IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Antonio Gonzalez

January 2023

Outline

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

Executive Summary

- **Summary of methodologies**
- Describing for each task the method used and apporting information about all the subtracted conclusions
- **Summary of all results**
- Apporting all the graphs and analysis performed for each method

Introduction

- This project tries to show the different methodologies related to data processing and its analysis learned through the courses offered by IBM. In order to materialize the assimilated knowledge, the project has been related to the in-depth study of the takeoffs, orbits, landings and various information of SpaceX rockets.
- With this, we want to reach conclusions that can provide information on the status of both SpaceX and the different launches, such as the success rate of the different launches, the locations (and why) of the SpaceX facilities, predictions for upcoming launches, etc.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - requests.get() & pd.json_normalize()
  - BeautifulSoup()
- Perform data wrangling
  - Setting parameters
- Perform exploratory data analysis (EDA) using visualization and SQL
  - SQL statements
  - Scatter plots/charts

Methodology

Executive Summary

- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Build, tune, evaluate classification models

Data Collection

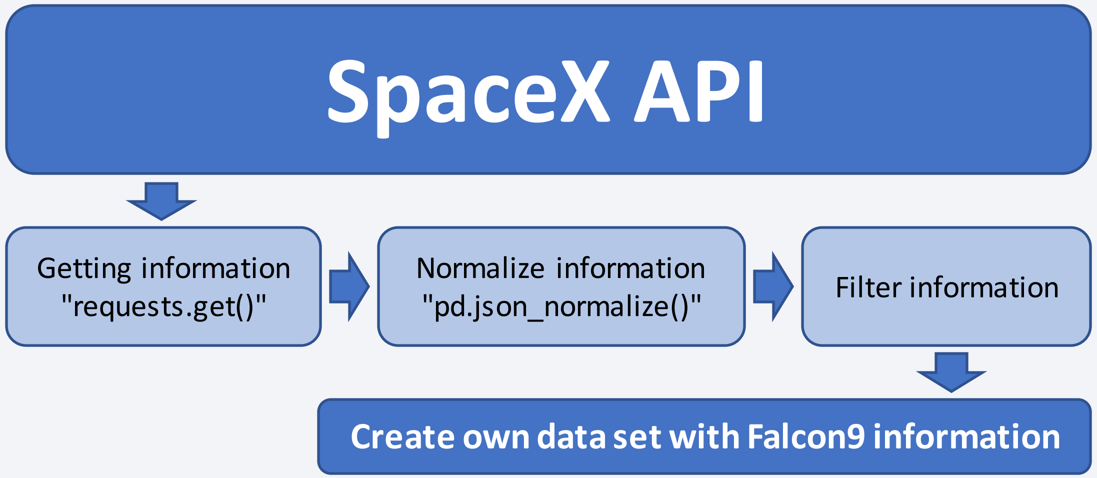
- The data sets were collected through SpaceX API, which can be accessed by this link: https://api.spacexdata.com/v4/launches/past
- And through the Wikipedia Falcon 9 and Heavy Launches webpage: https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches
- Depending on the collecting source, different methods were used:
  - SpaceX API: the python requests.get() method and the pandas library to normalize the data
  - Wikipedia: the BeautifulSoup library was used to perform the web scrapping to get the data

Data Collection – SpaceX API

- The data requested using a **requests.get()** method must be **normalized using pandas** in order to work in a comfortable way.
- We select the data considered relevant (removing all the data IDs or columns which have no information) **filtering the data set**.
- Finally we **remove the Falcon 1** launches keeping only the Falcon 9 launches creating our own data set.

GitHub link:

https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/Data%20Collection%20–%20SpaceX%20API.ipynb



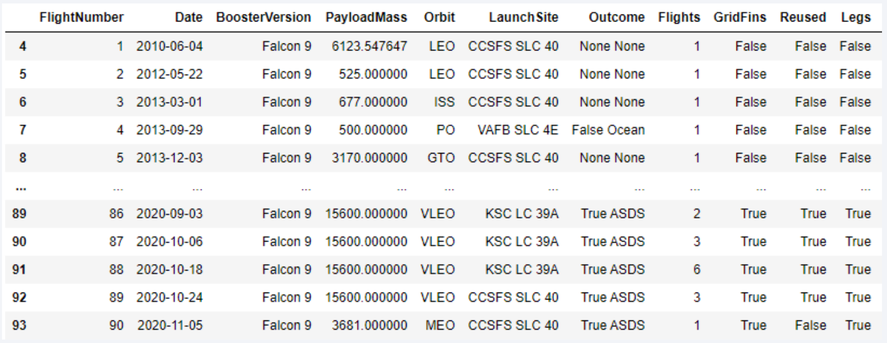
SpaceX API

Getting information "requests.get()"

Normalize information "pd.json_normalize()"

Filter information

Create own data set with Falcon9 information



	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...
89	86	2020-09-03	Falcon 9	15600.000000	VLEO	KSC LC 39A	True ASDS	2	True	True	True
90	87	2020-10-06	Falcon 9	15600.000000	VLEO	KSC LC 39A	True ASDS	3	True	True	True
91	88	2020-10-18	Falcon 9	15600.000000	VLEO	KSC LC 39A	True ASDS	6	True	True	True
92	89	2020-10-24	Falcon 9	15600.000000	VLEO	CCSFS SLC 40	True ASDS	3	True	True	True
93	90	2020-11-05	Falcon 9	3681.000000	MEO	CCSFS SLC 40	True ASDS	1	True	False	True

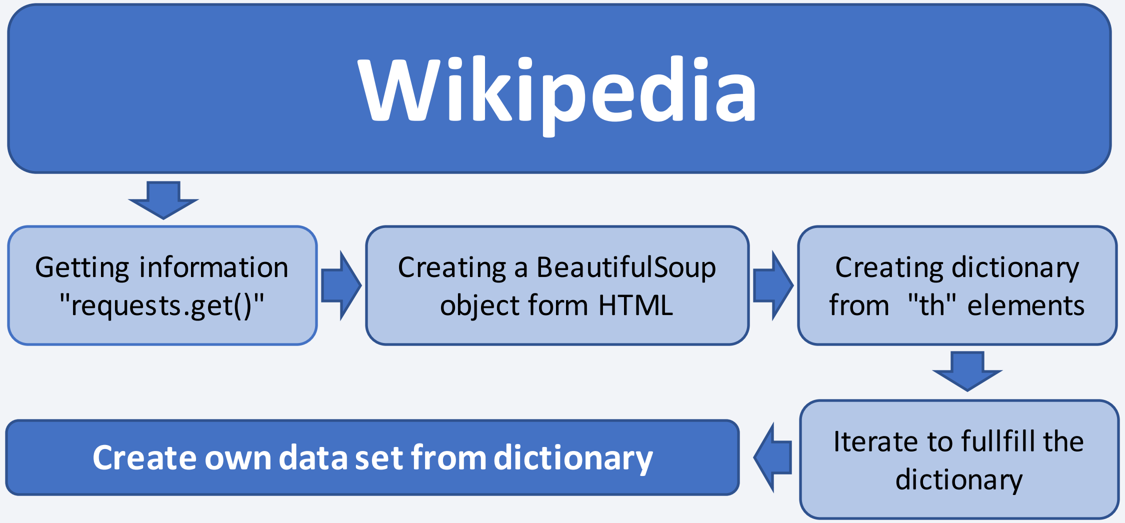
■■■

Data Collection - Scraping

- The data is requested through the Wikipedia static url and, with the **BeautifulSoup() method**, we can interpret the HTTP response and work with it.
- The "**th**" elements are extracted as columns.
- The **data set is filled iterating** each line and selecting the data for every column

GitHub Link:

https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/Data%20Collection%20-%20Scraping.ipynb



Wikipedia

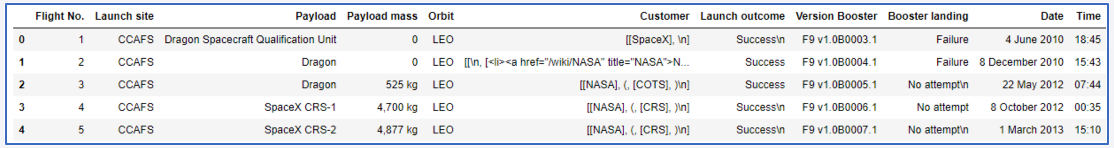
Getting information "requests.get()"

Creating a BeautifulSoup object form HTML

Creating dictionary from "th" elements

Iterate to fullfill the dictionary

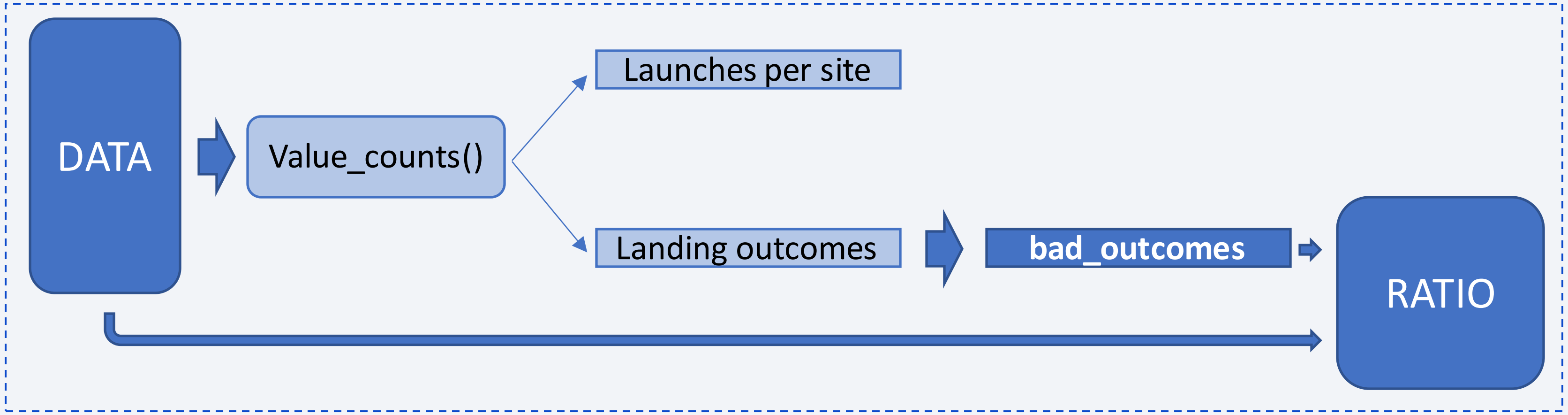
Create own data set from dictionary



	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	[[SpaceX], \n]	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	[[\n, [<li><a href="/wiki/NASA" title="NASA">N...	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	[[NASA], (, [COTS],)\n]	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	[[NASA], (, [CRS],)\n]	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	[[NASA], (, [CRS],)\n]	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

Data Wrangling

- The number of launches on each site, occurrences of each orbit and outcomes per orbit types is calculated using the **value_counts()** method. With the landing outcomes we are able to make a **set** with the bad outcomes and, by knowing this information see how many landings were good and its **ratio.**



Git Hub Link:

https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/Data%20Wrangling.ipynb

EDA with Data Visualization

- To find relations between different flights some **scatter plots** were used:
    - Payload Mass (kg) vs Flight Number
    - Launch Site vs Flight Number
    - Launch Site vs Payload Mass (kg)
    - Orbit vs Flight Number
    - Orbit vs Payload Mass (kg)

- For other proposes, the Orbit vs. Success **bar chart** and the Success vs. Year **line chart** were also shown.

GitHub Link:

https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/EDA%20with%20Data%20Visualization.ipynb

EDA with SQL

UNIQUE LAUNCH SITES

- Using DISTINCT statement to return only the different values

RECORDS BEGGINING WITH "KSC"

- Using the LIKE function and the "%" symbol before the launch site beggining

TOTAL PAYLOAD MASS

- Using the SUM() function to make the sum of all values

AVERAGE PAYLOAD MASS FOR F9 v1.1

- Using the AVG() function to get the average

FIRST SUCCESSFUL LANDING IN DRONE SHIP

- Using the MIN function from all the drone ship success

SUCCESS GROUND PAD WITH 4000 – 6000 PAYLOAD

- Selecting the ground pad success and the BETWEEN statement for the paylaod

TOTAL SUCCESS/FAILURE MISSIONS

- Using the COUNT() function for all the grouped by Mission outcomes

BOOSTER VERSIONS FOR MAXIMUM PAYLOAD

- Using a subquery to determine the maximum payload and selecting the booster versions

MONTH DISPLAY SUCCESSFUL LANDING OUTCOMES

- Using the MONTHNAME() function to format the date

RANKING SUCCESSFUL OUTCOMES BETWEEN DATES

- Using the COUNT function and the BETWEEN statement

GitHub Link

https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- In order to make easy and intuitive a map, some markers, circles and lines has been added.
  - **Markers**: highlight an specific point over the map
  - **Circles**: add a circle on the map to select a region
  - **Lines**: draw lines between points (e.g. distance between points)
- These elements were selected because of the way the user will interact with them and understand the map and the information given.

GitHub Link:

https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/Interactive%20Map%20with%20Folium.ipynb

Build a Dashboard with Plotly Dash

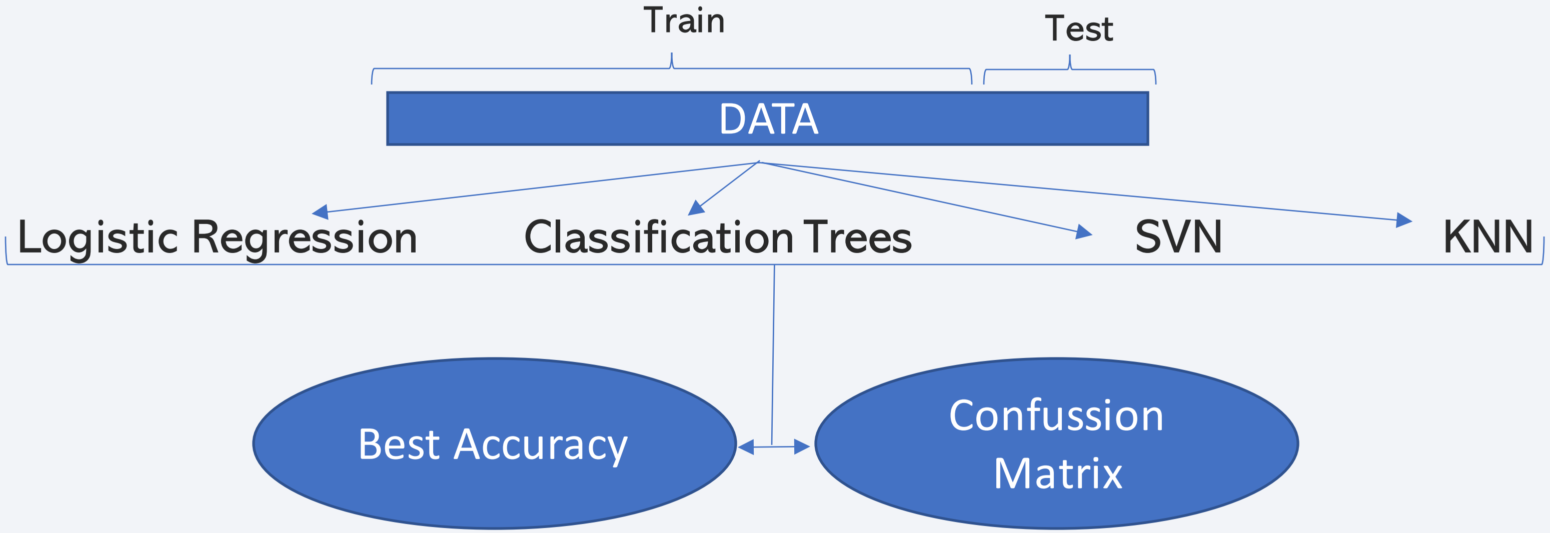
- To summarize the classification of the launches depending on its success or failure, some plots/graphs and interactions were added to the dashboard.
  - Dropdown: To select an specific launching site.
  - Pie chart: Useful to receive easy graphic information about the success/failure.
  - Slider: To select an specific payload range.
  - Scatter plot: Used to check the relation between the payload and the success/failure of the launch.

GitHub Link:

https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/Dashboard%20with%20Plotly%20Dash.py

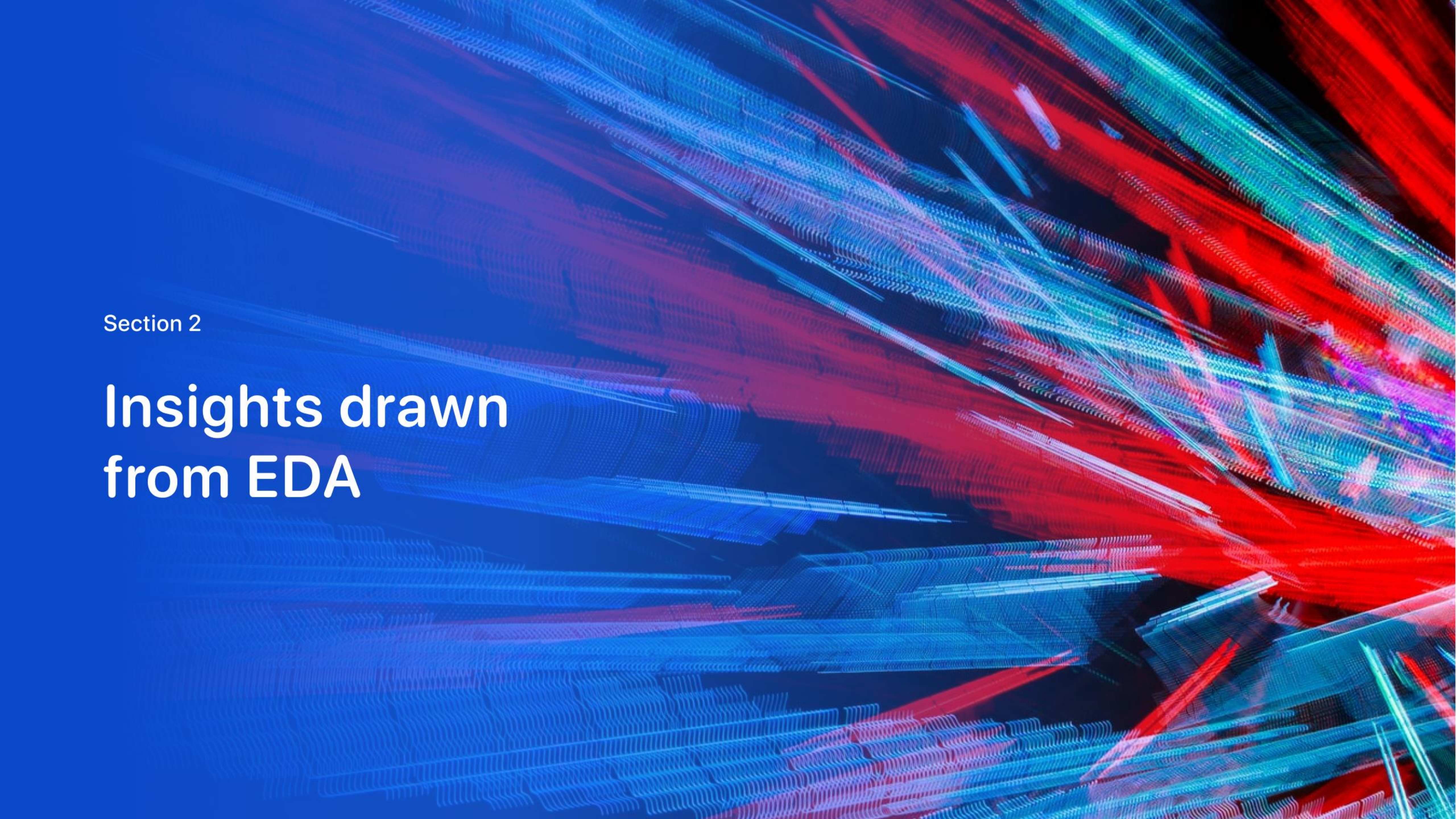
Predictive Analysis (Classification)

- To select the better method to predict the successful landings, Logistic Regression, Classification Trees, SVN and KNN were used on the data split to train. Then applied over the testing one.



GitHub Link:

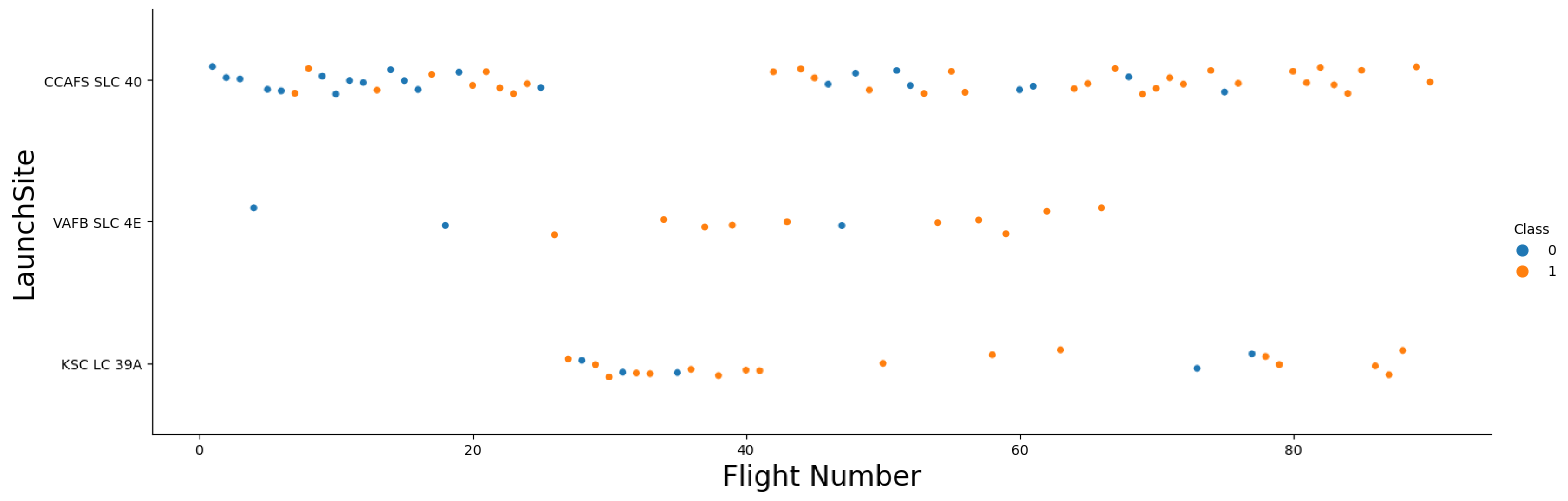
https://github.com/ToniKelao/PythonBasicsForDataScience/blob/main/Predictive%20Analysis%20(Classification).ipynb



Section 2

Insights drawn from EDA

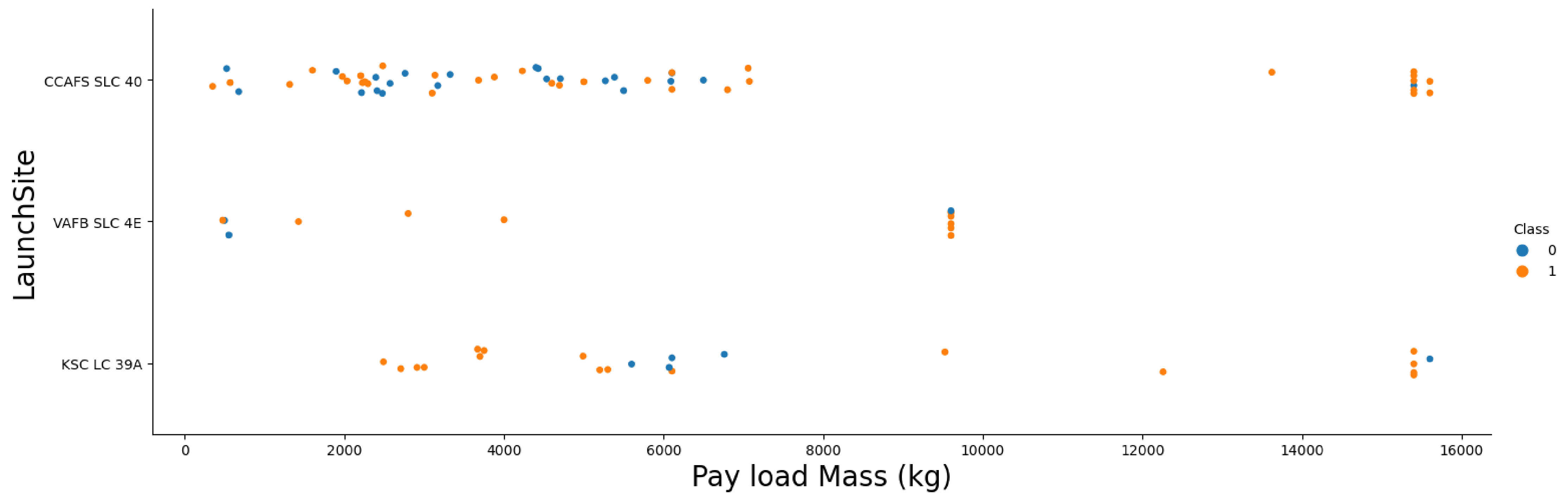
Flight Number vs. Launch Site



On this scatter plot we can see how the **CCAFS SLC 40** launch site was the site selected for the first attempts and, when this site is not used, the **KSC LC 39A** is more prominent.

The **VAFC SLC 4E** seems not to present an specific pattern but, on each launch site we can see how the success ratio has increased.

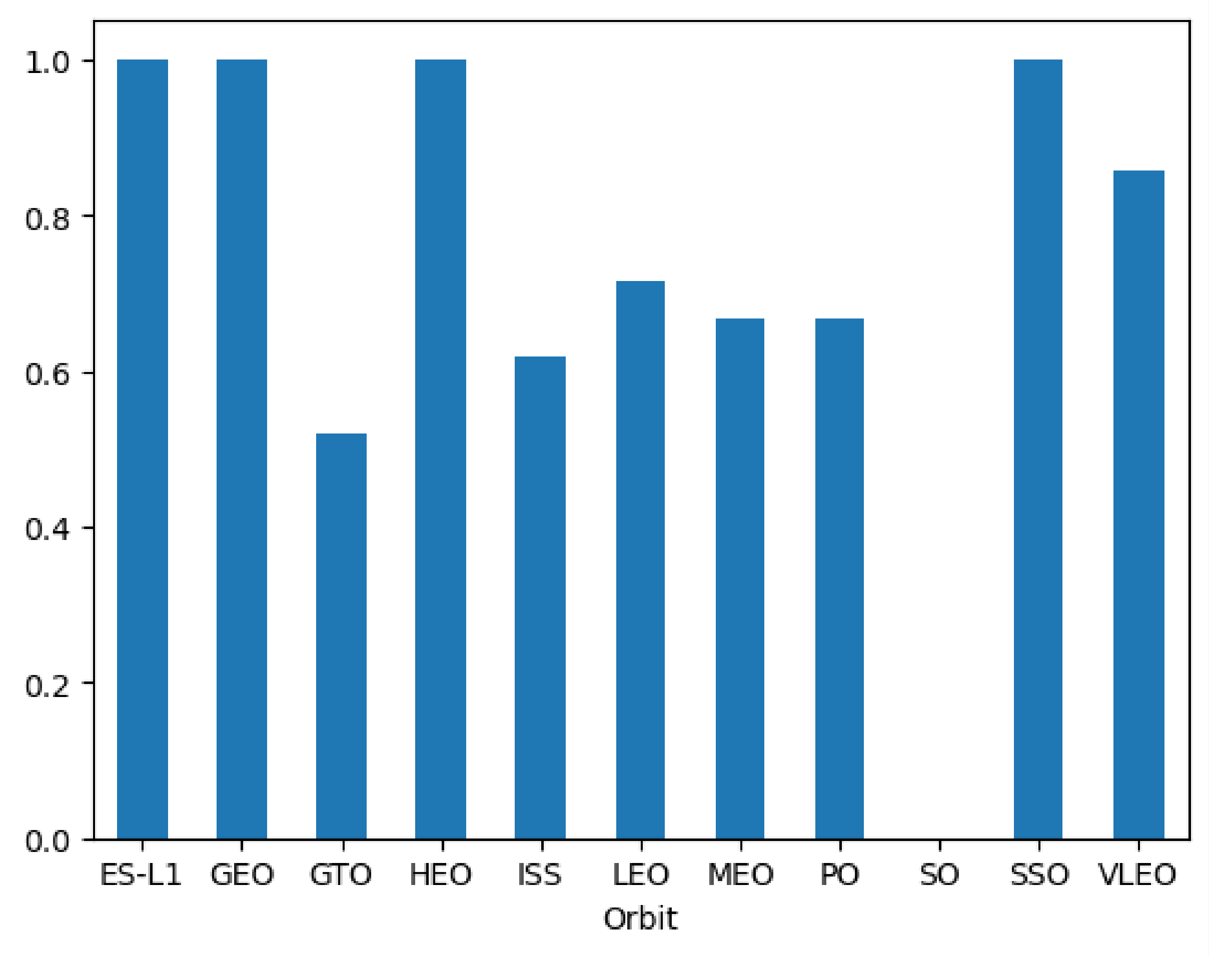
Payload vs. Launch Site



The **VAFB SLC 4E** does not launch rockets with heavy payload, while the **CCAFS SLC 40** and the **KSC LC 39A** had launched a couple times the maximum payload.

For the light payloads, the **CCAFS SLC 40** has launched more than the rest of launching sites.

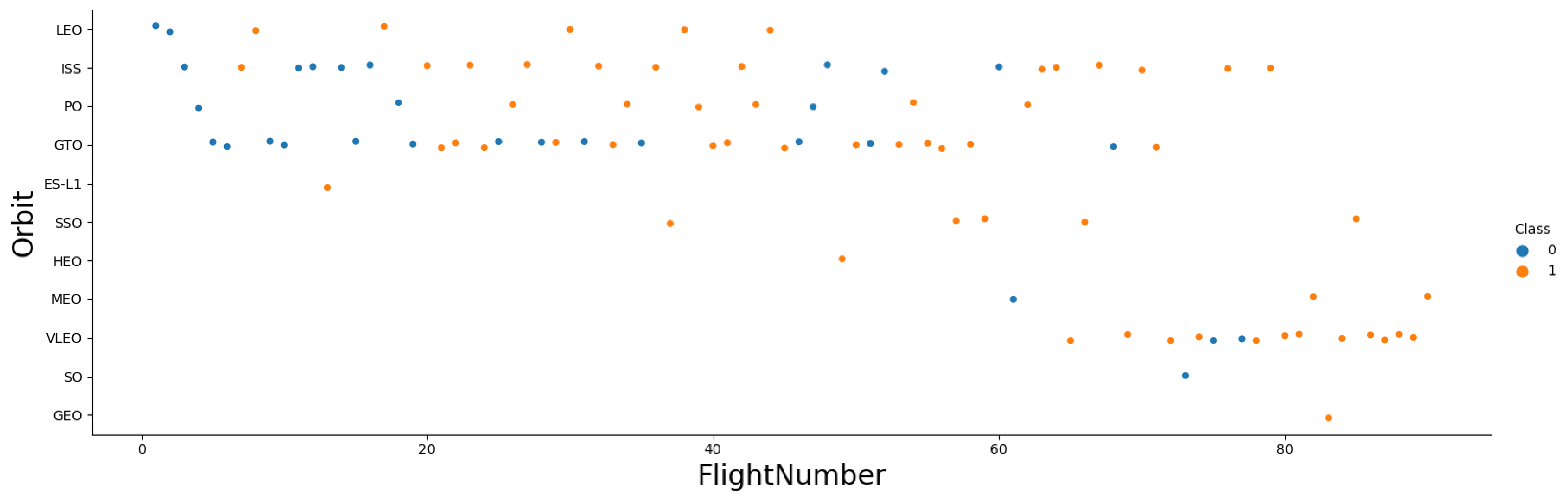
Success Rate vs. Orbit Type



The **ES-L1, GEO, HEO** and **SSO** orbits have a success rate of 100%, followed by the VLEO with more than 80% success ratio.

The rest of the orbits range between 50% and 65% of success with the exception of the SO ratio which have a 0% of success ratio.

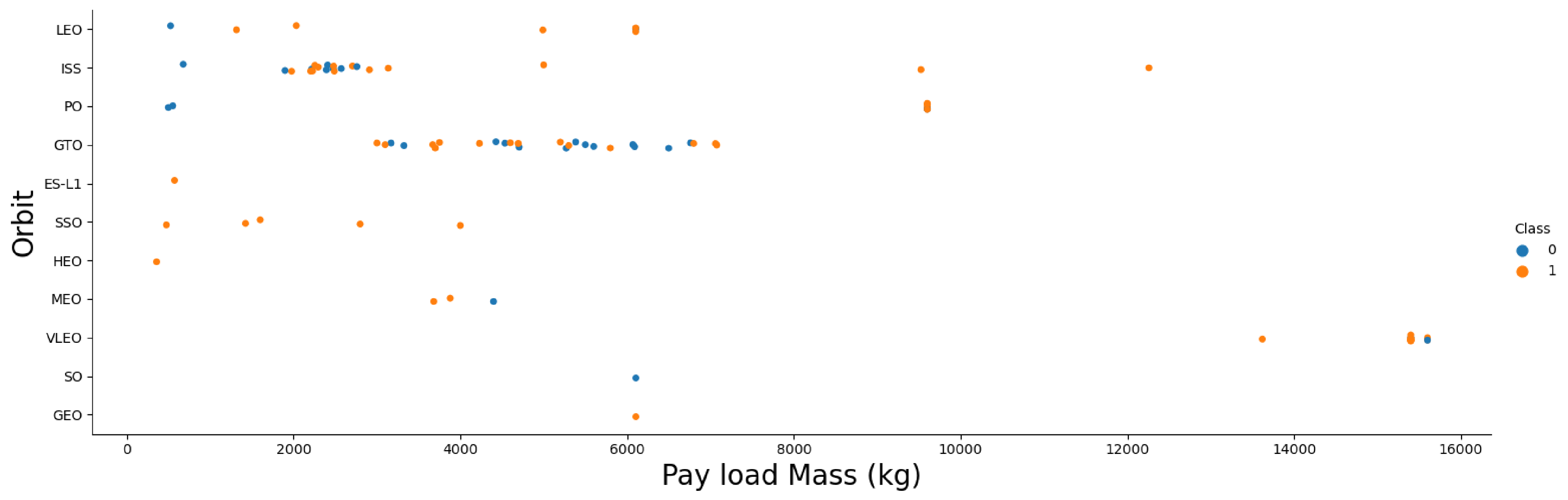
Flight Number vs. Orbit Type



The first flights were launched to the nearest orbits and, as the success ratio was increasing, the orbits selected were farther from the earth or with distinct shapes.

This scatter plot gives information about the last one were we have seen the 0% success ratio on the **SO** orbit. This ratio was because only one launch was sent to this orbit with a fail.

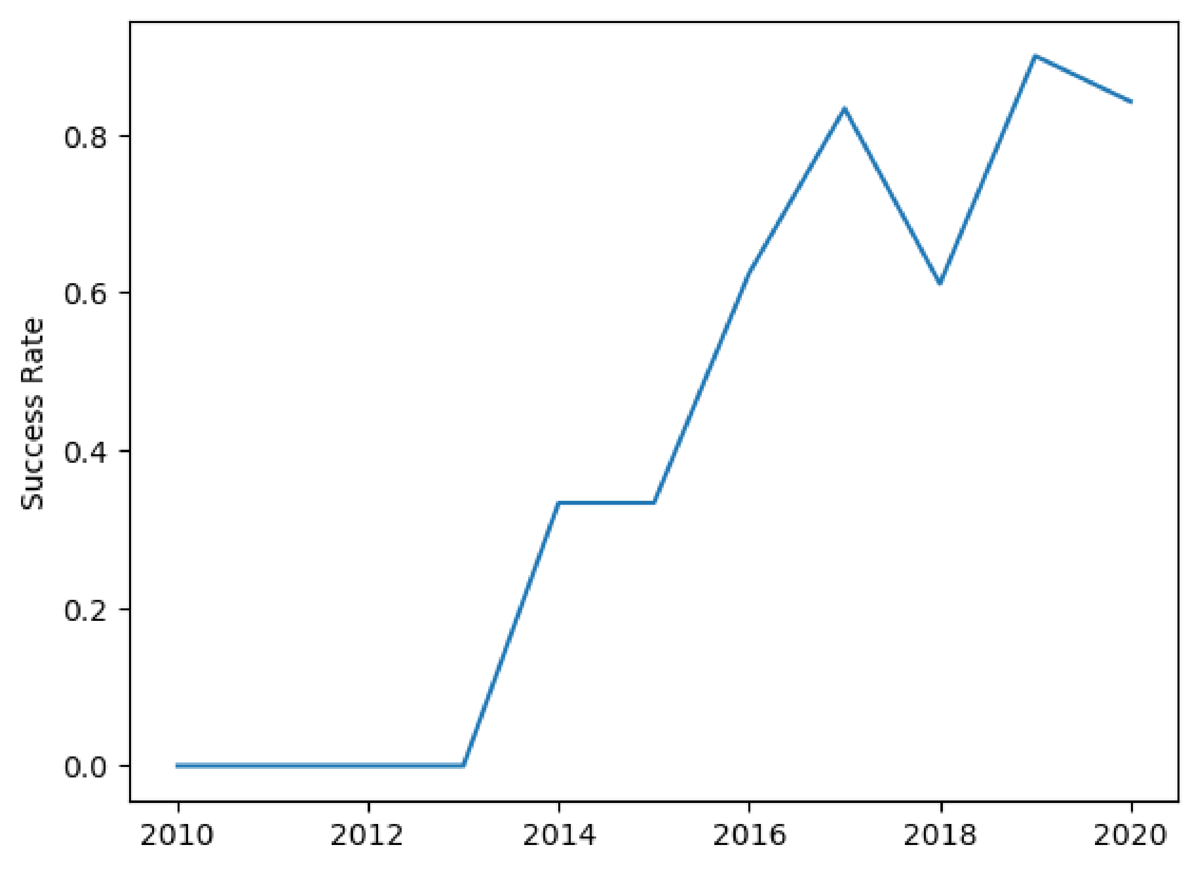
Payload vs. Orbit Type



With heavy payloads the success ratio is higher, this might be because the payload mass was increased once the "light" rockets were performing good.

Except for the **GTO** orbit where does not seem to be a success/failure clear pattern.

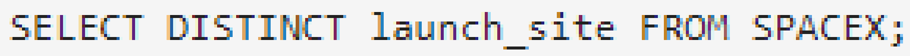
Launch Success Yearly Trend



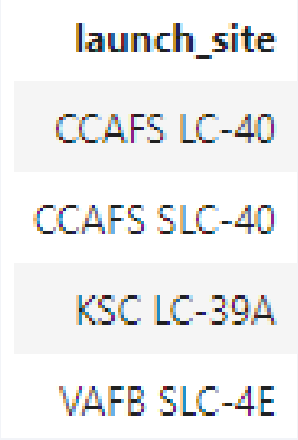
The success rate has been increasing since 2013 when the program started.

There was a little decrease over 2018 but the ratio had increased again until 2020 where it seems to be stuck on a good success rate above the 80%.

All Launch Site Names



```
SELECT DISTINCT launch_site FROM SPACEX;
```

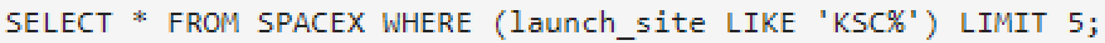


launch_site
CCAFS LC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

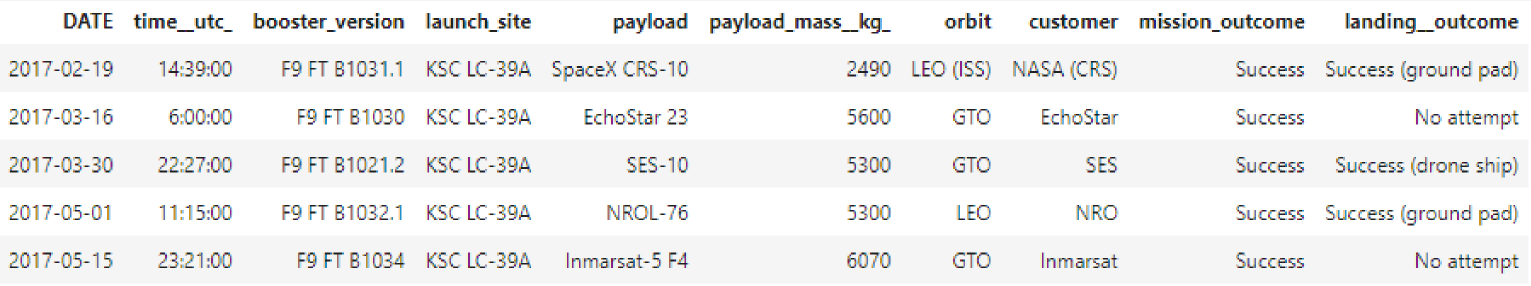
This query searches for all the unique Launch Sites by using the **DISTINCT** statement.

As we can see, there are 4 different launching sites, all presented in the table.

Launch Site Names Begin with 'KSC'



```
SELECT * FROM SPACEX WHERE (launch_site LIKE 'KSC%') LIMIT 5;
```

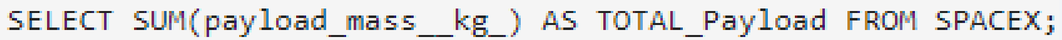


DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-03-16	6:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
2017-05-01	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
2017-05-15	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

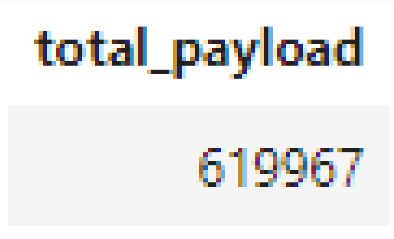
The query was programmed according to the launching sites that begin with the "KSC" string. To select similar strings, the **LIKE** statement has been used and, by writing the symbol **"%"** before the string, all the names that match with those three initial letters will be selected.

To show only 5 Site names, the **LIMIT** statement to 5 was used.

Total Payload Mass



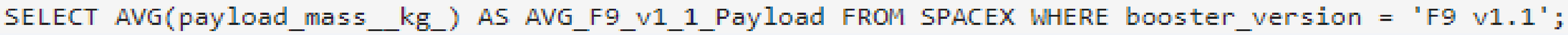
```
SELECT SUM(payload_mass__kg_) AS TOTAL_Payload FROM SPACEX;
```



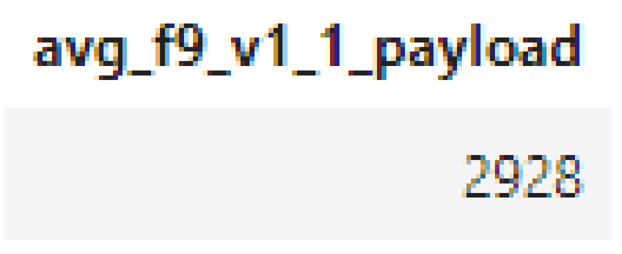
total_payload
619967

In order to find the total Payload Mass carried, the **SUM()** statement was used, this statement allow us to perform a sum of all the rows selected over a column, in this case, the payload mass column return a total of 619967 kg.

Average Payload Mass by F9 v1.1



```
SELECT AVG(payload_mass__kg_) AS AVG_F9_v1_1_Payload FROM SPACEX WHERE booster_version = 'F9 v1.1';
```

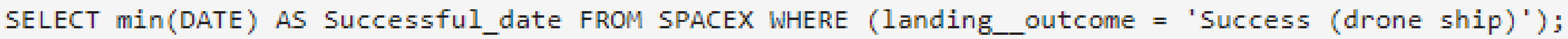


avg_f9_v1_1_payload
2928

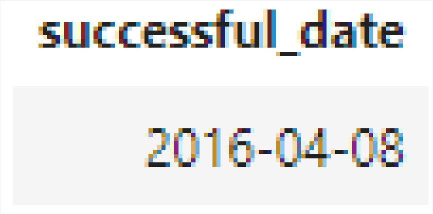
The **AVG()** statement works the same as the **SUM()** statement explained before but, in this case, the function returns the average instead of the sum of values.

The average payload mass of the F9 v1.1 booster version was 2928 kg.

First Successful Drone Ship Landing Date



```
SELECT min(DATE) AS Successful_date FROM SPACEX WHERE (landing__outcome = 'Success (drone ship)');
```

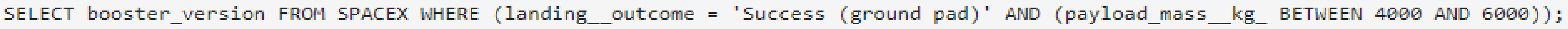


successful_date
2016-04-08

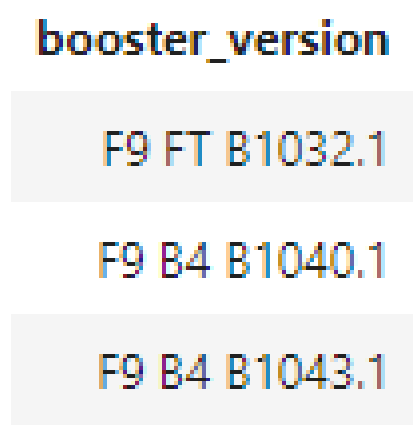
To search for the correct landing outcomes, only the "Success (drone ship)" were selected, which means we only selected the landings on drone ships that ended satisfactorily.

The **MIN()** function was used to only select the first landing over all the selected. Which was on the 2016-04-08.

Successful Ground Pad Landing with Payload between 4000 and 6000



```
SELECT booster_version FROM SPACEX WHERE (landing__outcome = 'Success (ground pad)' AND (payload_mass__kg_ BETWEEN 4000 AND 6000));
```

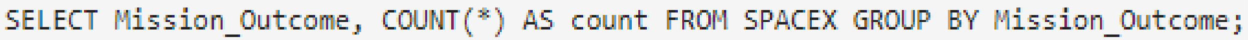


booster_version
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1

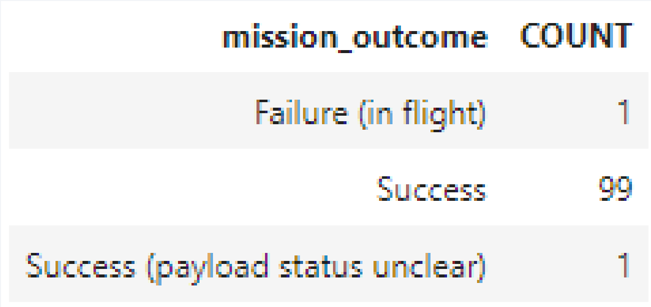
Using the same logic that on the previous query, the "Success (ground pad)" were selected but, adding one more restriction; to show only the versions where its payload is in between 4000 kg and 6000 kg.

To do this, the **BETWEEN** statement was used on the payload mass. Returning only four booster versions shown in the table.

Total Number of Successful and Failure Mission Outcomes



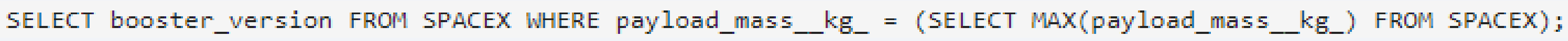
```
SELECT Mission_Outcome, COUNT(*) AS count FROM SPACEX GROUP BY Mission_Outcome;
```



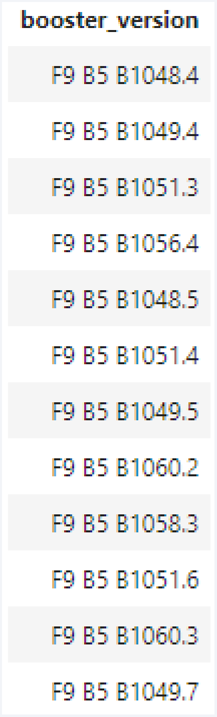
mission_outcome	COUNT
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

There were only three different mission outcomes, which includes the Failure (in flight), the Success and the Success but with an unclear payload status. Grouping by the mission outcome type and with the **COUNT()** function, that allow us to count all the different values for the same column, we reached the table shown.

Boosters Carried Maximum Payload



```
SELECT booster_version FROM SPACEX WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEX);
```



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

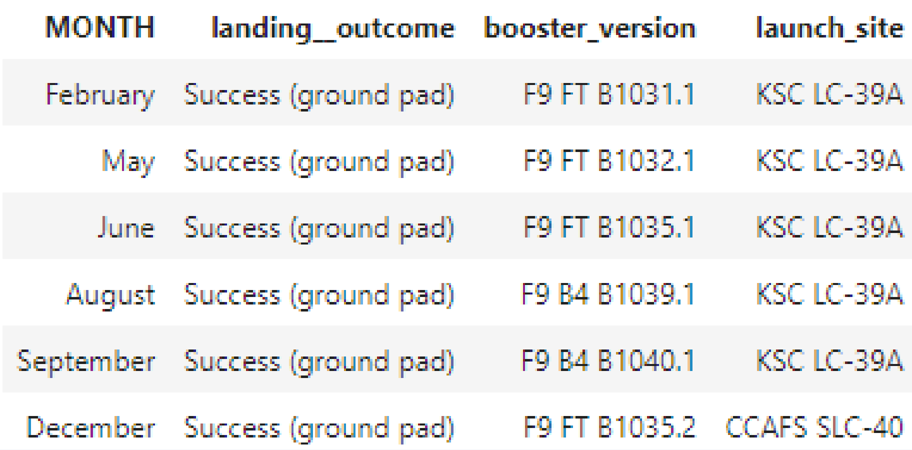
To search this information, was needed a subquery. A subquery is a SQL query nested inside a larger query.

In this case, the subquery is executed before the principal query, and it is used to know the maximum payload mass using the MAX() function.

Once the subquery has given its information, the bigger query runs and execute the rest of the code which looks for the versions carrying the maximum payload.

2017 Launch Records

```
SELECT {fn MONTHNAME(DATE)} AS MONTH, landing__outcome, booster_version, launch_site FROM SPACEX WHERE (landing__outcome = 'Success (ground pad)' AND (DATE BETWEEN '2017-01-01' AND '2018-01-01'));
```

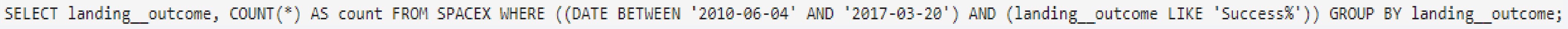


MONTH	landing__outcome	booster_version	launch_site
February	Success (ground pad)	F9 FT B1031.1	KSC LC-39A
May	Success (ground pad)	F9 FT B1032.1	KSC LC-39A
June	Success (ground pad)	F9 FT B1035.1	KSC LC-39A
August	Success (ground pad)	F9 B4 B1039.1	KSC LC-39A
September	Success (ground pad)	F9 B4 B1040.1	KSC LC-39A
December	Success (ground pad)	F9 FT B1035.2	CCAFS SLC-40

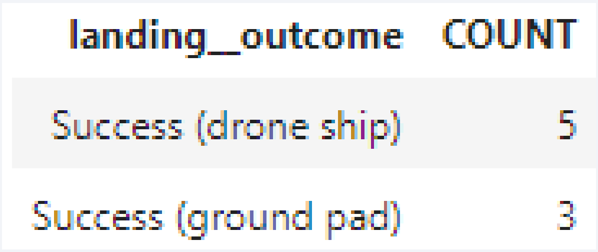
Selecting only the "Success (ground pad)" outcomes, the **BETWEEN** statement was applied to the DATE, giving only the information of the landings between the 2017-01-01 and the 2018-01-01 (all 2017).

With the **MONTHNAME()** function, the DATE has been formatted to showing only the month name instead of the DATE.

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



```
SELECT landing__outcome, COUNT(*) AS count FROM SPACEX WHERE ((DATE BETWEEN '2010-06-04' AND '2017-03-20') AND (landing__outcome LIKE 'Success%')) GROUP BY landing__outcome;
```



landing__outcome	COUNT
Success (drone ship)	5
Success (ground pad)	3

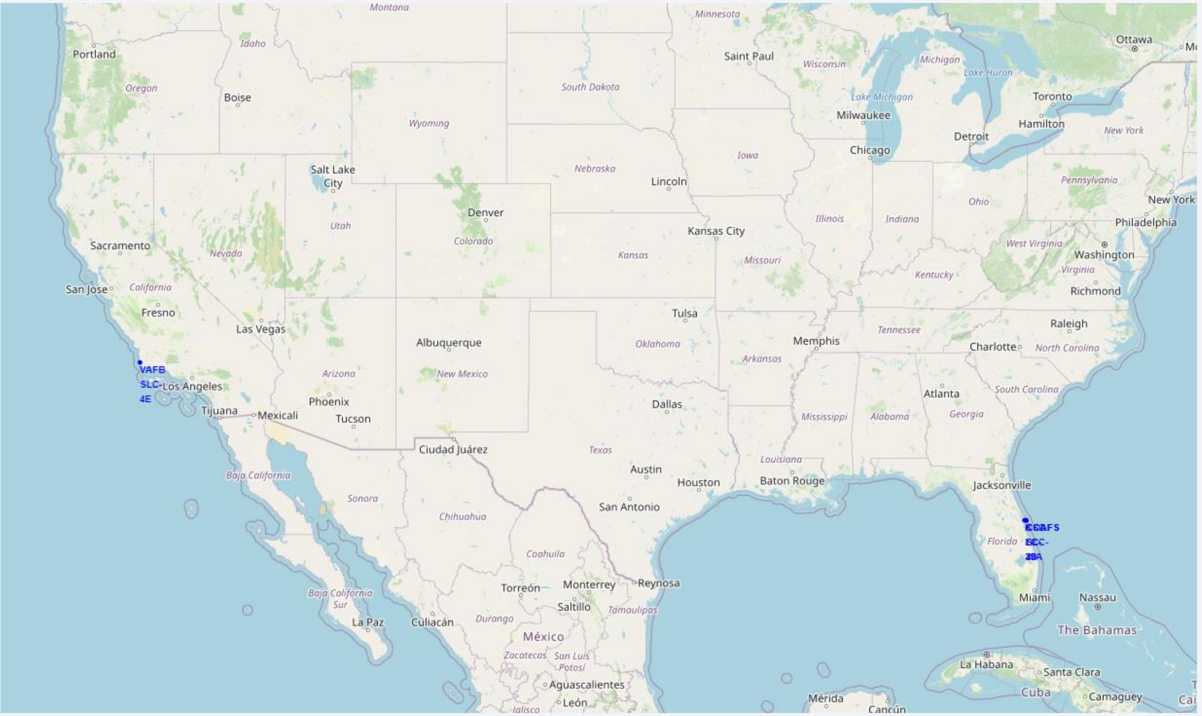
To select only the success outcomes between two dates, the **BETWEEN** statement was implemented (between dates) and the **LIKE** statement too (for all the landing outcomes that begin with Success + **%**).

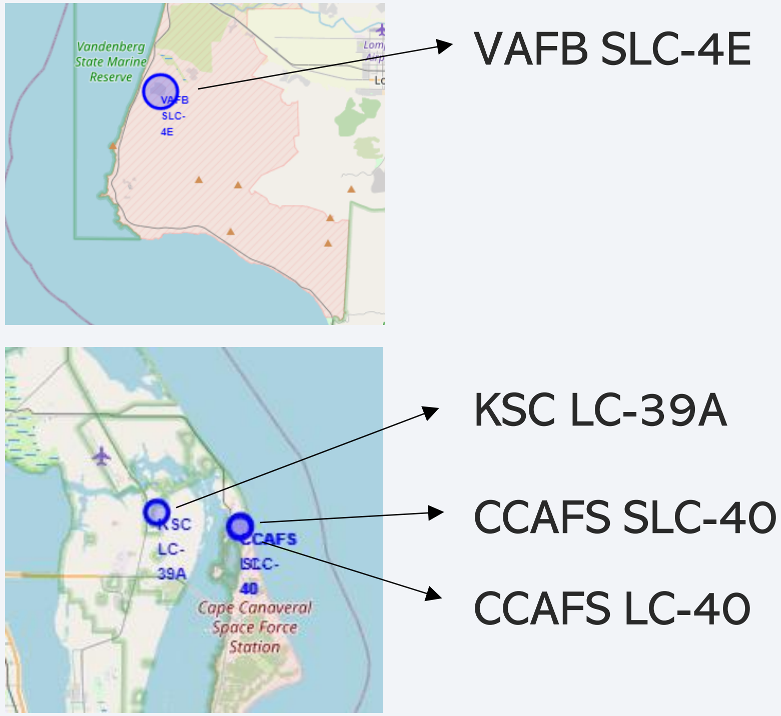
Grouping the information by the landing outcome (**GROUP BY** statement) and, with the **COUNT()** function, all the information has been ranked into 5 Success landings on drone ship and 3 Success landings on ground pad.

Section 4

Launch Sites Proximities Analysis

Launching Sites Map




Vandenberg State Marine Reserve
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
Cape Canaveral Space Force Station

VAFB SLC-4E

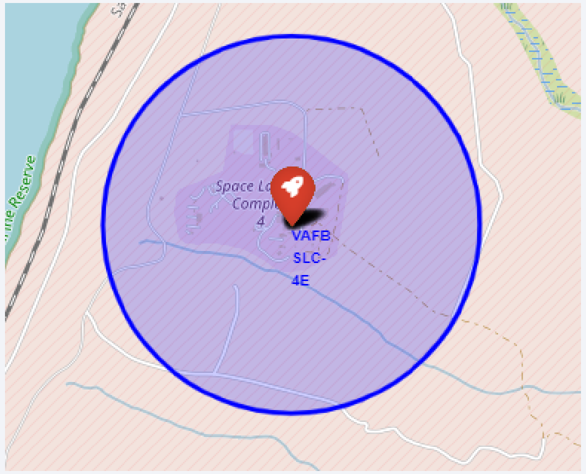
KSC LC-39A

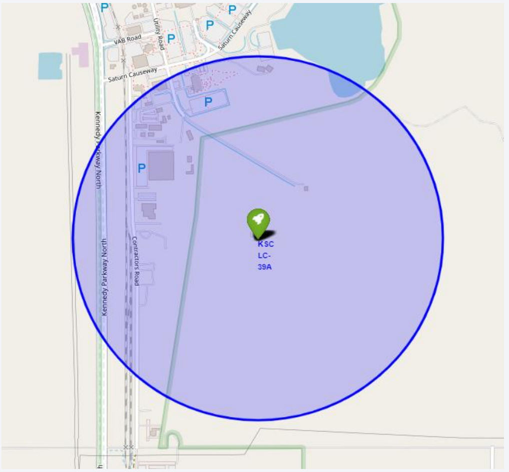
CCAFS SLC-40

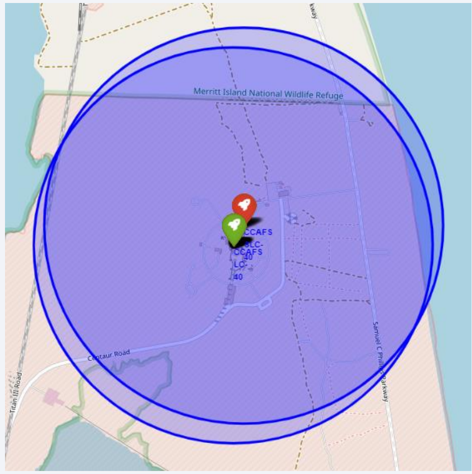
CCAFS LC-40

With the **.Circle** and **.Marker** functions and a for loop, the locations for all the launching sites were indicated on the map, with this information we can see how there are three landing sites on the East Coast (KSC LC-39A, CCAFS SLC-40 and CCAFS LC-40) while on the West Coast there is only one launching site (VAFB SLC-4E).

Launching Outcomes Map



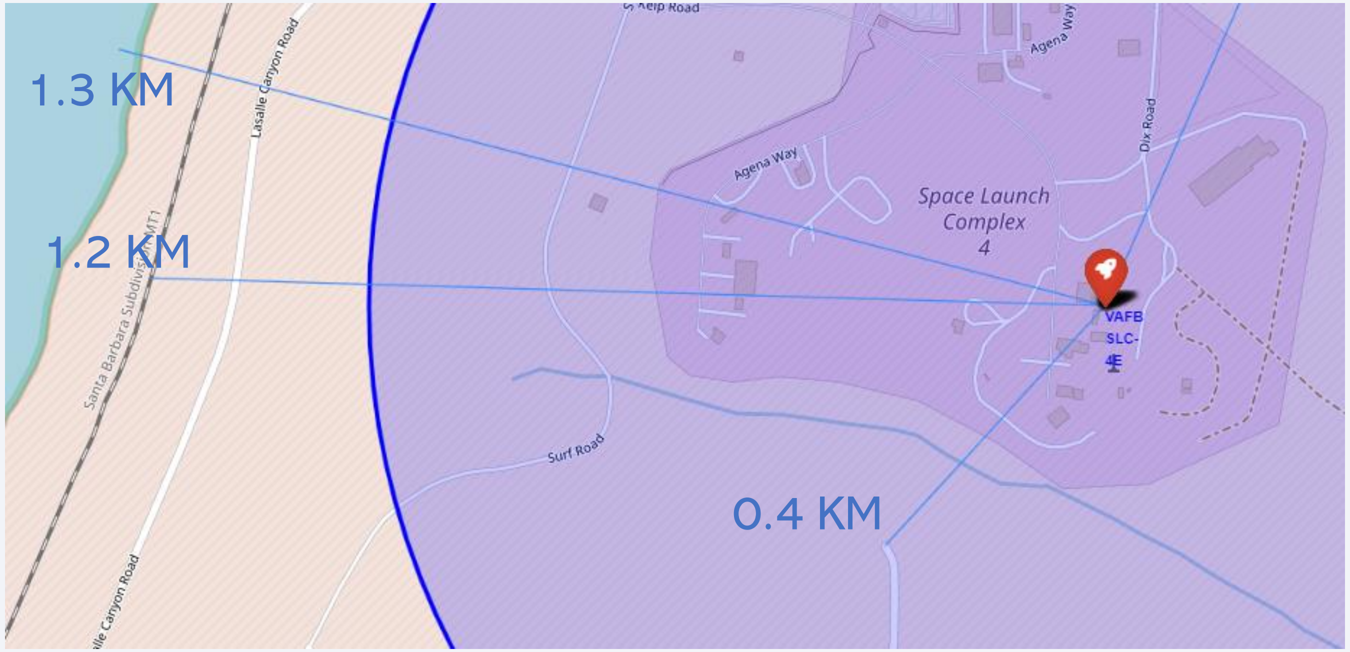


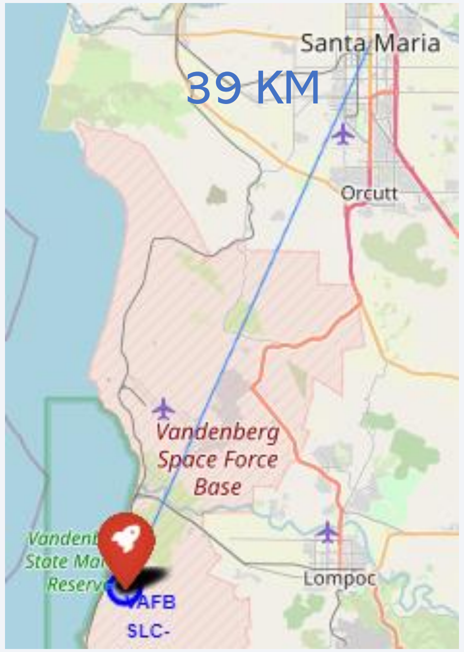


Since all the outcomes were indicated on the same Latitude and Longitude, the markers are overlayed and only one marker seems to be shown instead of all the outcomes.

In this case, the marker selected is a white rocket with a different colour depending on the success (green) or failure (red) of the mission.

Proximities Launching Sites



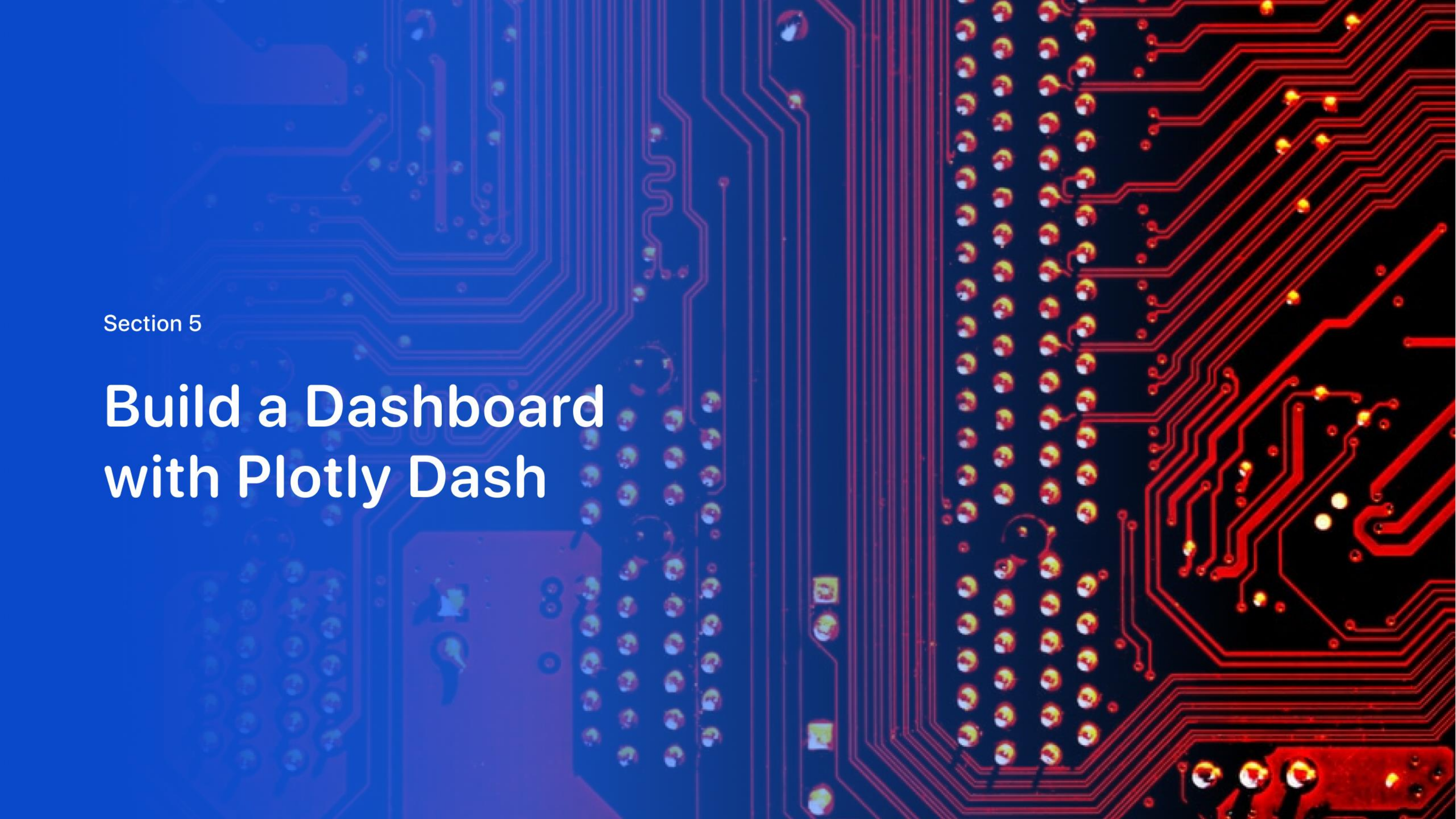


For the VAFB SLC-4E Launching site, there are 1.3 km for the nearest coastline, 1.2 km for the nearest railway, 0.4 km for the nearest highway and 39 to the nearest city.

The launching sites keep certain distance away from any human interaction because of the risk for the population if anything goes wrong. In this case, the VAFB SLC-4E is quite away of the population.

Section 5

Build a Dashboard with Plotly Dash

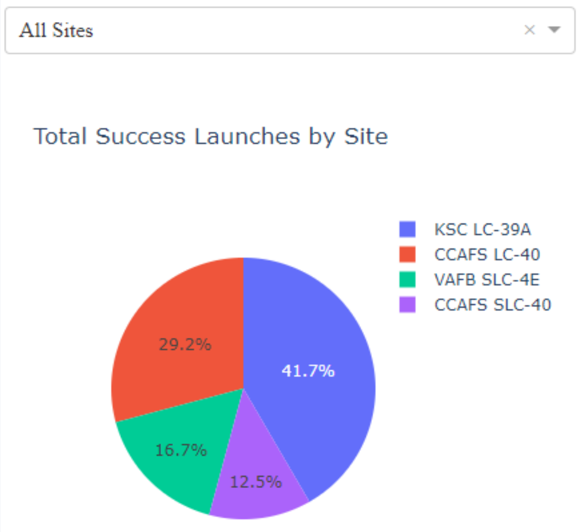


Launching Pie Chart (I)

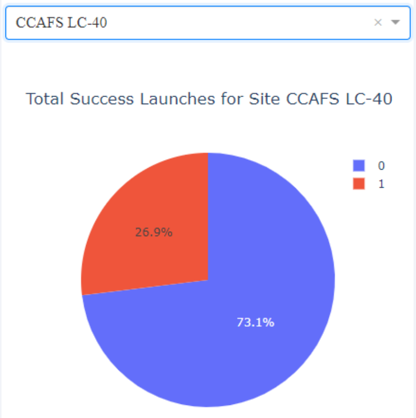
With dropdown we are able to select which is launching site we want to receive information from.

On the first case, if we select "All sites" all the success launches are shown in the pie chart but, in case we want to know of an specific site, the following 4 pie charts can be seen.

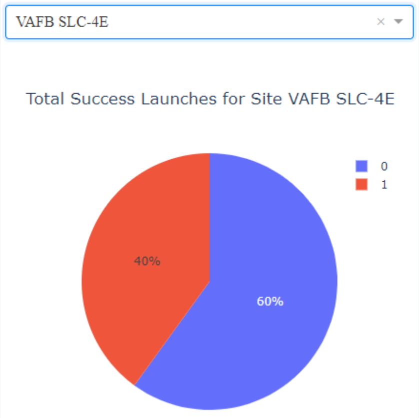
On each of these four pie charts the information about the success (red) and the failure (blue) is shown


All Sites

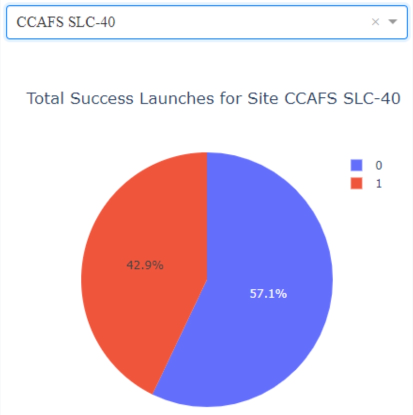
Total Success Launches by Site


CCAFS LC-40

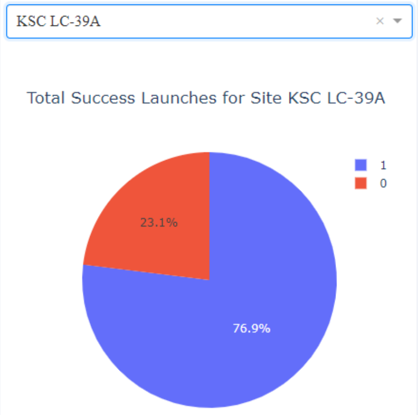
Total Success Launches for Site CCAFS LC-40


VAFB SLC-4E

Total Success Launches for Site VAFB SLC-4E

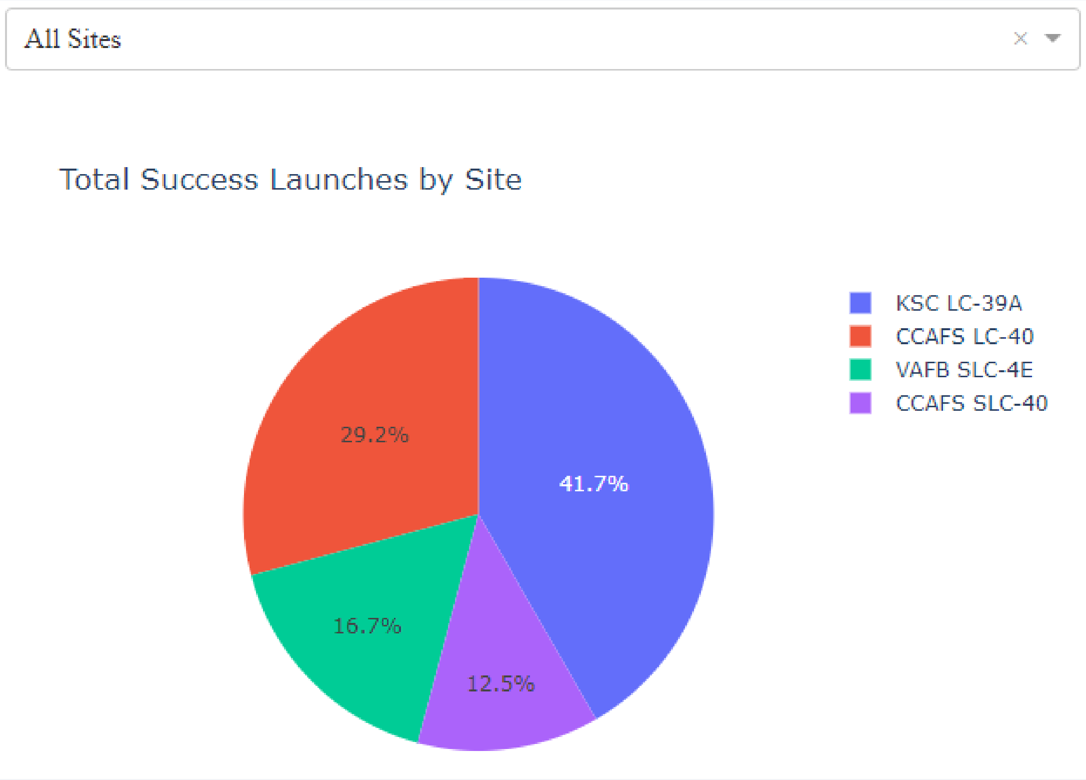

CCAFS SLC-40

Total Success Launches for Site CCAFS SLC-40


KSC LC-39A

Total Success Launches for Site KSC LC-39A

Launching Pie Chart (II)

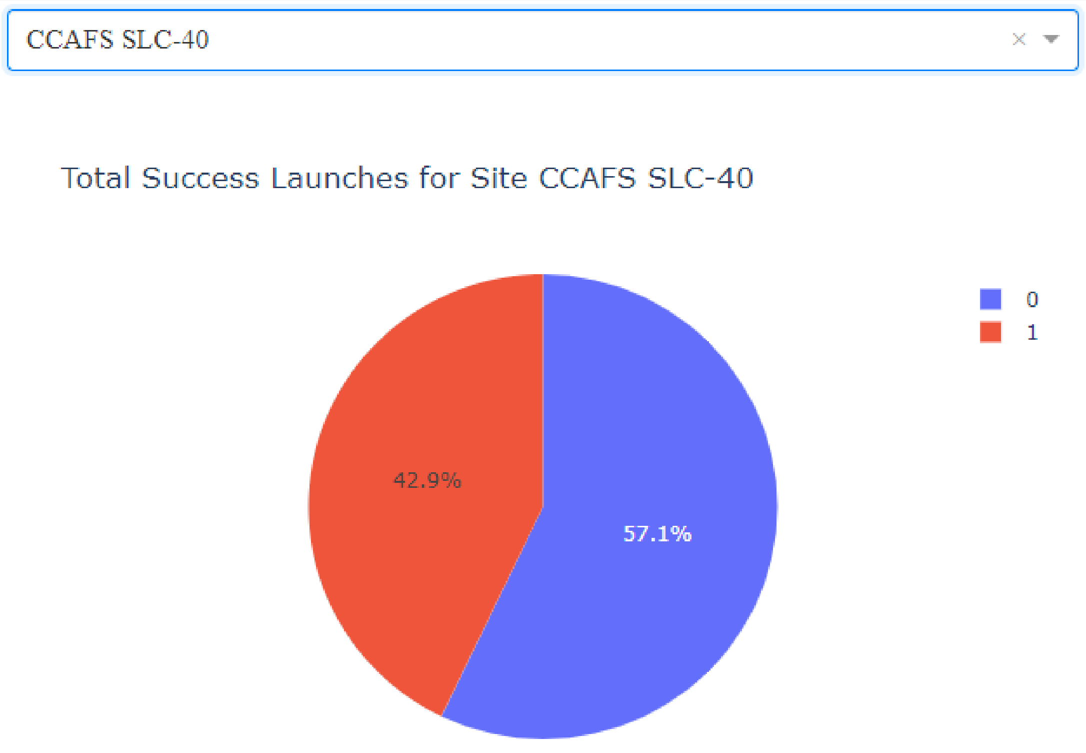


All Sites

For the "All Sites" option we can see how the KSC LC-39A is the site with the most number of success launches.

This might be because it was the last launching site used and there were a lot of launches from this site on the beginning of it usage.

Launching Pie Chart (III)

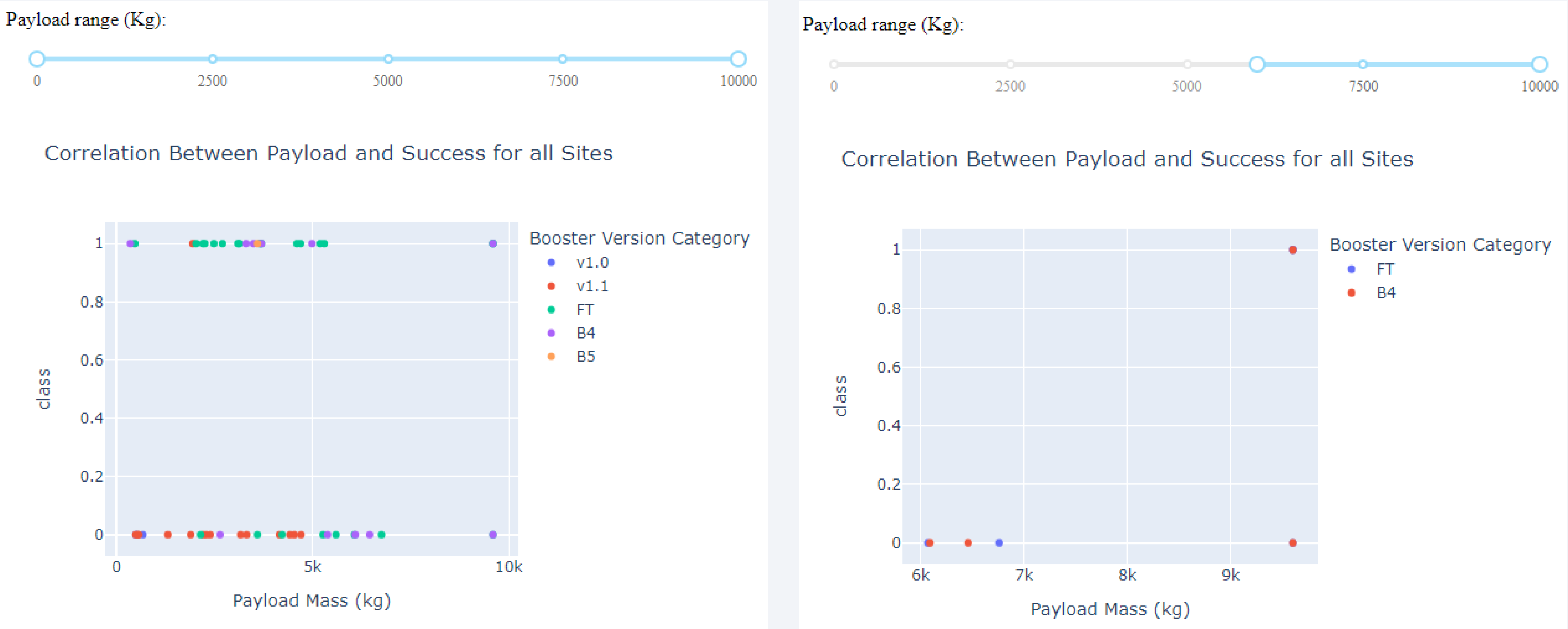


CCAFS SLC-40

The CCAFS SLC-40 is the launching site with the highest Success/Failure ratio with a 42.9%.

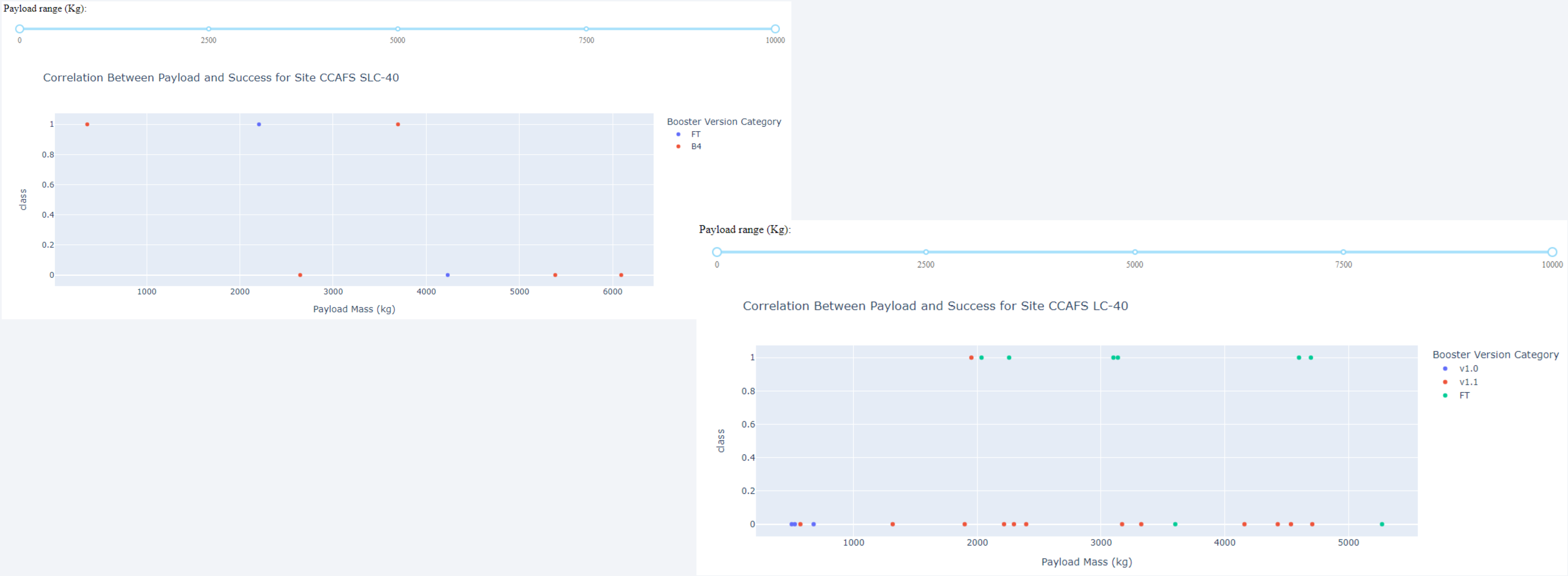
If we compare this information with the pie shown on the last slide, this site is also the site with the less number of launches of all 4.

Class vs Payload Mass scatter plot (I)



These two scatter plots give information about the ratio success and its relation with the payload mass launched. As we can see, there was a gap (zoomed in the right image) between 7000kg and 9000kg where there wasn't any launch with a payload of these dimensions.

Class vs Payload Mass scatter plot (II)



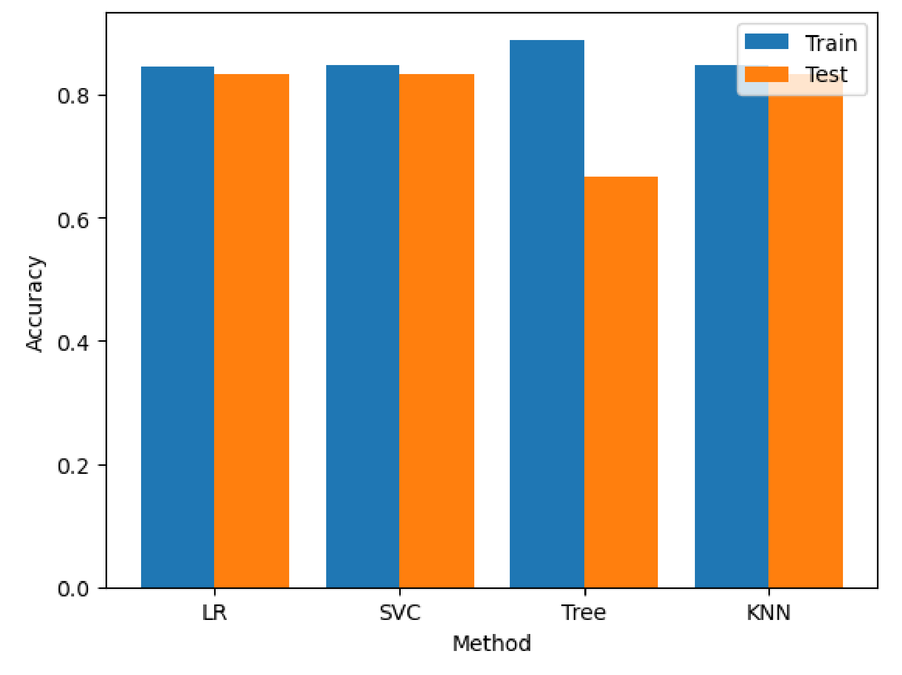
We can also see, for example, the difference between the amount of launches of two different sites. On the left, the CCAFS SLC-40 with only a few launches, against the CCAFS LC-40 on the right, with a couple more.

Section 6

Predictive Analysis (Classification)

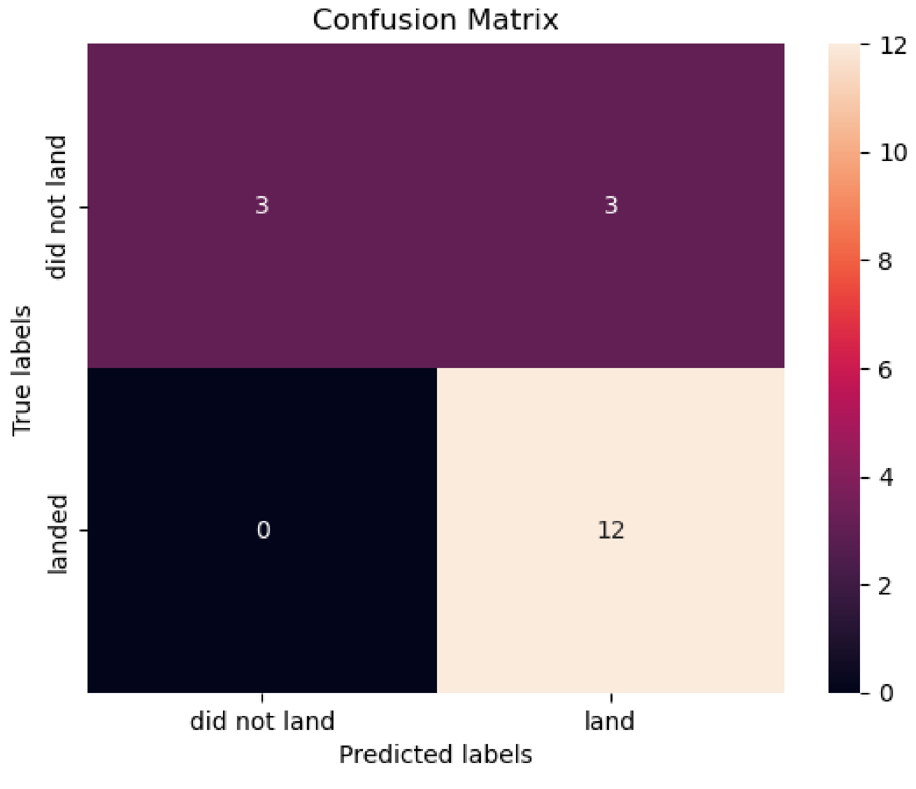
Classification Accuracy

- According to the accuracy on the trained data, the better method to perform the prediction are the Classification Trees but, according to the tested data, the LR, SVC and KNN methods returned the same.
- On the chart shown, the better method (higher average trained/tested data) is the KNN.



Confusion Matrix

The confusion matrix for the KNN method is the shown on the right. This confusion matrix means that the KNN predicted perfectly the landed data with 12/12 prediction rate but, over the not landed data, gets a 50% of probability to be right.



Conclusions

- The success rate of the landings have been increasing since 2013
- All the launching sites keep certain distance away from population
- The ES-L1, GEO, HEO and SSO orbits have a success rate against the 0% of SO
- For the light payloads, the CCAFS SLC 40 has launched more than the rest of launching sites
- The first flights were launched to the nearest orbits and, as the success ratio was increasing, the orbits selected were farther from the earth or with distinct shapes.

Appendix

- All the relevant data is appended on the submit task, that include:
- Data Collection – SpaceX API.ipynb
- Data Collection - Scraping.ipynb
- Data Wrangling.ipynb
- EDA with Data Visualization.ipynb
- EDA with SQL.ipynb
- Interactive Map with Folium.ipynb
- Dashboard with Plotly Dash.py
- Predictive Analysis (Classification).ipynb

Thank you!