IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Xilin Li
17.Oct.2021

Outline

Executive Summary

- Summary of methodologies
  1. Data collection
  2. Data wrangling
  3. EDA with data visualization
  4. EDA with SQL
  5. Building an interactive map with Folium
  6. Building a Dashboard with Plotly Dash
  7. Predictive analysis (Classification)
- Summary of all results
  1. Exploratory data analysis results
  2. Interactive analytics demo in screenshots
  3. Predictive analysis results

Introduction

- Project background and context

  We predicted whether the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 launches on its website, with a cost of 62 million dollars; other provider cost 165 million dollars each, much of the savings is because SpaceX can reuse the first stage rocket. Therefore, if we can determine if the first stage rocket will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

- Problems you want to find answers

  1. What influences if the rocket will land successfully?
  2. The effect each relationship with certain rocket variables will impact in determining the success rate of a successful landing.
  3. What conditions does SpaceX have to achieve to get the best results and ensure the best rocket success landing rate.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SpaceX Rest API
  - Web scrapping from Wikipedia
- Perform data wrangling
  - One Hot Encoding data fields for Machine Learning and dropping irrelevant columns
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Plotting: Scatter Graphs, Bar Graphs to show relationships between variables to show patterns of data
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

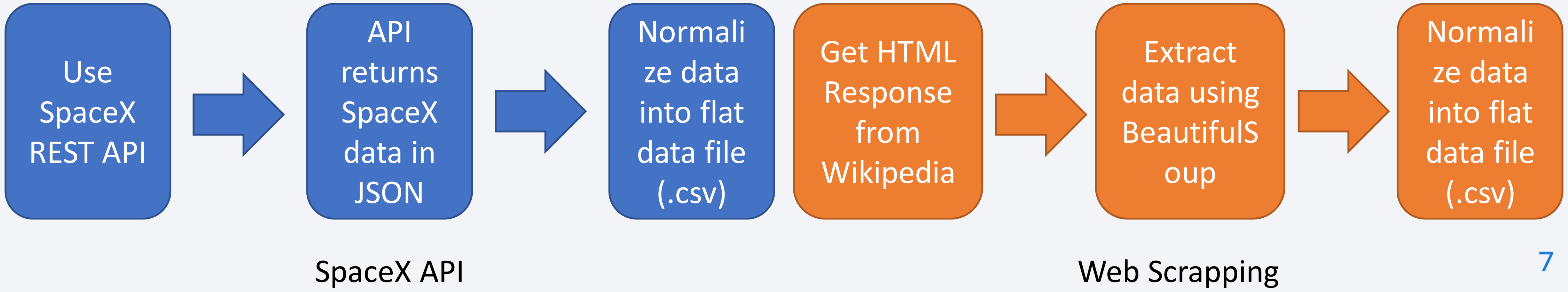
Data Collection

- Describe how data sets were collected.

  The data is collected from the SpaceX REST API, and the API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.

  Another data is collected from Wikipedia using BeautifulSoup.

- You need to present your data collection process use key phrases and flowcharts

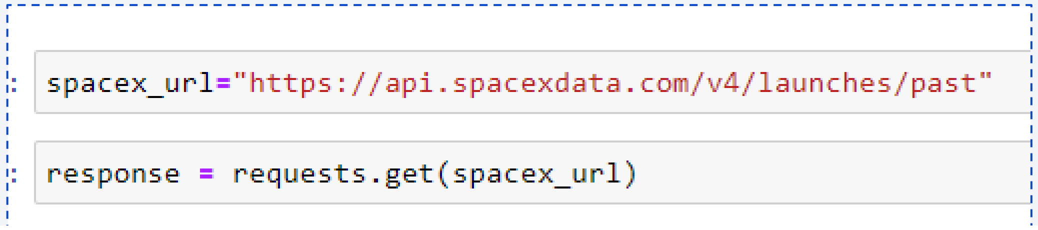

Use SpaceX REST API
API returns SpaceX data in JSON
Normalize data into flat data file (.csv)
Get HTML Response from Wikipedia
Extract data using BeautifulSoup
Normalize data into flat data file (.csv)
SpaceX API
Web Scrapping

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts

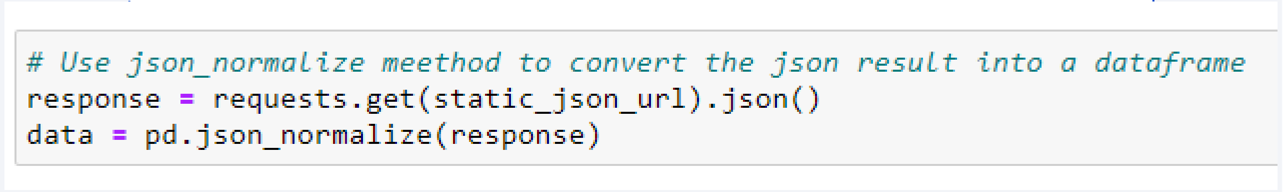
- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose

https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

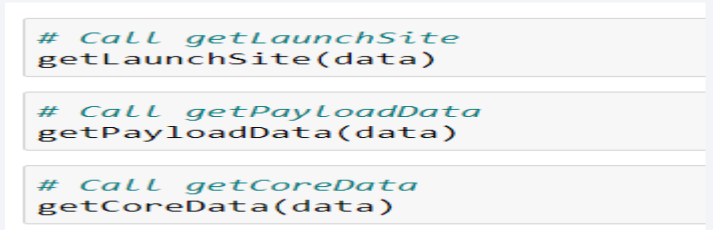


```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```



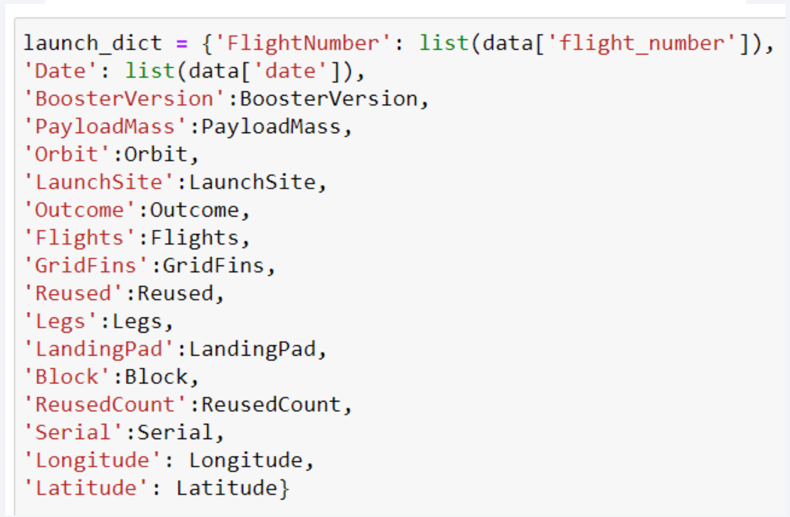
```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```



```
# Call getLaunchSite
getLaunchSite(data)
```

```
# Call getPayloadData
getPayloadData(data)
```

```
# Call getCoreData
getCoreData(data)
```

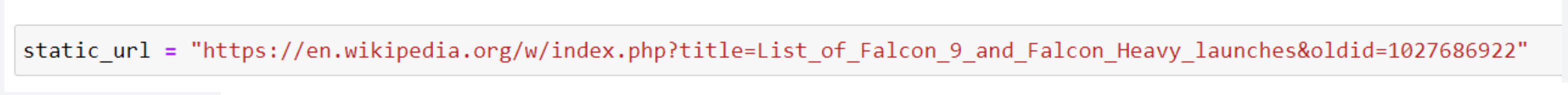


```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

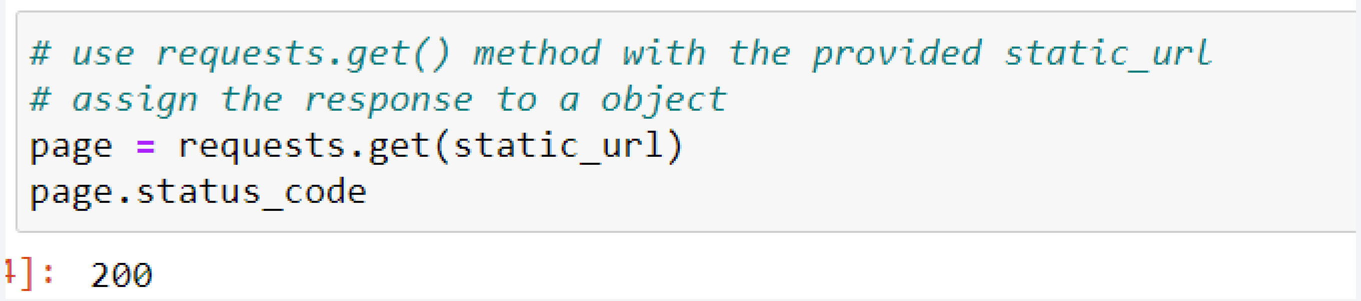
Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts

- https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

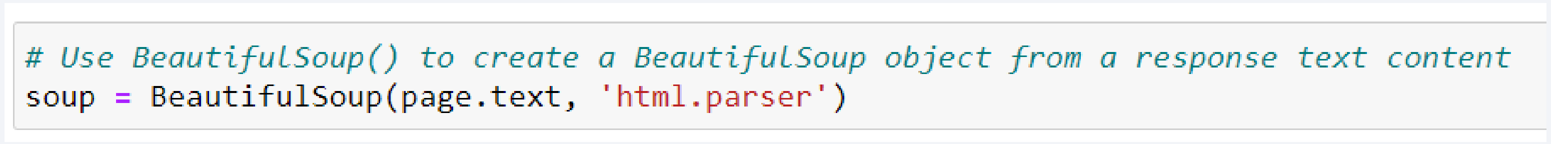


```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

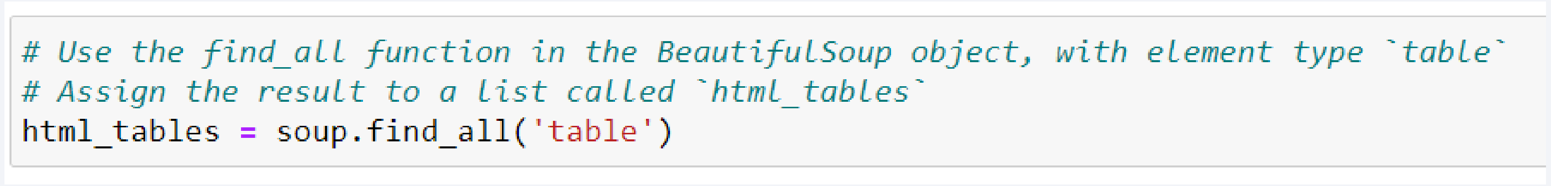


```
# use requests.get() method with the provided static_url
# assign the response to a object
page = requests.get(static_url)
page.status_code
```

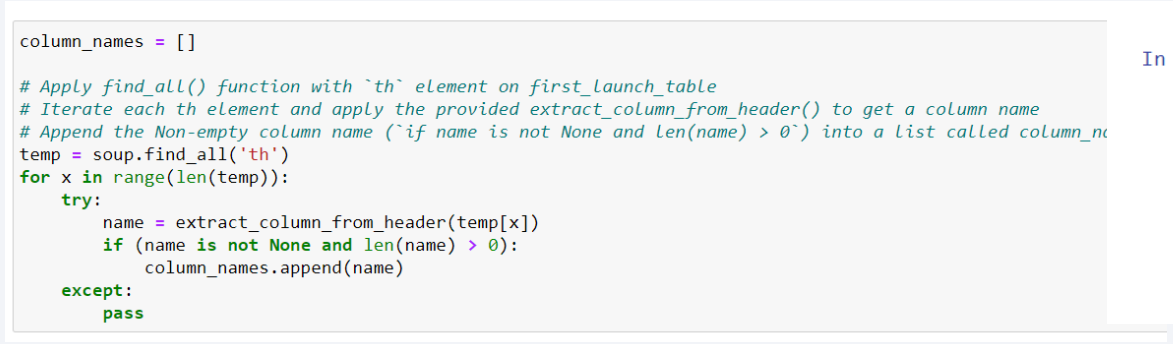
```
]:  200
```



```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(page.text, 'html.parser')
```

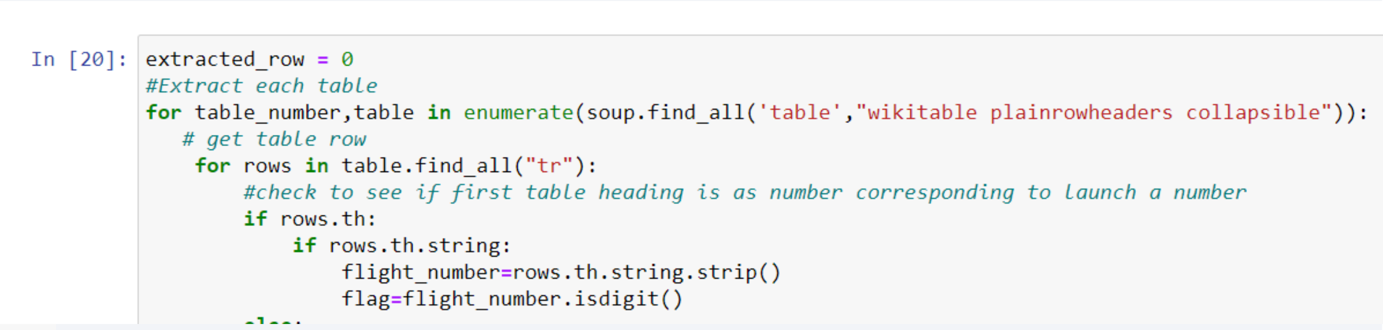


```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
```



```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_n
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
        name = extract_column_from_header(temp[x])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```



```
In [20]: extracted_row = 0
         #Extract each table
         for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
            # get table row
             for rows in table.find_all("tr"):
                 #check to see if first table heading is as number corresponding to launch a number
                 if rows.th:
                     if rows.th.string:
                         flight_number=rows.th.string.strip()
                         flag=flight_number.isdigit()
```

Data Wrangling

- Describe how data were processed

  1. Data Analysis

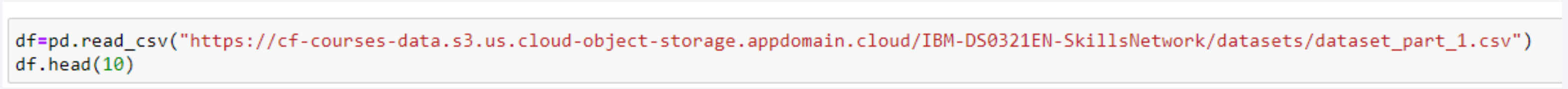
  2. Calculate the number of launches on each site

  3. Calculate the number and occurrence of each orbit

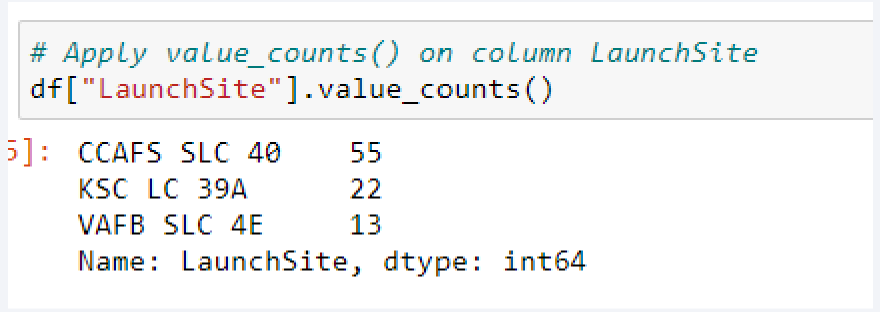
  4.Calculate the number and occurrence of mission outcome per orbit type

  5. Create a landing outcome label from Outcome column

- You need to present your data wrangling process using key phrases and flowcharts
- https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

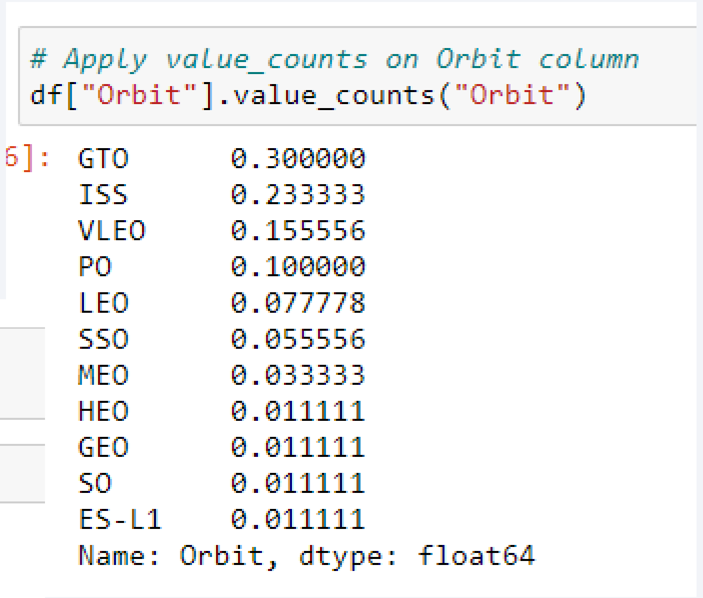


```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_1.csv")
df.head(10)
```



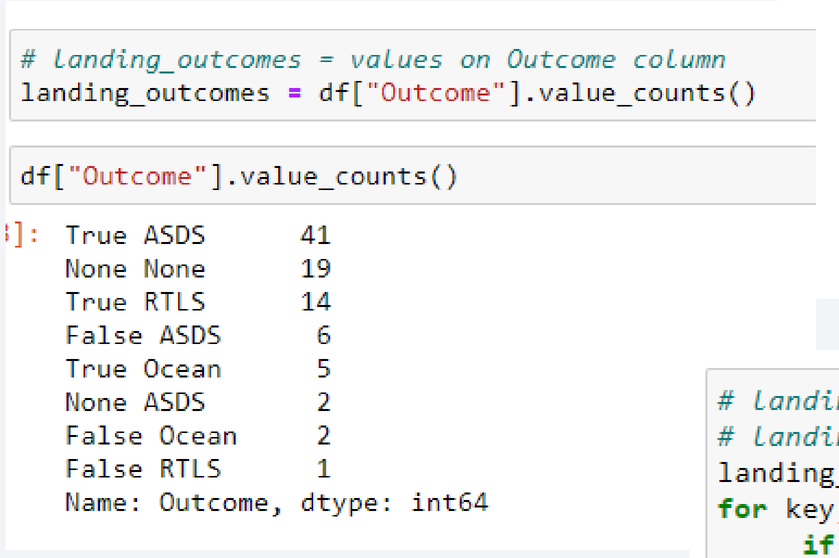
```
# Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

```
5]: CCAFS SLC 40    55
    KSC LC 39A      22
    VAFB SLC 4E     13
    Name: LaunchSite, dtype: int64
```



```
# Apply value_counts on Orbit column
df["Orbit"].value_counts("Orbit")
```

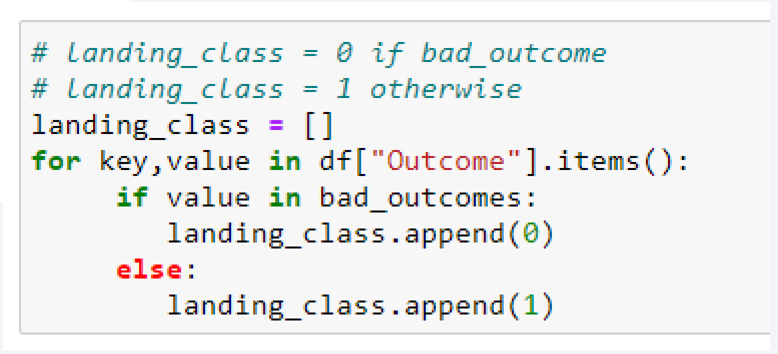
```
6]: GTO      0.300000
    ISS      0.233333
    VLEO     0.155556
    PO       0.100000
    LEO      0.077778
    SSO      0.055556
    MEO      0.033333
    HEO      0.011111
    GEO      0.011111
    SO       0.011111
    ES-L1    0.011111
    Name: Orbit, dtype: float64
```



```
# landing_outcomes = values on Outcome column
landing_outcomes = df["Outcome"].value_counts()
```

```
df["Outcome"].value_counts()
```

```
]: True ASDS      41
   None None      19
   True RTLS      14
   False ASDS      6
   True Ocean      5
   None ASDS       2
   False Ocean     2
   False RTLS      1
   Name: Outcome, dtype: int64
```



```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
       landing_class.append(0)
    else:
       landing_class.append(1)
```

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts
  - Scatter Graphs show how much one variables is affected by another. The relationship between two variables is called their correlation. Scatter plots usually consist of a larger body of data. Fight Number VS. Payload Mass, Fight Number VS. Launch Site, Payload VS. Launch Site, Orbit VS. Fight Number, Payload VS. Orbit Type, Orbit VS. Payload Mass
  - Bar Graph makes it easy to compare sets of data between different groups at a glance. The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time. Mean VS. Orbit
  - Line Graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded.
- https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - Connect to the database
  - Display the names of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string 'CCA'
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first successful landing outcome in ground pad was achieved.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failure mission outcomes
  - List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
  - List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  - Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
- https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
  - Mark all launch sites on a map: we took the latitude and longitude coordinates at each launch site and added a Circle Marker around each launch site with a label of the name of the launch site
  - Mark the success/failed launches for each site on the map: we assigned the dataframe launch_outcomes (failures, success) to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()
  - Calculate the distances between a launch site to its proximities: using Haversine's formula we calculated the distance from the Launch Site to various landmakers to find various trends about what is around the Launch Site to measure patterns. Lines are dran on the map to measure distance to lanmarks.
- https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
  - Graphs: pie chart showing the total launches by a certain site/all sites, display relative proportions of multiple classes of data, size of the circle can be made proportional to the total quantity it represents
  - Scatter Graph showing the relationship with outcome and payload mass (kg) for the different booster versions, it is the best method to show you a non-linear pattern, the range of data flow, observation and reading are straightforward.
- https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
  - Building model: load our dataset into Numpy and Pandas, transform data, split our data into training and test data sets, check how many test samples we have, decide which type of machine learning algorithms we want to use, set our parameters and algorithms to GridsearchCV, fit our datasets into the GridsearchCV objects and train our dataset.
  - Evaluating model: check accuracy for each model, get tuned hyperparameters for each type of algorithms, plot confusion matrix
  - Improving model: feature engineering, algorithm tuning
  - Finding the best performing classification model: the model with the best accuracy score wins the best performing model, in the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.
- https://github.com/Simon170624/Winning-Space-Race-with-Data-Science

Results

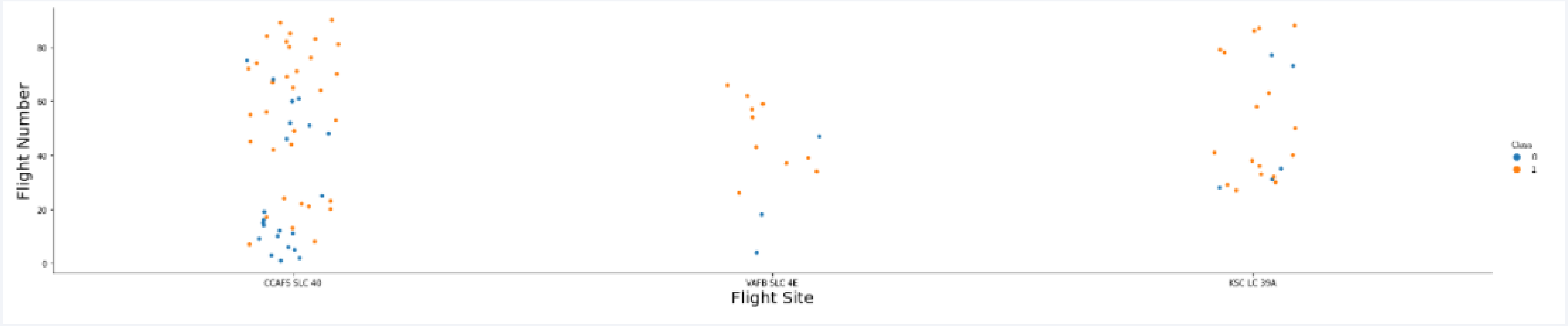
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

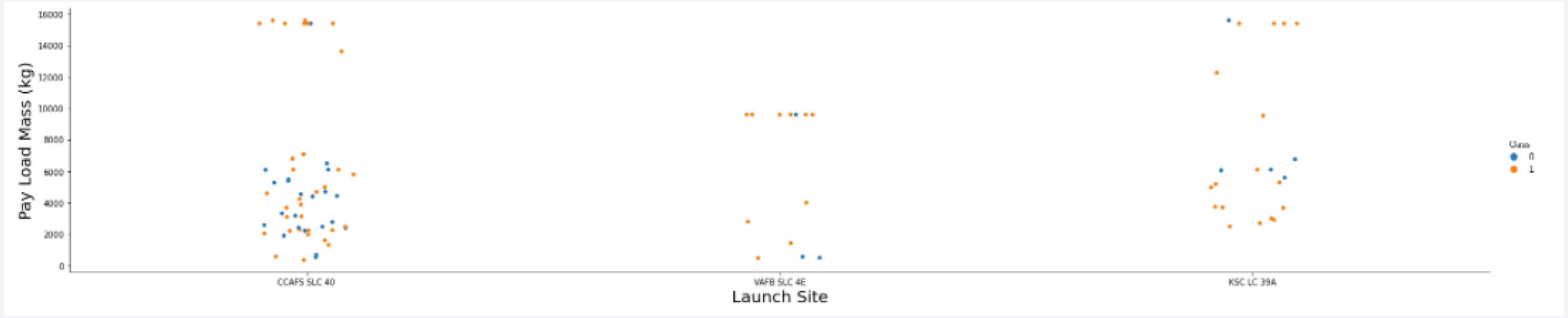
- Show a scatter plot of Flight Number vs. Launch Site



- Show the screenshot of the scatter plot with explanations
  - The more amount of flights at a launch site the greater the success rate at a launch site.

Payload vs. Launch Site

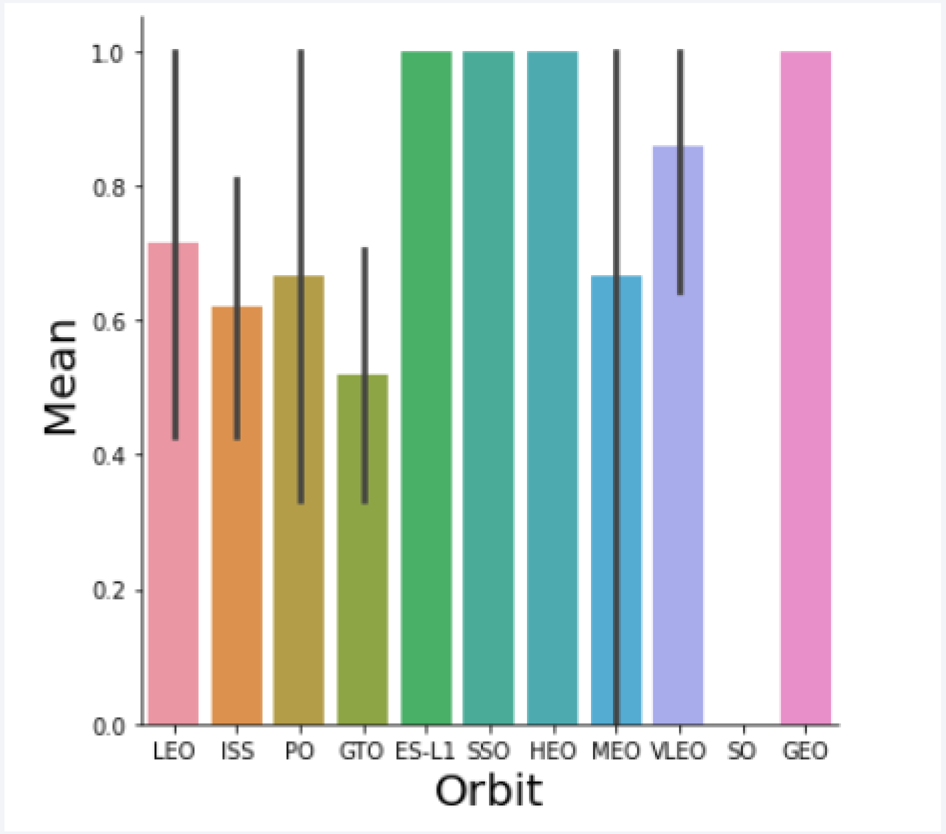
- Show a scatter plot of Payload vs. Launch Site



- Show the screenshot of the scatter plot with explanations
  - The greater the payload mass for launch stie CCAFS SLC 40 the higher the success rate for the rocket. There is not quite a clear pattern to be found using this visualization to make a decision if the launch site is dependant on pay load mass for a success launch.

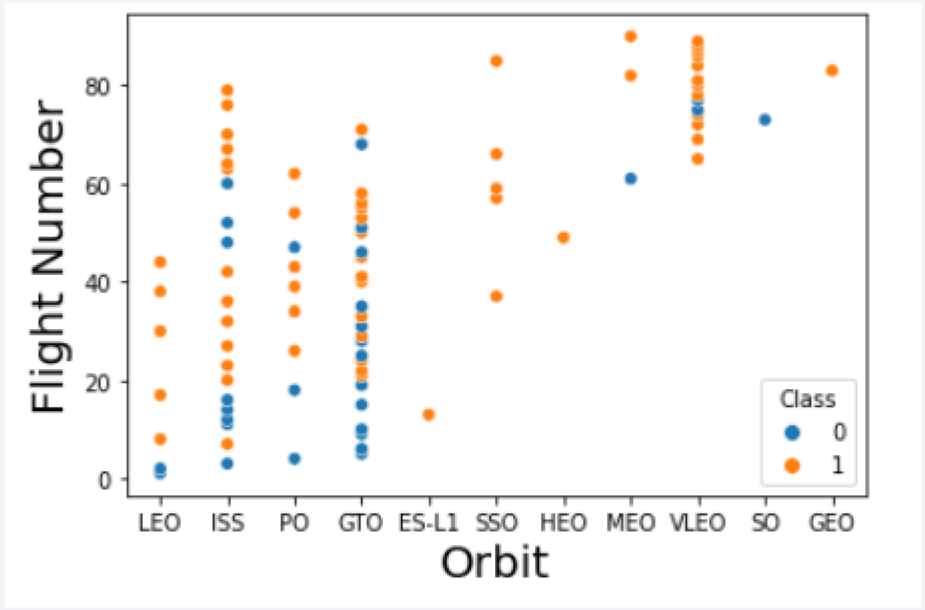
Success Rate vs. Orbit Type

- Show a bar chart for the success rate of each orbit type
- Show the screenshot of the scatter plot with explanations
  - Orbit GEO, HEO, SSO, ES-L1 has the best success rate.



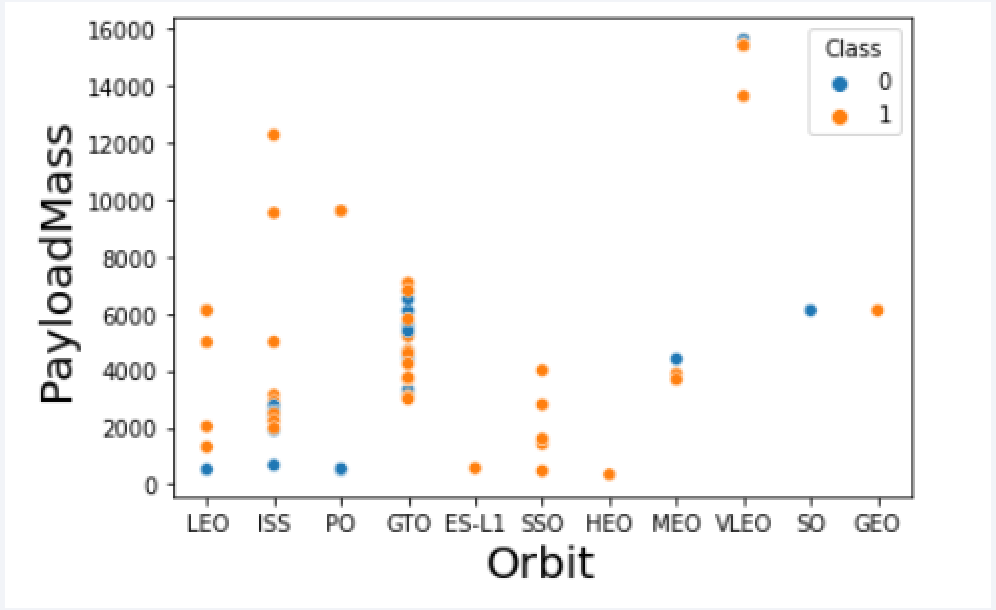
Flight Number vs. Orbit Type

- Show a scatter point of Flight number vs. Orbit type
- Show the screenshot of the scatter plot with explanations
    - You should see that in the LEO orbit the success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



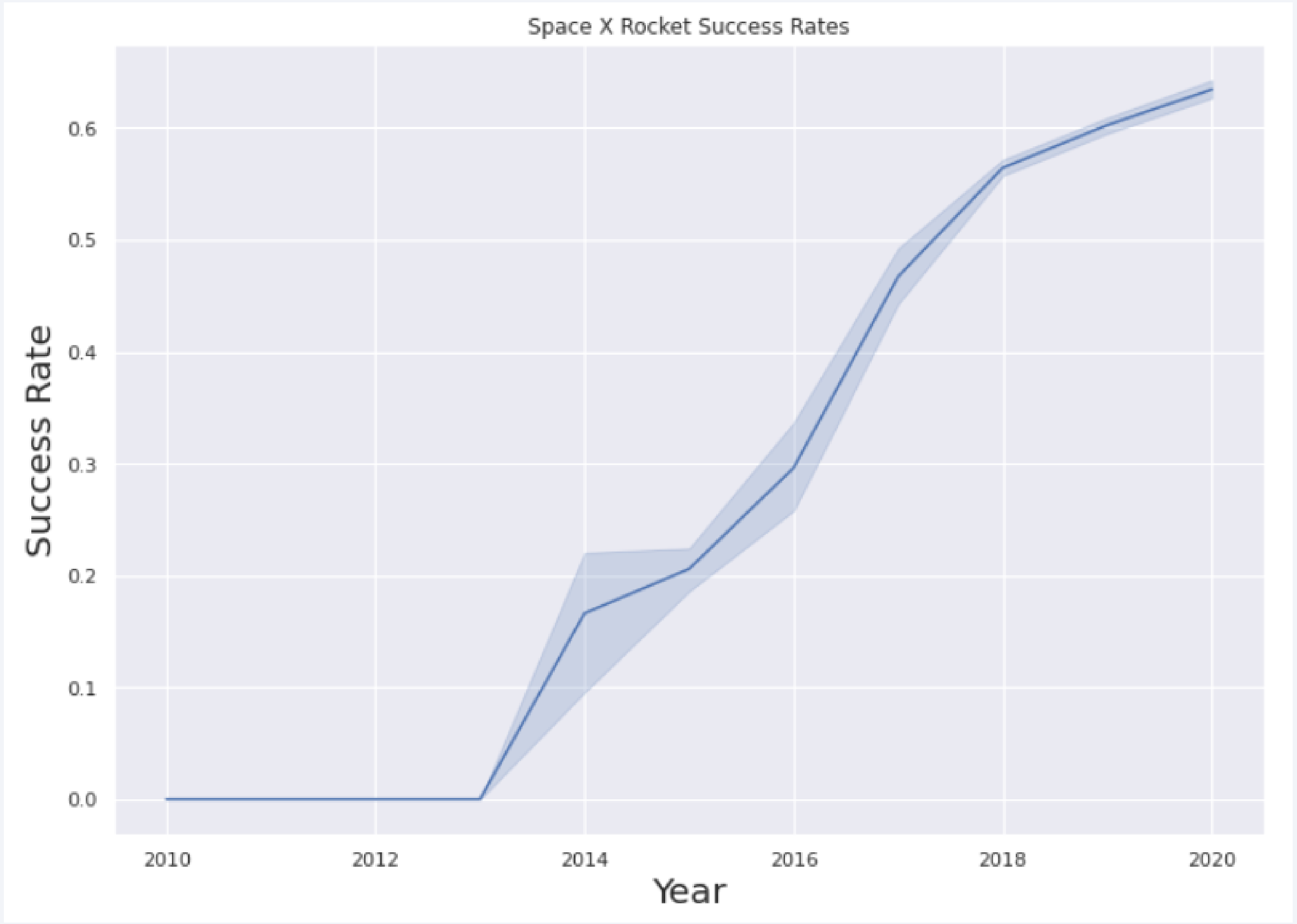
Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type
- Show the screenshot of the scatter plot with explanations
    - You should observe that heavy payloads have a negative influence on GTO orbits and positive on GTO and polar LEO (ISS) orbits.



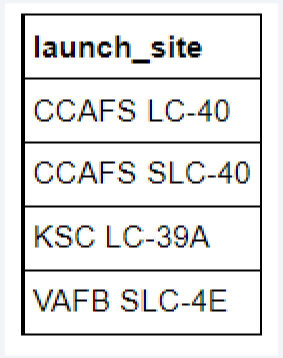
Launch Success Yearly Trend

- Show a line chart of yearly average success rate
- Show the screenshot of the scatter plot with explanations
  - You can observe that the success rate since 2013 kept increasing till 2020



All Launch Site Names

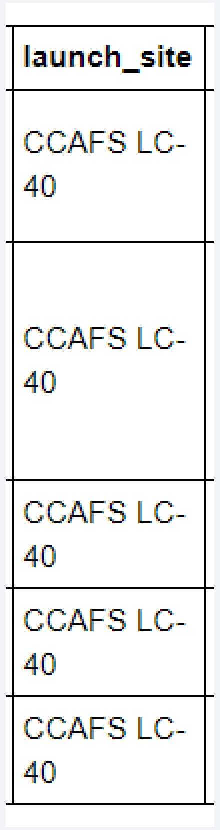
- Find the names of the unique launch sites
    - %sql select Unique(LAUNCH_SITE) from SPACEXDATASET;
- Present your query result with a short explanation here
    - Using the word DISTINCT in the query means that it will only show unique in the LAUNCH_SITE column from SPACEXDATASET



launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'

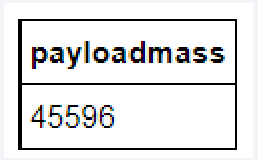
- Find 5 records where launch sites begin with `CCA`
  - %sql SELECT * from SPACEXDATASET where (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
- Present your query result with a short explanation here
  - Using the word LIMIT 5 in the query means that it will only show 5 records from SPACEXDATASET and LIKE keyword has a wild card with the words ‘CCA%’ the percentage in the end suggests that the launch_site name must start with CCA



launch_site
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- Calculate the total payload carried by boosters from NASA
  - %sql select sum(PAYLOAD_MASS__KG_) as payloadmass from SPACEXDATASET WHERE customer = 'NASA (CRS)';
- Present your query result with a short explanation here
  - Using the function SUM summates the total in the column payload_mass_kg_. The where clause filters the dataset to only perform calculations on customer NASA (CRS)

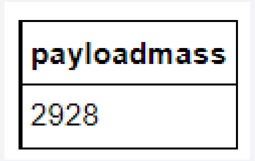


payloadmass
45596

```
%sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXDATASET where Booster_Version = 'F9 v1.1';
```

Average Payload Mass by F9 v1.1

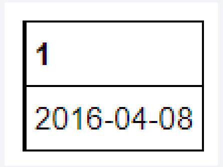
- Calculate the average payload mass carried by booster version F9 v1.1
  - %sql select avg(PAYLOAD_MASS__KG_) as payloadmass from SPACEXDATASET where Booster_Version = 'F9 v1.1';
- Present your query result with a short explanation here
  - Using the function AVG works out the average in the column PAYLOAD_MASS__KG_. The where clause filters the dataset to only perform calculate on Booster_Version = 'F9 v1.1'.



payloadmass
2928

First Successful Ground Landing Date

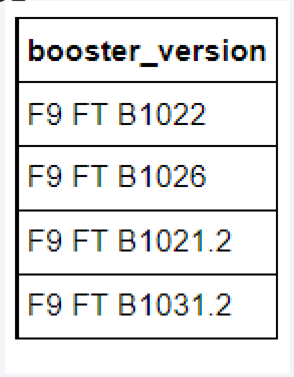
- Find the dates of the first successful landing outcome on ground pad
  - %sql select min(DATE) from SPACEXDATASET where landing__outcome = 'Success (drone ship)';
- Present your query result with a short explanation here
  - Using the function MIN works out the minimum data in the column DATE, the where clause filters the dataset to only perform calculates on landing__outcome = 'Success (drone ship)'.



1
2016-04-08

Successful Drone Ship Landing with Payload between 4000 and 6000

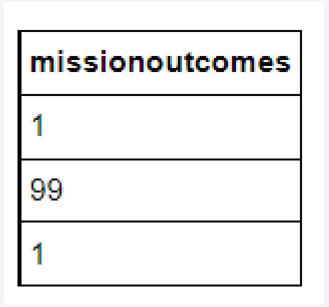
- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000
  - %sql select Booster_Version from SPACEXDATASET where landing__outcome = 'Success (drone ship)' and payload_mass__kg_ > 4000 AND payload_mass__kg_ < 6000;
- Present your query result with a short explanation here
  - Selecting only Booster_Version, the where filters the dateset to landing__outcome = 'Success (drone ship)', the AND clause specifies additional filter conditions payload_mass__kg_ > 4000 AND payload_mass__kg_ < 6000.



booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

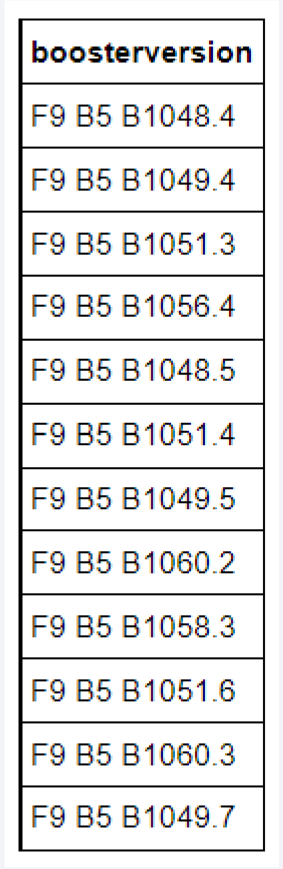
- Calculate the total number of successful and failure mission outcomes
  - %sql select count(MISSION_OUTCOME) as missionoutcomes from SPACEXDATASET GROUP BY MISSION_OUTCOME;
- Present your query result with a short explanation here
  - Count shows the total number, group by shows different kinds.



missionoutcomes
1
99
1

Boosters Carried Maximum Payload

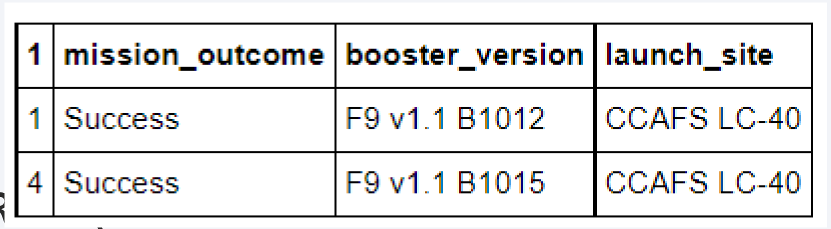
- List the names of the booster which have carried the maximum payload mass
  - %sql select BOOSTER_VERSION as boosterversion from SPACEXDATASET where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXDATASET);
- Present your query result with a short explanation here
  - Subquery put the maximum payload mass, select BOOSTER_VERSION from SPACEXDATASET, and where evaluates PAYLOAD_MASS__KG_= max value.



boosterversion
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

2015 Launch Records

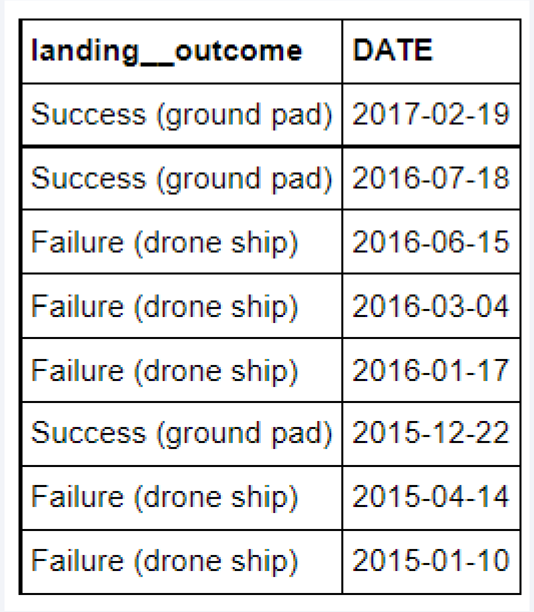
- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015
    - %sql SELECT MONTH(DATE),MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACEXDATASET where landing__outcome = 'Failure (drone ship)' and EXTRACT(YEAR FROM DATE)='2015';
- Present your query result with a short explanation here
    - where filters landing__outcome = 'Failure (drone ship)' and EXTR
      DATE)='2015', and select shows information



1	mission_outcome	booster_version	launch_site
1	Success	F9 v1.1 B1012	CCAFS LC-40
4	Success	F9 v1.1 B1015	CCAFS LC-40

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order
  - %sql SELECT LANDING__OUTCOME, DATE FROM SPACEXDATASET WHERE (landing__outcome IN ('Failure (drone ship)', 'Success (ground pad)')) AND (DATE BETWEEN '2010-06-04' AND '2017-03-20') ORDER BY DATE DESC;
- Present your query result with a short explanation here
  - Select LANDING__OUTCOME and DATE columns, from SPACEXDATASET, Where filters date between 2010-06-04 and 2017-03-20 and landing__outcome is Failure (drone ship) or Success (ground pad).



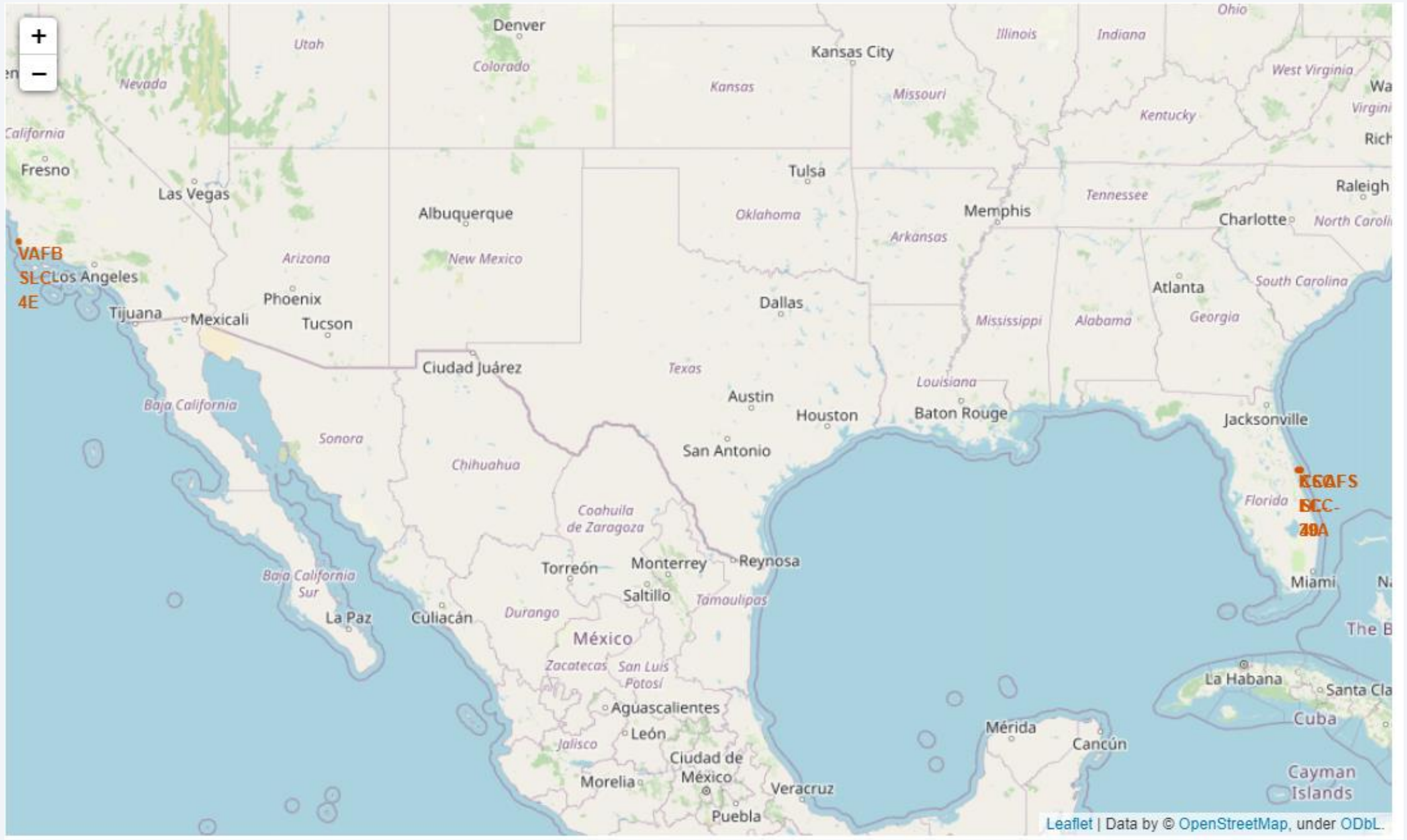
landing__outcome	DATE
Success (ground pad)	2017-02-19
Success (ground pad)	2016-07-18
Failure (drone ship)	2016-06-15
Failure (drone ship)	2016-03-04
Failure (drone ship)	2016-01-17
Success (ground pad)	2015-12-22
Failure (drone ship)	2015-04-14
Failure (drone ship)	2015-01-10

Section 4

Launch Sites Proximities Analysis

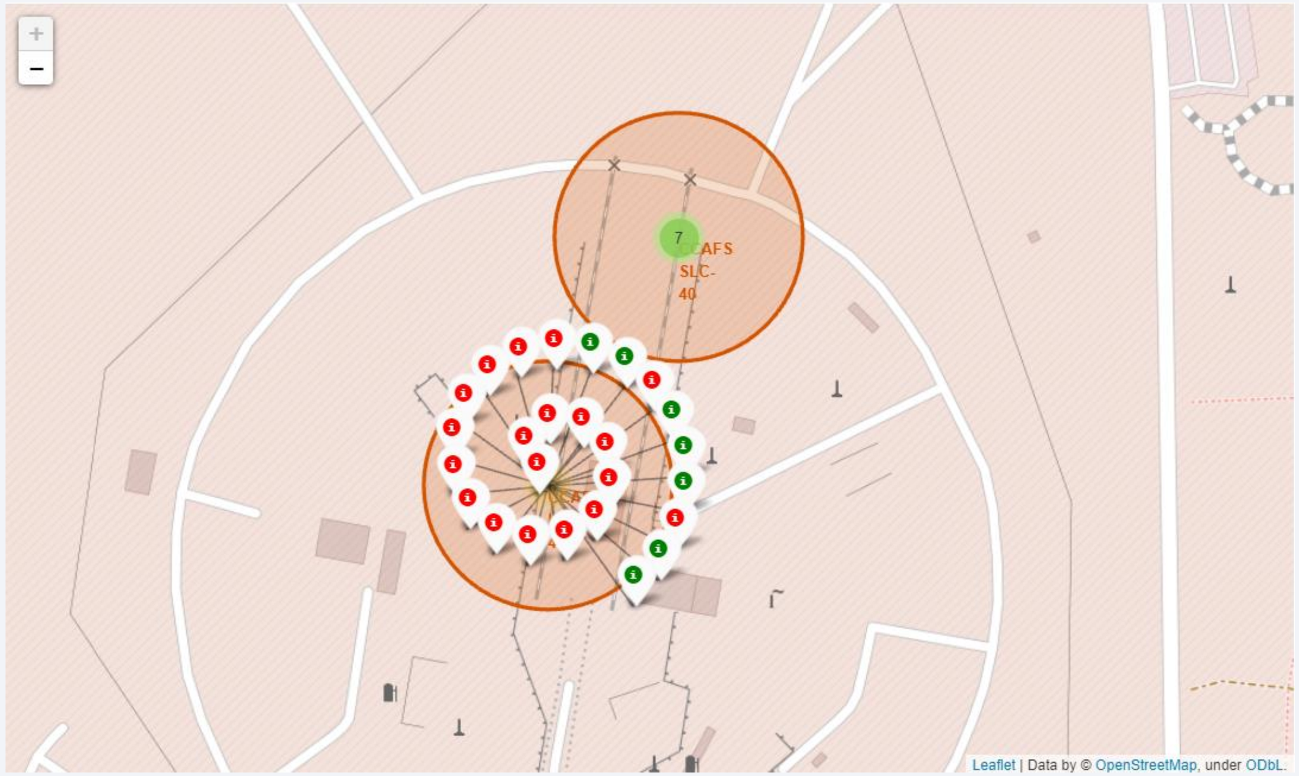
Mark all launch sites on a map

- Explore the generated folium map and make a proper screenshot to include all launch sites' location markers on a global map
- Explain the important elements and findings on the screenshot
  - SpaceX launch sites are in America coasts, Florida, and California.



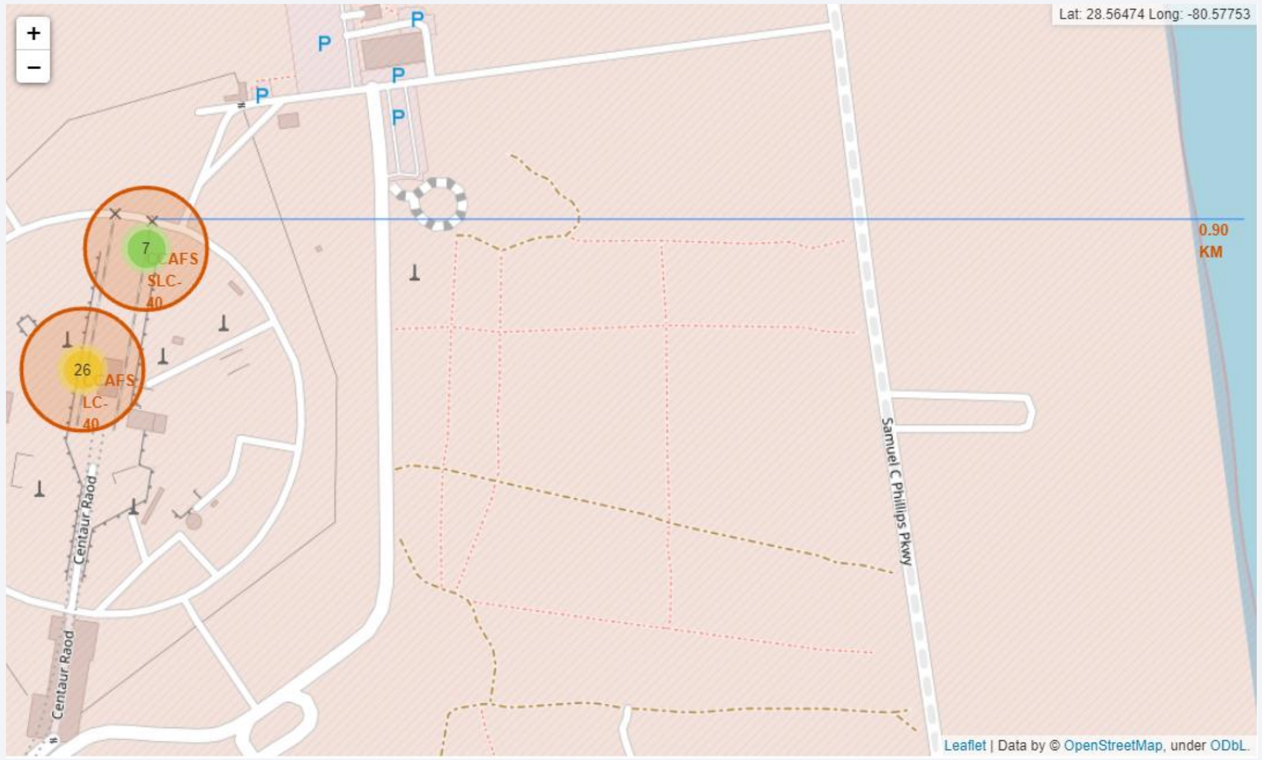
Mark the success/failed launches for each site on the map

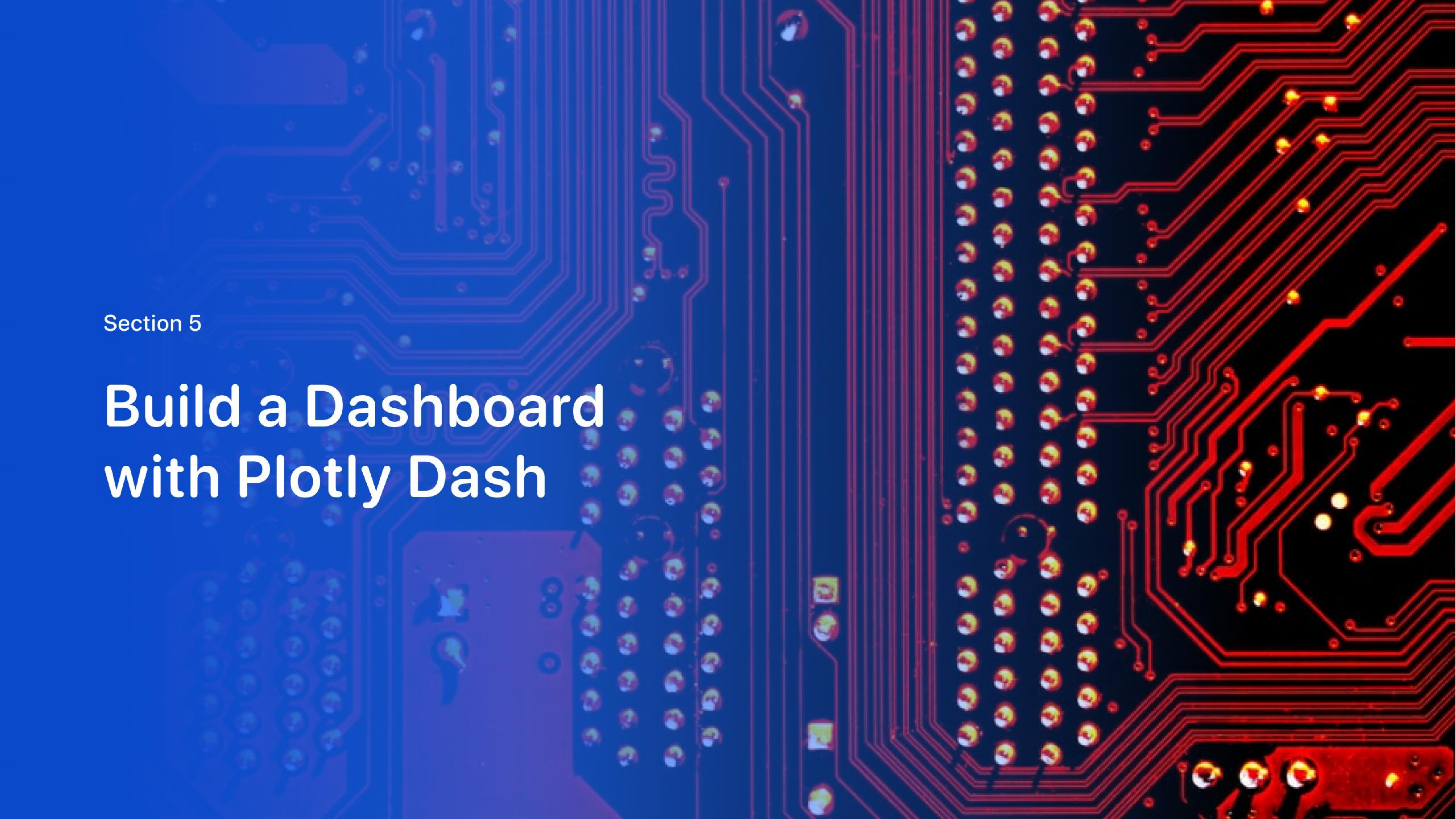
- Explore the folium map and make a proper screenshot to show the color-labeled launch outcomes on the map
- Explain the important elements and findings on the screenshot
    - Green shows successful and red shows failures.



Calculate the distances between a launch site to its proximities

- Explore the generated folium map and show the screenshot of a selected launch site to its proximities such as railway, highway, coastline, with distance calculated and displayed
- Explain the important elements and findings on the screenshot
    - For example, showing the distance to coasts

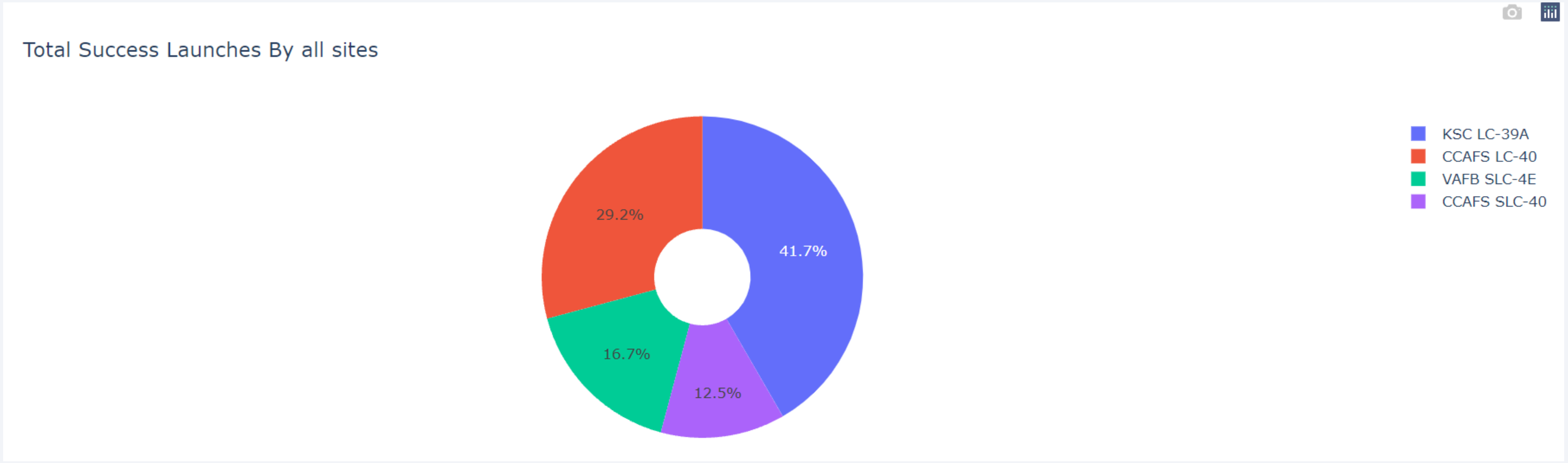

Lat: 28.56474 Long: -80.57753
0.90 KM
7
CCAFS SLC-40
26
CCAFS LC-40
Centaur Raod
Samuel C Phillips Pkwy
Leaflet | Data by © OpenStreetMap, under ODbL.



Section 5

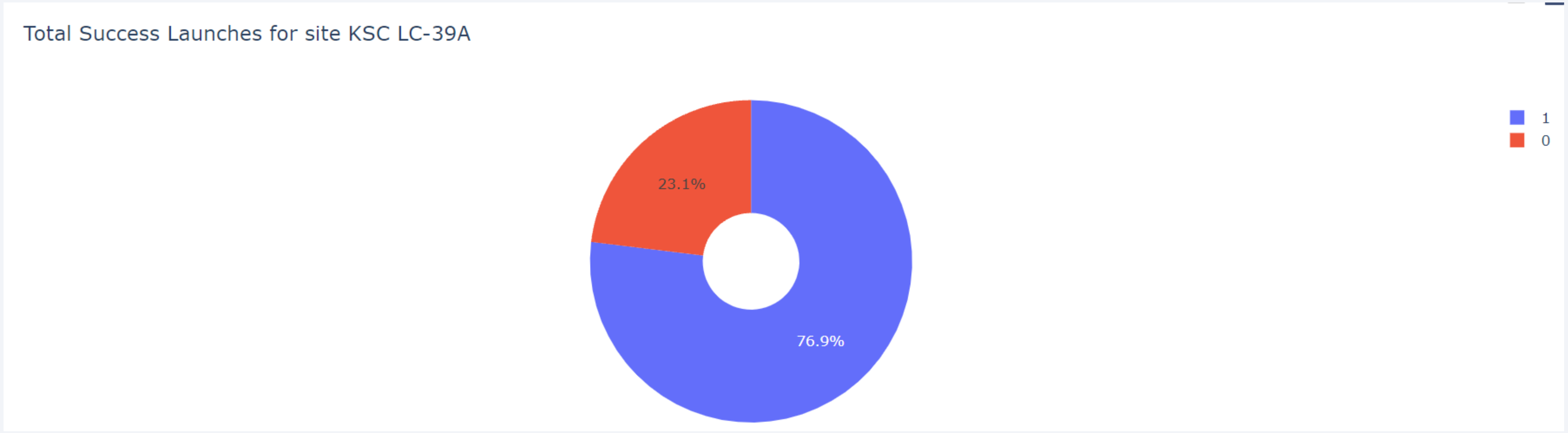
Build a Dashboard with Plotly Dash

Total Success Launches By all sites



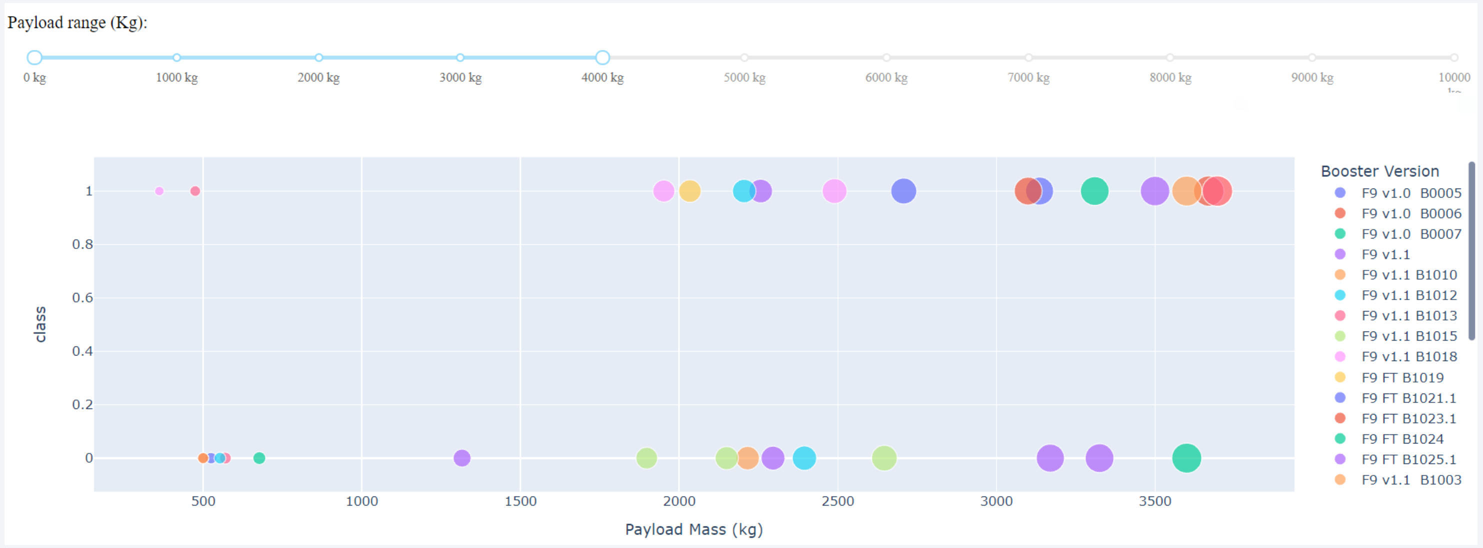
- Show the screenshot of launch success count for all sites, in a piechart
- Explain the important elements and findings on the screenshot
    - we can see that KSC LC-39A had the most successful launches.

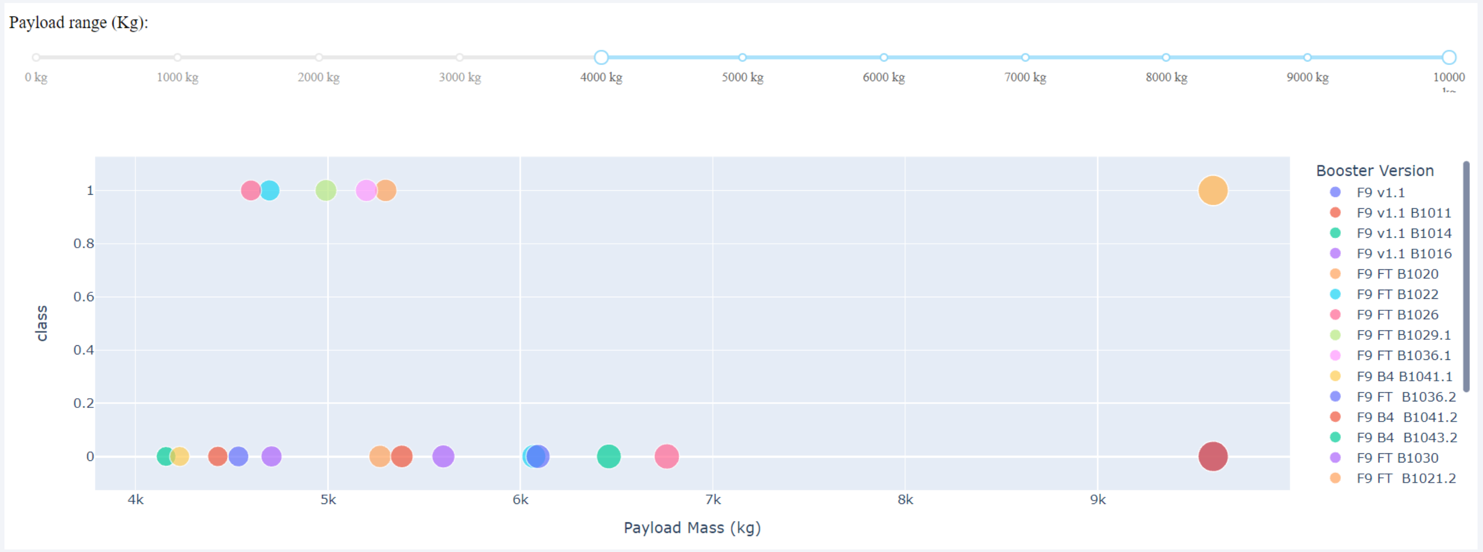
Total Success Launches for site KSC LC-39A



- Show the screenshot of the piechart for the launch site with highest launch success ratio
- Explain the important elements and findings on the screenshot
  - KSC LC-39A achieved a 76.9% success rate while a 23.1% failure rate.

Payload VS. Mass with different paylaod





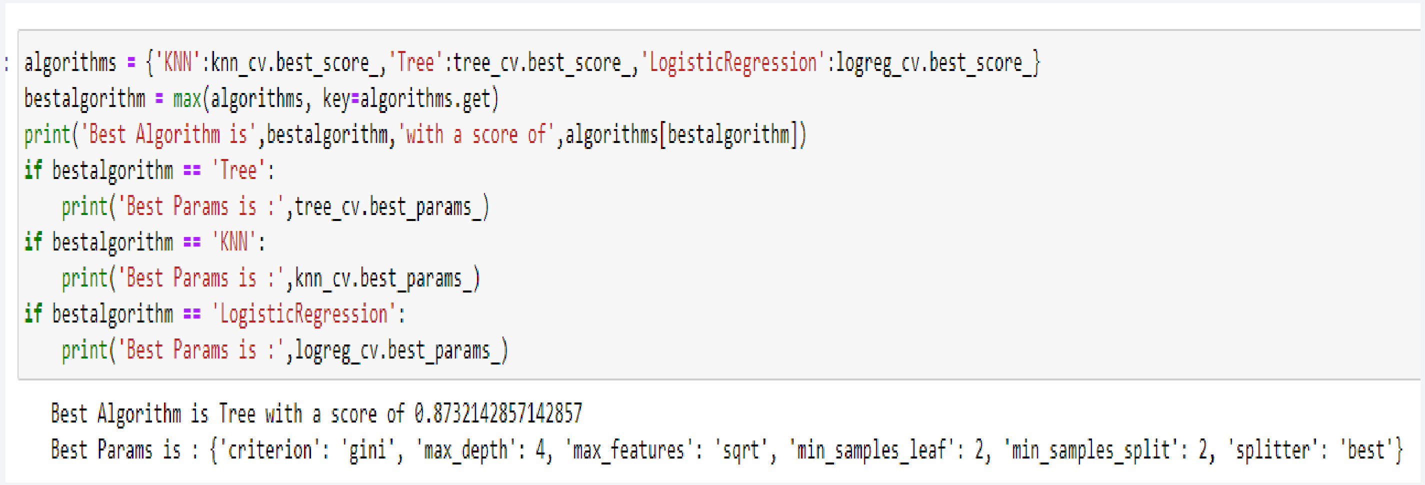
- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.
  - The success rates for low weighted payloads is higher than the heavy.

Section 6

Predictive Analysis (Classification)

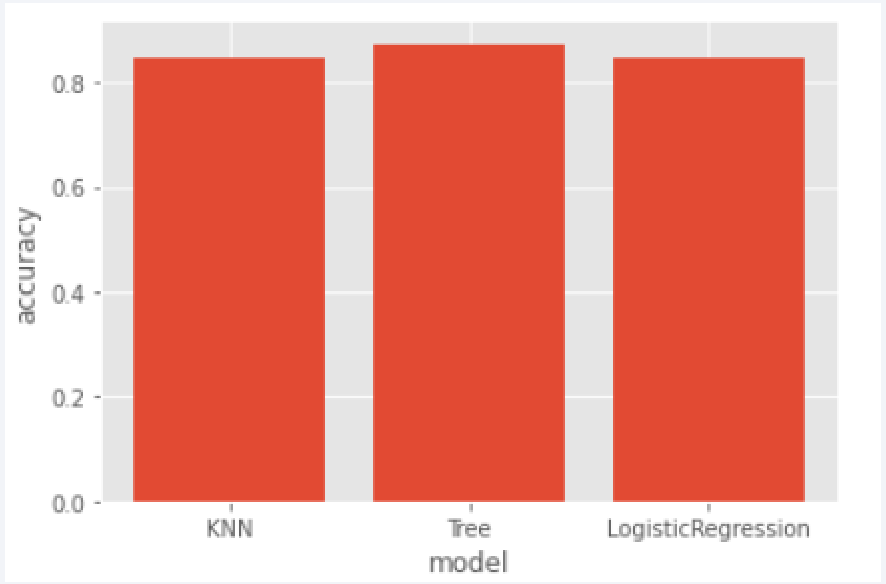
Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart
- Find which model has the highest classification accuracy



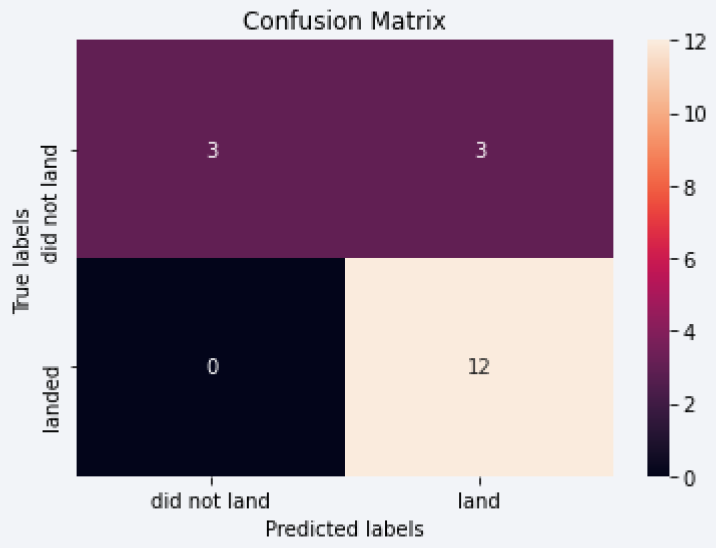
```
algorithms = {'KNN':knn_cv.best_score_,'Tree':tree_cv.best_score_,'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

```
Best Algorithm is Tree with a score of 0.8732142857142857
Best Params is : {'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'best'}
```



Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation
  - Examining the confusion matrix, we see that Tree can distinguish between the different classes. We see the major problem is false positives.



Conclusions

- The Tree is the best for machine learning for this dataset
- Low weighted payloads perform better than the heavier payloads
- The success rates for SpaceX launches is proportional time in years they will eventually perfect the launches
- We can see that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO, HEO, SSO, ES-L1 has the best Success Rate.

Appendix

- Haversine formula
- ADGGoogleMaps Module
- Module sqlserver
- PythonAnywhere 24/7 dashboard

Thank you!