IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Yevhenii Horbatiuk
06.07.2023

Outline

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

Executive Summary

- Summary of methodologies. During investigation using next step of methodologies: Data Collection, Data Preparation, Modelling and Evaluation.
- Summary of all results. Finally, as a result have been discover some relationship and patterns in the data that relate to the landing of rocket. Also, was building prediction model which predicts the probability of the launch outcome.

Introduction

- Project background and context. The background for the research was the experience of other companies involved in similar developments.

- Problems you want to find answers. The main issues raised were:
  - How effective is this technology?
  - At what stages and under what conditions are they a hindrance?
  - What benefit was bring?
  - Explore patterns. What can be patterns?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - API. Getting some base data from SpaceX API.
  - Web Scraping. Extract a Falcon 9 launch data from Wikipadia
- Perform data wrangling
  - In this case, it will perform some Exploratory Data Analysis to find some patterns in the data and determine what would be the label for training supervised models.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Creating pipline for data preparing and using SVM, Classification Trees and Logistic Regression with diff Hyperparameters to find best estimator, also it was evaluation by accuracy metric

Data Collection

2 way:

- API

- Web Scraping from Wikipedia

Data Collection – SpaceX API

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

Now let's start requesting rocket launch data from SpaceX API with the following URL:

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

```
[9]: response = requests.get(spacex_url)
```

Check the content of the response

```
10]: print(response.content)
```

b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patch":{"small":"https://images2.imgbox.com/94/f2/NN6Ph45r_o.png","large":"https://images2.imgbox.com/5b/02/QcxHUb5V_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flickr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=0a_00nJ_Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html","wikipedia":"https://en.wikipedia.org/wiki/DemoSat"},"static_fire_date_utc":"2006-03-17T00:00:00.000Z","static_fire_date_unix":1142553600,"net":false,"window":0,"rocket":"5e9d0d95eda69955f709d1eb","success":false,"failures":[{"time":33,"altitude":null,"reason":"merlin engine failure"}],"details":"Engine failure at 33 seconds and loss of vehicle","crew":[],"ships":[],"capsules":[],"payloads":["5eb0e4b5b6c3bb0006eeb1e1"],"launchpad":"5

Data Collection - Scraping

```
[5]: # use requests.get() method with the provided static_url
     response = requests.get(static_url)
     # assign the response to a object
```

Create a `BeautifulSoup` object from the HTML `response`

```
[6]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
     soup = BeautifulSoup(response.content)
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
[7]: # Use soup.title attribute
     soup.title
```

```
[7]: <title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

EDA with Data Visualization

Because they are easy to read and allow you to visually evaluate the data

Build an Interactive Map with Folium

Folium is a Python library that enables you to create interactive maps and visualizations. Here are several reasons why you might consider using Folium for data visualization:

Ease of Use: Folium is built on top of the popular mapping library Leaflet.js, but it provides a Pythonic interface, making it relatively easy to create interactive maps and visualizations even if you have limited experience with JavaScript.

Interactive Visualizations: Folium allows you to create interactive maps that can be zoomed, panned, and clicked on to reveal additional information. This interactivity enhances the user experience and enables you to present complex data in an engaging and intuitive way.

Rich Map Features: Folium supports various tilesets, including Mapbox, OpenStreetMap, and Stamen, allowing you to choose the map style that best suits your visualization needs. Additionally, you can overlay markers, polygons, and other shapes on the map to represent your data in a geospatial context.

Integration with Pandas: Folium seamlessly integrates with Pandas, a popular data manipulation library in Python. This integration enables you to easily visualize data from Pandas DataFrames on interactive maps, making it convenient for exploring geospatial patterns and relationships in your data.

Build a Dashboard with Plotly Dash

Plotly Dash is a Python library that enables you to build interactive web applications for data visualization. Here are several reasons why you might consider using Plotly Dash for data visualization:

Interactive and Real-Time Updates: Plotly Dash allows you to create interactive visualizations that respond to user inputs in real-time. This means you can build dynamic dashboards and applications where users can explore and interact with the data, such as filtering, selecting, and zooming in on specific data points or regions of interest.

Easy Integration of Plotly Graphs: Plotly Dash is built on top of the Plotly library, which is known for its powerful and aesthetically pleasing visualizations. With Plotly Dash, you can easily incorporate Plotly graphs, such as scatter plots, line charts, bar plots, and heatmaps, into your web application. This gives you access to a wide range of customizable and interactive visualization options.

Predictive Analysis (Classification)

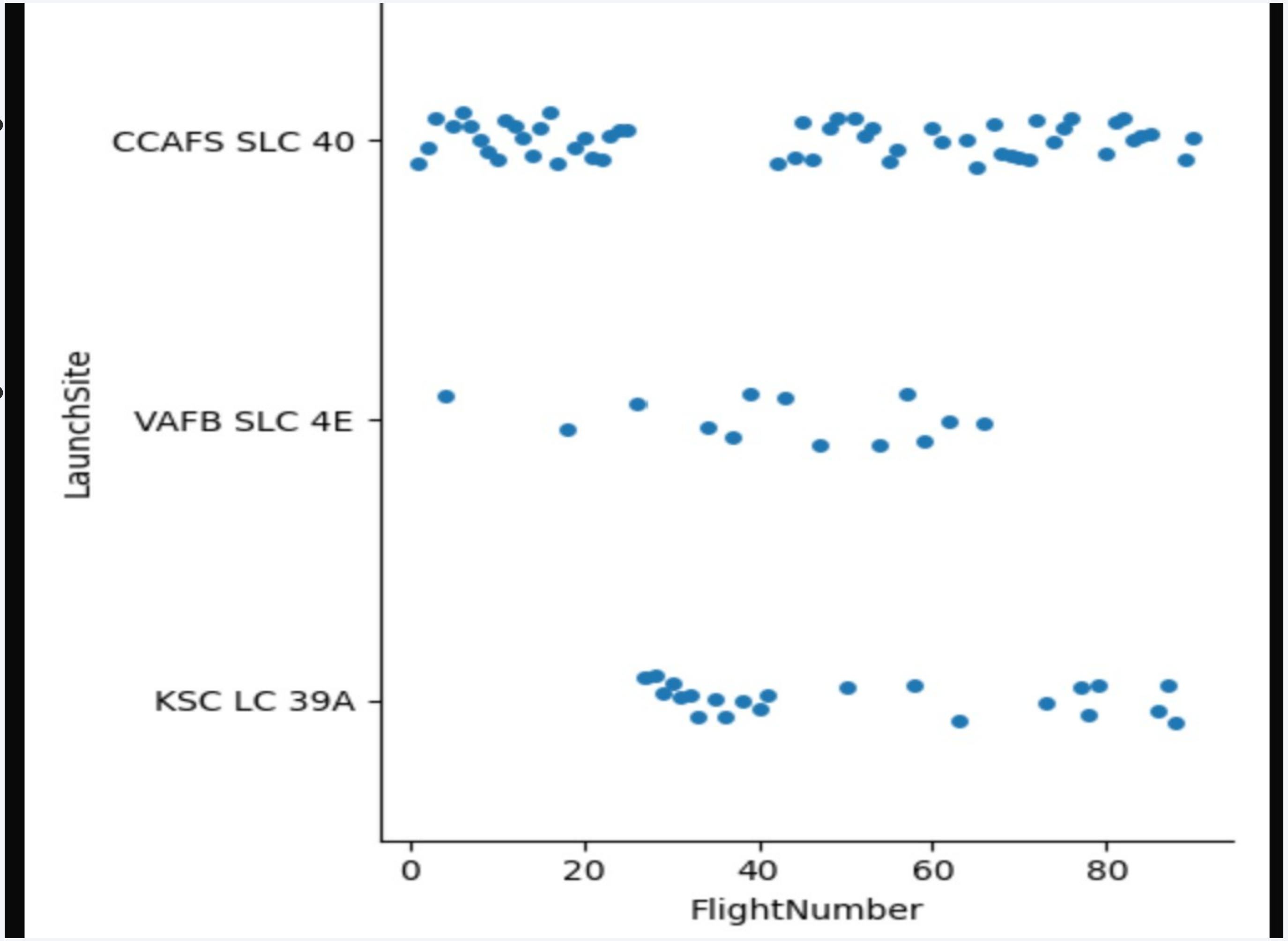
Data preparing to ML, normalized.

For define best model was build diff model KNN, LogReg, Tree, SVM and evaluated by accuracy

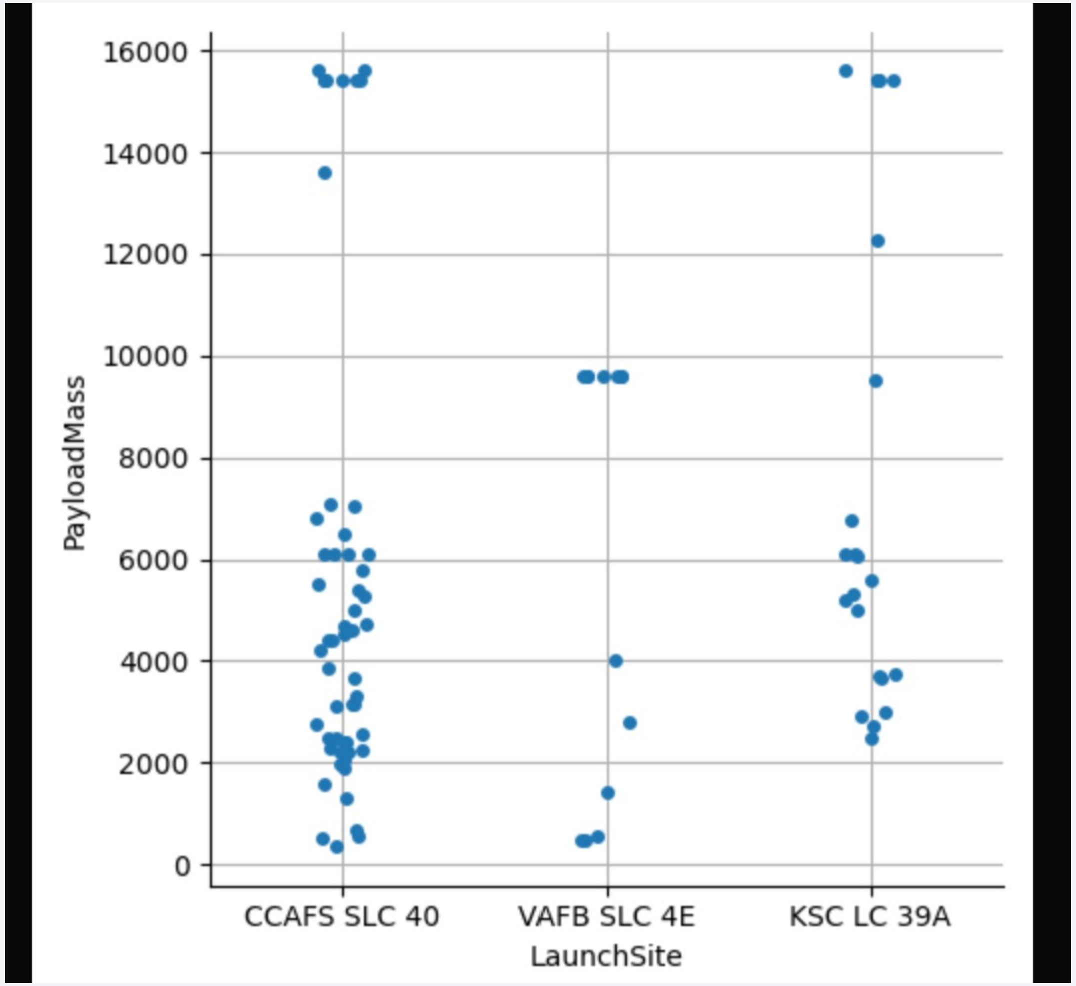
Section 2

Insights drawn from EDA

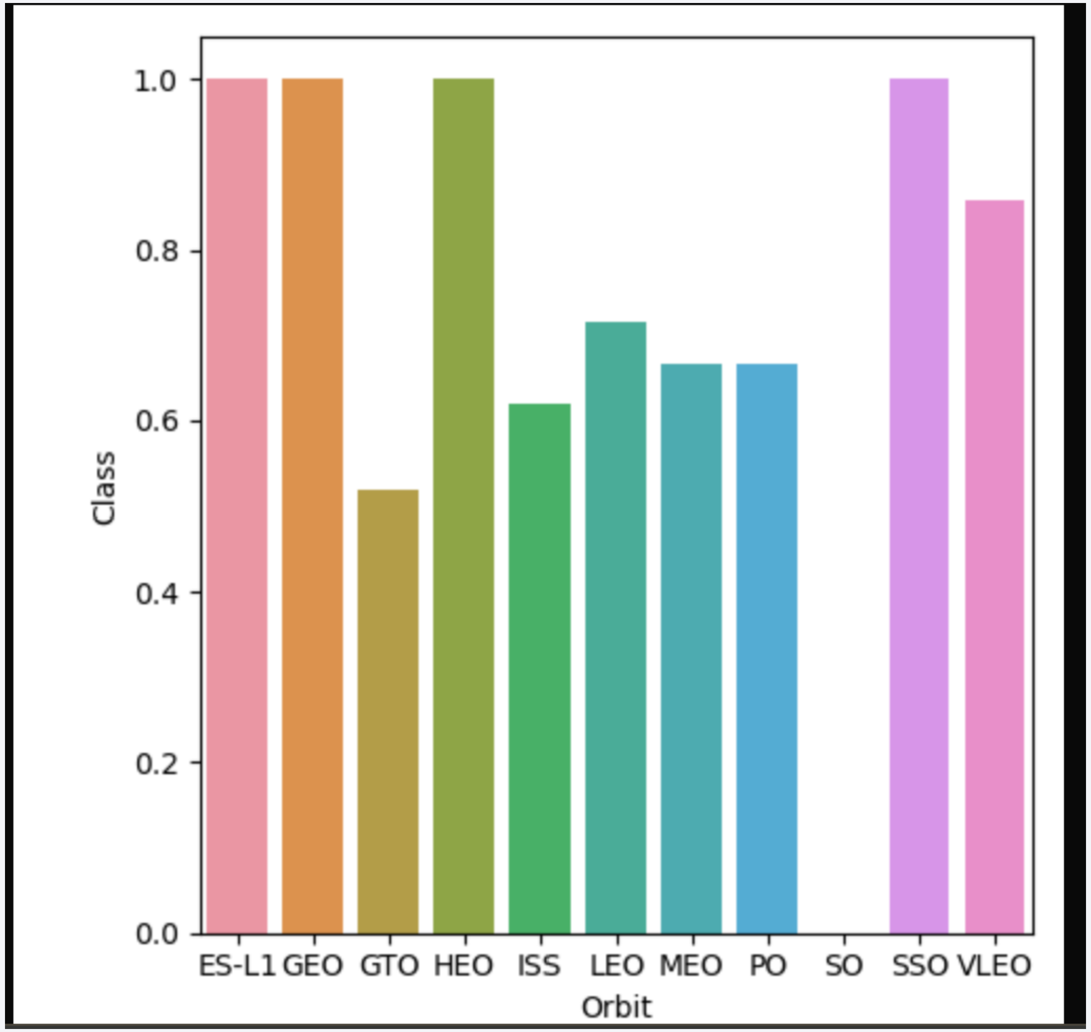
Flight Number vs. Launch Site



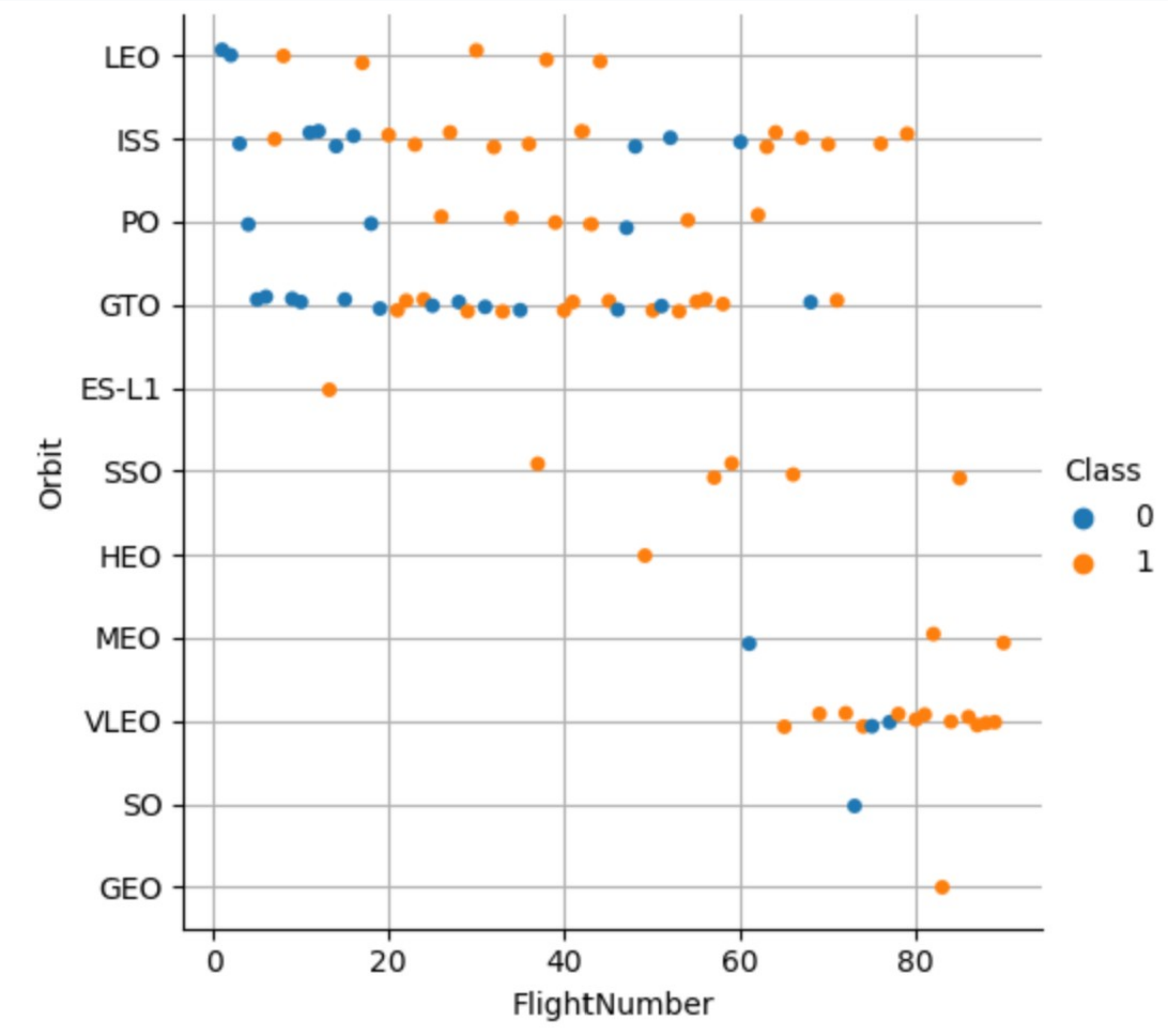
Payload vs. Launch Site



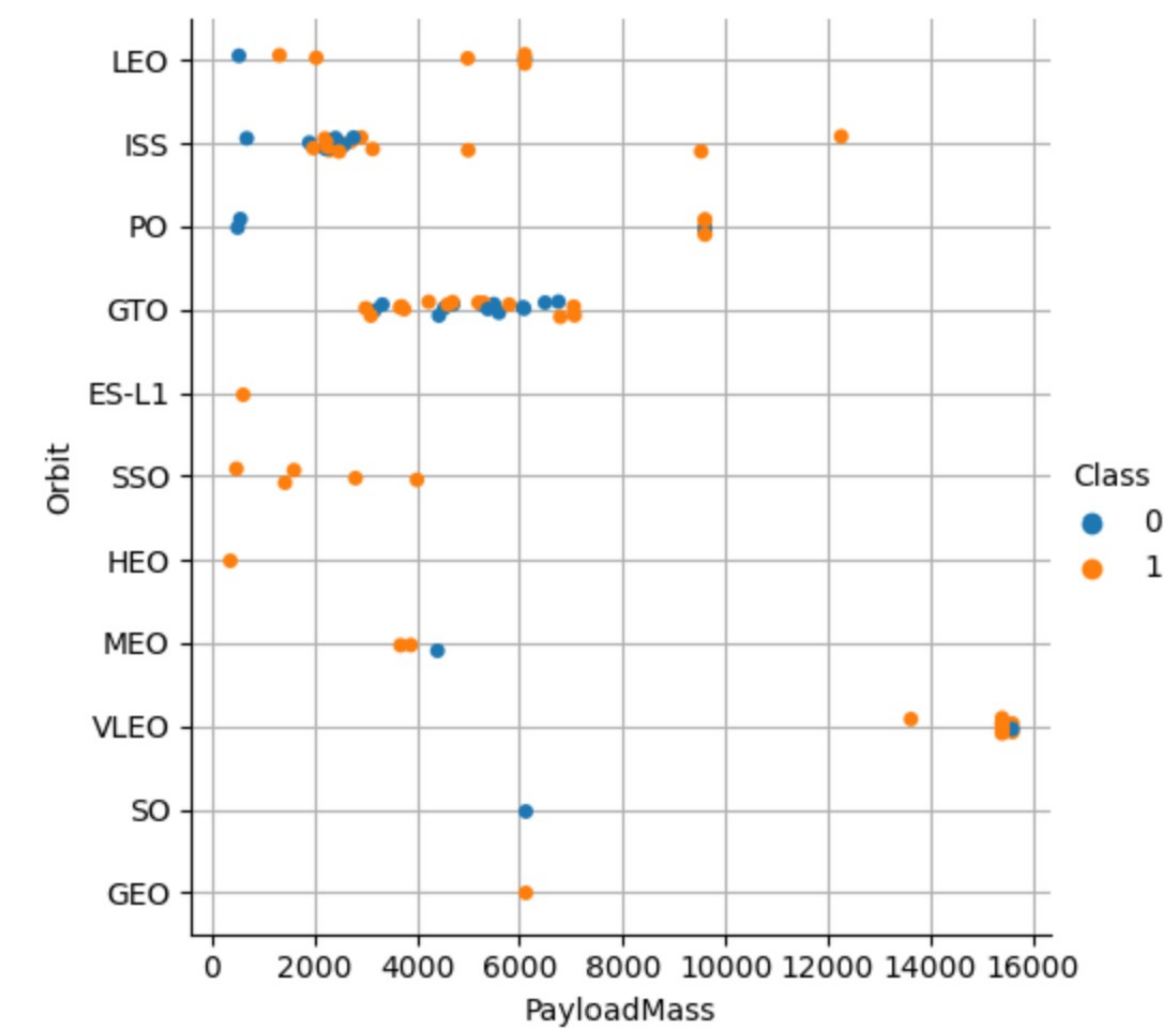
Success Rate vs. Orbit Type



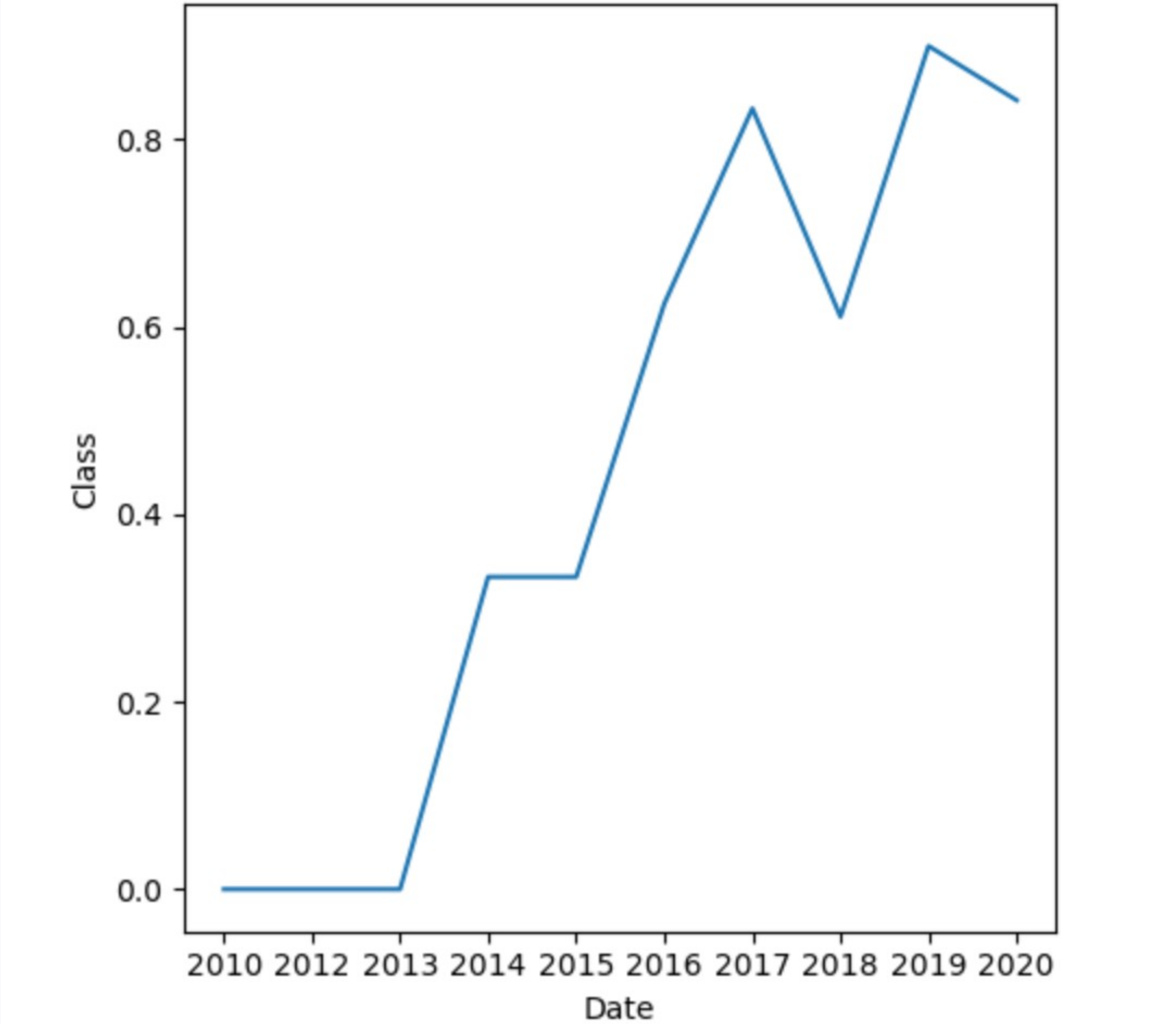
Flight Number vs. Orbit Type



Payload vs. Orbit Type



Launch Success Yearly Trend



All Launch Site Names

```
[8]: %%sql
     SELECT DISTINCT Launch_Site FROM SPACEXTBL

 * sqlite:///my_data1.db
Done.
```

[8]:

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

Launch Site Names Begin with 'CCA'

```
[9]: %%sql
SELECT *
FROM SPACEXTBL
WHERE Launch_Site like "CCA%"
LIMIT 5
```

```
 * sqlite:///my_data1.db
Done.
```

[9]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Out
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Su
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Su
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Su
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Su
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Su

Total Payload Mass

```
[10]: %%sql
SELECT SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTBL
WHERE Customer like "NASA (CRS)"

 * sqlite:///my_data1.db
Done.
```

[10]:

SUM(PAYLOAD_MASS__KG_)
45596.0

Average Payload Mass by F9 v1.1

Calculate the average payload mass carried by booster version F9 v1.1

Present your query result with a short explanation here

```
11]: %%sql
     SELECT SUM(PAYLOAD_MASS__KG_)
     FROM SPACEXTBL
     WHERE Booster_Version like "F9 v1.1"

      * sqlite:///my_data1.db
     Done.

11]: SUM(PAYLOAD_MASS__KG_)
                    14642.0
```

First Successful Ground Landing Date

Find the dates of the first successful landing outcome on ground pad

Present your query result with a short explanation here

```
[12]: %%sql
SELECT substr(Date, 7) || '-' || substr(Date, 4, 2)|| '-' || substr(Date, 1, 2) as date_n, *
FROM SPACEXTBL
WHERE Landing_Outcome like "Succ%"
ORDER by date_n
LIMIT 4

 * sqlite:///my_data1.db
Done.
```

[12]:

date_n	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	M
2015-12-22	22/12/2015	1:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2	2034.0	LEO	Orbcomm	

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

```
[13]: %%sql
SELECT *
FROM SPACEXTBL
WHERE PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000 and Landing_Outcome like "Success (drone ship
```

```
 * sqlite:///my_data1.db
Done.
```

[13]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outco
05/06/2016	5:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696.0	GTO	SKY Perfect JSAT Group	Succ
14/08/2016	5:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600.0	GTO	SKY Perfect JSAT Group	Succ
30/03/2017	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300.0	GTO	SES	Succ
10/11/2017	22:53:00	F9 FT B1031.2	KSC LC-39A	SES-11 / EchoStar 105	5200.0	GTO	SES EchoStar	Succ

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

```
[14]: %%sql
SELECT Mission_Outcome, COUNT(*)
FROM SPACEXTBL
GROUP BY Mission_Outcome

 * sqlite:///my_data1.db
Done.
```

[14]:

Mission_Outcome	COUNT(*)
None	898
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

```
[15]: %%sql

      SELECT *
      FROM SPACEXTBL
      WHERE PAYLOAD_MASS__KG_ ==
      (SELECT MAX(PAYLOAD_MASS__KG_)
      FROM SPACEXTBL)

 * sqlite:///my_data1.db
Done.
```

[15]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outco
11/11/2019	14:56:00	F9 B5 B1048.4	CCAFS SLC-40	Starlink 1 v1.0, SpaceX CRS-19	15600.0	LEO	SpaceX	Suc
01/07/2020	2:33:00	F9 B5 B1049.4	CCAFS SLC-40	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600.0	LEO	SpaceX	Suc
29/01/2020	14:07:00	F9 B5 B1051.3	CCAFS SLC-40	Starlink 3 v1.0, Starlink 4 v1.0	15600.0	LEO	SpaceX	Suc
[illegible]	[illegible]	[illegible]	CCAFS SLC-	Starlink 4 v1.0,	[illegible]	[illegible]	[illegible]	[illegible]

2015 Launch Records

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
[16]: %%sql

SELECT substr(Date,7,4) as year, *
FROM SPACEXTBL
WHERE year == "2015"
```

```
 * sqlite:///my_data1.db
Done.
```

[16]:

year	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	M
2015	01/10/2015	9:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395.0	LEO (ISS)	NASA (CRS)	
2015	02/11/2015	23:03:00	F9 v1.1 B1013	CCAFS LC-40	DSCOVR	570.0	HEO	U.S. Air Force NASA NOAA	
2015	03/02/2015	3:50:00	F9 v1.1 B1014	CCAFS LC-40	ABS-3A Eutelsat 115 West B	4159.0	GTO	ABS Eutelsat	
2015	14/04/2015	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898.0	LEO (ISS)	NASA (CRS)	
2015	27/04/2015	23:03:00	F9 v1.1 B1016	CCAFS LC-40	Turkmen 52 / MonacoSAT	4707.0	GTO	Turkmenistan National Space Agency	
2015	28/06/2015	14:21:00	F9 v1.1 B1018	CCAFS LC-40	SpaceX CRS-7	1952.0	LEO (ISS)	NASA (CRS)	

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

06-04 and 2017-03-20, in descending order.

[17]:
```
%%sql
SELECT substr(Date, 7) || '-' || substr(Date, 4, 2)|| '-' || substr(Date, 1, 2) as date_n, *
FROM SPACEXTBL
WHERE Landing_Outcome like "Succ%" or Landing_Outcome like "Failure (drone ship)%"
ORDER by date_n
```

•••

[18]:
```
%%sql
select  count("Mission_Outcome") as MISSION_OUTCOME_COUNT,Launch_Site  from SPACEXTBL group by "L
```

```
 * sqlite:///my_data1.db
Done.
```

[18]:

MISSION_OUTCOME_COUNT	Launch_Site
0	None
26	CCAFS LC-40
34	CCAFS SLC-40
25	KSC LC-39A
16	VAFB SLC-4E

ip) or
7-03-

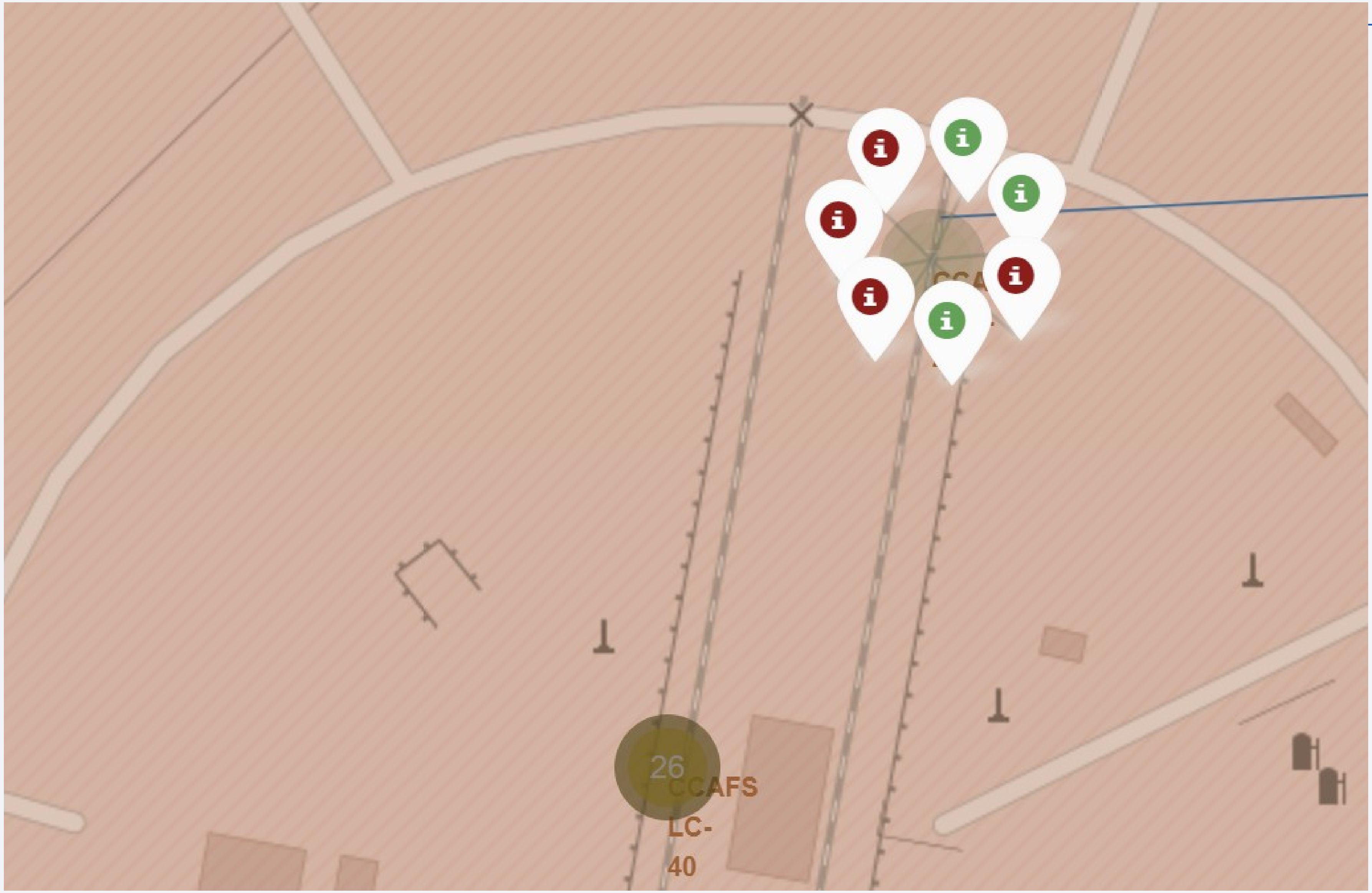
Section 3

Launch Sites Proximities Analysis

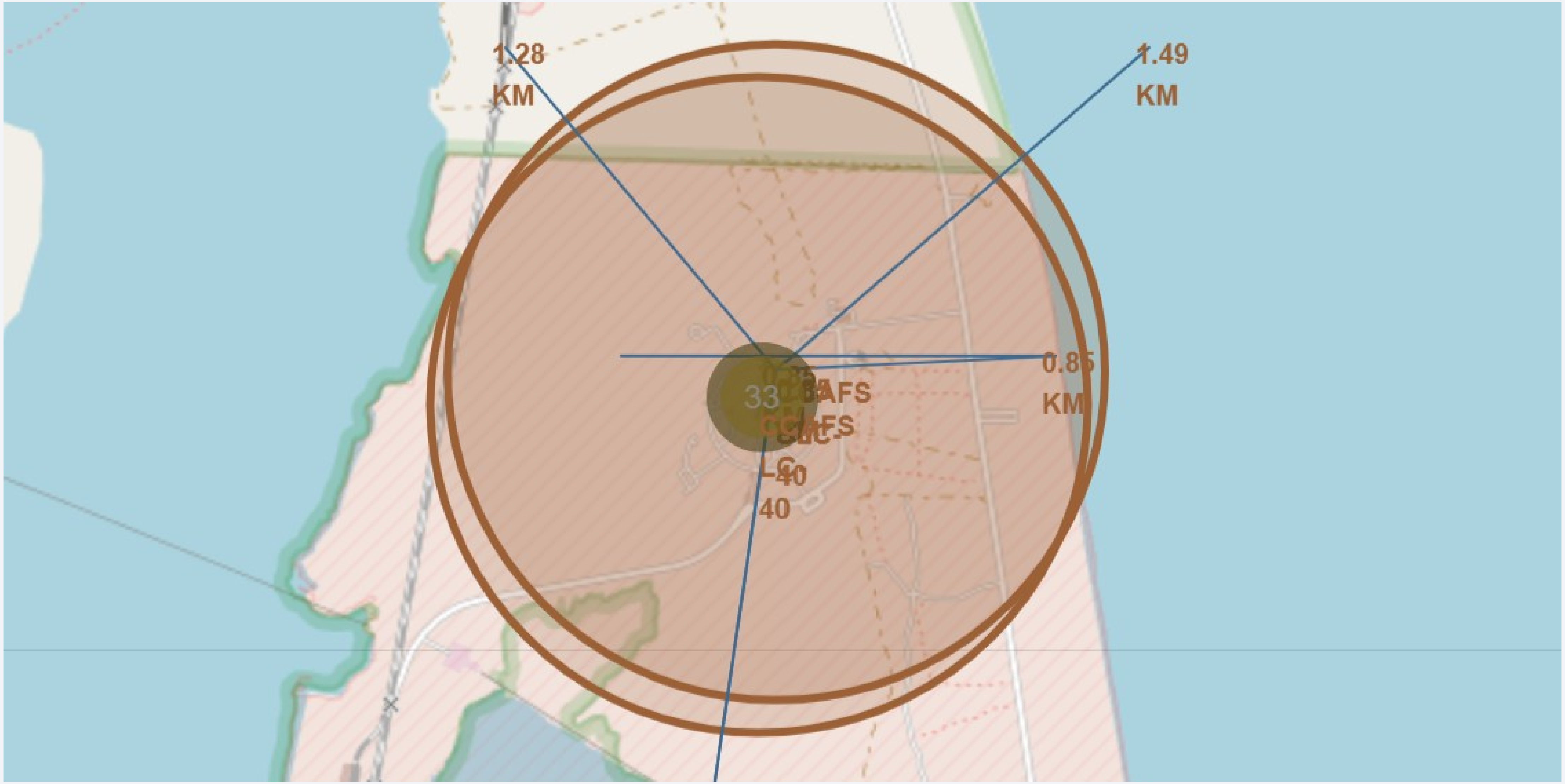
Launch site

Toronto
New Y
Washington
United States
Phoenix
Los Angeles
10
VAFB
SLC-
4E
46
KSC
CCAFS
The Bahamas
La Habana
Cuba
México
Ciudad
de México
Kingston
Ciudad
de Guatemala
Honduras

Launch outcome lable



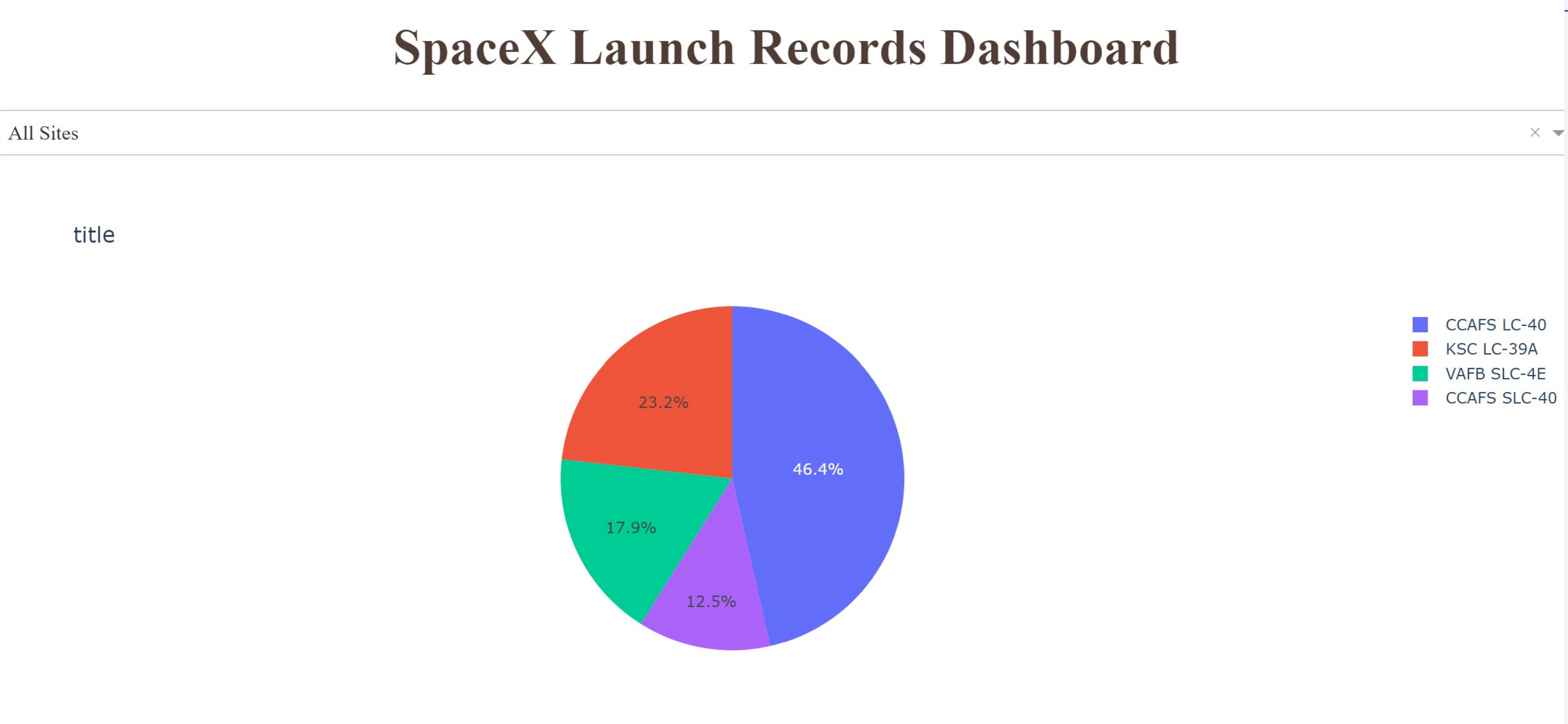
Distance to different objects



Section 4

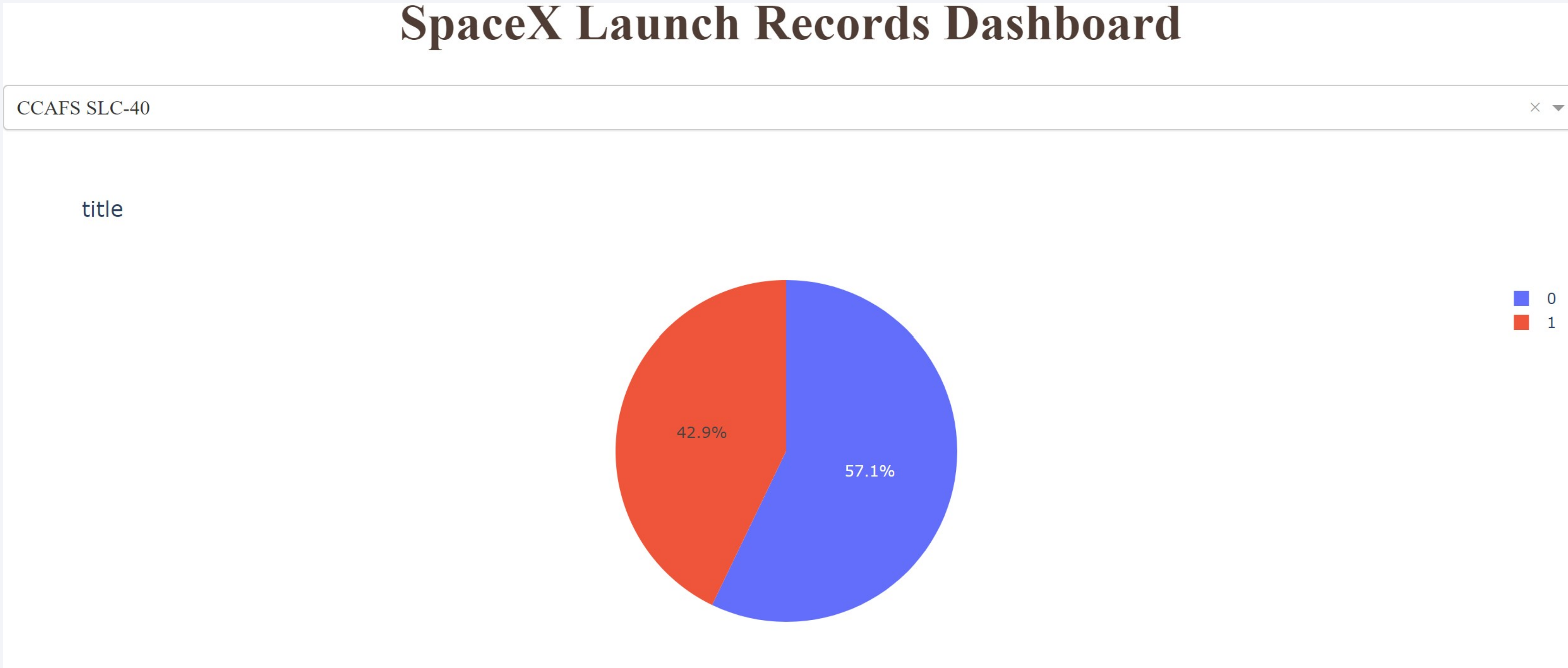
Build a Dashboard with Plotly Dash

All Sites



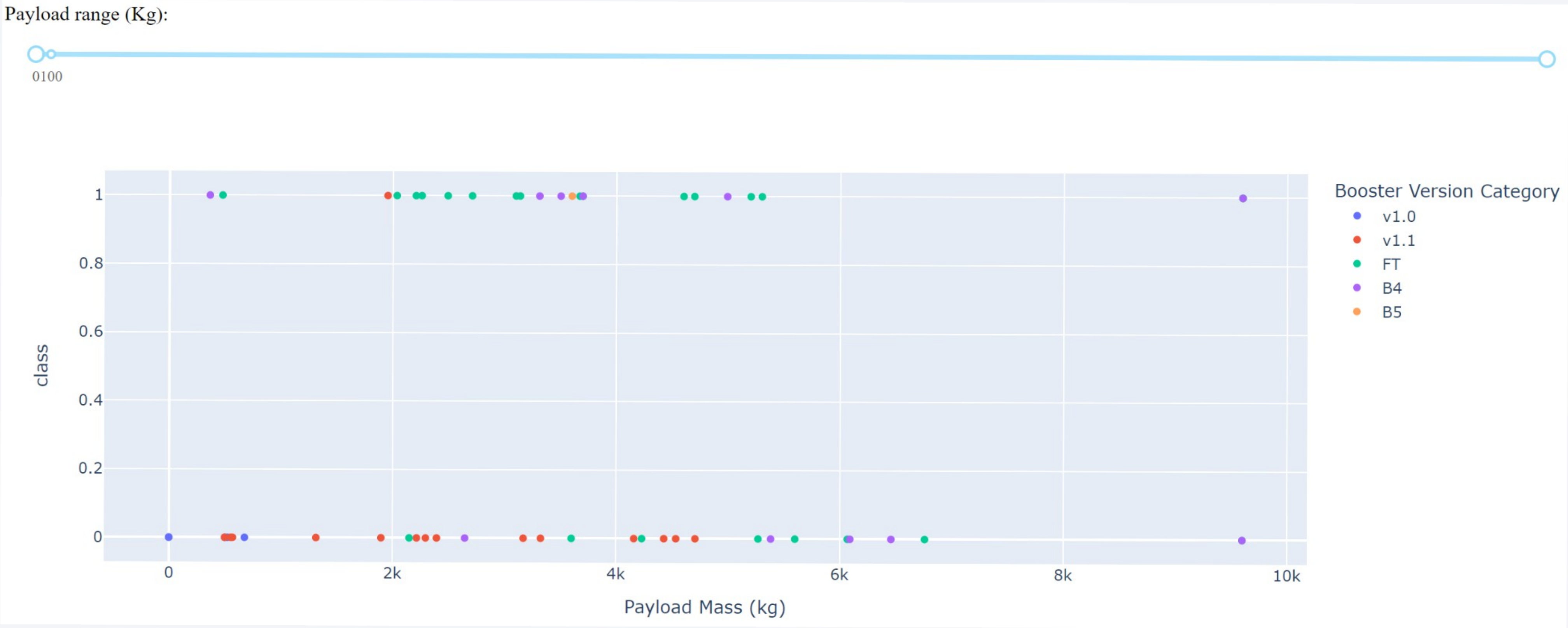
We see the percentage of successful launches

CCAFS SLC-40



On CCAFS SLC-40 the most successful percentage of launches in relation to other places. however, this may be due to the total minority.

Payload vs Outcome



We can see that loading mass does not affect the outcome as much as the Booster Version type. The Booster Version brings more impact to the outcome

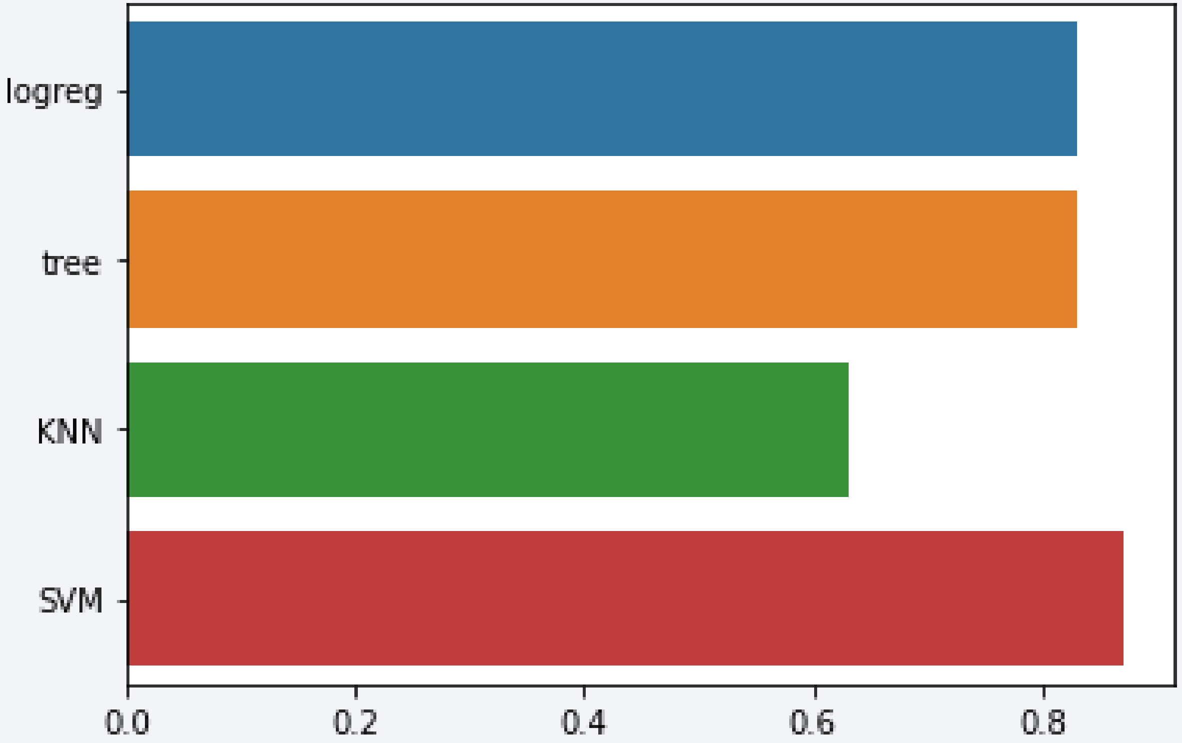
Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Visualize the built model accuracy for all built classification models, in a bar chart

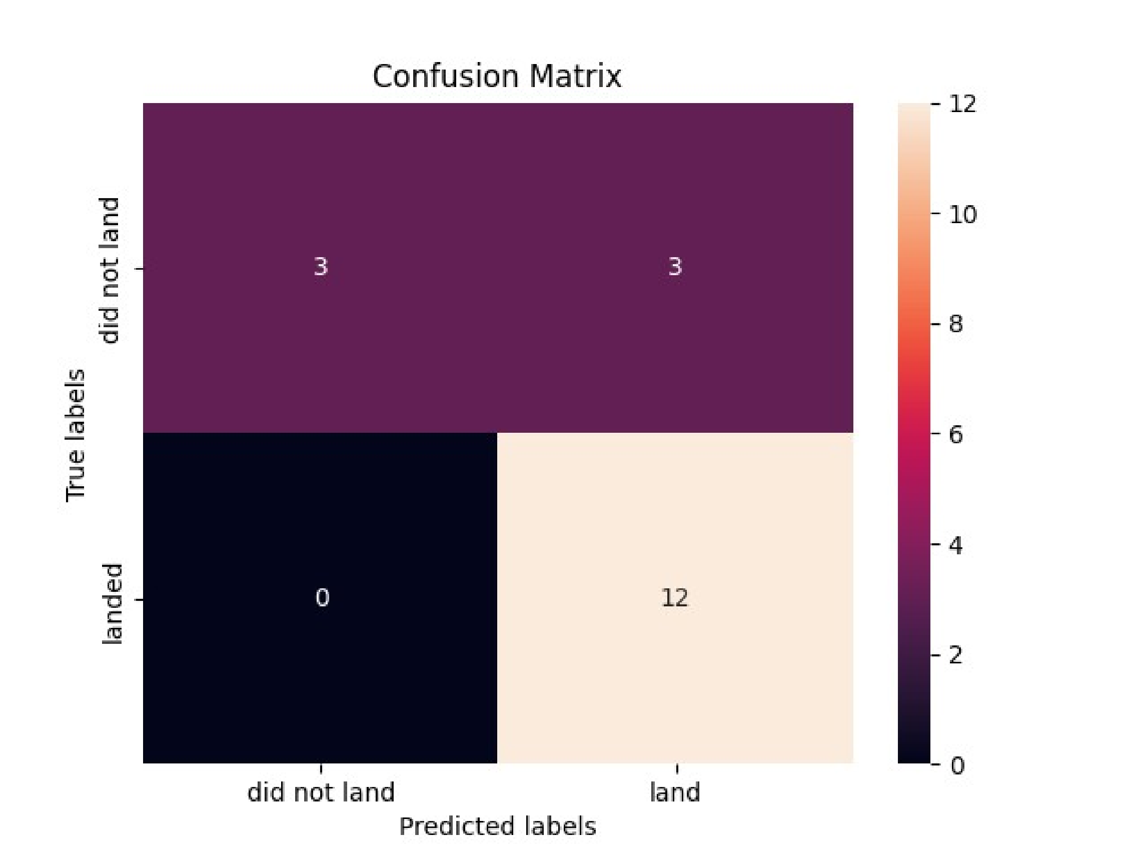
- Find which model has the highest classification accuracy



Best accuracy : 0.87
SVM model

Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation



Conclusions

Point 1

Point 2

Point 3

Point 4

...

Appendix

Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!