IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Name>
<Date>

Outline

Executive Summary

SpaceX F9 First Stage Booster Landing Success Rate.

I. **Problem:**

According to SpaceX, the manufacture of F9 first stage booster constitutes about 60% of its launch price. This estimation excludes environmental cost, and space debris these rockets create by not being re-used.

II. **Recommended Solution:**

By analyzing past launch data records, we can create machine learning models to predict which F9 first stage boosters will land successfully for re-use.

III. **Value:**

This will allow stakeholders to predict and reduce future F9 production and launch costs.

Introduction

In this Project, I'll be collecting data from SpaceX REST API and Wikipedia to predict whether SpaceX will attempt to land a Falcon 9 first stage Rocket or not. In the course of this project I'll be taking a descriptive and predictive analytical approach.

Additionally, I'll be performing several queries to extract meaningful information from data.

NOTE:

Refer to **Appendix** notebook at page 51 for any modifications created in this presentation if deviates from the course narrative.

Section 1

Section 1

Methodology

Methodology

Methodology:

- **Data Collection:**
  - Data was collected using SpaceX REST API and Scraped using Wikipedia SpaceX Page.
- **Data Wrangling:**
  - Data was processed using Watson Notebook along with Python libraries like Pandas and NumPy.
- **Visualization Analysis & Feature Engineering:**
  - Data was visualized using matplotlib, seaborn and transformed for modeling using pandas.
- **EDA with SQL:**
  - Access DB2 database using Python SQL and perform queries to answer notebook questions.
- **Build Interactive Map with Folium:**
  - Started a map with the zoom started at the NASA JSC. Added three SpaceX Launch Sites. Add Cluster Marker. Added Icon Marker. Calculated CCSFS site distance to its proximities.
- **Build a Dashboard with** Plotly **Dash:**
  - Built a dashboard using Python environment. Added a Pie Chart with counts of launches, and a Scatter Plot with payload, booster version and landing class. Performed EDA analysis to extract keys insights about.
- **Predictive Classification Analysis:**
  - Built four classification models using Sci-Kit Learn libraries. Used Grid Search CV for hyperparameters. Predicted the probability. Performed evaluation using grid score, log loss, and cross validation score.

Data Collection

How data was collected was collected in this project:

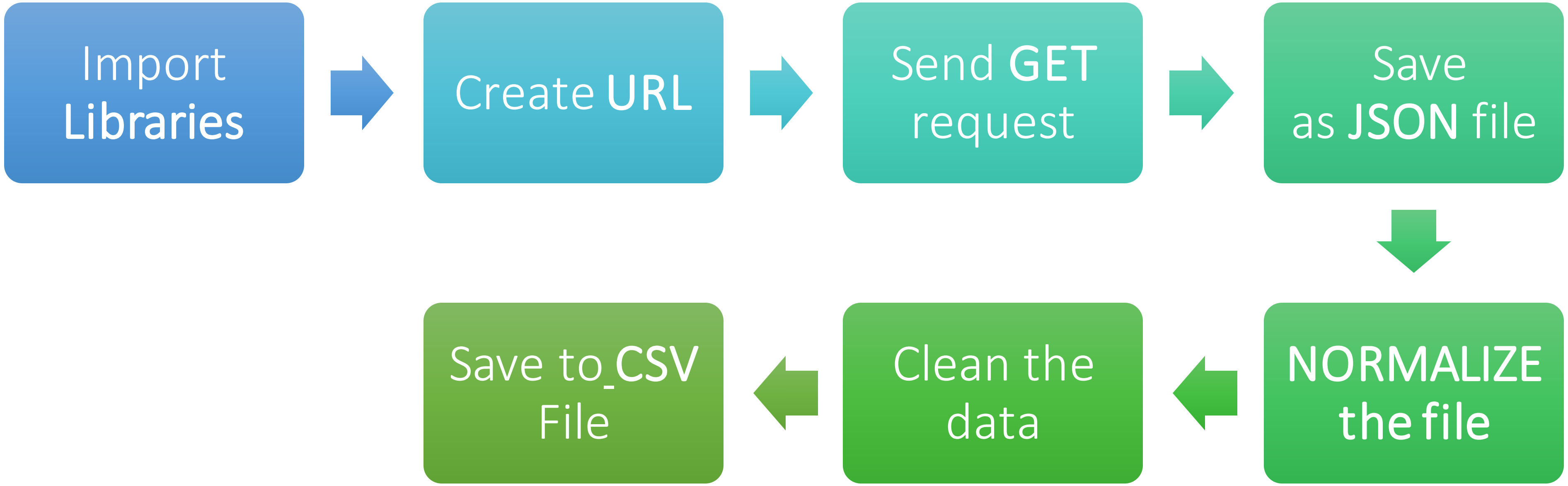
- **SpaceX REST API.**
- **Scraping the SpaceX Wikipedia Pages with Python Beautiful Soup.**

SpaceX **Rest API**

Scrape Wiki Pages using **Python Beautiful Soup**

Data Collection – SpaceX API

Flowchart with the SpaceX API data collection process:



[GitHub SpaceX API Request Link]()

Data Collection - Scraping

Below is a flowchart with collection process using Python Beautiful Soup:

1. Import Libraries
2. Create URL object
3. Send **GET** Request
4. Save Response
5. Create BeautifulSoup object
6. .find_all() tables
7. Target specific table[2]
8. Iterate Table To Extract Elements
9. Create DataFrame
10. Save to CSV File

Web Scraping Notebook Link

Falcon 9 Launches Wiki Link

Data Wrangling

Flowchart showing the process of data transformation and creation of a new feature called '*Outcome*':

Import
- Import **Libraries**

Read
- Read **CSV** File

Count
- Count the Value of Mission Outcomes

Create
- Create **Outcome** Column

File
- Save to **CSV** File

[Data Wrangling Notebook Link]

Flowchart showing the process for visualization analysis and feature engineering:

[Visualization Analysis & Feature Engineering Notebook Link]

Visualization Analysis & Feature Engineering

EDA with SQL

Flowchart showing how DB2 database was accessed using SQL:

Load CSV File to DB2 Database

↓

Import/Install Libraries

↓

Load SQL Magic

↓

Connect to DB2 with SQL Magic

↓

SELECT * SpaceX Table in DB2

↓

Execute SQL Queries To Explore The DataTable

[Exploratory Data Analysis with SQL Notebook Link]

Build an Interactive Map with Folium

Summary of objects added to the folium map:

1. First step was to initialize a folium map with the coordinates with NASA Johnson Space Center in Texas.
2. Add a circle with a popup name indicating Nasa's Space Center and each SpaceX launch site location.
3. Add column with color labels related to landing outcomes. Green for success landing, and red for failed landings.
4. Create a marker cluster object.
5. Add the color coded map icons into the marker cluster using the coordinates and the color labels previously created by iterating through each row in dataset.
6. Add a distance calculation from CCSFS launch site and its nearest proximities (Coast, City, Highway, Railroad).

Launch Sites Locations Analysis with Folium Notebook Link

Build a Dashboard with Plotly Dash

SpaceX Dash App Notebook Link

Summary of functions added to SpaceX Dashboard App:

Pie Chart: Added to visualize the successful landing rate for each site

Range Slider: Added to control the payload mass in the scatter plot

Scatter Plot: Added to visualize each Rocket Booster success rate based on its payload mass.

EDA Results

Exploratory Data Analysis Results with SpaceX Dash App:

- **All Sites:**
  - Pie Chart: The Launch Site with the highest landing percentage::
    - CCSFS SLC 40: This site has 52 landings which represents 51% of all successful landings. (Fig.1 Page 17)
  - Scatter Plot: The First Stage Booster with the most landings:
    - The Booster Version B5 stands out from all other boosters, successfully hauling payloads of all weights. However, the success ratio really increases after 12,000kg with 33 out of 3 successful landings. (Fig.2 Page 17)
- **Top Site KSC LC-39A:**
  - Pie Chart: Site Success rate:
    - While this site doesn't has the largest volume of launches, it has the highest success rate with 87 percent success ratio over 13 percent failed landings. (Fig.1 Page 18)
  - Scatter Plot:
    - Looking at scatter plot for this specific launch site, the Booster Version B5 has a successful landing record with only 2 failures out of 25 launches. (Fig.2 Page 18)

All Sites Graphic Charts

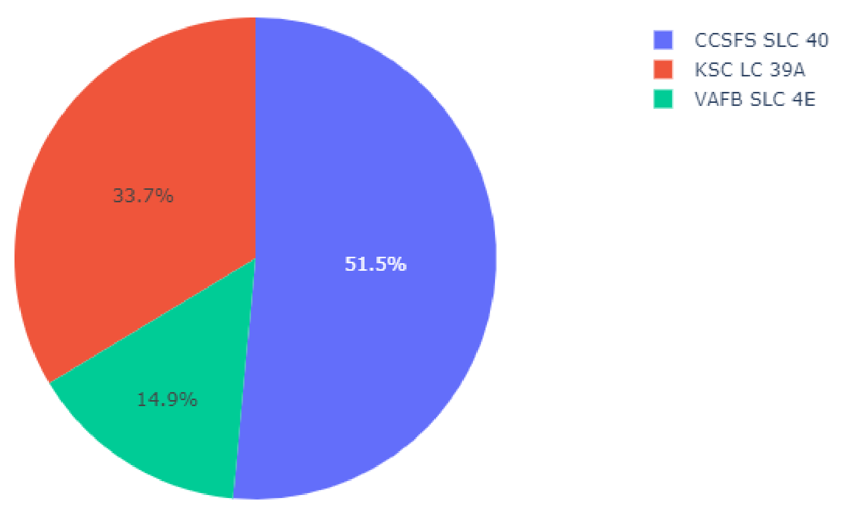


Fig. 1

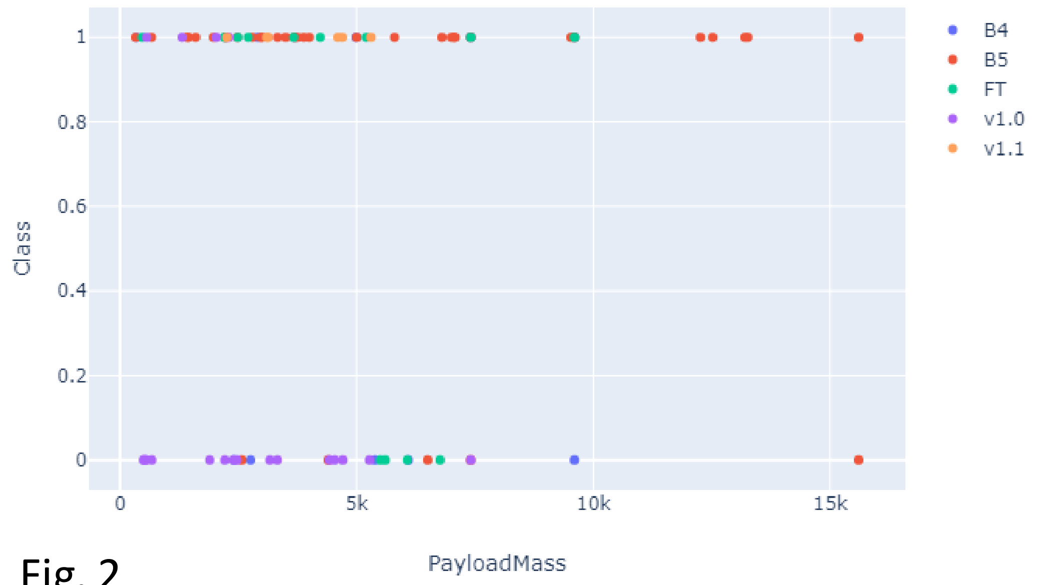


Fig. 2

KSC LC-39A Site Graphic Charts

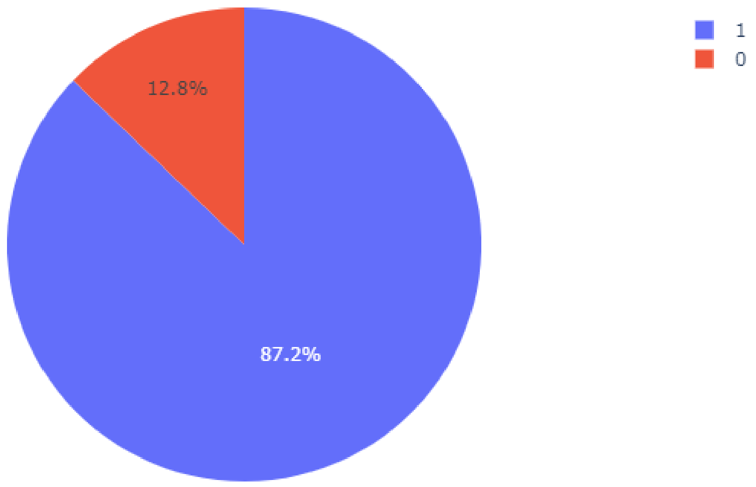


Fig. 1

Fig. 2

Predictive Analysis (Classification)

Summary of the classification models development and evaluation:

- **Four Models built in this project:**
- Logistic Regression, SVM, Decision Tree, and K-Nearest Neighbors.
- **Creating the Models:**
  - I Calculated the accuracy on test set based on the best parameters generated by Grid Search CV for each model. Created a prediction based on the test set. Plotted a Confusion Matrix to visualize the precision and recall distribution in each prediction. Calculated the probability and calculated the error rate using Log Loss function.
  - **Choosing the Best Model:**
  - Created a function to get a cross validation score and standard deviation for all 4 models. Called the function using the train dataset. Additionally, an independent evaluation was done with all the Grid Search CV scores for all models including the log loss rate. Visualized the results using Matplotlib.

Falcon9 First Stage Landing Prediction Lab Link

ML Prediction Results

Table Scores:

- **Best Classification Model:** (Page 20)
  - **Support Vector Machines(SVM) and KNN**: All models did equally well. Nevertheless, SVM is considered by many data scientists the best model for binary classification.
    - **Test set score**: 89%
    - **Log Loss**: 0.30%

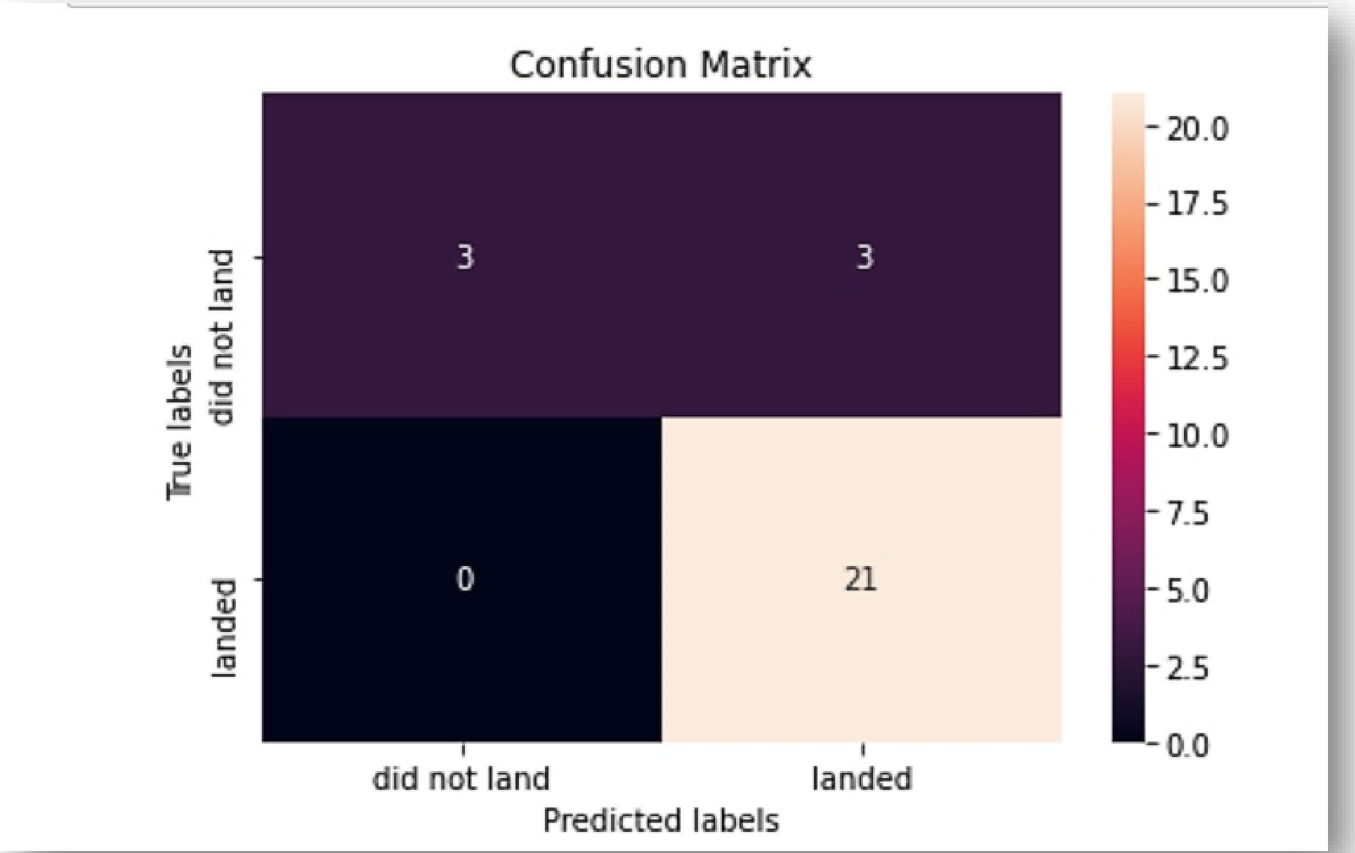
    All Models had the same score. SVM error rate was smaller than the others.
- **SVM Confusion Matrix:** (Page 21)
  - The test dataset has 27 total samples:
    - **Did not land label Prediction**: The model predicted 3 launches would not land. (1st column)
    - **Did not land true label**: The true target test labels shows 6 launches did not land. This means the prediction model missed 3. (1st row)
    - **Landed label prediction**: The model predicted 24 launches would land.(2nd column)
    - **Landed True Label**: Only 12 out of the 15 predicted landings would happen according to the true target test labels. (2nd row)

Classification Models Scores

Table with the Grid Search CV scores and Probability Error Rate(Log Loss).

Models	Test Set	Log Loss
Logistic Regression	.89	.32
Support Vector Machines	.89	.30
Decision Tree Classifier	.89	.33
K-Nearest Neighbours	.89	.30

All four models made same predictions.



SVM Confusion Matrix

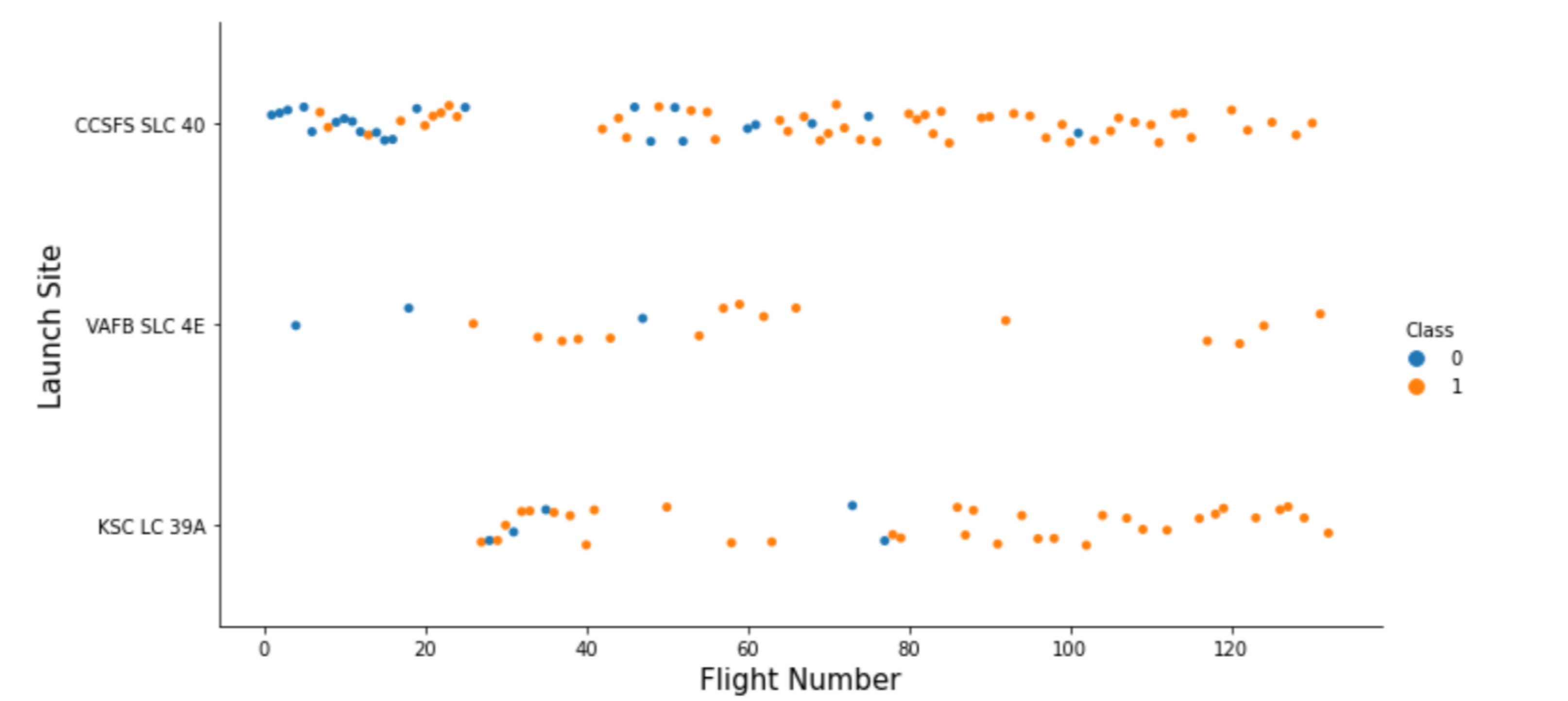
Section 2

Section 2

Insights drawn from EDA

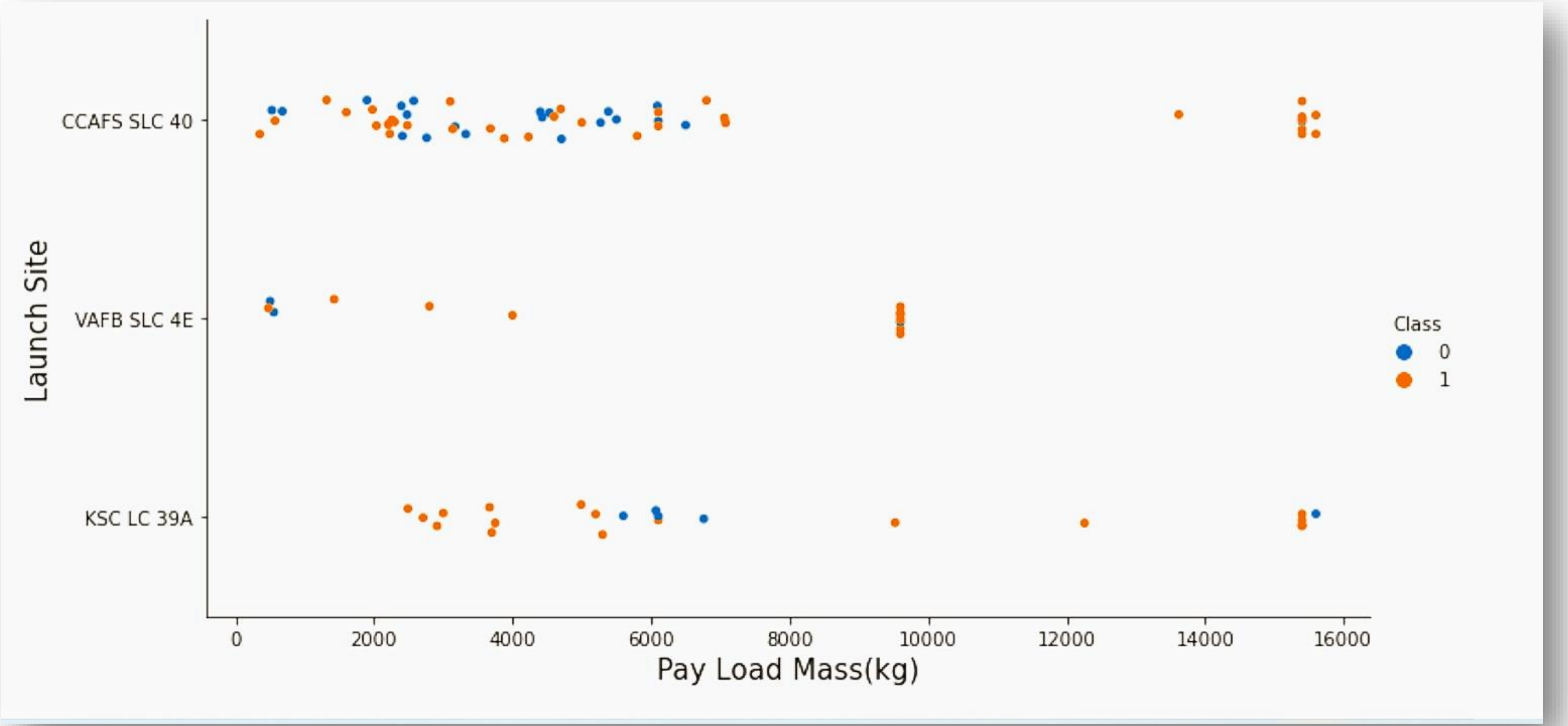
Flight Number vs. Launch Site

CCSFS has the largest volume of launches. Most failed landings were in the early stages(first flights).



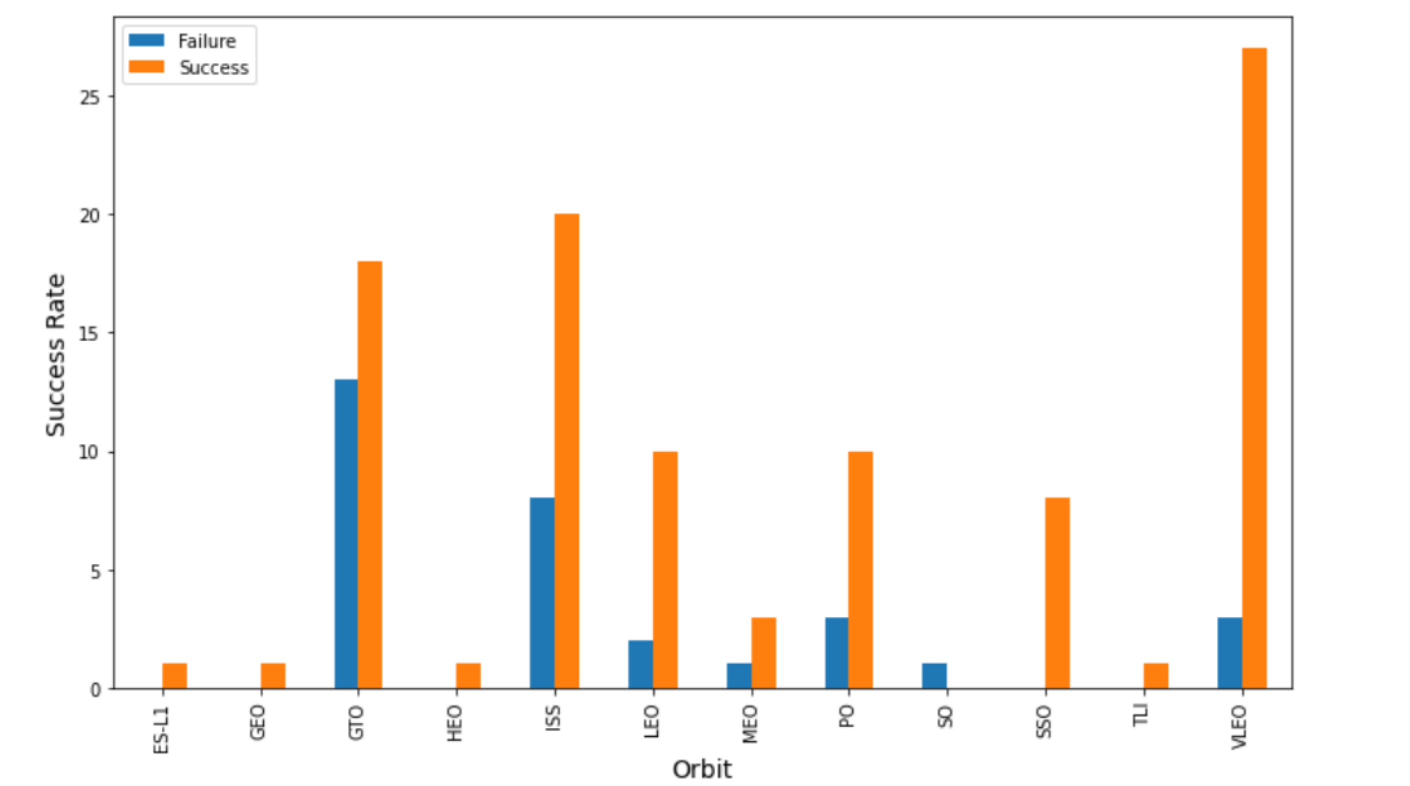
Payload vs. Launch Site

Largest cluster of launches are located under 8k for CCSFS. VAFB cluster is just over 9k and under 10k. KSC is between 2k and 8k.



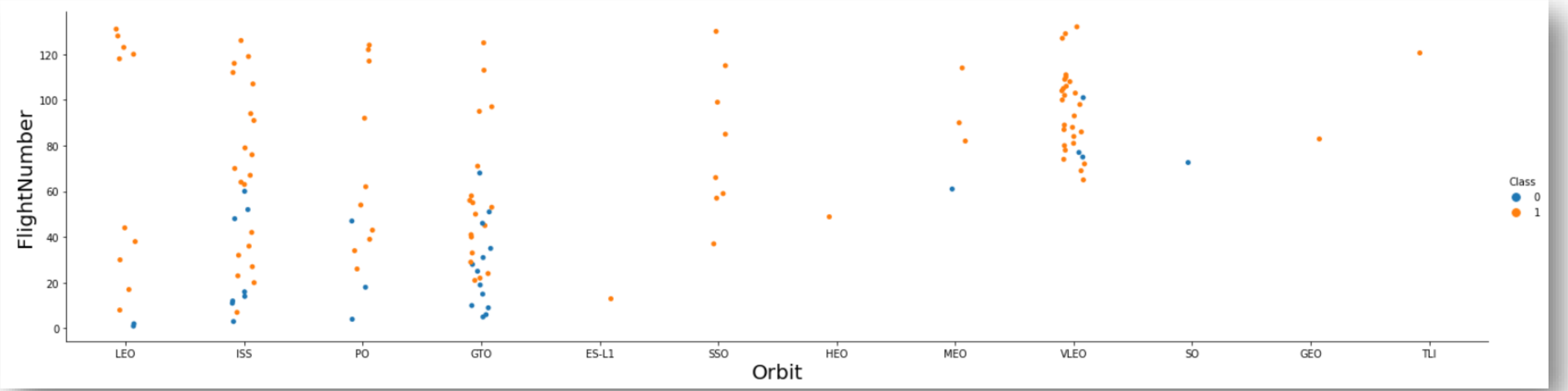
Success Rate vs. Orbit Type

VLEO shows the largest volume launches and highest success rate



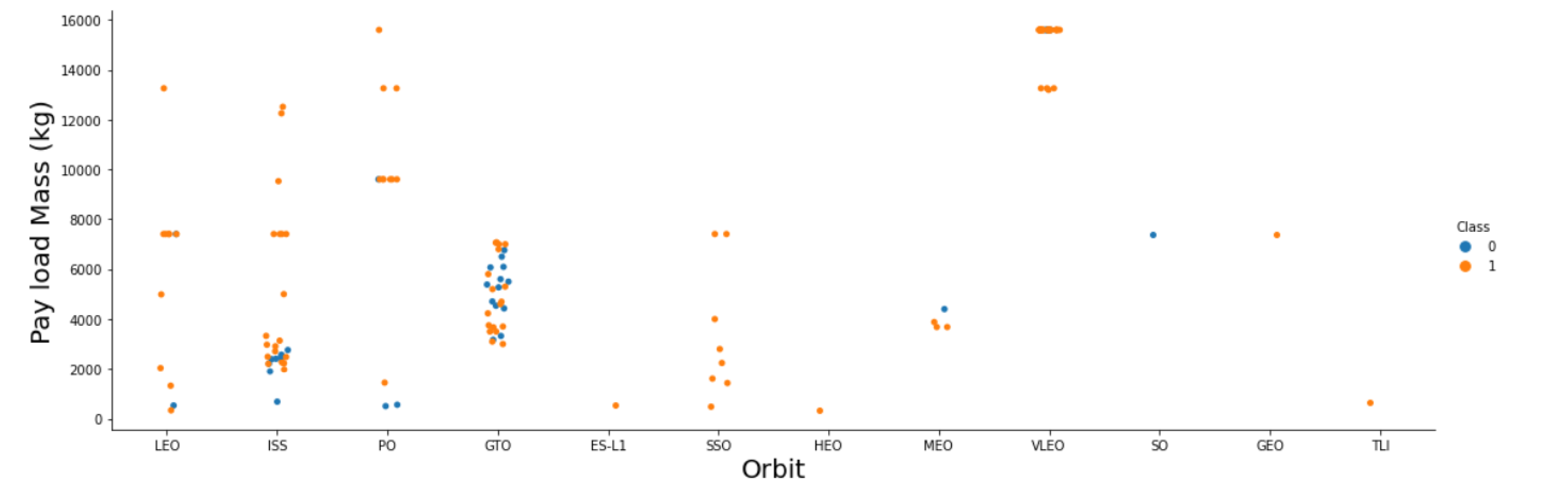
Flight Number vs. Orbit Type

ISS flight clusters are spread out through time. GTO had the largest number of failures. VLEO more recent flights.



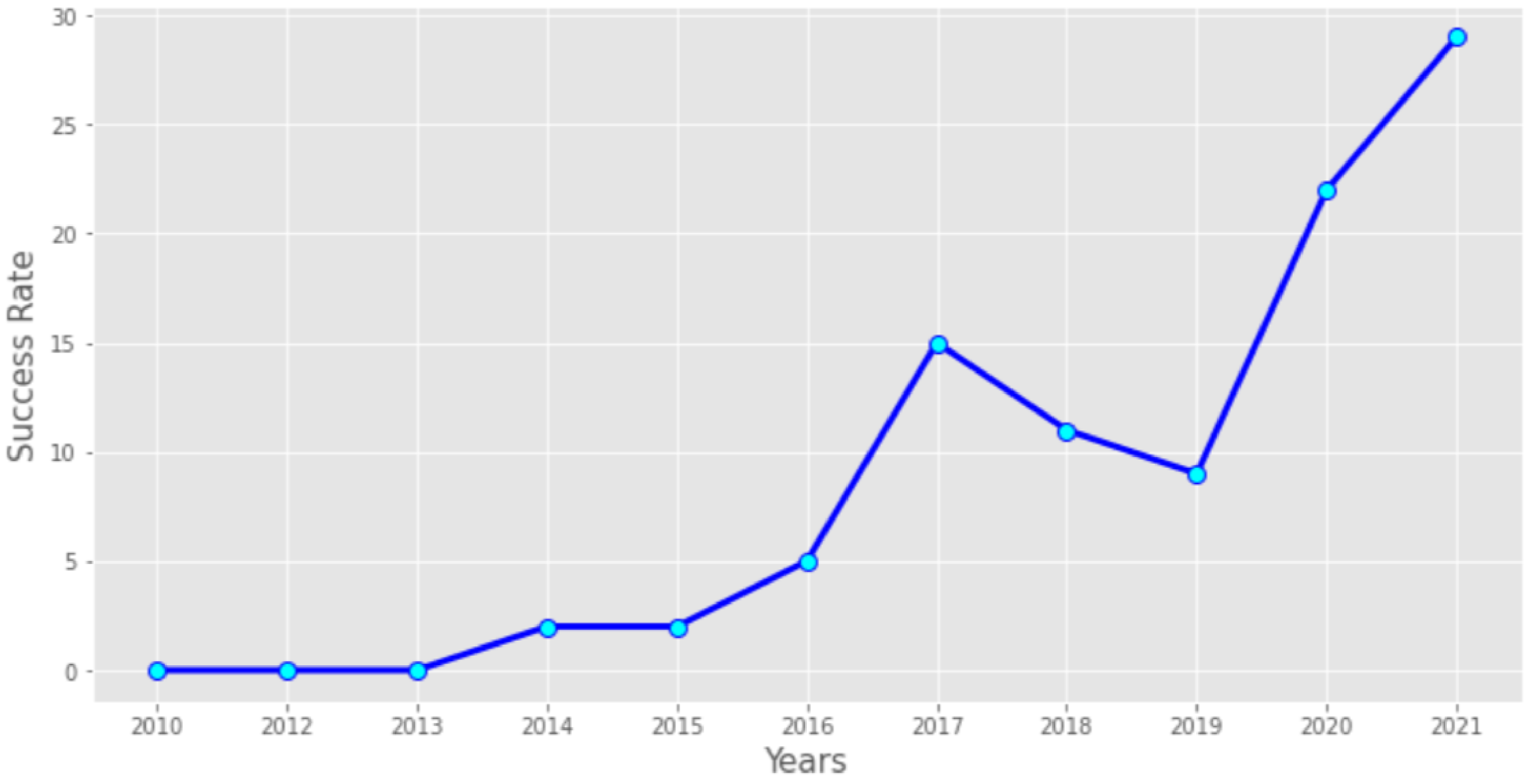
Payload vs. Orbit Type

ISS payloads are under 4k. GTO between 2k and 8k. VLEO over 12k.



Launch Success Rate Over The Years

Success rate increased up till 2017, declined till 2019, and spiked till the end of 2021



All Launch Site Names

All three SpaceX launch sites.

```
In [7]: unique_sites = spacex_df[['LaunchSite']].drop_duplicates()
        unique_sites
```

Out[7]:

	LaunchSite
0	CCSFS SLC 40
3	VAFB SLC 4E
26	KSC LC 39A

All CCSFS launch sites. Previously it was called CCAFS. See Appendix in page 51.

```
In [9]: ccs_site = spacex_df.loc[spacex_df['LaunchSite'].str.contains('CCS')]
        ccs_site.head()
```

Out[9]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	PayloadName	Customer	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	Serial	ReusedCount	LaunchSite	Longitude	Latitude	Block_Version	Class
0	1	2010-06-04	Falcon 9	7407.0	LEO	Dragon Qualification Unit	SpaceX	None None	1	False	False	False	NaN	1.0	B0003	0	CCSFS SLC 40	-80.577366	28.561857	v1.0	0
1	2	2012-05-22	Falcon 9	525.0	LEO	COTS Demo Flight 2	NASA(COTS)	None None	1	False	False	False	NaN	1.0	B0005	0	CCSFS SLC 40	-80.577366	28.561857	v1.0	0
2	3	2013-03-01	Falcon 9	677.0	ISS	CRS-2	NASA (CRS)	None None	1	False	False	False	NaN	1.0	B0007	0	CCSFS SLC 40	-80.577366	28.561857	v1.0	0
4	5	2013-12-03	Falcon 9	3170.0	GTO	SES-8	SES	None None	1	False	False	False	NaN	1.0	B1004	0	CCSFS SLC 40	-80.577366	28.561857	v1.0	0
5	6	2014-01-06	Falcon 9	3325.0	GTO	Thaicom 6	Thaicom	None None	1	False	False	False	NaN	1.0	B1005	0	CCSFS SLC 40	-80.577366	28.561857	v1.0	0

Launch Site Names Begin with 'CCS'

NASA's Total Payload Mass

Total weight in kilos SpaceX hauled into orbit for NASA.

```
In [56]: nasa_df = spacex_df.loc[spacex_df['Customer'].str.contains('NASA')]
         nasa_payload = nasa_df['PayloadMass'].sum()
         print(f'SpaceX launched into space a total payload of {nasa_payload} Kilograms on behalf of Nasa')

         SpaceX launched into space a total payload of 129170.7 Kilograms on behalf of Nasa
```

Average weight hauled per launch for Booster Version v1.1

```
In [57]: v1_df = spacex_df.loc[spacex_df['Block_Version'].str.contains('v1.1')]
         v1_payload_df = v1_df['PayloadMass'].mean().round(decimals=2)
         print(f'SpaceX launched into space an average of {v1_payload_df} Kilograms per payload using the F9 v1.1 Booster')

    SpaceX launched into space an average of 3848.17 Kilograms per payload using the F9 v1.1 Booster
```

Average Payload Mass by F9 v1.1

First Successful Ground Landing Date

First successful landing on ground.

```
In [37]: import time
         first_ground_land = spacex_df[spacex_df['Outcome'] == 'True RTLS'].min()['Date']
         f"The first ground landing date was in {first_ground_land.strftime('%B, %d, %Y')}"

Out[37]: 'The first ground landing date was in December, 22, 2015'
```

Successful Drone Ship Landing with Payload between 4000 and 6000

Successful landing on a drone with loads between 4k and 6k.

In [10]:
```
true_asds = spacex_df[spacex_df['Outcome'] == 'True ASDS']
asds_payload_range = true_asds[true_asds['PayloadMass'].between(4000,6000)]
print(asds_payload_range.shape)
asds_payload_range
```

(7, 21)

Out[10]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	PayloadName	Customer	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	Serial	ReusedCount	LaunchSite	Longitude	Latitude	Block_Version	Class
20	21	2016-05-06	Falcon 9	4696.0	GTO	JCSAT-2B	SKY Perfect JSAT Group	True ASDS	1	True	False	True	OCISLY	2.0	B1022	0	CCSFS SLC 40	-80.577366	28.561857	v1.1	1
23	24	2016-08-14	Falcon 9	4600.0	GTO	JCSAT-16	SKY Perfect JCSAT Group	True ASDS	1	True	False	True	OCISLY	2.0	B1026	0	CCSFS SLC 40	-80.577366	28.561857	v1.1	1
28	29	2017-03-30	Falcon 9	5300.0	GTO	SES-10	SES	True ASDS	2	True	True	True	OCISLY	2.0	B1021	1	KSC LC 39A	-80.603956	28.608058	v1.1	1
39	40	2017-10-11	Falcon 9	5200.0	GTO	SES-11 / Echostar 105	SES	True ASDS	2	True	True	True	OCISLY	3.0	B1031	1	KSC LC 39A	-80.603956	28.608058	FT	1
54	55	2018-08-07	Falcon 9	5800.0	GTO	Telkom-4	Telkom	True ASDS	2	True	True	True	OCISLY	5.0	B1046	3	CCSFS SLC 40	-80.577366	28.561857	B5	1
58	59	2018-12-03	Falcon 9	4000.0	SSO	SSO-A	Spaceflight Industries, Inc	True ASDS	3	True	True	True	JRTI	5.0	B1046	3	VAFB SLC 4E	-120.610829	34.632093	B5	1
69	70	2019-12-05	Falcon 9	5000.0	ISS	CRS-19	NASA (CRS)	True ASDS	1	True	False	True	OCISLY	5.0	B1059	5	CCSFS SLC 40	-80.577366	28.561857	B5	1

Total Number of Successful and Failure Mission Outcomes

Successful and failed mission outcomes.

In [41]:
```
count = spacex_df['Outcome'].str.contains('True').value_counts()
print(f'SpaceX had a total of {count.values[0]} successful missions, and a total of {count.values[1]} failed missions.')
```

```
SpaceX had a total of 101 successful missions, and a total of 31 failed missions.
```

Boosters Carried Maximum Payload

Heaviest load hauled by a rocket in a single launch.

```
In [40]: max_payload = spacex_df['PayloadMass'].max()
         max_payload_idx = spacex_df.loc[spacex_df['PayloadMass'].idxmax()]
         booster = max_payload_idx['BoosterVersion']
         print(f'The {booster} booster carried the heaviest payload with a total of {max_payload} Kilograms')

           The Falcon 9 booster carried the heaviest payload with a total of 15600.0 Kilograms
```

2015 Failed Drone Ship Landing Records

Failed landings for the year of 2015 on drone ships.

In [43]:
```
import datetime as dt
spacex_df['Date'] = pd.to_datetime(spacex_df['Date'])
launches_2015 = spacex_df[spacex_df['Date'].dt.year == 2015]
false_asds = launches_2015[launches_2015['Outcome'] == 'False ASDS']
filtered_false_asds = false_asds[['Outcome', 'BoosterVersion', 'LaunchSite', 'Date' ]]
filtered_false_asds
```

Out[43]:

	Outcome	BoosterVersion	LaunchSite	Date
11	False ASDS	Falcon 9	CCSFS SLC 40	2015-01-10
13	False ASDS	Falcon 9	CCSFS SLC 40	2015-04-14

Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

All landing outcomes ranked in a descending order for the dates mentioned above.

```
In [44]: date_2010_2017 = spacex_df[(spacex_df['Date'] >= '2010-06-04') & (spacex_df['Date'] <= '2017-03-20')]
outcome_value_count = date_2010_2017['Outcome'].value_counts().sort_values(ascending=False)

x = 0
for i in outcome_value_count.iteritems():
    x = x + 1
    if x < 2:
        print(f'{x}st landing outcome is "{i[0]}" with "{i[1]}" outcomes')
    elif x < 3:
        print(f'{x}nd landing outcome is "{i[0]}" with "{i[1]}"  outcomes')
    elif x < 4:
        print(f'{x}rd landing outcome is "{i[0]}" with "{i[1]}"  outcomes')
    else:
        print(f'{x}th landing outcome is "{i[0]}" with "{i[1]}"  outcomes')
```

```
1st landing outcome is "None None" with "9" outcomes
2nd landing outcome is "True ASDS" with "5"  outcomes
3rd landing outcome is "False ASDS" with "4"  outcomes
4th landing outcome is "True Ocean" with "3"  outcomes
5th landing outcome is "True RTLS" with "3"  outcomes
6th landing outcome is "False Ocean" with "2"  outcomes
7th landing outcome is "None ASDS" with "2"  outcomes
```

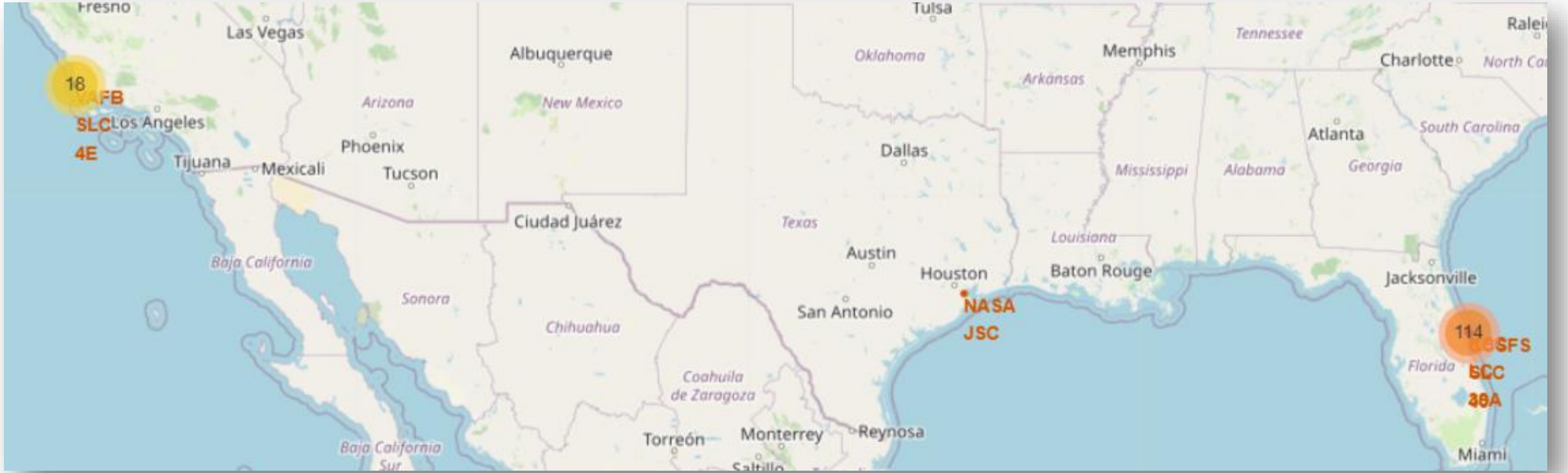
Section 4

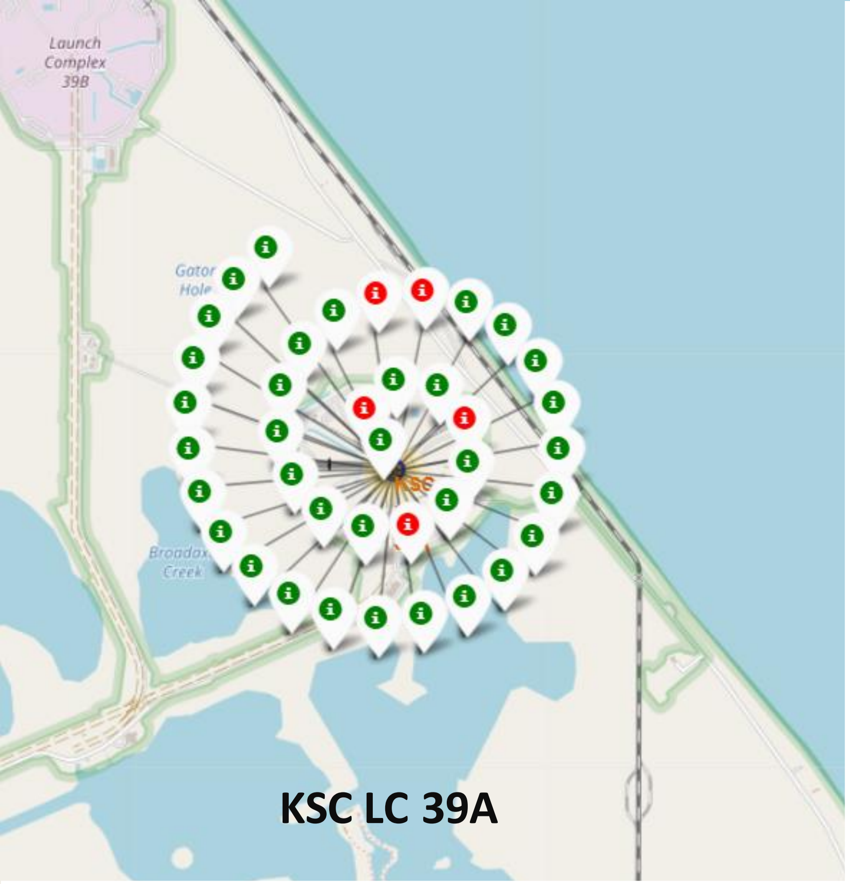
Section 3

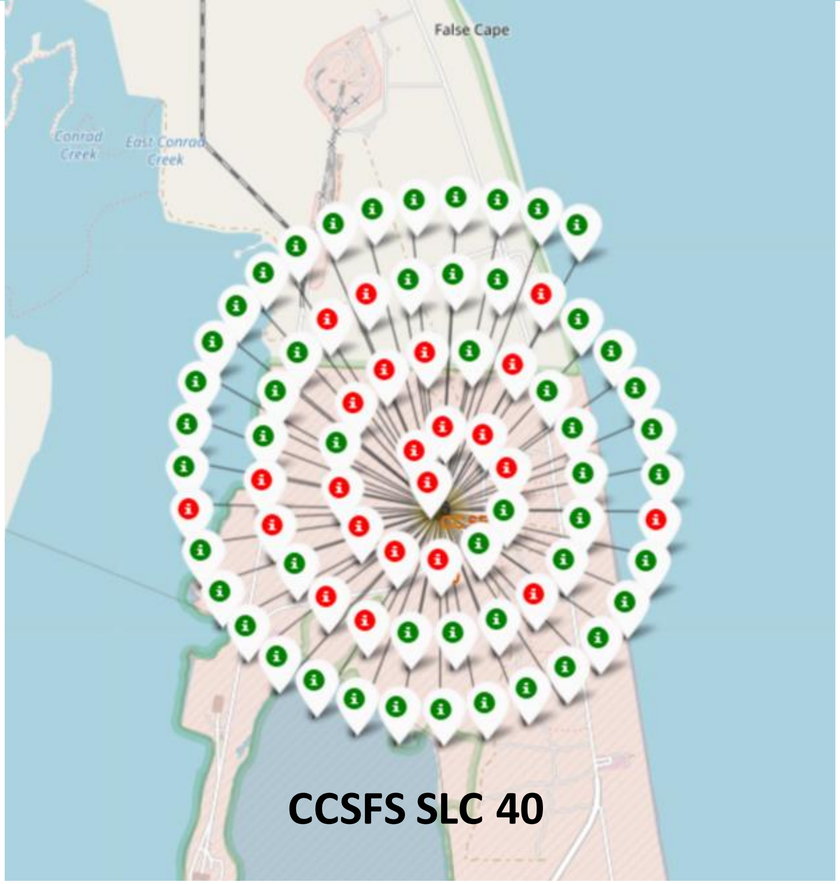
Launch Sites Proximities Analysis

SpaceX Launch Sites with Folium Map

All launch sites are located by the coast line.



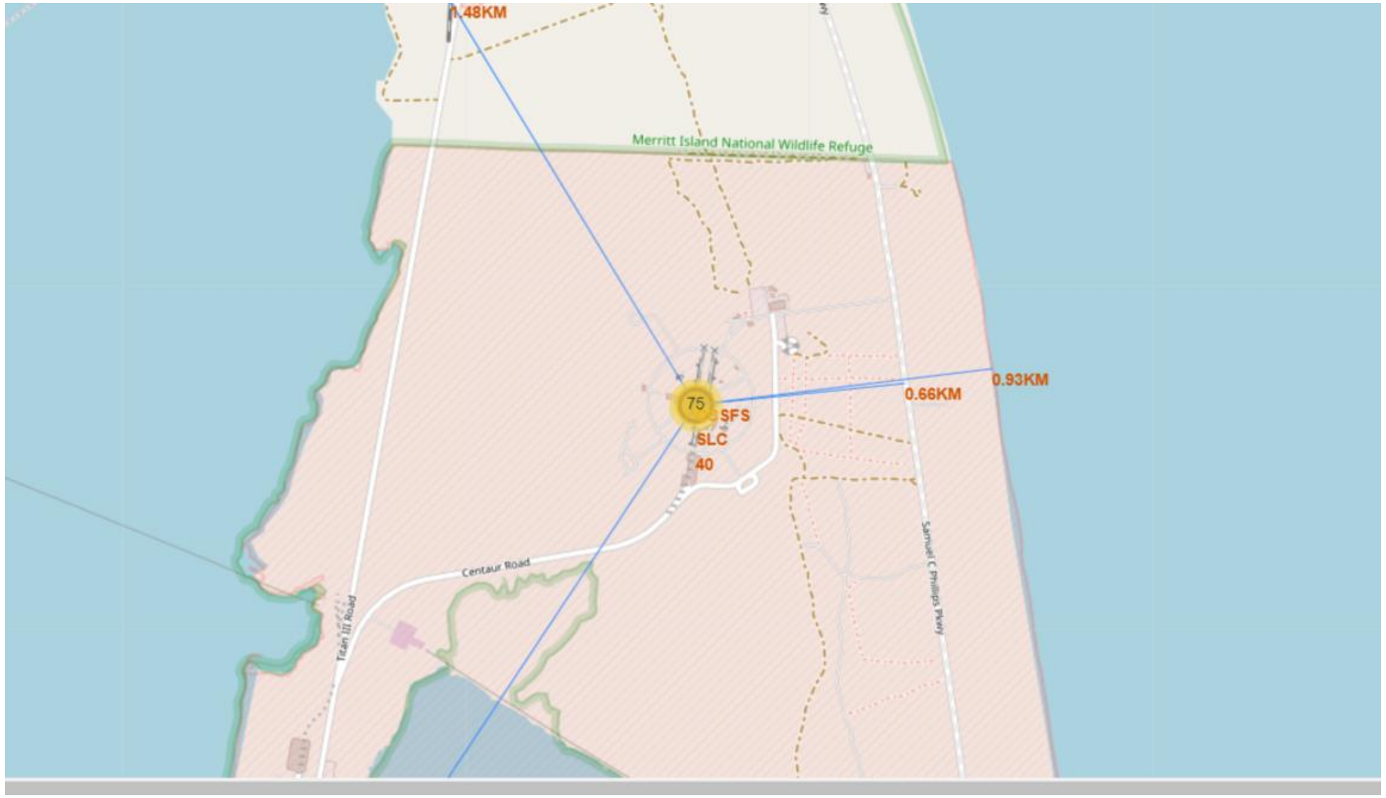




Launch Site Landing Results with Folium

Launch Site Proximities

Plotted line from CCSFS to its proximities.



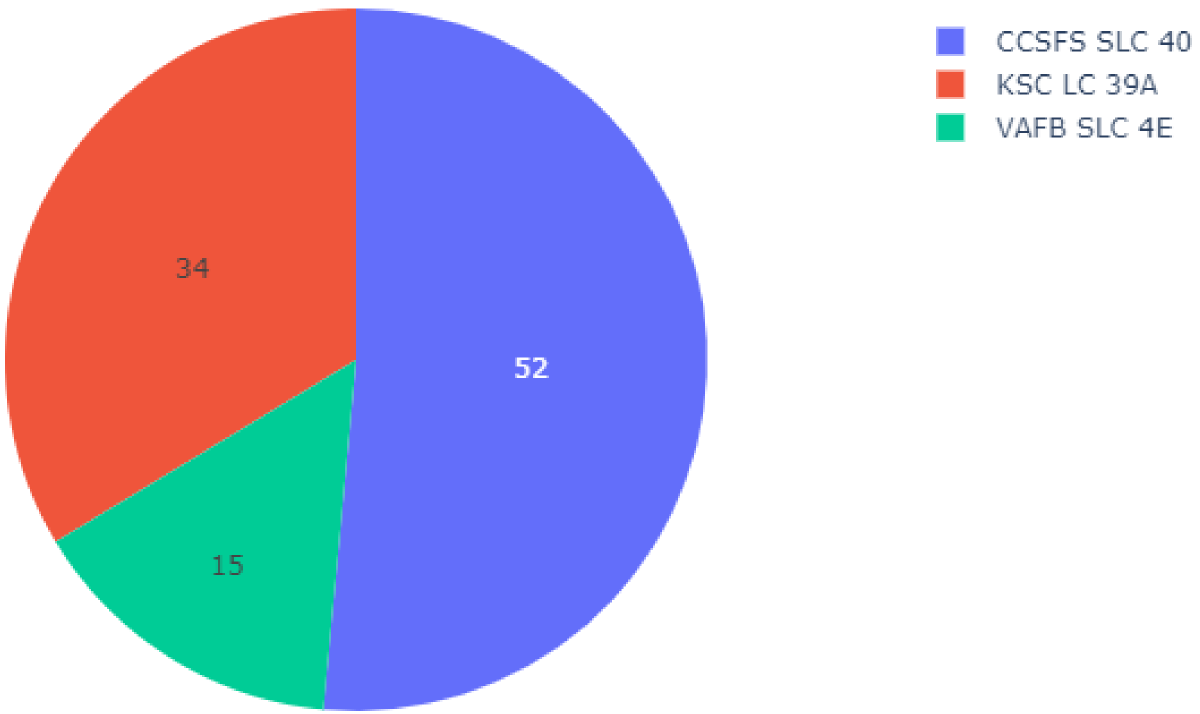
Section 5

Section
4

Build a Dashboard with Plotly Dash

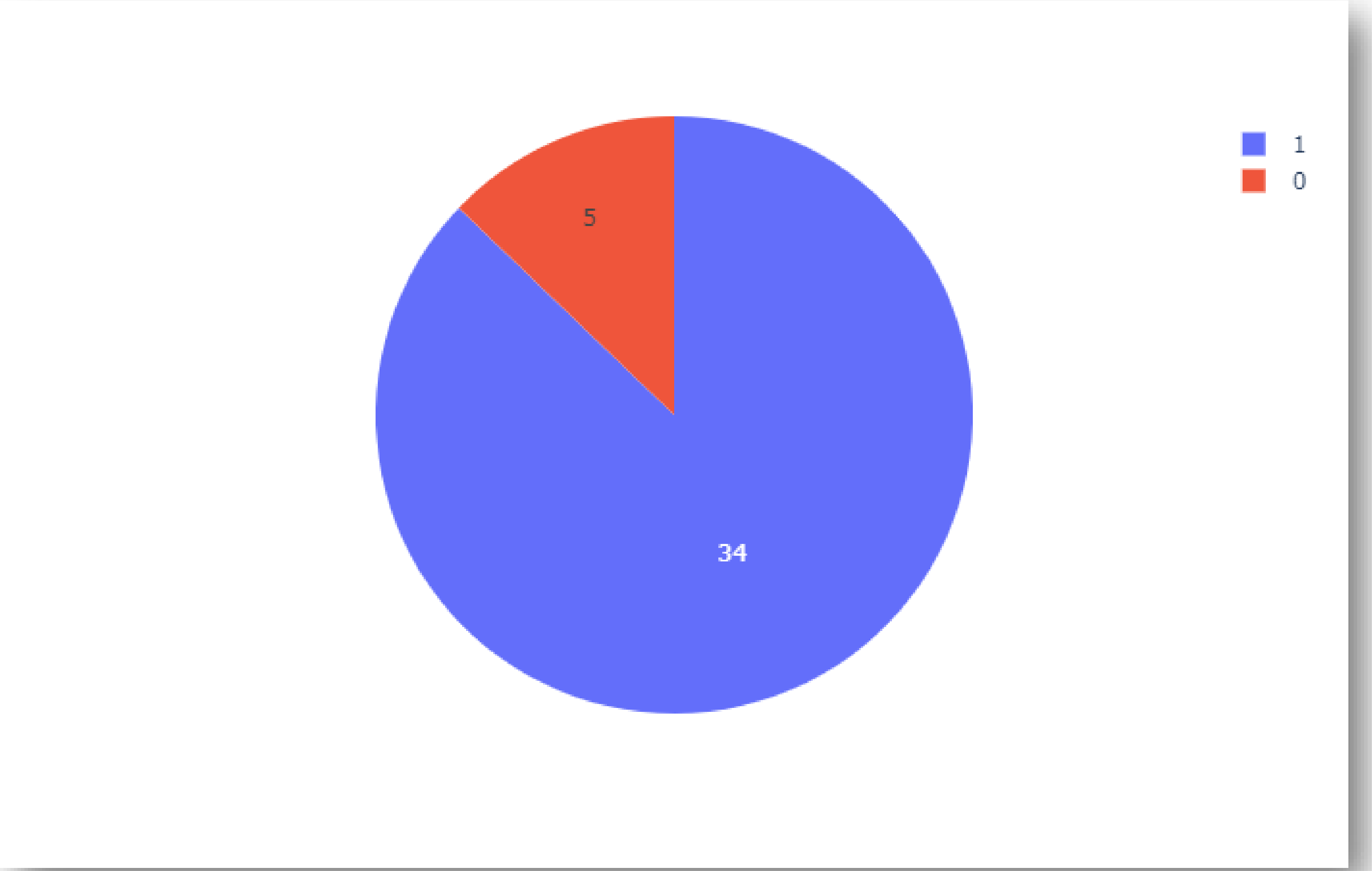
Launch Sites Total Successful Landings

CCSFS has the most successful launches. That’s 52 launches.



Launch Site With Best Success Ratio

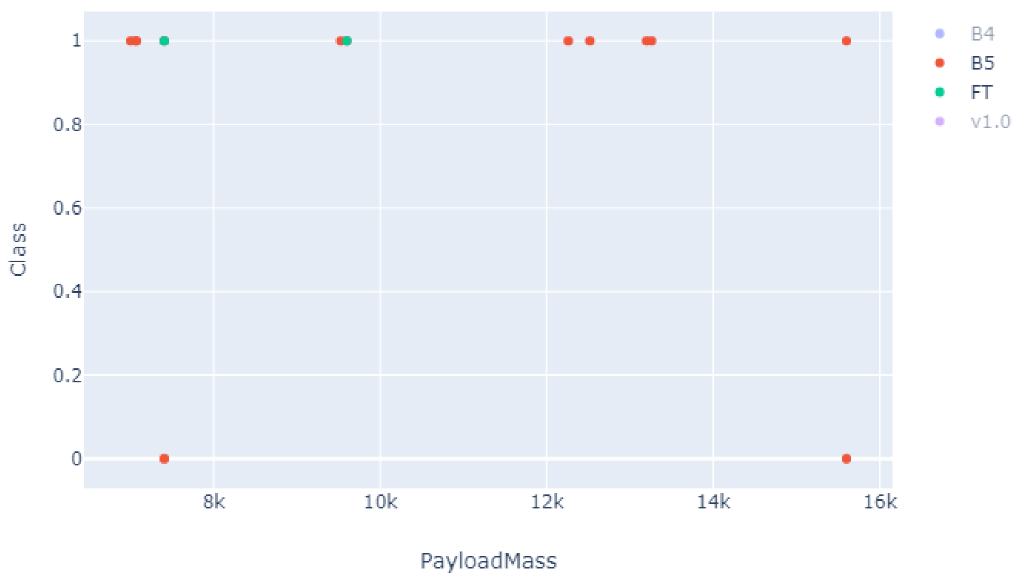
KSC LC 39A has the best success ratio. This is not to confuse with volume of launches.



Scatter Plot with Best Booster Versions

Large clusters of successful landings under 7k for all sites.

Scatter plot it's not showing but there is cluster at 16k mark with 24 count in class1 for block B5.



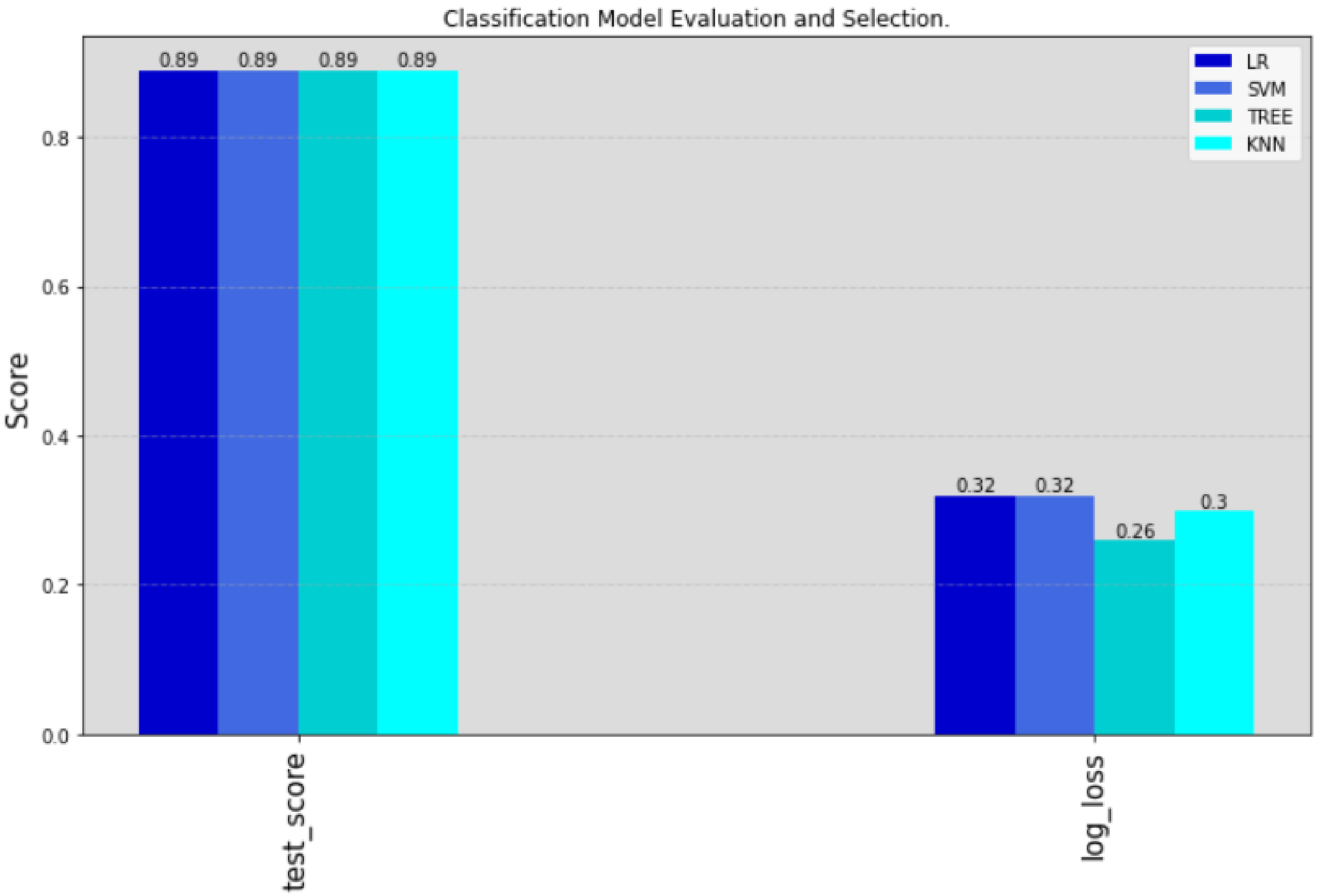
Section 6

Section 5

Predictive Analysis (Classification)

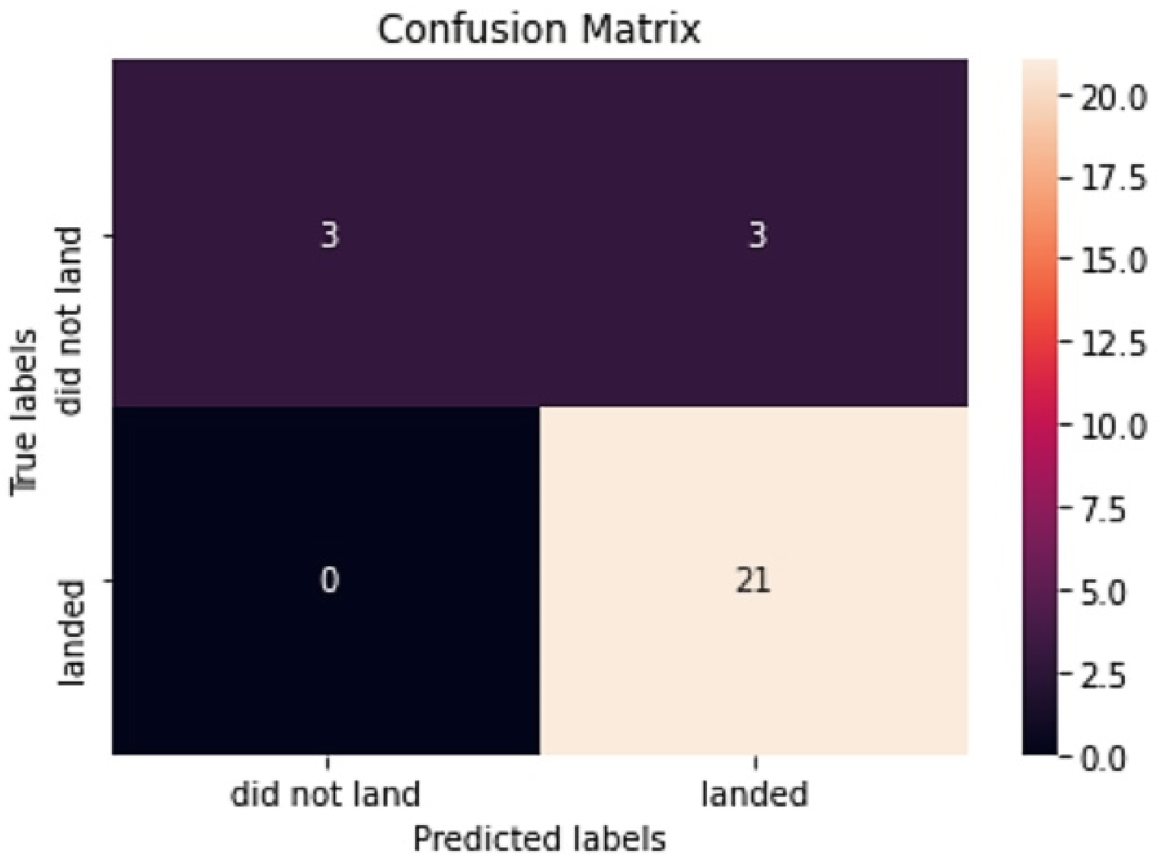
Classification Accuracy

All models are performing equally, Tree Classifier is showing more confidence with the error rate(Log Loss).



Confusion Matrix

Out of 27 predictions for the test set, all models only made the wrong prediction in 3 occasions. (Top right square).



Conclusions

- **Point 1:**
  - **What do the model scores represent?**
  - The 'test score tell us that the model can predict F9 future landing outcomes with an 89% accuracy. The Log Loss tell us the margin of error for all predictions. The smaller the error, the greater chance of each prediction being correct.
- **Point 2:**
  - Overall, all models do equally well if train test split random state parameters are set appropriately. In this case, Decision Tree Classifier model shows slightly more confident predictions than the others. It displays a Log Loss Rate of .26. However the Decision Tree seems to be unstable to any feature change, or any slightly change in the amount of samples. For this reason it's not a reliable model.
- **Point 3:**
  - Each model can be individually optimize by simultaneously changing the train test split and models parameters when applicable.
- **Point 4:**
  - Some of the boosters gets damaged beyond repair during the successful landing procedure, or just reassigned to be a Falcon Heavy Side Booster. For this reason a new model needs to be developed to determine which rockets will be reused after successful landing.

Appendix

Appendix Notebook Link

Thank you!