

A background image of a SpaceX Falcon Heavy rocket launching at night. The rocket is positioned vertically in the center, with a large, billowing plume of fire and smoke trailing behind it. The sky is dark, and the ground is visible at the bottom. The text is overlaid on the image.

# SpaceX

Capstone Project

Professional Certificate IBM Python for Data Science

Giovanni Fiscon 22<sup>th</sup> September 2022

# Outline

---

Executive Summary

---

Introduction

---

Methodology

---

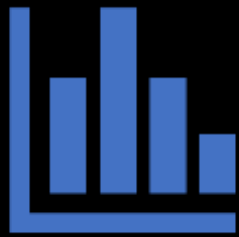
Results

---

Conclusion



# Executive Summary



## Methodologies Summary

- Data Collection using API
- Data Collection using Web Scraping
- Data Wrangling
- Exploratory Data Analysis using SQL
- Exploratory Data Analysis and Visualization
- Visual Analytics with Folium
- Machine learning Prediction



## Results Summary

- Exploratory Data Analysis result
- Dash in Screenshots
- Predictive Analytic result

# Introduction



## Target

The goal of this project is to design a Machine learning pipeline to predict if the first stage will land successfully.



## Context

SpaceX claims that Falcon 9 rocket launch cost 62 million; Other providers cost 165 million dollars. Much of the savings are because Space X can reuse the first stage. If we can determine if the first stage will land, we can determine the cost of the launch.



## Problems

Which factor can determine if the rocket will land successfully?

Which Interaction between various features can determine the success rate of a successful landing?

What conditions need to be ensured for a successful landing?

# Methodology

- Data Collection using API
- Data Collection using Web Scraping
- Data Wrangling
- Exploratory Data Analysis using SQL
- Exploratory Data Analysis and Visualization
- Visual Analytics with Folium
- Machine learning Prediction

# The data was collected using various methods

- Data collection was done using get request to the SpaceX API.
- Next, the response content as a Json using `.json()` function call was decoded and turned into a pandas data frame using `.json_normalize()`.
- Data were cleaned and checked for missing values and fill in missing values where necessary.
- In addition, we performed web scraping from Wikipedia for Falcon 9 launch records with BeautifulSoup.
- The objective was to extract the launch records as an HTML table, parse the table and convert it to a pandas data frame for future analysis.

# Data Collection – SpaceX API

```
1 Task 2: Filter the dataframe to only include Falcon 9 launches

1 data_falcon9=df.loc[(df['BoosterVersion']!='Falcon 1')]
2 data_falcon9

1 # Now that we have removed some values we should reset the FlightNumber column
2 pd.options.mode.chained_assignment = None
3 data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
4 data_falcon9

1 Task 3: Dealing with Missing Values
2 Calculate below the mean for the PayloadMass using the .mean(). Then use the mean and the .replace() function to
replace np.nan values in the data with the mean you calculated.

1 data_falcon9.isnull().sum()

1 # Calculate the mean value of PayloadMass column
2 x=data_falcon9['PayloadMass'].mean()
3 # Replace the np.nan values with its mean value on PayloadMass only
4 data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, x)
```

```
1 Task 1: Request and parse the SpaceX launch data using the GET request

1 static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-Skills
<

1 Use json_normalize method to convert the json result into a dataframe

1 data=pd.json_normalize(response.json())

1 The data have been cleaned:
```

- We used the get request to the SpaceX API to collect data:
- Clean the requested data and do some basic data wrangling and formatting.
- The link to the notebook is <https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/jupyter-labs-spacex-data-collection-api.ipynb>



# Data Collection Scraping

```
1 column_names = []
2
3 temp = first_launch_table.find_all('th')
4
5 for x in range(len(temp)):
6     name=extract_column_from_header(temp[x])
7     if name is not None and len(name)>0:
8         column_names.append(name)
```

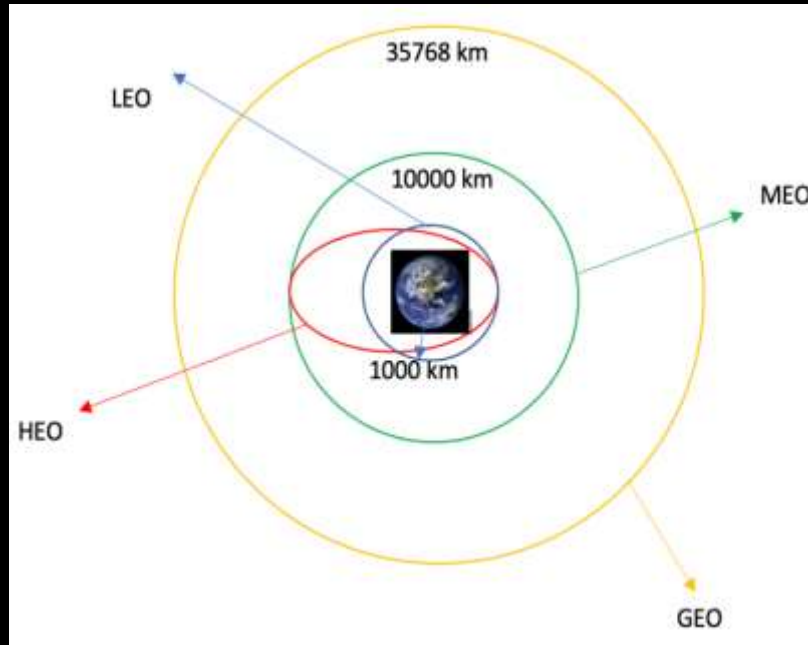
```
1 launch_dict= dict.fromkeys(column_names)
2
3 # Remove an irrelevant column
4 del launch_dict['Date and time ( )']
5
6 # Let's initial the launch_dict with each value to be an empty list
7 launch_dict['Flight No.'] = []
8 launch_dict['Launch site'] = []
9 launch_dict['Payload'] = []
10 launch_dict['Payload mass'] = []
11 launch_dict['Orbit'] = []
12 launch_dict['Customer'] = []
13 launch_dict['Launch outcome'] = []
14 # Added some new columns
15 launch_dict['Version Booster']=[ ]
16 launch_dict['Booster landing']=[ ]
17 launch_dict['Date']=[ ]
18 launch_dict['Time']=[ ]
```

```
1 TASK 1: Request the Falcon Launch site page from its URL.
2
3 status_code = "https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches"
4 response=requests.get(status_code).text
5
6 Use BeautifulSoup() to create a BeautifulSoup object from a response text content
7
8 soup=BeautifulSoup(response, 'HTML5LIB')
9
10 TASK 2: Extract all column/variable names from the HTML table header
11
12 first_table=soup.find_all('table')
```

- We applied web scrapping to web scrap Falcon 9 launch records with BeautifulSoup
- We parsed the table and converted it into a pandas data frame.
- The link to the notebook is <https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/jupyter-labs-webscraping.ipynb>



# Data Wrangling



CAPE CANAVERAL SPACE LAUNCH COMPLEX 40  
KENNEDY SPACE CENTER LAUNCH COMPLEX 39A  
VANDENBERG AIR FORCE BASE SPACE LAUNCH COMPLEX 4E

- We performed exploratory data analysis and determined the training labels.
- We calculated the number of launches at each site, and the number and occurrence of each orbit
- We created a landing outcome label from the outcome column and exported the results to CSV.
- The link to the notebook is <https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb>

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

CCAFS SLC 40	55
KSC LC 39A	22
VAFB SLC 4E	13

Name: LaunchSite, dtype: int64

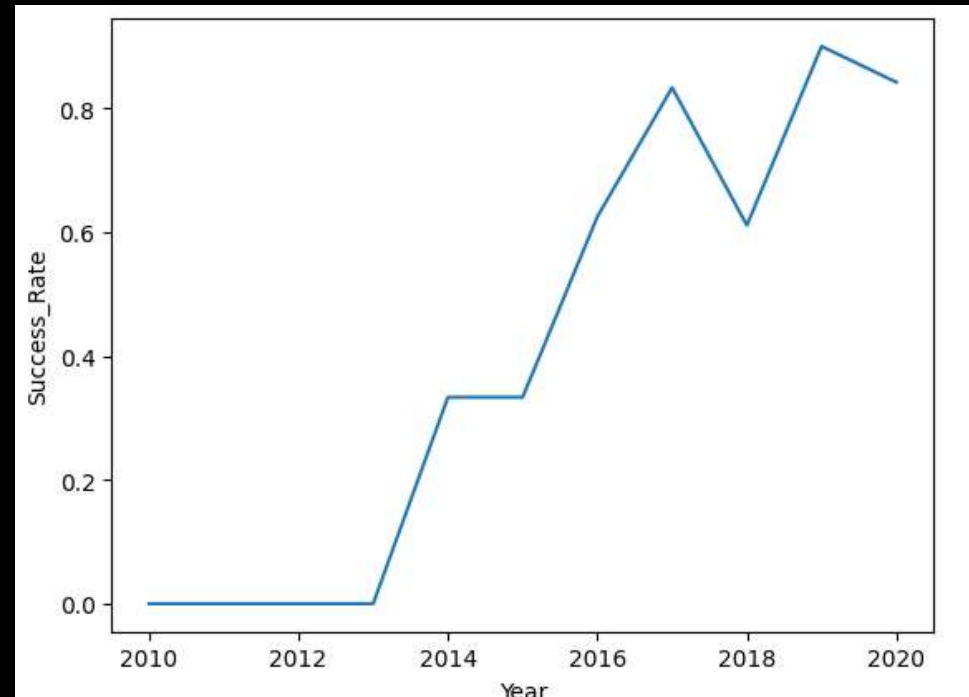
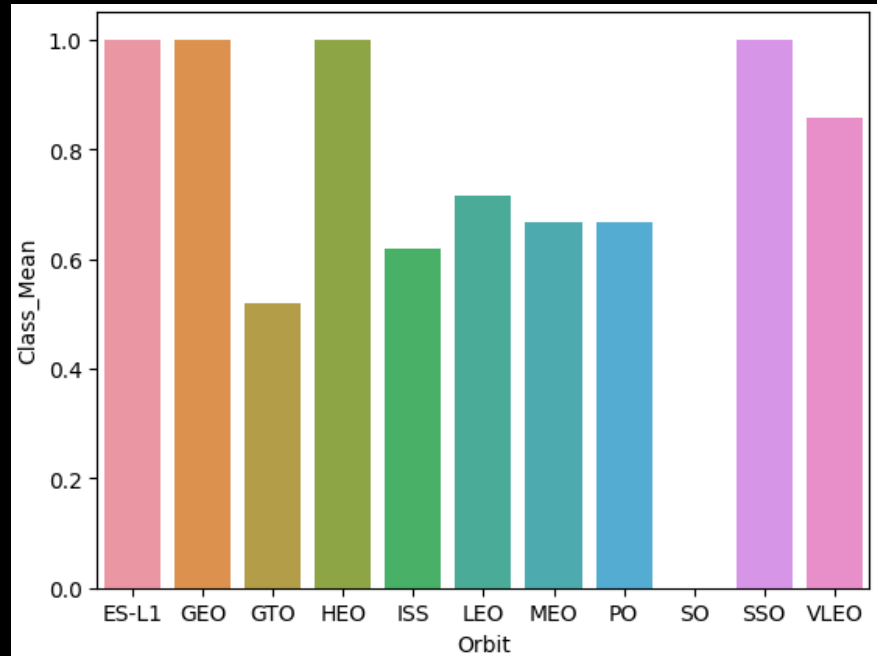
```
1 df['Class']=landing_class
2 df[['Class']].value_counts()
```

Class	
1	60
0	30

dtype: int64

# EDA with Data Visualization

- We explored the data by visualizing the relationship between flight number and launch Site, payload and launch site, the success rate of each orbit type, flight number and orbit type, and the launch success yearly trend.



The link to the notebook is <https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/jupyter-labs-eda-dataviz.ipynb>

# EDA with SQL

- We loaded the SpaceX dataset into a PostgreSQL database without leaving the jupyter notebook.
- We applied EDA with SQL to get insight from the data. We wrote queries to find out for instance:
  - The names of unique launch sites in the space mission.
  - The total payload mass carried by boosters launched by NASA (CRS)
  - The average payload mass carried by booster version F9 v1.1
  - The total number of successful and failure mission outcomes
  - The failed landing outcomes in drone ship, their booster version and launch site names.
- The link to the notebook <https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/eda-sql-edx.ipynb>

# Build an Interactive Map with Folium

- We marked all launch sites and added map objects such as markers, circles, and lines to mark the success or failure of launches for each site on the folium map.
- We assigned the feature launch outcomes (failure or success) to classes 0 and 1.
- 0 for failure and 1 for success.
- Using the color-labeled marker clusters, we identified which launch sites have a relatively high success rate.
- We calculated the distances between a launch site to its proximities. We answered some questions, for instance:
  - Are launch sites near railways, highways, and coastlines?
  - Do launch sites keep a certain distance away from cities?
- [https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/lab\\_jupyter\\_launch\\_site\\_location.ipynb](https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/lab_jupyter_launch_site_location.ipynb)

# Built a Dashboard with Plotly Dash

- An interactive dashboard with Plotly dash was built
- We plotted pie charts showing the total launches by specific sites
- We plotted a scatter graph showing the relationship between Outcome and Payload Mass (Kg) for the different booster versions.
- The link to the notebook is [https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/spacex\\_dash.ipynb](https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/spacex_dash.ipynb)

# Predictive Analysis Classification

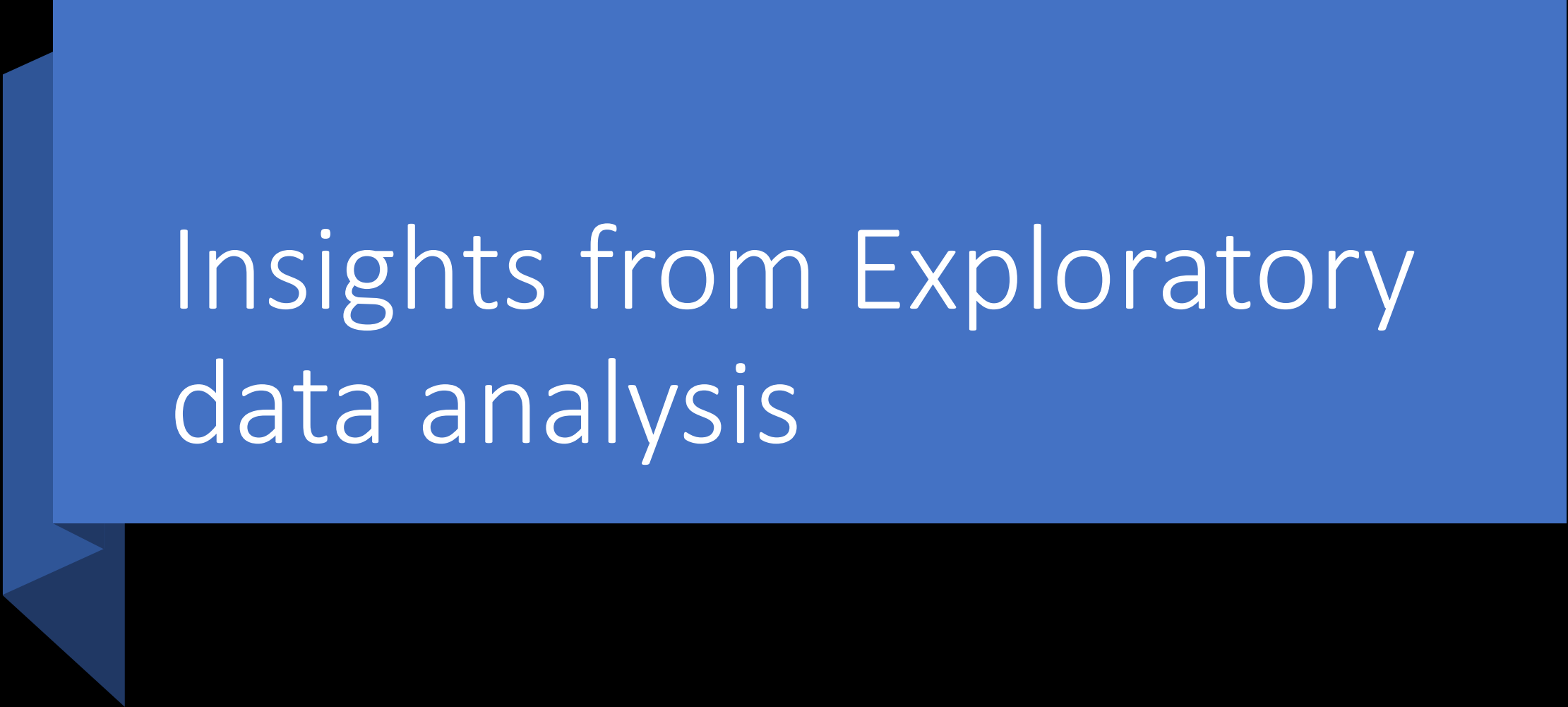
- Data have been loaded using NumPy and pandas, transformed, and split into training and testing.
- We built different machine learning models and tuned different hyperparameters using GridSearchCV.
- We used accuracy as the metric for our model and improved the model using feature engineering and algorithm tuning.
- We found the best performing classification model.
- The link to the notebook is [https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/SpaceX Machine%20Learning%20Prediction.ipynb](https://github.com/GioFis/Data-Science-and-Machine-Learning-Capstone-Project/blob/main/SpaceX%20Machine%20Learning%20Prediction.ipynb)

# Results

- EDA results
- Interactive Analytics demo in screenshots
- Predictive analysis results



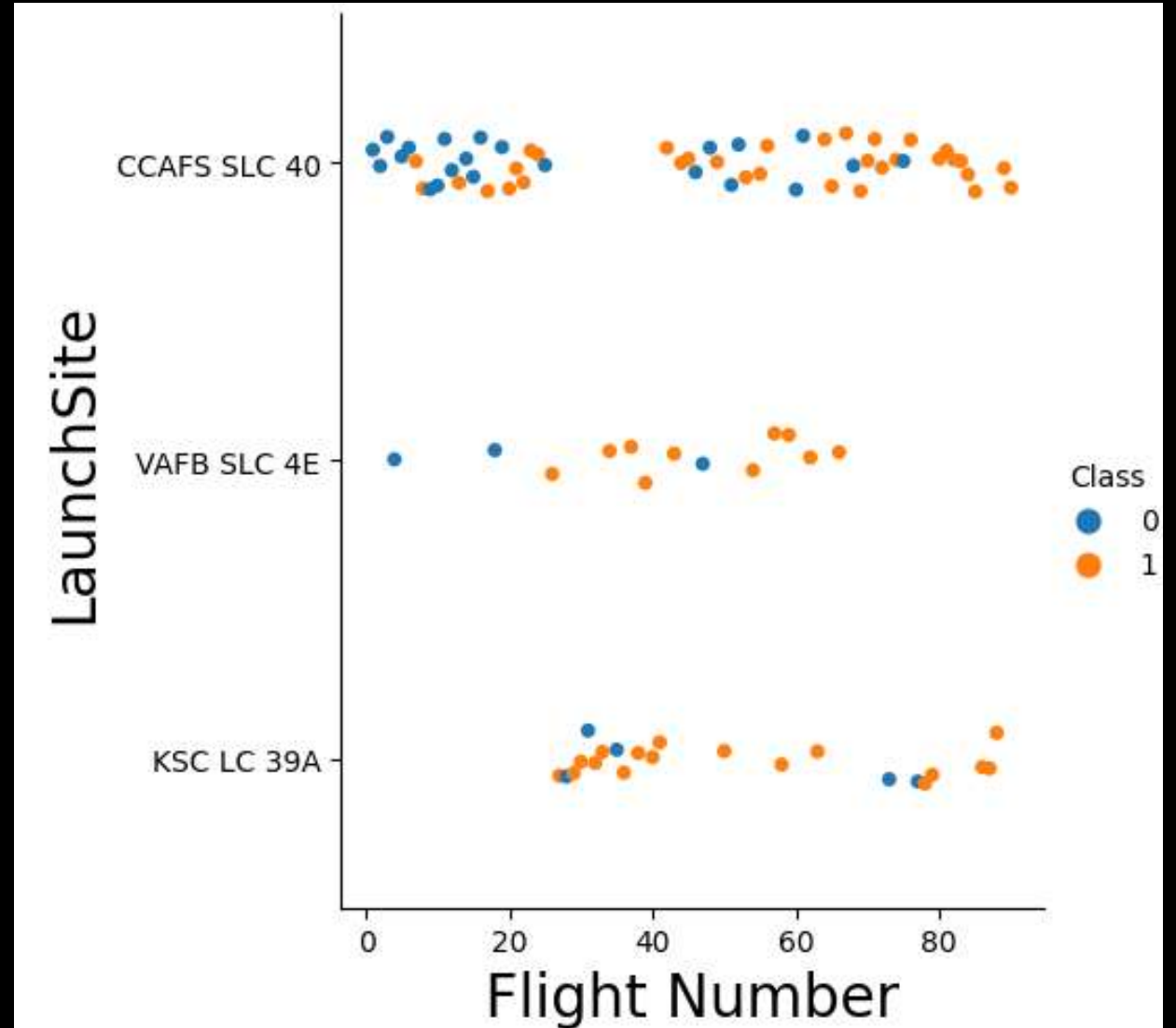


A blue rectangular box with a 3D effect, featuring a darker blue shadow on its left side. The box is centered on a black background.

# Insights from Exploratory data analysis

# Flight Number vs. Launch Site

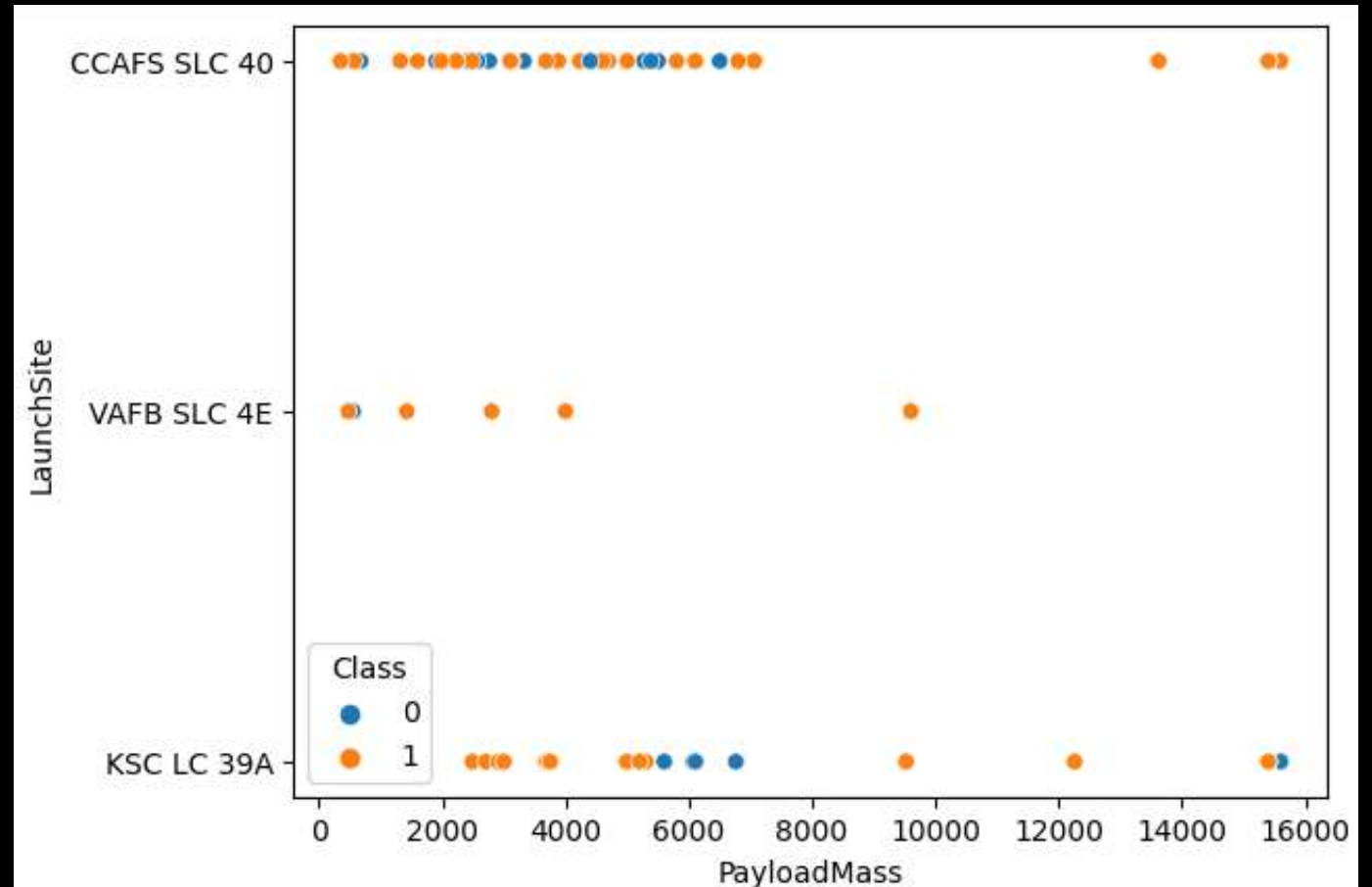
From this graph, we can deduce that CCAFS SLC 40 was the most used launch site, the progressively rare number of blue dots (failure) suggest a progressive improvement over time



# Payload vs Launch Site

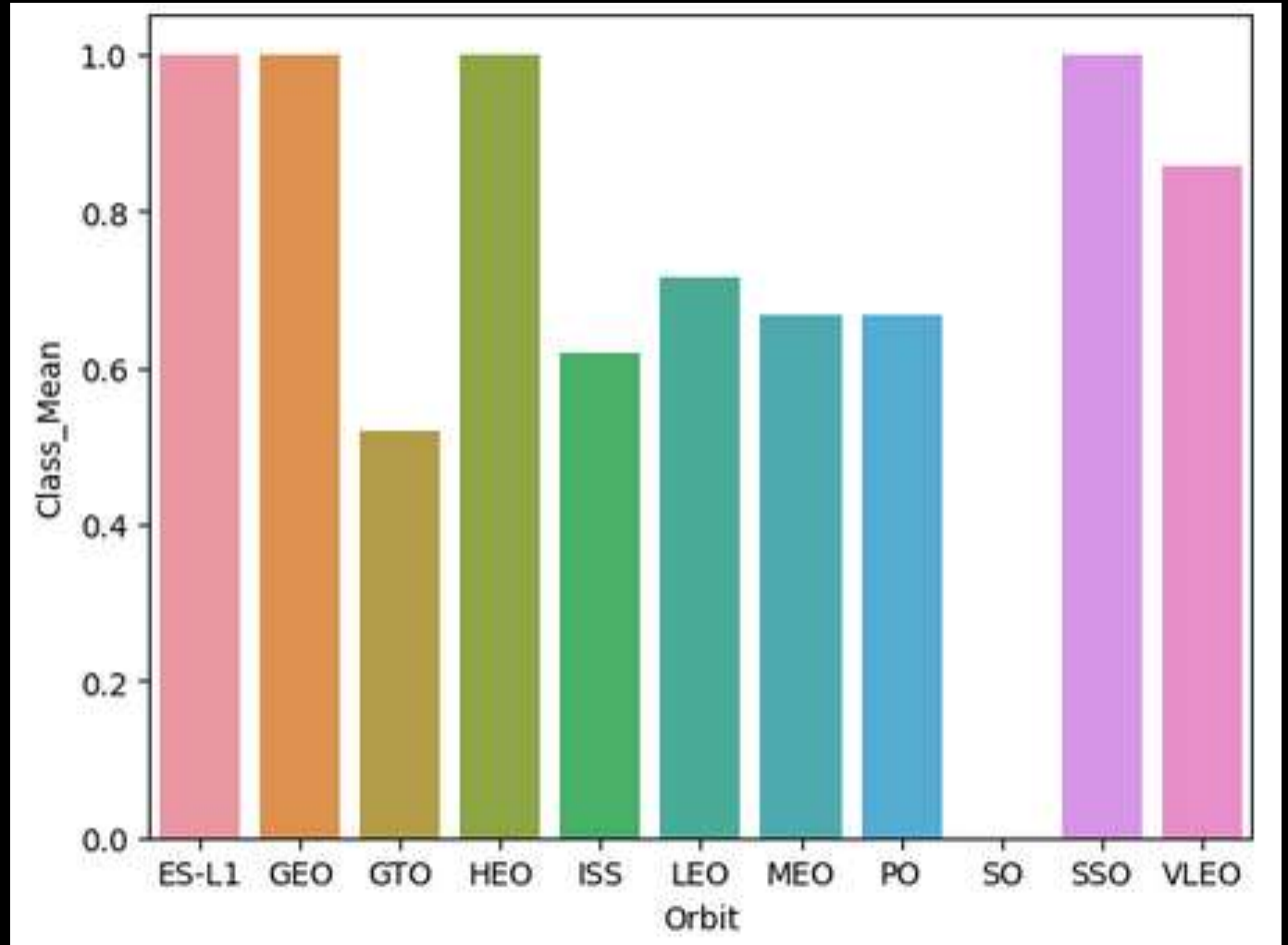
With a closer look at the graph, we can see that:

- payload mass >8k kg has a high success rate in all launch sites,
- $2k < \text{Payload mass} < 8k$  Kg. We have a significant concentration of failure
- VAFB-SLC launch site, there are no rockets launched for heavy payload mass (greater than 10000)



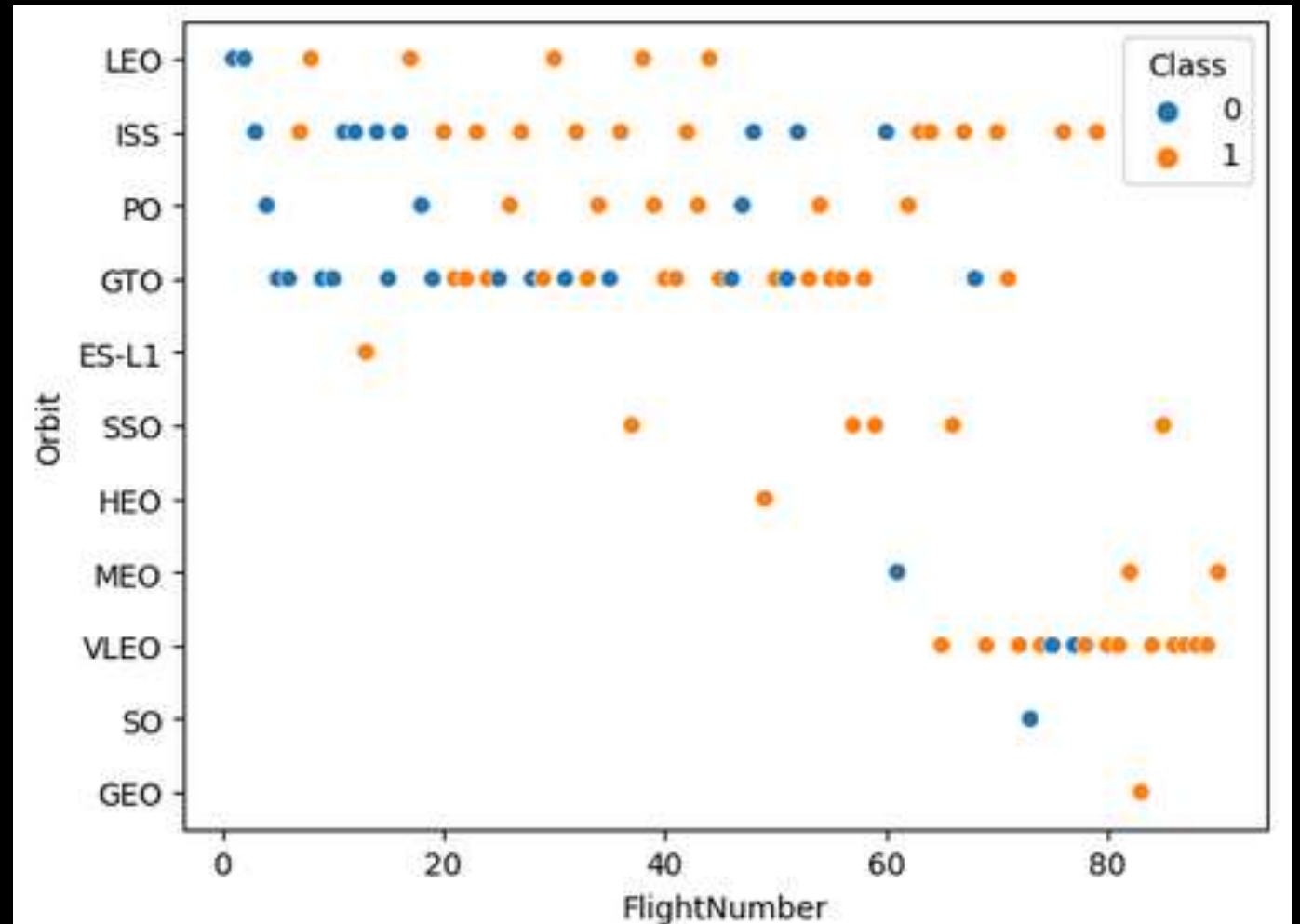
# Success Rate vs Orbit Type

ES-L1, GEO, HEO, SSO, and VLEO had the most success rate, as we can see on this graph



# Flight Number vs Orbit Type

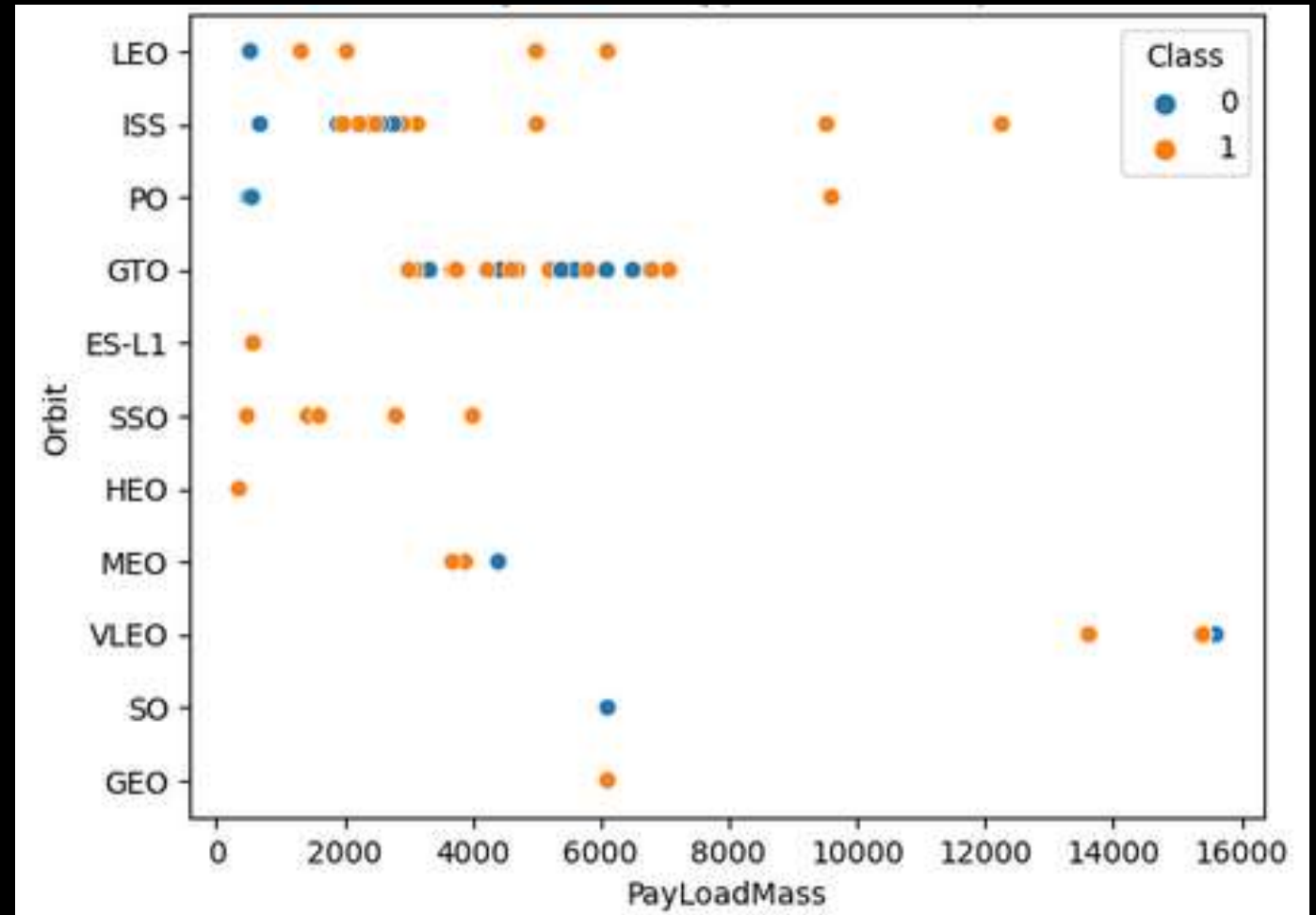
- We observe that 51% of the 27 GTO Flight numbers have been successful
- LEO 71.5%
- ISS 62%
- VLEO 86%
- ES-L1 and SSO performed well ( 100% successful ) but a fewer number of launches were committed to reach those orbits



# Payload vs Orbit Type

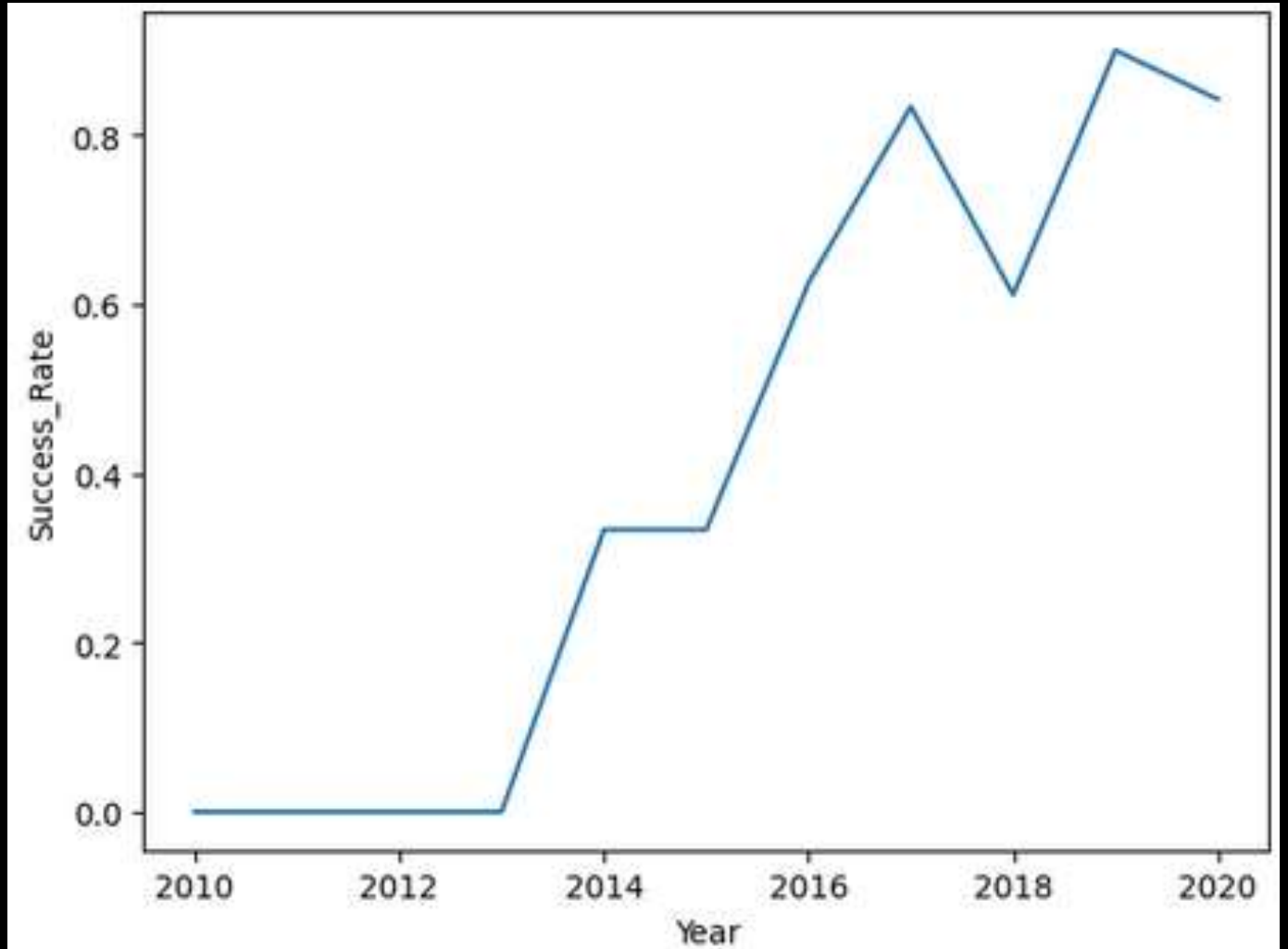
Successful landings of heavy payloads (more than 6000Kg) are more for LEO, ISS, and GTO orbits

A large part of the payload weight per launch was less than 6000Kg



## Launch Success Yearly Trend

Since 2013, the success rate kept on increasing till 2020, as we can see from this plot





the key word DISTINCT to  
show only unique launch  
sites

## All Launch Site Names

```
1 %sql SELECT DISTINCT(launch_site),COUNT(*) as count FROM SPACEXTBL GROUP BY launch_site;  
* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8lcg.databases.  
Done.
```

launch_site	COUNT
CCAFS LC-40	26
CCAFS SLC-40	34
KSC LC-39A	25
VAFB SLC-4E	16

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

```
1 %sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE '%KSC%' LIMIT 5;
```

\* ibm\_db\_sa://fbt48346:\*\*\*@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90108kqb1od81cg.databases.appdomain.cloud:30367/BLUDB  
Done.

column_0	DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
29	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)
30	2017-03-16	06:00:00	F9 FT B1030	KSC LC-39A	EchoStar 23	5600	GTO	EchoStar	Success	No attempt
31	2017-03-30	22:27:00	F9 FT B1021.2	KSC LC-39A	SES-10	5300	GTO	SES	Success	Success (drone ship)
32	2017-01-05	11:15:00	F9 FT B1032.1	KSC LC-39A	NROL-76	5300	LEO	NRO	Success	Success (ground pad)
33	2017-05-15	23:21:00	F9 FT B1034	KSC LC-39A	Inmarsat-5 F4	6070	GTO	Inmarsat	Success	No attempt

We used the query above to see only 5 of all site 'KSC'

# Launch Site Names Begin with KSC

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

```
1 %sql SELECT SUM(PAYLOAD_MASS__KG_) as Payload_NASA FROM SPACEXTBL WHERE customer LIKE '%NASA%(CRS)%';
2
```

```
* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8lcg.databases.appdomain.c
Done.
```

payload_nasa
--------------

48213
-------

Total Payload carried by boosters for NASA (CRS) is 48213.

# Total Payload Mass

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

```
1 %sql SELECT AVG(PAYLOAD_MASS__KG_) as AVG_MASS_F9_V1_1 FROM SPACEXTBL WHERE BOOSTER_VERSION LIKE '%F9%v1.1%';
* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:303
Done.
```

avg_mass_f9_v1_1
------------------

2534
------

Calculated the average payload mass carried by booster F9 v1.1 as 2534

# Average Payload Mass by F9 v1.1

List the date where the succesful landing outcome in drone ship was acheived.

Hint: Use min function

```
1 %sql SELECT MIN(DATE) as First_Succesful_landing FROM SPACEXTBL WHERE LANDING__OUTCOME LIKE '%Success (drone ship)%';
* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30367/BLUD
Done.
```

first_succesful_landing
-------------------------

2016-05-27
------------

We observed that the date of the first **successful landing outcome** in the drone ship was **27<sup>th</sup> May 2016**

# First Successful Ground Landing Date

*List the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000*

```
1 %sql SELECT booster_version as Successful_booster_version, PAYLOAD_MASS__KG_ FROM SPACEXTBL
2 WHERE PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_ < 6000 and LANDING__OUTCOME LIKE '%Success (ground pad)%';
```

```
* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8l1cg.databases.appdomain.cloud:30367/BLUDB
Done.
```

successful_booster_version	payload_mass__kg_
F9 FT B1032.1	5300
F9 B4 B1040.1	4990
F9 B4 B1043.1	5000

Using WHERE clause to filter successfully landed booster on ship pad and applied AND condition

# Successful Drone Ship Landing with Payload between 4000 and 6000

# Total Number of Successful and Failure Mission Outcomes

*List the total number of successful and failure mission outcomes*

```
1 %sql SELECT mission_outcome,COUNT(mission_outcome) as count FROM SPACEXTBL GROUP BY mission_outcome;  
* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8l1cg.databases.appdomain.c  
Done.
```

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



List the names of the booster\_versions which have carried the maximum payload mass. Use a subquery

```
1 %sql SELECT DISTINCT(BOOSTER_VERSION) from SPACEXTBL
2 WHERE PAYLOAD_MASS_KG_ = (select MAX(PAYLOAD_MASS_KG_) from SPACEXTBL);|
3
4
```

```
* ibm_db_sa://f9t48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90108kqb1od81cg.databases.appdomain.cloud:30367/BLUD8
Done.
```

booster\_version

F9 B5 B1048.4

F9 B5 B1048.5

F9 B5 B1049.4

F9 B5 B1049.5

F9 B5 B1049.7

F9 B5 B1051.3

F9 B5 B1051.4

F9 B5 B1051.6

F9 B5 B1056.4

F9 B5 B1058.3

F9 B5 B1060.2

F9 B5 B1060.3

- Determined the booster that has carried the maximum payload using a subquery in the WHERE clause and the MAX() function

# Boosters Carried Maximum Payload

List the records which will display the month names, succesful landing\_outcomes in ground pad ,booster versions, launch\_site for the months in year 2017

Note: SQLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2017' for year.

```
1 %sql SELECT substr(Date,6,2) as month, booster_version, launch_site FROM SPACEXTBL
2 |WHERE landing__outcome like '%Success (ground pad)';
```

```
* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8l1cg.databases.appdomain.cloud:30367/BLUDB
Done.
```

MONTH	booster_version	launch_site
12	F9 FT B1019	CCAFS LC-40
07	F9 FT B1025.1	CCAFS LC-40
02	F9 FT B1031.1	KSC LC-39A
01	F9 FT B1032.1	KSC LC-39A
03	F9 FT B1035.1	KSC LC-39A
08	F9 B4 B1039.1	KSC LC-39A
07	F9 B4 B1040.1	KSC LC-39A
12	F9 FT B1035.2	CCAFS SLC-40
08	F9 B4 B1043.1	CCAFS SLC-40

Using WHERE clause with LIKE, AND, and BETWEEN conditions to filter for failed landing outcomes in drone ships, their booster versions, and launch site names for the year 2015

## 2015 Launch Records

```
1 %sql SELECT DISTINCT(LANDING__OUTCOME),COUNT(*) as count FROM SPACEXTBL
2 WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing__Outcome ORDER BY count DESC ;

* ibm_db_sa://fbt48346:***@815fa4db-dc03-4c70-869a-a9cc13f33084.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30367/BLUDB
Done.
```

landing__outcome	COUNT
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Success (ground pad)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	1
Precluded (drone ship)	1

We selected Landing outcomes and the COUNT of landing outcomes from the data and used the WHERE clause to filter for landing outcomes BETWEEN 2010-06-04 to 2010-03-20.

We applied the GROUP BY clause to group the landing outcomes and the ORDER BY clause to order the grouped landing outcome in descending order.

## Rank Landing Outcomes Between 2010 and 2017

# Launch Sites with Folium

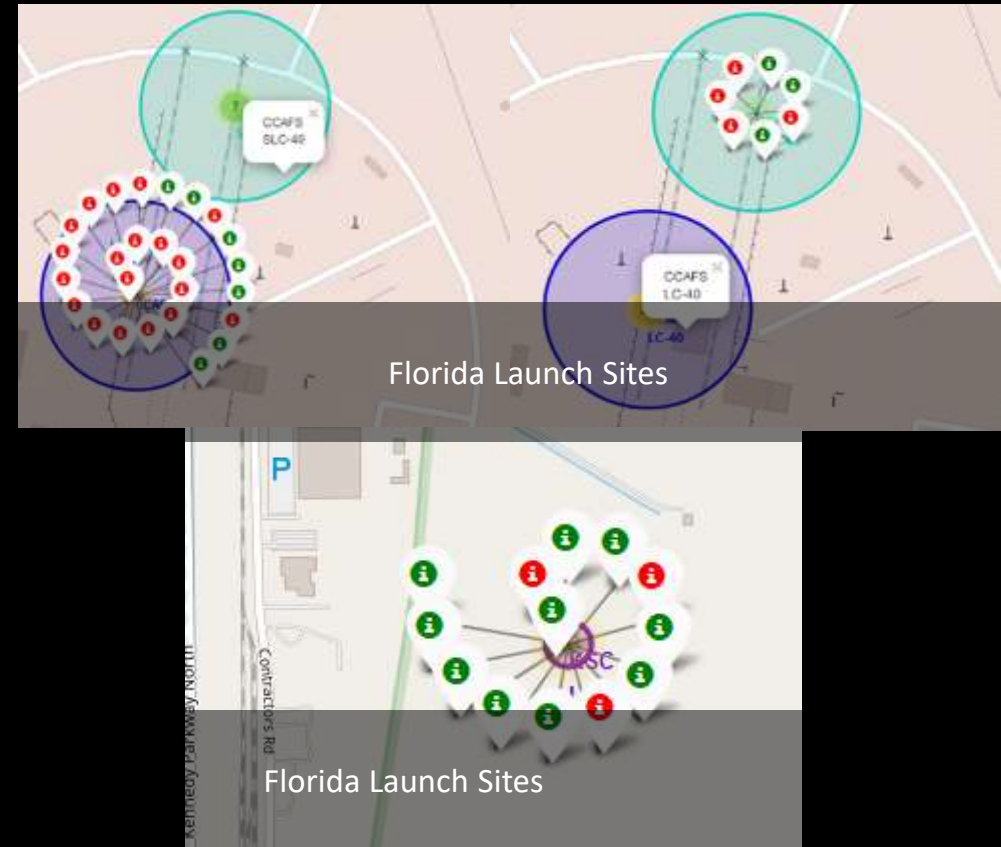
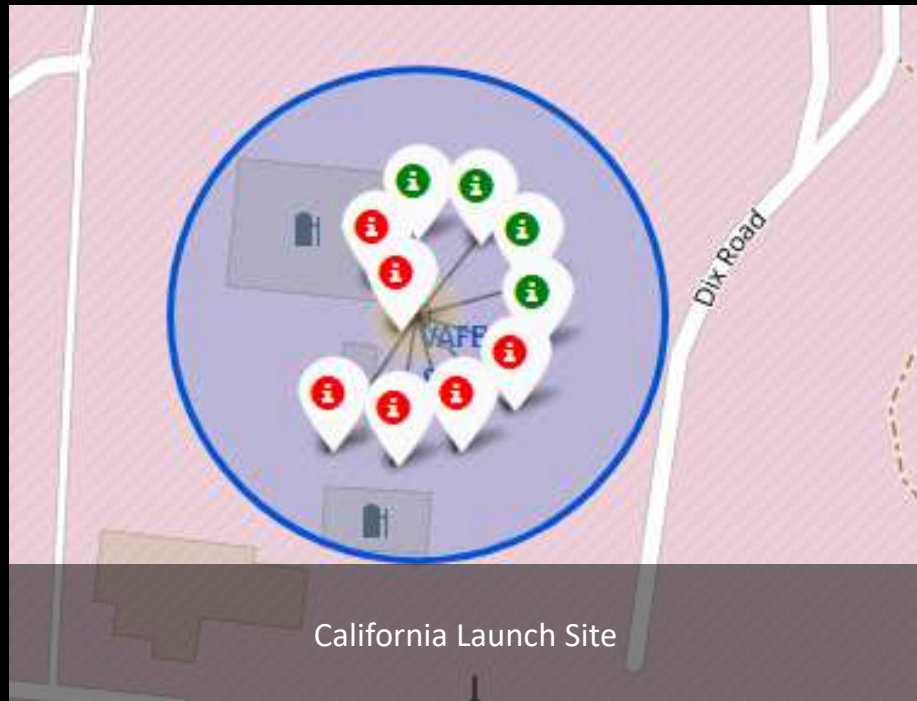


All Launch sites in  
the global map

As we can see Space X  
launch sites are in USA,  
Florida and California



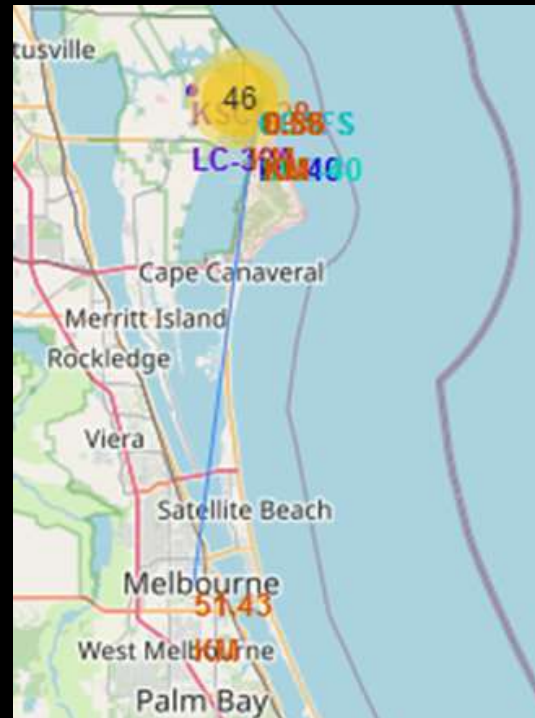
# Markers showing launch sites with color labels





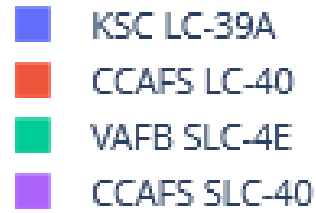
# Launch Site Distance to Landmarks

- Are launch sites near railways? 1.28km
- Are launch sites near highways? 30km
- Are launch sites near the coastline? 0.86km
- Do launch sites keep a certain distance away from cities? 51 km



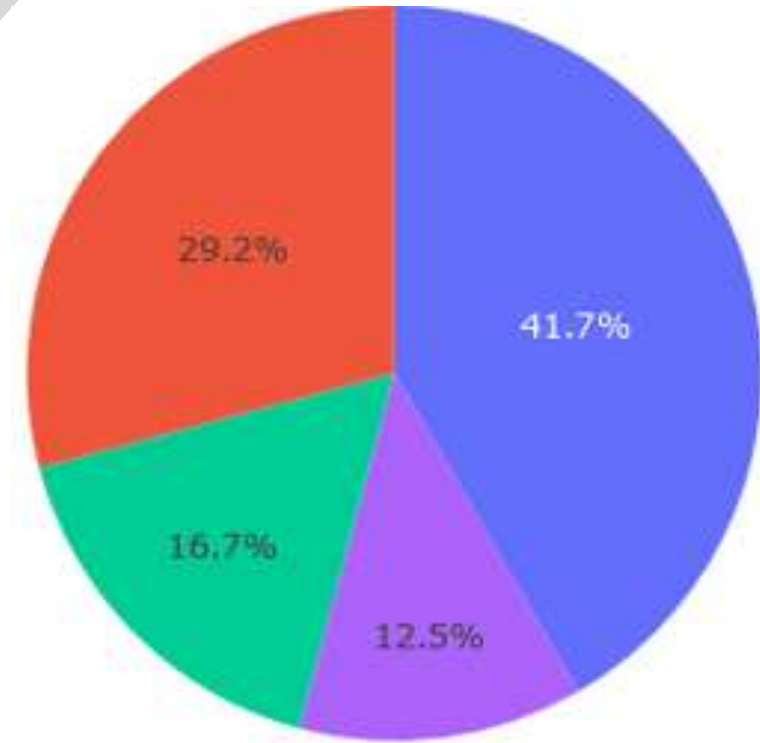


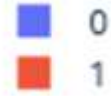
# Build a Dashboard with Plotly Dash



KSC LC-39A had the most successful of all the sites

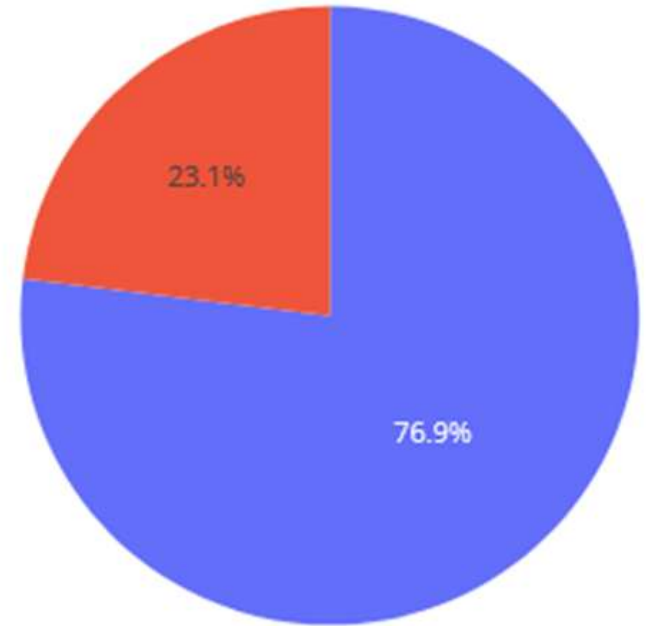
# Total Success Launches By all sites





KSC LC-39A was the most successful of all the sites it achieved a 76.9% success rate while getting a 23.1% failure rate

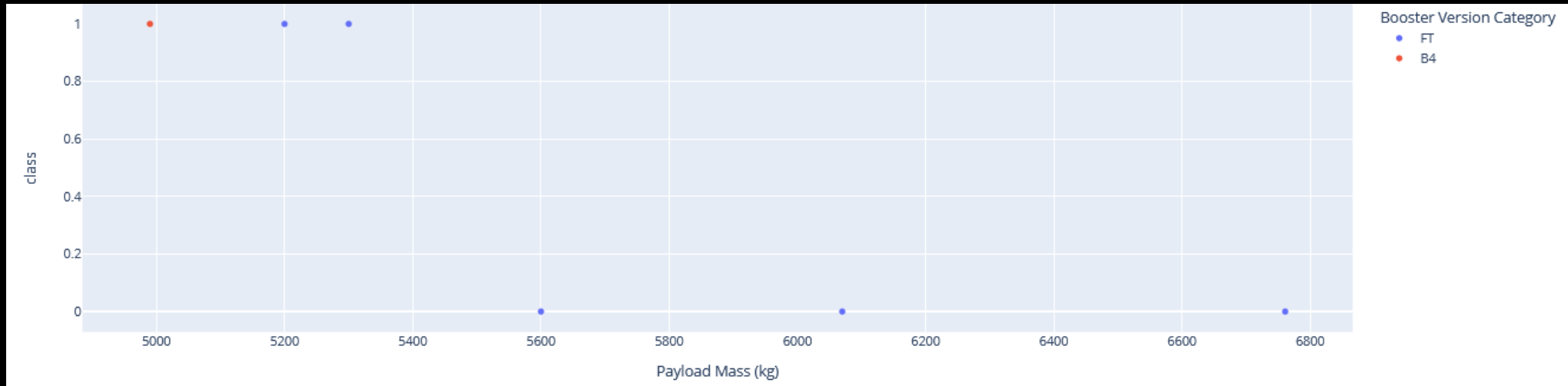
Launch Site with the highest launch success ratio



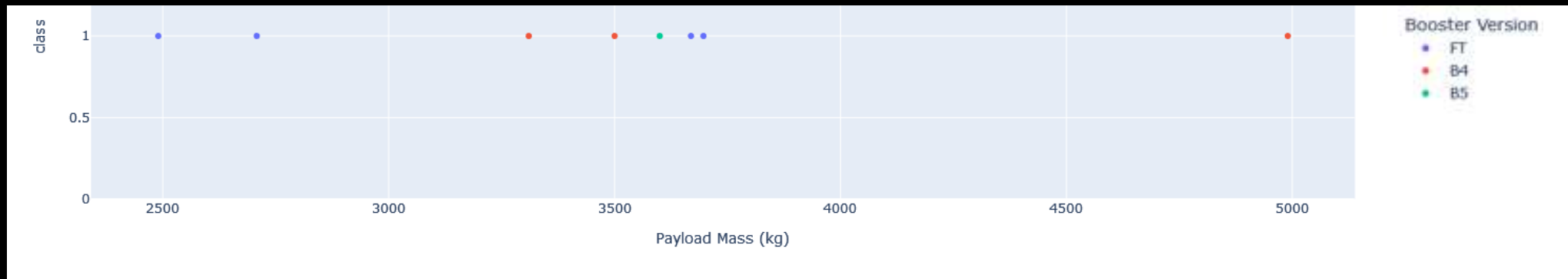
# Scatter plot

## Payload vs. Launch Outcome for all sites

Heavy Weighted Payload 4000 -10000 Kg



Low Weighted Payload 0-4000 Kg





# Predictive Analysis Classification

	Method	Accuracy	Best score
0	Logistic regression	0.833333	0.848429
1	Support vector machine	0.833333	0.848214
2	Decision Tree classifier	0.944444	0.891071
3	K nearest neighbors	0.833333	0.848214

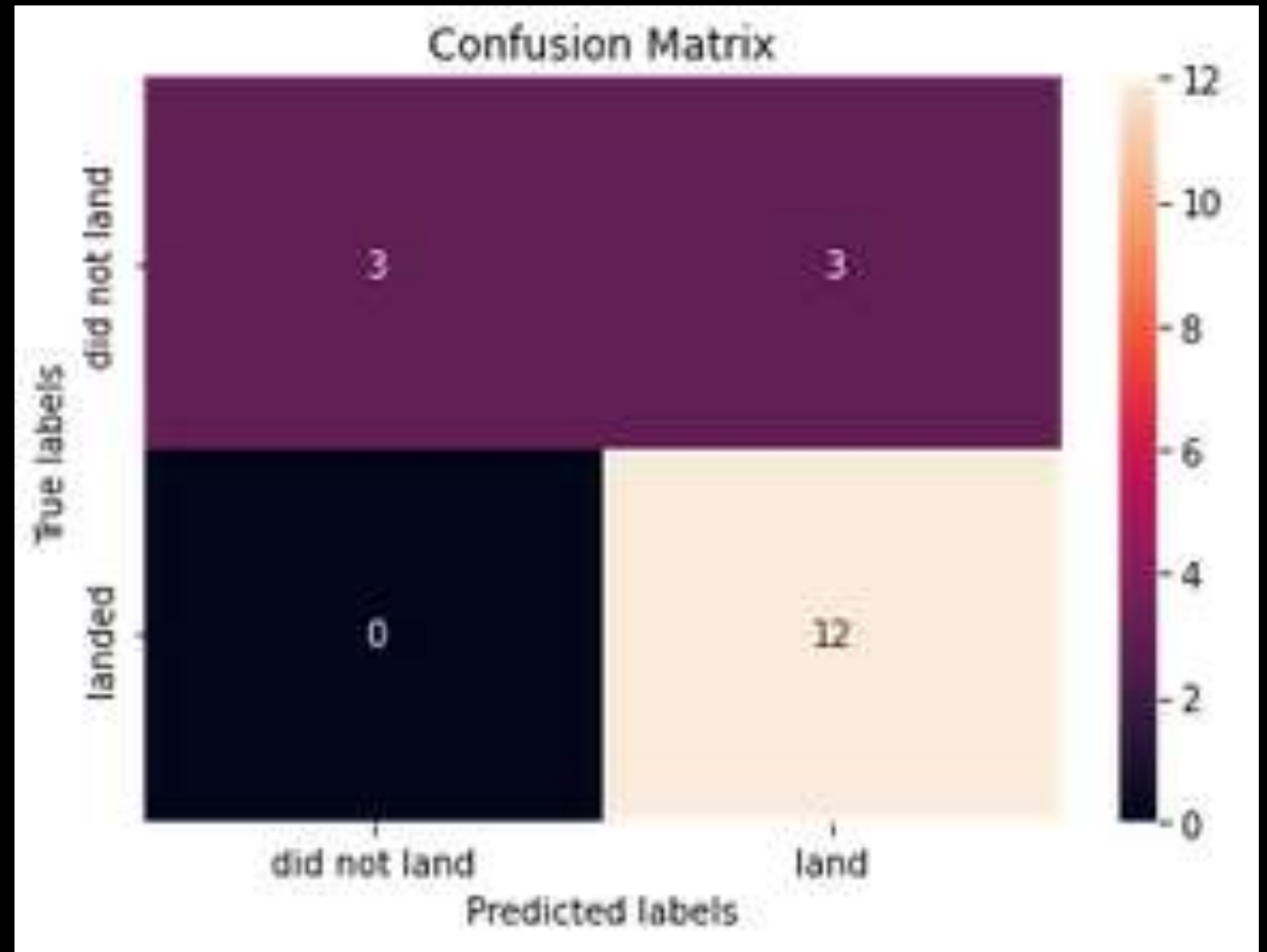
# Classification Accuracy

The decision tree classifier is the model with the highest classification accuracy

# Confusion Matrix

The confusion matrix shows that the classifier can distinguish between the different classes.

The major problem is the false positives, in this project, the unsuccessful landing is marked as a successful landing by the classifier

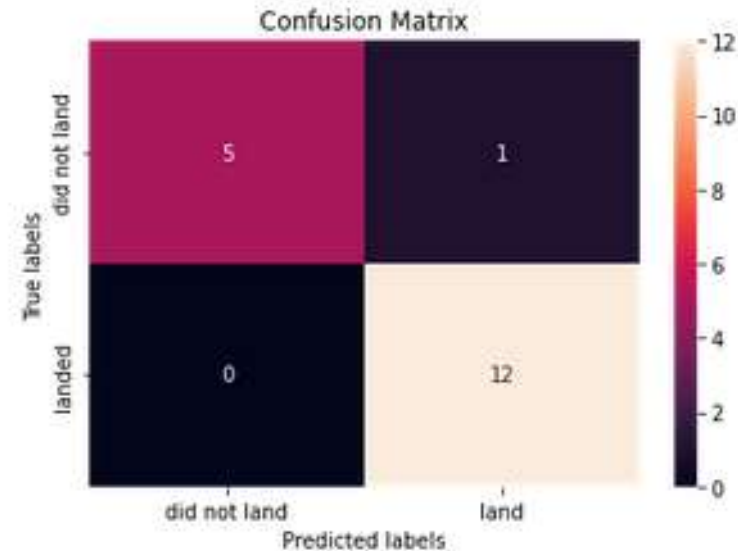


# Prediction tree classifier performs best

- Prediction tree classifier performs best in distinguishing false positives

We can plot the confusion matrix

```
1 yhat = tree_cv.predict(X_test)
2 plot_confusion_matrix(Y_test,yhat)
```





# Conclusions

Seen these data, we can conclude that

- Launch success rate started to increase in 2013 till 2020
- Orbits ES-L1, GEO, HEO, SSO, and VLEO had the most success rate
- KSC LC-39A had the most successful launches of any sites
- The Decision tree classifier is the best machine learning algorithm for this task

Thank you

