IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

MS
17. 5. 2022

Outline

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

Executive Summary

- Summary of methodologies
  - Collected data via an API and Web Scaping
  - Data wrangling
  - EDA with SQL and also Data Visualization
  - Folium Visual Intepretation
  - Machine learning & prediction
- Summary of all results
  - EDA results
  - Folium visuals
  - Prediction results

Introduction

- Project background and context
  - Today we are witnessing a new space race, where SpaceX claims it has the cheapest flight to orbit available. For the purpose of this project, we are working for an alternate company, which goes against SpaceX. We will try to predict, whether the first stage is going to land or not.
- Problems you want to find answers
  - What are the parameters that determine if the rocket lands successfully
  - What are the conditions to the landing

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - We collected data via SpaceX API and Wikipedia
- Perform data wrangling
  - We used one-hot encoding
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
  - Data was collected through SpaceX API and Wikipedie (web scrapping)
  - Then we decoded JSON and transformed it into a Pandas dataframe
  - We also cleaned the data and removed/filled missing values

Data Collection – SpaceX API

- Present your data collection with SpaceX REST calls using key phrases and flowcharts:
    - As seen on the right, first I used GET method to get data from API
    - Then I used .json() function to decode it
    - And then .json_normalize() to turn the JSON format into Pandas dataframe

- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose

In [9]:
```
static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json'
```

We should see that the request was successfull with the 200 status response code

In [10]:
```
response.status_code
```

Out[10]: 200

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

response = requests.get(static_json_url)
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

Using the dataframe `data` print the first 5 rows

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

Out[13]:

static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	crew	ships	capsules	payloads	launchpad
							Engine failure at					

Data Collection - Scraping

- Present your web scraping process using key phrases and flowcharts:
  - First I search for “table” in html
  - Then I can target column names
  - And then iterate through table with for loop
- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose

In [21]:
```python
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
#html_tables
```

Starting from the third table is our target table contains the actual launch records.

In [11]:
```python
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC
</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">
Version,<br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#cite_note-booster-11">[b]</a></
sup>
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-12">[c]</a></sup>
</th>
<th scope="col">Payload mass
</th>
<th scope="col">Orbit
</th>
<th scope="col">Customer
</th>
<th scope="col">Launch<br/>outcome
</th>
<th scope="col"><a href="/wiki/Falcon_9_first-stage_landing_tests" title="Falcon 9 first-stage landing tests">Booste
r<br/>landing</a>
</th></tr>
<tr>
```

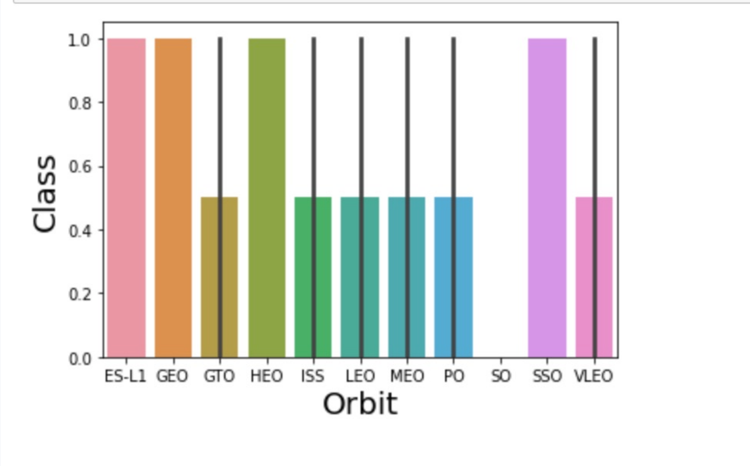
Data Wrangling

- Describe how data were processed:
  - At data wrangling lab, I first looked at the data we have
  - There are various methods for this, but I first looked at PD head
  - Then I checked for null values and data types that were not appropriate
  - Then I looked at various launch sites
  - Then of course, the landing outcomes

- Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose

EDA with Data Visualization

- Summarize what charts were plotted and why you used those charts:
  - I used matplotlib and seaborn for plotting the data
  - Plots used were: Scatter plot, Bar chart and of course plain dataframe too



- Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - First I looked at DISTINCT Launch sites with select query
  - Then all that had “CCA …” in name
  - Then also the sum of payload and average
  - Min date
  - Mission outcomes
- Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

Build an Interactive Map with Folium

- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map:
  - With folium we added circled markers, with folium.circle() to the map to visualize launch sites
  - Then we added launch results to the map using folium.marker()
  - We also calculated the distance between launch sites and proximities

- Explain why you added those objects
  - We added the objects to visualize what launch sites look like and where they are
- Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
  - I didn’t do the Plotly Dash, because I had problems running python on my computer and couldn’t use the Jupyter Notebooks for this lab unfortunately.
- Explain why you added those plots and interactions
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

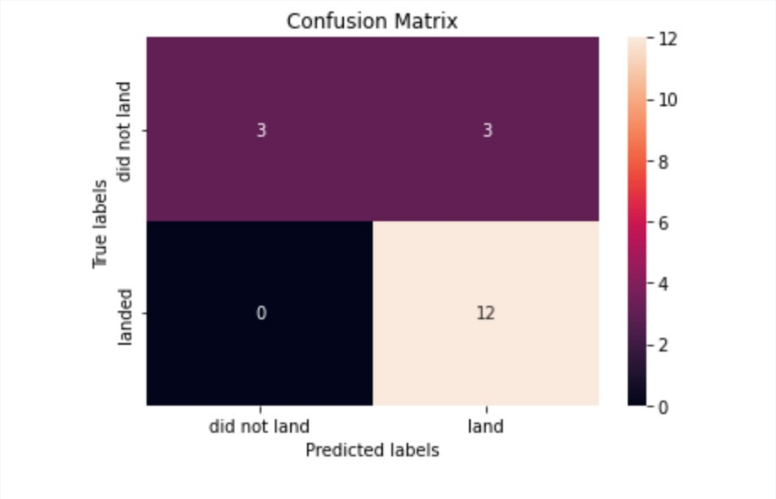
Predictive Analysis (Classification)

- Summarize how you built, evaluated, improved, and found the best performing classification model
  - We needed to first preprocess data with StandardScaller()
  - We divided data into train and test sets
  - Then we used logistic regression, SVM, Decision trees, KNN
  - In the end we printed calculated accuracies:

```
print(methods)
print(accuracy)

print(be)
```

```
['logistic regression', 'SVM', 'decision tree classifier', 'Decision tree', 'KNN']
[0.8333333333333334, 0.8333333333333334, 0.9444444444444444, 0.9444444444444444, 0.8333333333333334]
```



- Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

Results

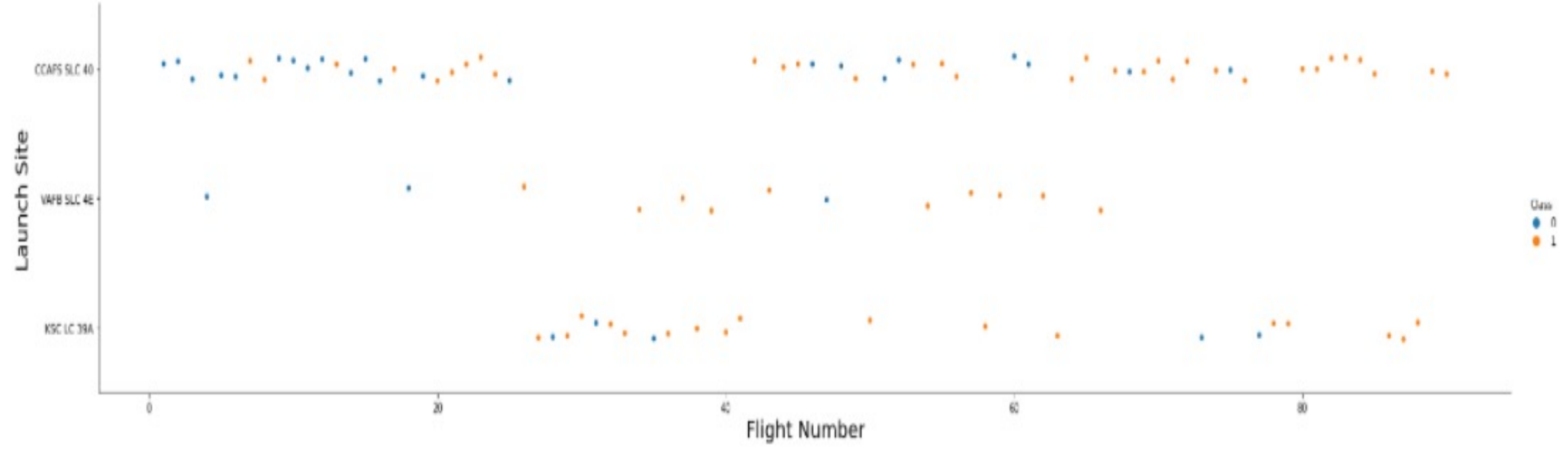
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

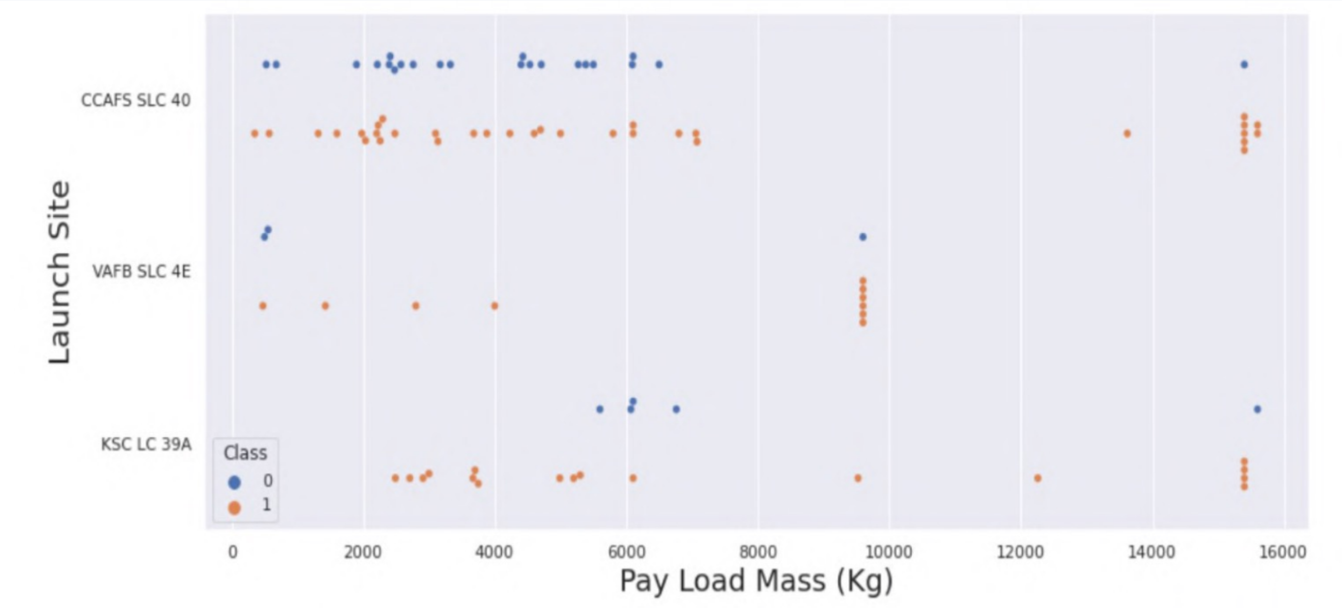
- Show a scatter plot of Flight Number vs. Launch Site



- The more flights came from a specific launch site, the more successful the flight.

Payload vs. Launch Site

- Show a scatter plot of Payload vs. Launch Site

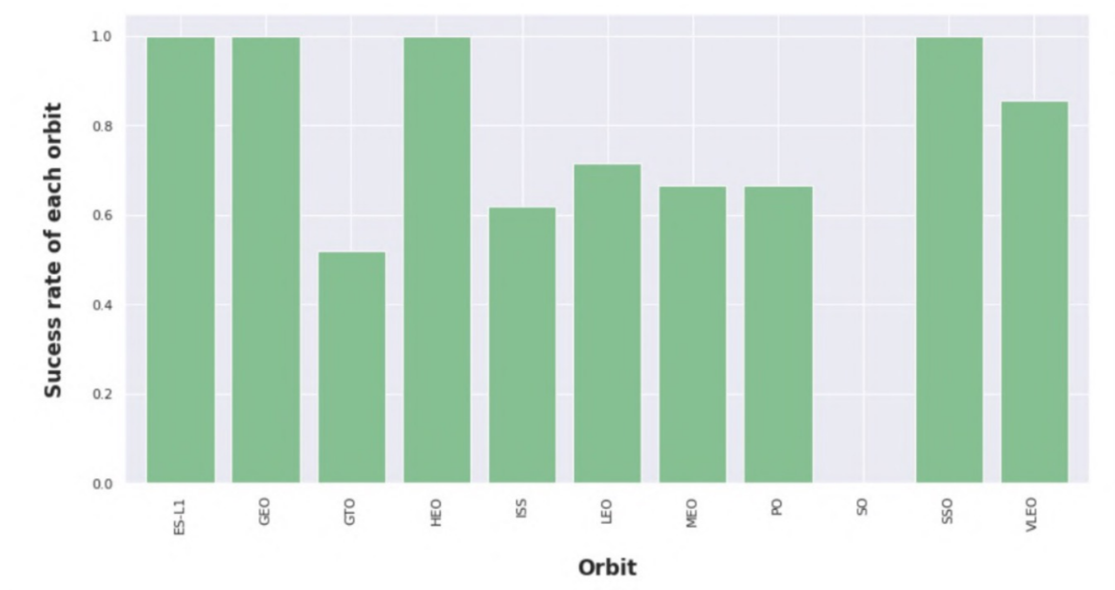


- Bigger payload means greater success in landing!

Success Rate vs. Orbit Type

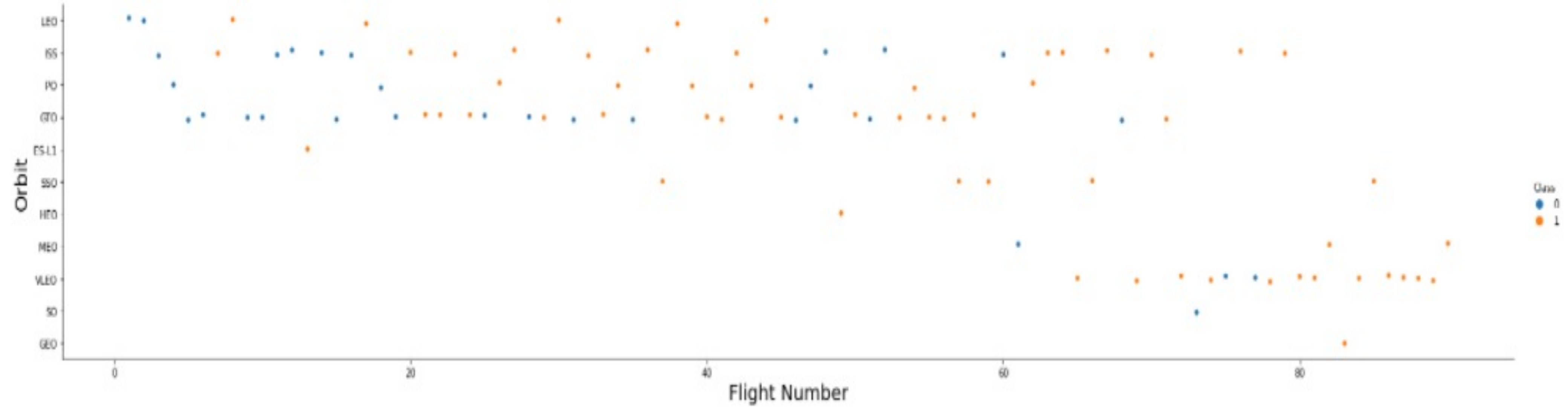
- Show a bar chart for the success rate of each orbit type

- The bar chart shows that the different orbit types have different success ratings



Flight Number vs. Orbit Type

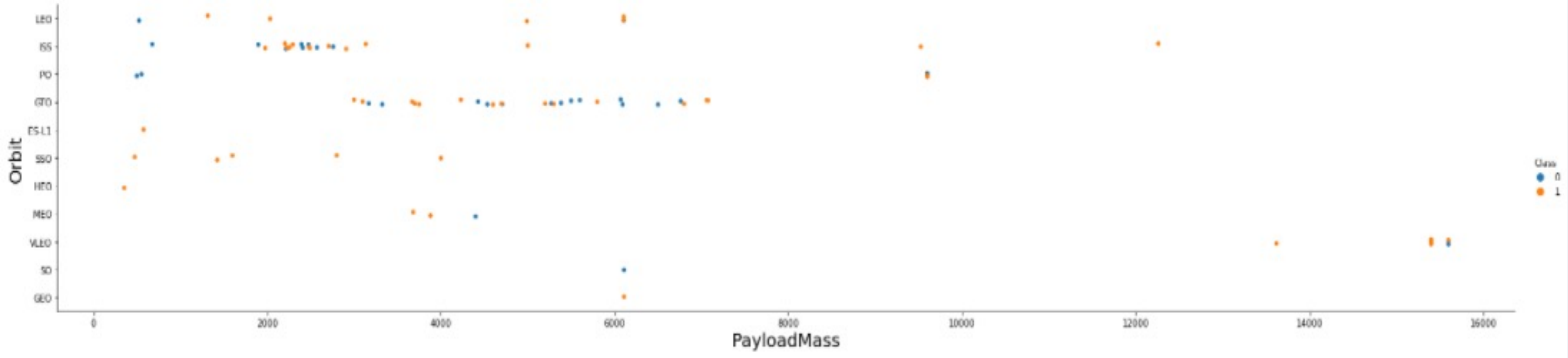
- Show a scatter point of Flight number vs. Orbit type
- Below scatter plot shows what flight numbers and orbits have success or unsuccessful outcomes



Payload vs. Orbit Type

- Show a scatter point of payload vs. orbit type

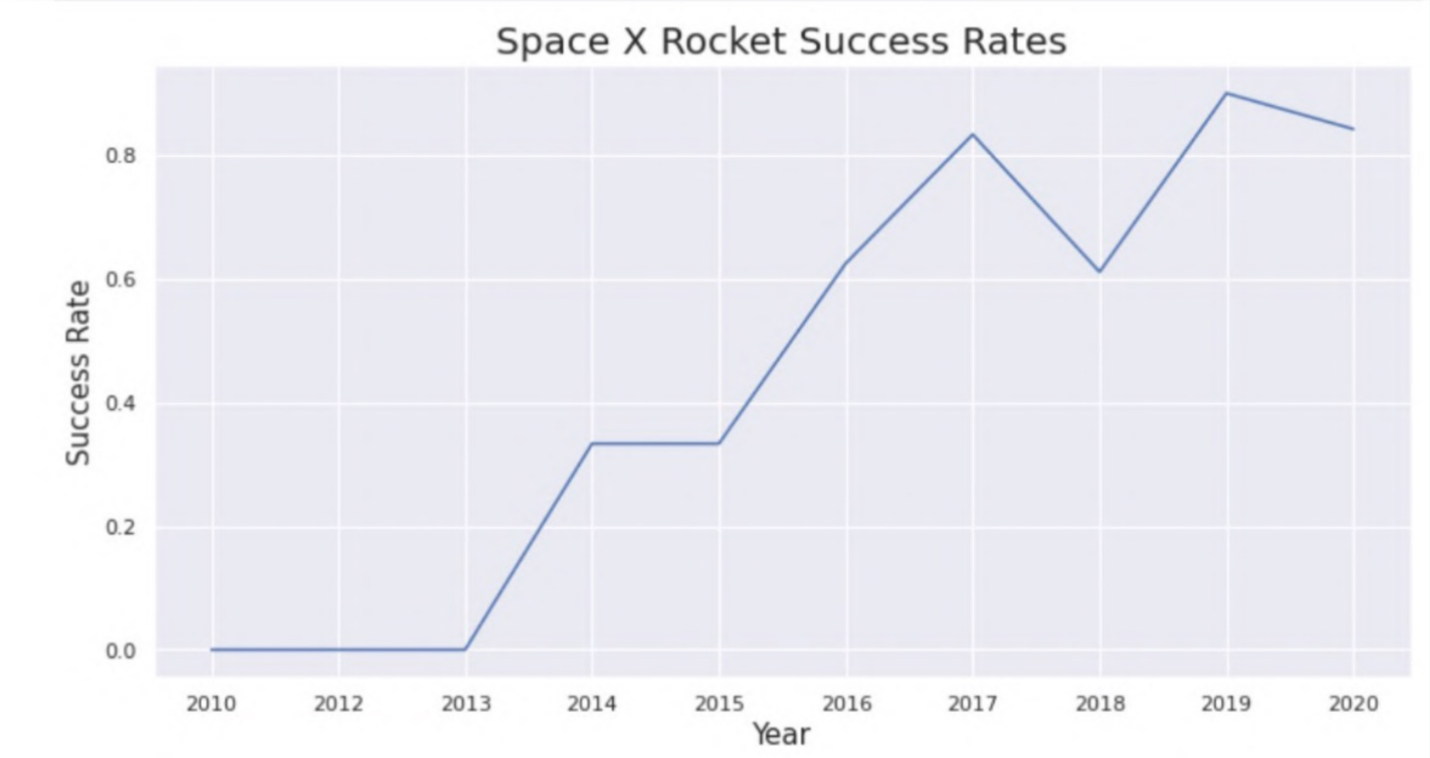
- This scatter plot shows the correlation between payload mass, orbit and outcome



Launch Success Yearly Trend

- Show a line chart of yearly average success rate

- This chart shows how the success rate grew over time at SpaceX



All Launch Site Names

- Find the names of the unique launch sites
- Selected distinct (different) launch sites from the table

```
%sql select DISTINCT LAUNCH_SITE from SPACEXTBL
```

Out[6]:

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

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA

```
In [8]: %sql select * from SPACEXTBL where launch_site like 'CCA%' limit 5
```

```
 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3087
5/bludb
Done.
```

Out[8]:

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

- Calculate the total payload carried by boosters from NASA
- Total sum is 45 596 kg

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

```
[10]: %sql select sum(payload_mass__kg_) as sum from SPACEXTBL where customer like 'NASA (CRS)'
```

```
 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3087
5/bludb
Done.
```

[10]:

SUM
45596

Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
- Average payload mass for F9 v1.1 is 2534 kg

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

[11]:
```
%sql select avg(payload_mass__kg_) as Average from SPACEXTBL where booster_version like 'F9 v1.1%'
```

```
 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

t[11]:

average
2534

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad
- First successful landing happened on 2010 06 04

```
In [12]: %sql select min(date) as Date from SPACEXTBL where mission_outcome like 'Success'

 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3087
5/bludb
Done.

Out[12]:
```

DATE
2010-06-04

Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

- Present your query result with a short explanation here

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes

List the total number of successful and failure mission outcomes

In [16]:
```
%sql SELECT mission_outcome, count(*) as Count FROM SPACEXTBL GROUP by mission_outcome ORDER BY mission_outcome
```

```
 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3087
5/bludb
Done.
```

Out[16]:

mission_outcome	COUNT
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

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [19]:
```
%sql select booster_version from SPACEXTBL where payload_mass__kg_ in (select max(payload_mass__kg_) from SPACEXTB
L)
```

```
 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3087
5/bludb
Done.
```

Out[19]:

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

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

```
In [23]: %sql select landing__outcome, booster_version, launch_site from SPACEXTBL where (DATE like '2015%') AND landing__ou
tcome like 'Failure (drone ship)'

 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:3087
5/bludb
Done.
```

Out[23]:

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 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

```
9]: %sql select landing__outcome, Date from SPACEXTBL where Date >= '2010-06-04' AND Date <= '2017-03-20' ORDER BY Date DESC

 * ibm_db_sa://wzn68468:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

Out[29]:

landing__outcome	DATE
No attempt	2017-03-16
Success (ground pad)	2017-02-19
Success (drone ship)	2017-01-14
Success (drone ship)	2016-08-14
Success (ground pad)	2016-07-18
Failure (drone ship)	2016-06-15
Success (drone ship)	2016-05-27
Success (drone ship)	2016-05-06
Success (drone ship)	2016-04-08
Failure (drone ship)	2016-03-04
Failure (drone ship)	2016-01-17
Success (ground pad)	2015-12-22
Precluded (drone ship)	2015-06-28
No attempt	2015-04-27
Failure (drone ship)	2015-04-14
No attempt	2015-03-02
Controlled (ocean)	2015-02-11
Failure (drone ship)	2015-01-10
Uncontrolled (ocean)	2014-09-21
No attempt	2014-09-07

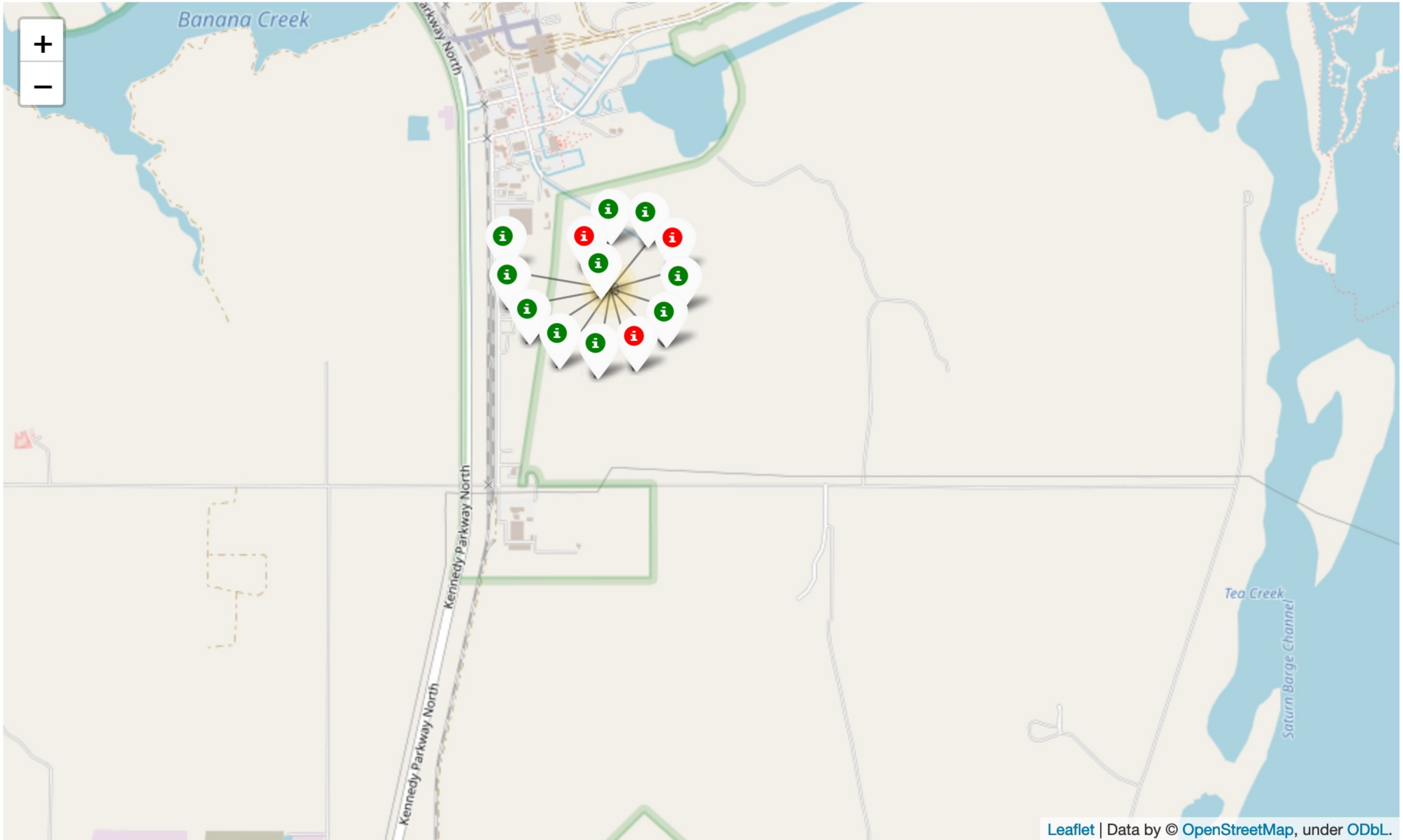
No attempt	2013-12-03
Uncontrolled (ocean)	2013-09-29
No attempt	2013-03-01
No attempt	2012-10-08
No attempt	2012-05-22
Failure (parachute)	2010-12-08
Failure (parachute)	2010-06-04

Section 3

Launch Sites Proximities Analysis

<Folium Map Screenshot 1>

<Folium Map Screenshot 2>



<Folium Map Screenshot 3>

Section 4

Build a Dashboard with Plotly Dash

<Dashboard Screenshot 1>

- Unfortunatelly, I couldn’t complete this lab as my computer didn’t support python and I couldn’t do it in Jupyter Notebooks

<Dashboard Screenshot 2>

- Unfortunatelly, I couldn't complete this lab as my computer didn't support python and I couldn't do it in Jupyter Notebooks

<Dashboard Screenshot 3>

- Unfortunatelly, I couldn't complete this lab as my computer didn't support python and I couldn't do it in Jupyter Notebooks

Section 5

Predictive Analysis (Classification)

Classification Accuracy

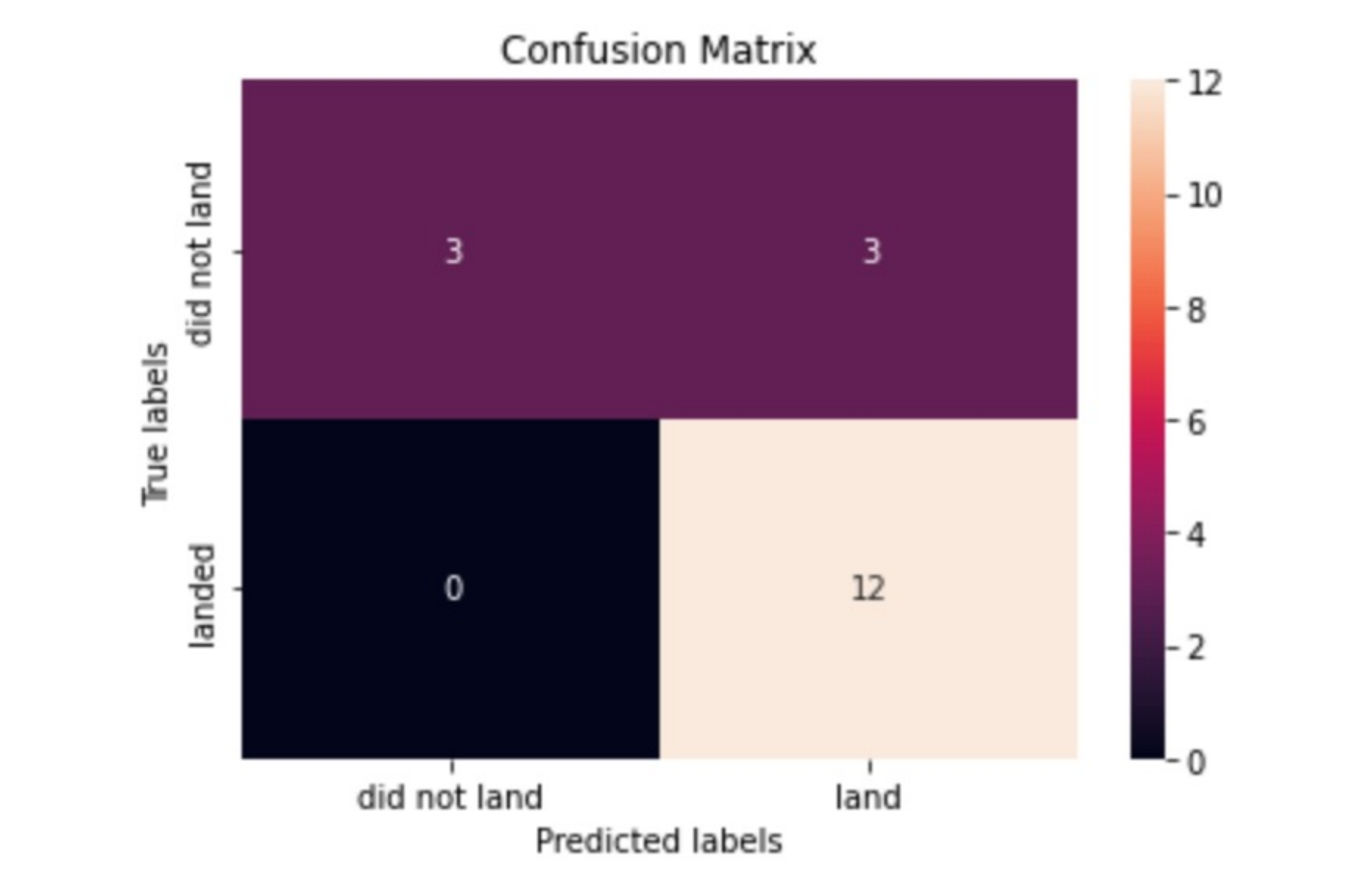
- The overall best models was Decision tree with 94,444 % accuracy

```
print(methods)
print(accuracy)

print(be)
```

```
['logistic regression', 'SVM', 'decision tree classifier', 'Decision tree', 'KNN']
[0.8333333333333334, 0.8333333333333334, 0.9444444444444444, 0.9444444444444444, 0.8333333333333334]
```

Confusion Matrix



Conclusions

- In conclusion, different orbits have very different outcomes
- Weird, but true, is that the larger the payload amount, the greater the success rate of the landing
- There is an overall increase in landing outcomes by years
- Decision tree model was by far better than other predictive models

Thank you!