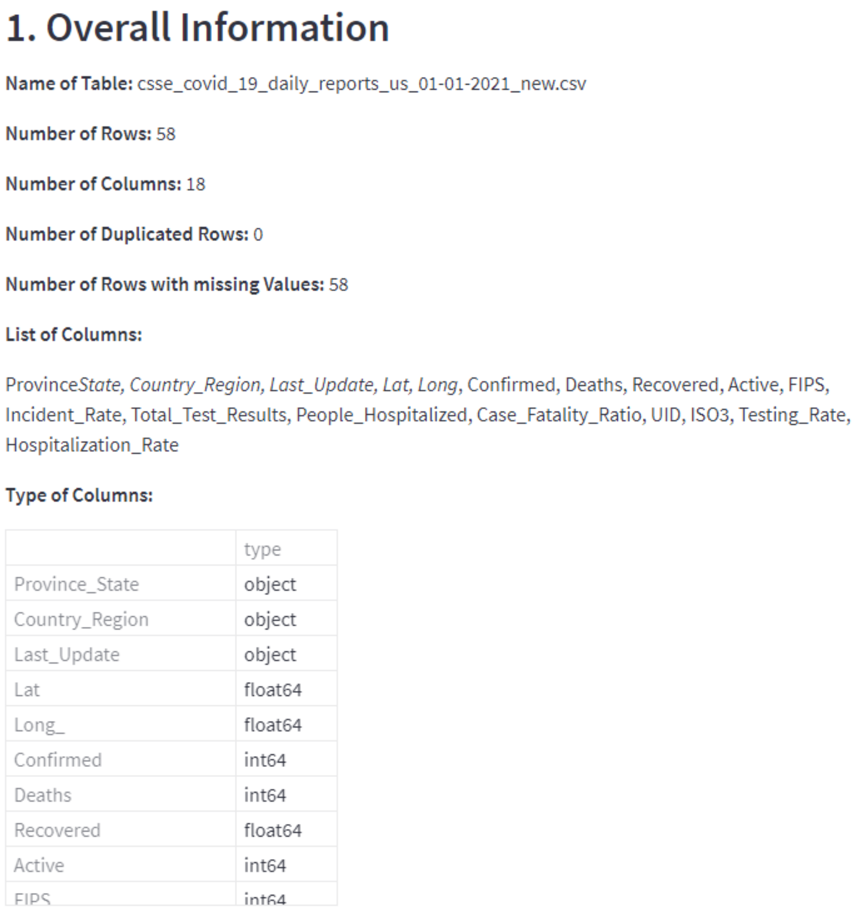
⚠ This app only accepts CSV file type. Please re-upload another file

⚠ The uploaded CSV file is empty (has no data)!

1. Overall Information

Section displays the following attributes of a whole dataset:

- Name of the uploaded file
- Number of rows
- Number of columns
- Number of duplicated rows
- Number of rows with missing values
- List of column names
- Type of each column
- A slider to allow users select how many n rows of the dataset to be displayed (defaulted at 5 rows) in the following 3 tables
- An interactive table displays the top n rows of the dataset
- An interactive table displays the bottom n rows of the dataset
- An interactive table displays random sample n rows of the dataset
- A multiselect button to allow users selecting which column(s) to be converted to datetime type



1. Overall Information

Name of Table: csse_covid_19_daily_reports_us_01-01-2021_new.csv

Number of Rows: 58

Number of Columns: 18

Number of Duplicated Rows: 0

Number of Rows with missing Values: 58

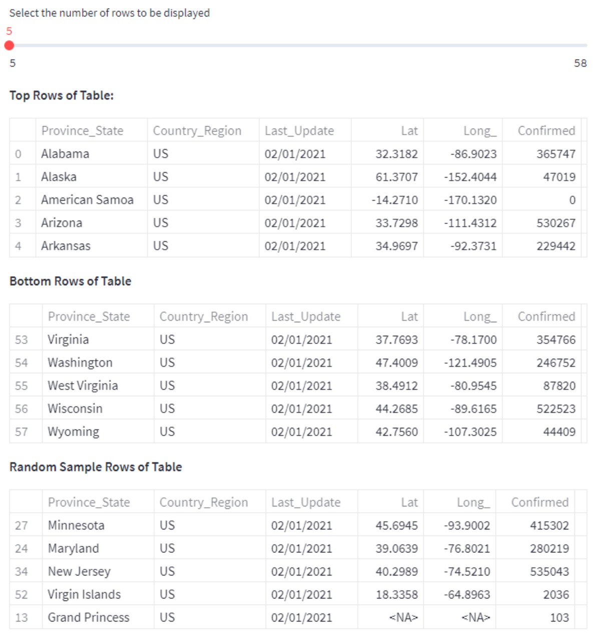
List of Columns:

Province*State, Country_Region, Last_Update, Lat, Long*, Confirmed, Deaths, Recovered, Active, FIPS, Incident_Rate, Total_Test_Results, People_Hospitalized, Case_Fatality_Ratio, UID, ISO3, Testing_Rate, Hospitalization_Rate

Type of Columns:

	type
Province_State	object
Country_Region	object
Last_Update	object
Lat	float64
Long_	float64
Confirmed	int64
Deaths	int64
Recovered	float64
Active	int64
FIPS	int64

Following the type of columns table, a slider will be used for the user to choose the number of rows displayed. The number is defaulted (and minimum) at 5 (left hand side below). But when the dataset has less than 5 rows of data, the default value will be 1 and a warning message will appear on the right hand side below.



Top Rows of Table:

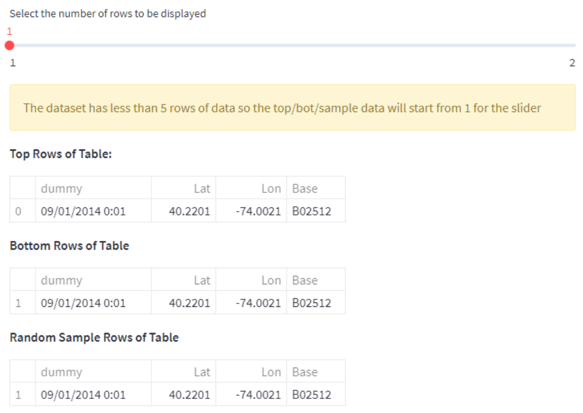
	Province_State	Country_Region	Last_Update	Lat	Long_	Confirmed
0	Alabama	US	02/01/2021	32.3182	-86.9023	365747
1	Alaska	US	02/01/2021	61.3707	-152.4044	47019
2	American Samoa	US	02/01/2021	-14.2710	-170.1320	0
3	Arizona	US	02/01/2021	33.7298	-111.4312	530267
4	Arkansas	US	02/01/2021	34.9697	-92.3731	229442

Bottom Rows of Table

	Province_State	Country_Region	Last_Update	Lat	Long_	Confirmed
53	Virginia	US	02/01/2021	37.7693	-78.1700	354766
54	Washington	US	02/01/2021	47.4009	-121.4905	246752
55	West Virginia	US	02/01/2021	38.4912	-80.9545	87820
56	Wisconsin	US	02/01/2021	44.2685	-89.6165	522523
57	Wyoming	US	02/01/2021	42.7560	-107.3025	44409

Random Sample Rows of Table

	Province_State	Country_Region	Last_Update	Lat	Long_	Confirmed
27	Minnesota	US	02/01/2021	45.6945	-93.9002	415302
24	Maryland	US	02/01/2021	39.0639	-76.8021	280219
34	New Jersey	US	02/01/2021	40.2989	-74.5210	535043
52	Virgin Islands	US	02/01/2021	18.3358	-64.8963	2036
13	Grand Princess	US	02/01/2021	<NA>	<NA>	103



The dataset has less than 5 rows of data so the top/bot/sample data will start from 1 for the slider

Top Rows of Table:

	dummy	Lat	Lon	Base
0	09/01/2014 0:01	40.2201	-74.0021	B02512

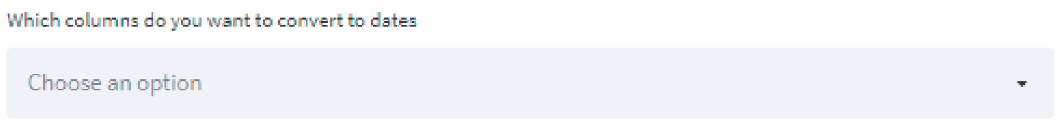
Bottom Rows of Table

	dummy	Lat	Lon	Base
1	09/01/2014 0:01	40.2201	-74.0021	B02512

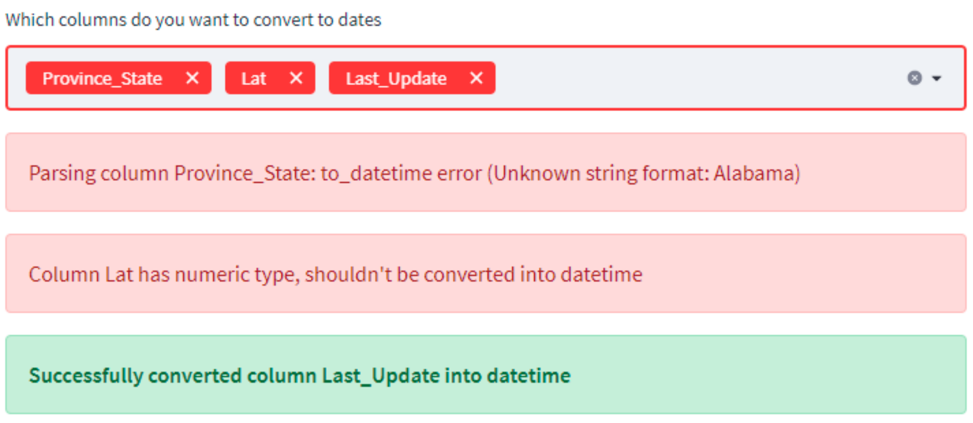
Random Sample Rows of Table

	dummy	Lat	Lon	Base
1	09/01/2014 0:01	40.2201	-74.0021	B02512

Following the table of a random sample from the dataset is a multiselect button which allows users to select what columns to be converted to datetime.



We implemented a mechanism which gives a warning if an error occurs in datetime conversion, or a column of numeric type that shouldn't be converted.



Parsing column Province_State: to_datetime error (Unknown string format: Alabama)

Column Lat has numeric type, shouldn't be converted into datetime

Successfully converted column Last_Update into datetime

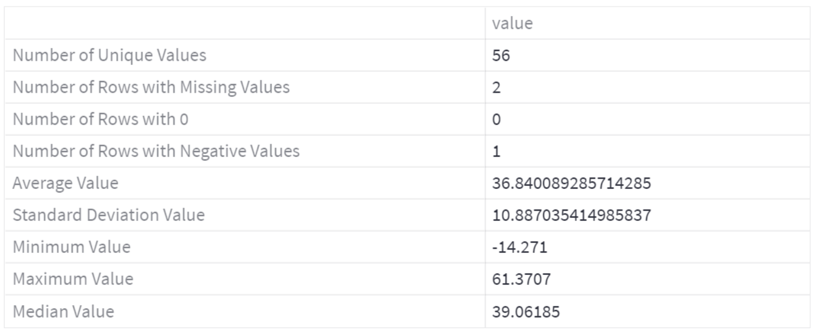
After the overall information section, the Data Explorer Tool will look at each of the columns and display their key features, starting with numeric columns, followed by text and finally datetime columns.

2. Numeric Columns

At a glance this information enables a user to view high level statistical information that saves manual work usually required in excel to perform the same calculations. Here are the key features that a numerical column from the dataset will be performing analysis on.

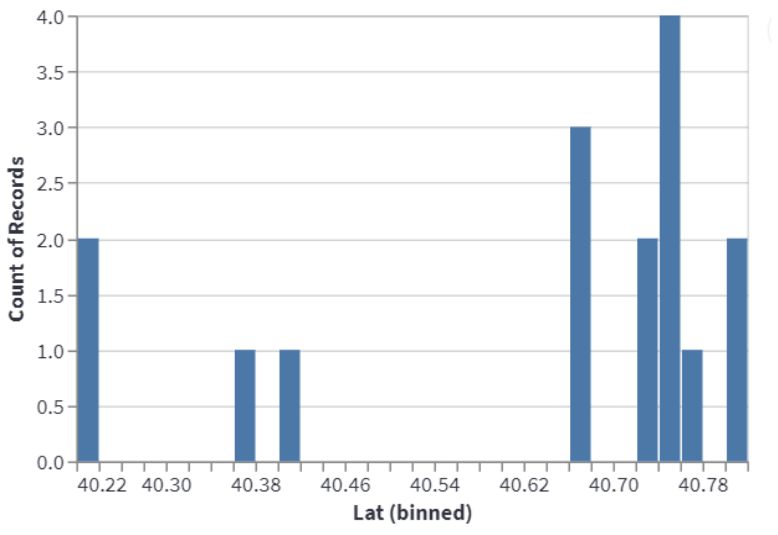
- Number of unique values
- Number of rows with missing values
- Number of rows with a value of zero
- Number of rows whose value is negative
- Mean of the column
- Standard deviation of column
- Maximum value of column
- Minimum value of column
- Median value of column
- Histogram chart with distribution count of unique values
- A table of most frequent values (top 20)

2.1 Field Name: *Lat*

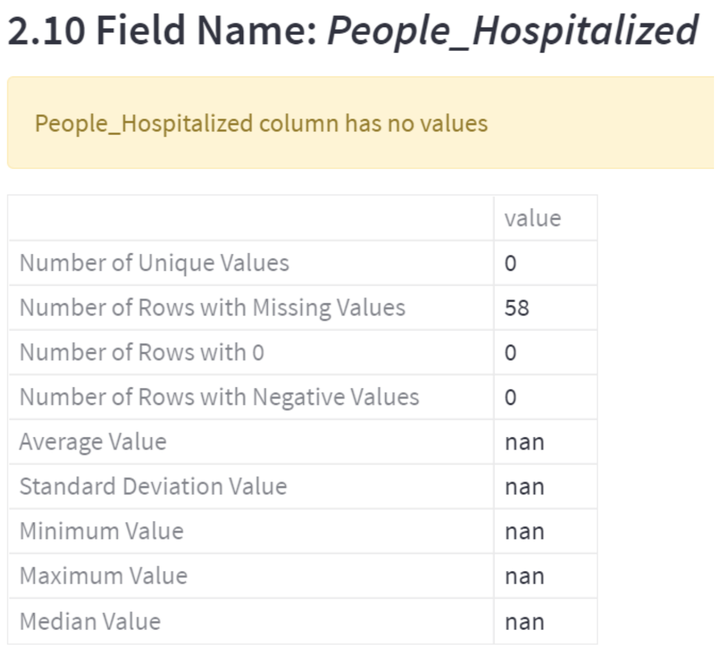


	value
Number of Unique Values	56
Number of Rows with Missing Values	2
Number of Rows with 0	0
Number of Rows with Negative Values	1
Average Value	36.840089285714285
Standard Deviation Value	10.887035414985837
Minimum Value	-14.271
Maximum Value	61.3707
Median Value	39.06185

Histogram



- If no values appear in a column, a warning message appears

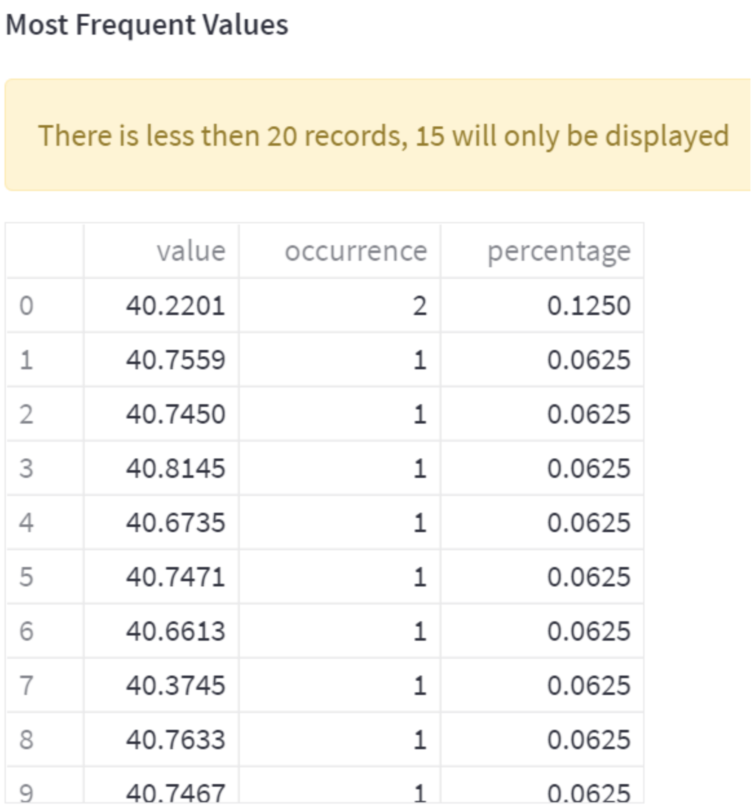


2.10 Field Name: *People_Hospitalized*

People_Hospitalized column has no values

	value
Number of Unique Values	0
Number of Rows with Missing Values	58
Number of Rows with 0	0
Number of Rows with Negative Values	0
Average Value	nan
Standard Deviation Value	nan
Minimum Value	nan
Maximum Value	nan
Median Value	nan

- If a column has less than 20 unique values a warning message will appear to indicate how many values are included in the column.



Most Frequent Values

There is less then 20 records, 15 will only be displayed

	value	occurrence	percentage
0	40.2201	2	0.1250
1	40.7559	1	0.0625
2	40.7450	1	0.0625
3	40.8145	1	0.0625
4	40.6735	1	0.0625
5	40.7471	1	0.0625
6	40.6613	1	0.0625
7	40.3745	1	0.0625
8	40.7633	1	0.0625
9	40.7467	1	0.0625

3. Text Columns

Section displays the following attributes of a text column

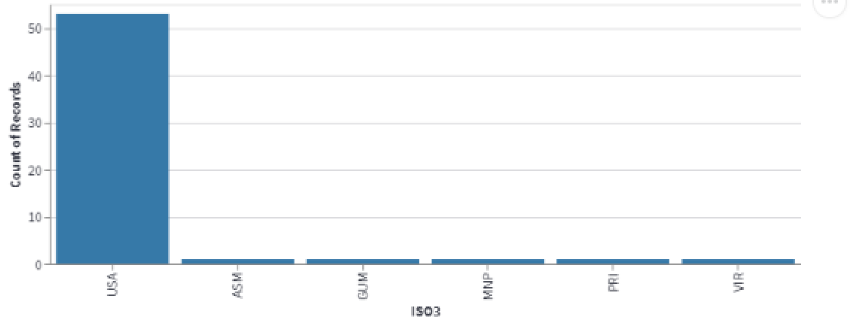
- Name of column as subtitle
- Number of unique values
- Number of missing values
- Number of rows with empty string
- Number of rows with only whitespaces
- Number of rows with only lower case characters
- Number of rows with only upper case characters
- Number of rows with only alphabet characters
- Number of rows with only numbers as characters
- Mode value
- A bar chart showing the number of occurrence for each value
- A table listing the occurrences and percentage of the top 20 most frequent values

Below is an example of such a display for the text column ISO3.

3.3 Field Name: *ISO3*

	value
Number of Unique Values	6
Number of Rows with Missing Values	0
Number of Empty Rows	0
Number of Rows with Only Whitespace	0
Number of Rows with Only Lowercases	0
Number of Rows with Only Uppercases	58
Number of Rows with Only Alphabet	58
Number of Rows with Only Digits	0
Mode Value	USA

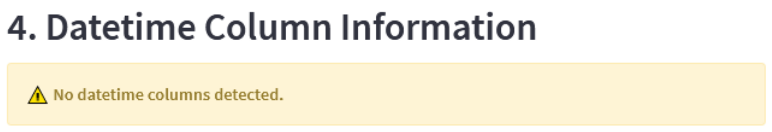
Bar Chart



	Frequency	Percetage
USA	53	0.9138
ASM	1	0.0172
GUM	1	0.0172
MNP	1	0.0172
PRI	1	0.0172
VIR	1	0.0172

4. Datetime Columns

If there are no datetime columns (original or converted), the following warning will be displayed.

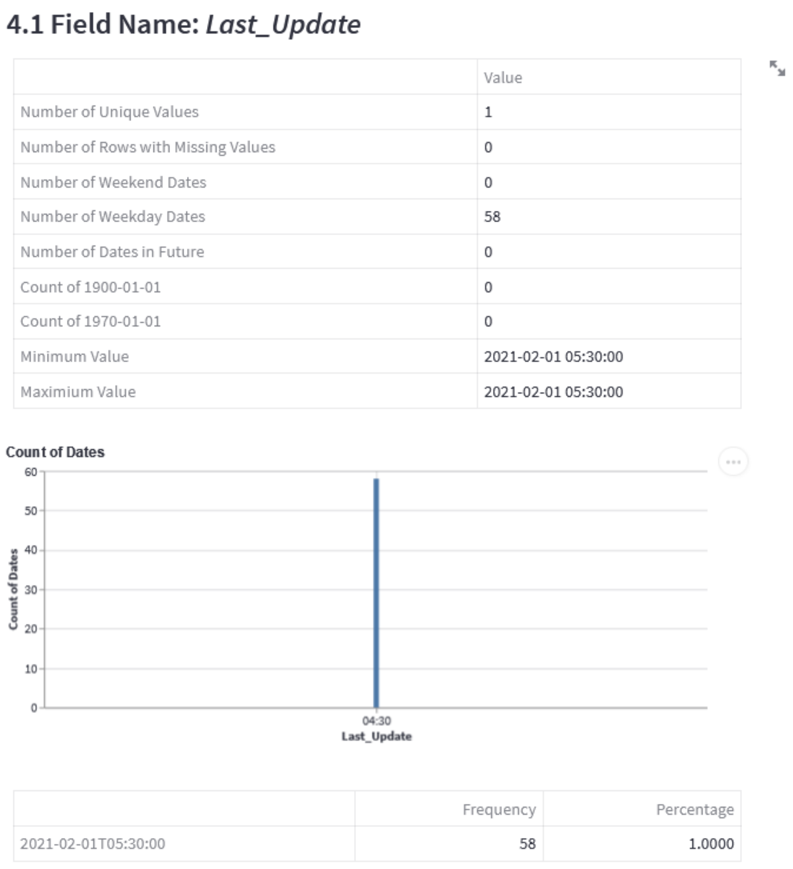


4. Datetime Column Information

⚠ No datetime columns detected.

Like the numeric column, the datetime column attribute table contains the number of unique values, missing data, & maximum and minimum. It also contains:

- Number of rows with dates on the weekend (Saturday, Sunday)
- Number of rows with dates on a weekday
- Number of rows with dates in future, from the time of uploading the data
- Number of rows with dates equalling 1/1/1900
- Number of rows with dates equalling 1/1/1970



4.1 Field Name: *Last_Update*

	Value
Number of Unique Values	1
Number of Rows with Missing Values	0
Number of Weekend Dates	0
Number of Weekday Dates	58
Number of Dates in Future	0
Count of 1900-01-01	0
Count of 1970-01-01	0
Minimum Value	2021-02-01 05:30:00
Maximium Value	2021-02-01 05:30:00

	Frequency	Percentage
2021-02-01T05:30:00	58	1.0000

After the entail table, there is a histogram of values and a table containing the frequency and percentage of the twenty most prolific values (or all if there are only twenty or less unique values).

Project procedure and cadences

- A github repository was created.
- Each member cloned the repo to work on their allocated branch. The naming convention for each branch was to use the corresponding team member's first name.
- The tasks allocated to each member were:
    - Darren: worked on datetime part
    - Jason: worked on text part
    - Liam: worked on data part
    - Nick: worked on numeric part
- Members pushed their code into github once ready. A member created a pull request and required all other 3 members to review.
- The required reviewers gave constructive and helpful feedback before a pull request was merged into the main *master* branch.
- Weekly meetings on Saturday on Zoom/Slack to discuss the progress and solve any obstacles encountered.
- Additional meetings on Tuesday and Thursday in the last week of the project to finalise loose ends.
- Supportive help sessions were given to cross-check the codes accuracy and efficiency
- Slack messaging during anytime of any days of the week to get quick responses and action taken.

Group collaboration on the project

Student A (**Liam**) worked on:

1. Dockerfile
2. Docker-compose.yml
3. README.md
4. Streamlit_app.py (student A section)
5. data.py and test_data.py
6. numeric.py and test_numeric.py (assist)
7. dummy_trim.csv (for testing purposes)
8. Final report

Student B: (**Nick**) worked on:

1. Streamlit_app.py (student B section)
2. numeric.py and test_numeric.py
3. dummy_trim_test_top_20_error.csv for error testing
4. Final report

Student C (**Jason**) worked on:

1. .gitignore
2. README.md
3. Streamlit_app.py (student C section)
4. Streamlit_app_jason.py (as an alternative version for testing purposes)

5. Text.py and test_text.py
6. Data_jason.py (as an alternative version for testing purposes)
7. csse_covid_19_daily_reports_us_01-01-2021 (for testing purposes)
8. Final report

Student D:(**Darren**) worked on:

1. Dockerfile
2. streamlit_app.py (student D section)
3. datetime.py and test_datetime.py
4. Final report

Problems faced and implemented solutions

The project was aimed to familiarise students with the process of building a streamlit application that is hosted on a docker container.

Surprisingly, we faced more challenges on using git than building the app and docker container, especially when merging an individual working branch to the main master branch where conflicts between the two branches arose.

We solved those conflicting merges by either cancelling a particular problematic merge, or using git merge tool to resolve the files' differences between the two branches.

Positively, those encountered problems created opportunities for us to improve on the following aspects:

- Gaining more experiences and better knowledge of using git
- Improving our communication skills and managing conflicts
- Improving knowledge of how to use docker and streamlit app

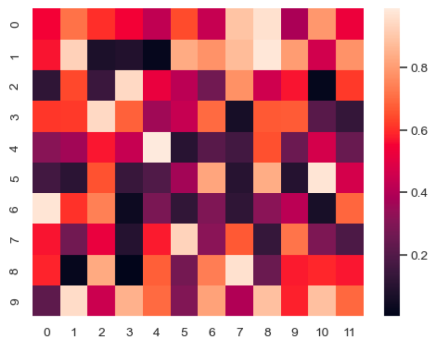
Suggested improvements

- Our team originally designed the file_uploader with the 'type' parameter as 'csv' which prevented a user from uploading a non-csv file. To be aligned with the assessment brief however we had no choice but to remove that parameter and implemented a mechanism that checks whether an appropriate file type (csv) was uploaded.

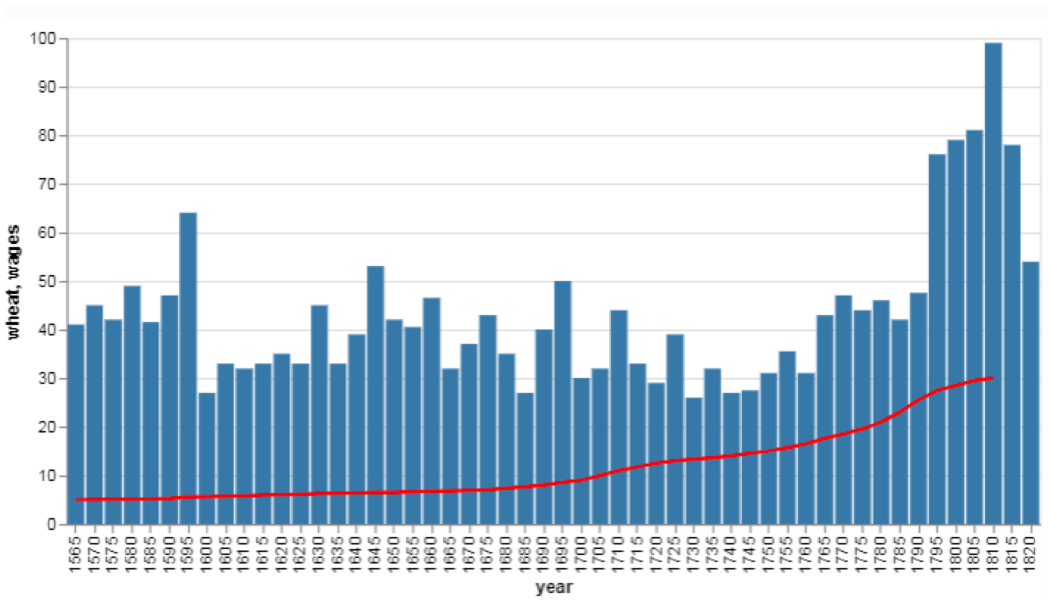
- The app should let users decide how many modes they want to be displayed in the graph. If a column has a large number of unique values, the graph would become less effective in communicating insights to the audience.

- The app should let users decide what column(s) they want to be explored (and visually displayed). Without it, the app display could become large where it takes more time to scroll up/down to the interested section if the uploaded file has a large number of columns.

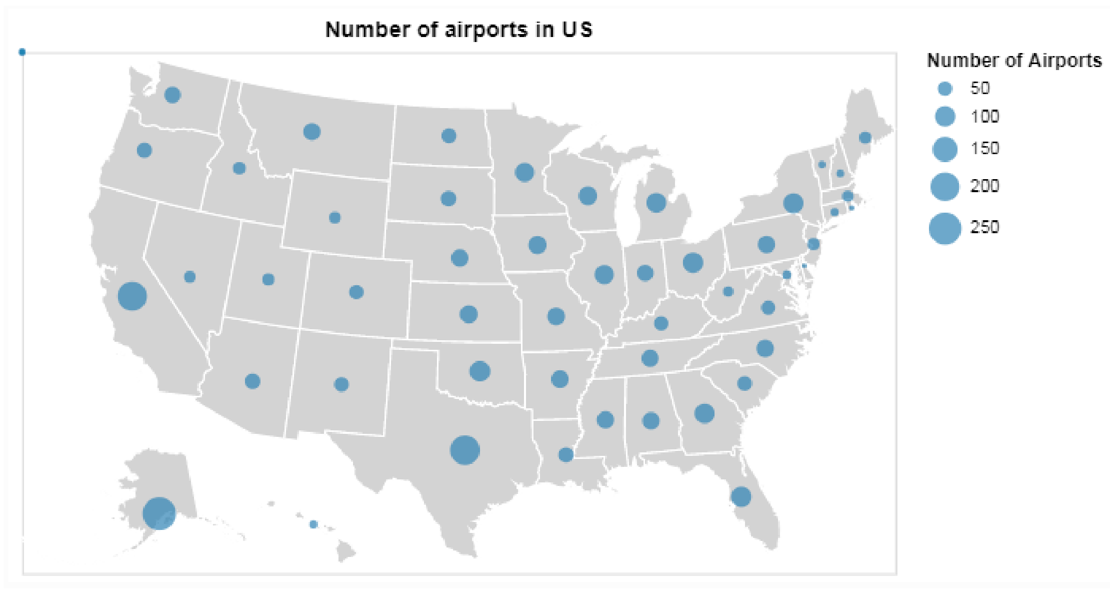
- The app should have a correlation heatmap that shows the relationship between two numeric columns



- The app should have a bar/line graph that shows the relationship between text/datetime and numeric column(s)



- The app should have a map if the data contains geospatial data



Number of airports in US
Number of Airports
50
100
150
200
250