

# **DATA SCIENCE**

---

# **CAPSTONE**

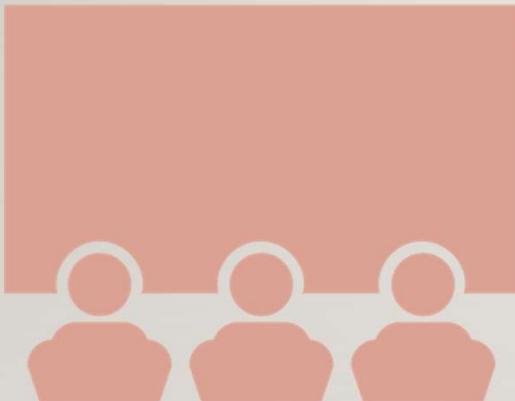
# **PROJECT**

CHEVIBS1020

2021-08-24



# OUTLINE



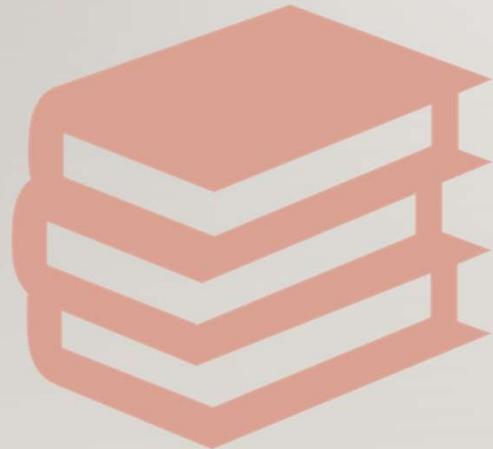
- **Executive Summary**
- **Introduction**
- **Methodology**
- **Results**
- **Conclusion**
- **Appendix**

# EXECUTIVE SUMMARY



- The methodologies involved in this analysis include using python and sql to clean, manipulate, and analyze the data. Finally applying 4 different machine learning models to predict successful SpaceX Falcon9 landings.
- Machine Learning models used are:
  - Logistic Regression
  - Support Vector Machine
  - Tree Classification
  - K Nearest Neighbor
- All 4 machine learning models yielded the same predictions for a successful landing.

# INTRODUCTION



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

# METHODOLOGY

- Data collection methodology:
  - Data were collected from both a SPACE X api and scraping a Wikipedia page using Beautiful Soup
- Perform data wrangling
  - Data Wrangling was performed using python and associated libraries
- Perform exploratory data analysis (EDA) using visualization and SQL
  - SQL magic was used to better analyze the data and visualizations were used to understand the data
- Perform interactive visual analytics using Folium and Plotly Dash
  - Maps were generated to better understand launch locations and surroundings
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models



# METHODOLOGY

# DATA COLLECTION

Data were collected from both a SPACE X API and scraping a Wikipedia page using Beautiful Soup

- API
  - The SpaceX REST API endpoints, or URL, starts with `api.spacexdata.com/v4/`
  - I performed a GET request using the `requests` library to obtain the launch data, which I used to get the data from the API.
  - The end result is viewed by calling the `.json()` method.
  - The response will be in the form of a JSON, specifically a list of JSON objects.
- Web Scraping
  - I used the Python BeautifulSoup package to web scrape some HTML tables that contain valuable Falcon 9 launch records.
  - I parsed the data from those tables and converted them into a Pandas data frame for further visualization and analysis.
  - Next steps will be to transform this raw data into a clean dataset which provides meaningful data on the situation we are trying to address

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
In [12]: # Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Using the dataframe `data` print the first 5 rows

```
In [13]: # Get the head of the dataframe
data.head()
```

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	ships	capsules
+09	False	False	0.0	5e9d0d95eda69955f709d1eb	False			Engine failure at 33 seconds and loss of vehicle		

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

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

```
In [7]: response = requests.get(spacex_url)
```

Check the content of the response

```
In [8]: print(response.content)
```

```
b'[{ "fairings": { "reused": false, "recovery_attempt": false, "recovered": false, "ships": [] }, "links": { "patch": { "small": "https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html", "original": [] }, "reddit": { "campaign": null, "launch": null, "media": null, "recovery": null, "flickr": { "small": [], "original": [] }, "presskit": "https://www.space.com/2196-spacex-inaugural-falcon-1-rocket-lost-launch.html", "wikipedia": "https://en.wikipedia.org/wikifalse", "net": false, "window": 0, "rocket": "5e9d0d95eda69955f709d1eb", "success": false, "details": "Engine failure at 33 seconds and loss of vehicle" }, "auto_update": true, "launch_library_id": null, "failures": [ { "time": 33, "altitude": null, "reason": "0.0002", "date_unix": 1143239400, "date_local": "2006-03-25T10:30:00+12:00", "date_precision": "hour", "upcoming": false, "core": "5e9d0d95eda69955f709d1eb", "details": "Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown at T+7 min 30 s, Failed to reach orbit, Failed to recover first stage" }, { "fairing": { "id": "5eb87cd9ffd86e000604b32a", "large": "https://images2.imgur.com/4f/e3/I0lkuj2e_o.png", "small": "https://images2.imgur.com/e7/1NqsgVYM_o.png" }, "reddit": { "c": "5e9d0d95eda69955f709d1eb", "webcast": "https://www.youtube.com/watch?v=Lk4zQ2wP-Nc", "youtube_id": "Lk4zQ2wP-Nc", "article": "https://www.space.comSat", "static_fire_date_utc": null, "static_fire_date_unix": null, "tbd": false, "net": false, "window": 0, "rocket": "5e9d0d95eda69955f709d1eb", "details": "Residual stage 1 thrust led to collision" }, "auto_update": true, "launch_library_id": null, "failures": [ { "time": 301, "altitude": 289, "reason": "harmonic oscillation", "date_unix": 1174439400, "date_local": "2007-03-21T13:10:00+12:00", "date_precision": "hour", "upcoming": false, "core": "5e9d0d95eda69955f709d1eb", "details": "Failed to reach orbit, Failed to recover first stage" }, { "fairing": { "id": "5eb87cdaffd86e000604b32a", "large": "https://images2.imgur.com/4b/bd/d8UxLh4q_o.png", "small": "https://images2.imgur.com/3d/86/cnu0pan8_o.png" }, "reddit": { "c": "5e9d0d95eda69955f709d1eb", "webcast": "https://www.youtube.com/watch?v=w0w9p3U8860", "youtube_id": "w0w9p3U8860", "article": "https://en.wikipedia.org/wiki/Traillblazer_(satellite)", "static_fire_date_utc": null, "static_fire_date_unix": null, "tbd": false, "net": false, "window": 0, "rocket": "5e9d0d95eda69955f709d1eb", "details": "Failed to reach orbit, Failed to recover first stage" }, { "fairing": { "id": "5eb87cd9ffd86e000604b32a", "large": "https://images2.imgur.com/4f/e3/I0lkuj2e_o.png", "small": "https://images2.imgur.com/e7/1NqsgVYM_o.png" }, "reddit": { "c": "5e9d0d95eda69955f709d1eb", "webcast": "https://www.youtube.com/watch?v=w0w9p3U8860", "youtube_id": "w0w9p3U8860", "article": "https://en.wikipedia.org/wiki/Traillblazer_(satellite)", "static_fire_date_utc": null, "static_fire_date_unix": null, "tbd": false, "net": false, "window": 0, "rocket": "5e9d0d95eda69955f709d1eb", "details": "Failed to reach orbit, Failed to recover first stage" } ] }
```

You should see the response contains massive information about SpaceX launches. Next, let's try to discover some more relevant information for this project.

<https://github.com/chevibs1020/testrepo/blob/master/Complete%20the%20Data%20Collection%20API%20Lab.ipynb>

# DATA COLLECTION – WEB SCRAPING

## TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
In [33]: # use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
print(response.status_code)
print(response.headers)
# print(response.content)

200
{'Date': 'Mon, 23 Aug 2021 03:09:46 GMT', 'Server': 'mw2338.codfw.wmnet', 'X-Content-Type-Options': 'nosniff', 'P3p': 'CP="See https://en.wikipedia.org/wiki/Special:CentralAutoLogin/F3P for more info."', 'Content-Language': 'en', 'Var y': 'Accept-Encoding,Cookie,Authorization', 'Expires': 'Mon, 23 Aug 2021 03:09:46 GMT', 'Last-Modified': 'Sat, 21 Aug 2021 12:18:39 GMT', 'Content-Type': 'text/html; charset=UTF-8', 'Content-Encoding': 'gzip', 'Age': '0', 'X-Cache': 'c p1089 miss, cp1085 pass', 'X-Cache-Status': 'pass', 'Server-Timing': 'cache;desc="pass", host;desc="cp1085"', 'Strict -Transport-Security': 'max-age=106384710; includeSubDomains; preload', 'Report-To': '{ "group": "wm_nel", "max_age": 86400, "endpoints": [{"url": "https://intake-logging.wikimedia.org/v1/events?stream=w3c.reportingapi.network_error& schema_uri=/w3c/reportingapi/network_error/1.0.0" }]}', 'NEL': '{ "report_to": "wm_nel", "max_age": 86400, "failure_f raction": 0.05, "success_fraction": 0.0}', 'Permissions-Policy': 'interest-cohort=()', 'Set-Cookie': 'WMF-Last-Access =23-Aug-2021;Path=/;HttpOnly;secure;Expires=Fri, 24 Sep 2021 00:00:00 GMT, WMF-Last-Access-Global=23-Aug-2021;Path=/; Domain=.wikipedia.org;HttpOnly;secure;Expires=Fri, 24 Sep 2021 00:00:00 GMT, GeoIP=US:::37.75:-97.82;v4; Path=/; secu re; Domain=.wikipedia.org', 'X-Client-IP': '169.60.36.47', 'Cache-Control': 'private, s-maxage=0, max-age=0, must-rev alidate', 'Accept-Ranges': 'bytes', 'Transfer-Encoding': 'chunked', 'Connection': 'keep-alive'}
```

Create a BeautifulSoup object from the HTML response

```
In [35]: # Use BeautifulSoup() to create a BeautifulSoup object from a response text content
html_file = BeautifulSoup(response.text, 'html.parser')
print(html_file.prettify())

<!DOCTYPE html>
<html class="client-nojs" dir="ltr" lang="en">
<head>
<meta charset="utf-8"/>
<title>
List of Falcon 9 and Falcon Heavy launches - Wikipedia
</title>
<script>
document.documentElement.className="client-js";RLCONF={"wgBreakFrames":!1,"wgSeparatorTransformTable":{"":"",""}, "wgD igitTransformTable":[],"wgDefaultDateFormat":"dmy","wgMonthNames":["","January","February","March","April","Ma y","June","July","August","September","October","November","December"],"wgRequestId":"941c15f0-9d1e-4454-b423-2aaaf3e2
^?5e0" "wgContentLanguage": "en", "wgCanonicalNameSpace": "", "wgCanonicalSpecialPageName": "", "wgNamespaceNumber": 0, "wgPageName": "
```

<https://github.com/chevibs1020/testrepo/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb>

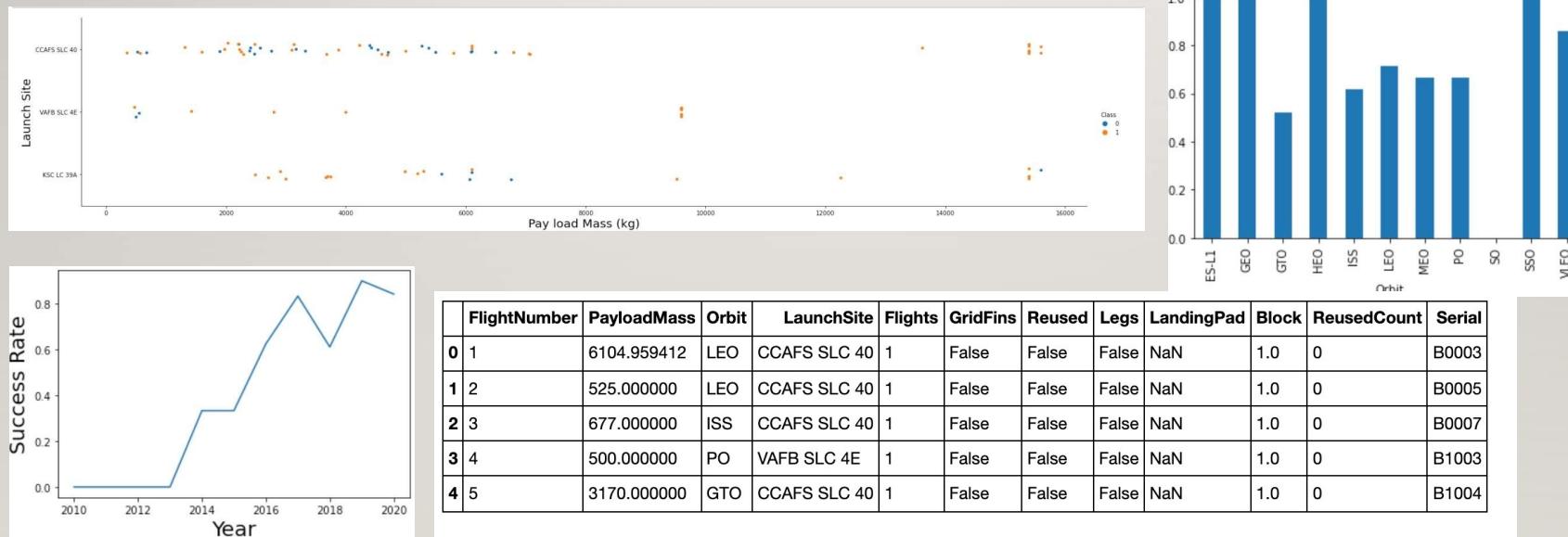
# DATA WRANGLING

- I performed some Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models.
- In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean. True RTLS means the mission outcome was successfully landed to a ground pad False RTLS means the mission outcome was unsuccessfully landed to a ground pad. True ASDS means the mission outcome was successfully landed on a drone ship False ASDS means the mission outcome was unsuccessfully landed on a drone ship.

<https://github.com/chevibs1020/testrepo/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb>

# EDA WITH DATA VISUALIZATION

- In the EDA with Data Visualization step, I created many different scatter plots, bar graphs, line graphs and crosstabs to visualize the data. Below are some examples.



<https://github.com/chevibs1020/testrepo/blob/master/EDA%20with%20Visualization%20lab.ipynb>

# EDA WITH SQL

- Exploratory Data Analysis is the first step of any data science project.
- We can then use these features with machine learning to automatically predict if the first stage can land successfully.
- We see that different launch sites have different success rates. As a result, they can be used to help determine if the first stage will land successfully.
- CCAFS LC-40 has a success rate of 60%, while KSC LC-39A and VAFB SLC 4E have a success rate of around 77%.
- Combining attributes also gives us more information.
- If we overlay the result of the landing outcomes as a color we see that CCAFS LC-40, has a success rate of 60%, but if the mass is above 10,000 kg the success rate is 100%.
- Therefore, we will combine multiple features using SQL
- <https://github.com/chevibs1020/testrepo/blob/master/EDA%20with%20SQL%20lab.ipynb>

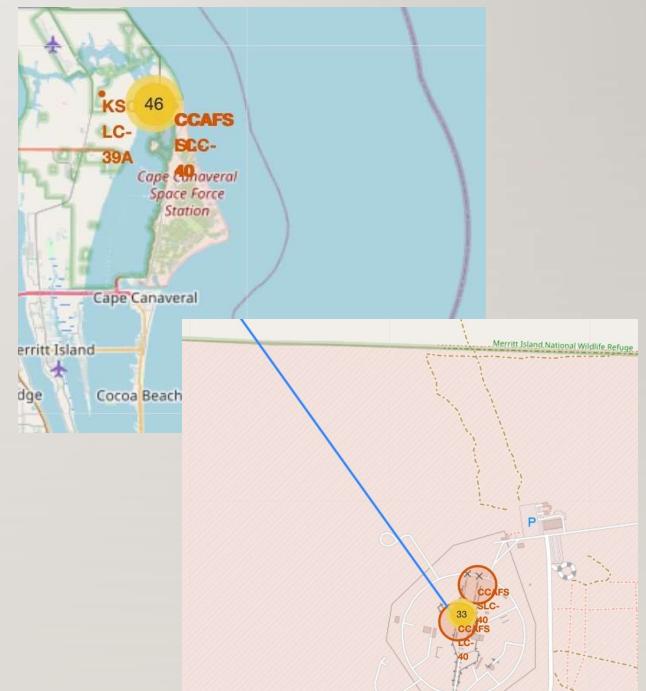
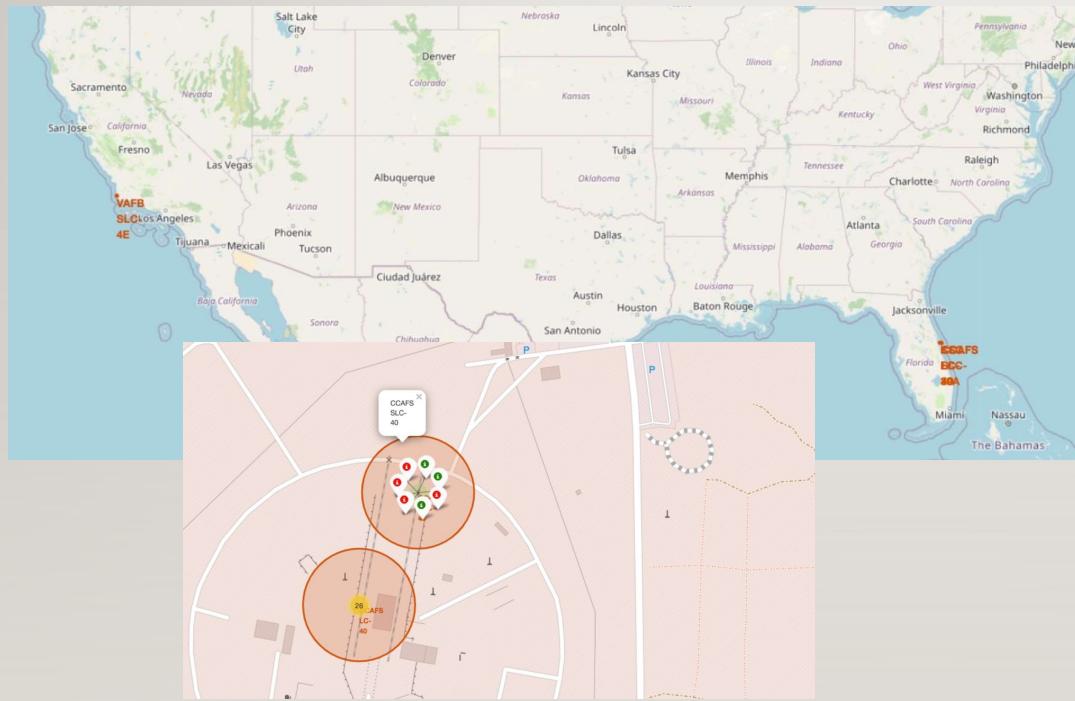
# BUILD AN INTERACTIVE MAP WITH FOLIUM

- The launch success rate may depend on many factors such as payload mass, orbit type, and so on. It may also depend on the location and proximities of a launch site, i.e., the initial position of rocket trajectories. Finding an optimal location for building a launch site certainly involves many factors and hopefully we could discover some of the factors by analyzing the existing launch site locations.
- Using Folium I will mark all Launch Sites on a map, mark the successful/failed launches for each site on the map, and calculate the distances between a launch site to its proximities.

<https://github.com/chevibs1020/testrepo/blob/master/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb>

# BUILD AN INTERACTIVE MAP WITH FOLIUM

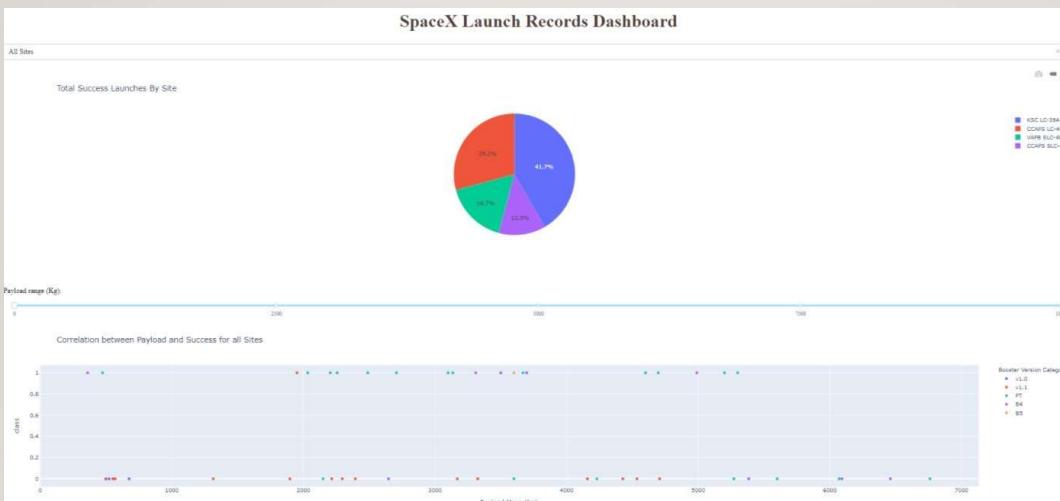
14



<https://github.com/chevibs1020/testrepo/blob/master/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb>

# BUILD A DASHBOARD WITH PLOTLY DASH

- This dashboard application contains input components of the site list and a range slider to interact with a pie chart and a scatter point chart. We can drill into the visuals to better understand successful launches, success rate at each site, and which payload ranges have the highest success rates.



# PREDICTIVE ANALYSIS (CLASSIFICATION)

I BUILT A MACHINE LEARNING PIPELINE TO PREDICT IF THE FIRST STAGE OF THE FALCON 9 LANDS SUCCESSFULLY.

This included Preprocessing, standardizing the data, Train\_test\_split, in order to split data into training and testing data.

I trained the model and perform Grid Search, allowing to find the hyperparameters that allow a given algorithm to perform best.

Using the best hyperparameter values, I determine the model with the best accuracy using the training data.

The following Machine Learning techniques were applied:

- Logistic Regression
- Support Vector machines
- Decision Tree Classifier
- K-nearest neighbors.

<https://github.com/chevibs1020/testrepo/blob/master/Machine%20Learning%20Prediction%20lab.ipynb>

# RESULTS

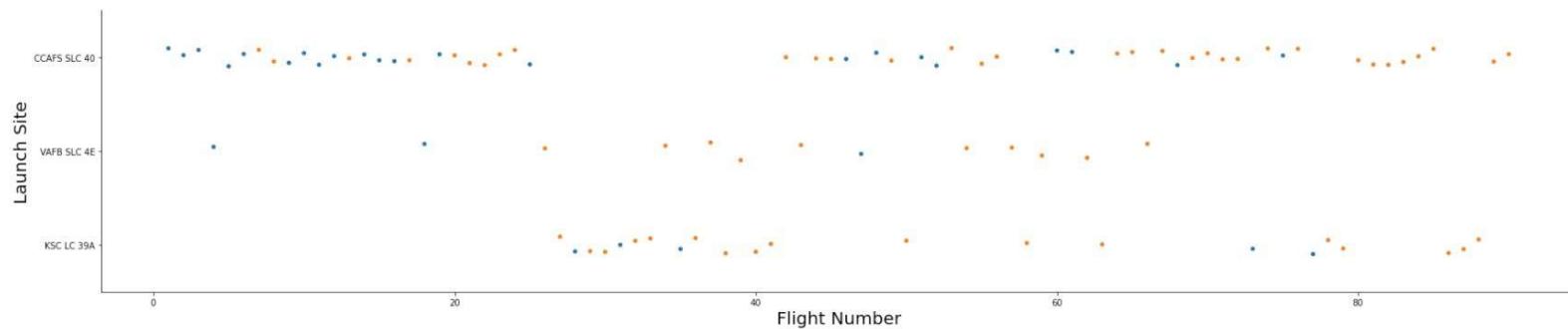


- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

# **EDA WITH VISUALIZATION**

# FLIGHT NUMBER VS. LAUNCH SITE

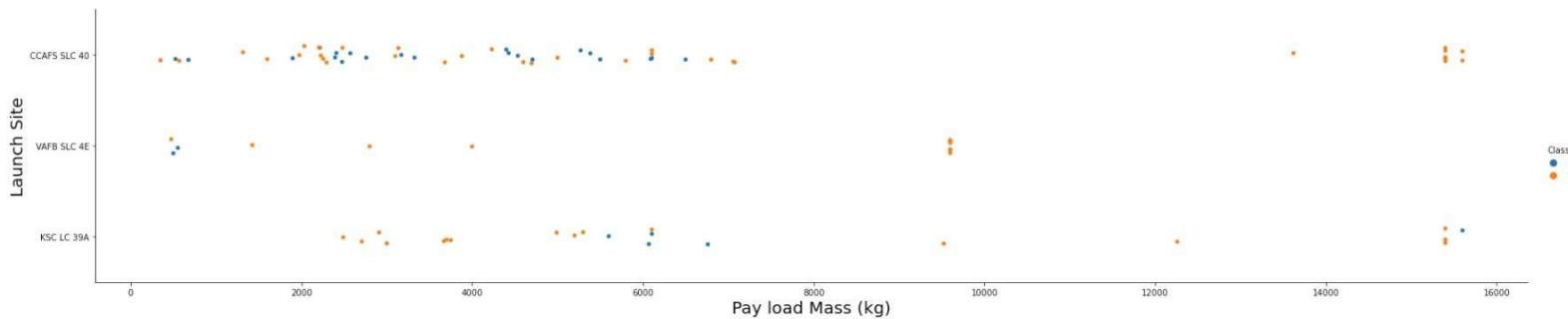
```
In [4]: # Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number", fontsize=20)
plt.ylabel("Launch Site", fontsize=20)
plt.show()
```



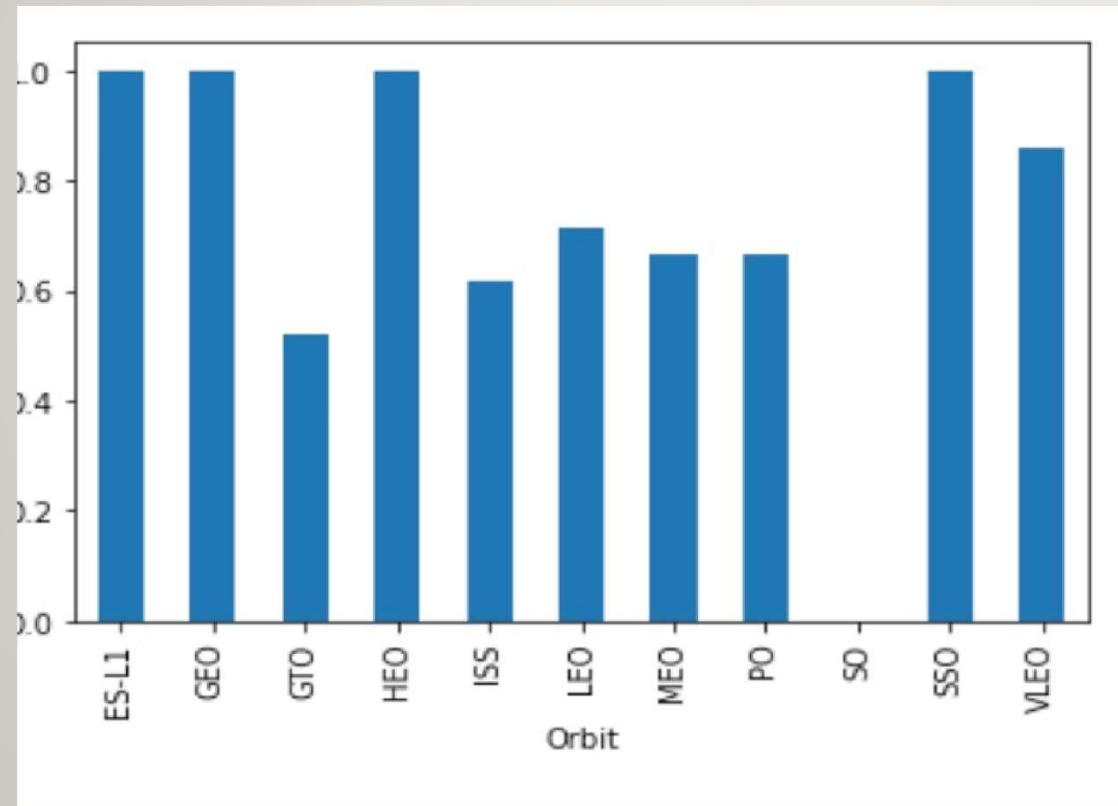
It appears the launch sites all have more successful landings as the flight number increases

## Payload vs. Launch Site

```
In [5]: # Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the  
class value  
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)  
plt.xlabel("Pay load Mass (kg)", fontsize=20)  
plt.ylabel("Launch Site", fontsize=20)  
plt.show()
```



It is very clear that the more heavy the payload, the more successful the landing



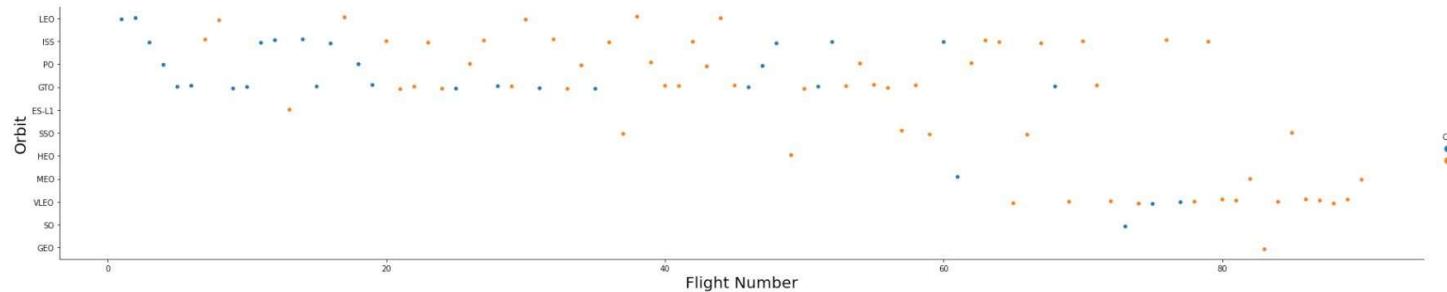
## SUCCESS RATE VS. ORBIT TYPE

GTO orbits have the lowest success rates while 4 of the orbits have a perfect success rate of 100%



# FLIGHT NUMBER VS. ORBIT TYPE

```
In [11]: # 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, aspect = 5)
plt.xlabel("Flight Number", fontsize=20)
plt.ylabel("Orbit", fontsize=20)
plt.show()
```

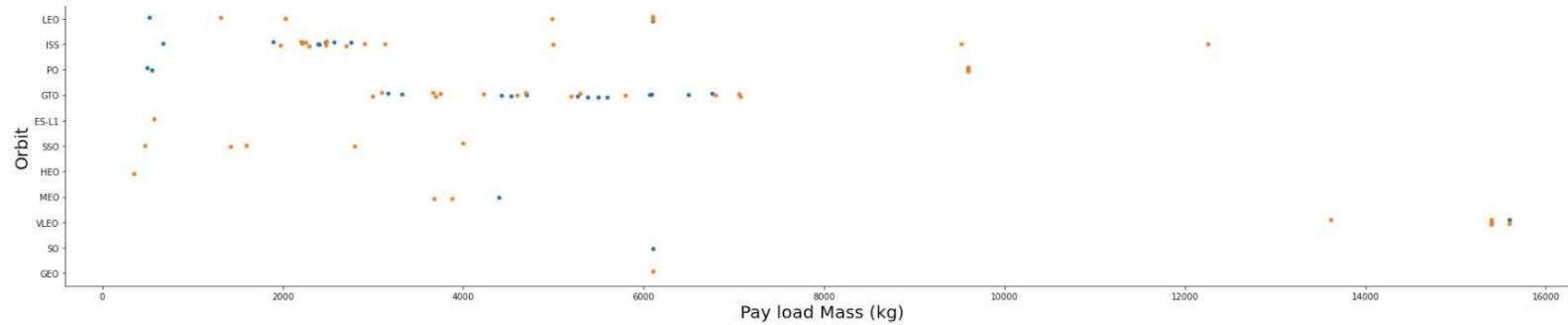


The LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



# PAYLOAD VS. ORBIT TYPE

```
# 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, aspect = 5)  
plt.xlabel("Pay load Mass (kg)", fontsize=20)  
plt.ylabel("Orbit", fontsize=20)  
plt.show()
```



Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.



```

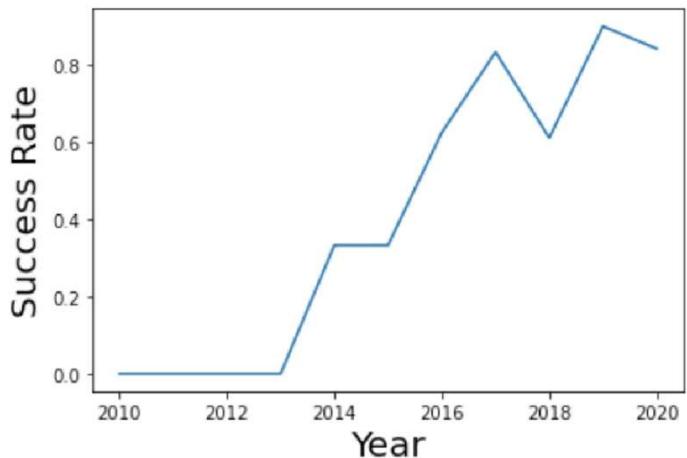
: # A function to Extract years from the date
year=[]
def Extract_year(date):
    for i in df["Date"]:
        year.append(i.split("-")[0])
    return year

: # Plot a line chart with x axis to be the extracted year and y axis to be the success
df['Year'] = pd.DataFrame(Extract_year(df['Date'])).astype('int')

sns.lineplot(x = df['Year'].unique() , y = df.groupby(['Year'])['Class'].mean())

plt.xlabel("Year",fontsize=20)
plt.ylabel("Success Rate",fontsize=20)
plt.show()

```



# LAUNCH SUCCESS YEARLY TREND

The sucess rate since 2013 kept increasing till 2020

# EDA WITH SQL



# ALL LAUNCH SITE NAMES

## Task 1

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

```
In [17]: %sql SELECT launch_site FROM SPACEXTBL GROUP BY launch_site
* ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0t
3/bludb
Done.
```

Out[17]:

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

# LAUNCH SITE NAMES BEGIN WITH 'CCA'

## Task 2

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

```
In [19]: %sql SELECT * FROM SPACEXTBL WHERE launch_site LIKE 'CCA%' LIMIT 5
```

```
* ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:3273
3/bludb
Done.
```

Out[19]:

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

# TOTAL PAYLOAD MASS

## Task 3

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

```
In [25]: %sql SELECT SUM(payload_mass_kg_) FROM SPACEXTBL WHERE customer = 'NASA (CRS)'  
* ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde  
3/bludb  
Done.
```

```
Out[25]:  
1  
45596
```

# AVERAGE PAYLOAD MASS BY F9 V1.1

## Task 4

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

```
In [29]: # %sql SELECT * FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'  
%sql SELECT AVG(payload_mass_kg_) FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'  
  
* ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0tgtu0lqde00.firebaseioapp.com/bludb  
Done.
```

Out[29]:

1
2928

# FIRST SUCCESSFUL GROUND LANDING DATE

## Task 5

*List the date when the first successful landing outcome in ground pad was achieved.*

*Hint: Use min function*

```
In [33]: # %sql SELECT * FROM SPACEXTBL  
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE landing_outcome = 'Success (ground pad)'  
  
* ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0tgtu01qde00.firebaseio.  
3/bludb  
Done.  
  
Out[33]:  
1  
2015-12-22
```

# SUCCESSFUL DRONE SHIP LANDING WITH PAYLOAD BETWEEN 4000 AND 6000

## Task 6

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

```
In [58]: %sql SELECT booster_version FROM SPACEXTBL where landing_outcome = 'Success (drone ship)' AND payload_mass_kg_ BETWEEN 4000 and 6000
```

```
* ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0tgtu0lgde00.databases.appdomain.cloud:32733/bludb
Done.
```

```
Out[58]: booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2
```

# TOTAL NUMBER OF SUCCESSFUL AND FAILURE MISSION OUTCOMES

## Task 7

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

```
In [38]: %sql SELECT mission_outcome, count(*) FROM SPACEXTBL GROUP BY mission_outcome  
* ibm_db_sa://zxr37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0tgtu0lqde00.databases  
Done.
```

```
Out[38]:
```

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

# BOOSTERS CARRIED MAXIMUM PAYLOAD

## Task 8

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

```
In [43]: %sql SELECT booster_version, payload_mass_kg_ FROM SPACEXTBL  
WHERE payload_mass_kg_ = (SELECT max(payload_mass_kg_) FROM SPACEXTBL)  
GROUP BY booster_version, payload_mass_kg_ ORDER BY booster_version ASC  
  
* ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0tgtu0lqde00.databases.appdomai  
Done.
```

```
Out[43]: booster_version  payload_mass_kg_  
F9 B5 B1048.4          15600  
F9 B5 B1048.5          15600  
F9 B5 B1049.4          15600  
F9 B5 B1049.5          15600  
F9 B5 B1049.7          15600  
F9 B5 B1051.3          15600  
F9 B5 B1051.4          15600  
F9 B5 B1051.6          15600  
F9 B5 B1056.4          15600  
F9 B5 B1058.3          15600  
F9 B5 B1060.2          15600  
F9 B5 B1060.3          15600
```

# 2015 LAUNCH RECORDS

## Task 9

*List the failed landing\_outcomes in drone ship, their booster versions, and launch site names for the in year 2015*

```
In [54]: # %sql SELECT DATE FROM SPACEXTBL
%sql SELECT landing_outcome, booster_version, launch_site FROM SPACEXTBL
      WHERE landing_outcome = 'Failure (drone ship)' AND DATE BETWEEN '2015-01-01' AND '2015-12-31'
      * ibm_db_sa://zxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.apoc
Done.
```

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

# RANK SUCCESS COUNT BETWEEN 2010-06-04 AND 2017-03-20

## Task 10

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 [57]: # %sql SELECT * FROM SPACEXTBL
%sql SELECT landing__outcome, count(*) AS counts FROM SPACEXTBL
WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY landing__outcome ORDER BY counts DESC
* ibm_db_sa://zxxk37793:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.clogj3sd0tgtu01qde00.databases.appdomain.cloud:32733/bludb
Done.
```

```
Out[57]:
```

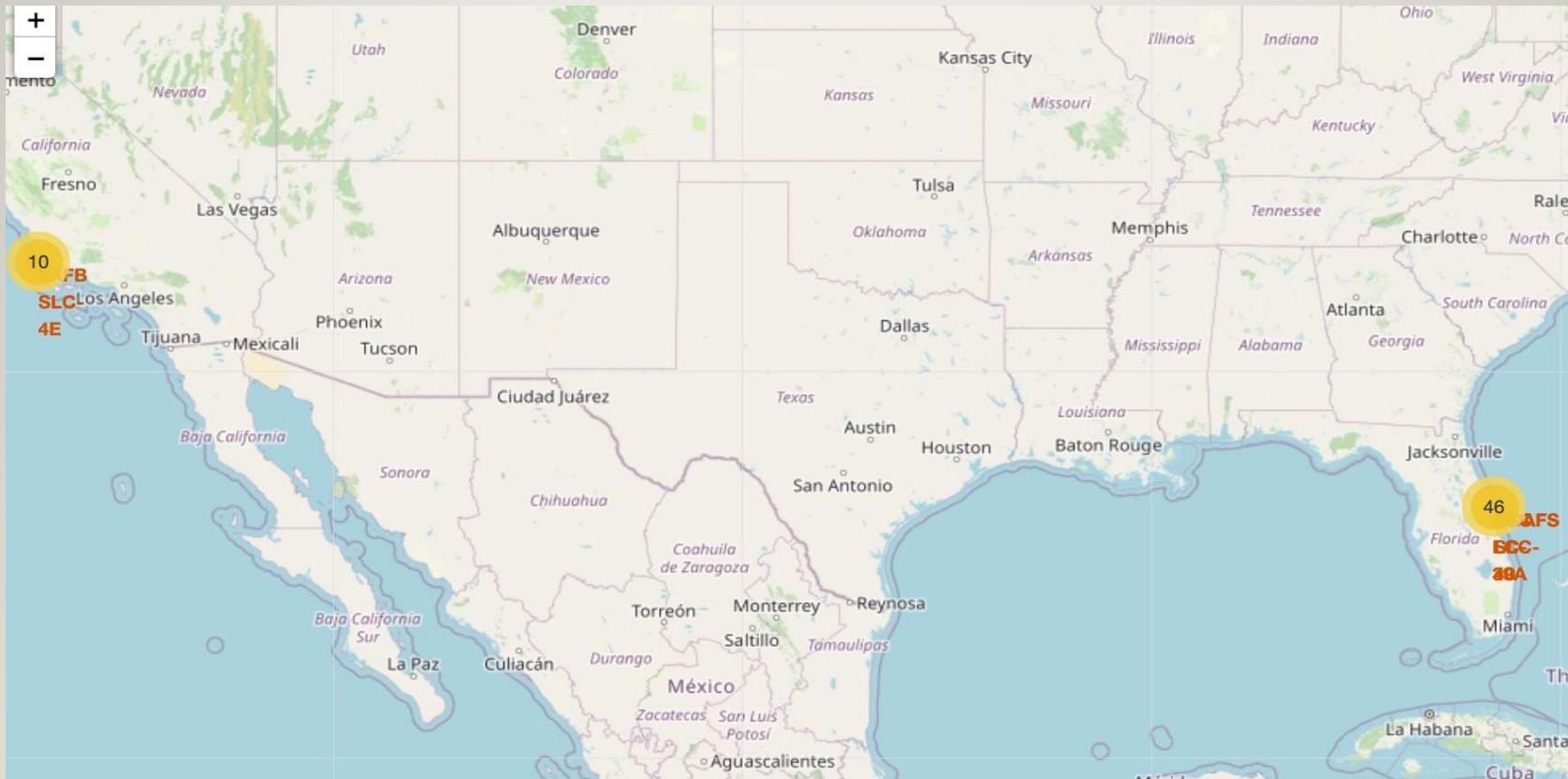
landing__outcome	counts
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

# **INTERACTIVE MAP WITH FOLIUM**

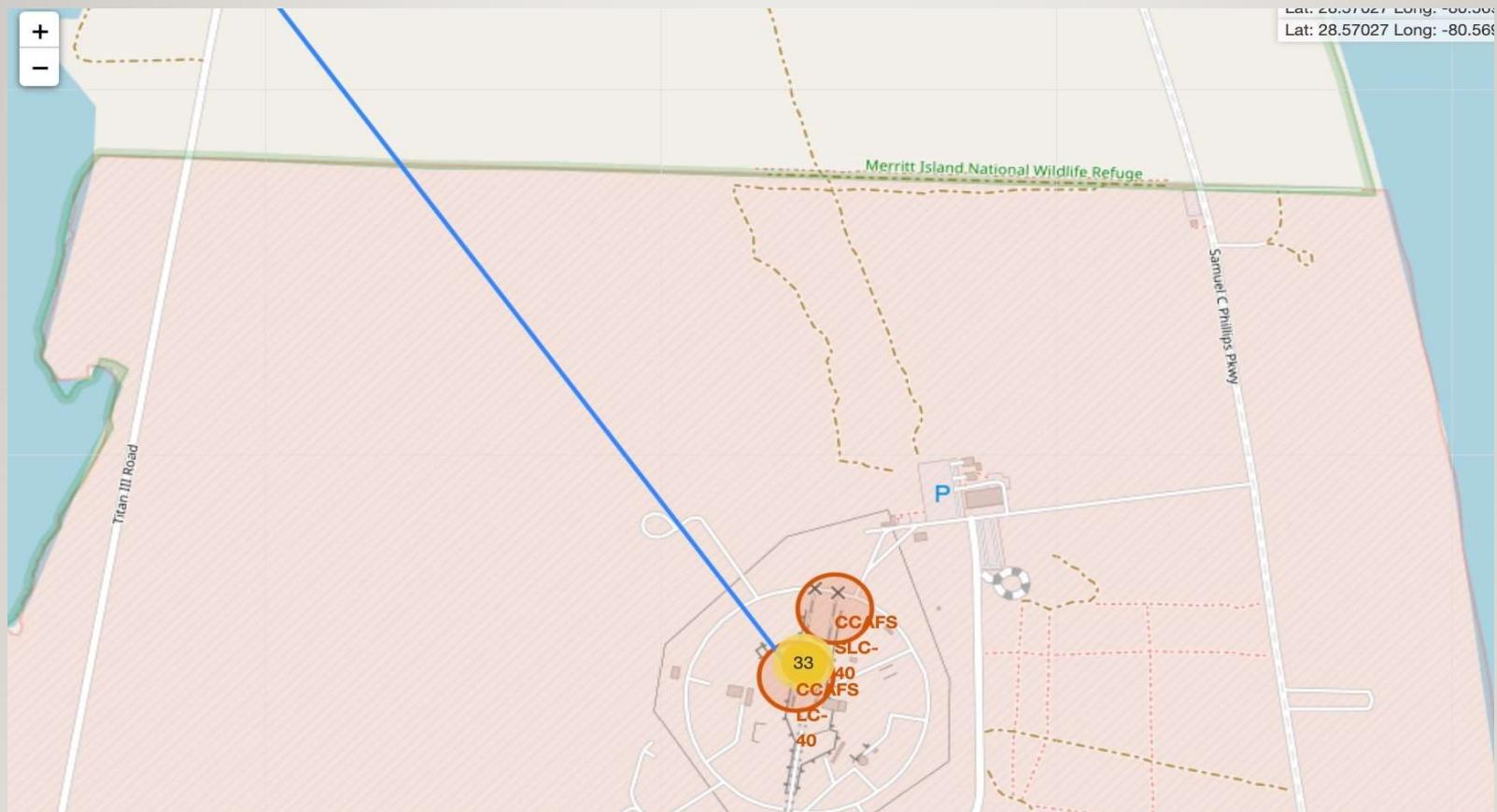
# ALL LAUNCH SITES



# NUMBER OF LAUNCHES AT EACH SITE

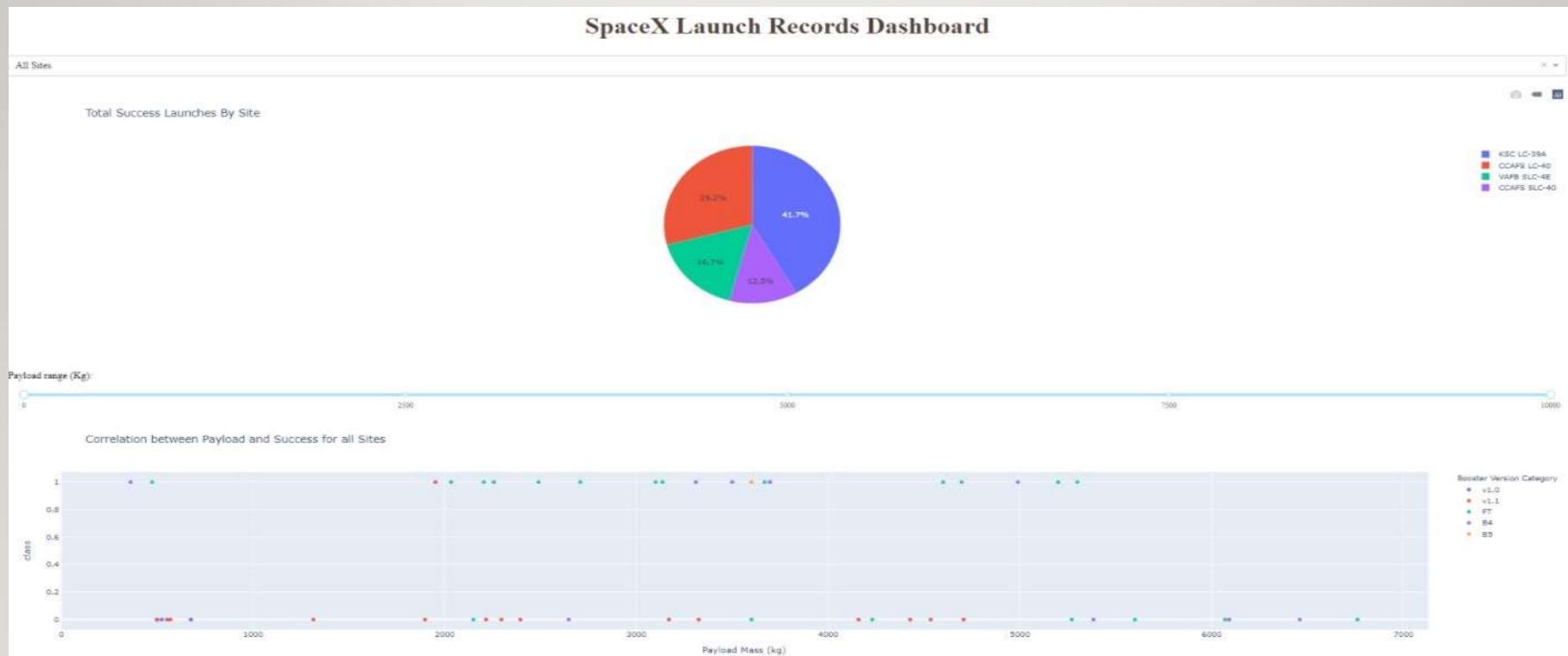


# LAUNCH SITE PROXIMITY TO RAIL LINE

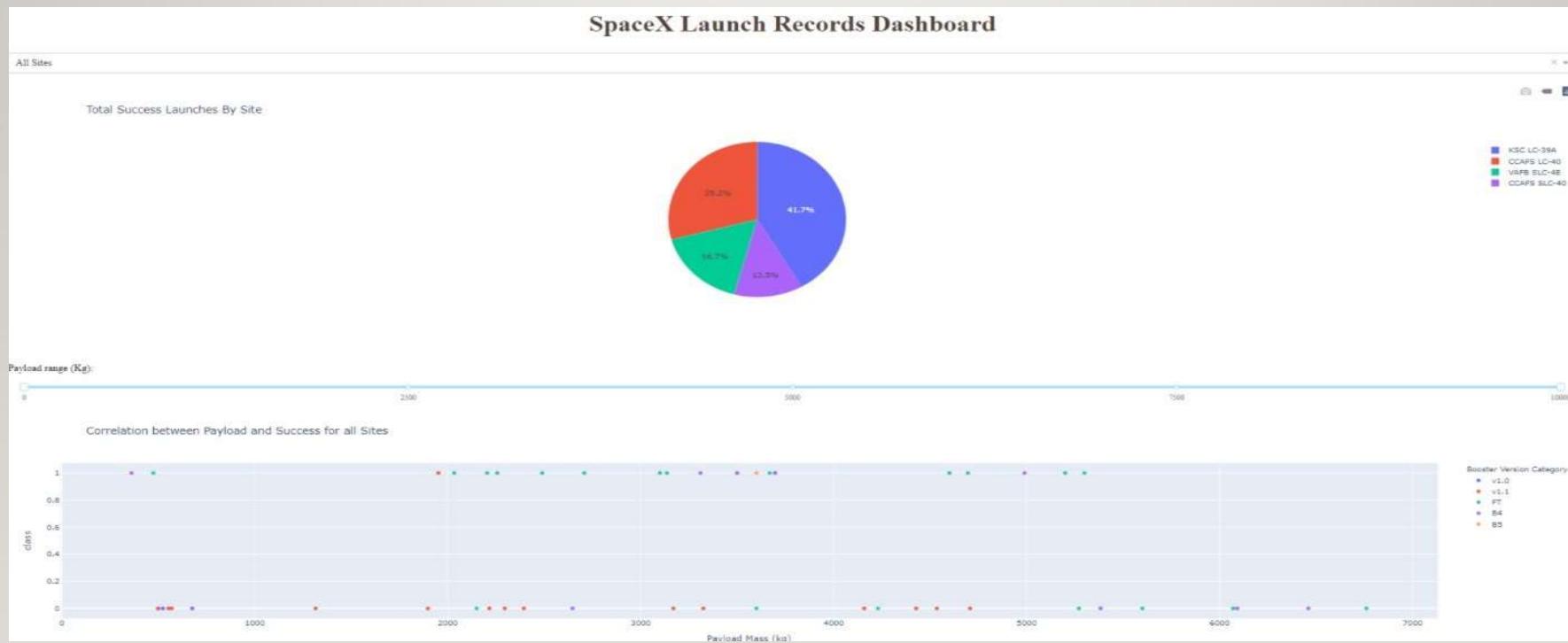


# **BUILD A DASHBOARD WITH PLOTLY DASH**

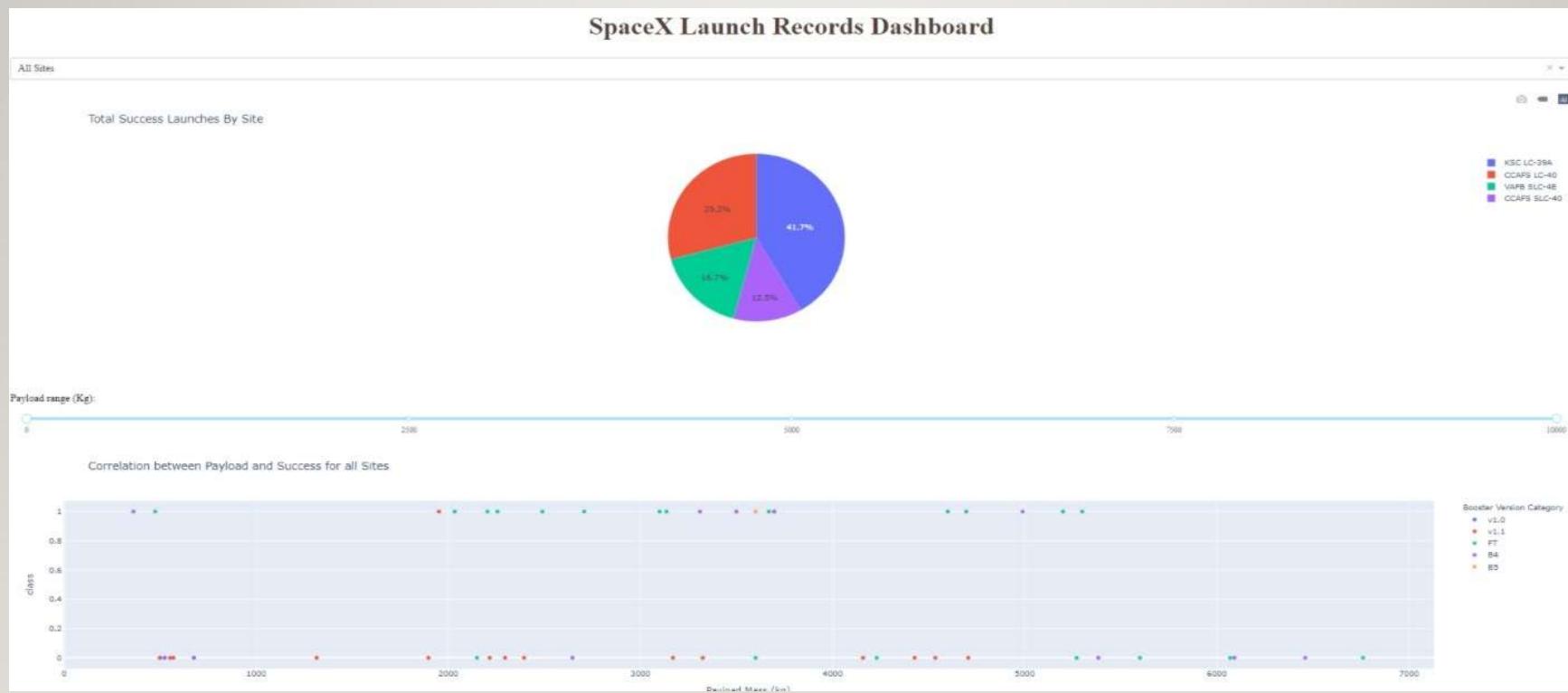
# PLOTLY LAUNCH RECORDS



# HIGHEST LAUNCH SUCCESS RATIO

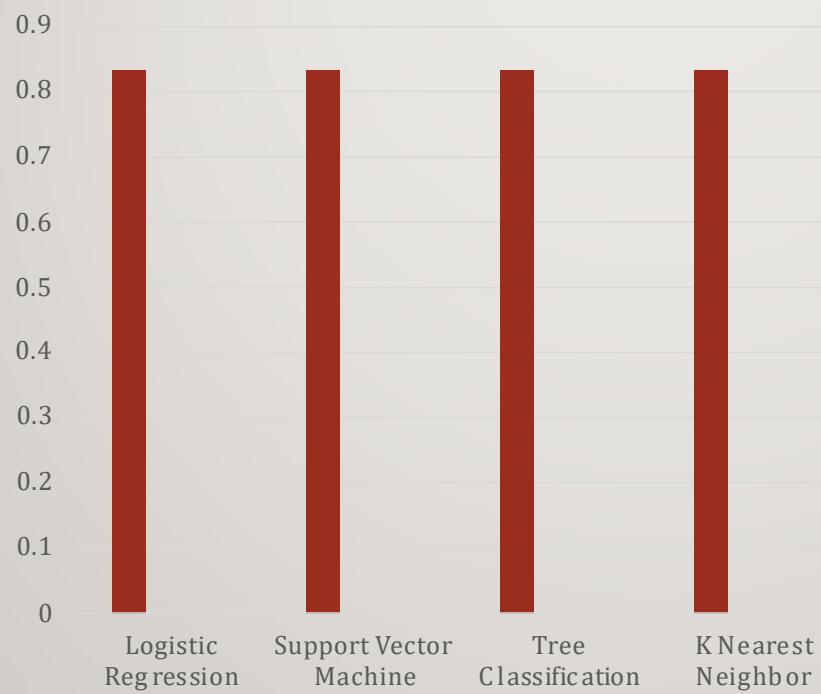


# PAYLOAD VS. LAUNCH OUTCOME SCATTER PLOT



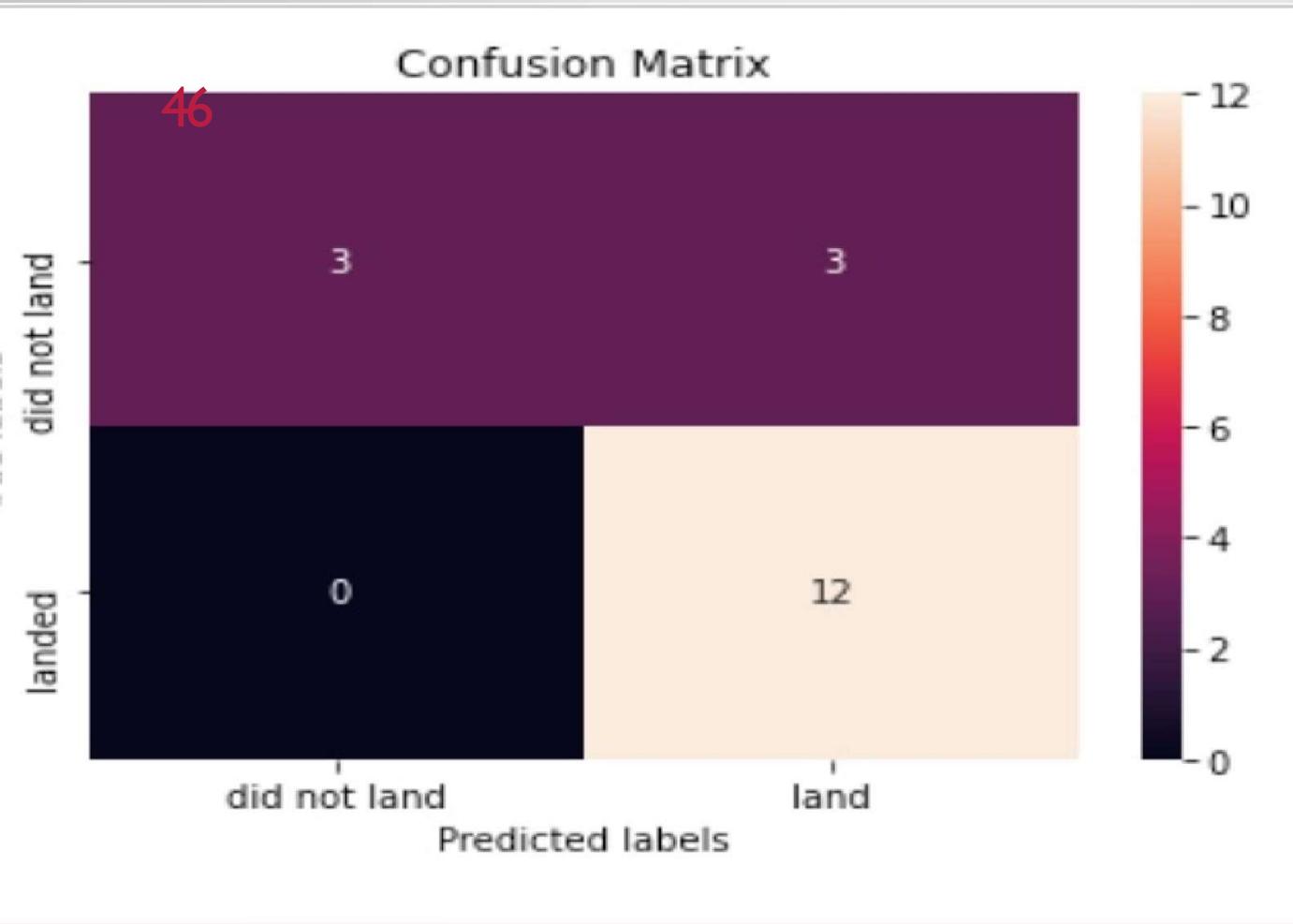
# **PREDICTIVE ANALYSIS (CLASSIFICATION)**

# CLASSIFICATION ON ACCURACY



All of the models have  
the same accuracy  
score of 0.833333333

## **CONFUSION MATRIX**



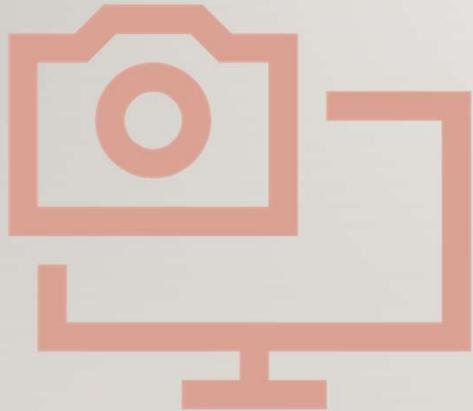
# CONCLUSION



- In conclusion, all of the models found there is a 83.33% probable success rate of the first stage of the rocket landing.
- Using this prediction we can estimate the cost of a launch and have strong leverage against competitors

48

# APPENDIX



- <https://github.com/chevibs1020/testrepo/tree/master>