IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Soumya Kanti Guha
25 September 2021

Outline

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

Executive Summary

Methodologies

Data Collection
- Data collected through SpaceX APIs
- Historical launch records are collected using Web-scrapping from 'List of Falcon 9 and Falcon Heavy launches' Wikipedia page

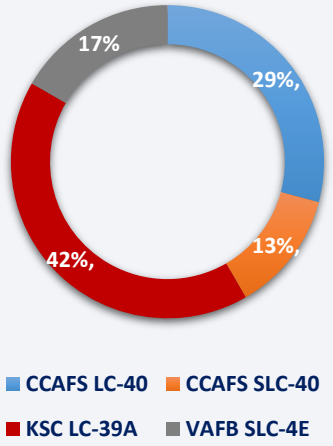
Exploratory Data Analysis
- EDA done to convert different mission outcomes to 1 and 0 labels
- Understand the relation among features (launch site, paly-load, booster version etc.) and mission success/failure using SQL
- Visualizing relation among features and feature engineering

Interactive Visualization
- Mark the success/failed launches for each site on the map and calculate distances between a launch site to its proximities
- Exploration of success rate among launch sites
- Correlation analysis between Payload and Booster Version

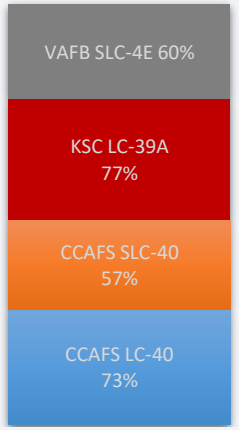
Predictive Model Development and Validation
- Load Dataframe and Define Auxiliary Function
- Data pre-processing and Train and Test Data split
- Model Development and Hyper-parameters optimization
- Validation and Model Performance Analysis

Result Summary

Largest Successful Launches



Launch Success Rate



- KSC LC-39A has largest saucerful launches (42%)among all sites
- KSC LC-39A is also having highest launch success rate (77%)

1900-5300	is the Payload Mass Range for most of the successful launches
FT	is the F9 Booster version recorded highest success rate
94%	is the best Accuracy Score achieved by Decision Tree Algorithm

Introduction

Background

Space Tourism are very much in vogue just now. SpaceX made history as the first private company to send humans into space during May 2020[1]. Companies are trying to make various affordable space travel opportunities for everyone. **Virgin Galactic**, **Rocket Lab** and **Blue Origin** are the frontrunners in this space travel industry. Virgin Galactic is providing suborbital spaceflights. Rocket Lab is a small satellite provider. Blue Origin manufactures sub-orbital and orbital reusable rockets. **SpaceX** is the leader and most successful player in this game. SpaceX's accomplishments varies from sending spacecraft to the International Space Station, providing satellite Internet access through Starlink, a satellite internet constellation. Sending manned missions to Space etc.

SpaceX sends **Falcon 9** rockets to space with relatively cheaper cost of **62 million dollars**, whereas competitors spend **165 million dollars** each, much of the savings is because SpaceX can reuse the first stage. Unlike other rocket providers, SpaceX's Falcon 9 Can recover the first stage. Sometimes the first stage does not land. The price of the successful space mission is heavily depended on the outcome of first stage reuse.

Problem Statement

Competitors of SpaceX wanted to know if the Falcon 9 first stage will land successfully. **Therefore, if we can predict weather the first stage will land, we can determine the cost of a launch**. Instead of using rocket science to determine if the first stage will land successfully or not, we will train a machine learning model and use the information available in public domain to predict if SpaceX will reuse the first stage.

Section 1

Methodology

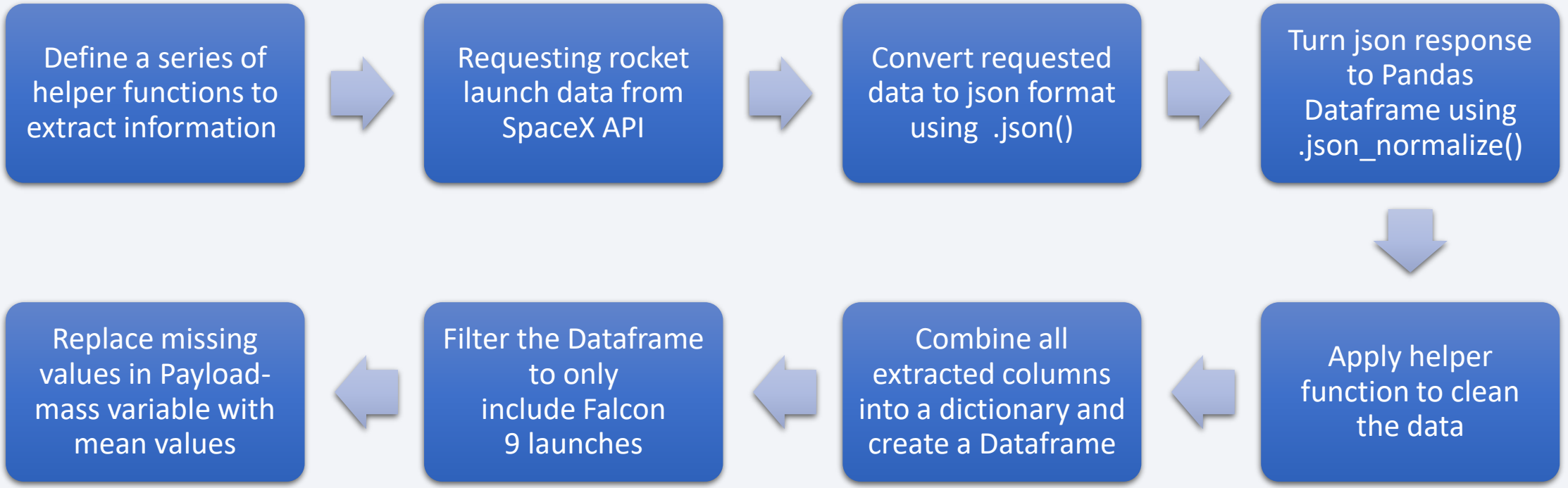
Methodology

Executive Summary

- Data collection methodology:
  - Launch Data Collection through SpaceX REST API
  - Web-scrapping launch records from Wikipedia
- Data wrangling
  - Determine Target Variable
- Exploratory data analysis (EDA) using visualization and SQL
  - Missing Value Treatment and Data Engineering
- Interactive visual analytics using Folium and Plotly Dash
  - Analyze Geospatial Data and developed site wise Dashboards
- Perform predictive analysis using classification models
  - Developed multiple Machine Learning Algorithms on train data, tune hyperparameters and validated using test data

Data Collection – SpaceX API

Objective: In this lab, we use SpaceX REST API to pull data about Falcon 9 Rocket launches, booster version, payload delivered, launch specifications, landing specifications, and landing outcome.



GitHub Link

Data Collection – SpaceX API (Continued…)

1. Requesting rocket launch data from SpaceX API

```
[6]  1 spacex_url="https://api.spacexdata.com/v4/launches/past"
```

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

2. Convert requested data to json format and turn json response to Pandas Dataframe

```
1 # Use json_normalize meethod to convert the json result into a dataframe
2 jsonresponse = response.json()
3 data = pd.json_normalize(jsonresponse)
```

3. Apply helper function to clean the data

```
1 getBoosterVersion(data)
2 getLaunchSite(data)
3 getPayloadData(data)
4 getCoreData(data)
```

4. Combine all extracted columns into a dictionary and create a Dataframe

```
1 launch_dict = {'FlightNumber': list(data['flight_number']),
2 'Date': list(data['date']),
3 'BoosterVersion':BoosterVersion,
4 'PayloadMass':PayloadMass,
5 'Orbit':Orbit,
6 'LaunchSite':LaunchSite,
7 'Outcome':Outcome,
8 'Flights':Flights,
9 'GridFins':GridFins,
10 'Reused':Reused,
11 'Legs':Legs,
12 'LandingPad':LandingPad,
13 'Block':Block,
14 'ReusedCount':ReusedCount,
15 'Serial':Serial,
16 'Longitude': Longitude,
17 'Latitude': Latitude}
18
```

```
1 # Create a data from launch_dict
2 df_launch = pd.DataFrame(launch_dict)
```

5. Filter the Dataframe to only include Falcon 9 launches

```
1 # Hint data['BoosterVersion']!='Falcon 1'
2 data_falcon9 = df_launch[df_launch['BoosterVersion']!='Falcon 1']
3 data_falcon9
```

6. Replace missing values in Payload-mass variable with mean values

```
1 # Calculate the mean value of PayloadMass column
2 avg_PayLoadsMass = data_falcon9['PayloadMass'].mean()
3 # Replace the np.nan values with its mean value
4 data_falcon9['PayloadMass'].replace(np.nan,avg_PayLoadsMass,inplace=True)
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Lat
4	6	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.5
5	8	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.5
6	10	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.5
7	11	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.6

Data Collection - Scraping

Objective: In this lab, we will perform web scraping to collect Falcon 9 historical launch records from a Wikipedia page List of Falcon 9 and Falcon Heavy launches and parse the data to Pandas Dataframe.

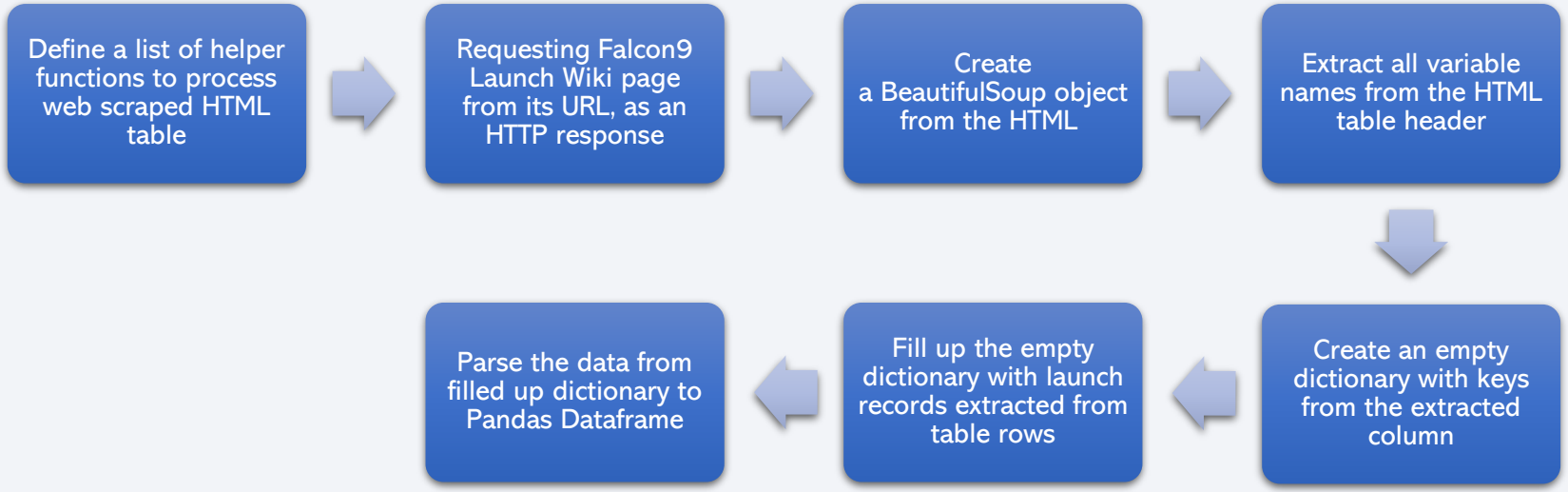


Table from List of Falcon 9 and Falcon Heavy launches Wikipedia page

2020 [edit]

In late 2019, Gwynne Shotwell stated that SpaceX hoped for as many as 24 launches for Starlink satellites in 2020,[490] in addition to 14 or 15 non-Starlink launches. At 26 launches, 13 of which for Starlink satellites, Falcon 9 had its most prolific year, and Falcon rockets were second most prolific rocket family of 2020, only behind China's Long March rocket family.[491]

[hide] Flight No.	Date and time (UTC)	Version, Booster[b]	Launch site	Payload[c]	Payload mass	Orbit	Customer	Launch outcome	Booster landing
78	7 January 2020, 02:19:21[492]	F9 B5 △ B1049.4	CCAFS, SLC-40	Starlink 2 v1.0 (60 satellites)	15,600 kg (34,400 lb)[5]	LEO	SpaceX	Success	Success (drone ship)
	Third large batch and second operational flight of Starlink constellation. One of the 60 satellites included a test coating to make the satellite less reflective, and thus less likely to interfere with ground-based astronomical observations.[493]								
	19 January 2020, 15:30[494]	F9 B5 △ B1046.4	KSC, LC-39A	Crew Dragon in-flight abort test[495] (Dragon C205.1)	12,050 kg (26,570 lb)	Sub-orbital[496]	NASA (CTS)[497]	Success	No attempt

GitHub Link

Data Collection – Scraping (Continued…)

1. Requesting Falcon9 Launch Wiki page from its URL, as an HTTP response

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

2. Create a BeautifulSoup object from the HTML

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(data, 'html5lib')
```

3. Extract all variable names from the HTML table header

```
column_names = []

# Apply find_all() function with `th` element on first_laun
# Iterate each th element and apply the provided extract_co
# Append the Non-empty column name (`if name is not None an
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if name is not None and len(name)>0:
#                 print(name)
        column_names.append(name)
```

6. Parse the data from filled up dictionary to Pandas Dataframe

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

```
<bound method NDFrame.head of     Flight No. Launch site  ...
0            1       CCAFS  ...      4 June 2010  18:45
1            2       CCAFS  ...  8 December 2010  15:43
2            3       CCAFS  ...     22 May 2012  07:44
3            4       CCAFS  ...   8 October 2012  00:35
4            5       CCAFS  ...    1 March 2013  15:10
..         ...         ...  ...              ...    ...
116        117       CCSFS  ...      9 May 2021  06:42
117        118         KSC  ...     15 May 2021  22:56
118        119       CCSFS  ...     26 May 2021  18:59
119        120         KSC  ...      3 June 2021  17:29
120        121       CCSFS  ...      6 June 2021  04:26

[121 rows x 11 columns]>
```

4. Create an empty dictionary with keys from the extracted column

```
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']=[]
# launch_dict
```

5. Fill up the empty dictionary with launch records extracted from table rows

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value
            # TODO: Append the flight_number into launch_dict with key `Flight No.`
            launch_dict['Flight No.'].append(flight_number)
            #print(flight_number)
            datatimelist=date_time(row[0])

            # Date value
            # TODO: Append the date into launch_dict with key `Date`
            date = datatimelist[0].strip(',')
            launch_dict['Date'].append(date)
            #print(date)
```

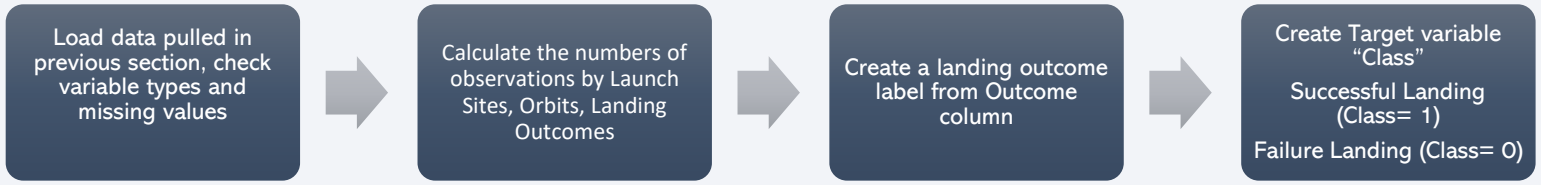
Data Wrangling

Objective: This lab will help us to explore the inherent pattern in the data and also define the label for supervised model.

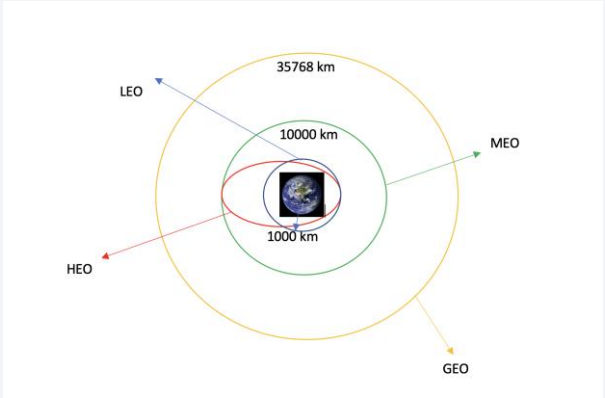
It is observed in the data that there are several different cases where the booster did not land successfully and landing attempt was failed due to an accident; for example, True Ocean means the mission outcome was successfully landed to a specific region of the ocean while False Ocean means the mission 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.

All these various landing outcomes are labeled as success = 1 and failure = 0 and created new variable “class” as target.

EDA Workflow:



Orbit:



Final Outcome:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0

GitHub Link

EDA with Data Visualization

Objective: In this lab, we will visualize the interrelation between different features and perform feature engineering.

Scatter Plot

1. **FlightNumber vs. PayloadMass:** The first stage is more likely to land successfully as flight number increases. High payload has less chances of return from the first stage
2. **Flight Number and Launch Site:** Success rate increase with Flight numbers for Rockets launched from *CCAFS LC-40* site
3. **Payload and Launch Site:** Rocket launched from *CCAFS SLC 40* with payloads mass > 10k with almost 100% success rate
4. **FlightNumber and Orbit:** *LEO* orbit the Success appears related to the number of flights
5. **Payload and Orbit type:** Heavy payloads have a negative impact on *GTO* orbits and positive on Polar, LEO and ISS orbits

Bar Plot

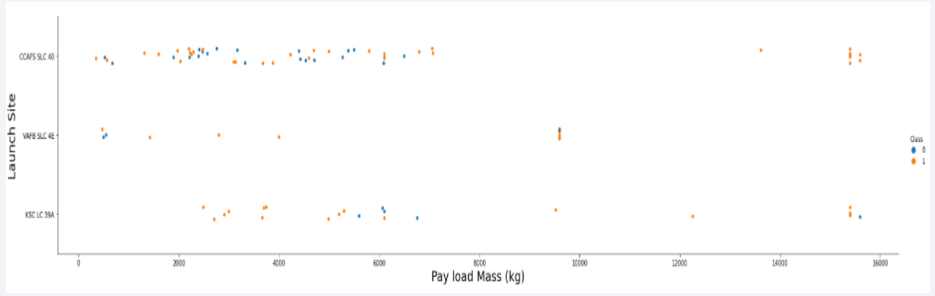
1. **Success rate of each Orbit type:** *VELO* has more than 80% success rate while *GTO* and *ISS* have around 50% and 62% success rate respectively

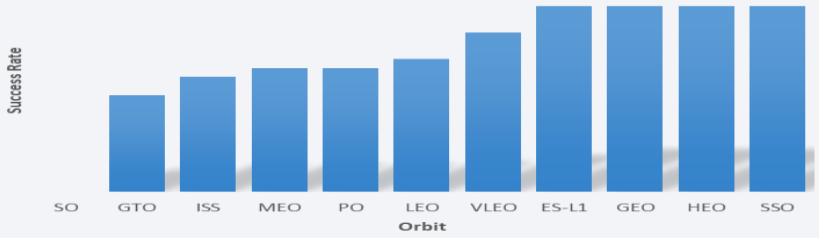
Line Plot

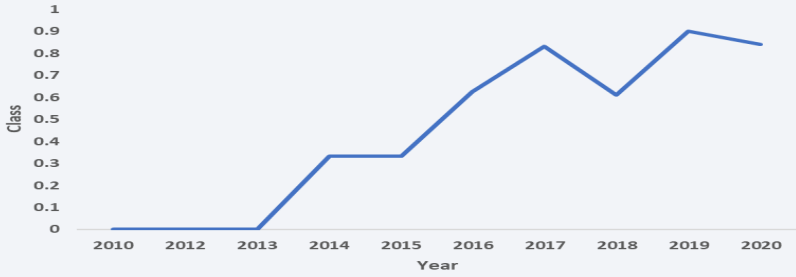
1. **Yearly Success Tend:** *Success rate since 2013 kept increasing till 2020*

Feature Engineering: Create dummy variables for *Orbit*, *LaunchSite*, *LandingPad* and *Serial* using **One Hot Encoding**

Chart Example:







GitHub Link

EDA with SQL

Objective: Load the dataset from Db2 database and execute SQL queries to understand the data

1. Display the names of the unique launch sites in the space mission
2. Display 5 records where launch sites begin with the string 'CCA'
3. Display the total payload mass carried by boosters launched by NASA (CRS)
4. Display average payload mass carried by booster version F9 v1.1
5. List the date when the first successful landing outcome in ground pad was achieved
6. List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
7. List the total number of successful and failure mission outcomes
8. List the names of the booster versions which have carried the maximum payload mass. Use a subquery
9. List the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015
10. 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

GitHub Link

Build an Interactive Map with Folium

Objective: This Lab will tries to examine some geographical patterns about launch sites.

Summary:

1. Create a map object using **folium.map** with an initial center location
2. Add launch site to map object
3. Create and add **folium.Circle** and **folium.Marker** to locate each launch site on the site map
4. Create a marker cluster object using **MarkerCluster()** method
5. Assign marker color green for successful (class = 1) and red for failed launch (class = 0)
6. Create a marker based on marker color and add it to marker cluster and create map to show successful/failed launch by sites
7. Add a mouse position on the map using **MousePosition** function to to get coordinate for a mouse over a proximities on the map
8. Draw line from proximities to launch sites using **folium.PolyLine** method

Question:

1. Are all launch sites in proximity to the Equator line? No
2. Are launch sites in close proximity to railways? No
3. Are launch sites in close proximity to highways? No
4. Are launch sites in close proximity to coastline? Yes
5. Do launch sites keep certain distance away from cities? Yes

GitHub Link

Build a Dashboard with Plotly Dash

Objective: SpaceX Launch Records Dashboard helps to understand relevant question about launch site and success rate and interrelation between payload success rate.

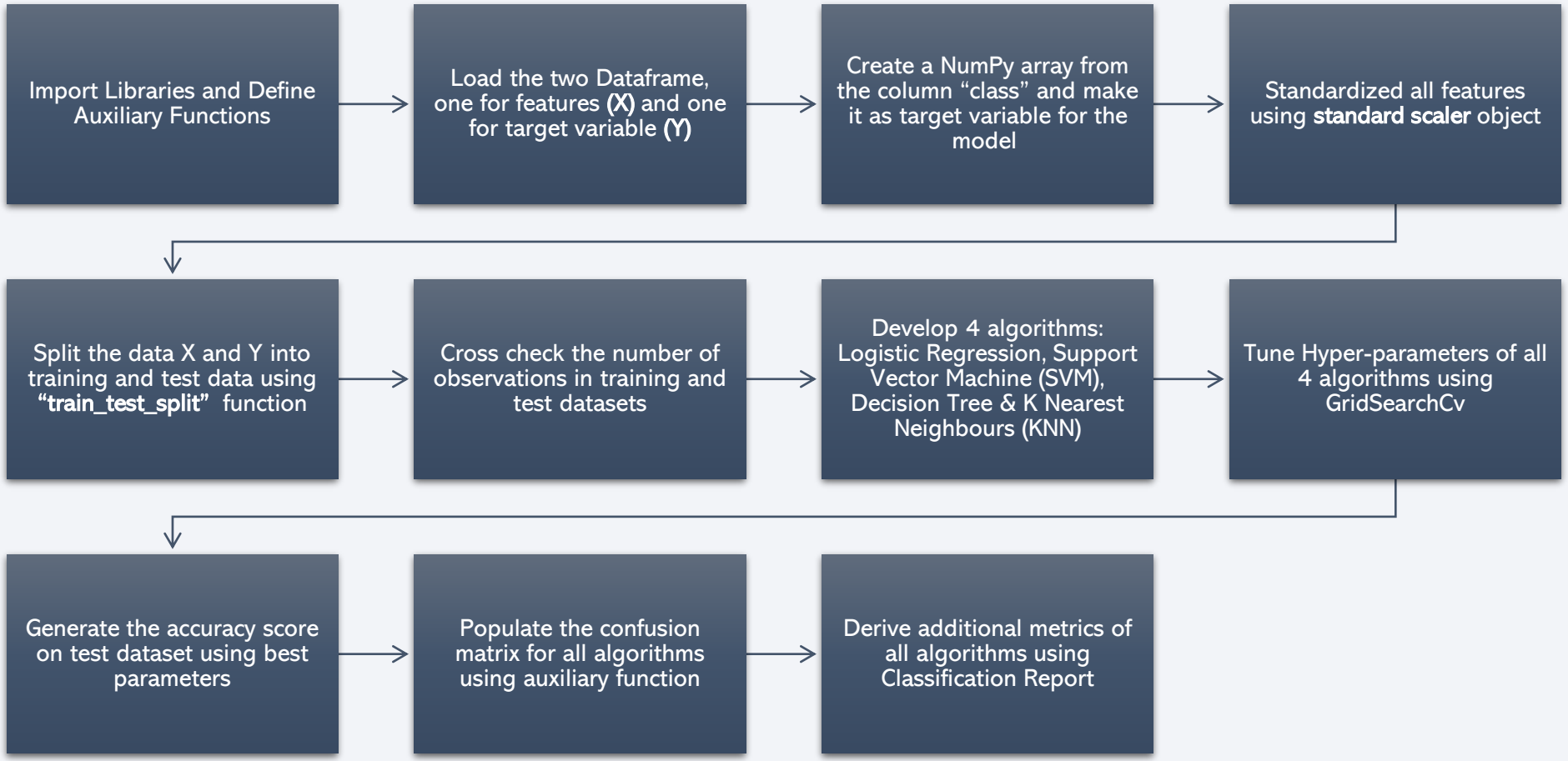
Summary:

1. Load SpaceX data and create min and max of payload for range slider
2. Initiate Dash app
3. Design Dash app layout
4. Inside Dash app layout add
    1. Dashboard title
    2. drop down list of all sites and individual sites using **dcc.Dropdown** component
    3. Pie chart using **dcc.Graph** component
    4. Range slider using **dcc. RangeSlider** component
5. Add a callback function for pie chart and incorporate a function to update pie chart for different values of drop down input
6. Add another callback for success-payload-scatter-chart and incorporate a function to update scatter plot for different values of drop down input and range slider
7. Run the app to generate the dashboard

GitHub Link

Predictive Analysis (Classification)

Objective: In this lab, we will develop a machine learning pipeline to predict if the first stage will land given the data from the preceding labs



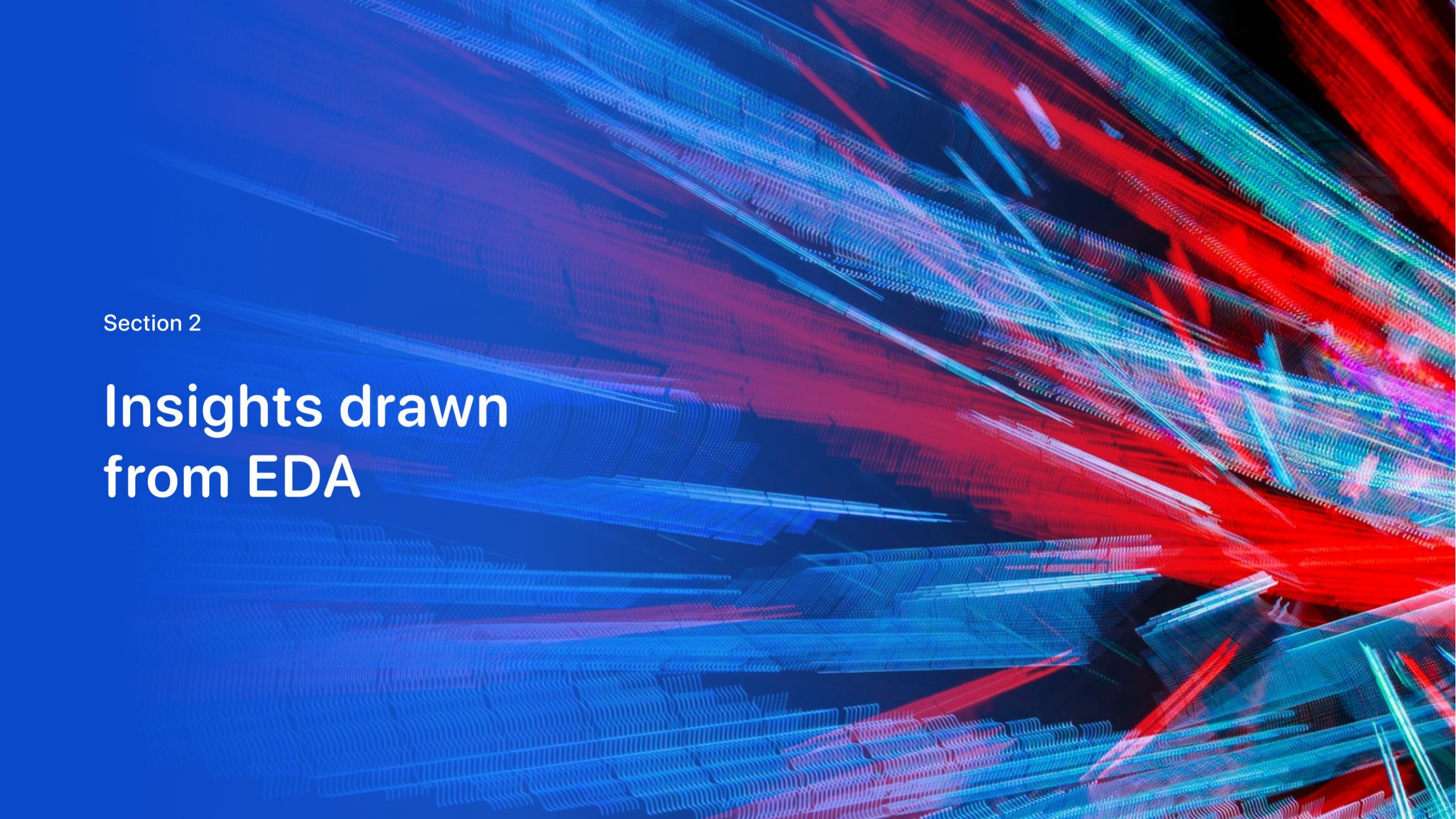
GitHub Link

Results

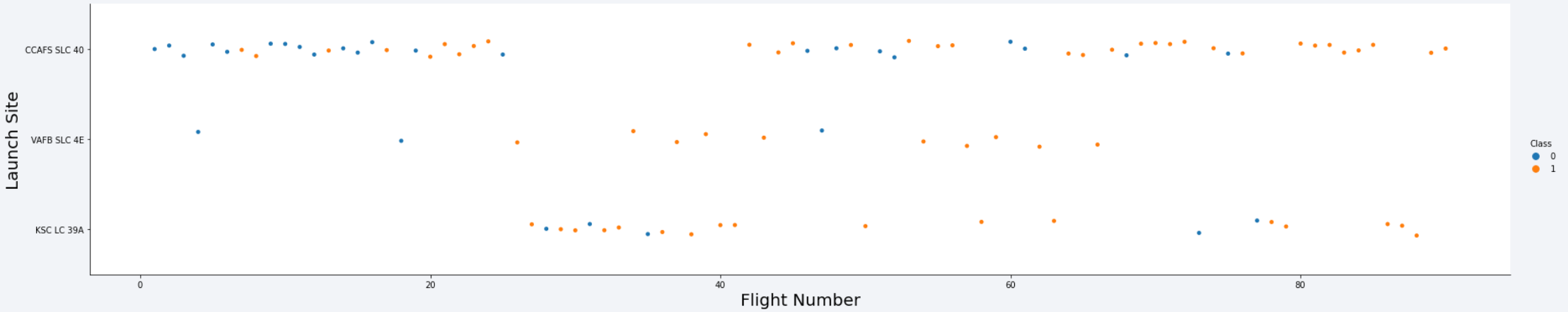
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

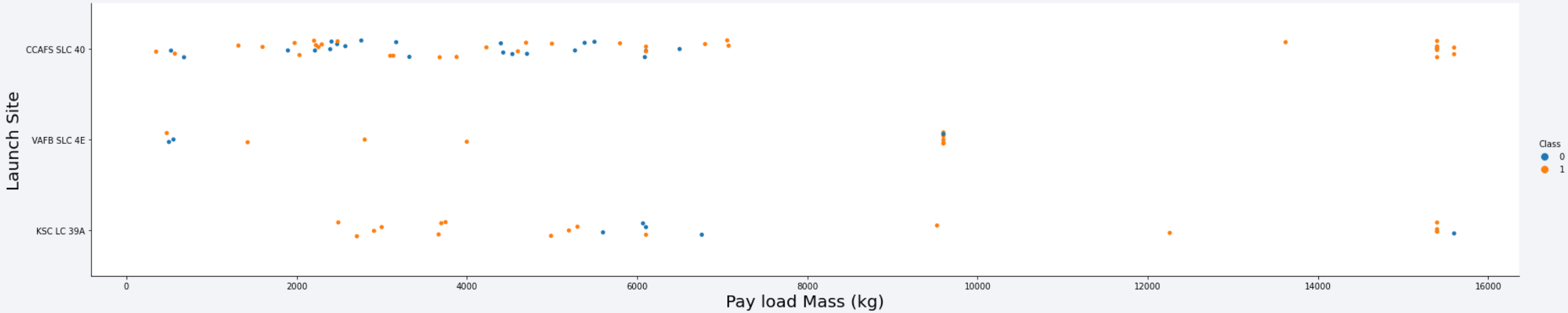


Flight Number vs. Launch Site



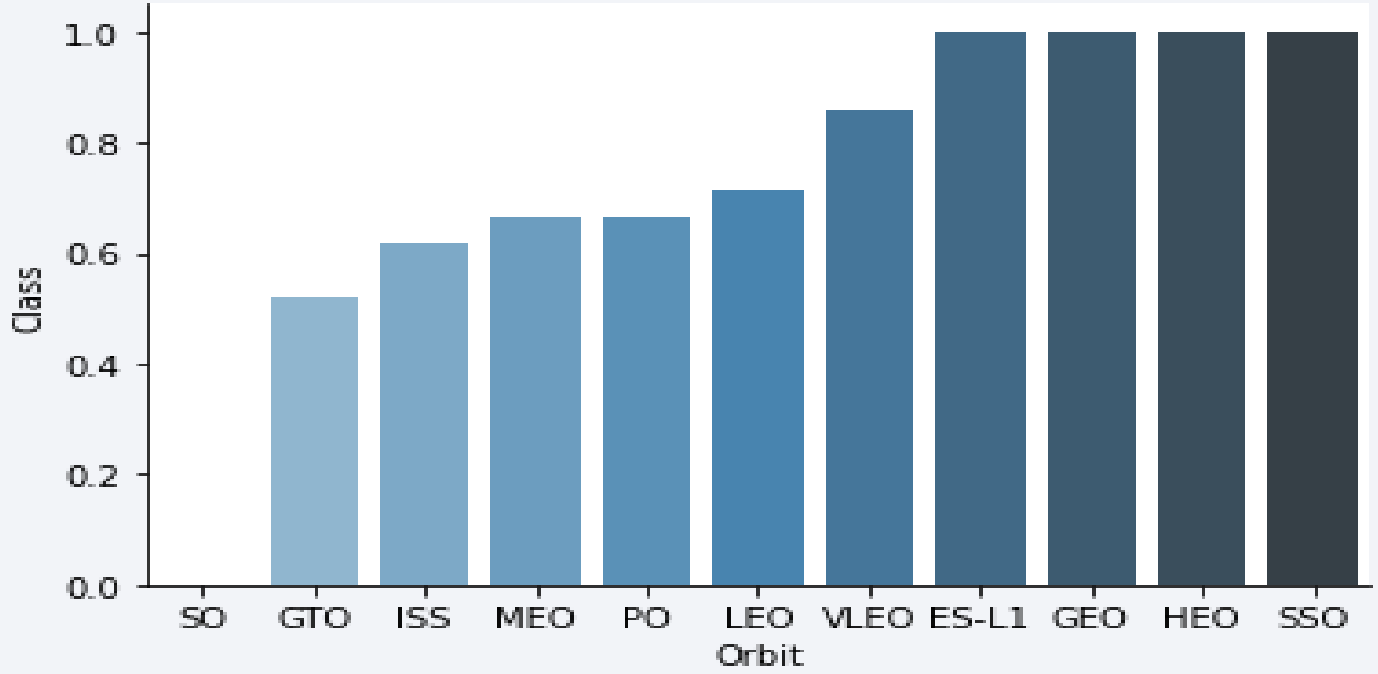
Observation: Higher flight number results in more successful launch outcome. We can clearly see less number of blue dots (unsuccessful launches) when Flight Number exceeds 30.

Payload vs. Launch Site



Observation : Rocket launched from CCAFS SLC 40 with payloads mass > 10k with almost 100% success rate. This also holds good for KSC LC 39A site when payloads mass > 8k

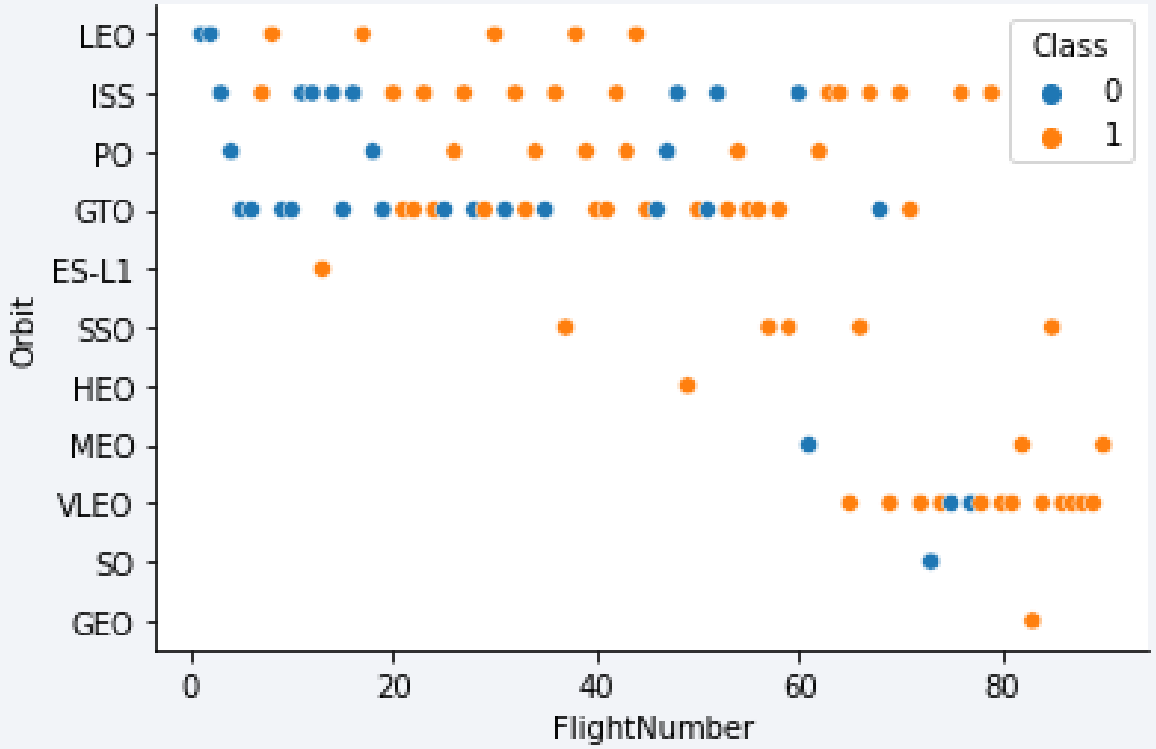
Success Rate vs. Orbit Type



Observation :

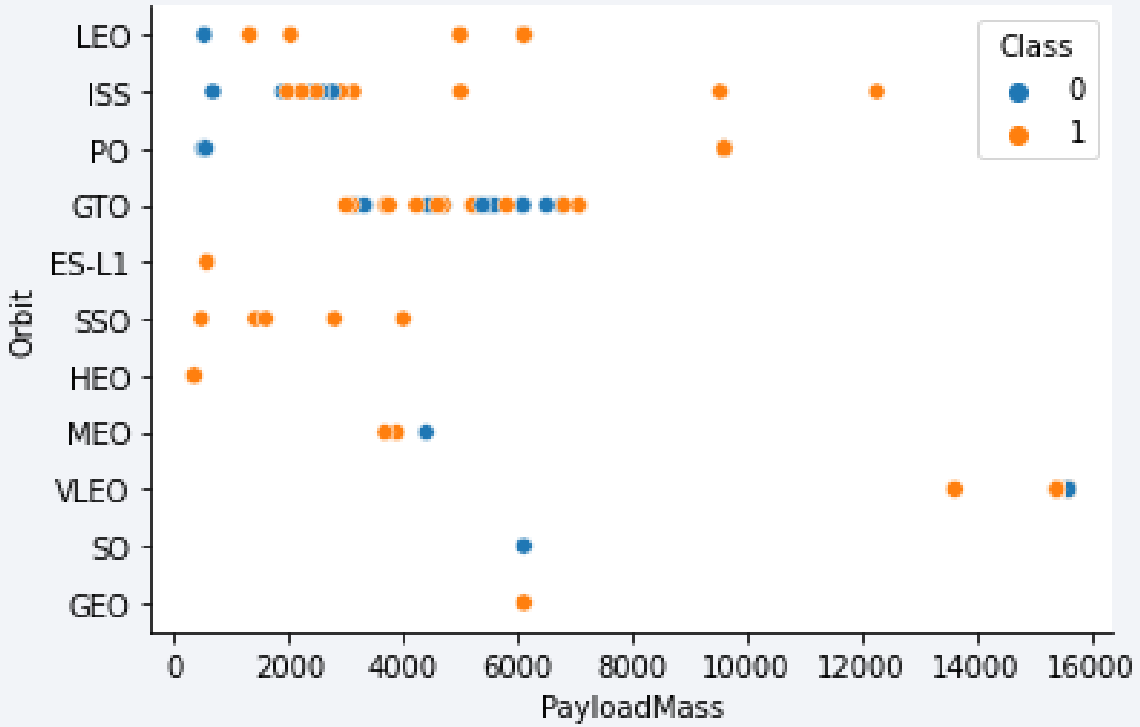
- *VELO* has more than 80% success rate while *GTO* and *ISS* have around 50% and 62% success rate respectively.
- ES-L1, GEO, HEO & SSO though have 100% success rate but with very low number of missions

Flight Number vs. Orbit Type



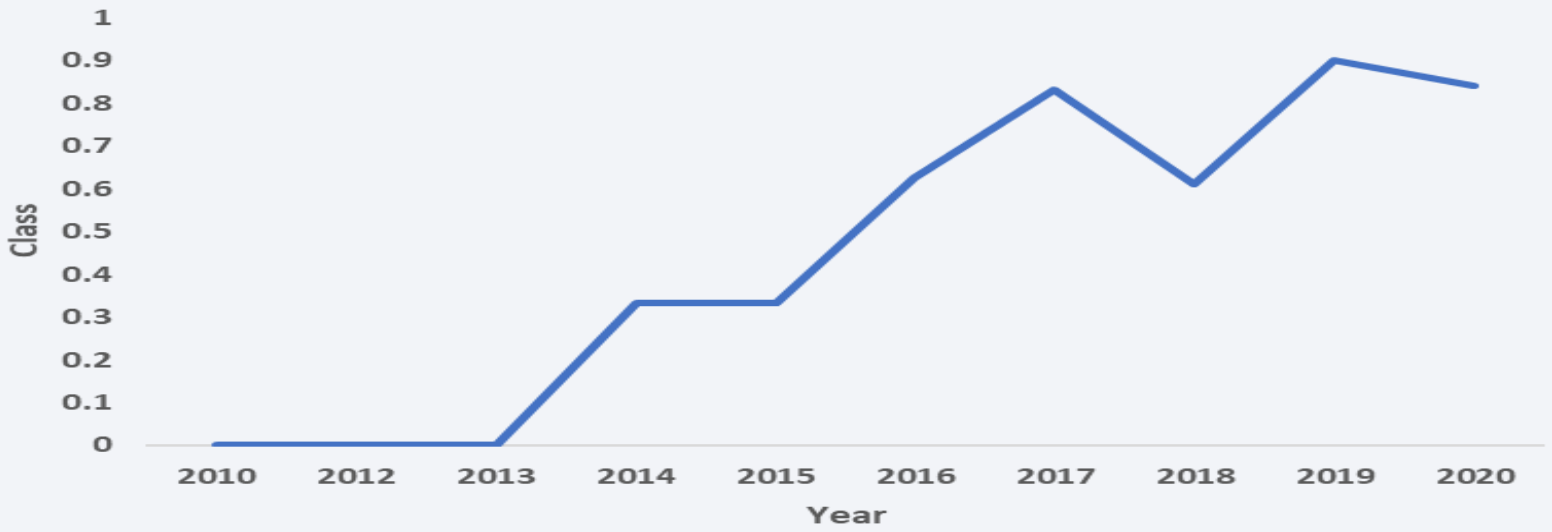
Observation : LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit

Payload vs. Orbit Type



Observation : Heavy payloads have a negative impact on *GTO* orbits and positive on Polar, LEO and ISS orbits

Launch Success Yearly Trend



Observation : Success rate is keep increasing post 2013

All Launch Site Names

There are 4 unique launch sites retrieved from “select distinct launch_site”, namely

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

SQL Query :

```
1 %sql select distinct Launch_Site from SPACEXTBL
```

Launch Site Names Begin with 'CCA'

5 records where launch sites begin with `CCA` are using below query where we are selecting top 5 rows (limit 5) all columns (select *) from SPACEXTBL where launch contains CCA ('%CCA%'')

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

SQL Query :

```
1 sql select * from SPACEXTBL where Launch_Site like '%CCA%' limit 5
```

Total Payload Mass

Total payload carried by boosters from NASA is **45596** Kg.

The below SQL query select sum of payload mass (sum(PAYLOAD_MASS__KG_)) from SPACEXTBL table and filtering “Customer” as NASA (CRS) (where Customer = 'NASA (CRS)')

SQL Query :

```
1 sql select sum(PAYLOAD_MASS__KG_) as Total_PAYLOAD_MASS__KG from SPACEXTBL where Customer = 'NASA (CRS)'
```

Average Payload Mass by F9 v1.1

Average payload mass carried by booster version F9 v1.1 is **2928 kg.**

The below SQL query select average of payload mass (avg(PAYLOAD_MASS__KG_)) from SPACEXTBL table and filtering "Booster Version" as F9 v1.1 (where Booster_Version = ' F9 v1.1 ')

SQL Query :

```
%sql select avg(PAYLOAD_MASS__KG_) as Avg_PAYLOAD_MASS__KG from SPACEXTBL where Booster_Version = 'F9 v1.1'
```

First Successful Ground Landing Date

The dates of the first successful landing outcome on ground pad is **2015-12-22.**

The below SQL query select minimum value of Date (min(Date)) from SPACEXTBL table and filtering “Landing Outcome” as Success (ground pad)' (where landing__outcome = 'Success (ground pad)')

SQL Query :

```
%sql select min(Date) from SPACEXTBL where landing__outcome = 'Success (ground pad)'
```

Successful Drone Ship Landing with Payload between 4000 and 6000

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

booster_version	landing__outcome	payload_mass__kg_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

The below SQL query select Bosster Version, Landing Outcome and Payload Mass (select Booster_Version, landing__outcome, PAYLOAD_MASS__KG_) from SPACEXTBL table and filtering “Landing Outcome” as Success Drone Ship for payload mass range 4000 to 6000 kg (where landing__outcome like '%Success (drone ship)%' and (PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_<6000))

SQL Query :

```
%%sql select Booster_Version, landing__outcome, PAYLOAD_MASS__KG_ from SPACEXTBL where landing__outcome like '%Success (drone ship)%'
and (PAYLOAD_MASS__KG_ > 4000 and PAYLOAD_MASS__KG_<6000)
```

Total Number of Successful and Failure Mission Outcomes

The total number of successful and failure mission outcomes

number_of_successful_mission	number_of_failed_mission
100	1

The below SQL query select sum of cases when mission outcome contains Success (sum(case when Mission_Outcome like '%Success%' then 1 else 0 end)) and sum of cases when mission outcome does not contains Success (sum(case when Mission_Outcome not like '%Success%' then 1 else 0 end)) from SPACEXTBL table

SQL Query :

```
%%sql
select
sum(case when Mission_Outcome like '%Success%' then 1 else 0 end) as number_of_successful_mission,
sum(case when Mission_Outcome not like '%Success%' then 1 else 0 end) as number_of_failed_mission
from SPACEXTBL;
```

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass:

The below SQL query select Bosster Version, Payload Mass from SPACEXTBL table (select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTBL) where payload mass is same as (where PAYLOAD_MASS__KG_ =) maximum payload mass derived from virtual table created by subquery ((select max(PAYLOAD_MASS__KG_) as MAX_PAYLOAD_MASS from SPACEXTBL))

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

SQL Query :

```
%%sql select Booster_Version, PAYLOAD_MASS__KG_ from SPACEXTBL
    where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) as MAX_PAYLOAD_MASS from SPACEXTBL)
```

2015 Launch Records

List the failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015:

booster_version	launch_site	DATE	landing__outcome
F9 v1.1 B1012	CCAFS LC-40	2015-01-10	Failure (drone ship)
F9 v1.1 B1015	CCAFS LC-40	2015-04-14	Failure (drone ship)

The below SQL query select Bosster Version, Launch site, Date and landing outcoem from SPACEXTBL table (%sql select Booster_Version, Launch_Site, Date, landing__outcome from SPACEXTBL) where landing outcome contains “Failure Drop Ship” and year = 2015 (where landing__outcome like '%Failure (drone ship)%' and year(Date) = 2015)

SQL Query :

```
%sql select Booster_Version, Launch_Site, Date, landing__outcome from SPACEXTBL
where landing__outcome like '%Failure (drone ship)%' and year(Date) = 2015
```

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

Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order:

The below SQL query select landing outcome, count of records, rank ordered by count of record from SPACEXTBL table (select landing__outcome, count(*) as cnt, DENSERANK() OVER(ORDER BY COUNT(*) DESC) AS ranking from SPACEXTBL) where Data ranges between 2010-06-04 and 2017-03-20 grouped by landing outcome (where Date between '2010-06-04' and '2017-03-20' group by landing__outcome)

landing__outcome	cnt	ranking
No attempt	10	1
Failure (drone ship)	5	2
Success (drone ship)	5	2
Controlled (ocean)	3	3
Success (ground pad)	3	3
Failure (parachute)	2	4
Uncontrolled (ocean)	2	4
Precluded (drone ship)	1	5

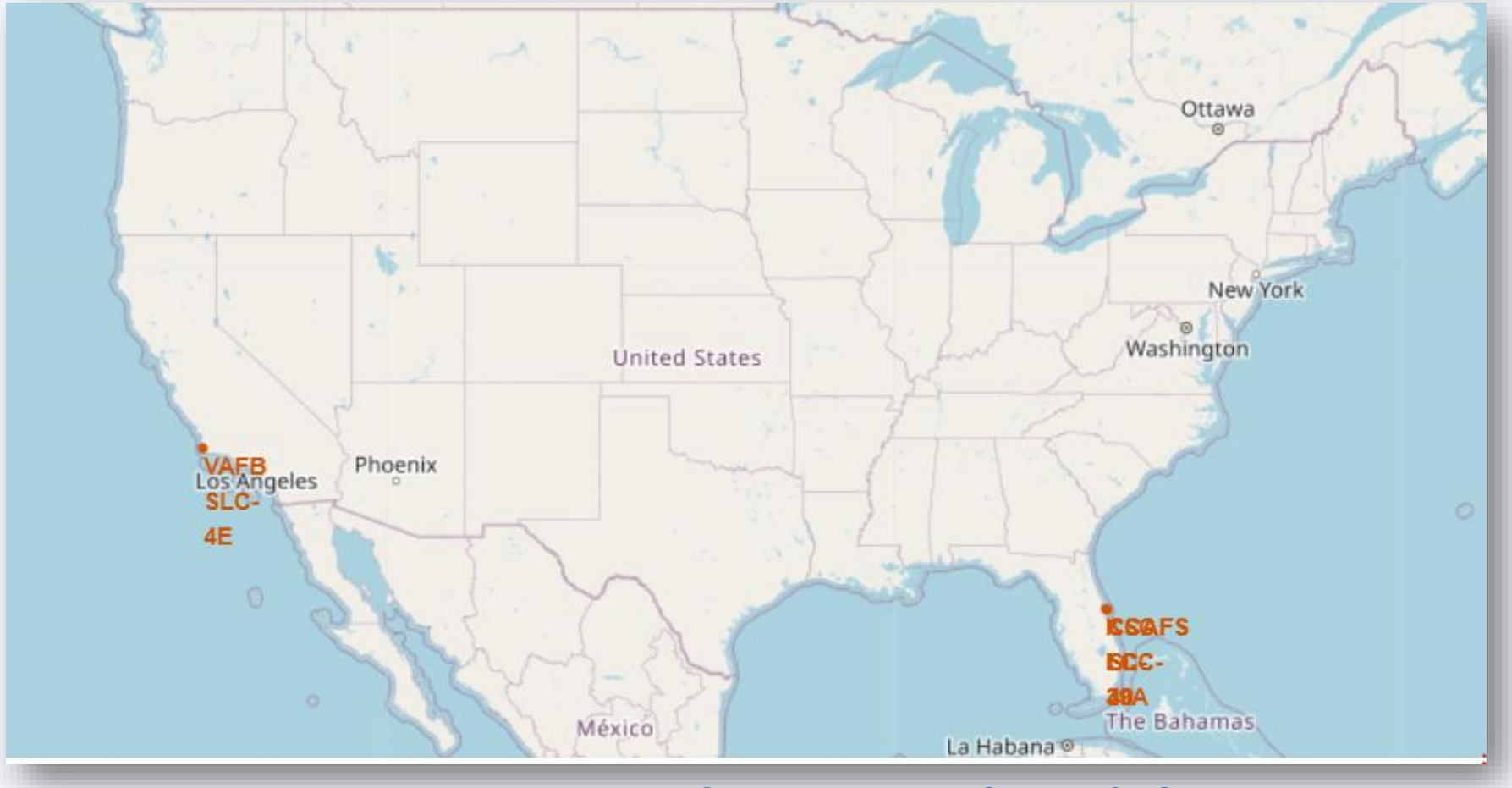
SQL Query :

```
%sql select landing__outcome, count(*) as cnt, DENSERANK() OVER(ORDER BY COUNT(*) DESC) AS ranking from SPACEXTBL
  where Date between '2010-06-04' and '2017-03-20' group by landing__outcome
```

Section 4

Launch Sites Proximities Analysis

SpaceX Launch Sites

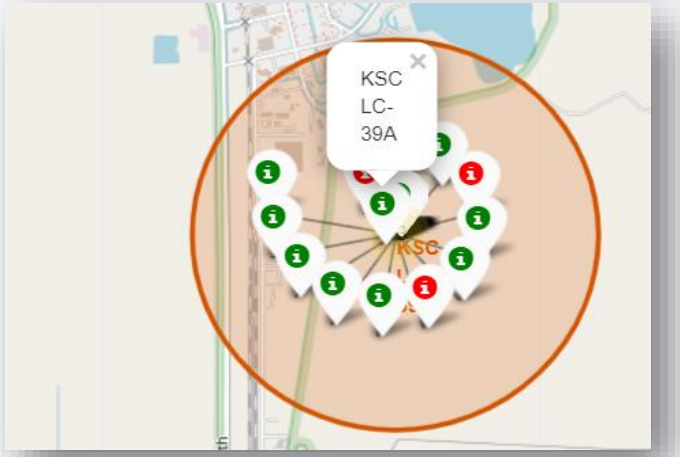


Launch sites are located primarily at East Coast and West Coast of USA in close proximity to sea.

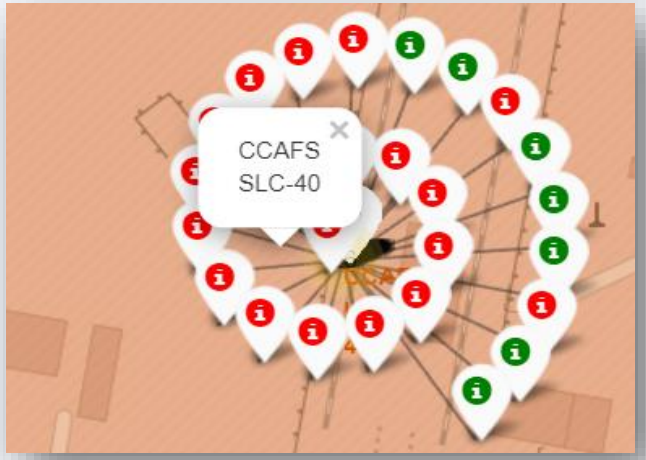
SpaceX Launch Sites: Successful and Failed Launches

East Coast

KSC LC-39A



CCAFS LC-40



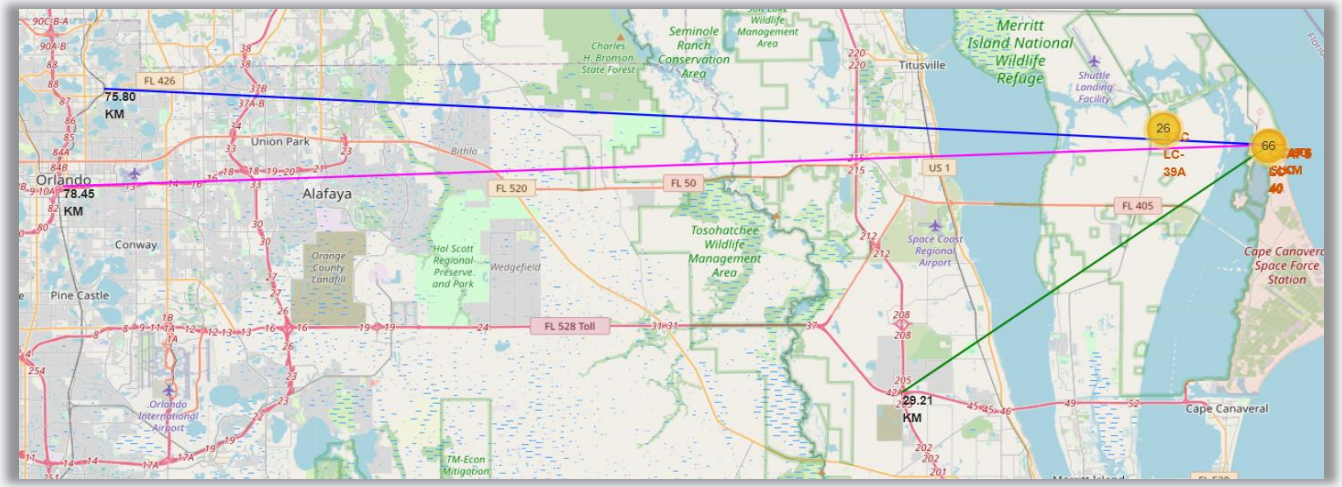
CCAFS SLC-40

West Coast

VAFB SLC-4E

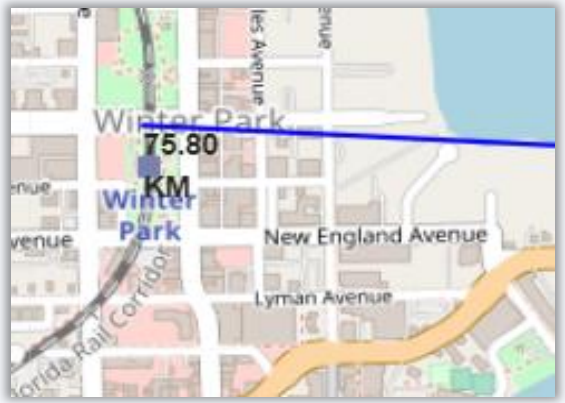
- Green icons are successful launches and Red icons are filed launches
- 3 sites are located in East Coast and 1 site is located in West Coast
- Most Successful site is KSC LC-39A

SpaceX Launch Sites: Proximity Distance to CCAFS SLC-40



- Railway Station and City are far (>70km) from the Launch Site
- Highway is 29km far away from Launch Site
- Distance from Coastline is very close, less than 1 km

Railway Station



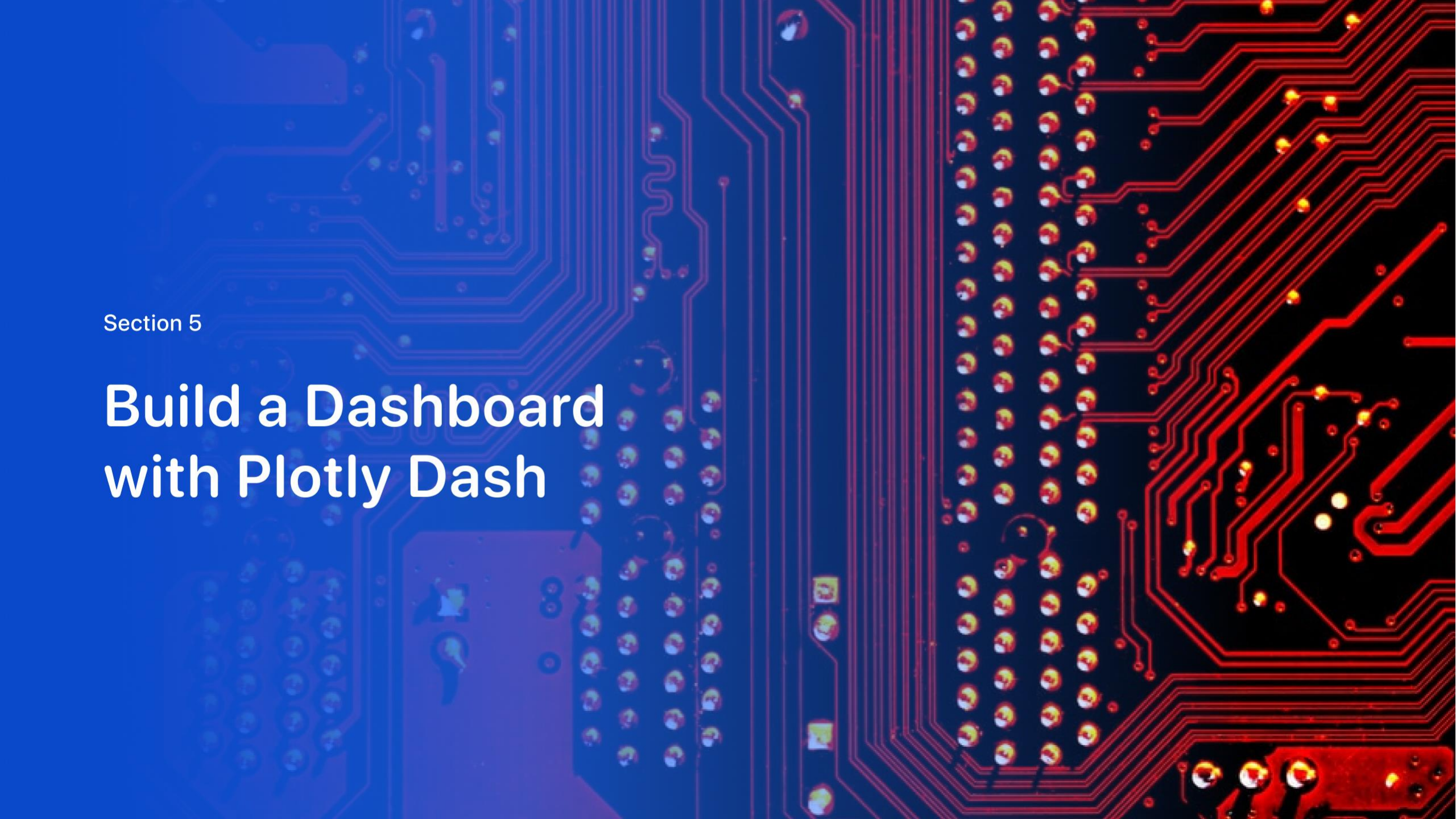
Highway

Coastline

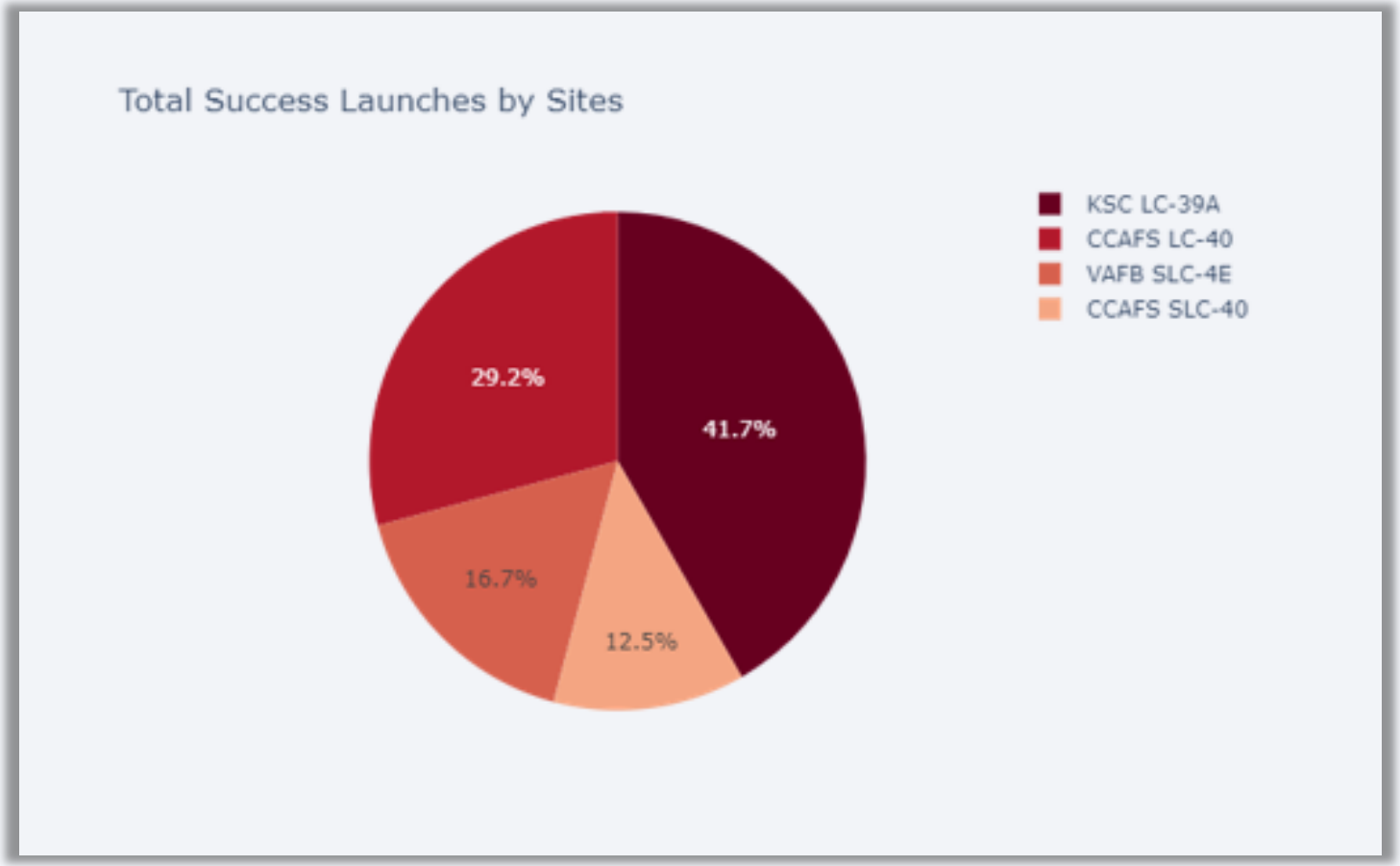
City

Section 5

Build a Dashboard with Plotly Dash



SpaceX Launch Site: Highest Success Rate

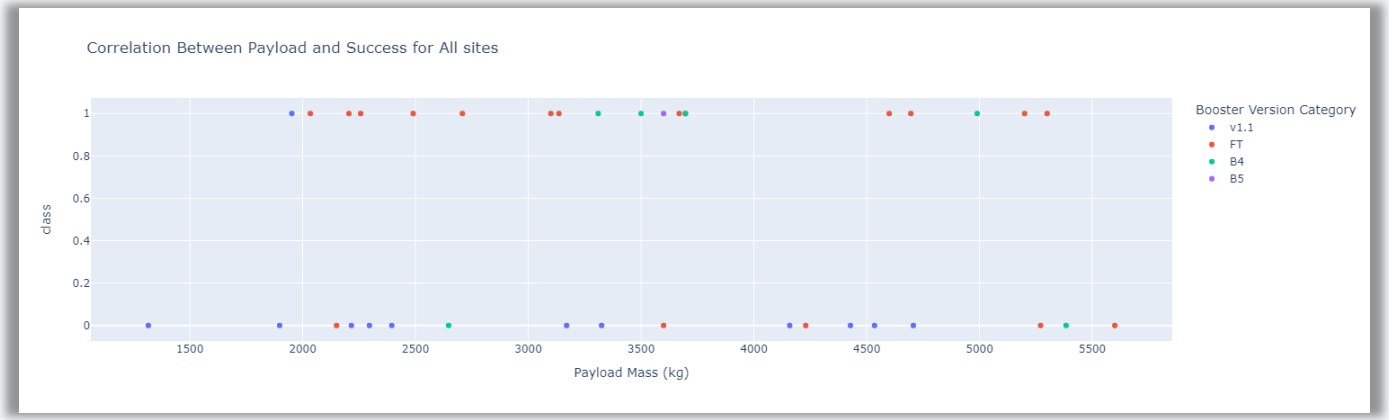


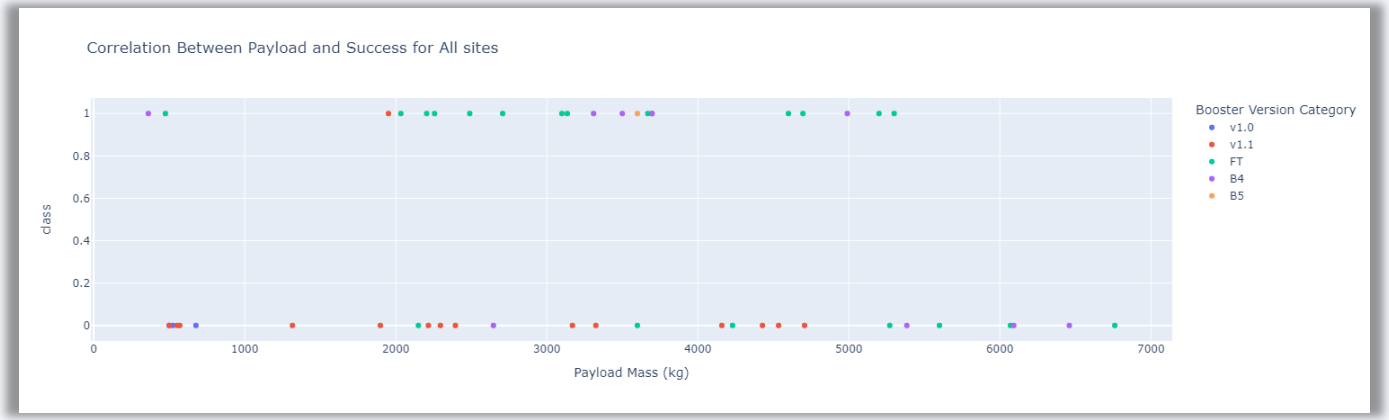
41.7% of total successful launches are conducted from KSC LC-39A among all sites.

SpaceX Launch Site: Highest Success Ratio

KSC LC-39A has 76.9% success ratio among all sites.

SpaceX Launch Site: Payload Mass Vs. Success Rate



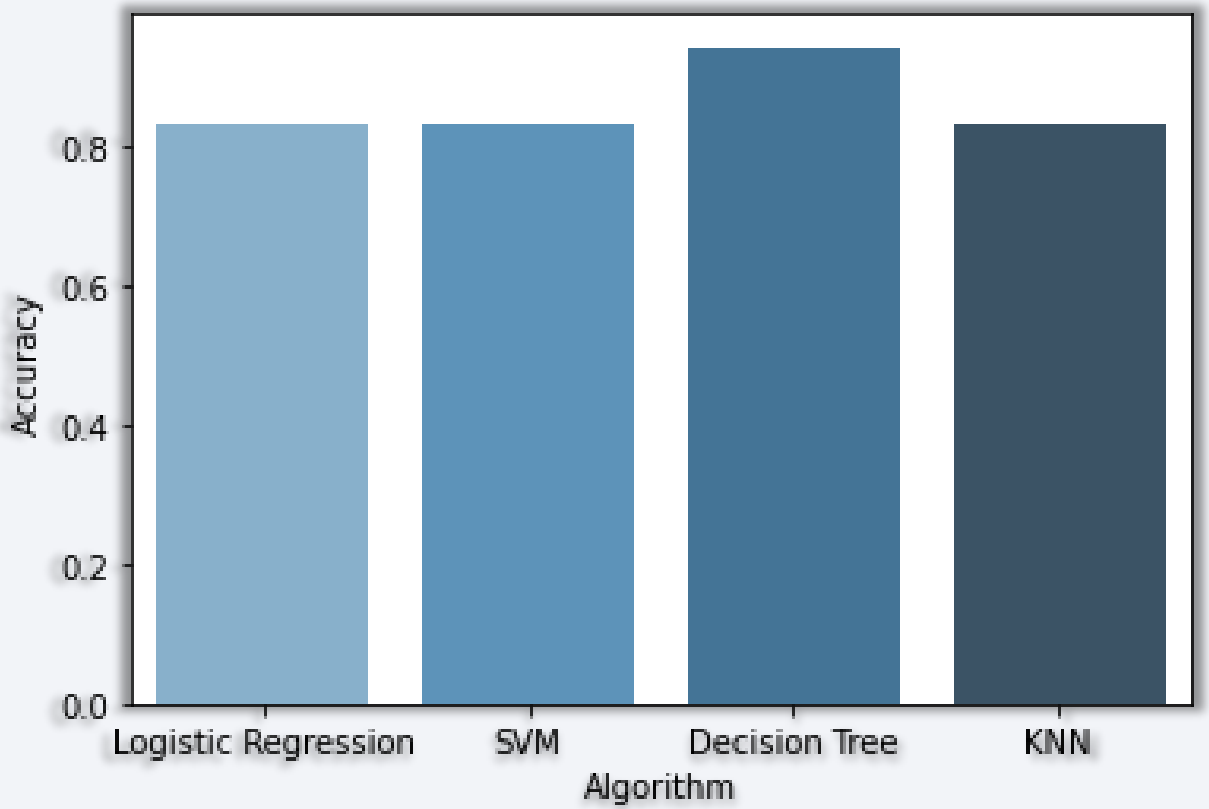


- Considering all sites, most of the successful launches are conducted with payload ranging from 1900-5300 kg
- Booster Version- FT is observed as most success full Booster Version
- There is almost no successful launches beyond 5500kg payload
- Unlike successful launches Payload varies wide ranges from 500- 6800 kg for unsuccessful launches
- v1.1 seems to be most unsuccessful Booster Version

Section 6

Predictive Analysis (Classification)

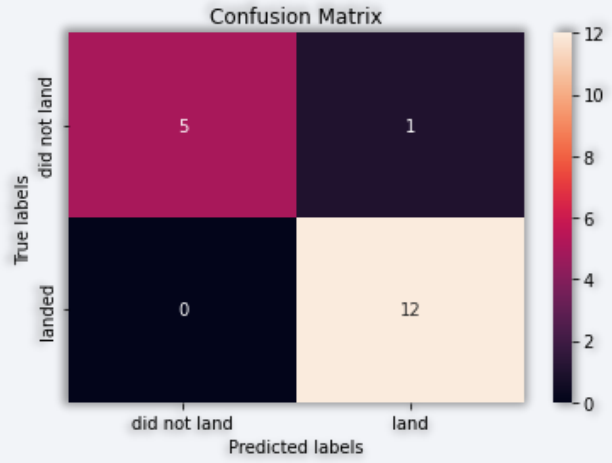
Classification Accuracy



Decision Treen is the most accurate model with 94% accuracy score

Confusion Matrix

Decision Tree



Classification Report

Algorithm000	Accuracy	Launch Outcome	Precision	Recall	F1- Score
Decision Tree	0.94	Does not Land	1.00	0.83	0.91
		Land	0.92	1.00	0.96
Logistic Regression	0.83	Does not Land	1.00	0.50	0.67
		Land	0.80	1.00	0.89
Support Vector Machine	0.83	Does not Land	1.00	0.50	0.67
		Land	0.80	1.00	0.89
Support Vector Machine	0.83	Does not Land	1.00	0.50	0.67
		Land	0.80	1.00	0.89

- In terms of Confusion Matrix, Decision Tree turns out to be best Classification Algorithm
- Decision Tree has only one misclassification error when one true unsuccessful landing is predicted as landed successfully
- In case of other algorithms, 3 misclassification errors are observed
- Decision Tree is able to score highest precision (0.92) in classifying successful landing compare to other algorithms
- Decision Tree recall score is very high at 0.83 compared 0.50 of other algorithms in case of classifying failed landing outcome

Conclusions

- Flight Number is directly proportional to the successful return from first stage
- The chance of first stage will return is less when payload mass increases
- **VALEO** Orbit Type recorded more than 80% success rate with considerable number of missions
- Launch success rate is showing an increasing trend since 2013
- Most successful launch site is **KSC LC-39A** with highest number of successful launches and success to failure ratio
- **Decision Tree** is the best classification algorithm with highest accuracy score

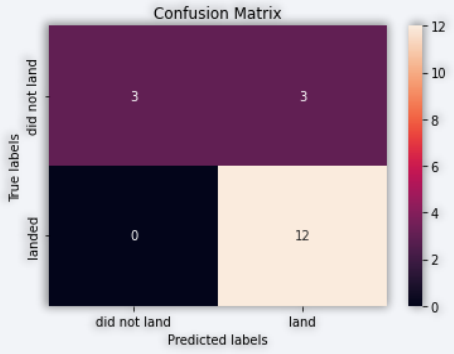
Additional Insights

Way Forward:

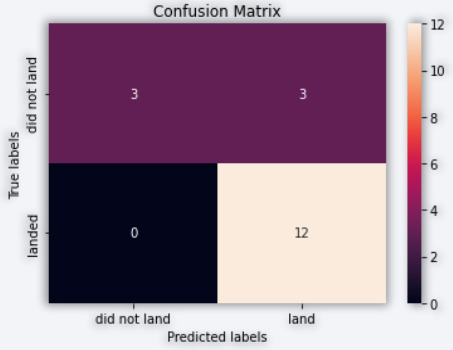
- The model is using very small data to train and test, as a result, misses out many small details of Space Missions. This can be improved by extracting more launch records and retrain and validate the model
- There is scope for more in depth Feature Engineering by using various feature transformation, creating derived variables and interactions among variable
- Some more sophisticated algorithms like XGBosst, Naïve Bayes, Random Forest etc. can be tried and obtain the best algorithm by comparing the evaluation matrix

Appendix

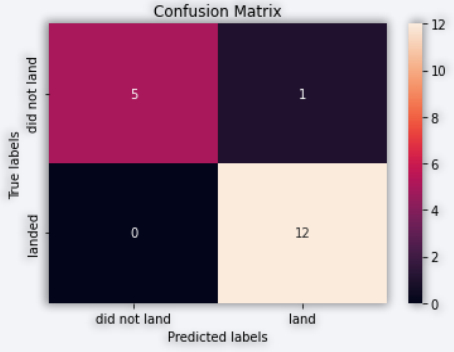
Logistic Regression



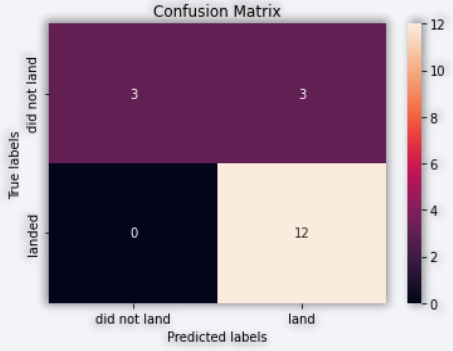
SVM



Decision Tree



KNN



Code Snippet: Best Accuracy Bar plot

Best Accuracy: Bar Plot

```
[ ]  1 lr_acc = logreg_cv.score(X_test,Y_test)
     2 svm_acc = svm_cv.score(X_test,Y_test)
     3 tree_acc = tree_cv.score(X_test,Y_test)
     4 knn_acc = knn_cv.score(X_test,Y_test)
     5 perf_df = pd.DataFrame({'Algorithm':['Logistic Regression','SVM','Decision Tree','KNN'],
     6                         "Accuracy":[lr_acc,svm_acc,tree_acc,knn_acc]})
     7
     8 fig = sns.barplot(x=perf_df['Algorithm'],y=perf_df['Accuracy'],data=perf_df, palette="Blues_d",)
     9 fig
```

Code Snippet: Best Algorithm

Find the method performs best:

```
[ ]  1 print("Logistic Regression Test accuracy ", logreg_cv.score(X_test,Y_test))
     2 print("SVM Test accuracy ", svm_cv.score(X_test,Y_test))
     3 print("Decision Tree Test accuracy ", tree_cv.score(X_test,Y_test))
     4 print("KNN Test accuracy ", knn_cv.score(X_test,Y_test))

Logistic Regression Test accuracy  0.8333333333333334
SVM Test accuracy  0.8333333333333334
Decision Tree Test accuracy  0.9444444444444444
KNN Test accuracy  0.8333333333333334
```

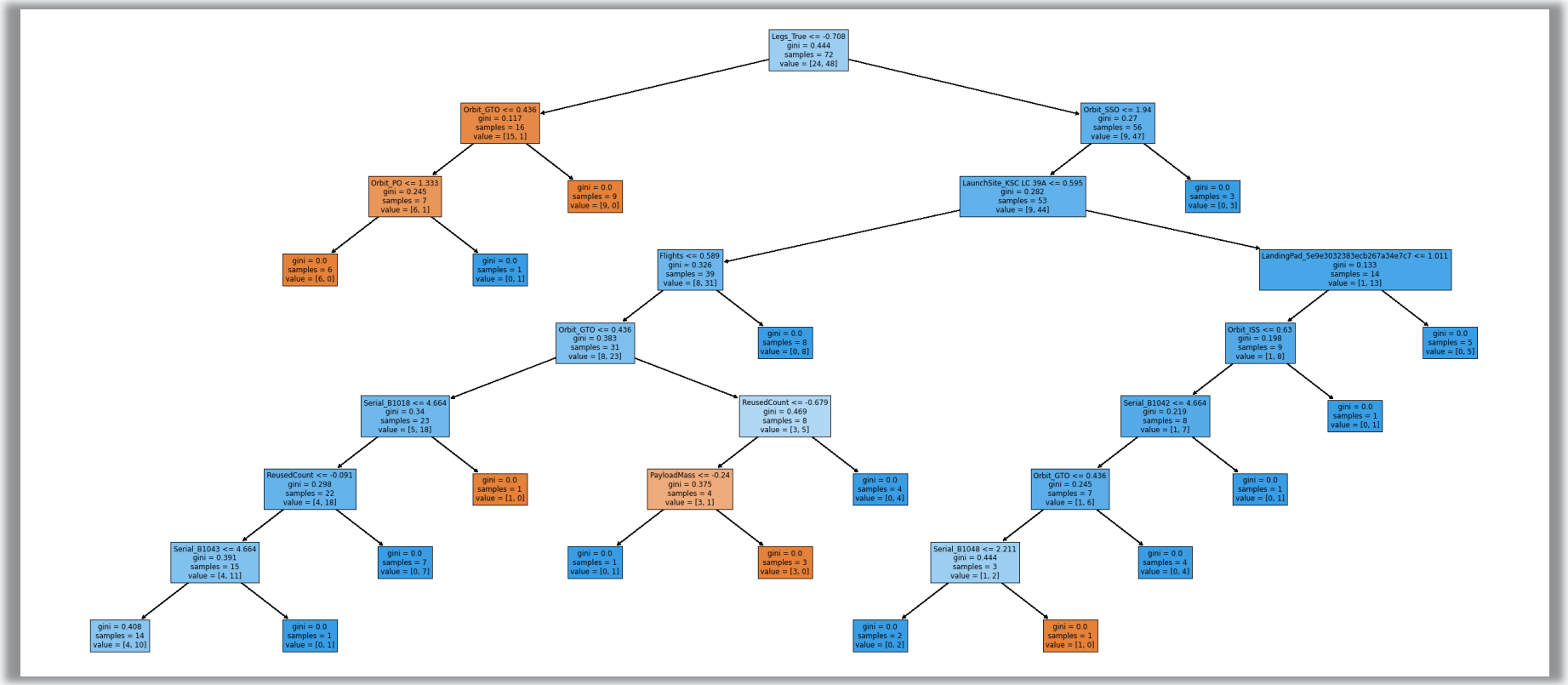
Appendix

Decision Tree: Best Parameters

```
1 print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
2 print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'gini', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}
accuracy : 0.8767857142857143
```

Decision Tree: Plot



GitHub

Thank you!