

Data Science Capstone project

Space X Falcon 9 First Stage Landing Prediction

Bao Pham

09/01/2021

Outline



- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion
- Appendix

Executive Summary



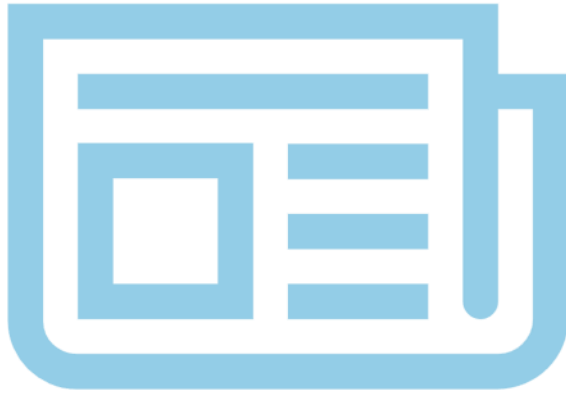
- This project involved studying rocket launches of SpaceX company which is founded by Billionaire industrialist Allon Musk. The first flight of a reusable Falcon 9 cost SpaceX less than 15% of the launch cost, and the reusable first stage costs approximately 12% of the cost to create. At the end, this project can predict the price of each launch, whether SpaceX reuse first stage of their rockets, and more by building machine learning models from collected SpaceX's data.
- Decision Tree is the best model which return 94.4% accuracy when predicting whether a rocket lands or not

Introduction



- The commercial space age is here, companies are making space travel affordable for everyone. Virgin Galactic is providing suborbital spaceflights. Rocket Lab is a small satellite provider. Blue Origin manufactures sub-orbital and orbital reusable rockets. Of course, the cost of these activities is still very expensive, which approximately 165 million for each launch. However, SpaceX, which is considered as the most successful personal space company, advertises their Falcon 9 rocket launches with a cost of 62 million dollars, which is much cheaper than others. The main reason for this is because SpaceX can reuse the first stage. Reusable rockets are not just beneficial for SpaceX, they are beneficial for our space program as a whole. This means less fuel consumption for manned spacecrafts, cheaper launches and more usable space for technology development.
- Therefore, if we can determine if the first stage of SpaceX Rockets will land, we can determine whether SpaceX can recover the first stage which is leading to be able to predict the cost of a launch.

Methodology



- Data collection methodology:
 - Describe how data were collected
- Perform data wrangling
 - Describe how data were processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
 - How to build, tune, evaluate classification models

Methodology

Data collection

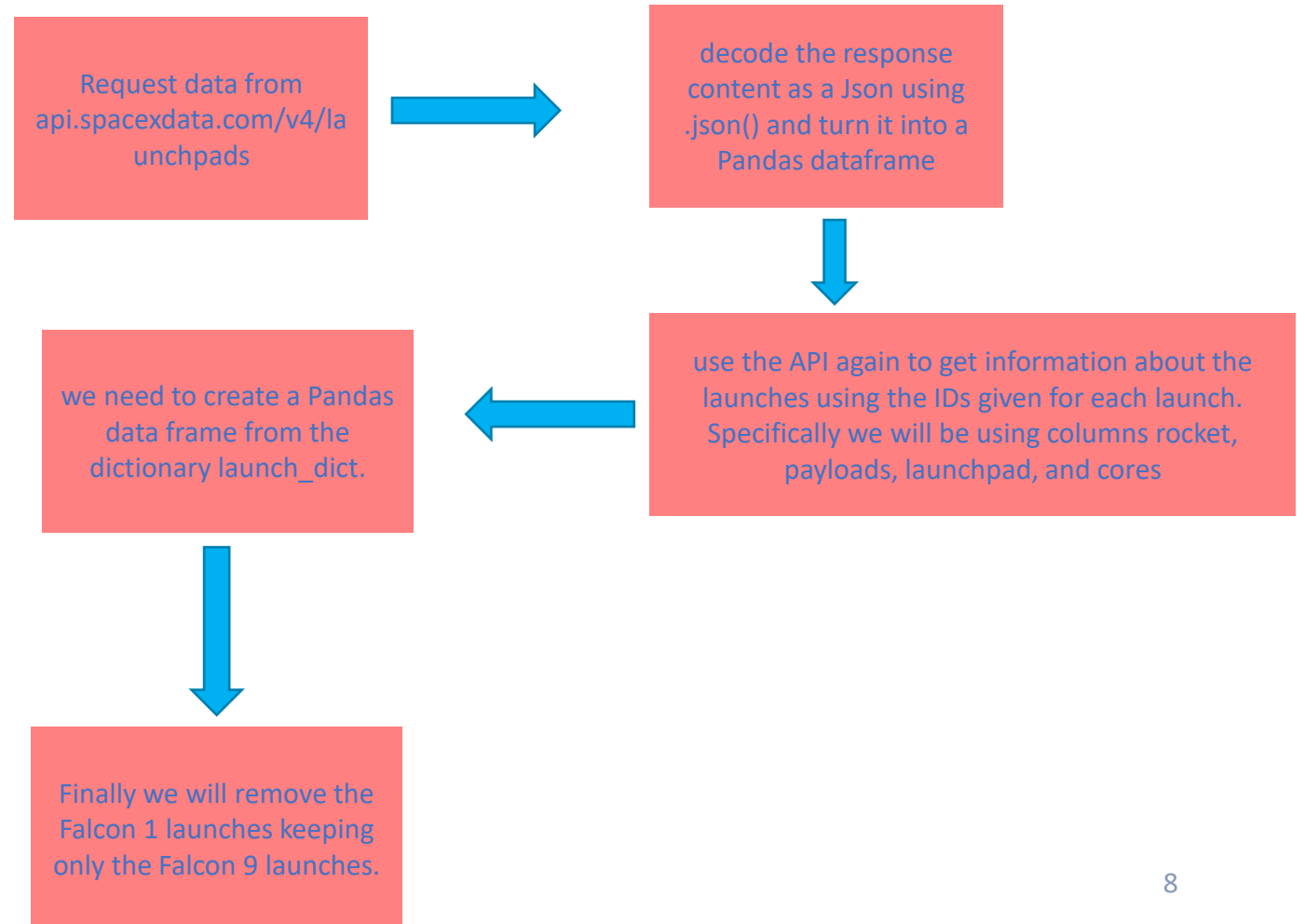
- In this project, we worked with SpaceX launch data which is gathered from SpaceX REST API.

Data collection – SpaceX API

GitHub URL of the completed SpaceX API
calls notebook:

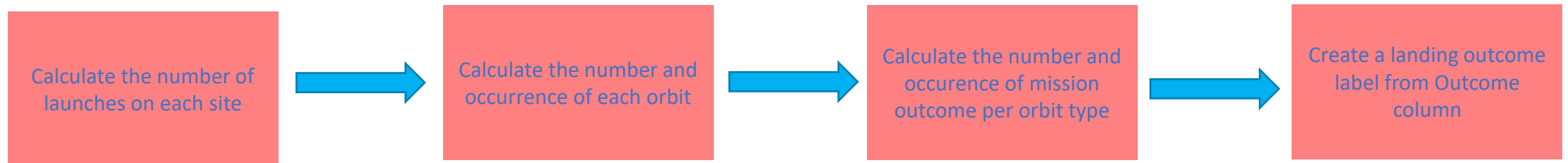
<https://github.com/poker4196/IBMFinal/blob/master/jupyter-labs-spacex-data-collection-api.ipynb.ipynb>

flowchart of SpaceX API calls



Data wrangling

In this lab, we worked with SpaceX launch data which is prepared from last session. Then we perform exploratory Data Analysis and determine Training Labels



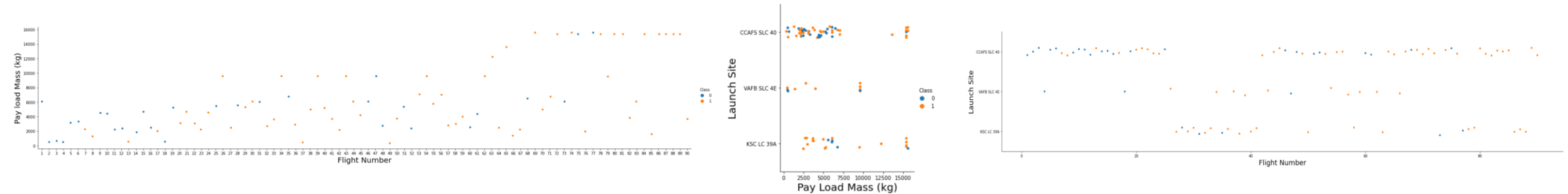
GitHub URL of the completed SpaceX Data wrangling notebook:

<https://github.com/poker4196/IBMFinal/blob/master/labs-jupyter-spacex-Data-wrangling.ipynb>

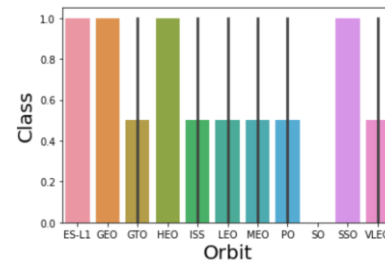
EDA with data visualization

<https://github.com/poker4196/IBMFinal/blob/master/jupyter-labs-eda-dataviz.ipynb>

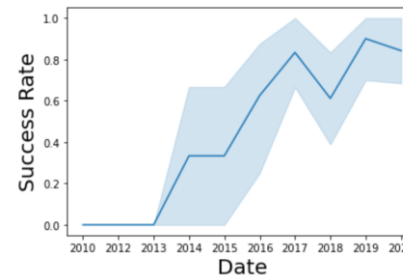
- 1. I use catplot to show relationship of the FlightNumber vs. PayloadMass and, or FlightNumber vs LaunchSite, overlay the outcome of the launch. Since there are 3 variables, catplot is a good choice



- 2. I use bar plot to check if there are any relationship between success rate and orbit type. Since there are many types of orbit type, a bar chart fit it grades.



- 3. I plot a line chart with x axis to be year and y axis to be average success rate, to get the average launch success trend. Since to determine a trend, a line char is useful



EDA with SQL

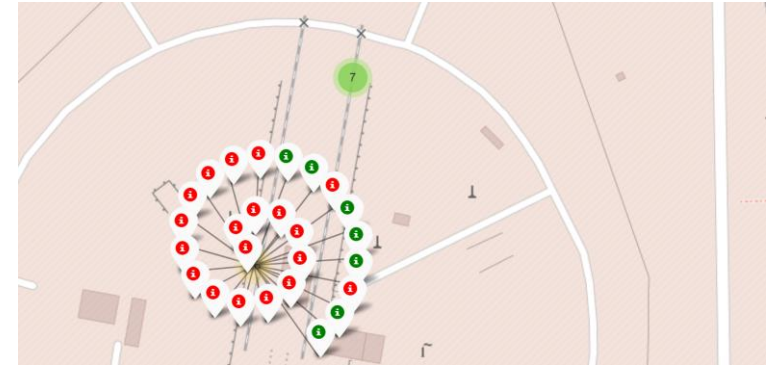
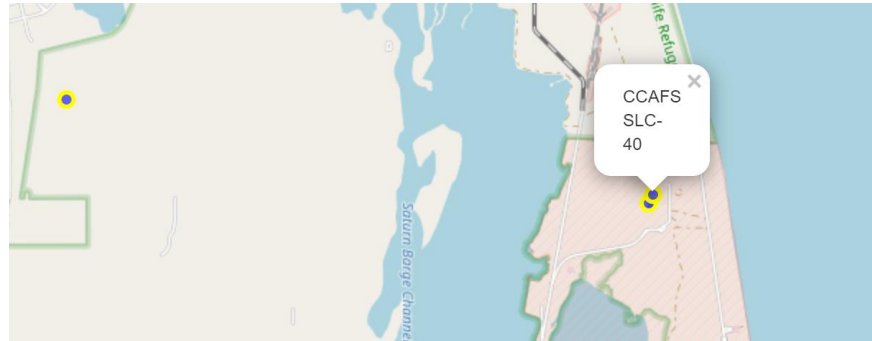
<https://github.com/poker4196/IBMFinal/blob/master/jupyter-labs-eda-sql-coursera.ipynb>

- In this lab:
 - Understand the SpaceX DataSet
 - Load the dataset into the corresponding table in a Db2 database
 - Execute SQL queries to answer assignment questions, such as:
 - 1. Display the names of the unique launch sites in the space mission
 - 2. Display 5 records where launch sites begin with the string 'CCA'
 - 3. Display the total payload mass carried by boosters launched by NASA (CRS)
 - 4. More and more

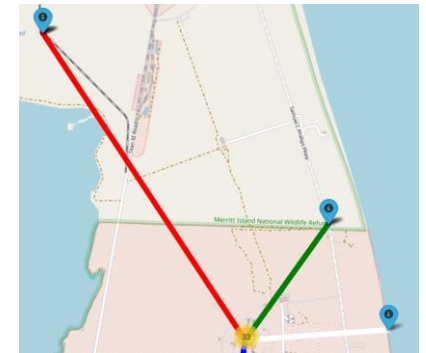
Build an interactive map with Folium

https://github.com/poker4196/IBMFinal/blob/master/lab_jupyter_launch_site_location.ipynb

- I added a marker to mark Nasa position, launch sites, the success/failed launches for each site on the map



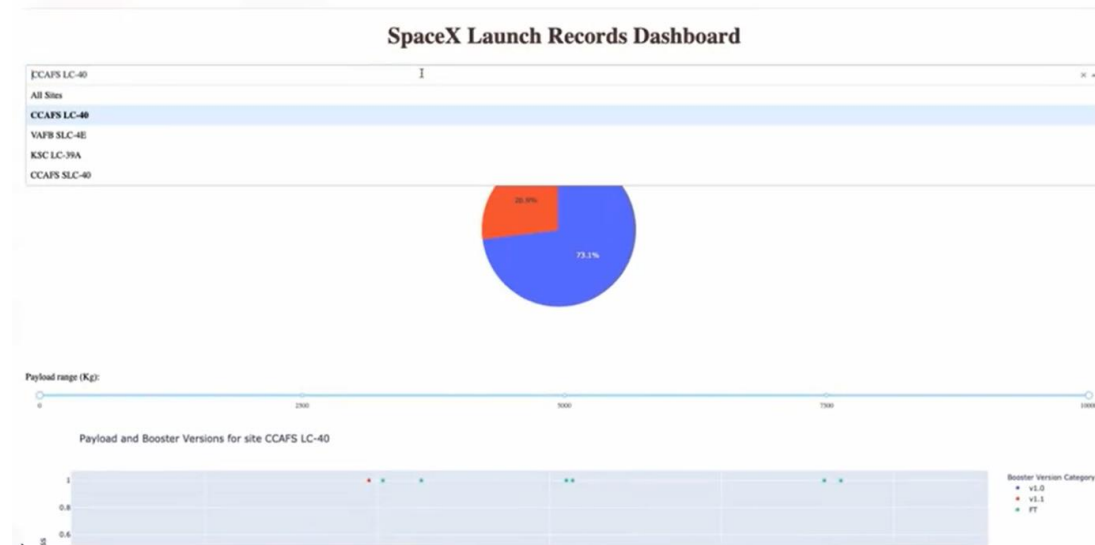
- I added PolyLines between a launch site to a point of railroad, closest city, coastline, and highway.



- The main idea of these is to determine whether the launch success rate may depend on the location and proximities of a launch site, besides many factors such as payload mass, orbit type, and so on.

Build a Dashboard with Plotly Dash

- I added to the dashboard a drop-down menu, a pie chart, and a scatter points chart

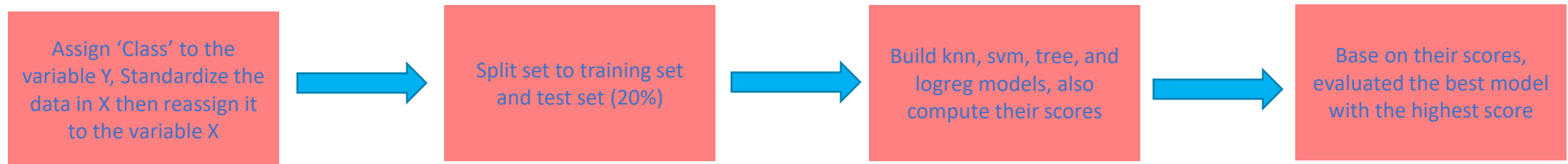


- This dashboard is to perform interactive visual analytics on SpaceX launch data in real-time. We can see the site has the largest successful launches, the highest launch success rate, the payload range(s) has the highest launch success rate, payload range(s) has the lowest launch success rate, and more.

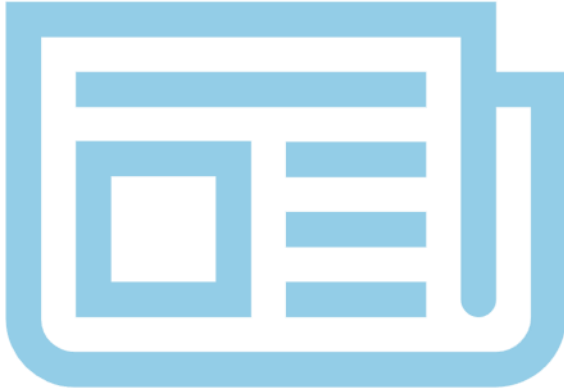
Predictive analysis (Classification)

https://github.com/poker4196/IBMFinal/blob/master/SpaceX_Machine_Learning_Prediction_Part_5.ipynb

- I firstly assigned X and Y(class) and split them to train and test set. Then use train set to build model k nearest neighbors, support vector machine, logistic regression, and decision tree. Then I based on their scores to find out the best model. My highest score is 0.83333333333333334. Best models are k nearest neighbors, support vector machine(kernel is sigmoid), logistic regression.



Results

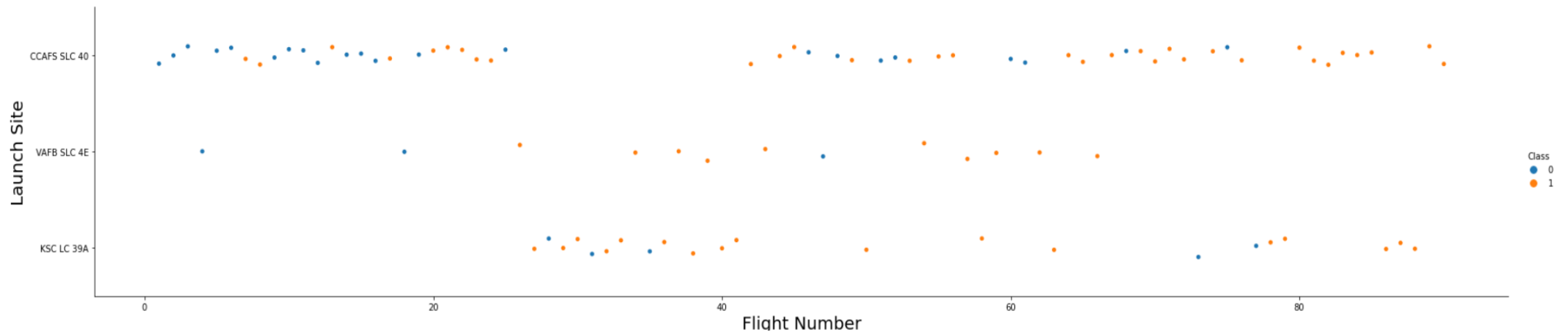


- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

EDA with Visualization

Flight Number vs. Launch Site

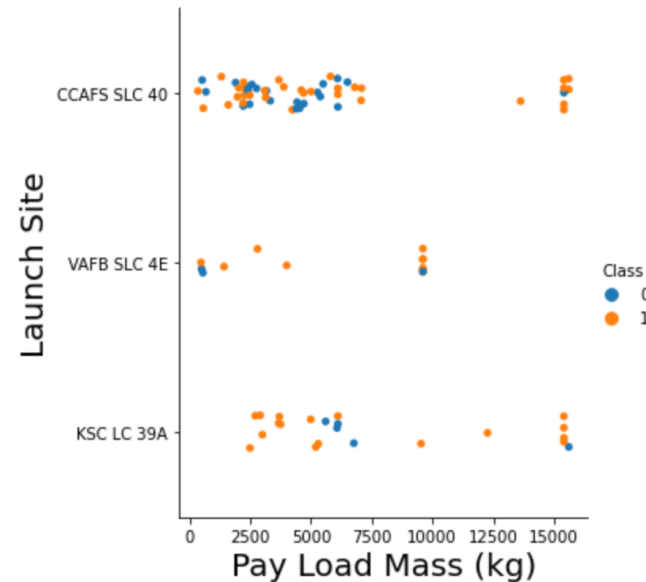
```
In [19]: # Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



We can see that launch site CCAFS SLC 40 has about 66% successful rate which is lowest, while VAFB SLC 4E has highest successful rate

Payload vs. Launch Site

```
In [20]: # Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the Launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df)
plt.xlabel("Pay Load Mass (kg)", fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```

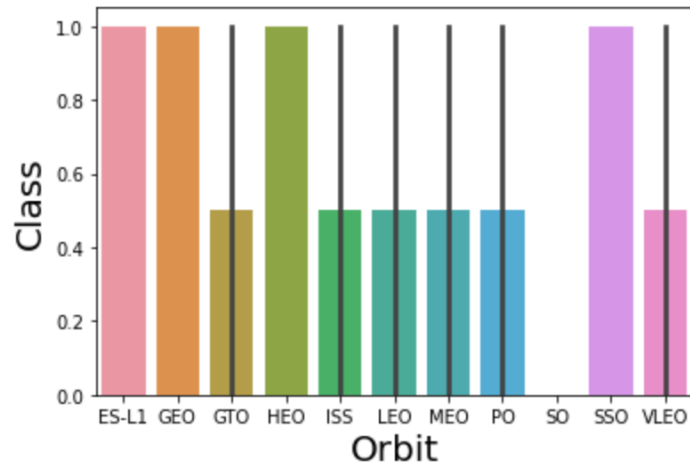


We can see that launch site CCAFS SLC 40 has about 66% successful rate which is lowest, but when its pay load mass is over 12500 kg, its rate is almost 100% which is very impressive. For lower pay load mass, launch site VAFB SLC 4E and KSC KC 39A do much better which around 78% successful rate.

Success rate vs. Orbit type

```
In [21]: # HINT use groupby method on Orbit column and get the mean of Class column
t = df.groupby(['Orbit', 'Class'])['Class'].agg(['mean']).reset_index()
sns.barplot(y="Class", x="Orbit", data=t)

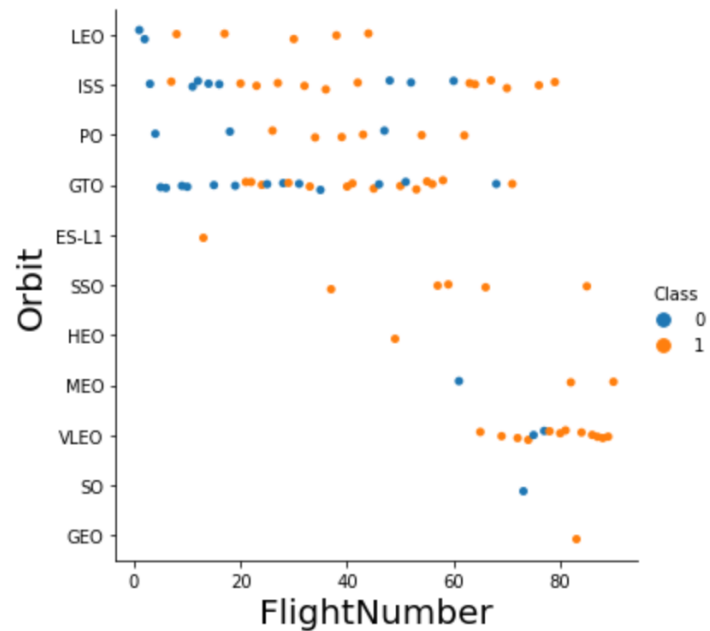
plt.xlabel("Orbit", fontsize=20)
plt.ylabel("Class", fontsize=20)
plt.show()
```



We can see that Orbit as ES-L1, GEO, HEO, SSO have highest successful rate, while Orbit SO has the lowest rate. Other Orbits have mixed rate.

Flight Number vs. Orbit type

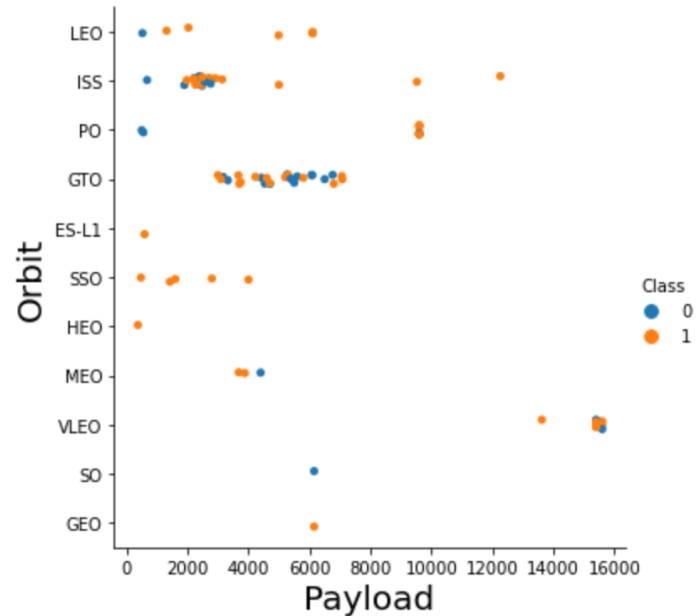
```
In [22]: # Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df)
plt.xlabel("FlightNumber", fontsize=20)
plt.ylabel("Orbit", fontsize=20)
plt.show()
```



We can see that Orbit GTO is increasing its successful rate while trying more flight number. Orbit SO has only fail result. The more Flight they take, the more successful rate is.

Payload vs. Orbit type

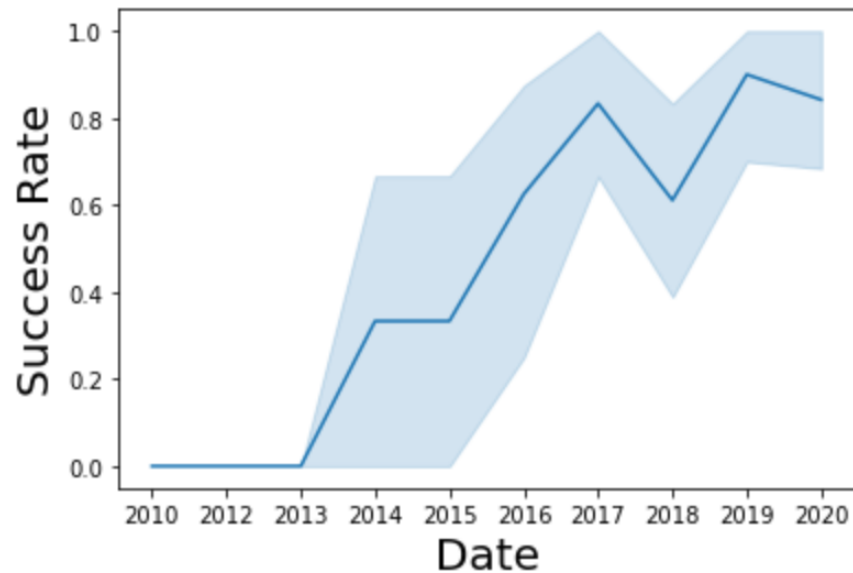
```
In [23]: # Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df)
plt.xlabel("Payload", fontsize=20)
plt.ylabel("Orbit", fontsize=20)
plt.show()
```



We can see that Orbit GTO has mixed result, even with high payload or not. Other Orbits shows that they have good results with high payload.

Launch success yearly trend

```
In [31]: sns.lineplot(data=df1, x="Date", y="Class")  
plt.xlabel("Date", fontsize=20)  
plt.ylabel("Success Rate", fontsize=20)  
plt.show()
```



We can see that the more years, the higher successful rate is. Especially the rate is increasing impressively since 2013 and 2016.

EDA with SQL

All launch site names

Display the names of the unique launch sites in the space mission

In [5]: %sql SELECT DISTINCT launch_site FROM SPACEXDATASET

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[5]:

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

— . . .

The key in this question is using distinct function.

Launch site names begin with `CCA`

Display 5 records where launch sites begin with the string 'CCA'

In [8]: %sql SELECT * FROM SPACEXDATASET WHERE launch_site LIKE 'CCA%' LIMIT 5

* ibm_db_sa:///qxx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[8]:

DATE	time__utc__	booster_version	launch_site	payload	payload_mass__kg__	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

To find all the launch_site begin with 'CCA', we add % behind it.

Total payload mass

Display the total payload mass carried by boosters launched by NASA (CRS)

In [10]: %sql SELECT SUM(payload_mass__kg_) FROM SPACEXDATASET GROUP BY customer HAVING customer LIKE 'NASA (CRS)'

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[10]:

1
45596

We use sum function to compute total of payload. Of course we have to use group by to compute total of each customer which is like NASA as requests.

Average payload mass by F9 v1.1

Display average payload mass carried by booster version F9 v1.1

In [11]: %sql SELECT AVG(payload_mass__kg_) FROM SPACEXDATASET GROUP BY Booster_Version HAVING Booster_Version LIKE 'F9 v1.1'

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[11]:

1
2928

This is as question 3 but change condition from customer to booster version

First successful ground landing date

List the date when the first succesful landing outcome in ground pad was acheived.

Hint: Use min function

In [28]: %sql SELECT MIN(Date) FROM SPACEXDATASET WHERE landing__outcome LIKE 'Success (ground pad)'

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[28]:

1
2015-12-22

This is as question 2, the difference is we add function Min to Date attribute to find earliest day.

Successful drone ship landing with payload between 4000 and 6000

In [31]: %sql SELECT Booster_Version FROM SPACEXDATASET where landing__outcome like 'Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[31]:

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

This question is as question 2. We add one more condition after 'Where' by using 'and'

Total number of successful and failure mission outcomes

In [44]: %sql SELECT mission_outcome, COUNT(*) as Total FROM SPACEXDATASET group by mission_outcome

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.clogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[44]:

mission_outcome	total
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

I used function count(*) to count unique data from mission_outcome column. We can see 1 failure, 99 success, and 1 success with status unclear.

Boosters carried maximum payload

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
In [56]: %sql SELECT Booster_Version FROM SPACEXDATASET WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXDATASET)
* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.
```

```
Out[56]:
```

booster_version
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

I used nested queries. The nested query is used to find max payload, when the outside query is used to display booster version from it.

2015 launch records

In [59]: %sql SELECT Booster_Version, Launch_Site, landing__outcome FROM SPACEXDATASET WHERE landing__outcome LIKE 'Failure (drone ship)' AND YEAR (Date) = '2015'

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

Out[59]:

booster_version	launch_site	landing__outcome
F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

This question is as question 6. We use function YEAR() to return year of date attribute.

Rank success count 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

In [65]: %sql SELECT landing__outcome, COUNT, RANK() OVER (ORDER BY COUNT DESC) AS RANK FROM (SELECT landing__outcome, COUNT(*) AS COUNT FROM SPAC EXDATASET GROUP BY landing__outcome)

* ibm_db_sa://qqx48001:***@fbd88901-ebdb-4a4f-a32e-9822b9fb237b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32731/bludb
Done.

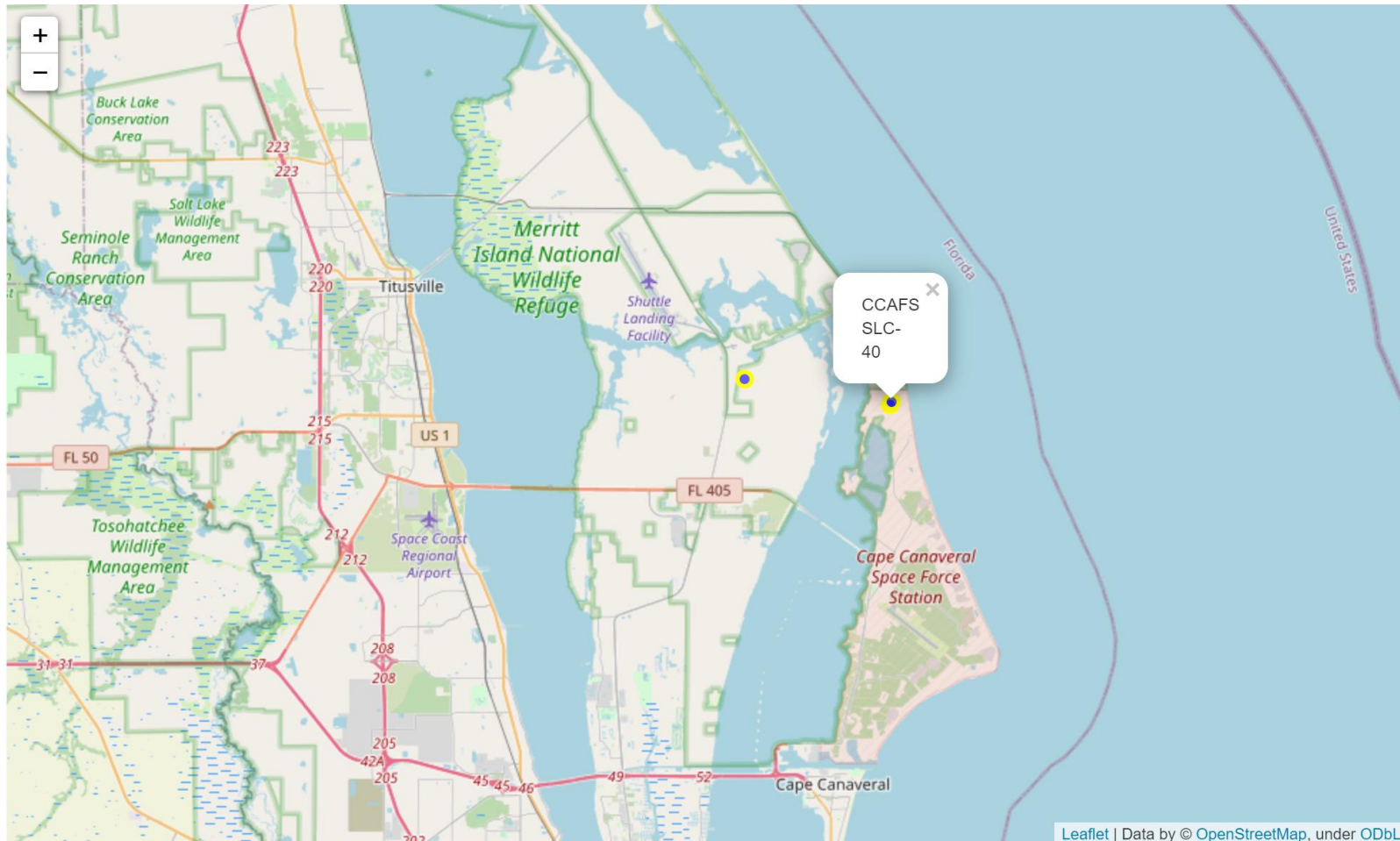
Out[65]:

landing__outcome	COUNT	RANK
Success	38	1
No attempt	22	2
Success (drone ship)	14	3
Success (ground pad)	9	4
Controlled (ocean)	5	5
Failure (drone ship)	5	5
Failure	3	7
Failure (parachute)	2	8
Uncontrolled (ocean)	2	8
Precluded (drone ship)	1	10

I used nested queries. The nested query is used to display number of each landing_outcome. The outside query is used to display rank from it.

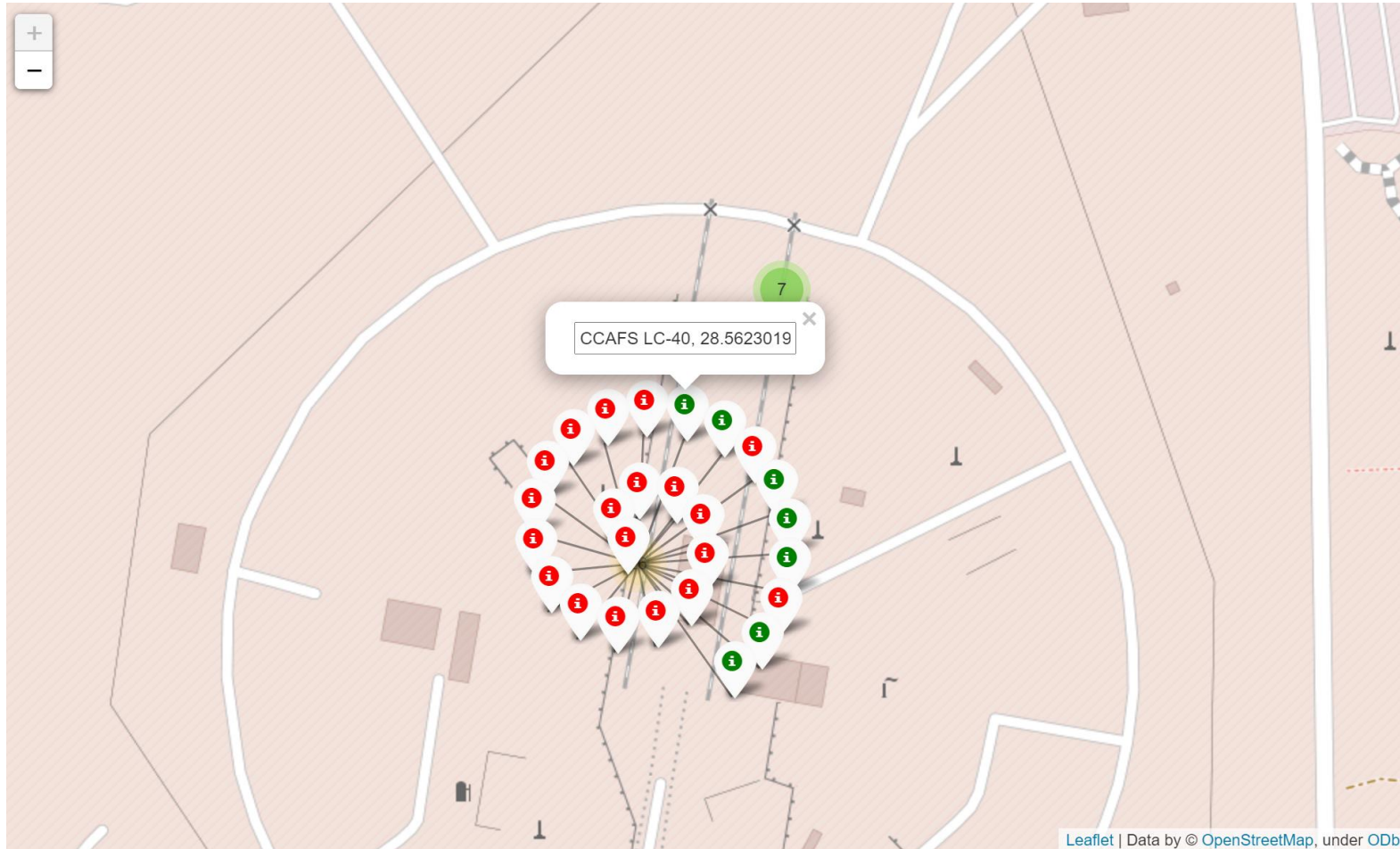
Interactive map with Folium

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



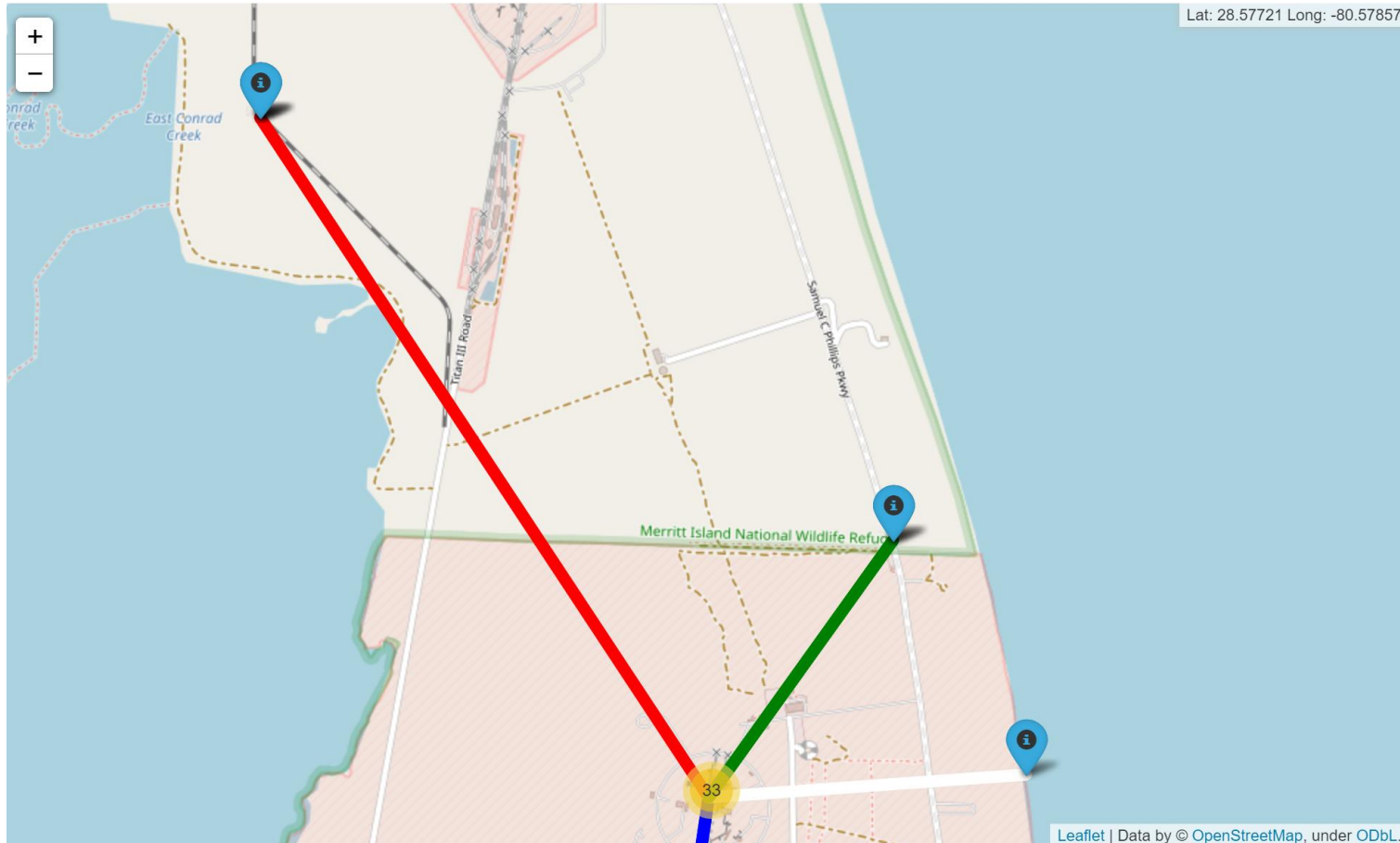
When clicking on the dot, name of launch site will appear.

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



When clicking on the marker, name of launch site will appear.

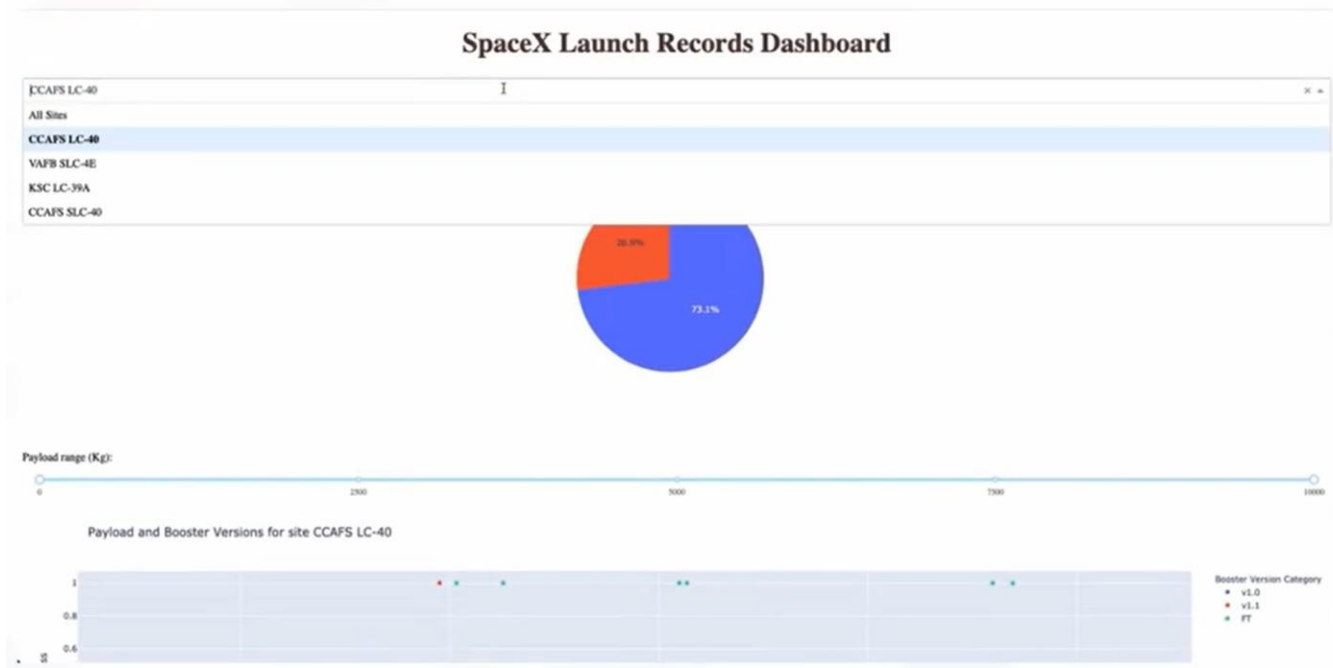
<Folium map screenshot 3>



Red line is distance from a marker to railway. Green line is to highway. White line is to coastline. Blue line is to closest city.

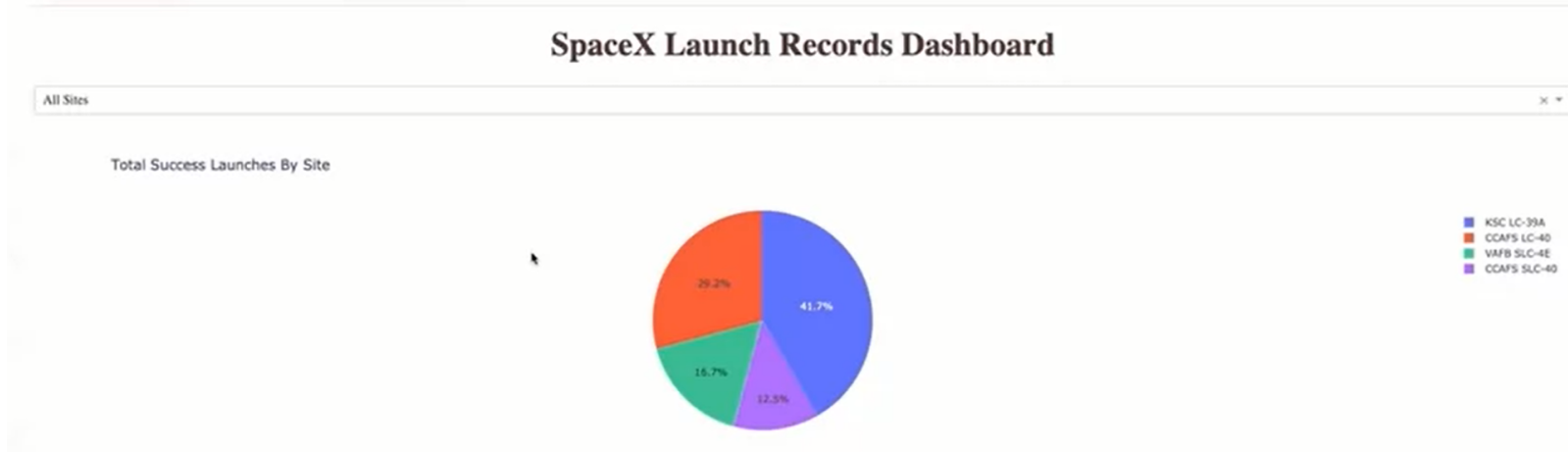
Build a Dashboard with Plotly Dash

<Drop-down menu>

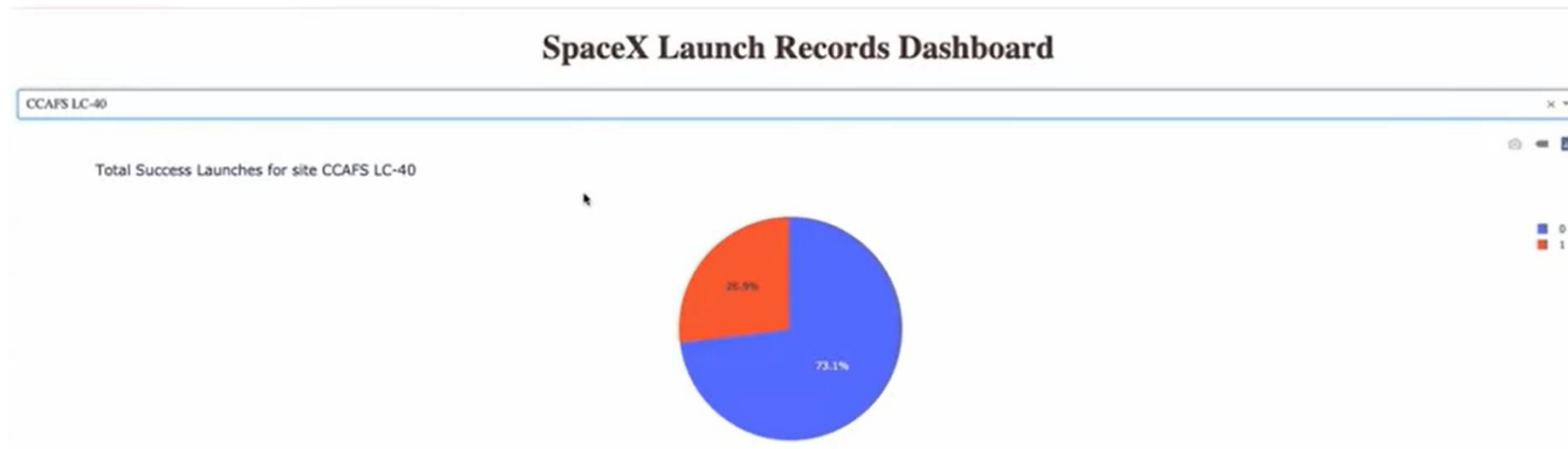


Drop-down menu with 5 options.

<Pie chart>

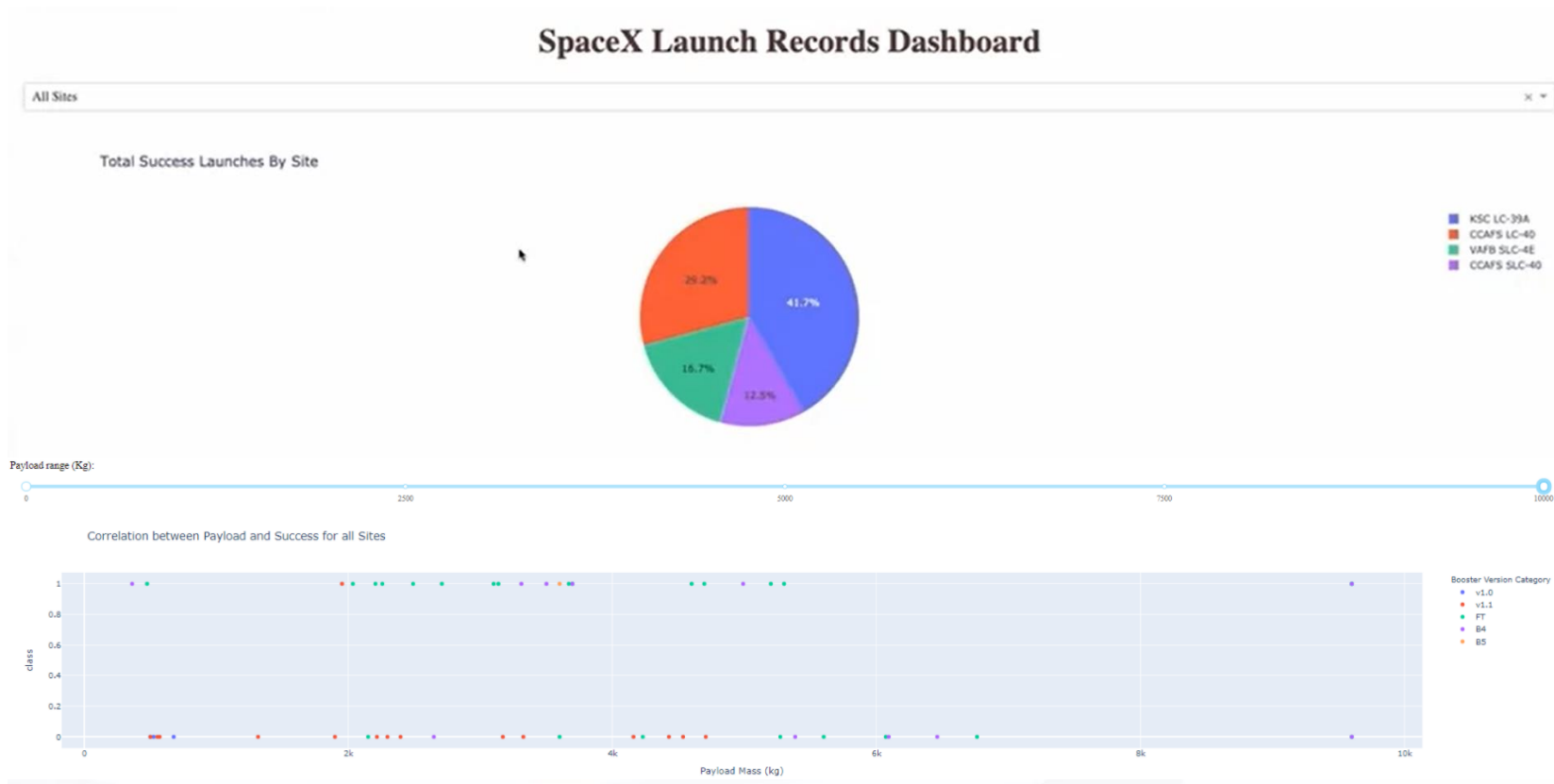


Pie Chart with All Sites



Pie Chart of CCAFS LC 40 Site

Scatter Chart



Scatter Chart of
All Sites

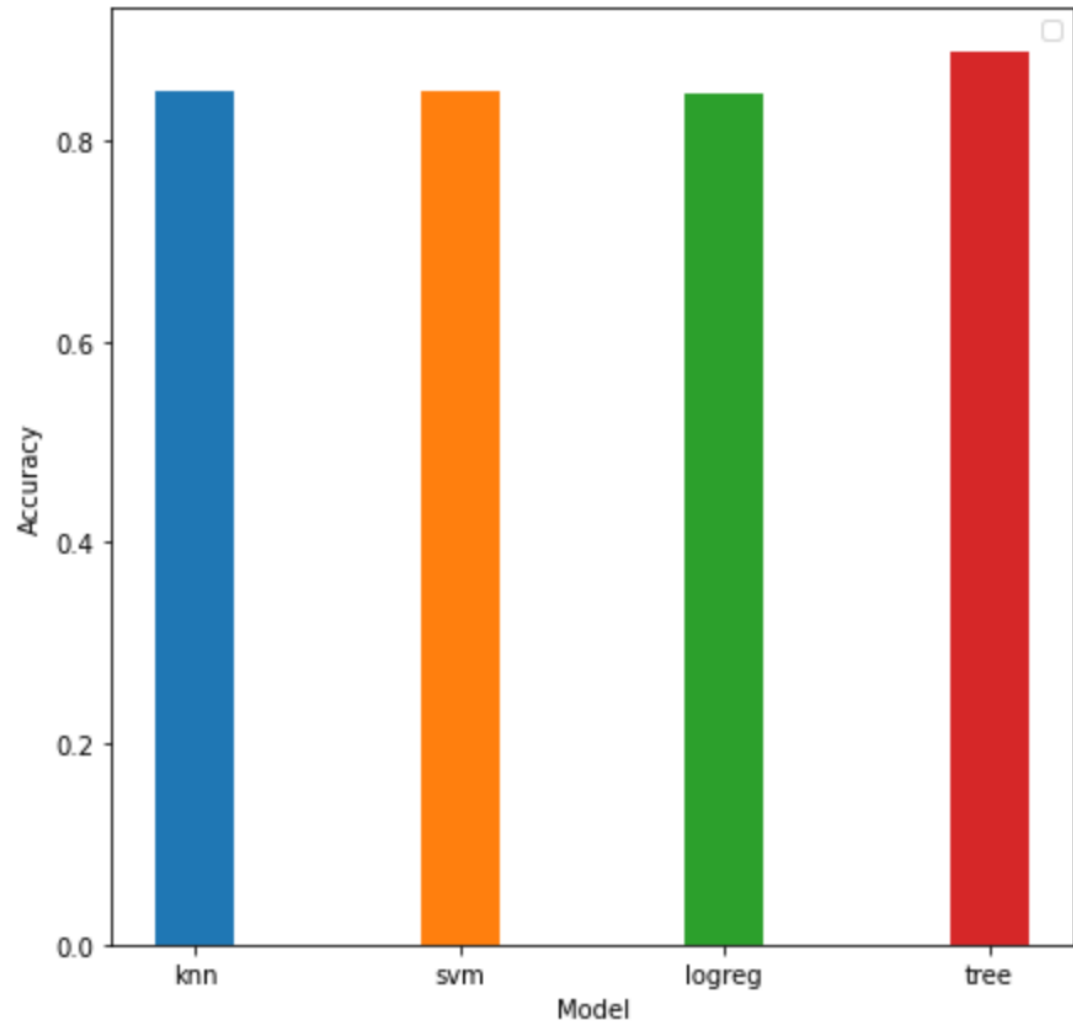
Predictive analysis (Classification)

Classification Accuracy

We can see Decision Tree model has the highest Accuracy which is 94.4%

```
plt.xlabel('Model')  
plt.ylabel('Accuracy')  
plt.xticks([1,2,3,4],['knn','svm','logreg','tree'])  
plt.show()
```

No handles with labels found to put in legend.



Confusion Matrix

We have True Negative (TN): 5

False Positive (FP): 1

False Negative (FN): 0

True Positive (TP): 12

→ Actual Did not land: 6


Actual Landed: 12

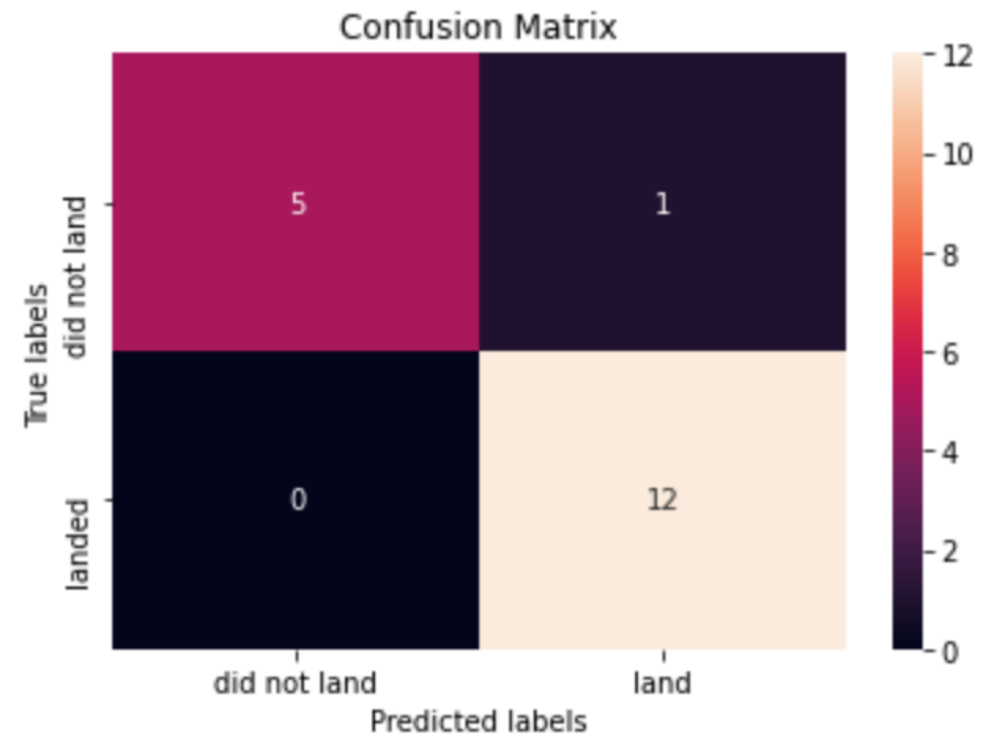
Predicted did not land: 5

Predicted landed: 13

→ Accuracy: $(TP+TN)/total = (12+5)/18 = 0.944444444444$

→ This is a good accuracy when we predict landed 13 times which has actual 12 landed with accuracy of 94%. It means this model only wrong 1

In [37]:  `yhat = tree_cv.predict(X_test)`
`plot_confusion_matrix(Y_test,yhat)`



CONCLUSION



- In this project, using Decision Tree model, we found the best model which create 94.4% accuracy of predicting whether a rocket lands or not. Other models also have a high accuracy.
- I feel rewarded with the efforts and believe this course with all the topics covered is well worthy of appreciation. This project has shown me a practical application to resolve a real situation that has impacting personal and space industry impact using Data Science tools. The mapping with Folium is a very powerful technique to consolidate information and make the analysis and decision better with confidence.

APPENDIX



- <https://github.com/poker4196/IBMFinal>