Winning Space Race with Data Science

Mark Smith

Jan 2022

Outline

Executive Summary

Summary of methodologies

- Data Collection
- Data Wrangling
- Exploratory Data Analysis
  - EDA with SQL
  - EDA With Visualisation
- Interactive Visual Analytics and Dashboards
  - Building an interactive map with Folium
  - Building a Dashboard with Plotly Dash
- Predictive Analysis (Classification)

Summary of all results

- EDA results
- Analytics Results
- Predictive Analysis results

Introduction

Project background and context

The commercial space age is here, companies are making space travel affordable for everyone. Virgin Galactic is providing suborbital spaceflights. Rocket Lab is a small satellite provider. Blue Origin manufactures sub-orbital and orbital reusable rockets. SpaceX's has sent spacecraft to the International Space Station and sent manned missions to Space. One reason SpaceX can do this is the rocket launches are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

We will use data to predict if the Falcon 9 first stage will land successfully. 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 SpaceX for a rocket launch.

Problems to find answers to:

- What determines if a rocket will land successfully?
- The effect each variable has on a successful rocket landing?
- What are the optimum conditions to achieve a successful landing?

Methodology

Executive Summary

- Data collection methodology:
  - Collecting the Data with an API. Specifically, the SpaceX REST API. This API will give us data about launches, including information about the rocket used, payload delivered, launch specifications, landing specifications, and landing outcome. Our goal is to use this data to predict whether SpaceX will attempt to land a rocket or not.
  - Web Scrapping (from general information pages)
- Perform data wrangling
  - Transformation of data for machine learning. Removing irrelevant data and utilization of 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
  - Build, tune, evaluate classification models

Section 1

Methodology

Data Collection

Data Collection is the systematic approach of gathering and measuring information on variables to answer questions, evaluate outcomes and test hypothesis.

Methods:

- ***Collecting Data with an API***
- Space X launch data was gathered using the SpaceX REST API; providing data about launches, rocket used, payload delivered, launch specifications, landing specifications, and landing outcome.
- The SpaceX REST API endpoints, or URL, starts with api.spacexdata.com/v4/. We will be working with the endpoint api.spacexdata.com/v4/launches/past. This URL will be used to target a specific endpoint of the API to get past launch data.
- A get request will be performed using the requests library to obtain the launch data, which we will obtain the data from the API. This result can be viewed by calling the .json() method. Our response will be in the form of a JSON, specifically a list of JSON objects.
- To convert this JSON to a dataframe, we use the json_normalize function, normalizing the structured json data into a flat table.
- ***Webscraping***
- Another data source for obtaining Falcon 9 Launch data is web scraping related Wiki pages using the Python BeautifulSoup package to web scrape HTML tables that contain valuable Falcon 9 launch records. Data is then parsed from those tables and converted into a Pandas data frame for further visualization and analysis.

Data Collection – SpaceX API

Link to Notebook

Using a get request to obtain the data from the API. Result can be viewed by calling the .json() method

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

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

Convert this JSON to a dataframe, we use the json_normalize function, normalizing the structured json data into a flat table

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

Clean Data : example

```
getLaunchSite(data)
```

Construct dataset using data obtained. Combine the columns into a dictionary. Create dataframe from dictionary

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

```
launch_dataframe = pd.DataFrame.from_dict(launch_dict)
```

Convert to dataframe, filter dataset and produce flat file.

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1
...	...	...	...	...	...	...	...	...

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection - Scraping

- Perform HTTP GET method to request HTML Page
- Create Beautiful Soup object from HTML response
- Extract all column/variable names from HTML table header
- Create empty dictionary with keys to be converted to dataframe
- Create dataframe
- Export to flat file

```
data  = requests.get(static_url).text
```

```
soup = BeautifulSoup(data,"html.parser")
```

```
html_tables = soup.find_all("table")
column_names = []

temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with each value to be an empty list
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

```
df = pd.DataFrame.from_dict(launch_dict)
```

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	<generator object Tag._all_strings at 0x7f5569...	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	<generator object Tag._all_strings at 0x7f5569...	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	<generator object Tag._all_strings at 0x7f5569...	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	<generator object Tag._all_strings at 0x7f5568...	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	<generator object Tag._all_strings at 0x7f5568...	Success\n	F9 v1.0B0007.1	No attempt\n	1 March 2013	15:10

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Github link

Data Wrangling – Definition & Steps

Definition:

Data Wrangling refers to a variety of processes designed to transform and clean the raw data into more readily format for access and analysis. Commonly referred steps are Discovery, Structuring, Cleaning, Enriching and Validating.

Introduction:

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.

Outcomes will mainly be converted into Training Labels with 1 meaning the booster successfully landed, 0 means it was unsuccessful

Data Wrangling - Perform Exploratory Data Analysis on Dataset

Calculate the number of launches on each site

```
df.LaunchSite.value_counts()
```

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

Calculate the number and occurrence of each orbit

```
# Apply value_counts on Orbit column
df.Orbit.value_counts()
```

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

Calculate the number and occurrence of mission outcome per orbit type

```
landing_outcomes = df.Outcome.value_counts()
landing_outcomes
```

```
True ASDS       41
None None       19
True RTLS       14
False ASDS       6
True Ocean       5
None ASDS        2
False Ocean      2
False RTLS       1
Name: Outcome, dtype: int64
```

Create a landing outcome label from Outcome column

```
landing_class = df['Outcome'].isin(bad_outcomes).eq(False).astype(int)
df['Class']=landing_class
df[['Class']].head(8)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Lo
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80

Determine success rate of each landing

```
df["Class"].mean()
```

```
0.6666666666666666
```

Export File

```
df.to_csv("dataset_part_2.csv", index=False)
```

[Github Link]

EDA with Data Visualization

Scatter graphs – a visual graph of plotted points that shows the relationship between data sets. The relation between the variables and potential patterns.

- Pay Load Mass (kg) vs Flight Number
- Launch Site versus Flight Number
- Launch Site versus Pay load Mass (kg)
- Orbit versus Flight Number
- Orbit versus PayLoadMass

Bar graph - an easy method to compare data between different groups. Which orbits have the highest probability of success

- Success Rate versus Orbit

Line Graph – used to show trends and to aid predictions.

- Space X rocket success rate versus Year

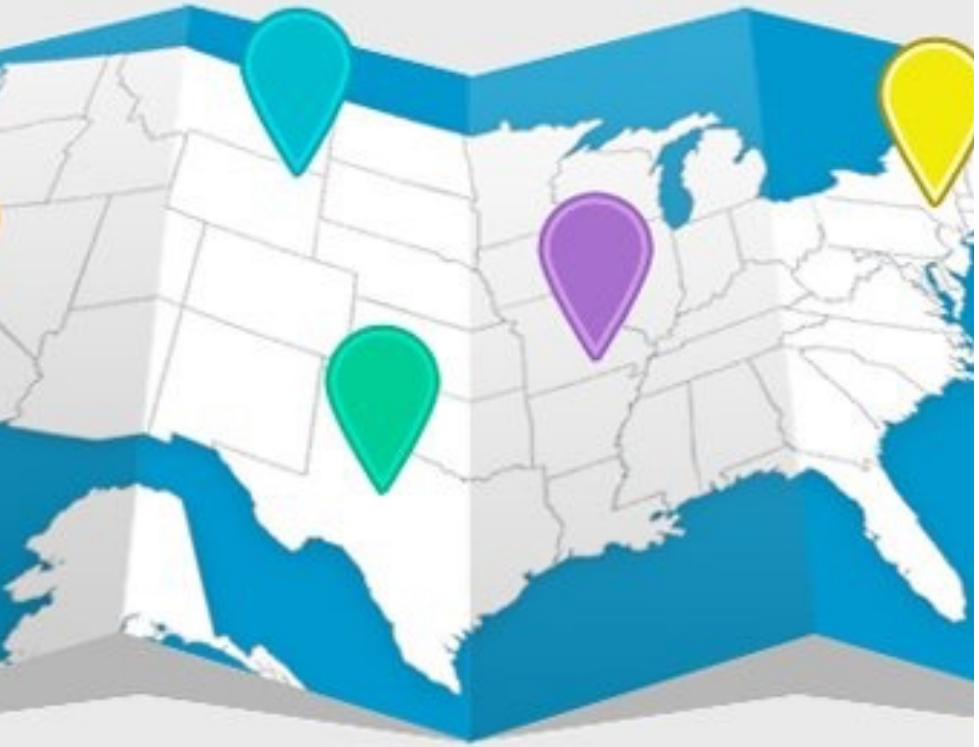
Github link

EDA with SQL

SQL is a widely used programming language for working with databases, an essential tool for performing various operations on databases such as viewing, updating, deleting, creating and modifying data.

- In this project the following questions will be answered by using SQL queries on the dataset.
  - ***Displaying the names of the unique launch sites in the space mission***
  - ***Displaying 5 records where launch sites begin with the string 'CCA'***
  - ***Display the total payload mass carried by boosters launched by NASA (CRS)***
  - ***Displaying average payload mass carried by booster version F9 v1.1***
  - ***Listing the date when the first successful landing outcome in ground pad was achieved.***
  - ***Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000***
  - ***Listing the total number of successful and failure mission outcomes***
  - ***Listing the names of the booster_versions which have carried the maximum payload mass. Use a subquery***
  - ***Listing the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015***
  - ***Ranking 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***

Github Link



Github link

Build an Interactive Map with Folium

- Folium is a Python library that helps create several types of Leaflet maps. Since Folium results are interactive, the library is useful for dashboard building.

- Folium makes it easy to visualize data that's been manipulated in Python on an interactive leaflet map. It enables both the binding of data to a map for choropleth visualizations as well as passing rich vector/raster/HTML visualizations as markers on the map.

- To Visualize this data:

- Longitude and Latitude coordinates are used for each launch site marker.

- Circle Markers with a text label are then used to highlight specific areas.

- Colour coded markers are then used to identify and visualize success and failure for each launch site. Green being successful and Red being unsuccessful

- Codes:
  - Folium.marker – map object to make a marker on a map
  - Folium.circle – create a circle around the marker
  - MarkerCluster – can be a good way to simplify a map containing many markers with the same coordinates.

- Results can be useful in obtaining information such as:
  - Are launch sites located close to cities?
  - Are launch sites located close to major roads or coastlines?
  - Are launch sites positioned away from facilities?

Build a Dashboard with Plotly Dash

Dashboard interactions and plot/graphs:

- Pie Chart showing Launch Successes for all sites or each individual site.
- Scatter graph showing the relationship between Payload and Success for all sites or each individual site
- Interactive drop menu used to select site
- Graphs provide a visual add showing clear relationships between variables.

Using the dashboard as visual analysis insight to the following should be achievable :

- Which site has the largest successful launches?
- Which site has the highest launch success rate?
- Which payload range(s) has the highest launch success rate?
- Which payload range(s) has the lowest launch success rate?
- Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest
- launch success rate?

GitHub Link

Predictive Analysis (Classification)

Building The Model

- Import libraries/Load the data NumPy & Pandas
- Transform the data - standardize
- Split the data into training and testing data sets/set parameter test size
- Check number of test samples
- Identify which models/machine learning to utilize
- Calculate accuracy of test data
- Set parameters using GridSearchCV
- Train dataset

Evaluating and Refining the Model

- Calculate the accuracy on each model
- Get hyperparameters for each model
- Create Confusion Matrix

Finding Best Performing Classification Model

- Find the method that performs best by identifying the model with the best accuracy score.

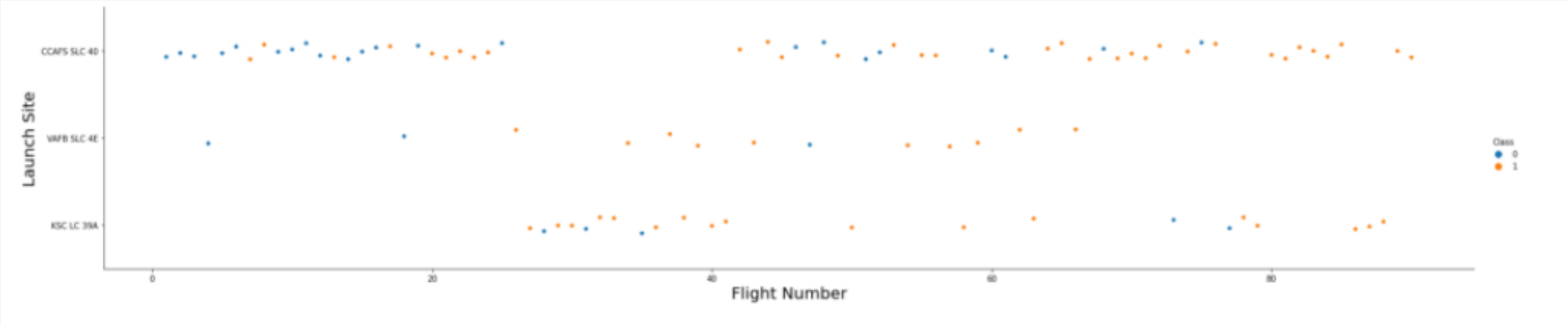
Results

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

Section 2

Insights drawn from EDA

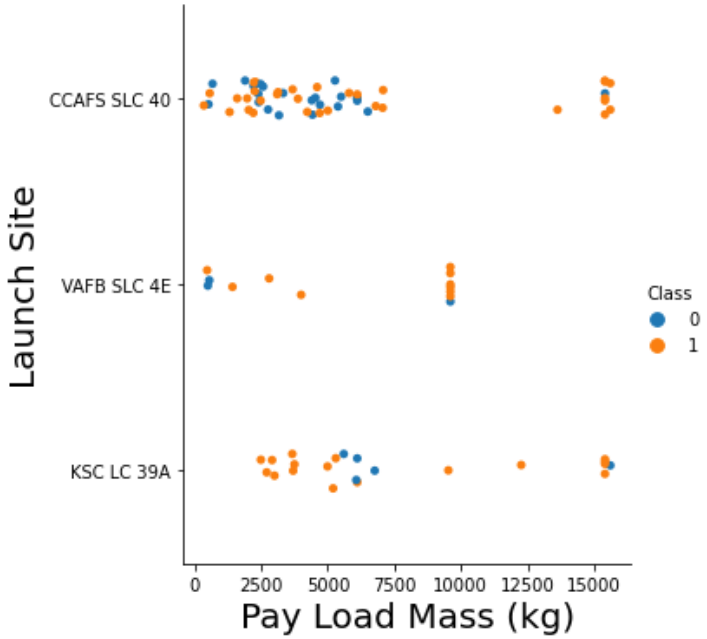
Flight Number vs. Launch Site



As the flight numbers at the launch site increase so does the success rate.

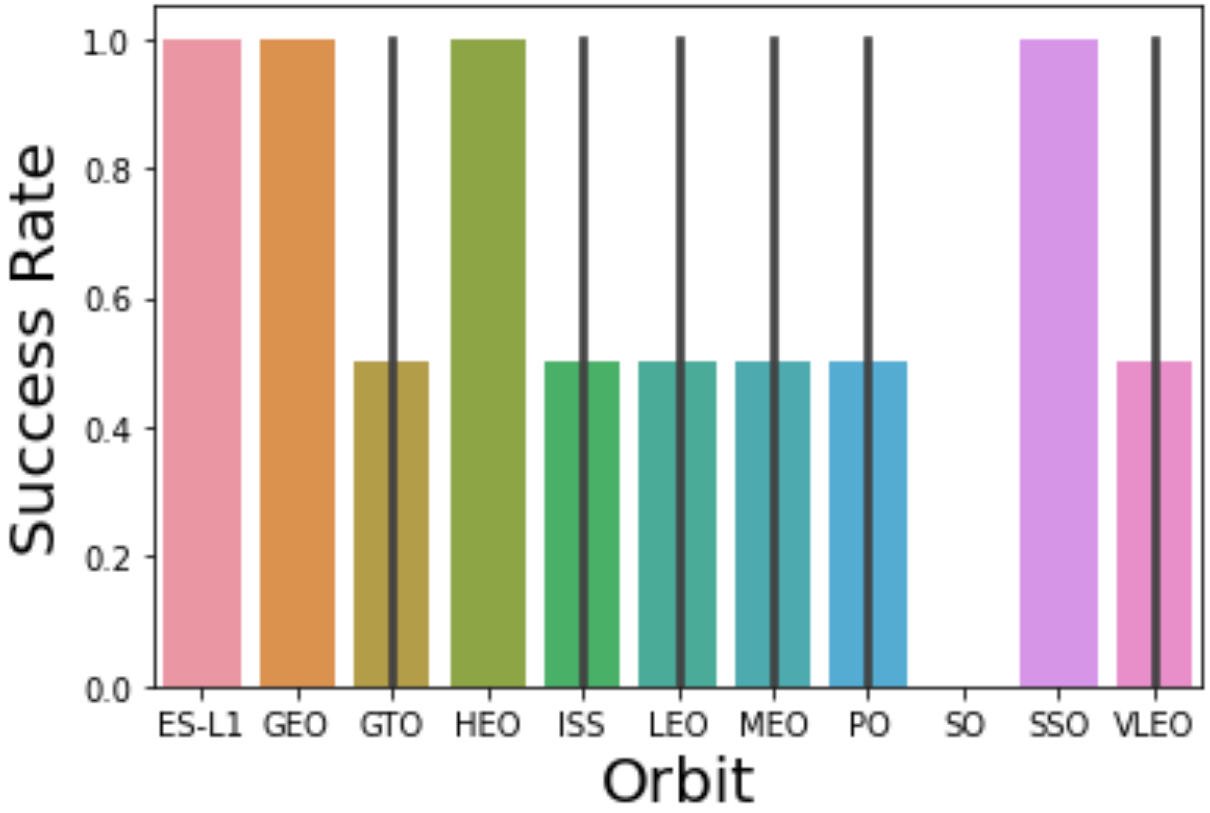
Payload vs. Launch Site

- Pay load mass does not exceed 10,000 for VAFB-SLC
- The greater the pay load mass at CCAFS SLC 40 the greater the higher the success.
- KSC has the most failures at payload 5000.
- No distinct pattern across all sites.



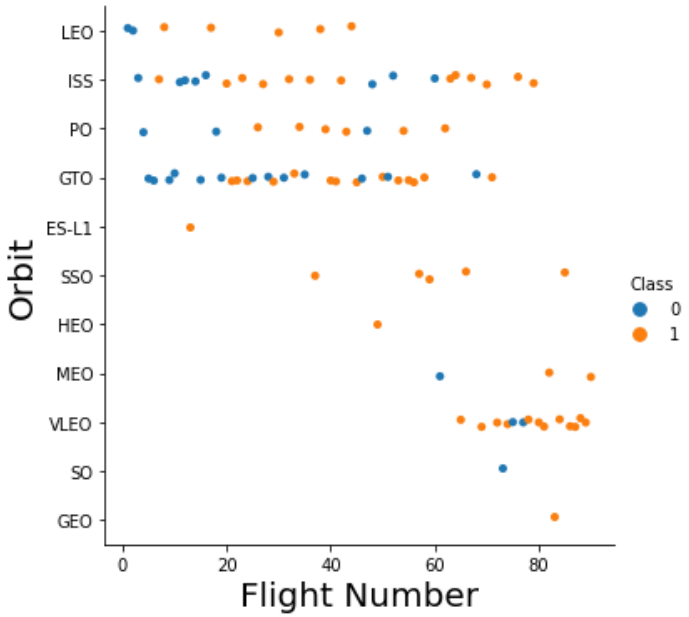
Success Rate vs. Orbit Type

- Orbits ES-L1, GEO, HEO and SSO have the best success rate



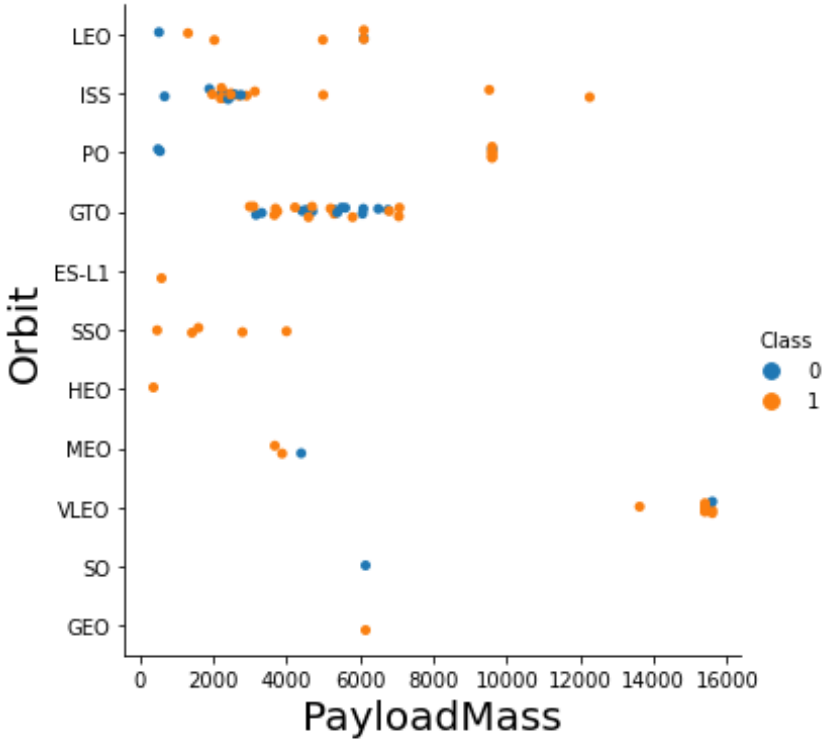
Flight Number vs. Orbit Type

- There is little to no relationship between the variables for the GTO orbit.
- There does appear to be some for the LEO orbit.
- VLEO no early flight numbers



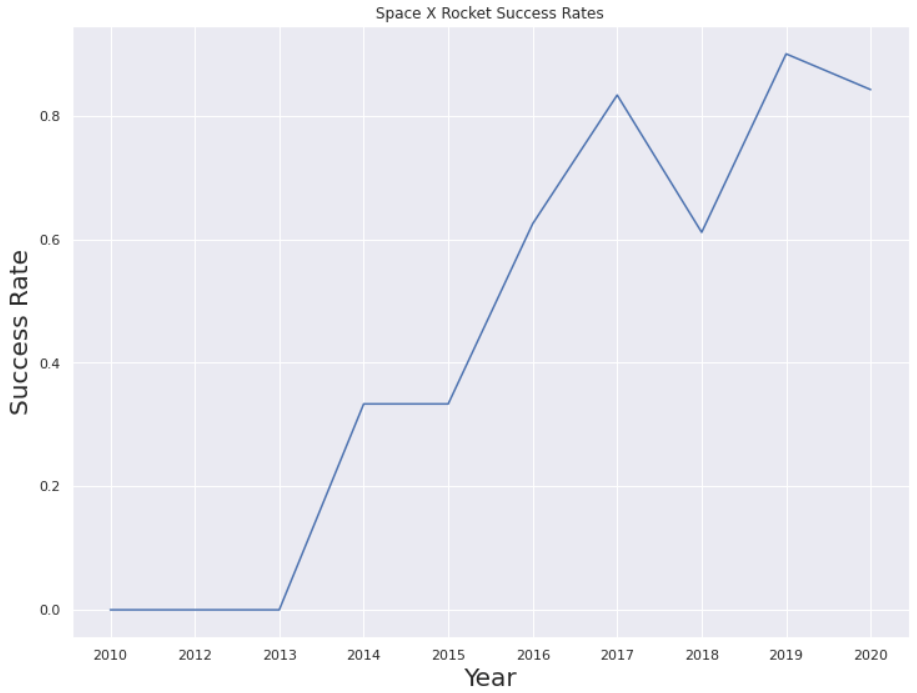
Payload vs. Orbit Type

- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- For GTO we cannot distinguish as both positive landing rate and negative landing(unsuccessful mission) both exist across the spectrum.



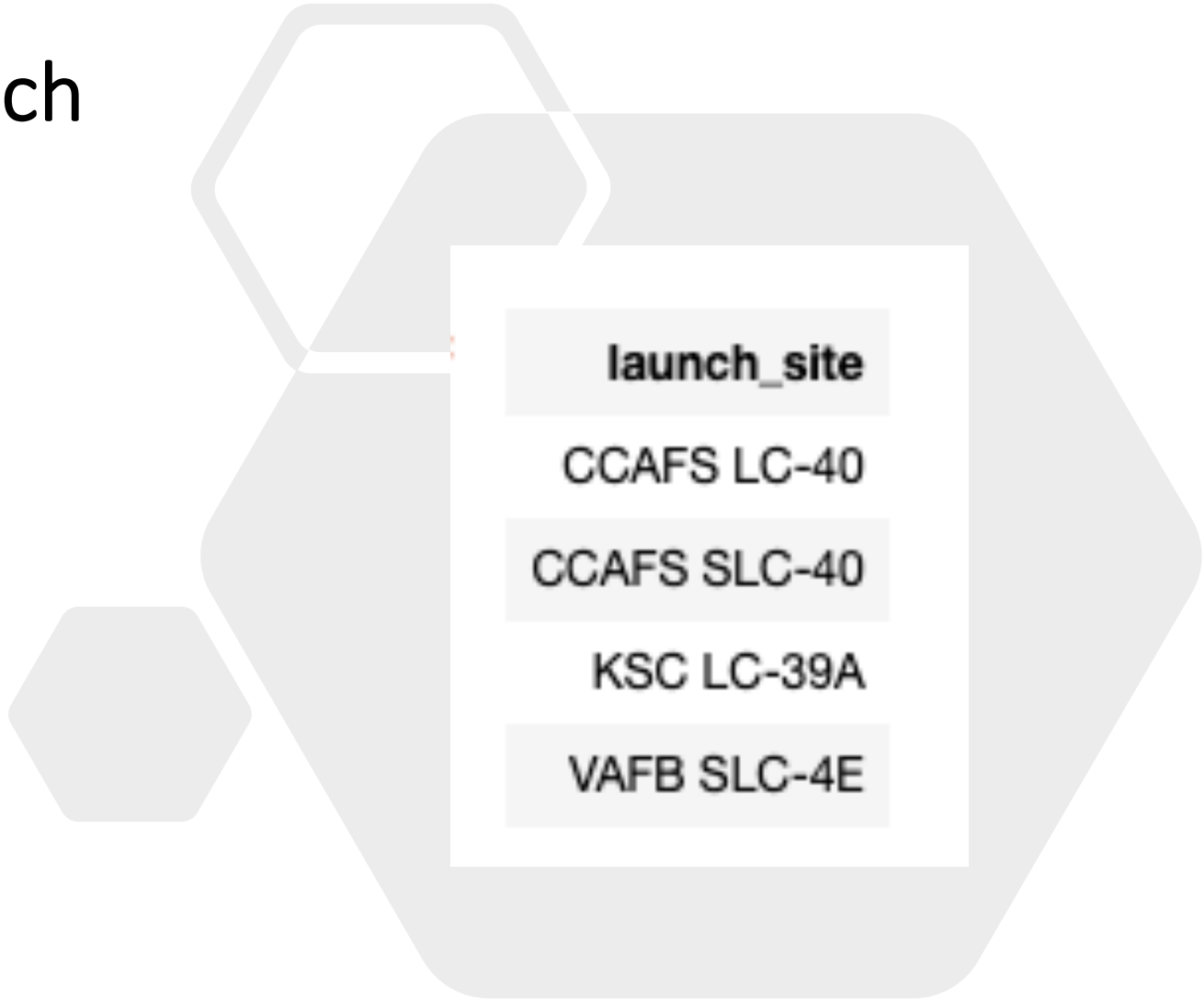
Launch Success Yearly Trend

- The success rate since 2013 has increased until 2019 when it declined.



EDL WITH SQL - All Launch Site Names

- *Task: Display the names of the unique launch sites in the space mission*
- *SQL Query: %sql select distinct LAUNCH_SITE from SPACEXTBL*



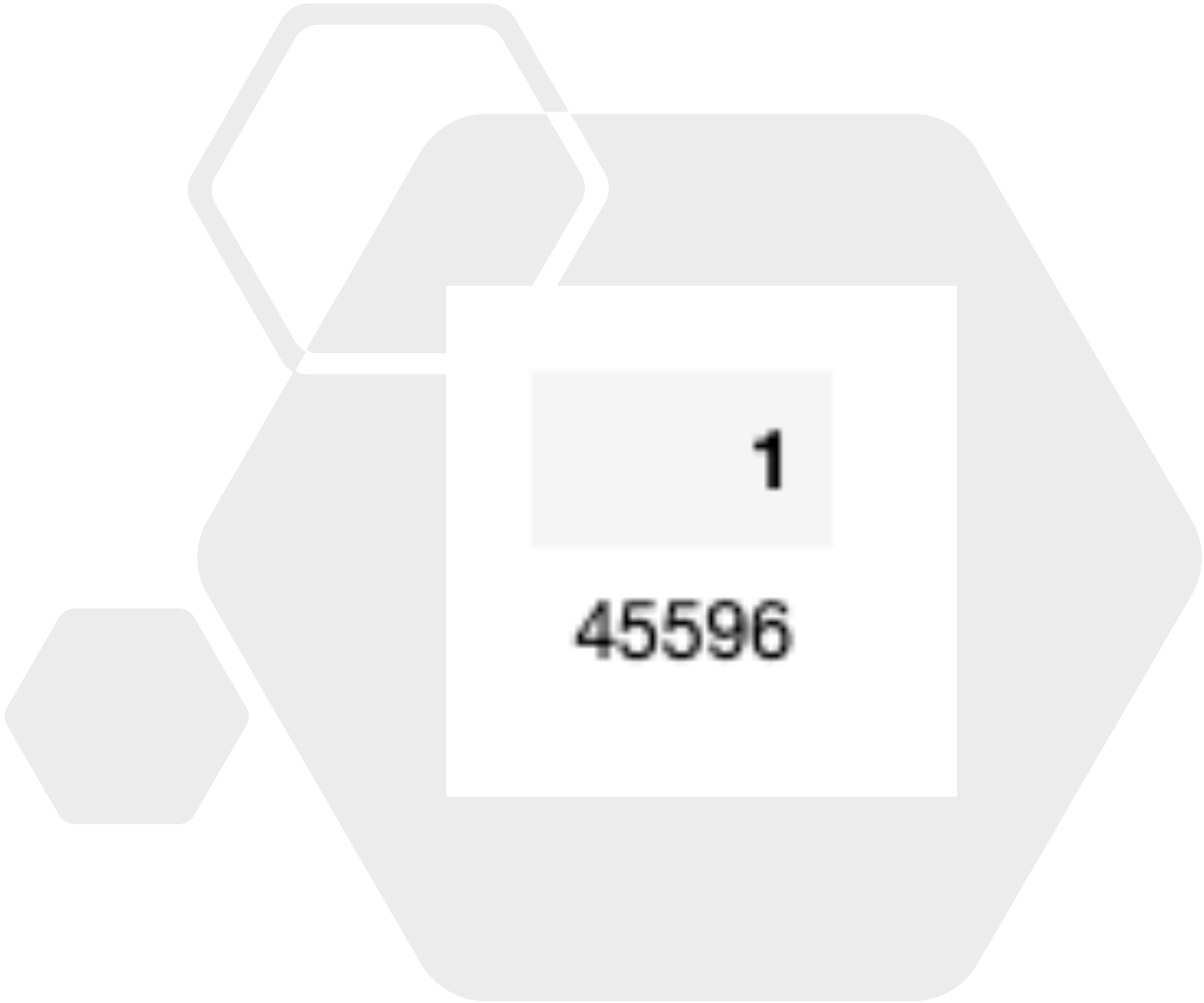
DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	None	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	None	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	None	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	None	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	None	677	LEO (ISS)	NASA (CRS)	Success	No attempt

EDL WITH SQL – Launch Site Names Begin with ‘CCA’

- *Task: Display 5 records where launch sites begin with the string ‘CCA’*
- *SQL Query: %sql select * from SPACEXTBL where LAUNCH_SITE like 'CCA%' fetch first 5 row only*

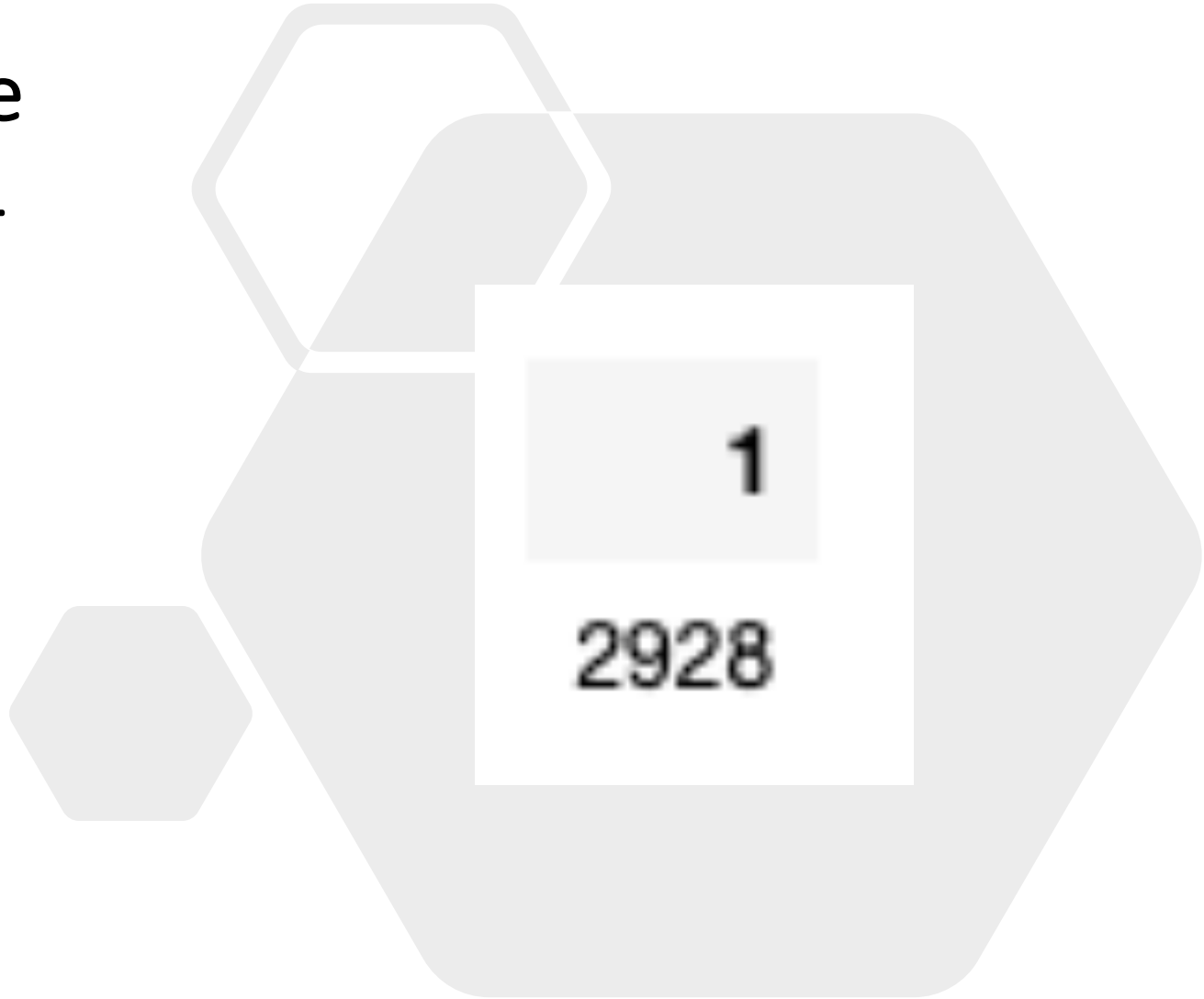
EDL WITH SQL – Total Payload Mass

- *Task: Display the total payload mass carried by boosters launched by NASA*
- *SQL Query: %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTBL where CUSTOMER = 'NASA (CRS)'*



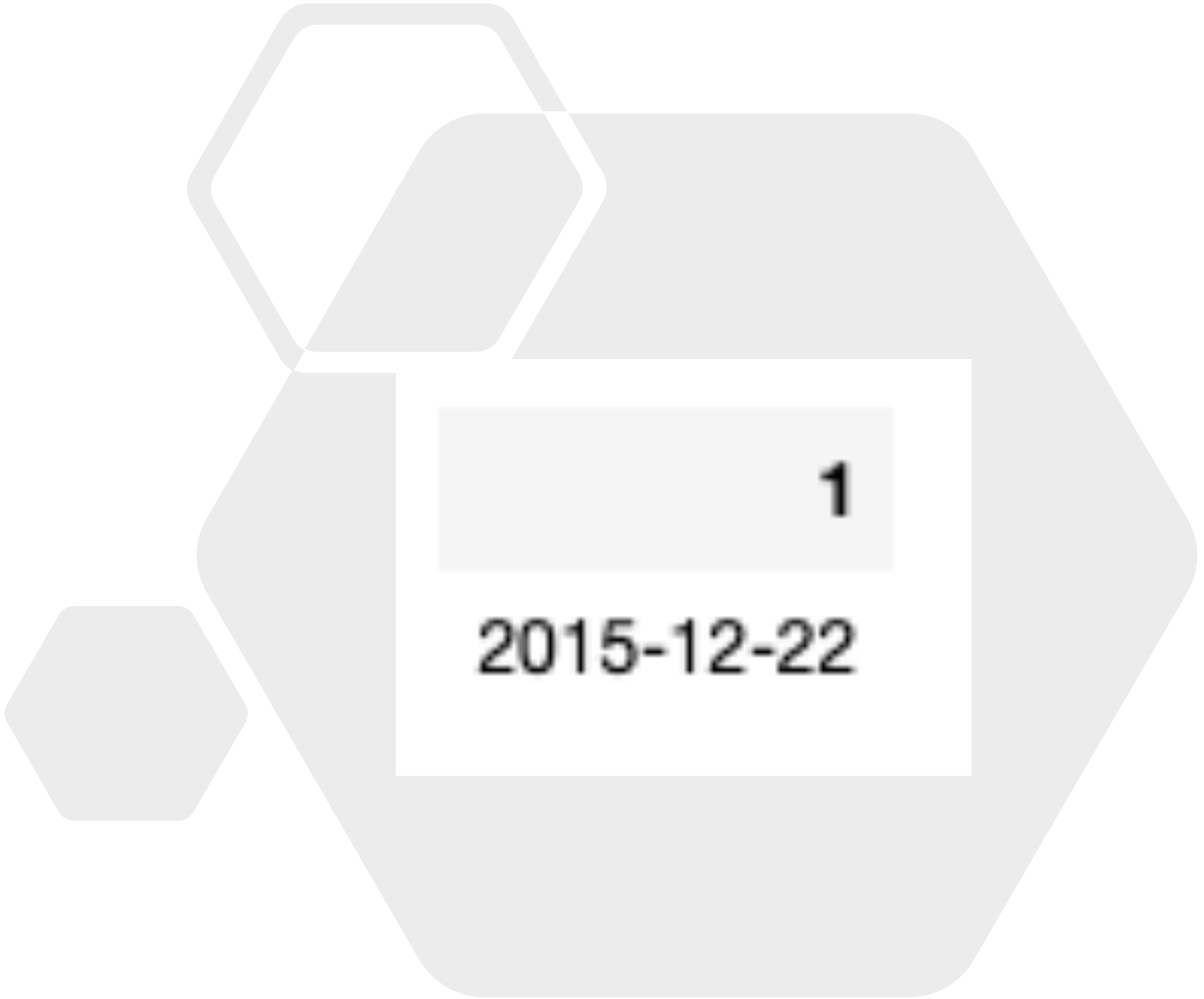
EDL WITH SQL – Average Payload Mass by F9 v1.1

- *Task: Display average payload mass carried by booster version v1.1*
- *SQL Query: %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTBL where BOOSTER_VERSION = 'F9 v1.1'*



EDL WITH SQL – First Successful Ground

- *Task: List the date when the first successful landing outcome in ground pad was achieved.*
- *SQL Query: %sql select min(date) from SPACEXTBL where LANDING__OUTCOME = 'Success (ground pad)'*



EDL WITH SQL – Successful Drone Ship Landing with Payload between 4000 and 6000

- *Task: List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000*
- *SQL Query: %sql select BOOSTER_VERSION from SPACEXTBL where LANDING__OUTCOME ='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000*

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

EDL WITH SQL – Total Number of Successful and Failure Mission Outcomes

- *Task: List the total number of successful and failure mission outcomes*
- *SQL Query: %sql Select MISSION_OUTCOME,count(MISSION_OUTCOME) as count from SPACEXTBL GROUP BY MISSION_OUTCOME*

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

EDL WITH SQL – Boosters Carried Maximum Payload

- *Task: List the names of the booster_versions which have carried the maximum payload mass*
- *SQL Query: %sql select distinct BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_ = (select MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)*

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

EDL WITH SQL – 2015 Launch Records

- *Task: List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in the year 2015*
- *SQL Query: %sql select LANDING__OUTCOME,BOOSTER_VERSION,LAUNCH_SITE from SPACEXTBL where YEAR(DATE) = '2015' and LANDING__OUTCOME = 'Failure (drone ship)'*

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

EDL WITH SQL – Rank Landing Outcome Between 2010-06-04 and 2017-03-20

- *Task: 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*

- *SQL Query:*

```
%%sql

SELECT LANDING__OUTCOME,
COUNT(LANDING__OUTCOME) AS COUNT

FROM SPACEXTBL

WHERE DATE BETWEEN '2010-06-04'  AND '2017-03-20'

GROUP BY LANDING__OUTCOME

ORDER BY COUNT DESC
```

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 4

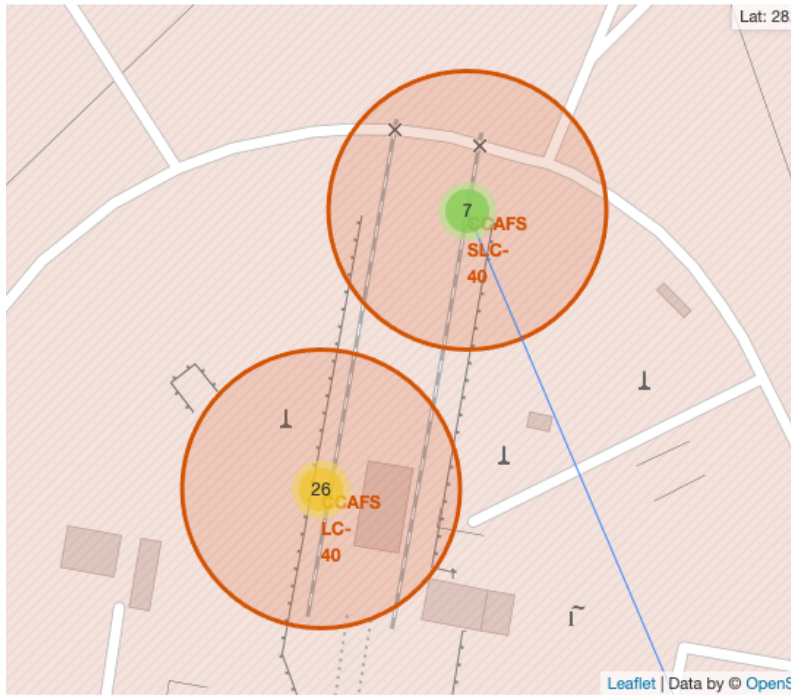
Launch Sites Proximities Analysis

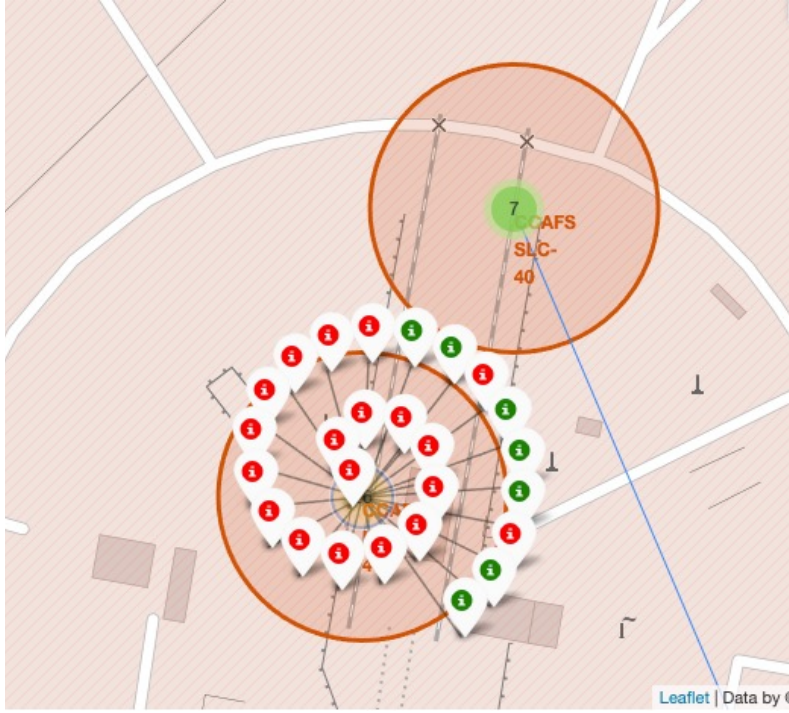
Launch Sites on Folium Global Map

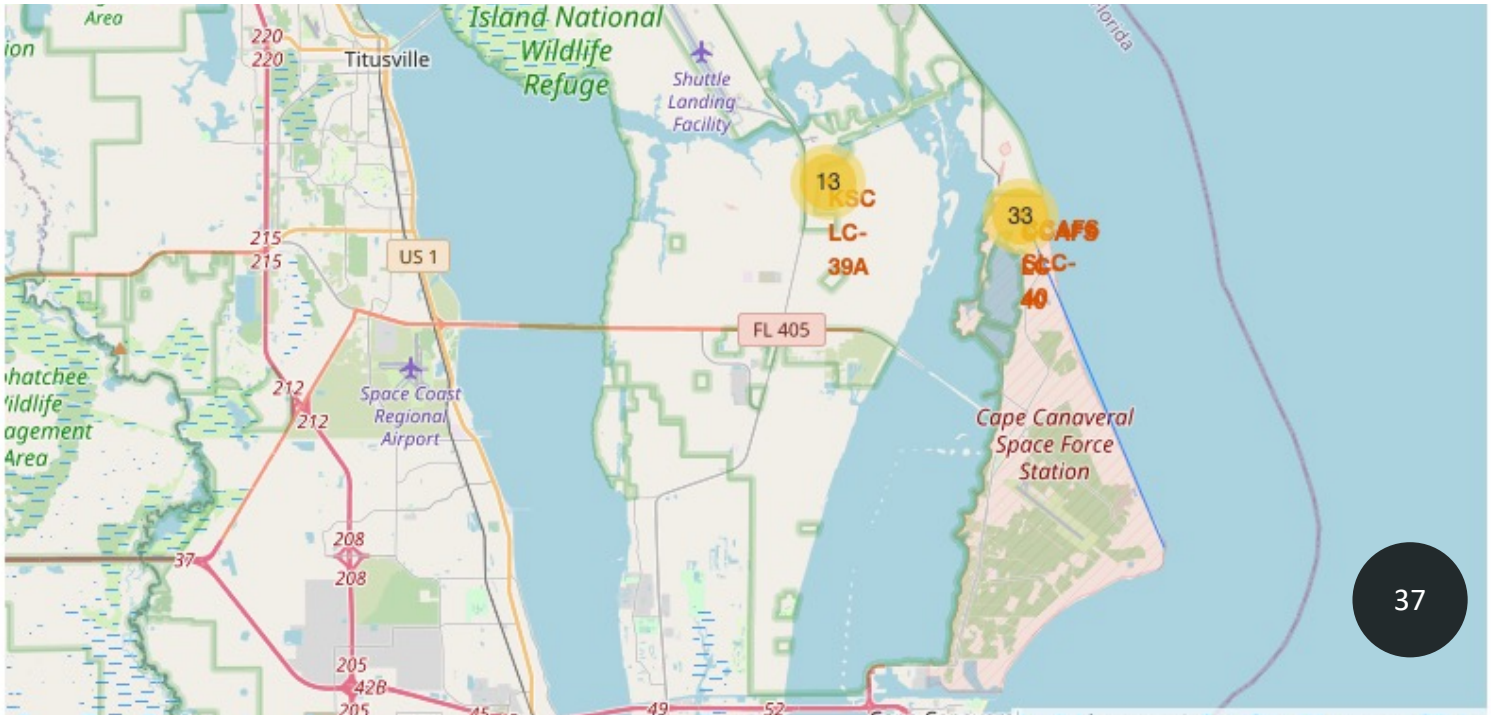
- Launch sites located in the United States of America on the coast.

Colour Labelled Markers

- Florida launch sites
- CCAFS SLC- 40
  - 7 markers
- CCAFS LC-40
  - 26 markers
- Green Marker denotes success launches, whilst red denotes failure.

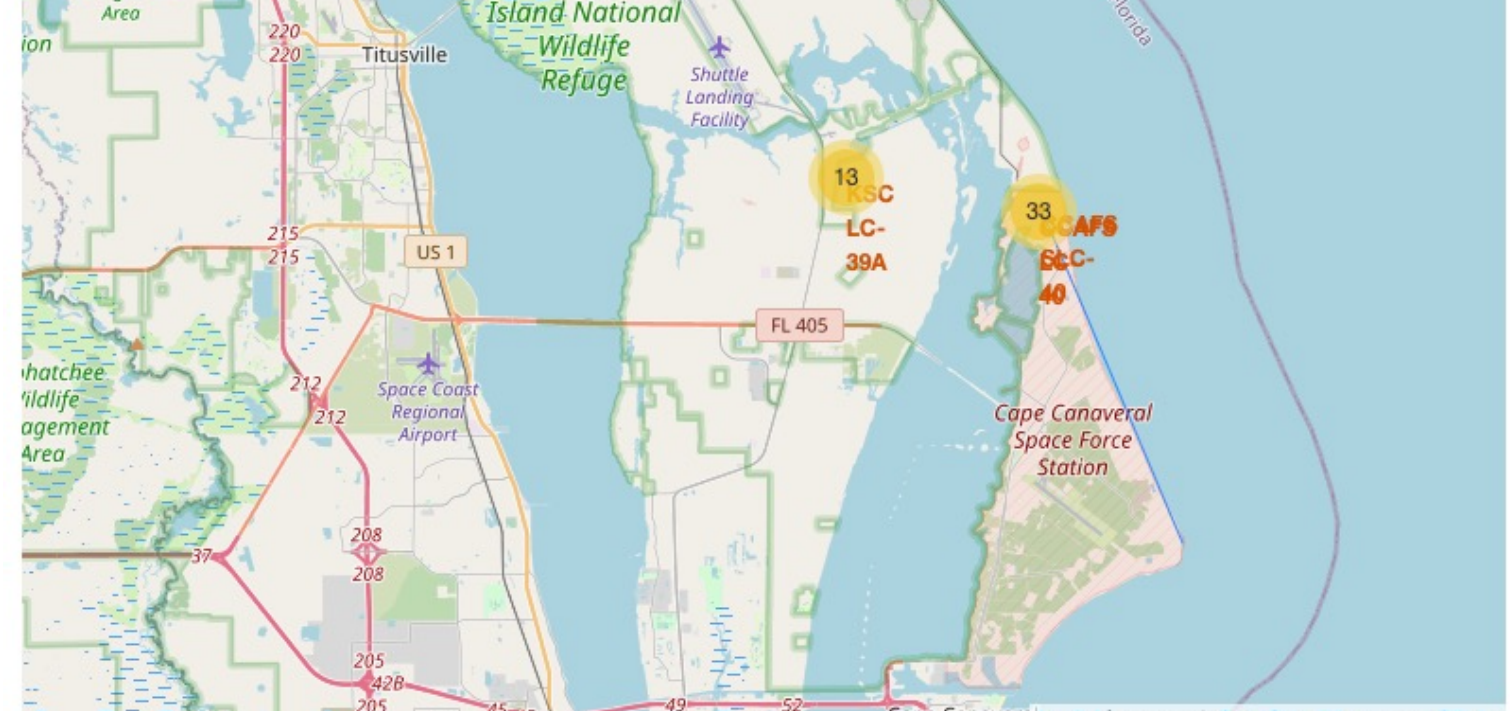


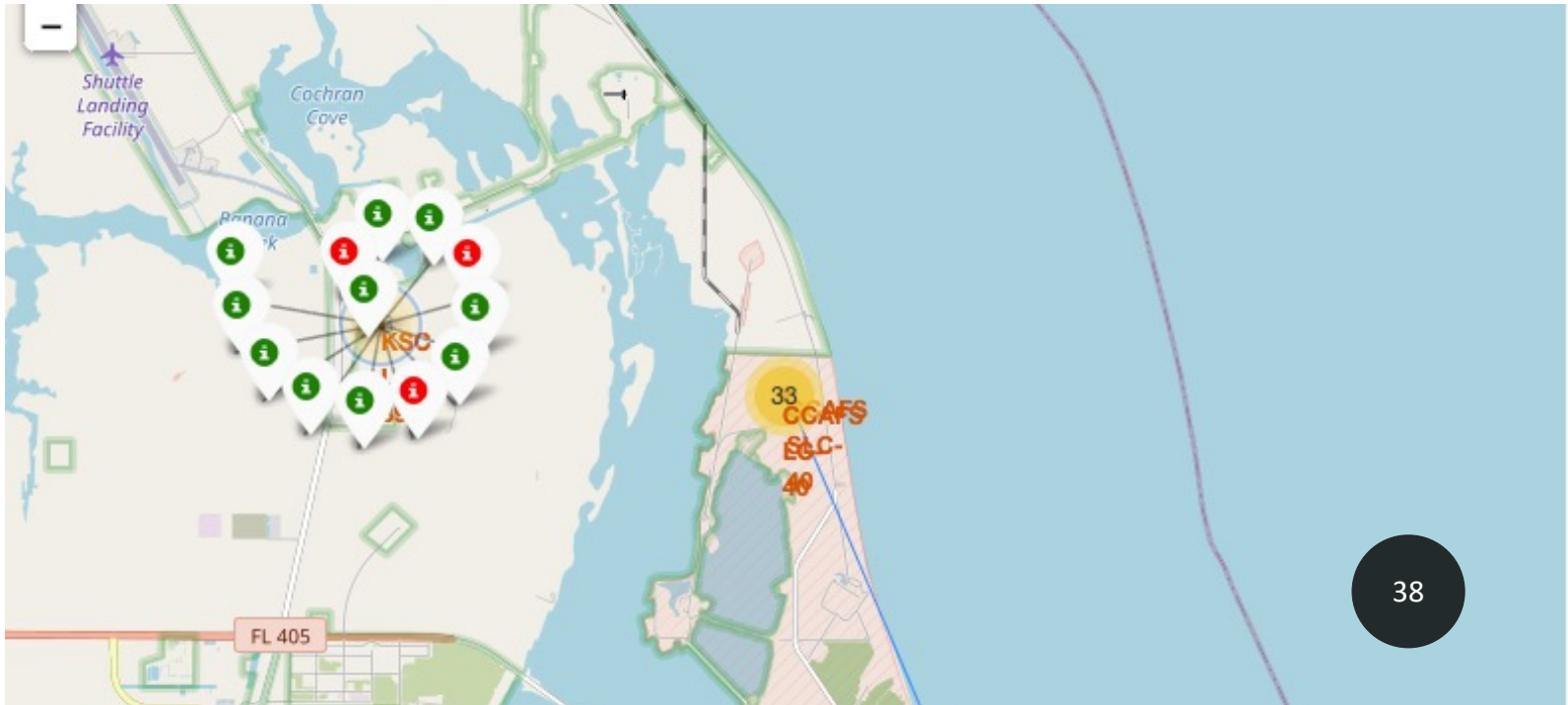


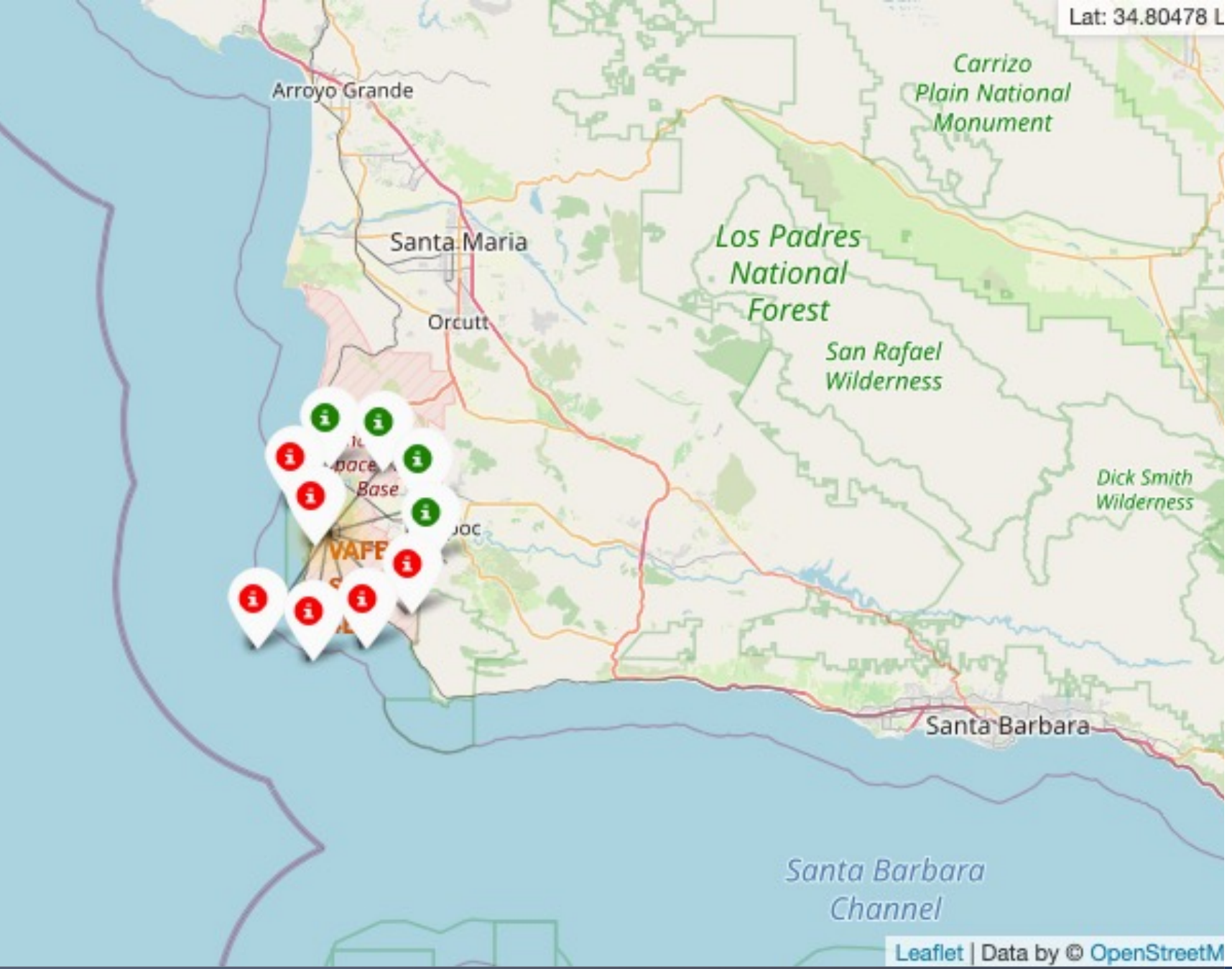


Colour Labelled Markers

- Florida launch sites KSC LC- 39A
  - 13 Markers
  - Green Marker denotes success launches, whilst red denotes failure.

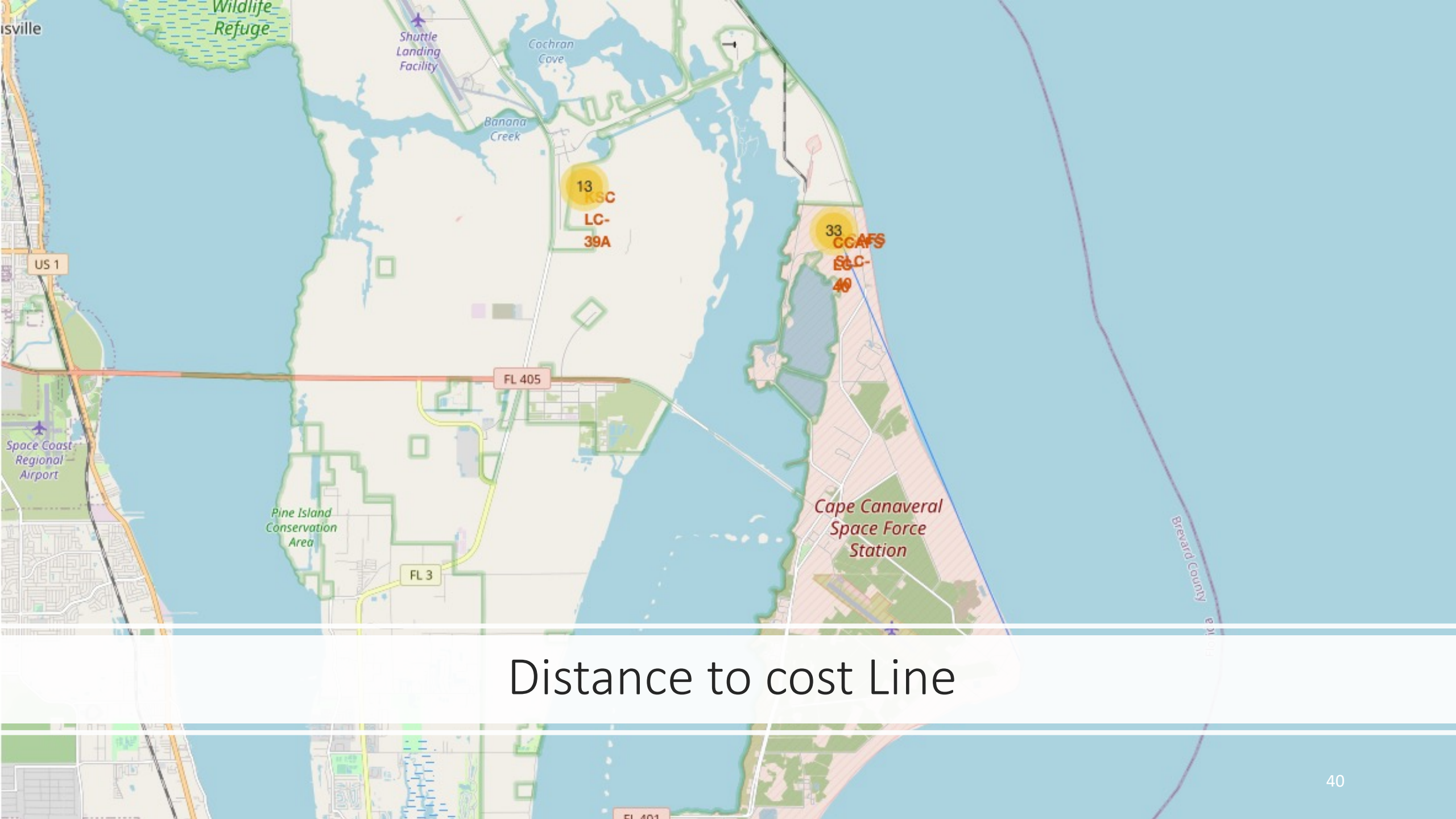




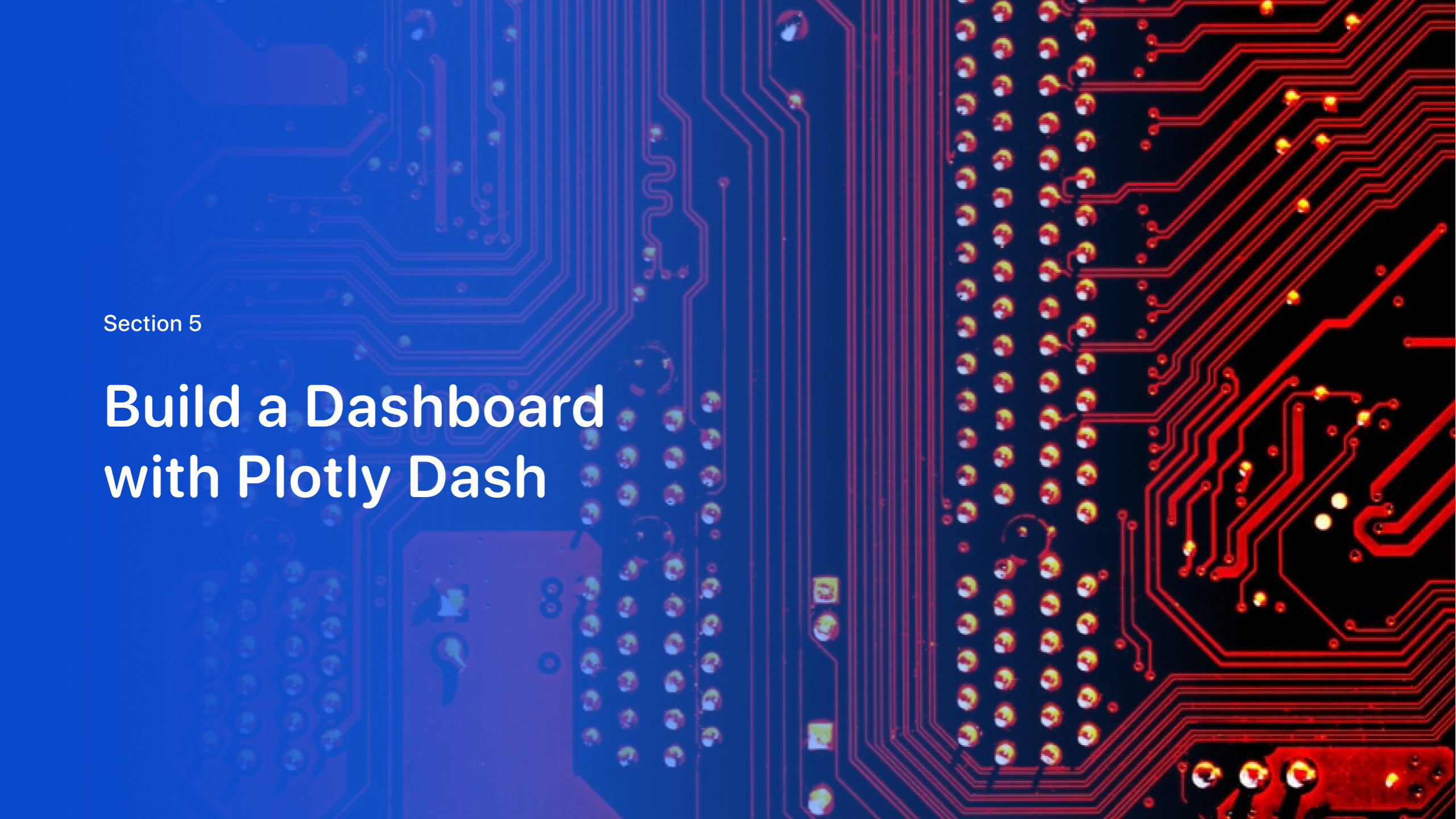


Colour Labelled Markers

- Los Angeles launch site – VAFB SLC – 4E
- Green Marker denotes success launches, whilst red denotes failure.

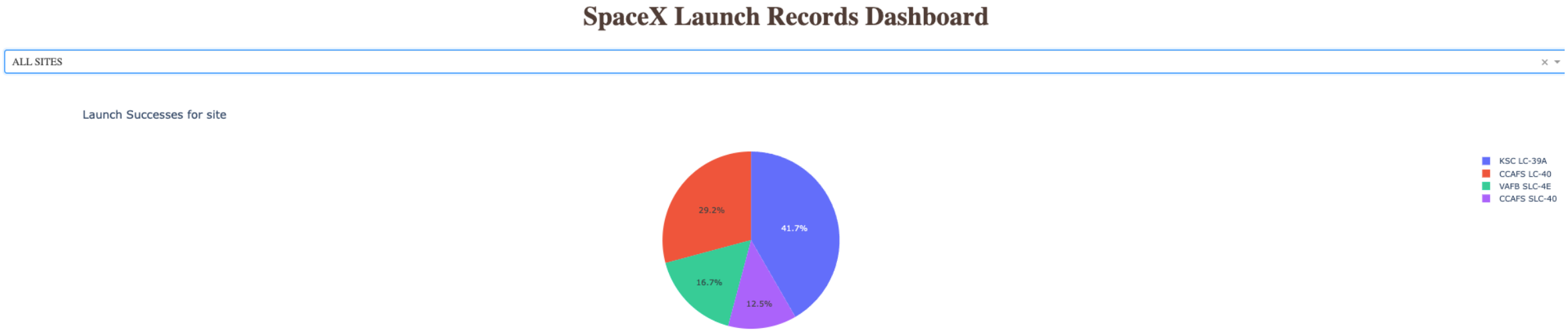


Distance to cost Line



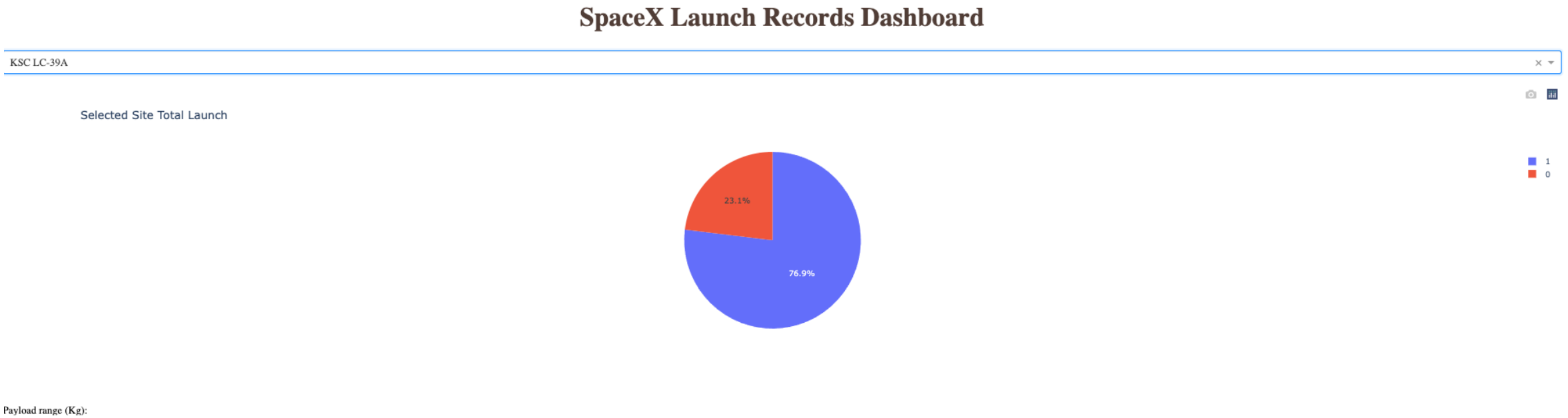
Section 5

Build a Dashboard with Plotly Dash



DASHBOARD – Pie Chart showing the success percentage for each launch site

- The Pie chart shows that KSC LC -39A had the most successful launches from all the sites at 41.7%.



DASHBOARD – Pie Chart showing the launch site with the highest success ratio.

- KSC LC -39A achieved aa success rate of 76.9% and a failure rate of 23.1%

DASHBOARD – Payload vs Launch Outcome Scatter Plot for Sites

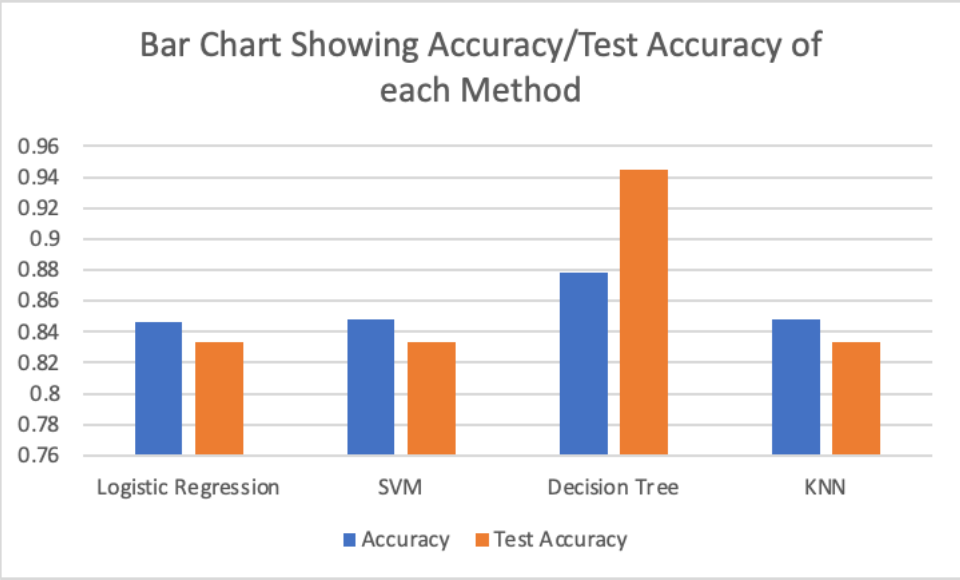
- The outcome being that the success rates for low weighted payloads exceeds that of heavy weighted payloads.

Section 6

Predictive Analysis (Classification)

Classification Accuracy

- The Model with the highest Test Accuracy is the Decision Tree



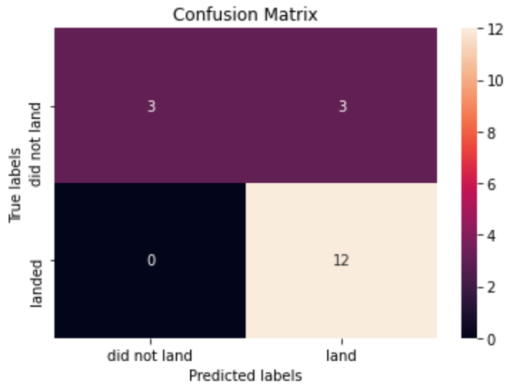
Method	Accuracy	Test Accuracy	Turned Hyperparameters
Logistic Regression	0.846428571	0.833333	{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
SVM	0.848214286	0.833333	{'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
Decision Tree	0.878571429	0.944444	{'criterion': 'entropy', 'max_depth': 16, 'max_features': 'sqrt', 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'random'}
KNN	0.848214286	0.833333	{'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}

Confusion Matrix

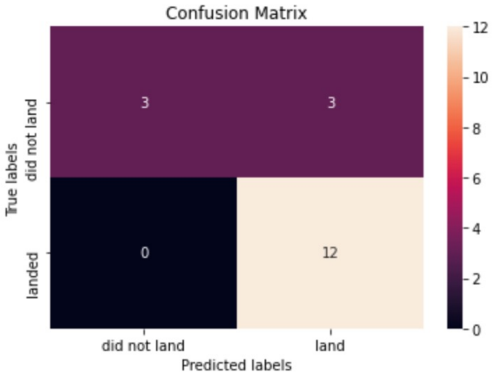
All four of the confusion matrixes produces the same outcome. The significant outcome being false positives.

		Predicted	
		Negative	Positive
Actual	Negative	True Negative (TN)	False Positive (FP) Type 1 Error
	Positive	False Negative (FN) Type 2 Error	True Positive (TP)

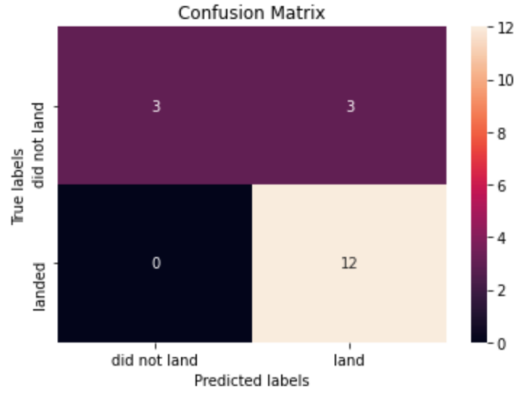
Logistic Regression



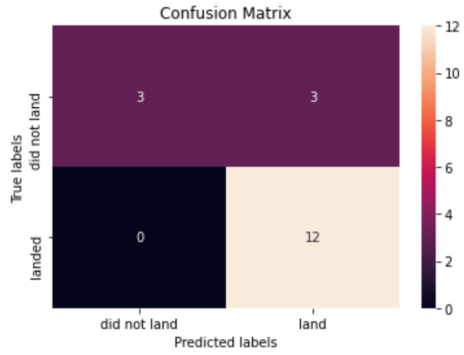
SVM



Tree



KNN



Conclusions

- As the flight number increase, so does the success Rate
- KSC LC -39A had the most successful launches from all the sites
- The success rates for low weighted payloads exceeds that of heavy weighted payloads
- The Decision Tree Method is the best model for the Data Set
- Orbits ES-L1, GEO, HEO and SSO have the best success rate

Thank you!