IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Gabriel Gallardo Giozza
3/19/2023

Outline

Executive Summary

- Summary of methodologies
- Summary of all results

Introduction

- SpaceX became a successful rocket launching leader by re-using their rocket's first stage, being able to offer reduced costs of USD 62 millions per launch, compared to USD 165 millions offered by other providers
- Analysis from SpaceX's rocket first stages data can be used to understand and improve learning curves focused on the design of first stages
- Objectives:
  - ➢What market segments are covered in terms of payload mass and orbits?
  - ➢What are global and segments' successful first stage landing rates?
  - ➢Which parameters play a major role for successful or unsuccessful landings?

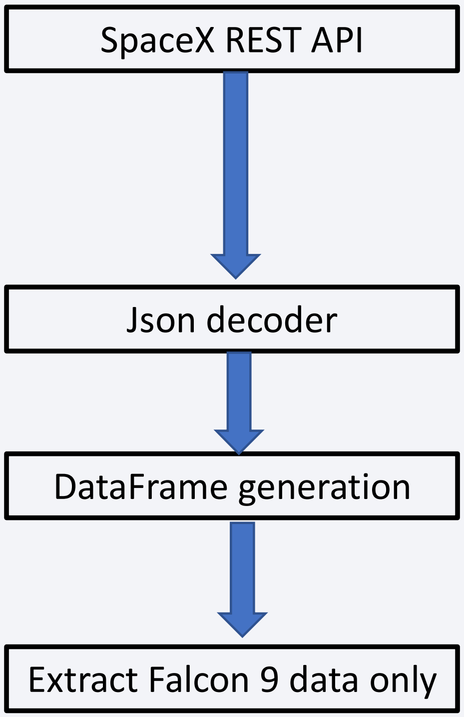
Section 1

Methodology

Methodology

- Data collection methodology:
  - SpaceX Rest API (https://api.spacexdata.com/v4)
  - Web Scrapping (https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches)
- Perform data wrangling
  - Data cleaning: Replacing PayloadMass missing values by mean values.
  - Identify orbits, launching sites
  - Wrangle missions' outcome into a single variable
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Initial dataset split into training and testing datasets
  - 4 Training Label methods used: Logistic Regression, Support Machine Vector, Decision Tree Classifier, K-Neighbors Classifier

Data Collection – SpaceX API



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

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

```
response = requests.get(spacex_url)
```

Check the content of the response

```
print(response.content)
```

b'[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"link

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

```
# Use json_normalize meethod to convert the json result into a dataframe
json_df_raw=response.json()
data=pd.json_normalize(json_df_raw)
```

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head(5)
```

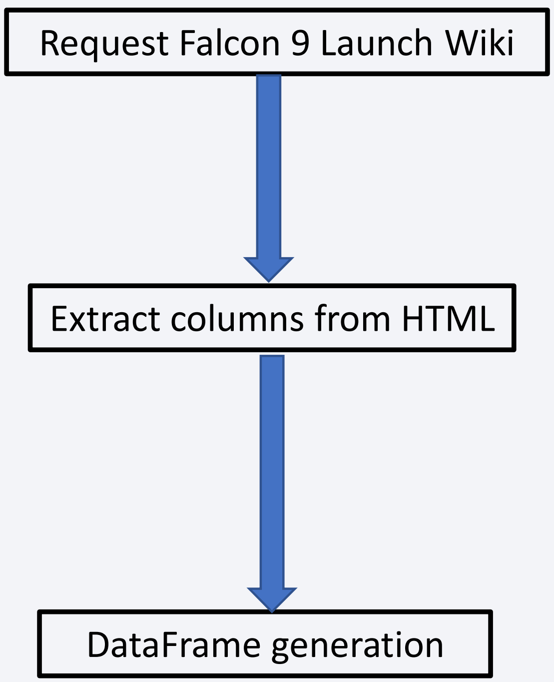
	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = launch_dict_df[launch_dict_df['BoosterVersion'] != 'Falcon 1']
data_falcon9.describe()
```

	FlightNumber	PayloadMass	Flights	Block	ReusedCount	Longitude	Latitude
count	90.000000	85.000000	90.000000	90.000000	90.000000	90.000000	90.000000

- GitHub URL: [IBMWatson/W1. Data Collection.ipynb at main · GGGiozzaG/IBMWatson (github.com)]()

Data Collection – Scraping



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

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

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

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
response_text = response.text
soup = BeautifulSoup(response_text, 'html.parser')
```

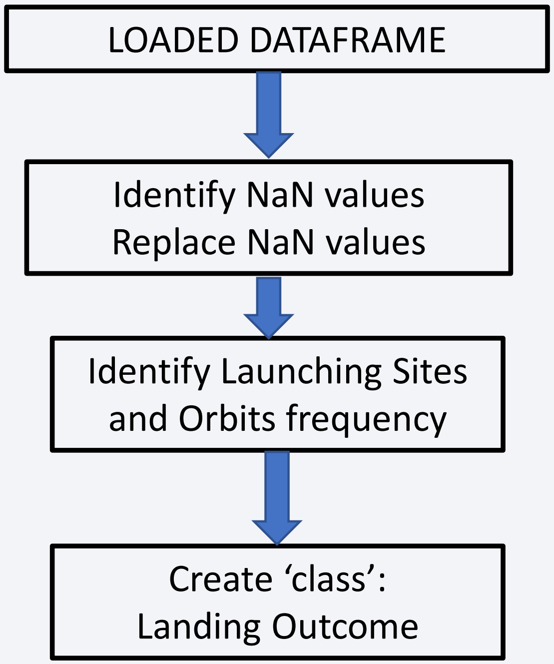
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
th_elements = first_launch_table.find_all('th')
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
column_names = []
for th in th_elements:
    name = extract_column_from_header(th)
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names
    if name is not None and len(name) > 0:
        column_names.append(name)
```

```
            # Customer
            # TODO: Append the customer into launch_dict with key `Customer`
            customer = row[6].a.string
            #print(customer)

            # Launch outcome
            # TODO: Append the launch_outcome into launch_dict with key `Launch outcome`
            launch_outcome = list(row[7].strings)[0]
            #print(launch_outcome)

            # Booster landing
            # TODO: Append the launch_outcome into launch_dict with key `Booster landing`
            booster_landing = landing_status(row[8])
            #print(booster_landing)
```

```
F9 v1.0B0003.1
F9 v1.0B0004.1
F9 v1.0B0005.1
F9 v1.0B0006.1
F9 v1.0B0007.1
```

- GitHub URL: [IBMWatson/W1. Data Collection with Web Scraping.ipynb at main · GGGiozzaG/IBMWatson (github.com)]

Data Wrangling




```python
# Apply value_counts() on column LaunchSite
LaunchSite_counts = df['LaunchSite'].value_counts()
print(LaunchSite_counts)
```

```
CCAFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
Name: LaunchSite, dtype: int64
```

```python
# Apply value_counts on Orbit column
Orbit_counts = df['Orbit'].value_counts()
print(Orbit_counts)
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```

```python
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = df['Outcome'].apply(lambda x: 0 if x in bad_outcomes else 1).tolist()
```

This variable will represent the classification variable that represents the outcome of each launch. If t
first stage landed Successfully

```python
df['Class']=landing_class
df[['Class']].head(8)
```

	Class
0	0
1	0
2	0
3	0
4	0

GitHub URL: [IBMWatson/W1. Data Collection.ipynb at main · GGiozzaG/IBMWatson (github.com)] – NaN Values
[IBMWatson/W1. Data Wrangling.ipynb at main · GGiozzaG/IBMWatson (github.com)] – Launching sites, and ‘class’ value

EDA with Data Visualization

Summary

- Flight Number vs Pay Load Mass → Internal correlations between flight number, payload mass and success
- Flight Number vs Class → Identify success correlations per launch site
- Payload vs Class → Identify success correlations per launch site
- Mean Class Bar Chart per Orbit → Identify orbits with higher success rate
- Flight Number vs Orbit → Identify success rate trends within orbits
- Payload vs Orbit → Identify success rate trends within orbits
- Success Rate vs Year → Identify success rate trend with time

- GitHub URL: [IBMWatson/W2. Exploratory Analysis using PD and PLT.ipynb at main · GGGiozzaG/IBMWatson (github.com)]

EDA with SQL

Summary

- Launch Sites → What are the main launch sites
- Payload mass → What is the total and average payload mass carried away for specific customers or with specific boosters
- Boosters → Identify boosters used for specific missions and their outcomes
- Outcomes → Dates and types of outcomes identification

- GitHub URL: [IBMWatson/W2. Exploratory Analysis using SQL.ipynb at main · GGGiozzaG/IBMWatson (github.com)]

Build an Interactive Map with Folium

Summary

• Location Circles (Launching Sites)	→	Know where launching sites are located
• Marker	→	See references for launching sites
• Success/Fail markers	→	Identify success/fails for launch sites
• Polyline markers	→	Interactive proximity analysis between map features and launch sites

- GitHub URL: IBMWatson/W3. Folium Data Visualization.ipynb at main · GGGiozzaG/IBMWatson (github.com)

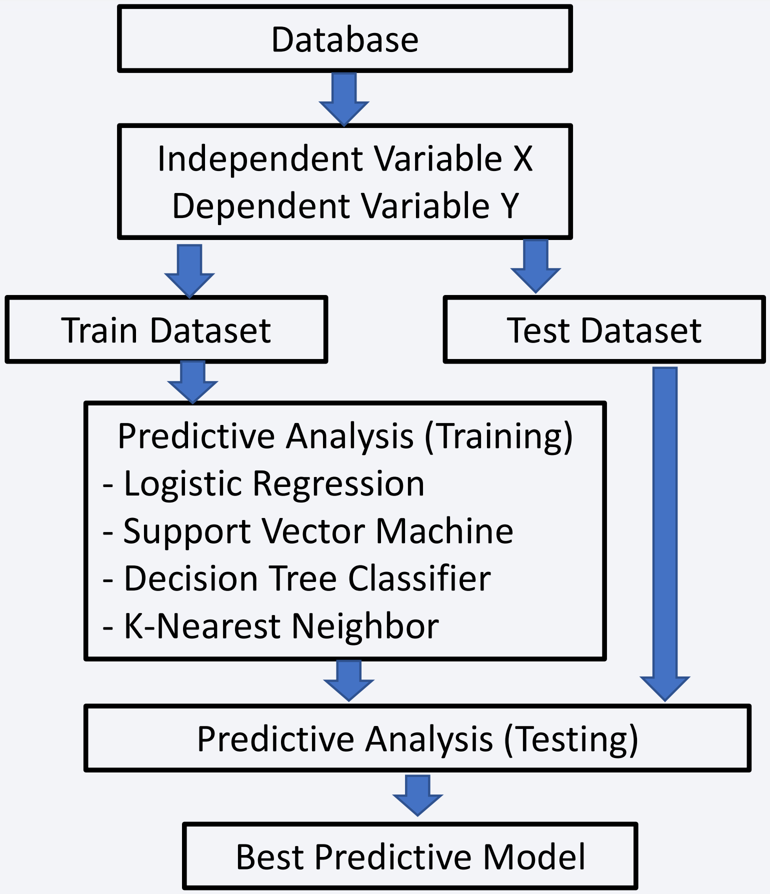
Build a Dashboard with Plotly Dash

Summary

• Total Launches by Site Pieplot	→	Distribution of launches between sites
• Success/Fail Ratio per Site Pieplot	→	Success/Fail Ratio interactive view
• Payload range	→	Define payload frame to analyze
• Success/Fail Ratio per Payload Plot	→	Success/Fail Ratio interactive view

- GitHub URL: [IBMWatson/W3. Dash Plotly Data Visualization.py at main · GGGiozzaG/IBMWatson (github.com)]

Predictive Analysis (Classification)

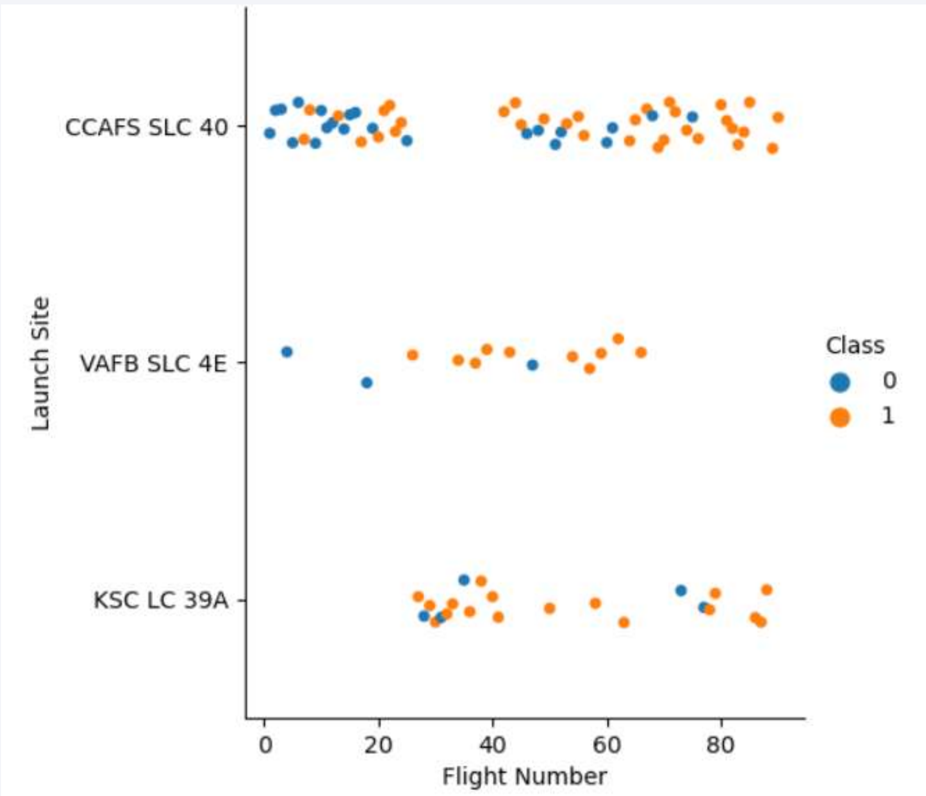


- GitHub URL: IBMWatson/W4. Predictive Analysis.ipynb at main · GGGiozzaG/IBMWatson (github.com)

Section 2

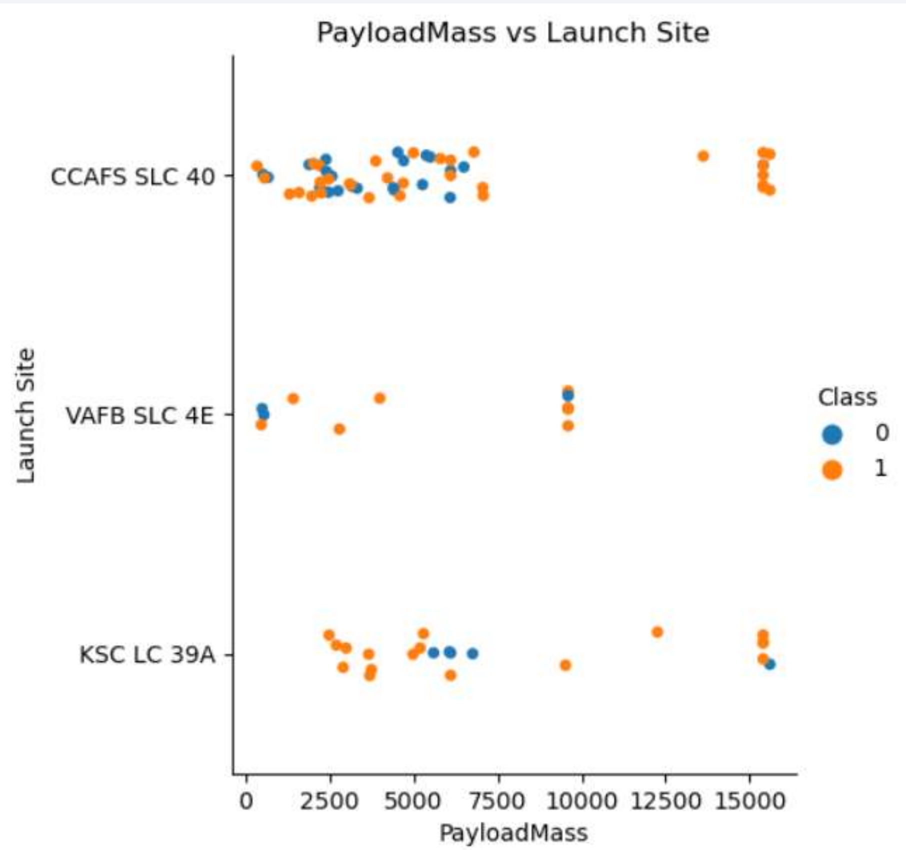
Insights drawn from EDA

Flight Number vs. Launch Site



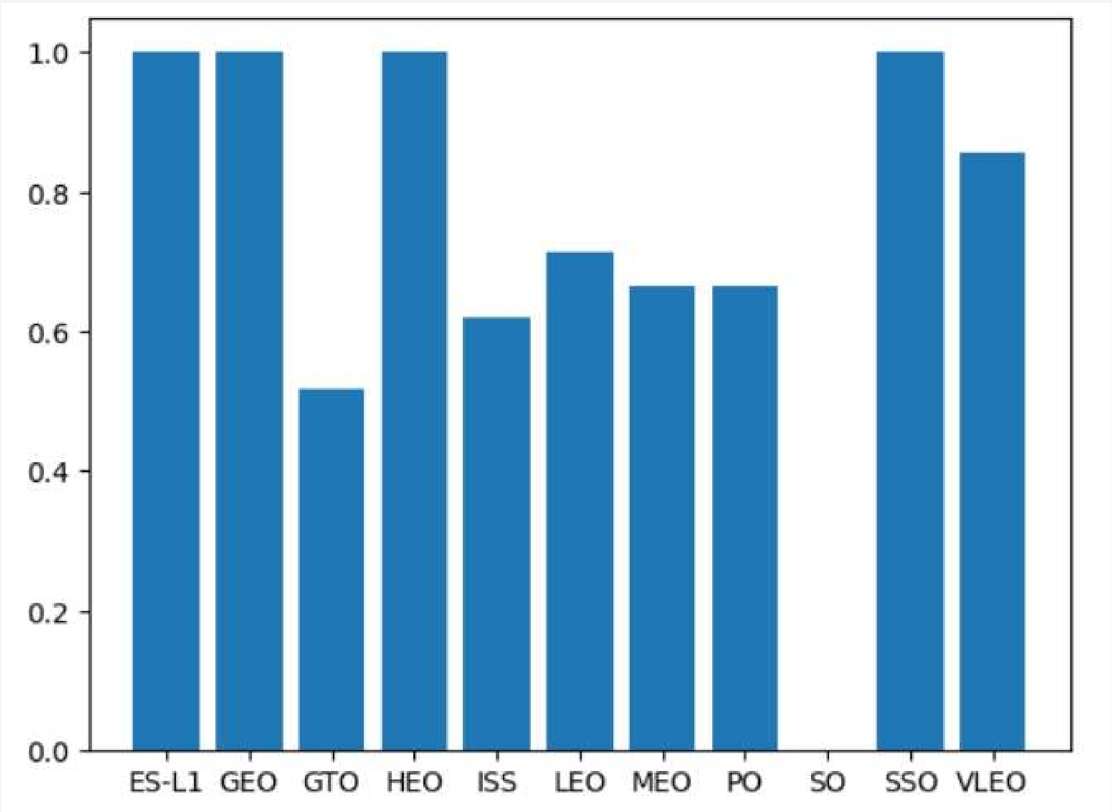
- CCAFS SLC 40 is the earliest launch site used
- CCAFS SLC 40 is the launch site with the highest number of flights
- KSC LC 39A is the latest launch site to start launching missions
- Unsuccessful mission rate (Class 0) decreases for higher flight rate numbers

Payload vs. Launch Site



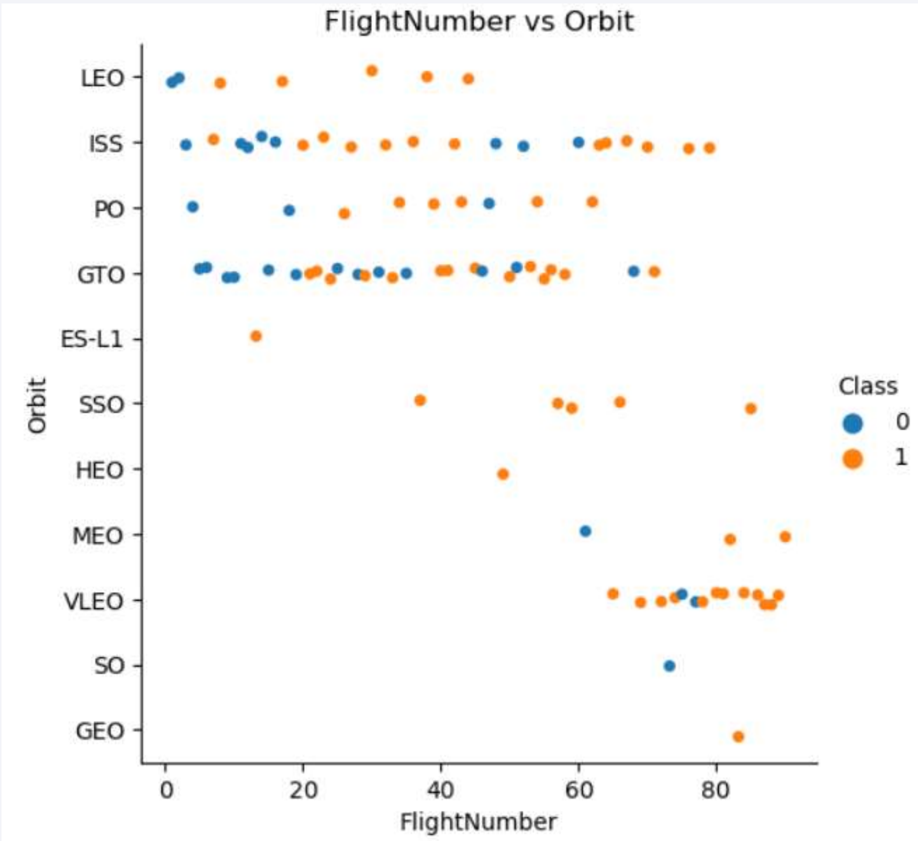
- Maximum payload for CCAFS SLC 40 and KSC LC 39A are16 Tons, and for VAFB SLV 4E is 10 Tons
- Maximum payload are outliers from payloads distributions for all stations
- There is no correlation between payload and landing success

Success Rate vs. Orbit Type



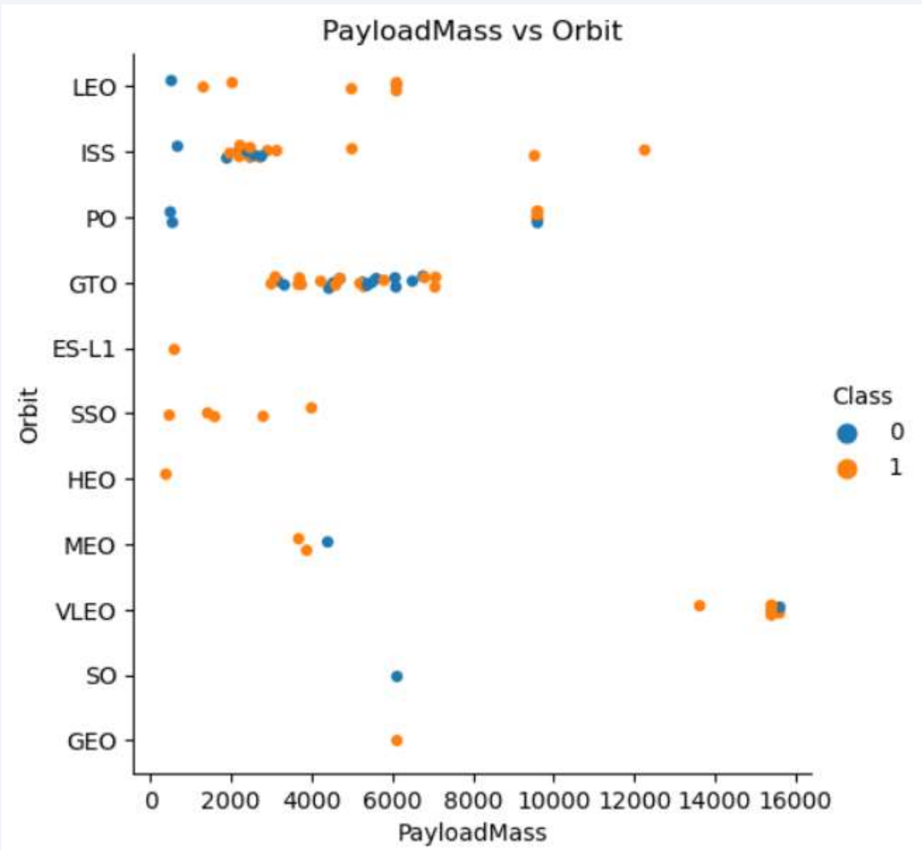
- 4 Orbit Type show 100% success rate: ES-L1, GEO, HEO and SSO
- 1 Orbit Type shows 0% success rate: 0%
- All Orbit Types, except SO, have higher rates equal or above 0.5

Flight Number vs. Orbit Type



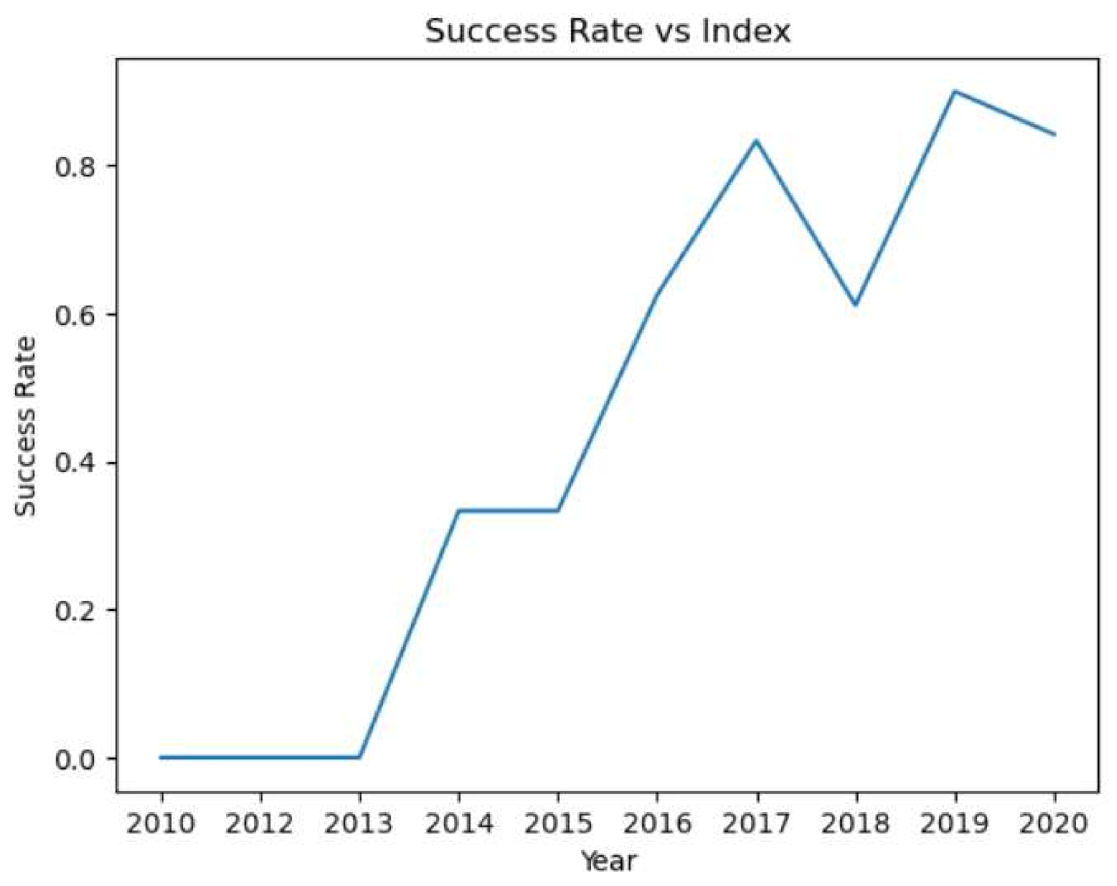
- Dominant Orbit Types changed from LEO, ISS, PO, and GTO to SSO, MEO, and VLEO, for later Flight Numbers
- GTO shows a high number of missions with a high number of failures
- Missions launched at GTO show slower increase of success rate when compared to LEO, ISS, and PO

Payload vs. Orbit Type



- LEO, ISS and PO show higher success likelihood of success for higher payloads
- GTO shows no clear relation between payload and mission success

Launch Success Yearly Trend



- Success rate increased with time since 2013

All Launch Site Names

- Unique launch sites: CCAFS LC-40, CCAFS SLC-40, KSC LC-39A, and VAFB SLC-4E
- This query looks for all launch sites, identifying all different names

```
%sql select distinct(launch_site) from spacex
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85
Done.
```

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

Launch Site Names Begin with 'CCA'

- This query aims to retrieve all data from 5 missions from any launching station with name starting with CCA

```
%sql SELECT * FROM spacex WHERE launch_site LIKE 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/BLUDB
Done.
```

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

- Total payload carried by boosters from NASA: 45596 kg
- This query recovers payload data from missions performed for NASA and sums them

```
%sql SELECT SUM(payload_mass__kg_) FROM spacex WHERE customer = 'NASA (CRS)'
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnr
Done.
```

1
45596

Average Payload Mass by F9 v1.1

- Average payload mass carried by booster version F9 v1.1: 2928 kg
- This query recovers payload data from missions using a F9 v1.1 booster and calculate its average

```
%sql SELECT AVG(payload_mass__kg_) FROM spacex WHERE booster_version = 'F9 v1.1';
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u9
Done.
```

1
2928

First Successful Ground Landing Date

- Date of the first successful landing outcome on ground pad: 12/22/2015
- This query request for the minimum date of a Success (ground) mission

```
%sql SELECT MIN(date) FROM spacex WHERE landing__outcome = 'Success (ground
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nq
Done.
```

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- This query ask for a list of names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

```
%sql SELECT booster_version FROM spacex WHERE landing__outcome = 'Success (drone ship)' AND payload_mass__kg_ > 4000 AND payload_mass__kg_ < 6000;
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/BLUDB
Done.
```

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

Total Number of Successful and Failure Mission Outcomes

- Total number of successful and failure mission outcomes
- This query defined success count based on whether a mission outcome is not null

```
%sql SELECT mission_outcome, COUNT(*) FROM spacex WHERE mission_outcome IS NOT NULL GROUP BY mission_outcome;
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/BLUDB
Done.
```

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

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Present your query result with a short explanation here

```
%sql SELECT booster_version FROM spacex WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM spacex)
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.clou
Done.
```

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

2015 Launch Records

- This query looks for failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
%sql SELECT landing__outcome, booster_version, launch_site FROM spacex WHERE YEAR(date) = 2015 AND landing__outcome LIKE '%Failure (drone ship)%';
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/BLUDB
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 Landing Outcomes Between 2010-06-04 and 2017-03-20

- Rank of 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

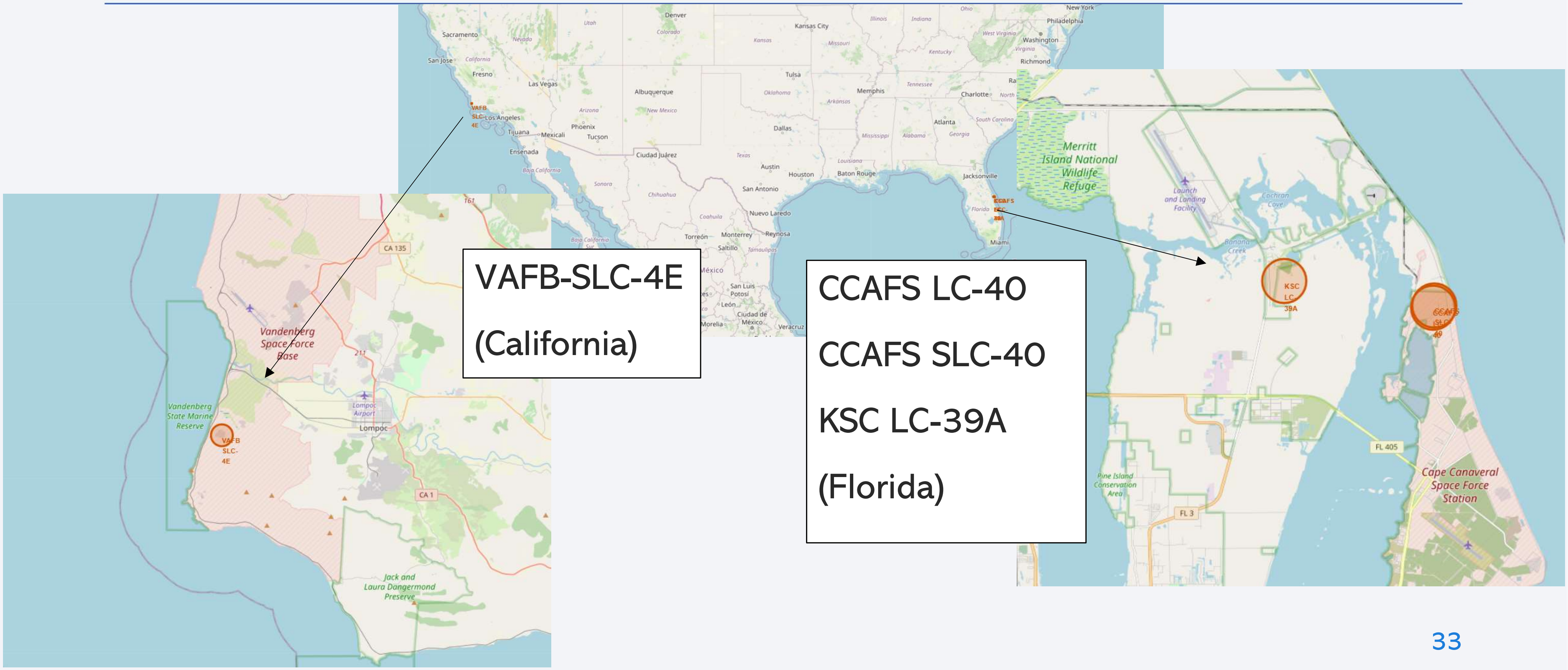
```
%sql SELECT landing__outcome, COUNT(*) AS count FROM spacex WHERE date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY landing__outcome ORDER BY count
```

```
 * ibm_db_sa://zfm23233:***@1bbf73c5-d84a-4bb0-85b9-ab1a4348f4a4.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:32286/BLUDB
Done.
```

landing__outcome	COUNT
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


Section 3
Launch Sites
Proximities Analysis

Launching Sites Location



VAFB-SLC-4E

(California)

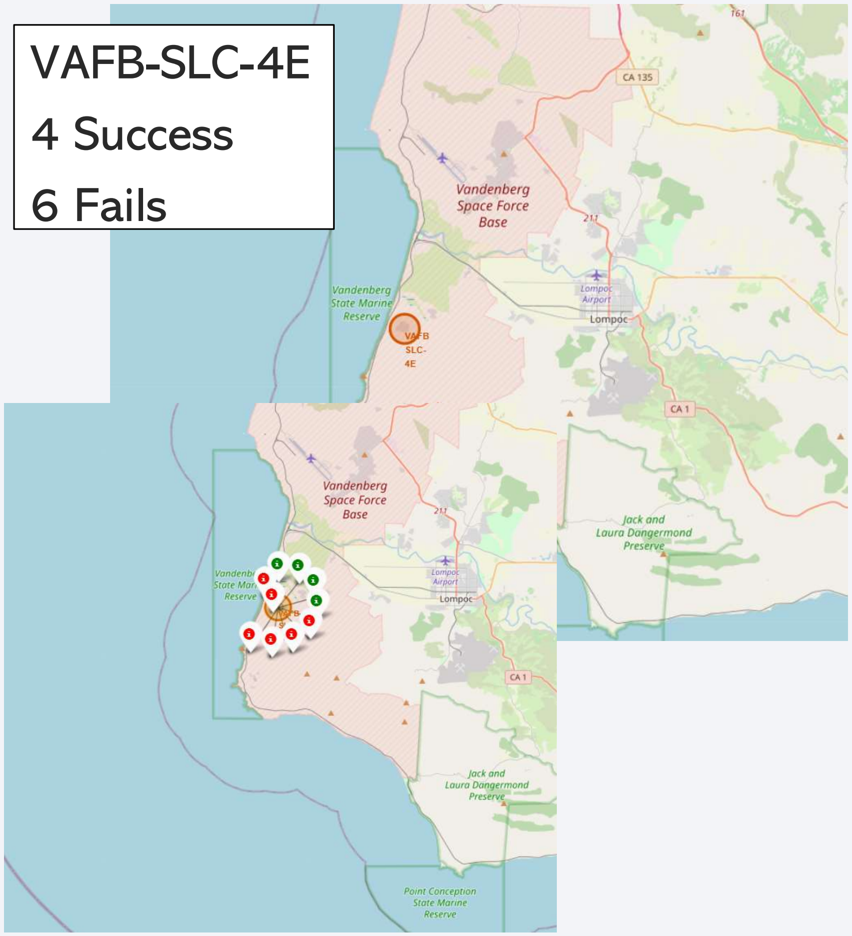
CCAFS LC-40

CCAFS SLC-40

KSC LC-39A

(Florida)

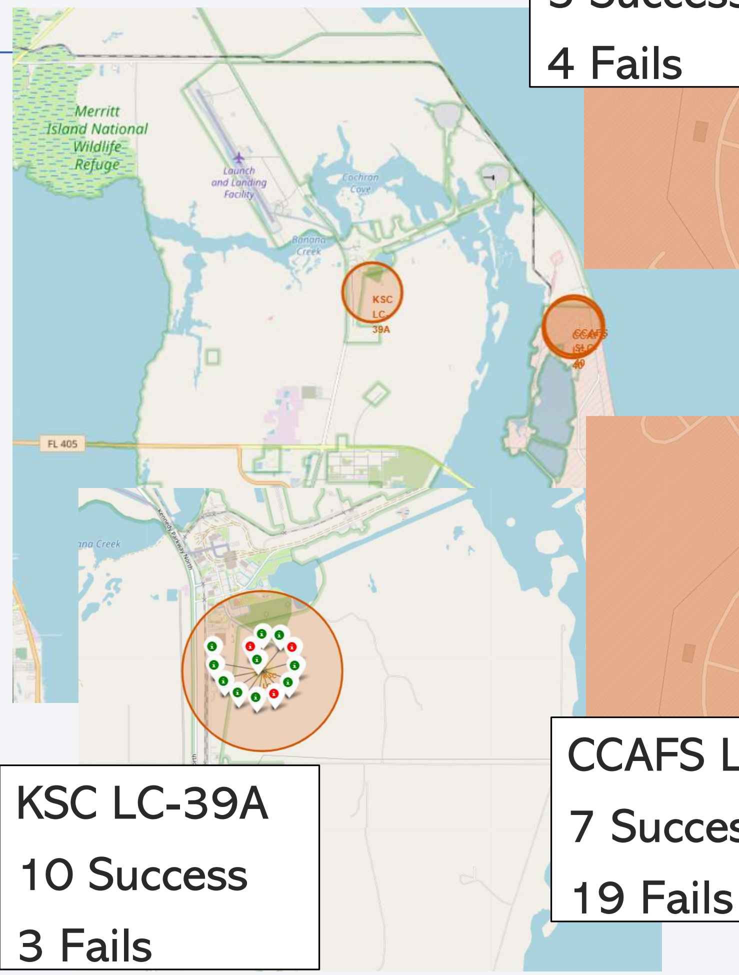
Launching Sites Success Rates



VAFB-SLC-4E

4 Success

6 Fails



KSC LC-39A

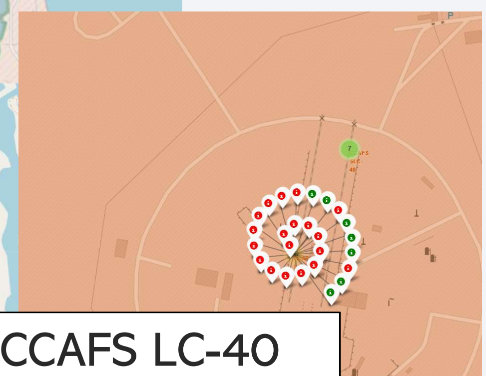
10 Success

3 Fails

CCAFS SLC-40

3 Success

4 Fails

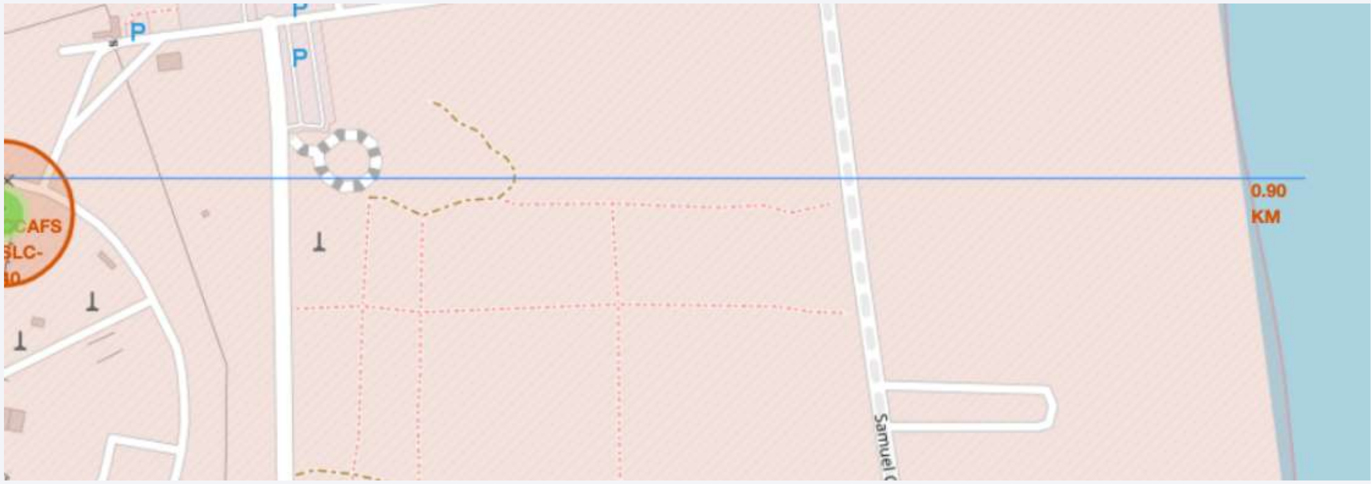


CCAFS LC-40

7 Success

19 Fails

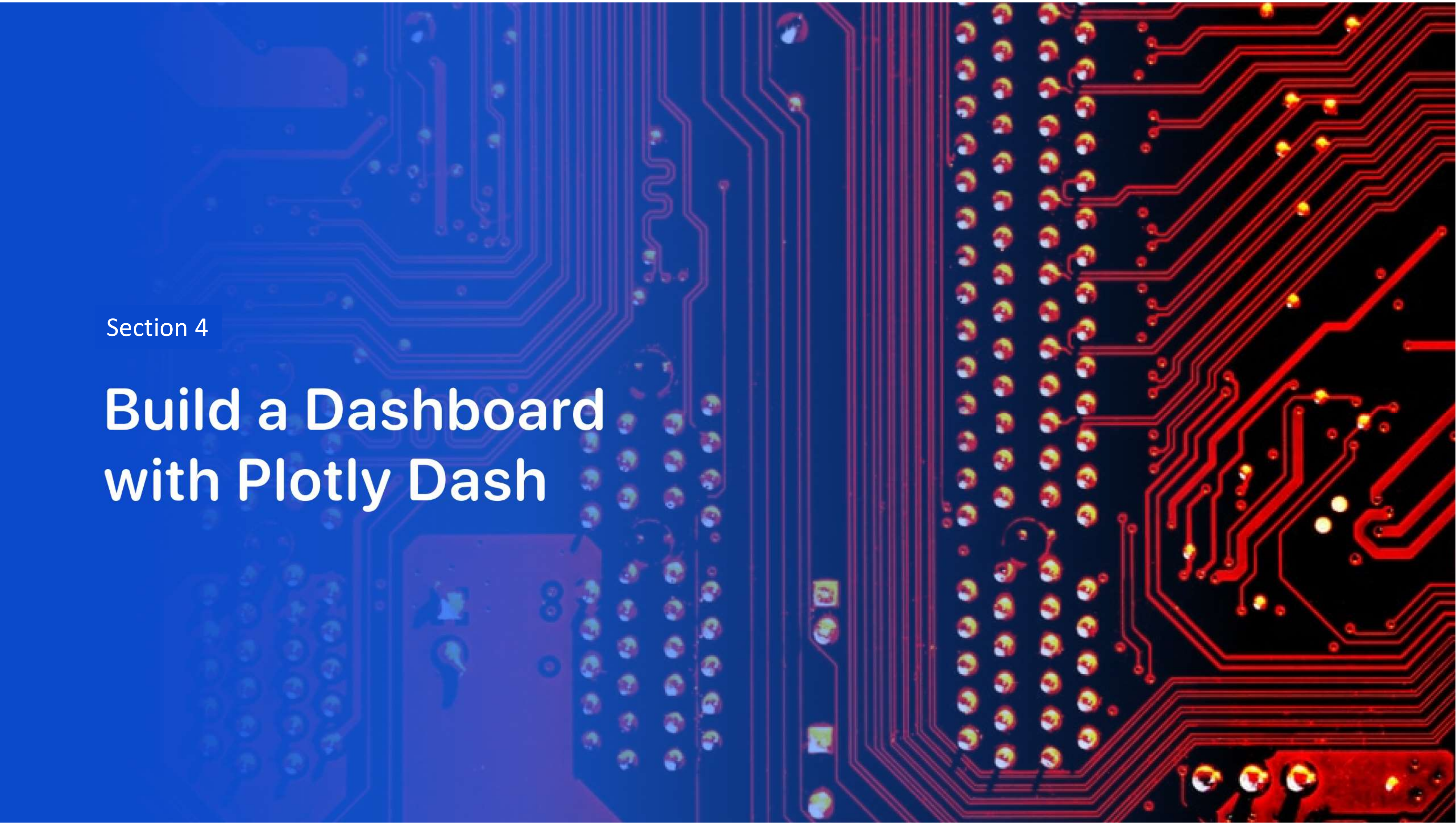
Features Proximity



Rechecked distance: CCAFS SLC-40 to coast is 0.9 Km

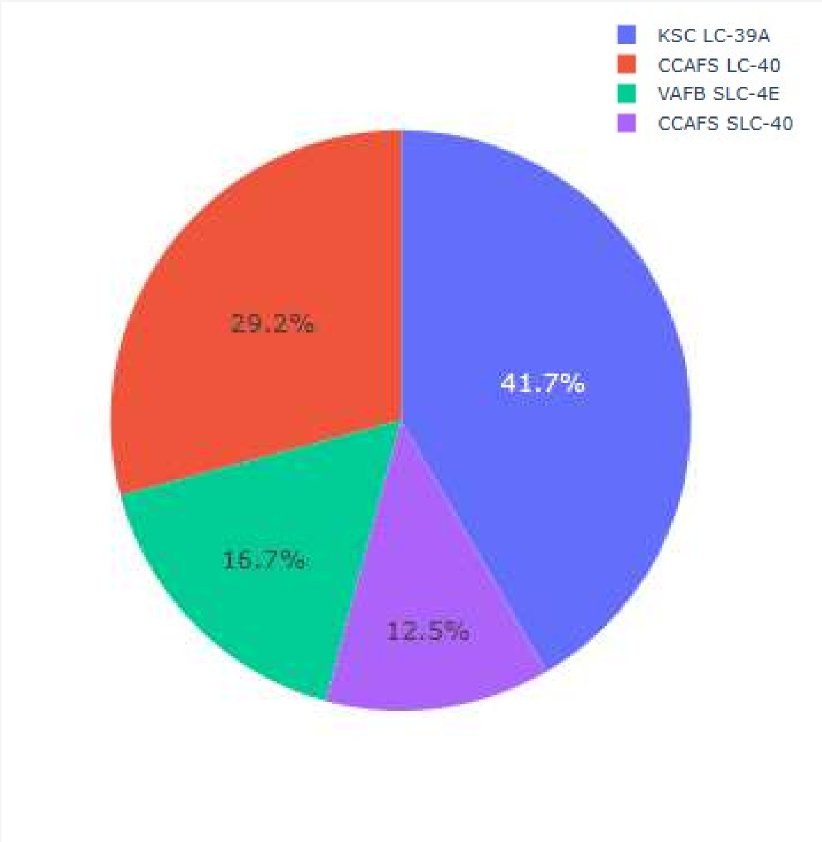
Section 4

Build a Dashboard with Plotly Dash



Success Count – All sites

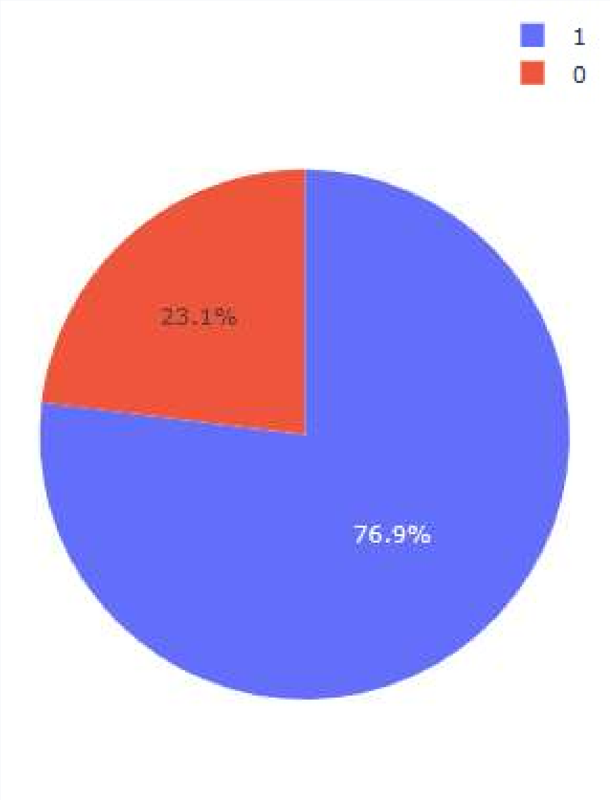
- Highest count of launch success is for KSC LC-39A
- Lowest count of launch success is for CCAFS SLC-40
- Values show total number of successful launches but not the best ratio



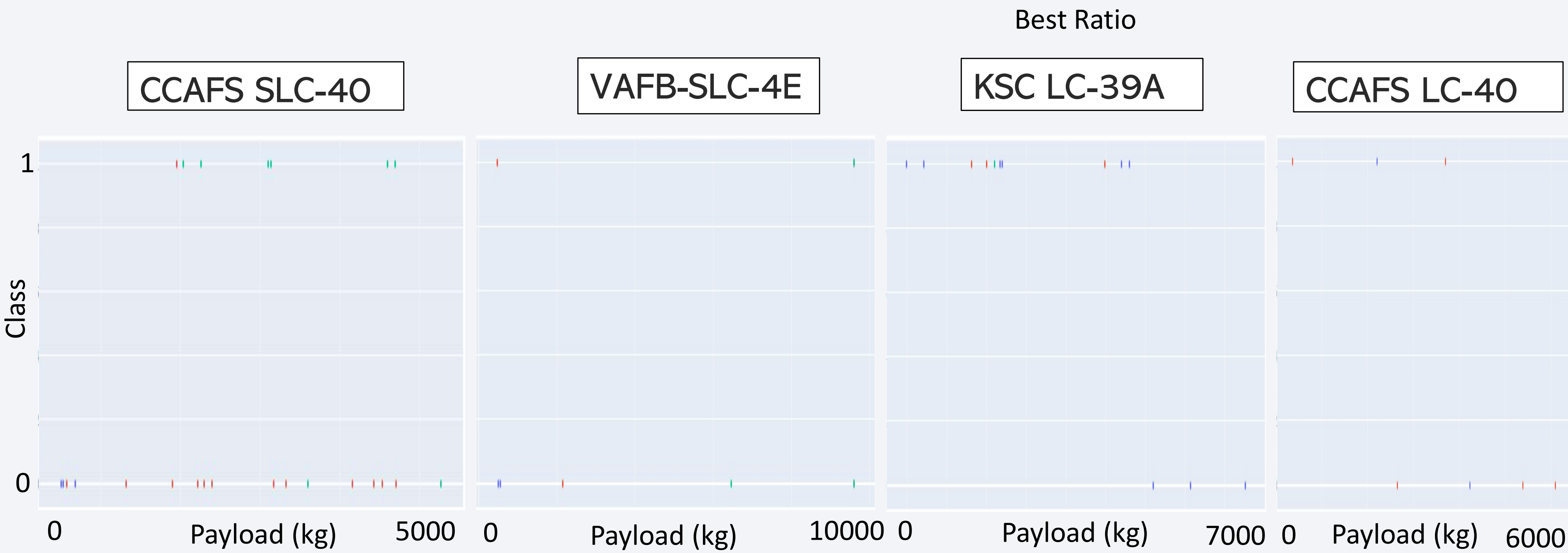
Success Ratio – Best Location

- Highest ratio of launch success is for KSC LC-39A

- Both highest number and ratio of launch success is for KSC LC-39A



Payload vs. Launch Outcome

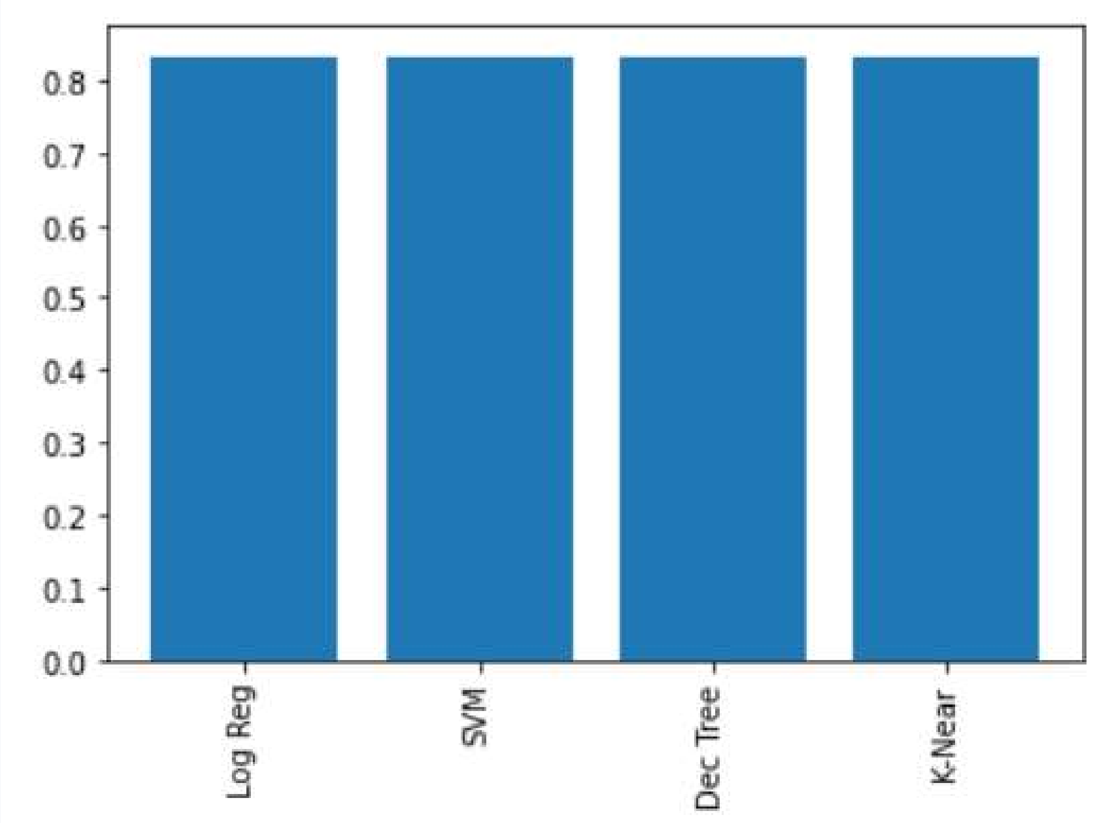


Section 5

Predictive Analysis (Classification)

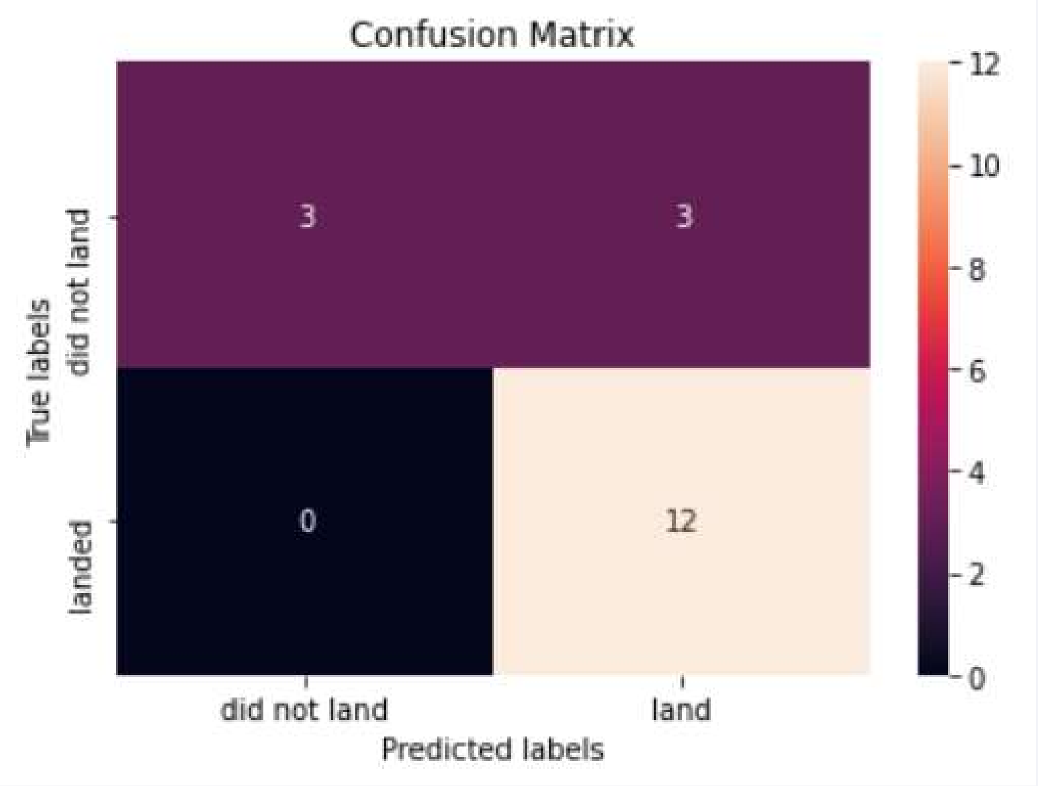
Classification Accuracy

- All models share same accuracy during prediction



Confusion Matrix

- Model is able to correctly predict successful landings for successful landings
- Model has issue predicting non-successful landings as successful landings



Conclusions

- Falcon 9 database allows to analyze mission outcomes probability in terms of multiple variables
- Visual EDA allowed to identify that there is a learning observed by the increase of success/fail ratio with respect of time
- Visual EDA allowed to identify that mission failure may be related to payload an orbit at LEO, ISS and PO. This is not observed for GTO
- Prediction analysis shows that Linear Regression, Support Machine Vector, Decision Tree, and K-Nearest Neighbor are equally successful to predict landing success based on our available data

Thank you!