IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Dileep Ambali
25 May 2023

Outline

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

Executive Summary

- Summary of methodologies
    - Data Collection using SpaceX API
    - Data Collection from Wikipedia by Data Scraping
    - Data Wrangling
    - Exploratory Data Analysis using SQL
    - Exploratory Data Analysis using Visualization
    - Machine Learning Predictions

- Summary of all results
    - Exploratory Data Analysis Results
    - Interactive Visual Analytics with Dashboard
    - Predictive Analysis Result

Introduction

The most successful commercial space company is SpaceX. They advertise Falcon 9 rocket launches on its website with a cost of **62 million dollars;** much of the savings is because SpaceX can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch.

We use publicly available information to predict if a SpaceX rocket will land successfully as well as predict the cost of operation. Competitors of SpaceX can use this information to beat SpaceX to the race.

Introduction

In this report we try to predict the cost of launch of SpaceX Falcon9 rockets from publicly available historical data.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web-scraping Wikipedia
- Perform data wrangling
  - Data was converted into a flat table from raw JSON format
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Different classification algorithms were used and tested to find the optimal predictive model

Data Collection

Data was collected in 2 ways: SpaceX API and Web-Scraping.

The data from the **API** was collected using HTTP requests and stored in pandas tables. The **'rockets' endpoint** was used to get launch details of all rockets. Since we are only interested in Falcon9 rockets, all other rocket types were omitted.

Certain records are easier found on websites like **Wikipedia**. We use **Web-Scraping** to extract required tables from the website. The table also contains unnecessary information which needed to be cleaned up.

Data Collection – SpaceX API

```
# We can use the requests library to make a GET request
to the SpaceX API
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)

# Data is returned in JSON format. We can convert it to a
python dictionary using the .json() method
data = pd.json_normalize(response.json())
```

```
# We will now create a new dataframe with the features we
want
# launch_dict is a dictionary with the data we want to
display in our dataframe
data_falcon9 = pd.DataFrame(launch_dict)

# We will only keep the Falcon 9 launches and remove the
rest
data_falcon9 = data_falcon9[data_falcon9['BoosterVersion']
=='Falcon 9']
```

```
# Calculate the mean value of PayloadMass column
MeanPayLoadMass = data_falcon9['PayloadMass'].mean()

# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].replace(np.nan,
MeanPayLoadMass, inplace=True)
```

Github: IBM-Applied-Data-Science-Capstone-Final-Assignment/1. Data Collection API.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub

Data Collection - Scraping

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

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

```
# 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')

# The first table contains our data.
first_launch_table = html_tables[2]
```

Fetch html from URL

↓

Parse HTML

↓

Find Necessary Table

↓

Extract information form Table

↓

Format to table

↓

Store data

Github: IBM-Applied-Data-Science-Capstone-Final-Assignment/2. Data Collectio with Web Scraping.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub

Data Wrangling

Data Wrangling is the process of cleaning and unifying messy and complex data sets for easy access and Exploratory Data Analysis (EDA).

In this stage, we process the data that we have collected to be more EDA friendly. We add a new data field called ‘Class’ which will be **One** if the first stage of the rocket landed successfully and **Zero** if the first stage did not land.

In other words, we create a binary landing outcome column in the data table.

Github: IBM-Applied-Data-Science-Capstone-Final-Assignment/3. Data Wrangling.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub

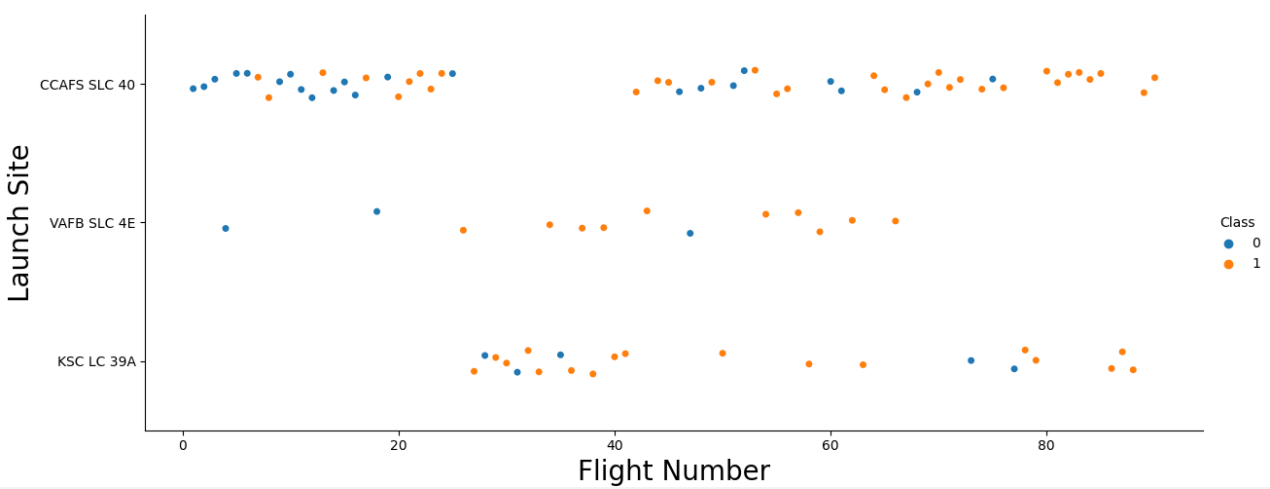
EDA with SQL

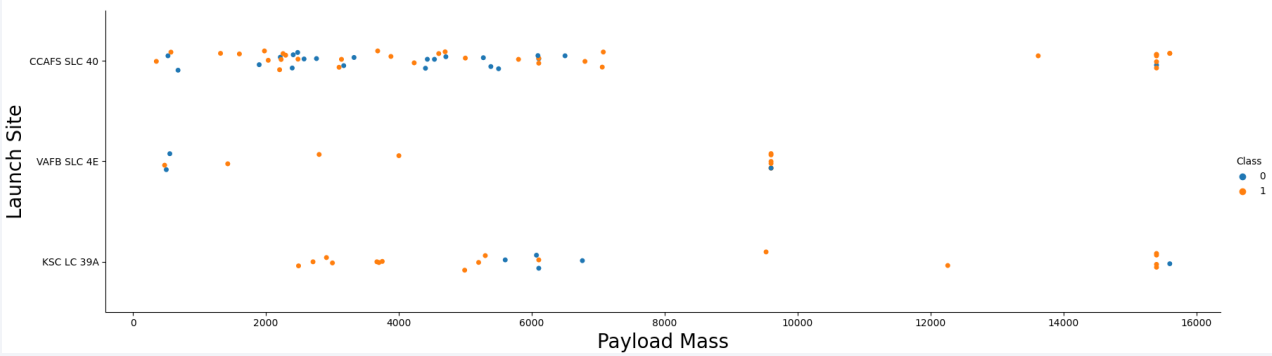
- Some of the many queries that we performed are:
  - Listing the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000.
  - Listing the total number of successful and failure mission outcomes.
  - Listing the names of the booster_versions which have carried the maximum payload mass.
  - Listing the failed landing_outcomes in drone ship, their booster versions, and launch sites names for in year 2015.
  - Rank the count of landing outcomes or success between the date 2010-06-04 and 2017-03-20, in descending order.

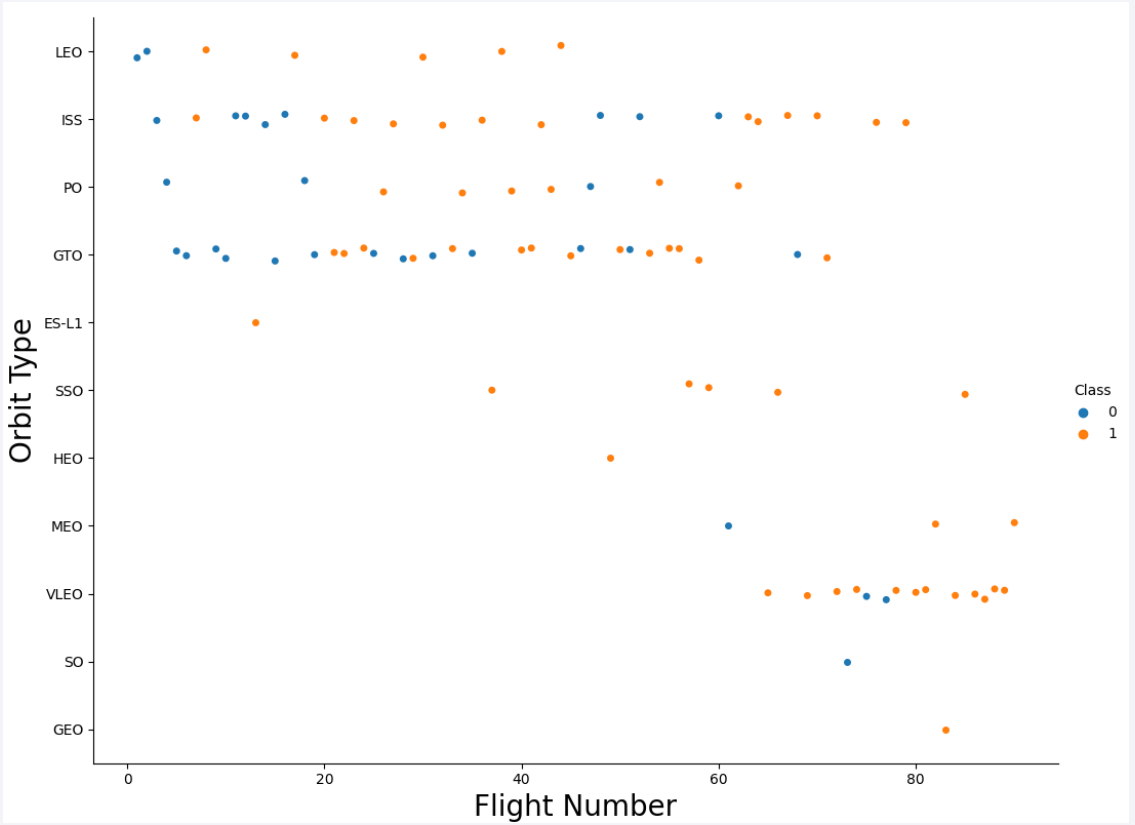
Github: IBM-Applied-Data-Science-Capstone-Final-Assignment/4. Exploratory Data Analysis with SQL.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub

EDA with Data Visualization

Categorical Plot: T

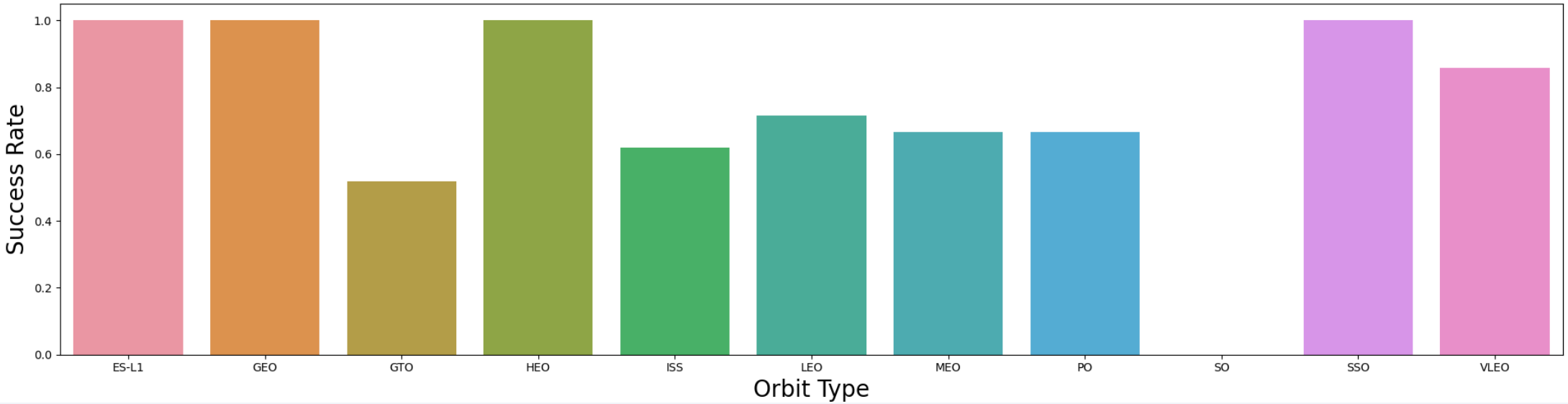






EDA with Data Visualization

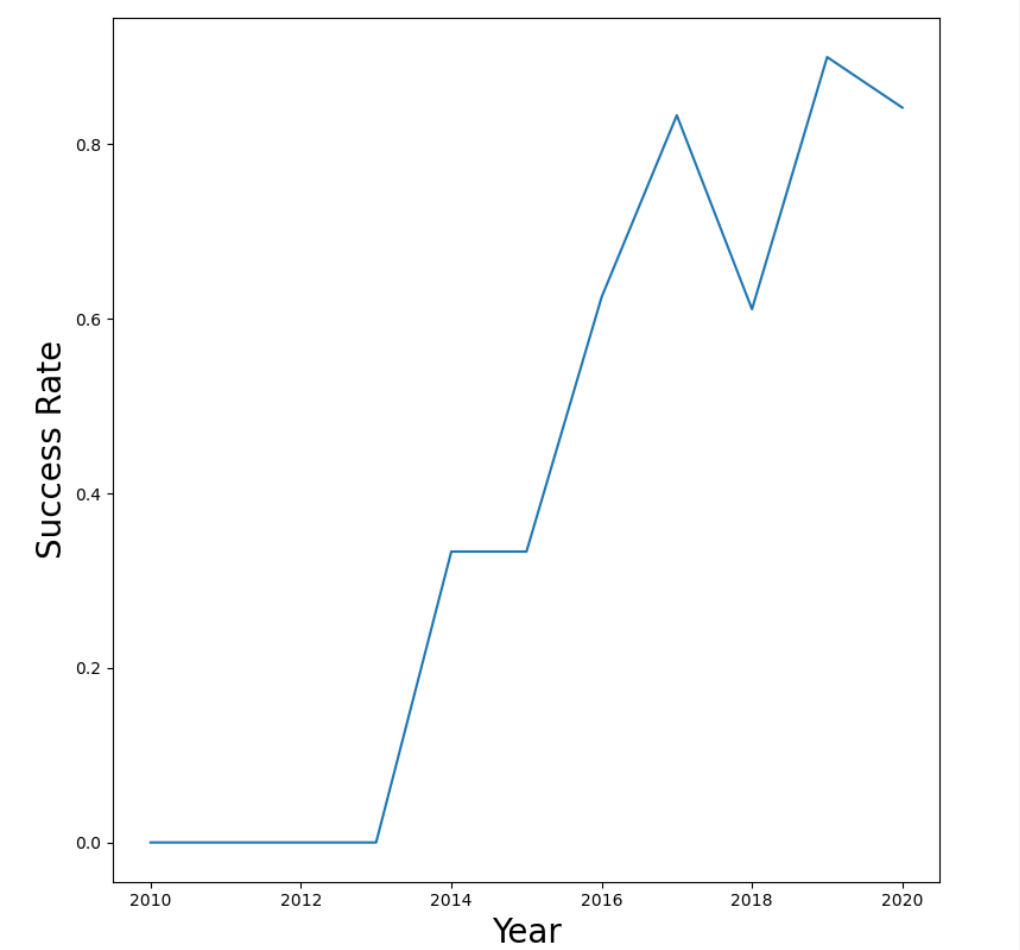
Bar Graph: To show relative values of different datapoints



Github: [IBM-Applied-Data-Science-Capstone-Final-Assignment/5. Exploratory Data Analysis with Matplotlib and Pandas.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub]

EDA with Data Visualization

Line Graph: To show general trends



Github: IBM-Applied-Data-Science-Capstone-Final-Assignment/5. Exploratory Data Analysis with Matplotlib and Pandas.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub

Build an Interactive Map with Folium

To visualize the launch data into an interactive map. We took the latitude and longitude coordinates at each launch site and added a circle marker around each launch site with a label of the name of the launch site.

We assigned markers for each launch at each site with 2 colours: Red (Fail) and Green (Success)

We then used the Haversine's formula to calculated the distance of the launch sites to various landmark to find answer to the questions of:

- How close the launch sites with railways, highways and coastlines?
- How close the launch sites with nearby cities?

Github: IBM-Applied-Data-Science-Capstone-Final-Assignment/6. Interactive Visual Analytics with Folium.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub

Build a Dashboard with Plotly Dash

We built an interactive dashboard with **Plotly Dash** which allowing the user to play around with the data as needed.

We plotted pie charts showing the total launches by a certain sites.

We then plotted scatter graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version.

The payload mass is selectable by **sliders** and the site is selectable by **dropdown**.

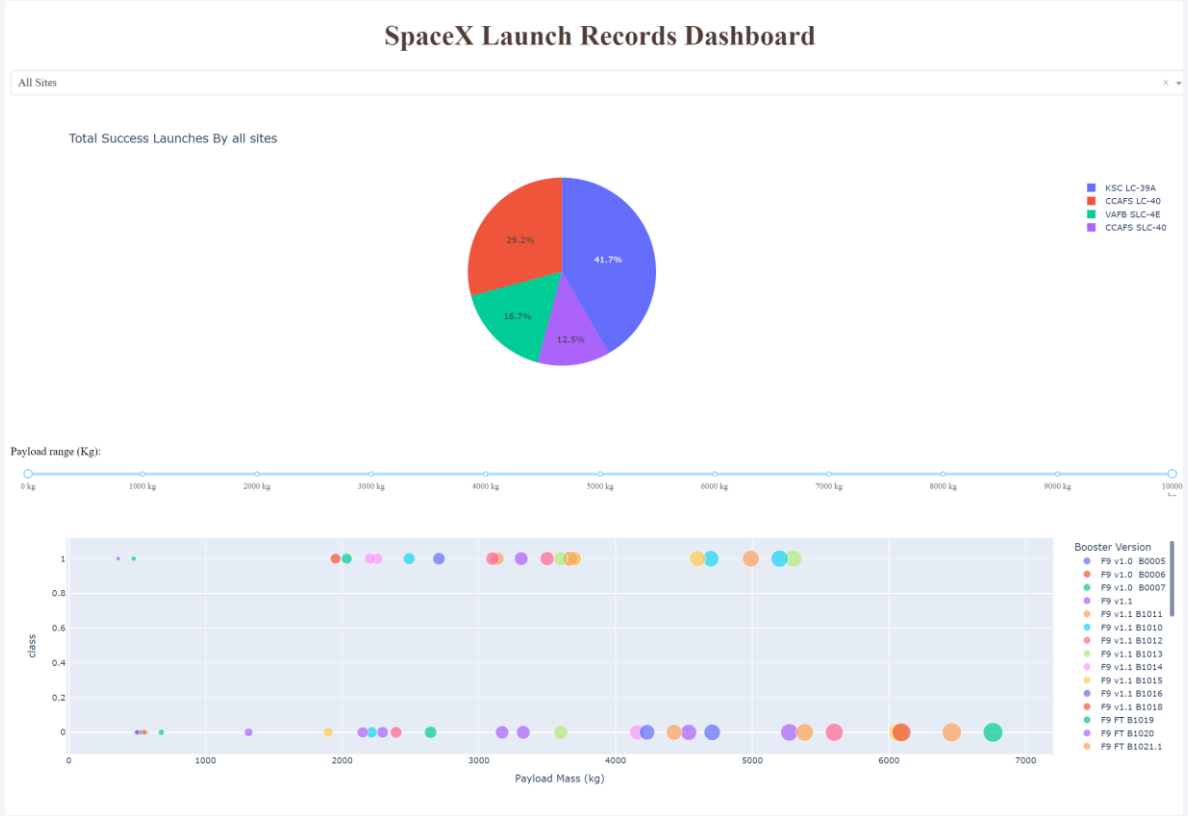
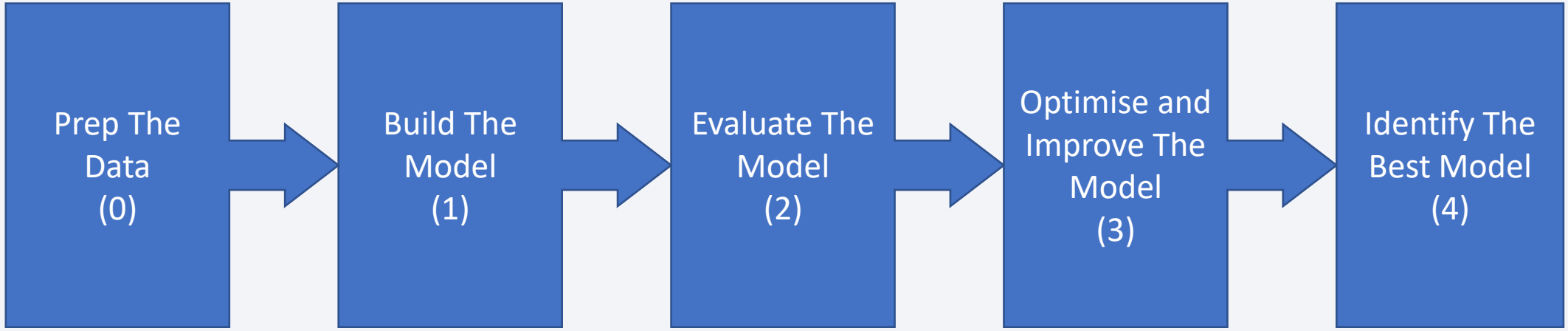


Fig. Screenshot of the Dashboard

Github: [IBM-Applied-Data-Science-Capstone-Final-Assignment/7. Interactive Dashboard with Ploty Dash.py at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub]

Predictive Analysis (Classification)



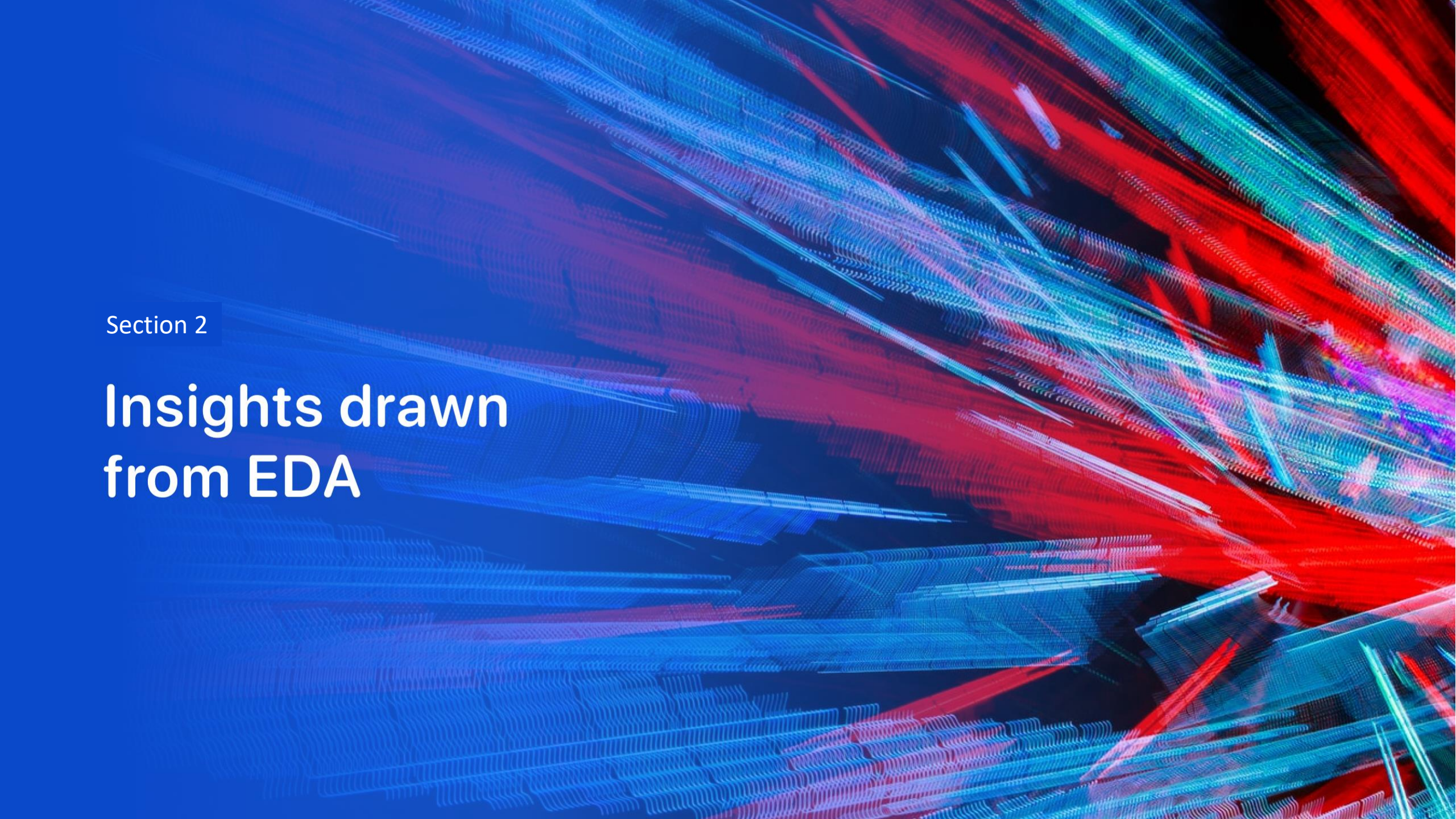
0. Split the standardised data into **'Test'** and **'Train'** sets
1. Pick and build a mode. Feed it the training data.
2. Find out the score (accuracy) of the model on the test set. Use Confusion matrix for visual confirmation.
3. Play around with different training and test sets. Choose a different model if you have to and go back to step 1.
4. List the accuracies of all the model. Choose the best performing model for the application.

Github: IBM-Applied-Data-Science-Capstone-Final-Assignment/8. Machine Learning Prediction.ipynb at main · DileepAmbali/IBM-Applied-Data-Science-Capstone-Final-Assignment · GitHub

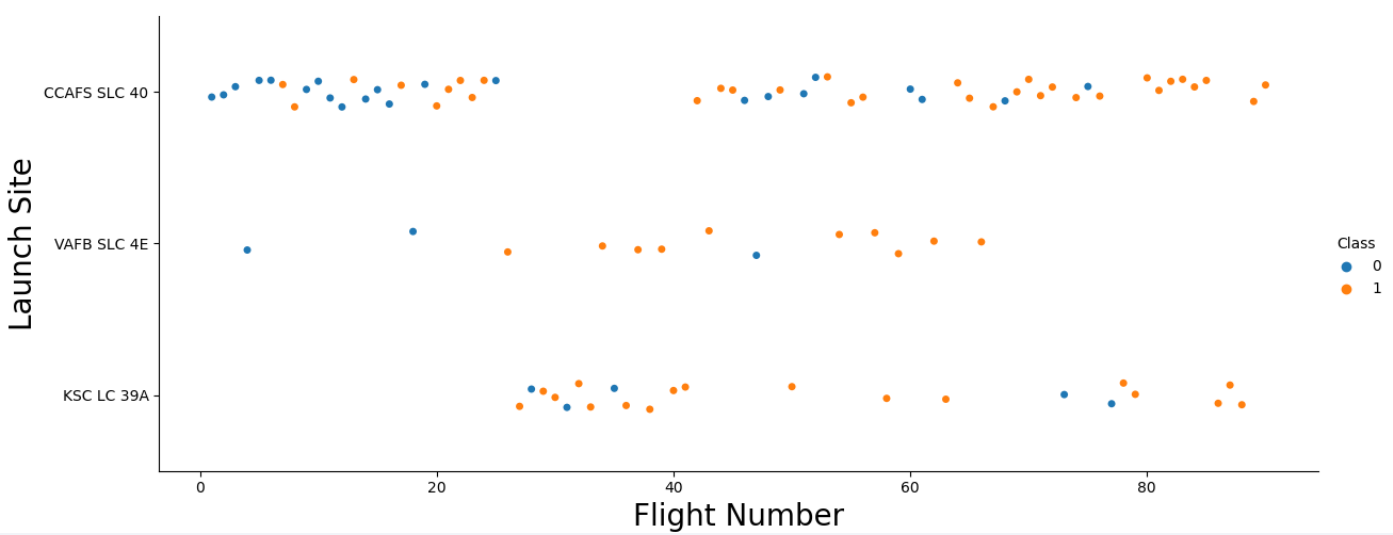
Results

The results will be categorized into 3 main groups:

- Exploratory Data Analysis results
- Interactive Analysis Result
- Predictive Analysis results

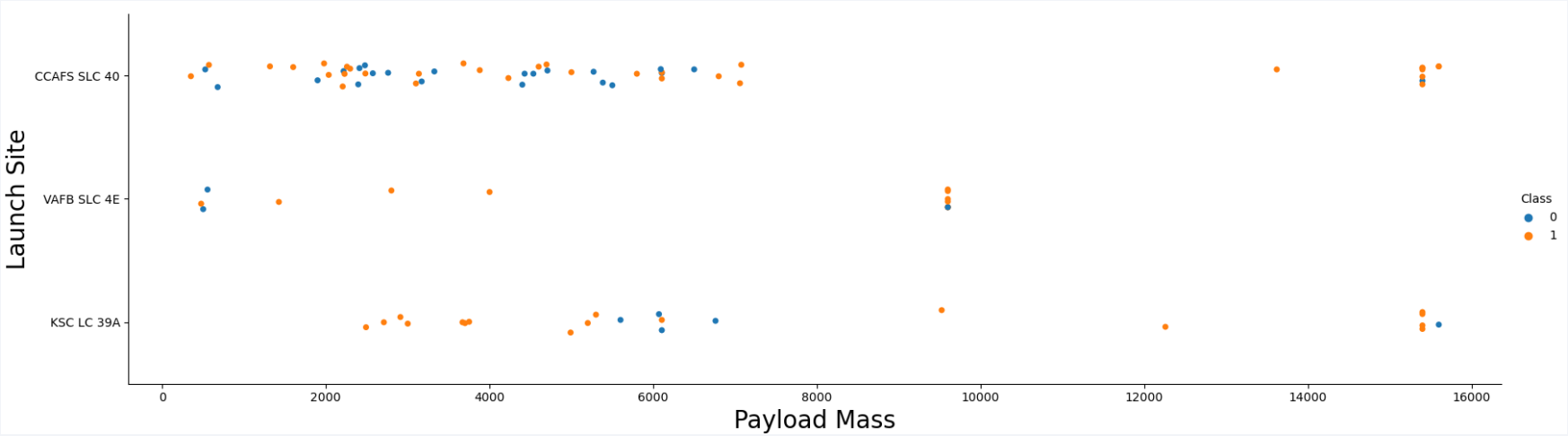

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



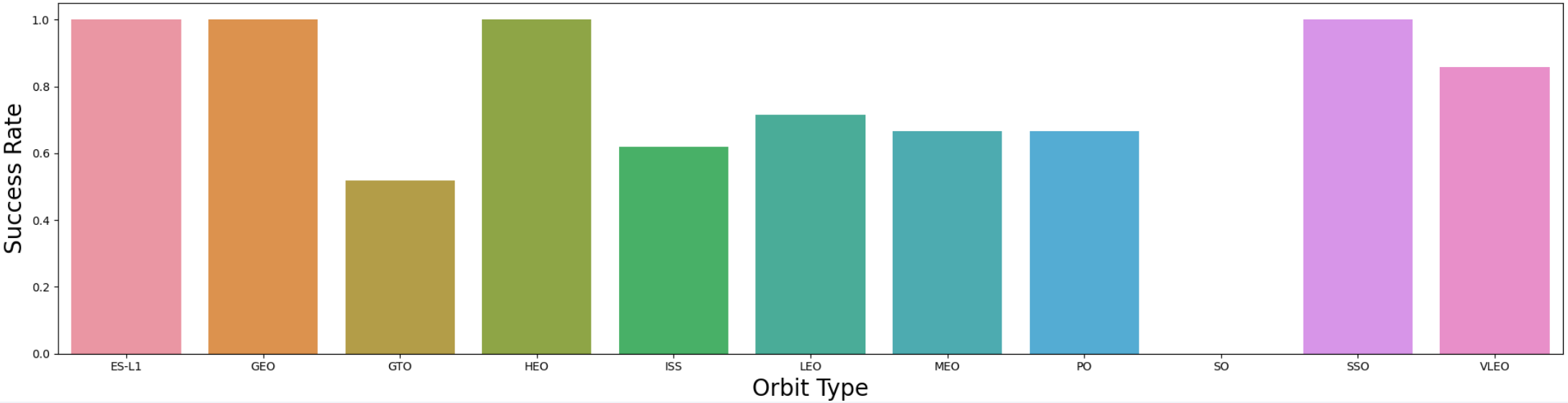
- Plot tells us where a particular launch took place. From the plot, its clear that launches did not take place at **'KSC LC 39A'** up until around the 25th launch.
- It also shows that launch site **'VAFB SLC 4E'** does not see any activity after around the 65th launch.
- Most of the launches takes place at **'CCAFS SLC 40'** and **'VAFB SLC 4E'** sees the least activity.
- It also shows that up until the 10th launch most were considered failures, where as after 80th launch all are considered successes.

Payload vs. Launch Site



- Plot shows us the relationship between a particular launch site and the payload mass that was launched.
- The plot shows that heavy payloads over 10000Kg were never launched from ‘VAFB SLC 4E’.
- It also shows that the vast majority of the payload only weighed less than 8000Kg

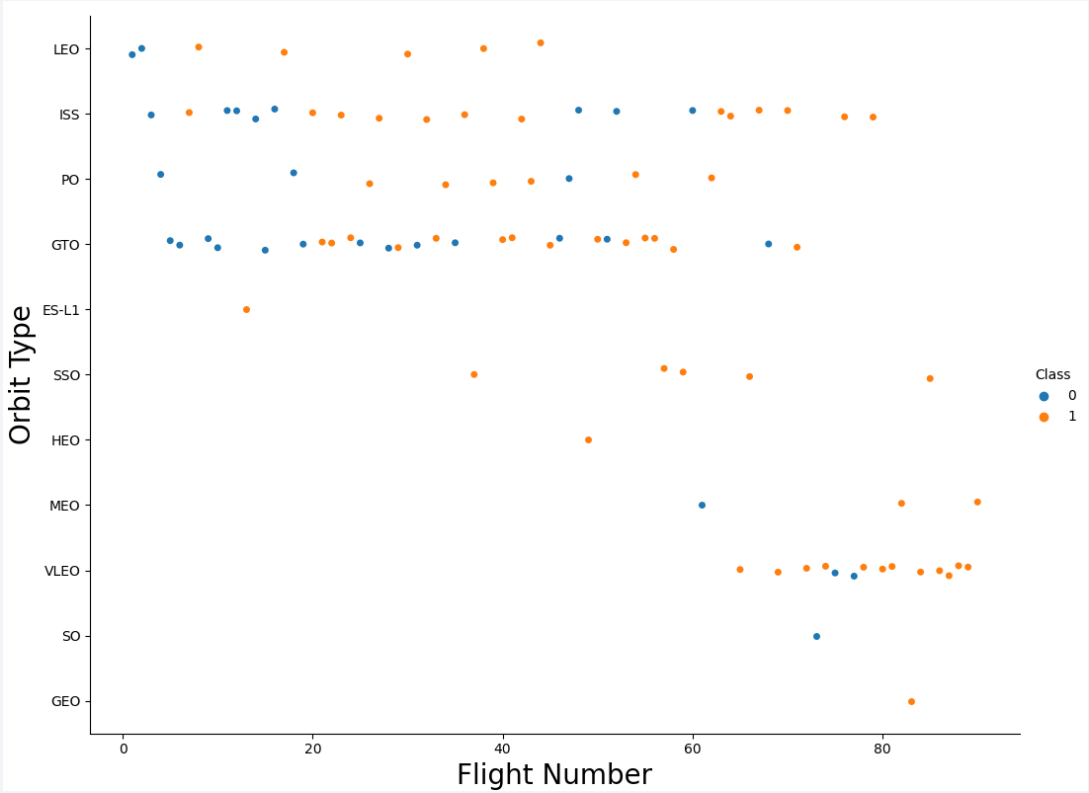
Success Rate vs. Orbit Type



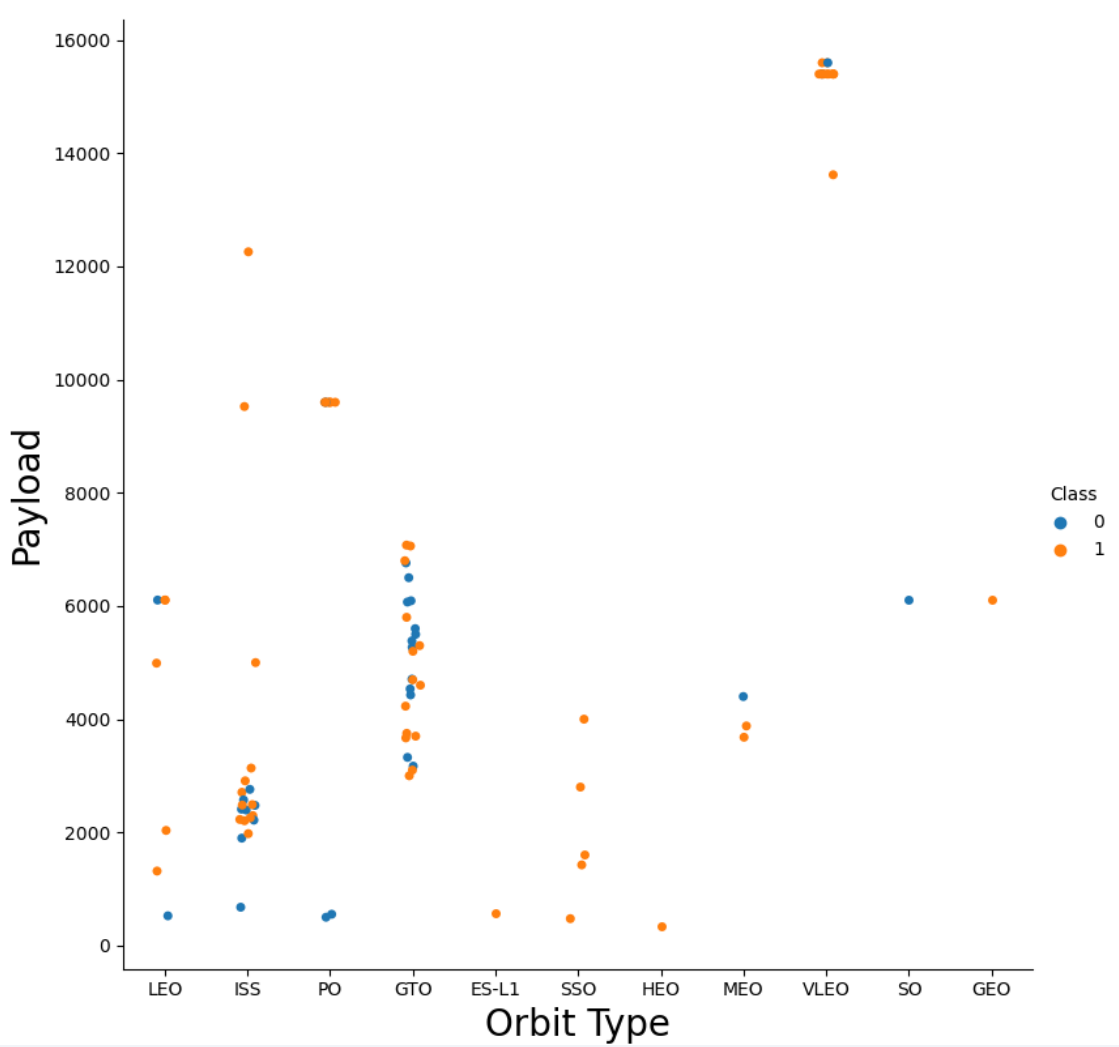
- VLEO seems to be the most successful when normalized for number of launches from the categorical plot.
- The statement regarding the success rate of VELO launches seems to hold up in this Bar Graph relating the orbit type and corresponding success rate.
- SSO mission has a 0% success rate as only 1 mission has ever taken place and that ended in a failure.

Flight Number vs. Orbit Type

- Plot tells us about the relation between the flights and its orbit types.
- From the plot its clear that most of the early efforts where put into ISS and GTO missions.
- Most of the initial launches were failures.
- SpaceX was able to consistently perform LEO missions after only 2 attempts, whereas GTO missions are more hit or miss.
- Almost no effort is put into ES-L1, SO, GEO, HEO missions. Interestingly, all those missions have 100% success rate.
- VLEO seems to be the most successful when normalized for number of launches.

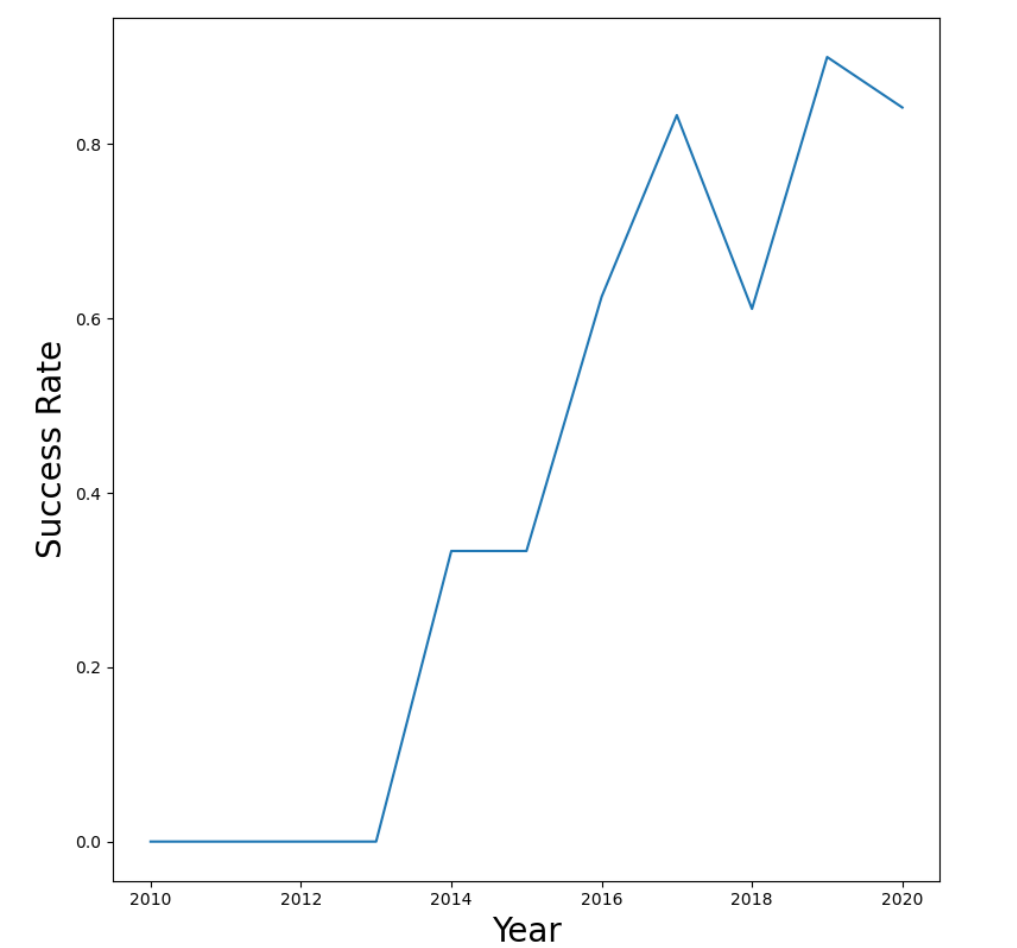


Payload vs. Orbit Type



- This is a plot that shows the relation between Orbit Type and Payload Mass.
- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- However, for GTO we cannot distinguish this well as both positive landing rate and negative landing are both there here.
- SSO missions seem to perform well with increasing mass
- ISS missions have similar payload but success rates don’t look good.
- GTO mission have wider spread of mass and just like ISS mission suffers form more failure than success.

Launch Success Yearly Trend



- This line graph shows the success rate as the years go by of SpaceX.
- It is very clear that SpaceX is on an upward trend when it comes to success rate.
- Starting from 2013, SpaceX has seen considerable amount of success and by 2016 more than half of their launches turn out to be successes

All Launch Site Names

```
%sql select distinct Launch_Site from SPACEXTBL
# This query will show the distinct launch sites from the table SPACEXTBL
```

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

Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40
None

Launch Site Names Begin with 'CCA'

```
%sql select * from SPACEXTBL where Launch_Site like 'CCA%' LIMIT 5
# This query will show 5 rows from the table SPACEXTBL where the Launch_Site begin with 'CCA'
```

MagicPython

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

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (parachute)
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No attempt
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No attempt
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
%sql select sum(PAYLOAD_MASS__KG_) as 'Total_Payload_Mass_NASA(CRS)' from SPACEXTBL where Customer="NASA (CRS)"
# This query will show the sum of the Payload Mass for NASA (CRS) from the table SPACEXTBL
```

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

Total_Payload_Mass_NASA(CRS)
45596.0

Average Payload Mass by F9 v1.1

```
%sql select avg(PAYLOAD_MASS__KG_) as 'Average payload mass carried by booster version F9 v1.1'  from SPACEXTBL where
Booster_Version="F9 v1.1"
# This query will show the average payload mass carried by booster version F9 v1.1 from the table SPACEXTBL
```

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

Average payload mass carried by booster version F9 v1.1
2928.4

First Successful Ground Landing Date

```
%sql SELECT MIN (DATE) AS "First Successful Landing" FROM SPACEXTBL WHERE LANDING_OUTCOME = 'Success (ground pad)'
# This query will show the first successful landing from the table SPACEXTBL
```

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

First Successful Landing
01/08/2018

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT BOOSTER_VERSION as 'Boosters having payload mass greater than 4000 but less than 6000' FROM SPACEXTBL WHERE
LANDING_OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ <6000
# This query will show the boosters having payload mass greater than 4000 but less than 6000 from the table SPACEXTBL
```

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

Boosters having payload mass greater than 4000 but less than 6000
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

```
%sql SELECT sum(case when MISSION_OUTCOME LIKE '%Success%' then 1 else 0 end) AS "No. of Successful Missions", \
sum(case when MISSION_OUTCOME LIKE '%Failure%' then 1 else 0 end) AS "No. of Failed Missions" FROM SPACEXTBL
# This query will show the number of successful and failed missions from the table SPACEXTBL
```

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

No. of Successful Missions	No. of Failed Missions
100	1

Boosters Carried Maximum Payload

```
%sql SELECT DISTINCT BOOSTER_VERSION AS "Booster Versions that have carried the Maximum Payload Mass" FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ =(SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
# This query will show the booster versions that have carried the maximum payload mass from the table SPACEXTBL
```

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

Booster Versions that have carried the Maximum Payload Mass
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

```
%sql SELECT substr(Date, 4, 2) as 'Month', BOOSTER_VERSION, LAUNCH_SITE FROM SPACEXTBL WHERE substr(Date,7,4)='2015' group by substr
(Date, 4, 2)
# This query will show the month, booster version and launch site from the table SPACEXTBL where the year is 2015
```

MagicPy

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

Month	Booster_Version	Launch_Site
02	F9 v1.1 B1014	CCAFS LC-40
04	F9 v1.1 B1015	CCAFS LC-40
06	F9 v1.1 B1018	CCAFS LC-40
10	F9 v1.1 B1012	CCAFS LC-40
11	F9 v1.1 B1013	CCAFS LC-40
12	F9 FT B1019	CCAFS LC-40

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

```
%sql SELECT LANDING_OUTCOME as "Landing Outcome", COUNT(LANDING_OUTCOME) AS "Total Count" FROM SPACEXTBL \
WHERE substr(Date,7,4)||substr(Date,4,2)||substr(Date,1,2) \
between '20106104' and '20170320' \
GROUP BY  LANDING_OUTCOME \
ORDER BY COUNT(LANDING_OUTCOME) DESC ;

# This query will show the landing outcome and total count from the table SPACEXTBL where the date is between 2010-04-06 and
2017-03-20
```

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

Landing Outcome	Total Count
No attempt	10
Success (ground pad)	5
Success (drone ship)	5
Failure (drone ship)	5
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1

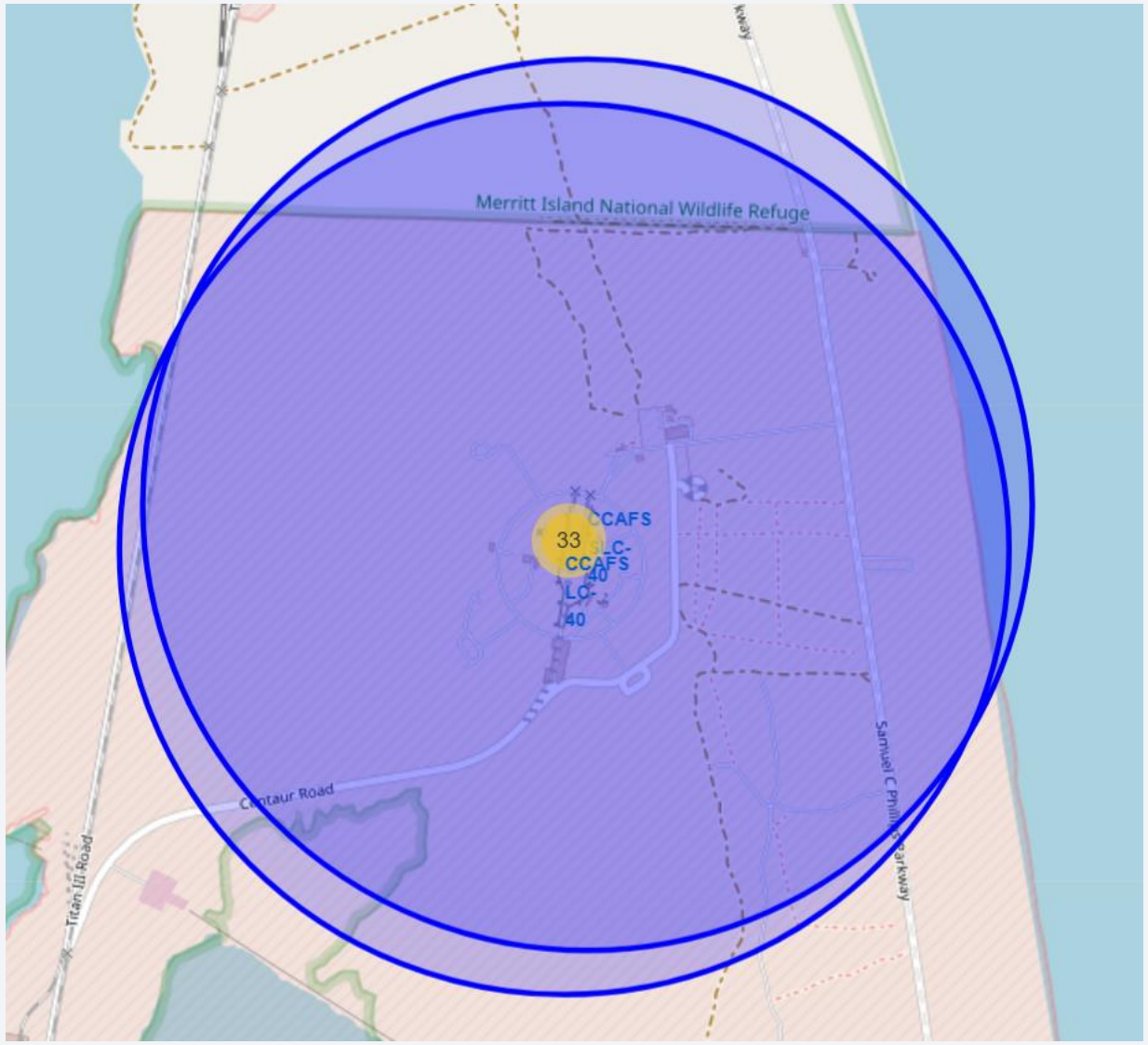
Section 3

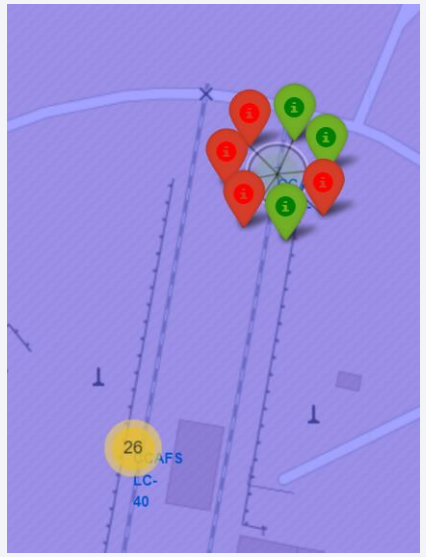
Launch Sites Proximities Analysis

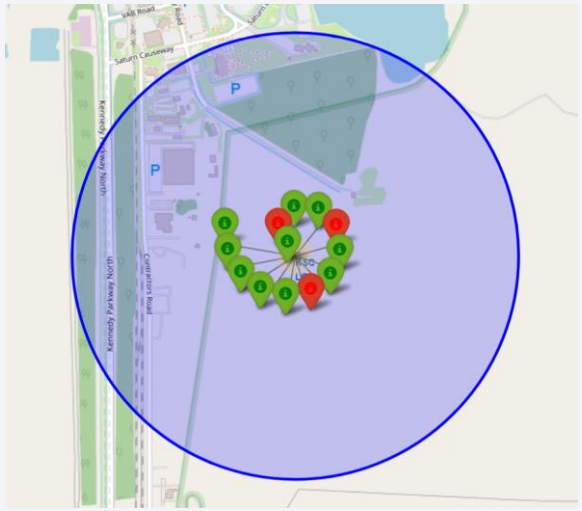
Location of all the Launch Sites

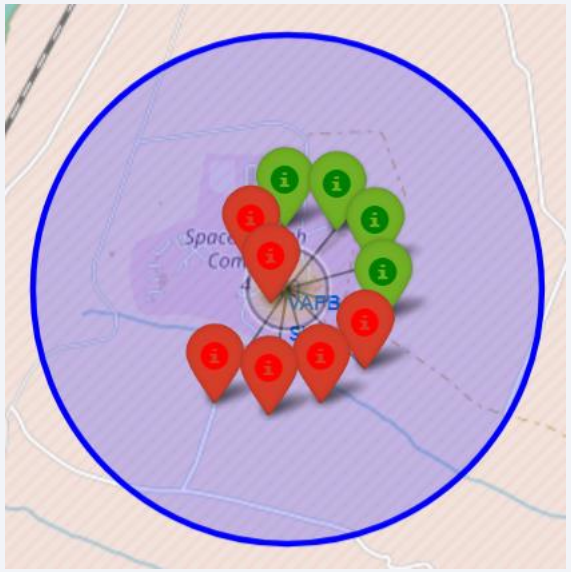
Location chosen seems to be at the southern side of USA close to the equator.

Markers showing success/failed launches for each site on the map

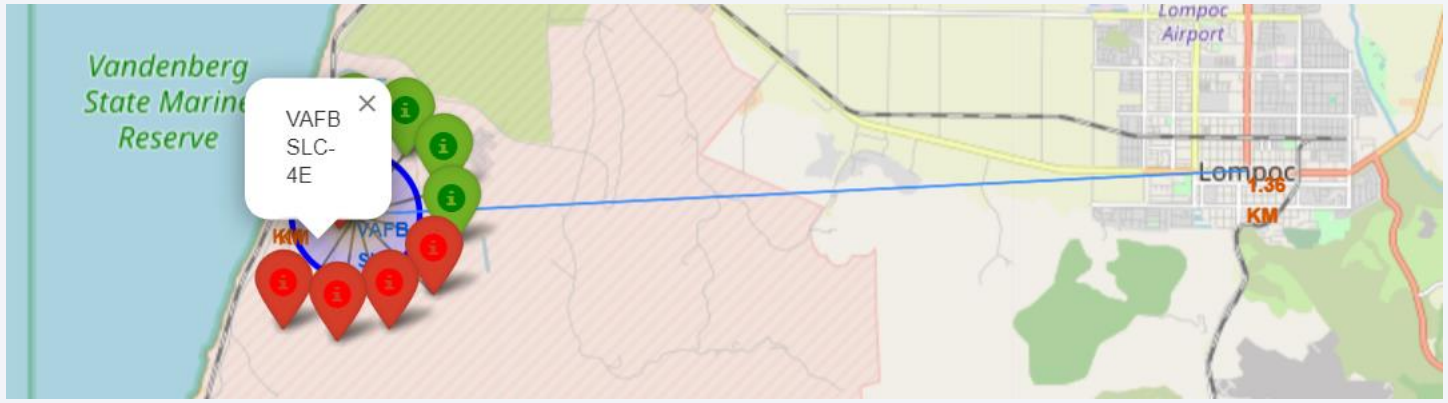








Distances between a launch site to its proximities

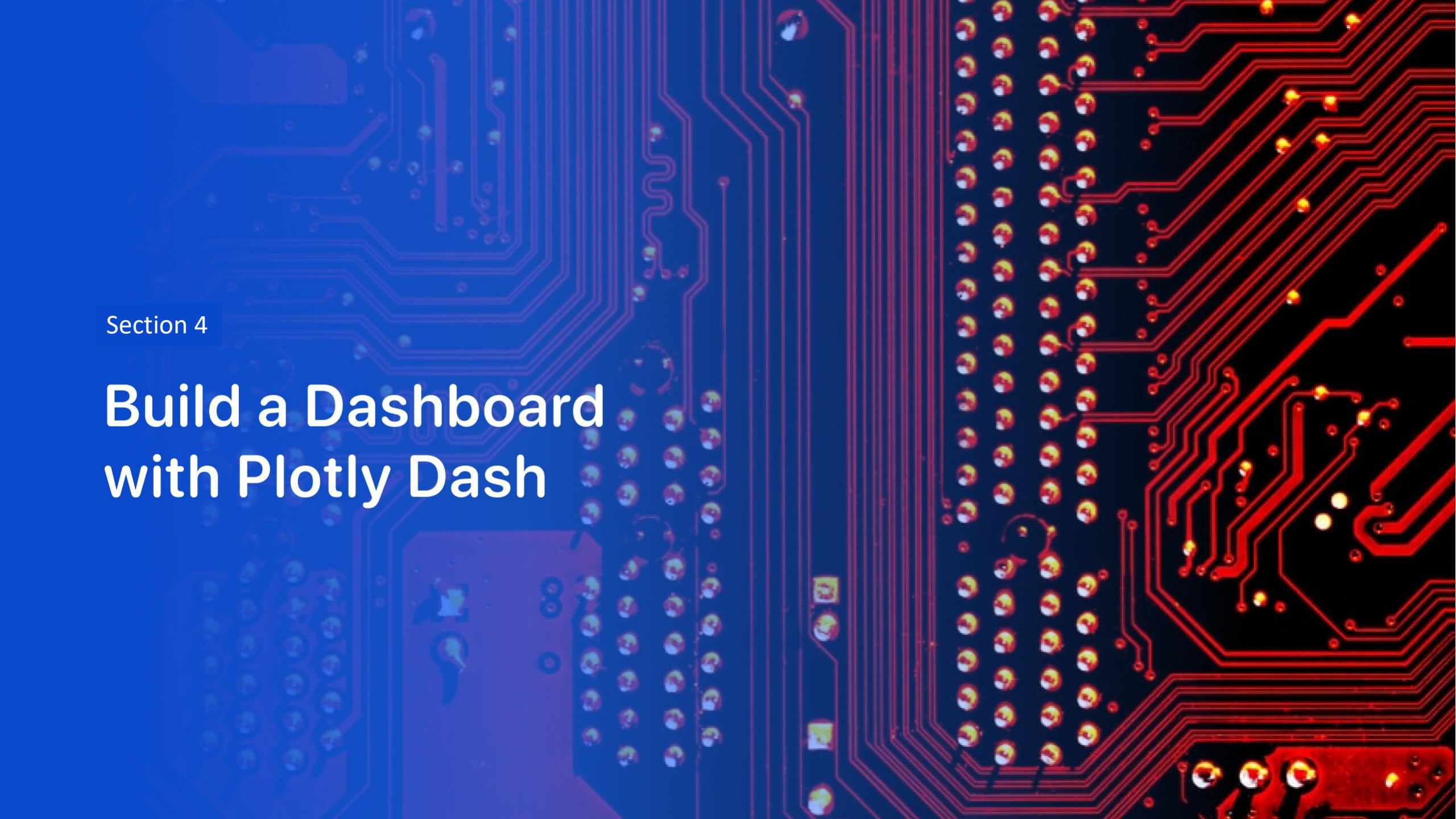


Lompac is the closes city to **'Vandenberg AFB Space Launch Complex 4'**

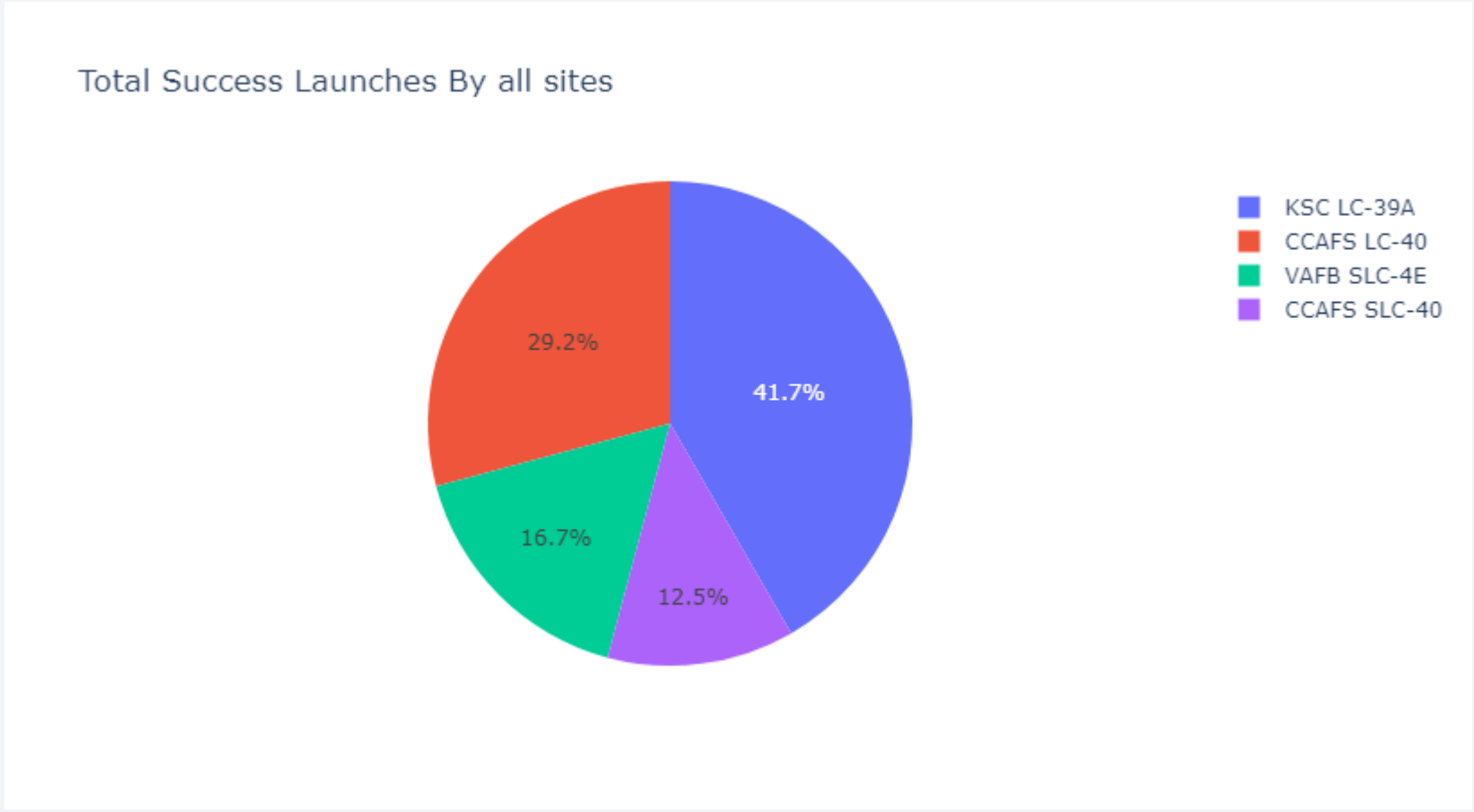
The highway **CA1** runs through Lompac and is hence the nearest high way.

Both those features are 1.36Km away from site.

The coastline is around 1Km away from the site.

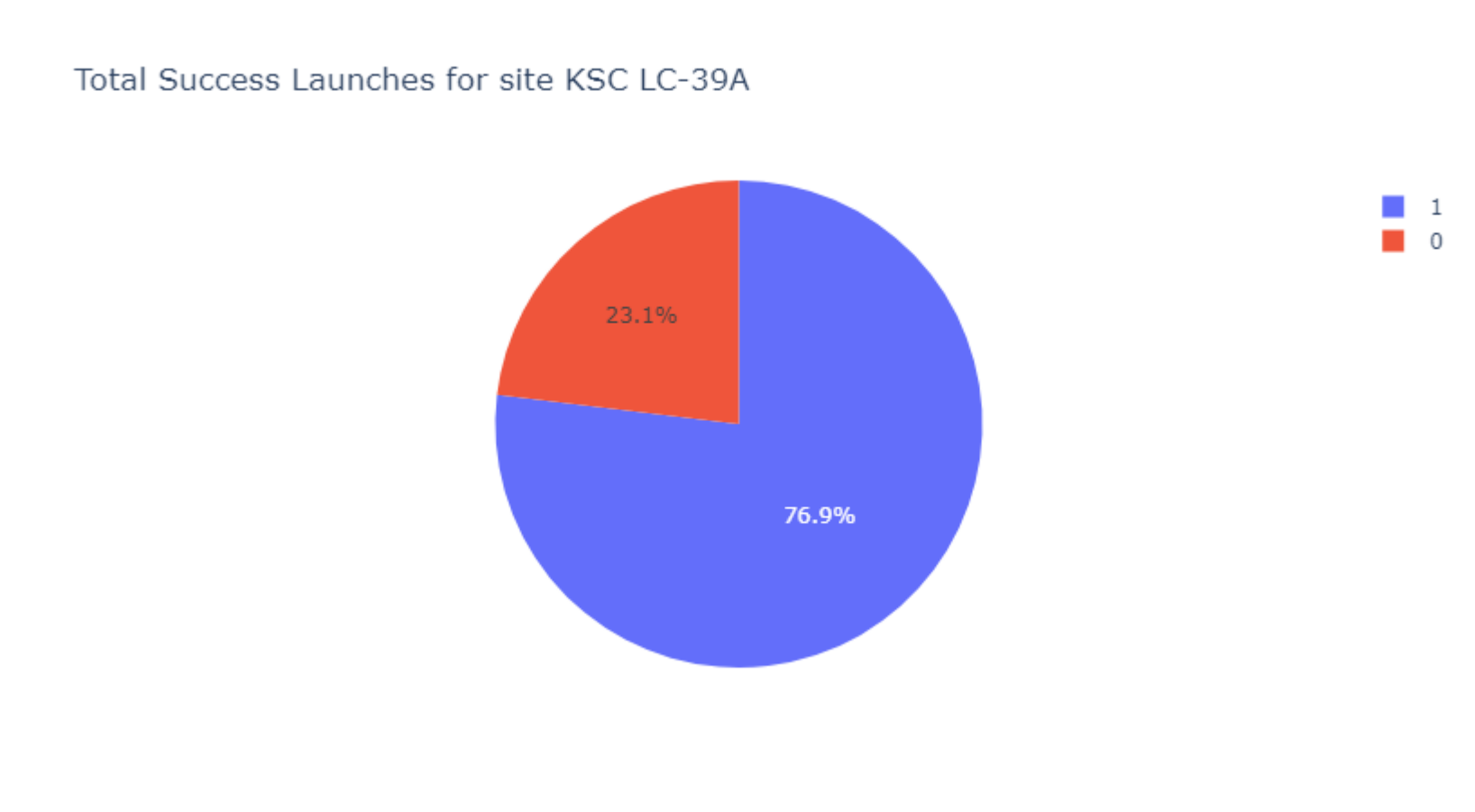

Section 4
Build a Dashboard with Plotly Dash

Success Percentage by Site



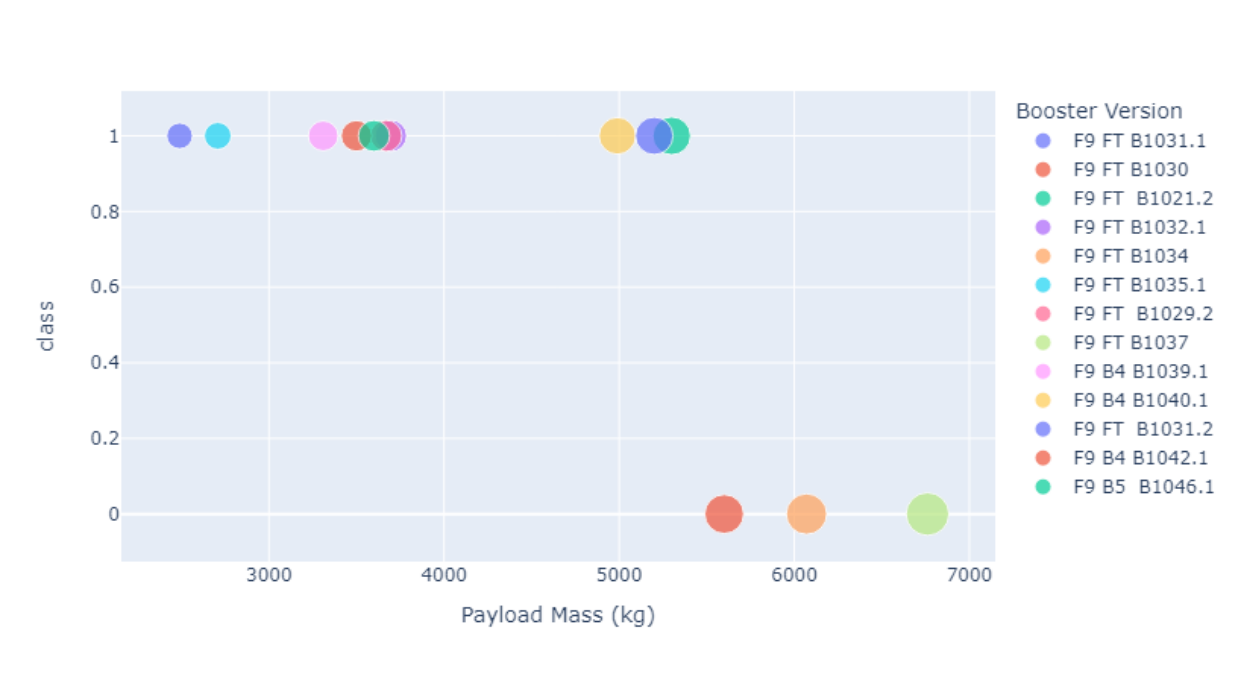
KSC LC-39A is responsible for almost half the success in SpaceX launch history.

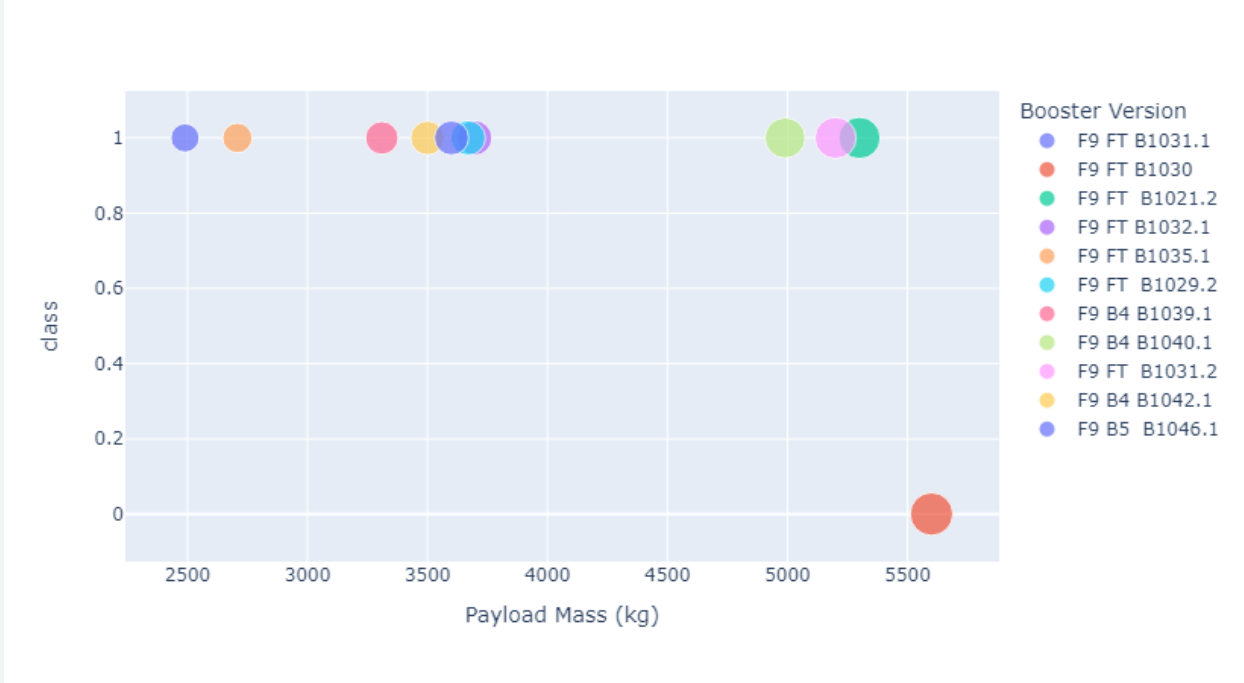
<Dashboard Screenshot 2>



76.9% of all launches from **KSC LC-39A** has been a success.

Payload vs Launch Outcome Scatter Plot

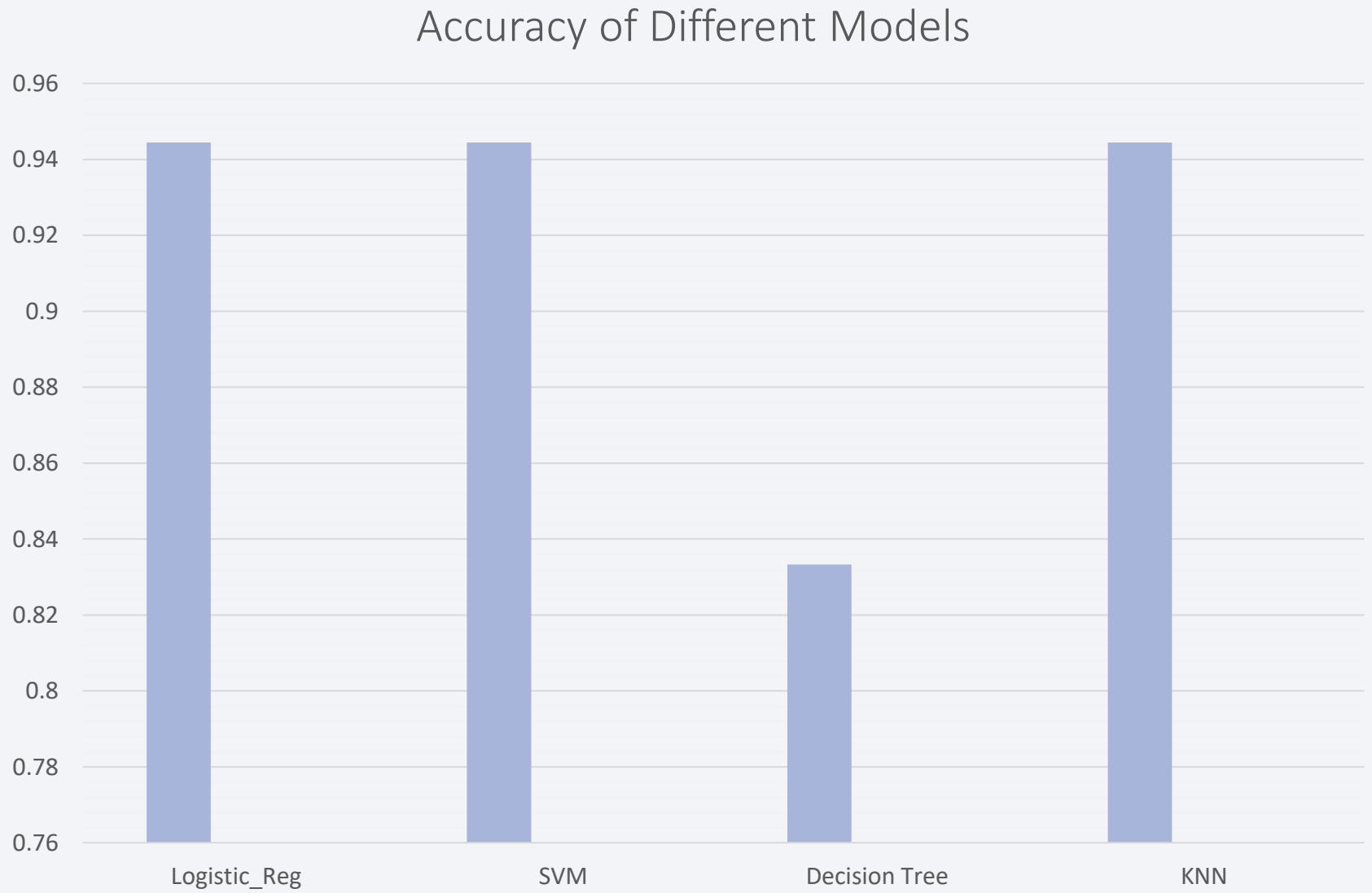




Section 5

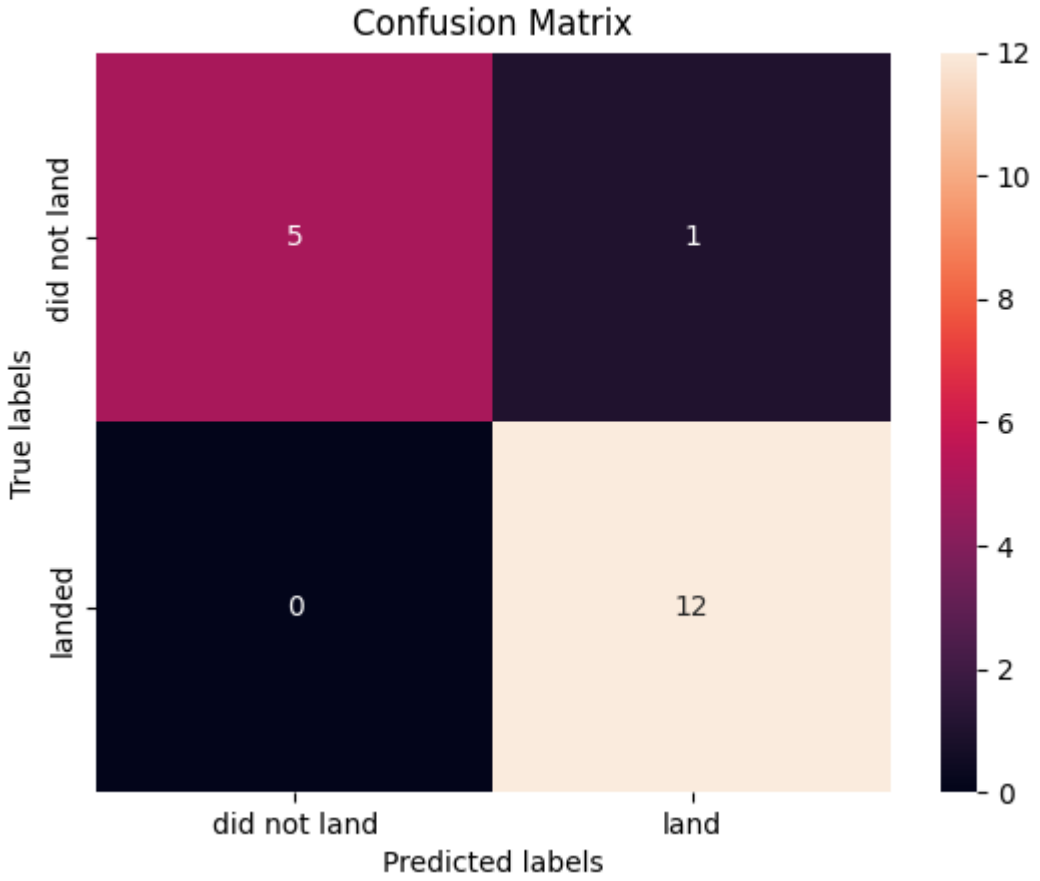
Predictive Analysis (Classification)

Classification Accuracy



- **Logistic Regression, SVM and KNN** seems to be performing the same.
- **Decision Tree** is noticeably worse.

Confusion Matrix



- Overall this model is excellent.
- There is one case of false positive which is cause for concern.

Conclusions

- We will choose LogReg, SVM or KNN as model for Machine Learning
- Heavier Payload seems to have more success rate.
- KSC LC-39A have the most successful launches of any sites; 76.9%
- SSO orbit have the most success rate

Thank you!