Data Science Capstone

Cláudia

July 2023

Outline

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

Executive Summary

Summary of methodologies

- Space X Data Collection using Space X API
- Space X Data Collection with Web Scraping
- Space X Data Wrangling
- Space X Exploratory Data Analysis using SQL
- Space X EDA DataViz Using Python Pandas and Matplotlib
- Space X Launch Sites Analysis with Folium-Interactive Visual Analytics and PlotyDash
- Space X Machine Learning Landing Prediction

Summary of all results

- EDA results
- Interactive Visual Analytics and Dashboards
- Predictive Analysis(Classification)

Background and problem-solving

- Space X Falcon 9 rocket launches has a cost of $62 million
- Other providers cost upward of $165 million each
- Save $$ because Space X can reuse the first stage

- Determine the cost of a launch
- Information can be used for market competition
- This capstone is to predict if the Falcon 9 first stage will land successfully using data from Falcon 9 rocket launches advertised on its website.

Methods

- Data collection:

  Describes how data sets were collected

- Perform data wrangling

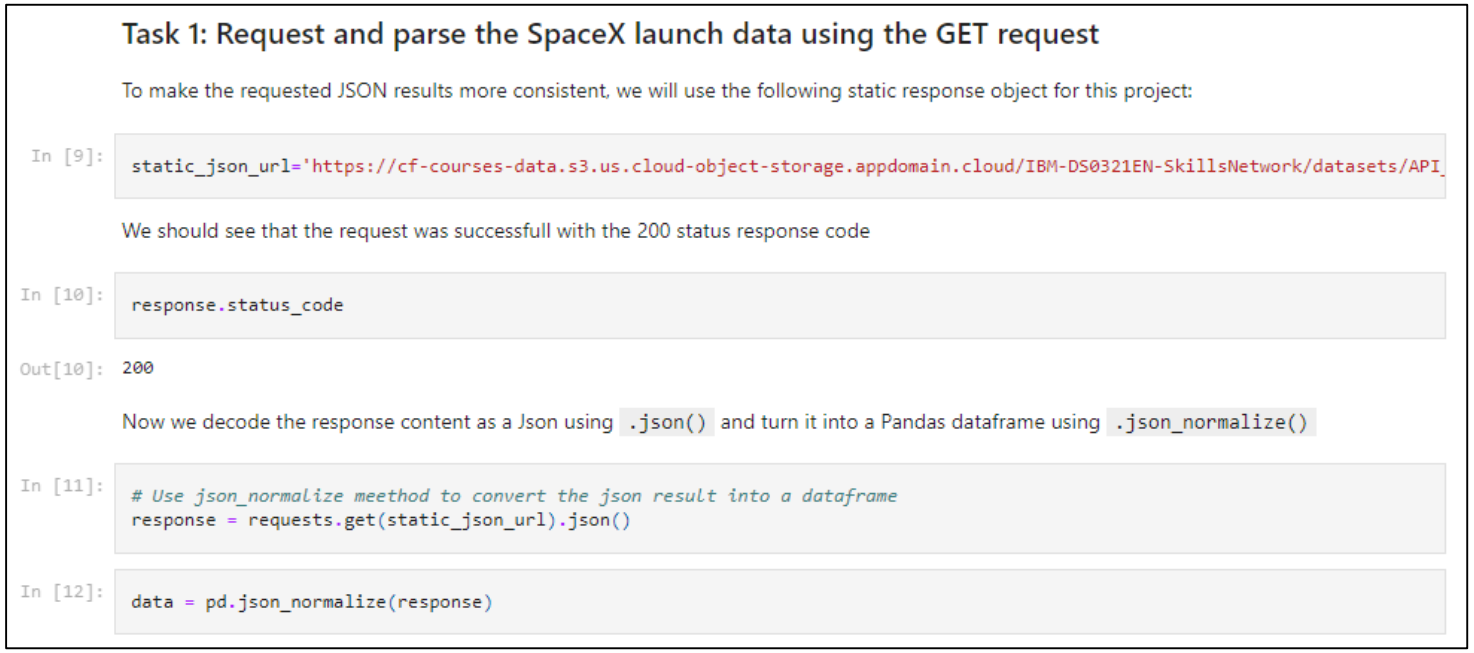
  Describes how data were processed

- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and PlotlyDash
- Perform predictive analysis using classification models

  How to build, tune and evaluate classification models

Data Collection

- GET request to the SpaceX API
- Request and parse data
- Request and decode response content - Json
- Convert into a Pandas data frame



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

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

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 [11]:
```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url).json()
```

In [12]:
```
data = pd.json_normalize(response)
```

URL: https://github.com/ClaudiaBrambilla/datasciencecoursera/blob/master/1_jupyter_labs_spacex_data_collection_api_final.ipynb

Data Collection – Web Scraping

- Web Scraping request to collect Falcon 9
- historical launch records from a Wikipedia
- Beautiful Soup and request
- Falcon 9 launch records from HTML table
- Wikipedia page
- Create a data frame by parsing the launch HTML

URL: https://github.com/ClaudiaBrambilla/datascience coursera/blob/master/2_jupyter_labs_webscraping_f.ipynb

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 [7]:
```
# 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`

In [8]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [9]:
```
# Use soup.title attribute
soup.title
```

Out[9]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

In [10]:
```
# 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')
```

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

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

Data Wrangling

- Data filtered - Booster Version column - Falcon 9 launches
- Missing data values in the Landing
- Missing data replaced using mean value of column
- Exploratory Data Analysis (EDA) - patterns in the data to determine label for training supervised models

URL:

https://github.com/ClaudiaBrambilla/datasciencecourse ra/blob/master/3_labs-jupyter-spacex-Data%20wrangling_comp.ipynb

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

In [113…

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise

landing_class = np.where(df['Outcome'].isin(set(bad_outcomes)), 0, 1)
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

In [114…

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

Out[114…

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

EDA – SQL Queries

- Launch site
- 5 records where launch sites started with CCA
- Total payload mass carried - NASA
- Average payload mass carried – F9 v1.1
- Date – successful landing achievement

URL:

https://github.com/ClaudiaBrambilla/datasciencecoursera/blob/r/4_SpaceX_EDA_SQL.ipynb

Task 1

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

```
In [31]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
```

```
 * sqlite:///my_data1.db
Done.
```

Out[31]:

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

Task 2

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

```
In [72]: %sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
```

```
 * sqlite:///my_data1.db
Done.
```

Out[72]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
								Success	No attempt
								Success	No attempt

Task 3

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

```
In [17]: %sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
```

```
 * sqlite:///my_data1.db
Done.
```

Out[17]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Task 4

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

```
In [19]: %sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version
```

```
 * sqlite:///my_data1.db
Done.
```

Out[19]:

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
In [21]: %sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
```

```
 * sqlite:///my_data1.db
Done.
```

Out[21]:

MIN(DATE)
01-05-2017

EDA - Data Visualization

- Pandas and Matplotlib:

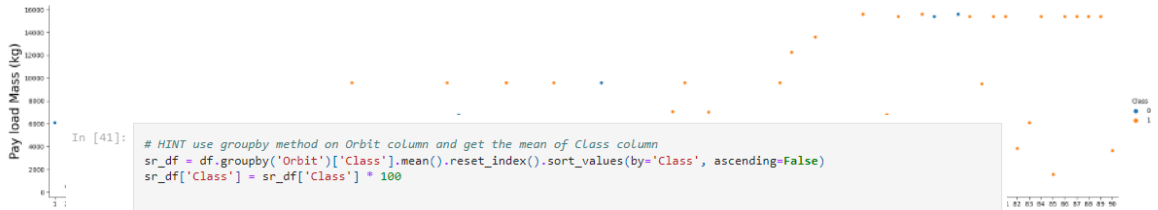
 - EDA and Prepare Data – Future Engineering

- Scatter plots – relationship between variables
- Bar charts – success rate per orbit type
- Line plots – success launches over time - trend

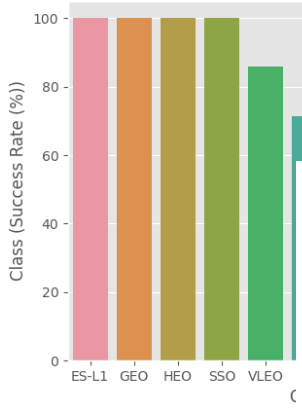
URL:

https://github.com/ClaudiaBrambilla/datasciencecoursera/blob/master/5_SpaceXEDA_DataViz_Pandas_Matplotlib.ipynb

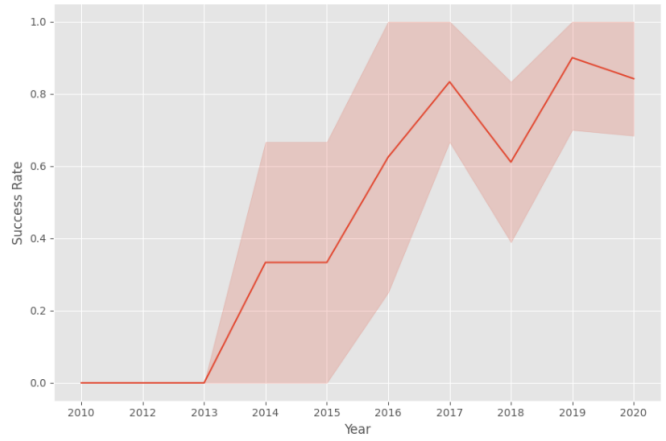
In [14]:
```
sns.catplot(y="PayloadMass", x="FlightNumber", hue="Class", data=df, aspect = 5, height=5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Pay load Mass (kg)",fontsize=20)
plt.show()
```



In [41]:
```
# HINT use groupby method on Orbit column and get the mean of Class column
sr_df = df.groupby('Orbit')['Class'].mean().reset_index().sort_values(by='Class', ascending=False)
sr_df['Class'] = sr_df['Class'] * 100


sns.barplot(data=sr_df, x='Orbit', y='Class')
plt.xlabel('Orbit')
plt.ylabel('Class (Success Rate (%))')
plt.show()
```



In [55]:
```
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
sns.lineplot(data=df, x="Date", y="Class")
plt.xlabel("Year")
plt.ylabel("Success Rate")
plt.show()
```



Iterative Maps with Folium

- Mark all the launch sites
- Mark the success or failure of launches per site
- Launch set outcomes (failure=0 or success=1)

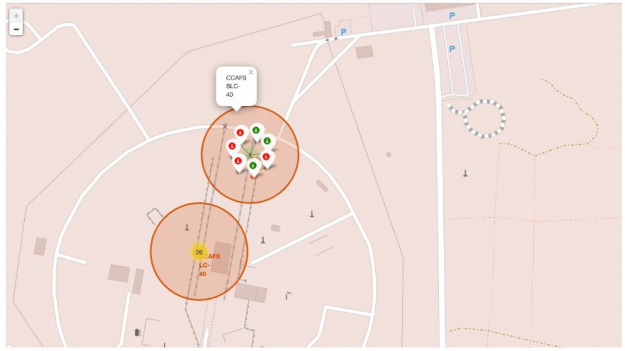
Task 2: Mark the success/failed launches for each site on the map

Next, let's try to enhance the map by adding the launch outcomes for each site, and see which sites have high success rates. Recall that data frame spacex_df has detailed launch records, and the `class` column indicates if this launch was successful or not

In [10]:
```
spacex_df.tail(10)
```

Out[10]:

	Launch Site	Lat	Long	class
46	KSC LC-39A	28.573255	-80.646895	1
47	KSC LC-39A	28.573255	-80.646895	1
48	KSC LC-39A	28.573255	-80.646895	1
49	CCAFS SLC-40	28.563197	-80.576820	1
50	CCAFS SLC-40	28.563197	-80.576820	1
51	CCAFS SLC-40	28.563197	-80.576820	0
52	CCAFS SLC-40	28.563197	-80.576820	0
53	CCAFS SLC-40	28.563197	-80.576820	0
54	CCAFS SLC-40	28.563197	-80.576820	1
55	CCAFS SLC-40	28.563197	-80.576820	0



From the color-labeled markers in marker clusters, you should be able to easily identify which launch sites have relatively high success rates.

URL:

https://github.com/ClaudiaBrambilla/datasciencecoursera/blob/master/6_Space-X_LaunchSitesLocationsAnalysisFolium.ipynb

Predictive Analysis - Classification

- Built, evaluated, improved, and find the best performance:

  - classification model

- Models tested:

  -SVM, Classification Trees, k nearest neighbors

  and Logistic Regression

TASK 12

Find the method performs best:

In [68]:

```
Report = pd.DataFrame({'Method' : ['Test Data Accuracy']})

knn_accuracy=knn_cv.score(X_test, Y_test)
Decision_tree_accuracy=tree_cv.score(X_test, Y_test)
SVM_accuracy=svm_cv.score(X_test, Y_test)
Logistic_Regression=logreg_cv.score(X_test, Y_test)

Report['Logistic_Reg'] = [Logistic_Regression]
Report['SVM'] = [SVM_accuracy]
Report['Decision Tree'] = [Decision_tree_accuracy]
Report['KNN'] = [knn_accuracy]



Report.transpose()
```

Out[68]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333

URL:
https://github.com/ClaudiaBrambilla/datasciencecoursera/blob/master/8_SpaceXMLPrediction.ipynb

Results

Task 1

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

```
In [31]: %sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEXTBL;
 * sqlite:///my_data1.db
Done.
```

Out[31]:

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

Task 2

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

```
In [72]: %sql SELECT * FROM 'SPACEXTBL' WHERE Launch_Site LIKE 'CCA%' LIMIT 5;
 * sqlite:///my_data1.db
Done.
```

Out[72]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Task 3

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

```
In [17]: %sql SELECT SUM(PAYLOAD_MASS__KG_) as "Total Payload Mass(Kgs)", Customer FROM 'SPACEXTBL' WHERE Customer = 'NASA (CRS)';
 * sqlite:///my_data1.db
Done.
```

Out[17]:

Total Payload Mass(Kgs)	Customer
45596	NASA (CRS)

Task 4

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

```
In [19]: %sql SELECT AVG(PAYLOAD_MASS__KG_) as "Payload Mass Kgs", Customer, Booster_Version FROM 'SPACEXTBL' WHERE Booster_Version l
 * sqlite:///my_data1.db
Done.
```

Out[19]:

Payload Mass Kgs	Customer	Booster_Version
2534.6666666666665	MDA	F9 v1.1 B1003

Task 5

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
In [21]: %sql SELECT MIN(DATE) FROM 'SPACEXTBL' WHERE "Landing _Outcome" = "Success (ground pad)";
 * sqlite:///my_data1.db
Done.
```

Out[21]:

MIN(DATE)
01-05-2017

Results

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 [26]:
```
# %sql SELECT * FROM 'SPACEXTBL'
```

In [27]:
```
%sql SELECT DISTINCT Booster_Version, Payload FROM SPACEXTBL WHERE "Landing _Outcome" = "Success (drone ship)" AND PAYLOAD_
```

```
 * sqlite:///my_data1.db
Done.
```

Out[27]:

Booster_Version	Payload
F9 FT B1022	JCSAT-14
F9 FT B1026	JCSAT-16
F9 FT B1021.2	SES-10
F9 FT B1031.2	SES-11 / EchoStar 105

Task 7

List the total number of successful and failure mission outcomes

In [28]:
```
%sql SELECT "Mission_Outcome", COUNT("Mission_Outcome") as Total FROM SPACEXTBL GROUP BY "Mission_Outcome";
```

```
 * sqlite:///my_data1.db
Done.
```

Out[28]:

Mission_Outcome	Total
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Results

Task 8

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

In [30]:
```
%sql SELECT "Booster_Version",Payload, "PAYLOAD_MASS__KG_" FROM SPACEXTBL WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_
```

```
* sqlite:///my_data1.db
Done.
```

Out[30]:

Booster_Version	Payload	PAYLOAD_MASS__KG_
F9 B5 B1048.4	Starlink 1 v1.0, SpaceX CRS-19	15600
F9 B5 B1049.4	Starlink 2 v1.0, Crew Dragon in-flight abort test	15600
F9 B5 B1051.3	Starlink 3 v1.0, Starlink 4 v1.0	15600
F9 B5 B1056.4	Starlink 4 v1.0, SpaceX CRS-20	15600
F9 B5 B1048.5	Starlink 5 v1.0, Starlink 6 v1.0	15600
F9 B5 B1051.4	Starlink 6 v1.0, Crew Dragon Demo-2	15600
F9 B5 B1049.5	Starlink 7 v1.0, Starlink 8 v1.0	15600
F9 B5 B1060.2	Starlink 11 v1.0, Starlink 12 v1.0	15600
F9 B5 B1058.3	Starlink 12 v1.0, Starlink 13 v1.0	15600
F9 B5 B1051.6	Starlink 13 v1.0, Starlink 14 v1.0	15600
F9 B5 B1060.3	Starlink 14 v1.0, GPS III-04	15600
F9 B5 B1049.7	Starlink 15 v1.0, SpaceX CRS-21	15600

Task 9

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [68]:
```
%sql SELECT substr(Date,7,4), substr(Date, 4, 2),"Booster_Version", "Launch_Site", Payload, "PAYLOAD_MASS__KG_", "Mission_O
```

```
* sqlite:///my_data1.db
Done.
```

Out[68]:

substr(Date,7,4)	substr(Date, 4, 2)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Mission_Outcome	Landing _Outcome
2015	01	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	Success	Failure (drone ship)
2015	04	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	Success	Failure (drone ship)

Task 10

Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

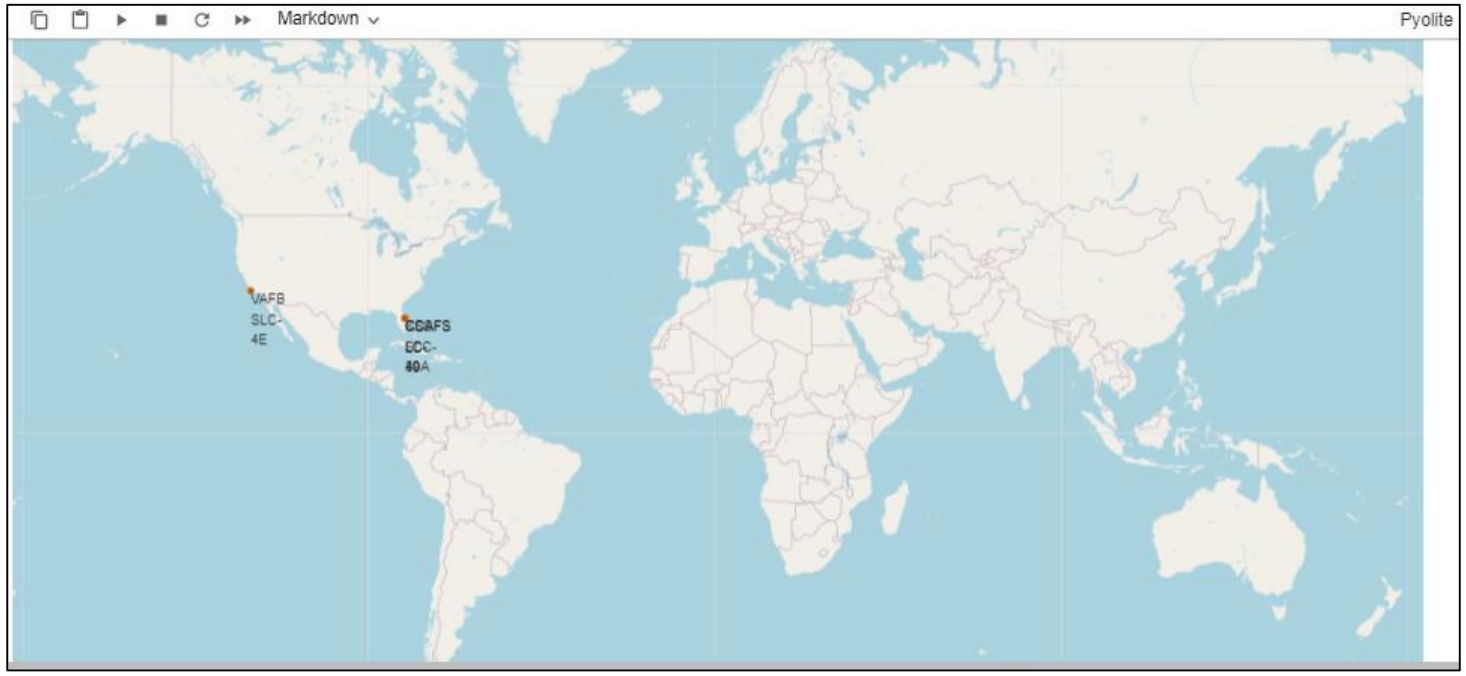
In [74]:
```
%sql SELECT * FROM SPACEXTBL WHERE "Landing _Outcome" LIKE 'Success%' AND (Date BETWEEN '04-06-2010' AND '20-03-2017') ORDE
```

```
* sqlite:///my_data1.db
Done.
```

Out[74]:

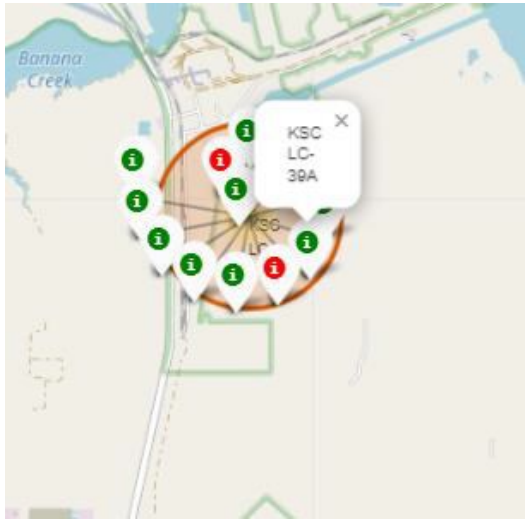
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
19-02-2017	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-10-2020	12:25:57	F9 B5 B1051.6	KSC LC-39A	Starlink 13 v1.0, Starlink 14 v1.0	15600	LEO	SpaceX	Success	Success
18-08-2020	14:31:00	F9 B5 B1049.6	CCAFS SLC-40	Starlink 10 v1.0, SkySat-19, -20, -21, SAOCOM 1B	15440	LEO	SpaceX, Planet Labs, PlanetIQ	Success	Success
18-07-2016	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
18-04-2018	22:51:00	F9 B4 B1045.1	CCAFS SLC-40	Transiting Exoplanet Survey Satellite (TESS)	362	HEO	NASA (LSP)	Success	Success (drone ship)
17-12-2019	00:10:00	F9 B5 B1056.3	CCAFS SLC-40	JCSat-18 / Kacific 1, Starlink 2 v1.0	6956	GTO	Sky Perfect JSAT, Kacific 1	Success	Success

Results

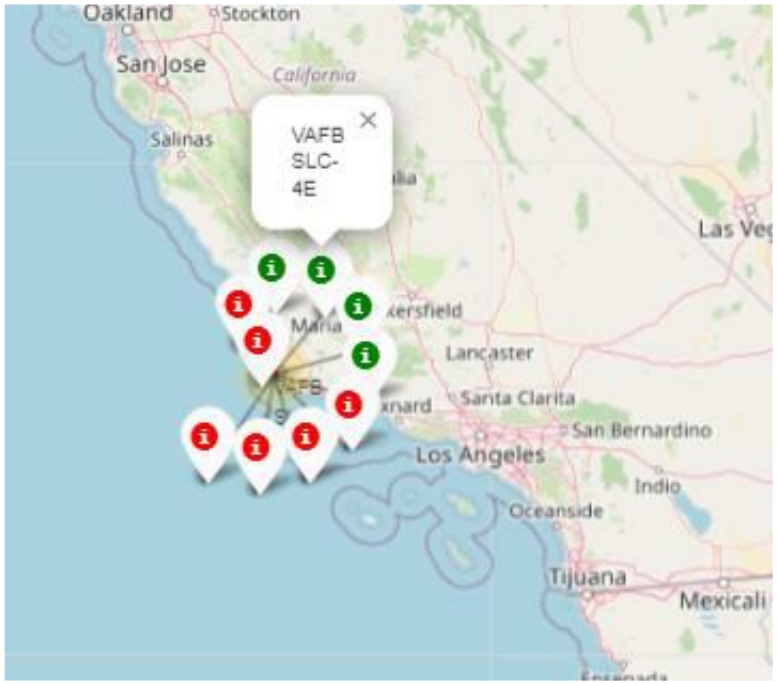

Markdown
Pyolite
VAFB
SLC-
4E
CCAFS
LC-
40A


CCAFS
LC-40


CCAFS
SLC-40
26
LC-
40

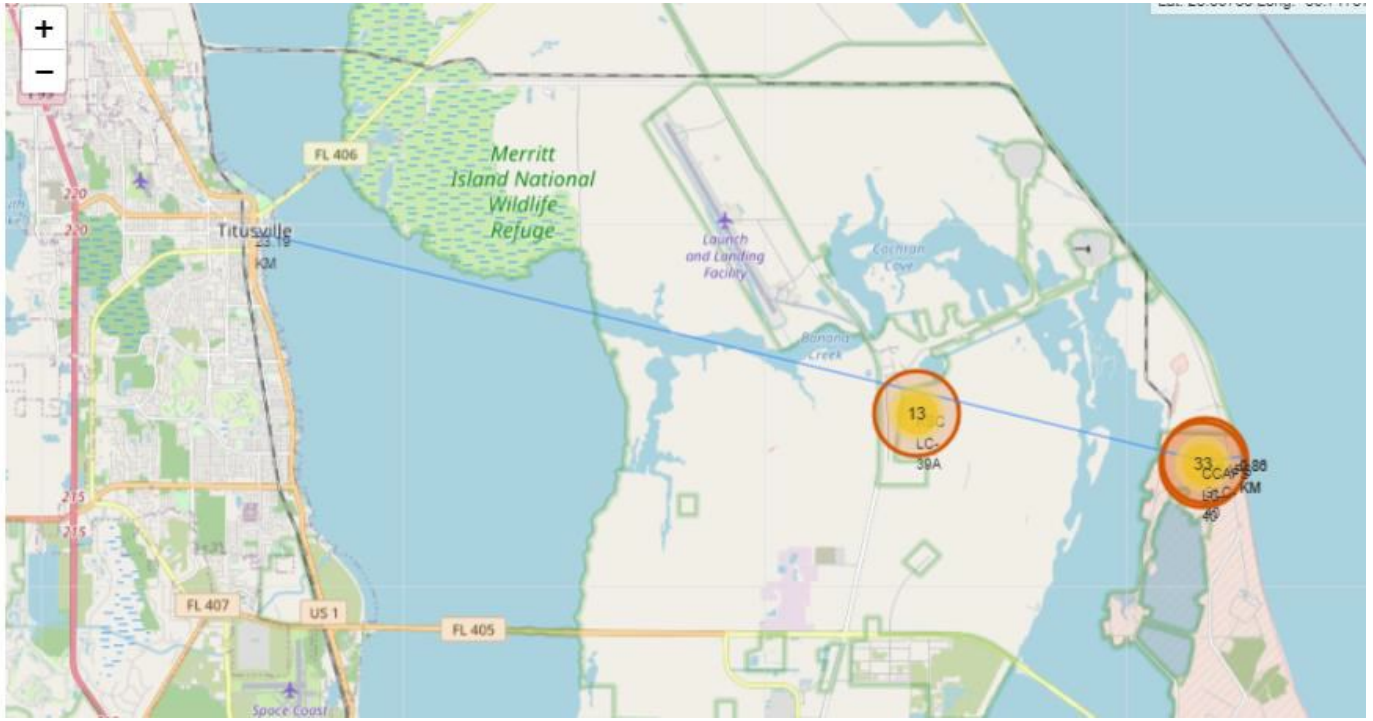

Banana
Creek
KSC
LC-
39A

Results

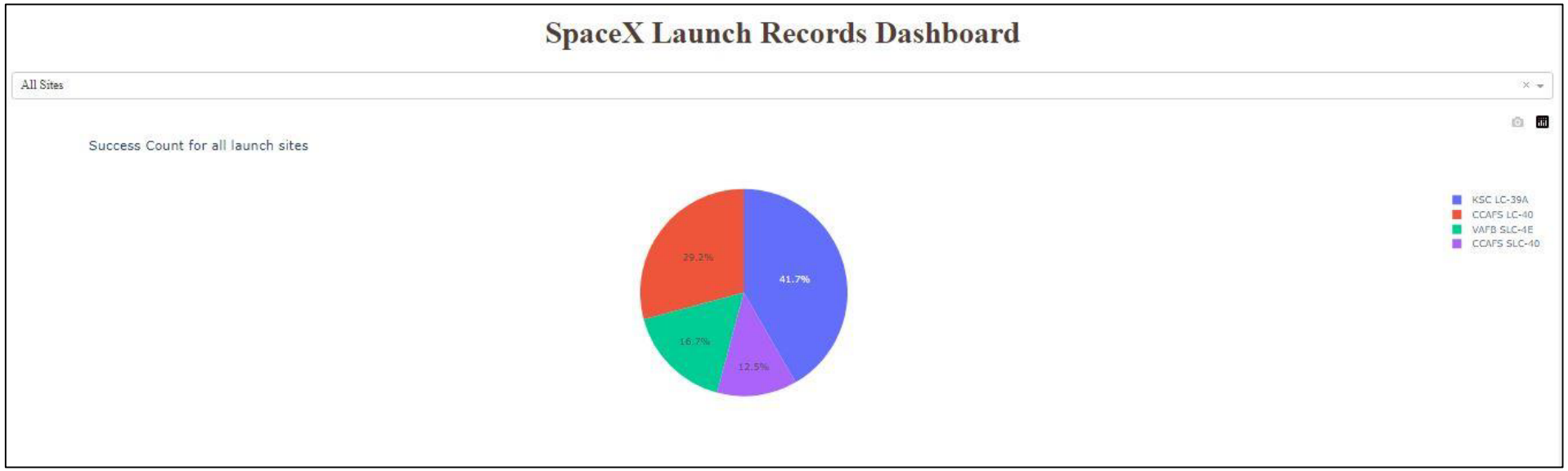

Oakland
Stockton
San Jose
California
Salinas
VAFB
SLC-
4E
Las Veg
Maria
Lancaster
Santa Clarita
San Bernardino
Los Angeles
Indio
Oceanside
Tijuana
Mexicali

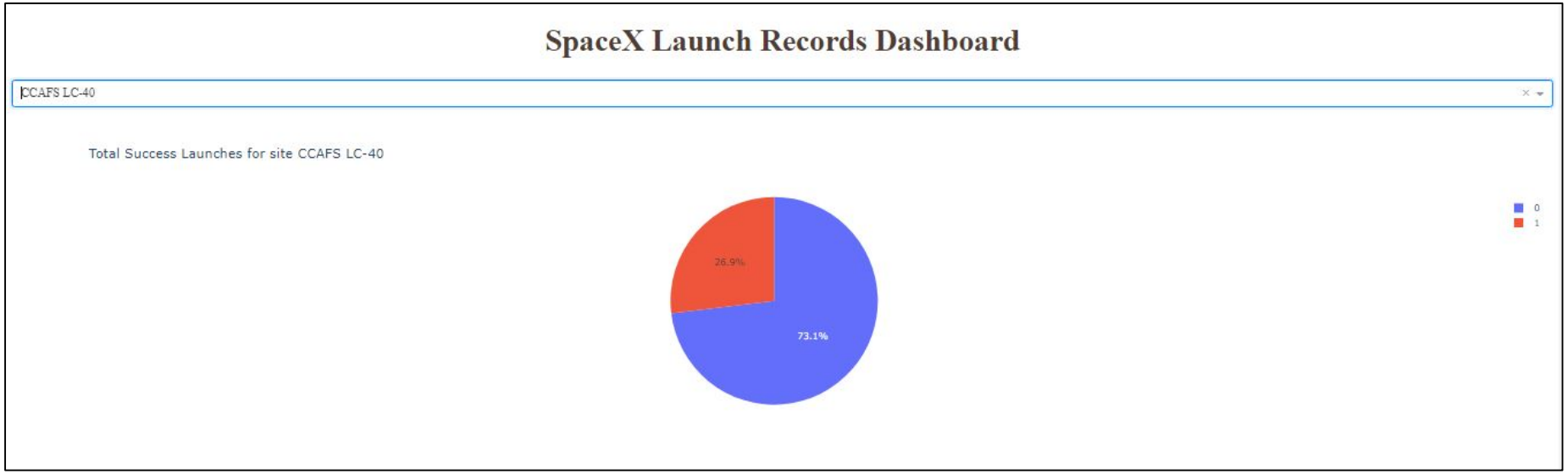

P
P
P
P
7
SLC-
40
26
LC-
40
0.88
KM
Centaur Road
Samuel C Phillips Parkwa

[21]:

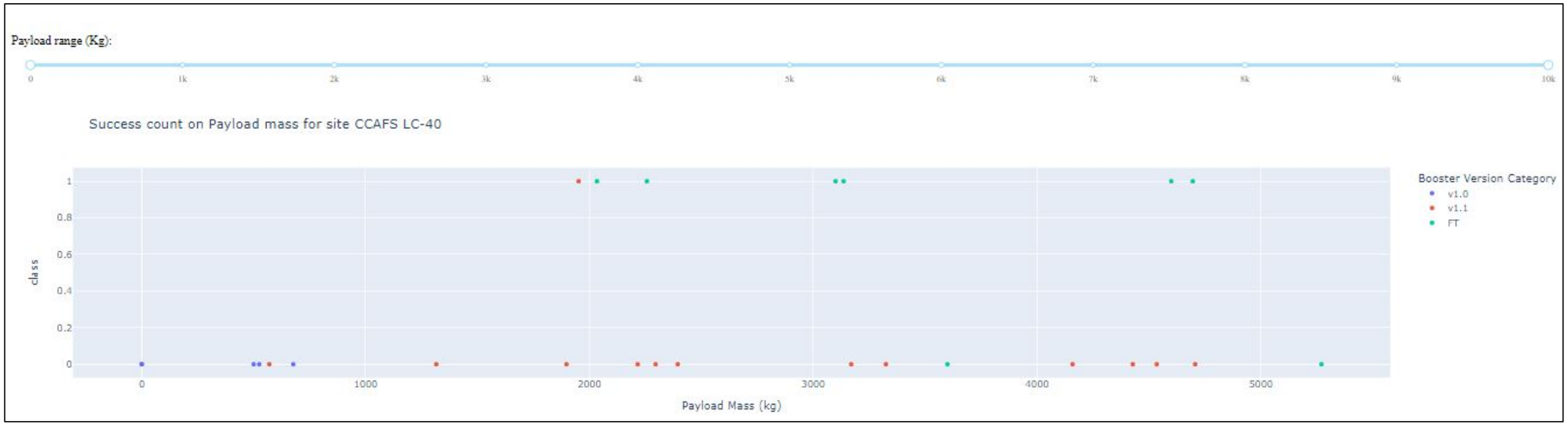

Merritt
Island National
Wildlife
Refuge
Titusville
FL 406
FL 407
US 1
FL 405
Launch
and Landing
Facility
Cochran
Cove
Banana
Creek
13
LC-
39A
33
0.88
KM
Space Coast

Results


SpaceX Launch Records Dashboard
All Sites
Success Count for all launch sites
41.7%
29.2%
16.7%
12.5%
KSC LC-39A
CCAFS LC-40
VAFB SLC-4E
CCAFS SLC-40


SpaceX Launch Records Dashboard
CCAFS LC-40
Total Success Launches for site CCAFS LC-40
26.9%
73.1%
0
1

Results


Payload range (Kg):
0
1k
2k
3k
4k
5k
6k
7k
8k
9k
10k
Success count on Payload mass for site CCAFS LC-40
class
1
0.8
0.6
0.4
0.2
0
0
1000
2000
3000
4000
5000
Payload Mass (kg)
Booster Version Category
v1.0
v1.1
FT

Results

TASK 12

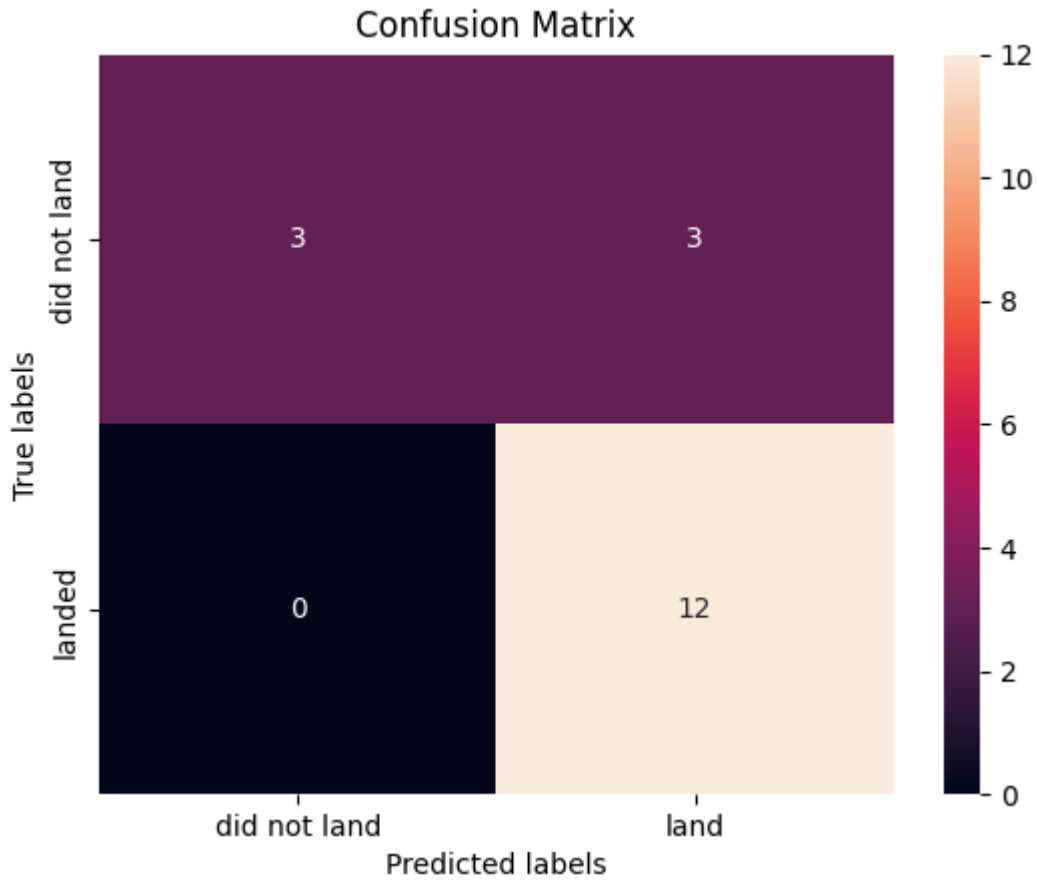
Find the method performs best:

In [68]:

```
Report = pd.DataFrame({'Method' : ['Test Data Accuracy']})

knn_accuracy=knn_cv.score(X_test, Y_test)
Decision_tree_accuracy=tree_cv.score(X_test, Y_test)
SVM_accuracy=svm_cv.score(X_test, Y_test)
Logistic_Regression=logreg_cv.score(X_test, Y_test)

Report['Logistic_Reg'] = [Logistic_Regression]
Report['SVM'] = [SVM_accuracy]
Report['Decision Tree'] = [Decision_tree_accuracy]
Report['KNN'] = [knn_accuracy]



Report.transpose()
```

Out[68]:

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.833333
KNN	0.833333



Conclusions

- Different launch success rates per site
  - CCAFS LC-40 = 60%
  - KSC LC-39A and VAFB SLC 4E = 77%
- Success rate increased with number of flights
  -VAFB SLC 4E = 100% after 50^{th}
  - KSC LC 39A = 100% after 80^{th}
- VAFB-SLC → no rockets launched for heavy payload mass (>10000 kg)
- Orbits ES-L1, GEO, HEO & SSO = 100%, SO nearly 50% and SO has 0% success rate
- Success rate rising from 2013 until 2020

Thank you

https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches