Launch Sites Locations Analysis with Folium

Estimated time needed: **40** minutes

The launch success rate may depend on many factors such as payload mass, orbit type, and so on. It may also depend on the location and proximities of a launch site, i.e., the initial position of rocket trajectories. Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.

In the previous exploratory data analysis labs, you have visualized the SpaceX launch dataset using `matplotlib` and `seaborn` and discovered some preliminary correlations between the launch site and success rates. In this lab, you will be performing more interactive visual analytics using `Folium`.

Objectives

This lab contains the following tasks:

- **TASK 1:** Mark all launch sites on a map
- **TASK 2:** Mark the success/failed launches for each site on the map
- **TASK 3:** Calculate the distances between a launch site to its proximities

After completed the above tasks, you should be able to find some geographical patterns about launch sites.

Let's first import required Python packages for this lab:

```
In [18]: %pip install folium
```

```
Requirement already satisfied: folium in /opt/anaconda3/lib/python3.11/sit
e-packages (0.17.0)
Requirement already satisfied: branca>=0.6.0 in /opt/anaconda3/lib/python
3.11/site-packages (from folium) (0.8.0)
Requirement already satisfied: jinja2>=2.9 in /opt/anaconda3/lib/python3.1
1/site-packages (from folium) (3.1.3)
Requirement already satisfied: numpy in /opt/anaconda3/lib/python3.11/site
-packages (from folium) (1.26.4)
Requirement already satisfied: requests in /opt/anaconda3/lib/python3.11/s
ite-packages (from folium) (2.32.3)
Requirement already satisfied: xyzservices in /opt/anaconda3/lib/python3.1
1/site-packages (from folium) (2022.9.0)
Requirement already satisfied: MarkupSafe>=2.0 in /opt/anaconda3/lib/pytho
n3.11/site-packages (from jinja2>=2.9->folium) (2.1.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /opt/anaconda3/
lib/python3.11/site-packages (from requests->folium) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /opt/anaconda3/lib/python3.
11/site-packages (from requests->folium) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /opt/anaconda3/lib/py
thon3.11/site-packages (from requests->folium) (1.26.19)
Requirement already satisfied: certifi>=2017.4.17 in /opt/anaconda3/lib/py
thon3.11/site-packages (from requests->folium) (2024.7.4)
Note: you may need to restart the kernel to use updated packages.
```

In [20]:
```python
import folium
import pandas as pd
```

In [22]:
```python
# Import folium MarkerCluster plugin
from folium.plugins import MarkerCluster
# Import folium MousePosition plugin
from folium.plugins import MousePosition
# Import folium DivIcon plugin
from folium.features import DivIcon
```

If you need to refresh your memory about folium, you may download and refer to this previous folium lab:

Generating Maps with Python

In [26]:
```python
## Task 1: Mark all launch sites on a map

# Example Data: Launch site names, latitudes, and longitudes
launch_sites_df = pd.DataFrame({
    'LaunchSite': ['CCAFS SLC 40', 'VAFB SLC 4E', 'KSC LC 39A', 'CCAFS LC
    'Latitude': [28.562302, 34.632834, 28.573255, 28.561857],
    'Longitude': [-80.577366, -120.610746, -80.651832, -80.577366]
})

# Create a map centered around a location (central Florida in this case)
launch_map = folium.Map(location=[30, -100], zoom_start=4)

# Add markers for each launch site
for index, site in launch_sites_df.iterrows():
    folium.Marker(
        [site['Latitude'], site['Longitude']],
        popup=site['LaunchSite'],
    ).add_to(launch_map)

# Display the map
launch_map
```

Out[26]:

First, let's try to add each site's location on a map using site's latitude and longitude coordinates

The following dataset with the name `spacex_launch_geo.csv` is an augmented dataset with latitude and longitude added for each site.

In [111…

```
import pandas as pd

# Correct approach: Using pandas to read CSV from URL
url = 'https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud

# Read the CSV file directly from the URL
spacex_df = pd.read_csv(url)

# Display the first few rows to check if the file has been loaded correct
spacex_df.head()
```

Out[111…

	Flight Number	Date	Time (UTC)	Booster Version	Launch Site	Payload	Payload Mass (kg)	Orbit	Custon
0	1	2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	Spac
1	2	2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel o...	0.0	LEO (ISS)	NA (CO N
2	3	2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2+	525.0	LEO (ISS)	NA (CO
3	4	2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NA (CI
4	5	2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NA (CI

Now, you can take a look at what are the coordinates for each site.

In [114…

```
# Select relevant sub-columns: `Launch Site`, `Lat(Latitude)`, `Long(Long
spacex_df = spacex_df[['Launch Site', 'Lat', 'Long', 'class']]
launch_sites_df = spacex_df.groupby(['Launch Site'], as_index=False).firs
launch_sites_df = launch_sites_df[['Launch Site', 'Lat', 'Long']]
launch_sites_df
```

Out[114…

	Launch Site	Lat	Long
0	CCAFS LC-40	28.562302	-80.577356
1	CCAFS SLC-40	28.563197	-80.576820
2	KSC LC-39A	28.573255	-80.646895
3	VAFB SLC-4E	34.632834	-120.610745

Above coordinates are just plain numbers that can not give you any intuitive insights about where are those launch sites. If you are very good at geography, you can interpret those numbers directly in your mind. If not, that's fine too. Let's visualize those locations by pinning them on a map.

We first need to create a folium `Map` object, with an initial center location to be NASA Johnson Space Center at Houston, Texas.

In [118…

```
# Start location is NASA Johnson Space Center
nasa_coordinate = [29.559684888503615, -95.0830971930759]
site_map = folium.Map(location=nasa_coordinate, zoom_start=10)
```

We could use `folium.Circle` to add a highlighted circle area with a text label on a specific coordinate. For example,

In [121…
```python
# Create a blue circle at NASA Johnson Space Center's coordinate with a p
circle = folium.Circle(nasa_coordinate, radius=1000, color='#d35400', fil
# Create a blue circle at NASA Johnson Space Center's coordinate with a i
marker = folium.map.Marker(
    nasa_coordinate,
    # Create an icon as a text label
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>'
        )
    )
site_map.add_child(circle)
site_map.add_child(marker)
```

Out[121…

and you should find a small yellow circle near the city of Houston and you can zoom-in to see a larger circle.

Now, let's add a circle for each launch site in data frame `launch_sites`

TODO: Create and add `folium.Circle` and `folium.Marker` for each launch site on the site map

An example of folium.Circle:

`folium.Circle(coordinate, radius=1000, color='#000000', fill=True).add_child(folium.Popup(...))`

An example of folium.Marker:

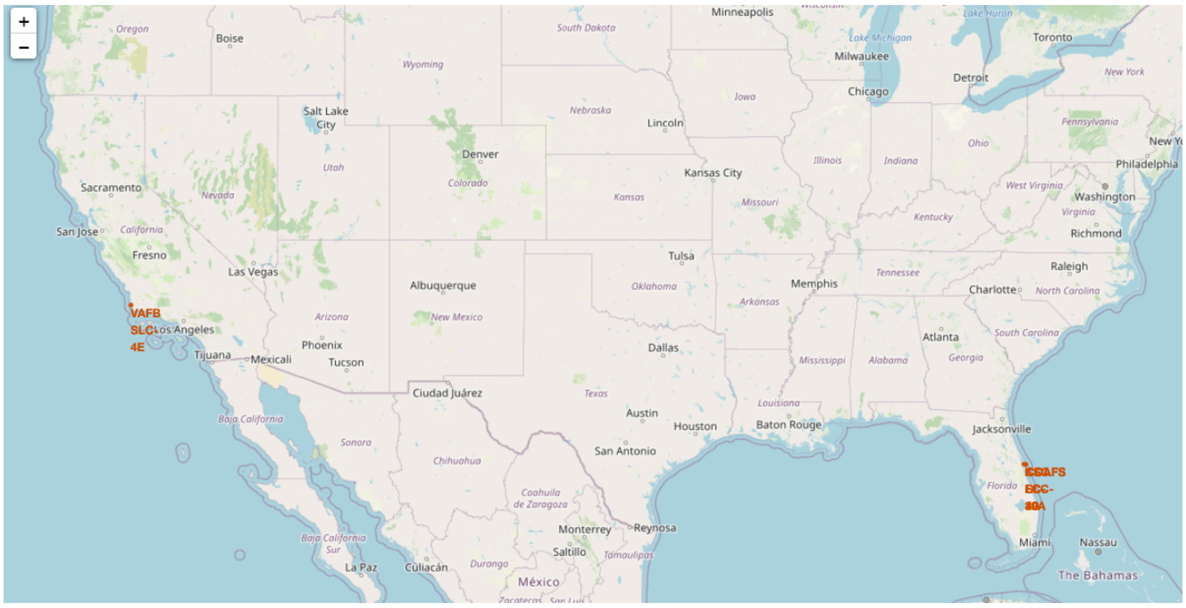
`folium.map.Marker(coordinate, icon=DivIcon(icon_size=(20,20),icon_anchor=(0,0), html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>' % 'label', ))`

In [130…

```python
# Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=5)
# For each launch site, add a Circle object based on its coordinate (Lat,

# Coordinates for the launch sites
launch_sites_df = pd.DataFrame({
    'LaunchSite': ['CCAFS SLC 40', 'VAFB SLC 4E', 'KSC LC 39A', 'CCAFS LC
    'Latitude': [28.562302, 34.632834, 28.573255, 28.561857],
    'Longitude': [-80.577366, -120.610746, -80.651832, -80.577366]
})

# Initialize a map centered around the US
site_map = folium.Map(location=[30, -100], zoom_start=5)

# Add a circle and a marker for each launch site
for index, site in launch_sites_df.iterrows():
    # Add a circle at the launch site
    folium.Circle(
        location=[site['Latitude'], site['Longitude']],
        radius=1000,  # Radius in meters
        color='#d35400',  # Circle color
        fill=True,
        fill_opacity=0.6,
        popup=site['LaunchSite']  # Popup label with launch site name
    ).add_to(site_map)

    # Add a marker for the launch site
    folium.Marker(
        location=[site['Latitude'], site['Longitude']],
        icon=folium.DivIcon(html=f'<div style="font-size: 12px; color:#d3
    ).add_to(site_map)

# Display the map
site_map
```

Out [130…

The generated map with marked launch sites should look similar to the following:



Now, you can explore the map by zoom-in/out the marked areas , and try to answer the following questions:

- Are all launch sites in proximity to the Equator line?
- Are all launch sites in very close proximity to the coast?

Also please try to explain your findings.

In [135…

```python
# Task 2: Mark the success/failed launches for each site on the map
import folium
from folium.plugins import MarkerCluster

# Initialize the map centered on one of the launch sites
site_map = folium.Map(location=[28.562302, -80.577366], zoom_start=5)
```

```
# Create a MarkerCluster object to group the markers
marker_cluster = MarkerCluster().add_to(site_map)

# Iterate through the DataFrame and add a marker for each launch
for index, row in spacex_df.iterrows():
    # Define the color of the marker based on the class (launch success/f
    marker_color = 'green' if row['class'] == 1 else 'red'

    # Add the marker to the map
    folium.Marker(
        location=[row['Lat'], row['Long']],
        popup=f"Launch Site: {row['Launch Site']}\nClass: {row['class']}"
        icon=folium.Icon(color=marker_color)
    ).add_to(marker_cluster)

# Display the map
site_map
```

Out[135…

Next, let's try to enhance the map by adding the launch outcomes for each site, and see which sites have high success rates. Recall that data frame spacex_df has detailed launch records, and the `class` column indicates if this launch was successful or not

In [138…

```
spacex_df.tail(10)
```

Out[138…

	Launch Site	Lat	Long	class
46	KSC LC-39A	28.573255	-80.646895	1
47	KSC LC-39A	28.573255	-80.646895	1
48	KSC LC-39A	28.573255	-80.646895	1
49	CCAFS SLC-40	28.563197	-80.576820	1
50	CCAFS SLC-40	28.563197	-80.576820	1
51	CCAFS SLC-40	28.563197	-80.576820	0
52	CCAFS SLC-40	28.563197	-80.576820	0
53	CCAFS SLC-40	28.563197	-80.576820	0
54	CCAFS SLC-40	28.563197	-80.576820	1
55	CCAFS SLC-40	28.563197	-80.576820	0

Next, let's create markers for all launch records. If a launch was successful `(class=1)`, then we use a green marker and if a launch was failed, we use a red marker `(class=0)`

Note that a launch only happens in one of the four launch sites, which means many launch records will have the exact same coordinate. Marker clusters can be a good way to simplify a map containing many markers having the same coordinate.

Let's first create a `MarkerCluster` object

In [143…
```
marker_cluster = MarkerCluster()
```

TODO: Create a new column in `spacex_df` dataframe called `marker_color` to store the marker colors based on the `class` value

In [146…
```
# Apply a function to check the value of `class` column
# If class=1, marker_color value will be green
# If class=0, marker_color value will be red

# Assuming the DataFrame is named 'spacex_df'

# Create a function to assign marker colors based on class
def assign_marker_color(launch_class):
    if launch_class == 1:
        return 'green'
    else:
        return 'red'
```

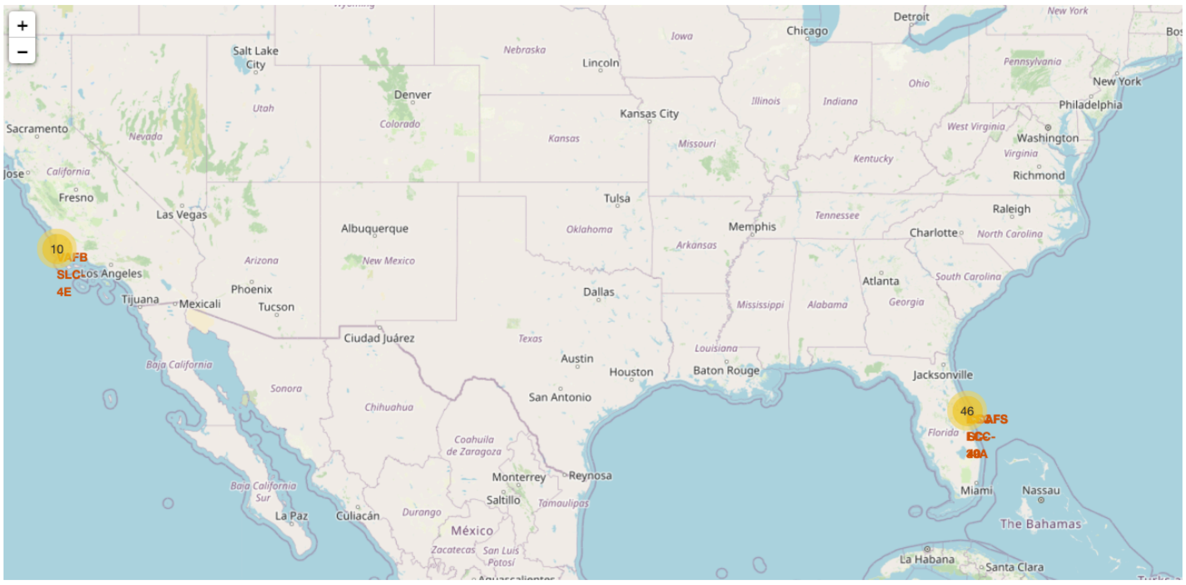
TODO: For each launch result in `spacex_df` data frame, add a `folium.Marker` to `marker_cluster`

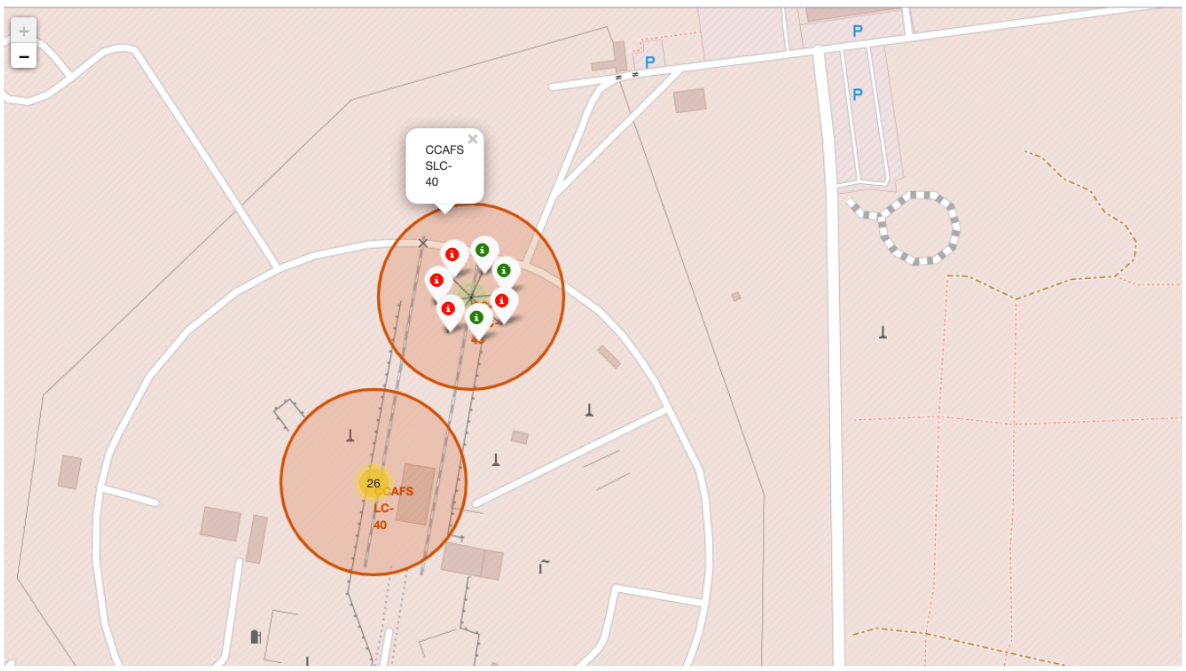
In [149…
```
# Add marker_cluster to current site_map
site_map.add_child(marker_cluster)
```

```
# for each row in spacex_df data frame
# create a Marker object with its coordinate
# and customize the Marker's icon property to indicate if this launch was
# e.g., icon=folium.Icon(color='white', icon_color=row['marker_color']
for index, record in spacex_df.iterrows():
    # TODO: Create and add a Marker cluster to the site map
    # marker = folium.Marker(...)
    marker_cluster.add_child(marker)

site_map
```

Out[149…

Your updated map may look like the following screenshots:





From the color-labeled markers in marker clusters, you should be able to easily identify which launch sites have relatively high success rates.

In [155…

```
# TASK 3: Calculate the distances between a launch site to its proximitie
from folium.plugins import MousePosition

# Initialize the map centered at a launch site
site_map = folium.Map(location=[28.562302, -80.577366], zoom_start=10)

# Add MousePosition plugin to capture the coordinates as you move the mou
MousePosition().add_to(site_map)

# Display the map
site_map
```

Out[155…

Next, we need to explore and analyze the proximities of launch sites.

Let's first add a `MousePosition` on the map to get coordinate for a mouse over a point on the map. As such, while you are exploring the map, you can easily find the coordinates of any points of interests (such as railway)

In [159…
```
# Add Mouse Position to get the coordinate (Lat, Long) for a mouse over o
formatter = "function(num) {return L.Util.formatNum(num, 5);};"
mouse_position = MousePosition(
    position='topright',
    separator=' Long: ',
    empty_string='NaN',
    lng_first=False,
    num_digits=20,
    prefix='Lat:',
    lat_formatter=formatter,
    lng_formatter=formatter,
)

site_map.add_child(mouse_position)
site_map
```

Out[159…

Now zoom in to a launch site and explore its proximity to see if you can easily find any railway, highway, coastline, etc. Move your mouse to these points and mark down their coordinates (shown on the top-left) in order to the distance to the launch site.

Now zoom in to a launch site and explore its proximity to see if you can easily find any railway, highway, coastline, etc. Move your mouse to these points and mark down their coordinates (shown on the top-left) in order to the distance to the launch site.

In [163…
```
from math import sin, cos, sqrt, atan2, radians

def calculate_distance(lat1, lon1, lat2, lon2):
    # approximate radius of earth in km
    R = 6373.0
    lat1 = radians(lat1)
    lon1 = radians(lon1)
    lat2 = radians(lat2)
    lon2 = radians(lon2)

    dlon = lon2 - lon1
    dlat = lat2 - lat1

    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    distance = R * c
    return distance
```

TODO: Mark down a point on the closest coastline using MousePosition and calculate the distance between the coastline point and the launch site.

In [166…

```
%pip install geopy
```

```
Requirement already satisfied: geopy in /opt/anaconda3/lib/python3.11/site
-packages (2.4.1)
Requirement already satisfied: geographiclib<3,>=1.52 in /opt/anaconda3/li
b/python3.11/site-packages (from geopy) (2.0)
Note: you may need to restart the kernel to use updated packages.
```

In [167…

```
# find coordinate of the closet coastline
# e.g.,: Lat: 28.56367  Lon: -80.57163
# distance_coastline = calculate_distance(launch_site_lat, launch_site_lo
from geopy.distance import geodesic
import folium

# Example coordinates for the launch site and coastline point
launch_site_coords = (28.562302, -80.577366)  # CCAFS SLC 40
coastline_coords = (28.56367, -80.57163)  # Example coastline point

# Calculate the distance between the launch site and the coastline
distance = geodesic(launch_site_coords, coastline_coords).km
print(f"Distance between launch site and coastline: {distance:.2f} km")

# Initialize the map centered around the launch site
site_map = folium.Map(location=launch_site_coords, zoom_start=14)

# Add a marker for the launch site
launch_marker = folium.Marker(
    location=launch_site_coords,
    popup=f"Launch Site: CCAFS SLC 40"
).add_to(site_map)

# Add a marker for the closest coastline point
coastline_marker = folium.Marker(
    location=coastline_coords,
    popup=f"Coastline Point: {distance:.2f} KM",
    icon=folium.DivIcon(html=f'<div style="font-size: 12px; color:#d35400
).add_to(site_map)

# Draw a PolyLine between the launch site and the coastline
```

```
line = folium.PolyLine(
    locations=[launch_site_coords, coastline_coords],
    weight=2,
    color='blue'
).add_to(site_map)

# Display the map
site_map
```

```
Distance between launch site and coastline: 0.58 km
```

Out[167…

In [168…

```
# Create and add a folium.Marker on your selected closest coastline point
# Display the distance between coastline point and launch site using the
# for example
# distance_marker = folium.Marker(
#    coordinate,
#    icon=DivIcon(
#        icon_size=(20,20),
#        icon_anchor=(0,0),
#        html='<div style="font-size: 12; color:#d35400;"><b>%s</b></div>
#        )
#    )
```

TODO: Draw a `PolyLine` between a launch site to the selected coastline point

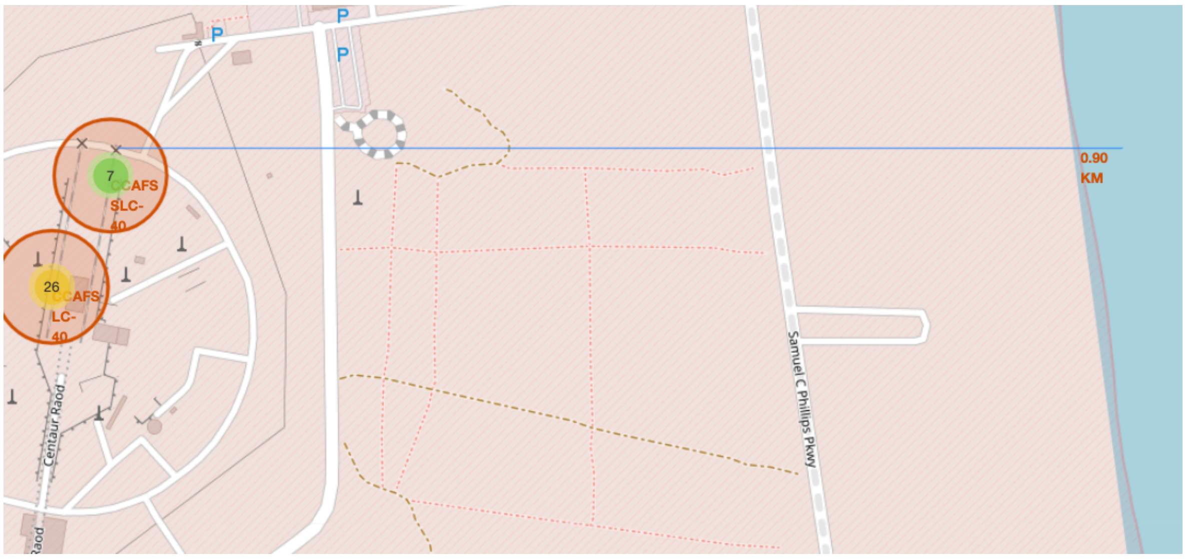
In [170…

```
# Create a `folium.PolyLine` object using the coastline coordinates and l

# Define the launch site and coastline coordinates
launch_site_coords = (28.562302, -80.577366)  # Example launch site coord
coastline_coords = (28.56367, -80.57163)  # Example coastline coordinates

# Create a list of coordinates
coordinates = [launch_site_coords, coastline_coords]

# Create a PolyLine using the coordinates
lines = folium.PolyLine(locations=coordinates, weight=1)
```

```
# Add the PolyLine to the map
site_map.add_child(lines)

# Display the map
site_map
```

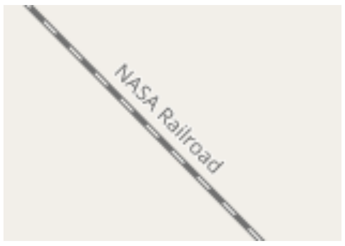
Out[170…

Your updated map with distance line should look like the following screenshot:



TODO: Similarly, you can draw a line betwee a launch site to its closest city, railway, highway, etc. You need to use `MousePosition` to find the their coordinates on the map first

A railway map symbol may look like this:



A highway map symbol may look like this:

A city map symbol may look like this:

In [184…

```
# Create a marker with distance to a closest city, railway, highway, etc.
# Draw a line between the marker to the launch site
```

In [186…

```
from geopy.distance import geodesic
import folium

# Coordinates for the launch site (replace with actual launch site coordi
launch_site_coords = (28.562302, -80.577366)  # Example coordinates for C

# Coordinates for the closest city, railway, highway, or point of interes
poi_coords = (28.0836, -80.6081)  # Example: Melbourne, FL (you can use M

# Calculate the distance between the launch site and the point of interes
distance = geodesic(launch_site_coords, poi_coords).km
print(f"Distance to Point of Interest: {distance:.2f} km")

# Initialize the map centered at the launch site
site_map = folium.Map(location=launch_site_coords, zoom_start=12)

# Add a marker for the launch site
launch_marker = folium.Marker(
    location=launch_site_coords,
    popup="Launch Site: CCAFS SLC 40"
).add_to(site_map)
```

```
# Add a marker for the point of interest (city, railway, etc.)
poi_marker = folium.Marker(
    location=poi_coords,
    popup=f"Point of Interest (City): {distance:.2f} KM",
    icon=folium.DivIcon(html=f'<div style="font-size: 12px; color:#d35400
).add_to(site_map)

# Draw a PolyLine between the launch site and the point of interest
line = folium.PolyLine(
    locations=[launch_site_coords, poi_coords],
    weight=2,
    color='blue'
).add_to(site_map)

# Display the map
site_map
```

```
Distance to Point of Interest: 53.14 km
```

Out[186…

In [ ]:

After you plot distance lines to the proximities, you can answer the following questions easily:

- Are launch sites in close proximity to railways?
- Are launch sites in close proximity to highways?
- Are launch sites in close proximity to coastline?
- Do launch sites keep certain distance away from cities?

Also please try to explain your findings.

Next Steps:

Now you have discovered many interesting insights related to the launch sites' location using folium, in a very interactive way. Next, you will need to build a dashboard using Ploty Dash on detailed launch records.

Authors

Pratiksha Verma

<!--## Change Log--!>

<!--| Date (YYYY-MM-DD) | Version | Changed By | Change Description | | ----------------- | ------- | ------------- | ----------------------- | | 2022-11-09 | 1.0 | Pratiksha Verma | Converted initial version to Jupyterlite|--!>