IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Viacheslav Vietluzhskykh
19 December 2022

Outline

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

Executive Summary

- Summary of methodologies:
  - Data Collection through Spacex's API and with Web Scraping of the SpaceX Launches page in Wikipedia
  - Data Wrangling (cleaning, normalizing/standardizing)
  - Exploratory Data Analysis (EDA) using with SQL queries to IBM's DB2 database
  - Further EDA by means of Data Visualization
  - Creation of and Interactive Dashboard with Folium and Plotly Dash to provide for easier visual analysis
  - Creation of Machine Learning model for Prediction of possible outcomes of return landing
- Summary of all results:
  - Necessary data collected, cleaned and analyzed (EDA)
  - Interactive dashboard created (Interactive Analytics tool), and
  - Proper ML predicting model created (Predictive Analysis tool)

Introduction

- **Project background and context:**

SpaceX advertises Falcon 9 rocket launches on its website, with a cost of 62 million dollars; other providers' cost starts from 165 million dollars each. Much of the savings in SpaceX is because SpaceX can reuse the first stage. To do so they provide for landing of intact first stages back to earth.

- **Problems you want to find answers:**

The ultimate goal of this project is: using publicly available data, to create a model for predictions of: "if the Falcon 9 first stage will land successfully".

What for?:

If we can statistically reliably predict if the first stage will be back and land, we can determine the cost of a planned launch.

This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Introduction – continued. Brief History of SpaceX Launches

- **2001–2004: Founding**
- **2005–2009: Falcon 1 and first orbital launches**
- **2010–2012: Falcon 9, Dragon, and NASA contracts**
- **2013–2015: Commercial launches and rapid growth**
- **2015–2017: Reusability milestones**
- **2019–present: Starship, Starlink, and first crewed launches**

Some of achievements by SpaceX, relevant to this project

Date	Achievement	Flight
28 September 2008	First privately funded fully liquid-fueled rocket to reach orbit.[102]	Falcon 1 flight 4
14 July 2009	First privately developed liquid-fueled rocket to put a commercial satellite in orbit.	RazakSAT on Falcon 1 flight 5
9 December 2010	First private company to successfully launch, orbit, and recover a spacecraft.	SpaceX Dragon on SpaceX COTS Demo Flight 1
25 May 2012	First private company to send a spacecraft to the International Space Station (ISS).[103]	Dragon C2+
22 December 2015	First landing of an orbital-class rocket's first stage on land.	Falcon 9 B1019 on Orbcomm OG2 M2
8 April 2016	First landing of an orbital-class rocket's first stage on an ocean platform.	Falcon 9 B1021 on SpaceX CRS-8
30 March 2017	First reuse, reflight and (second) landing of an orbital first stage.[68]	Falcon 9 B1021 on SES-10
30 March 2017	First controlled flyback and recovery of a payload fairing.[104]	SES-10
3 June 2017	First re-flight of a commercial cargo spacecraft.[105]	Dragon C106 on SpaceX CRS-11
6 February 2018	First private spacecraft launched into heliocentric orbit.	Elon Musk's Tesla Roadster on Falcon Heavy test flight

Source: https://en.wikipedia.org/wiki/SpaceX#Launch_market_competition_and_pricing_pressure

Section 1

Methodology

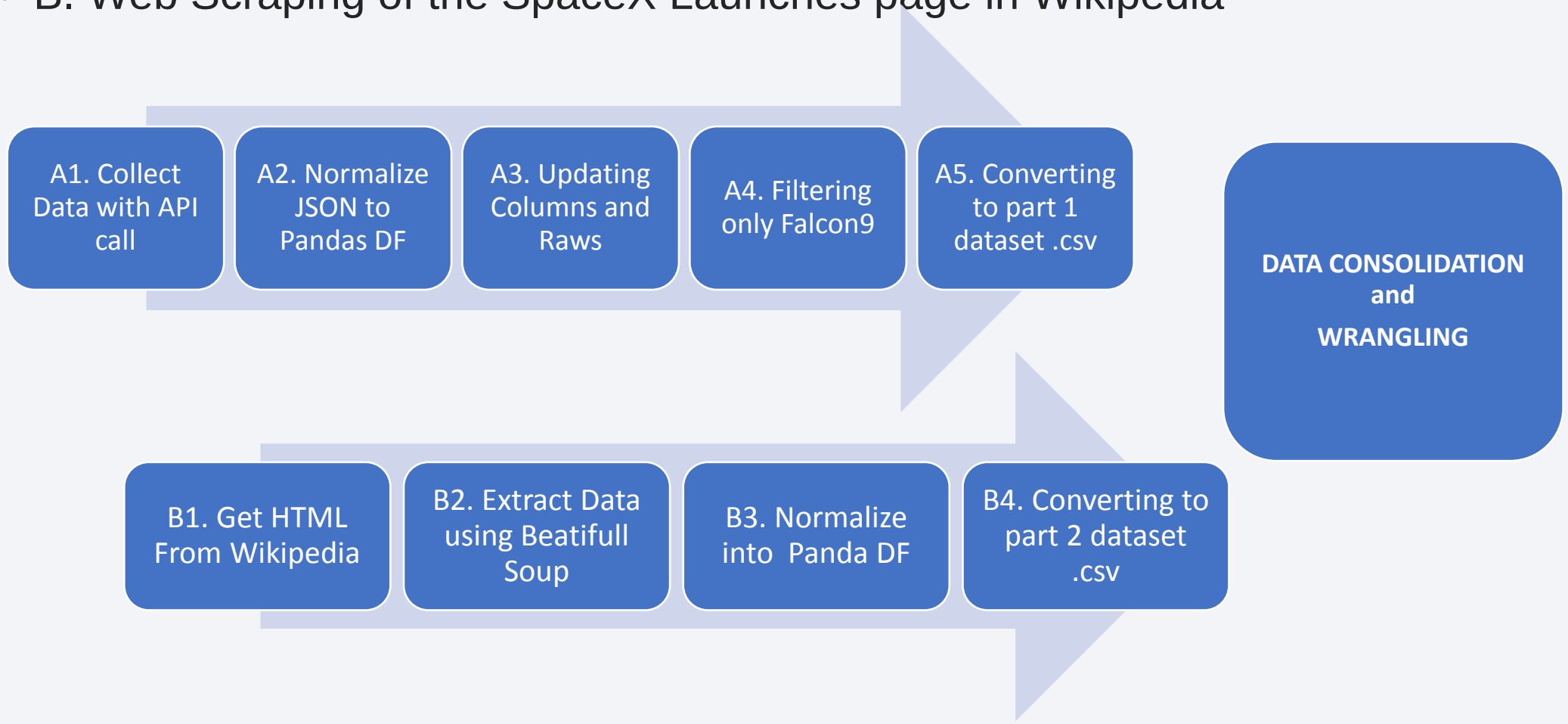
Methodology

Executive Summary

- Data collection methodology: REST API from SpaseX + Web Scraping of the SpaceX Launches page in Wikipedia with pandas BeautifullSoup Library
- Perform data wrangling
  - Converting JSON raw data into a dataframe, using the json_normalize function; Filtering only data for Falcon 9 (dropping ones for Falcon 1).
  - Replacing NULL values inside the PayloadMass with Mean value
  - Sampling the data sets. Create Outcome Attribute as converted to Classes y. y. (either 0 or 1).
  - One Hot encoding of non-numeral data fields
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - LR, SVM, Tree and KNN were created and the best set of parameters was chosen by means of GridSearchCV

Data Collection

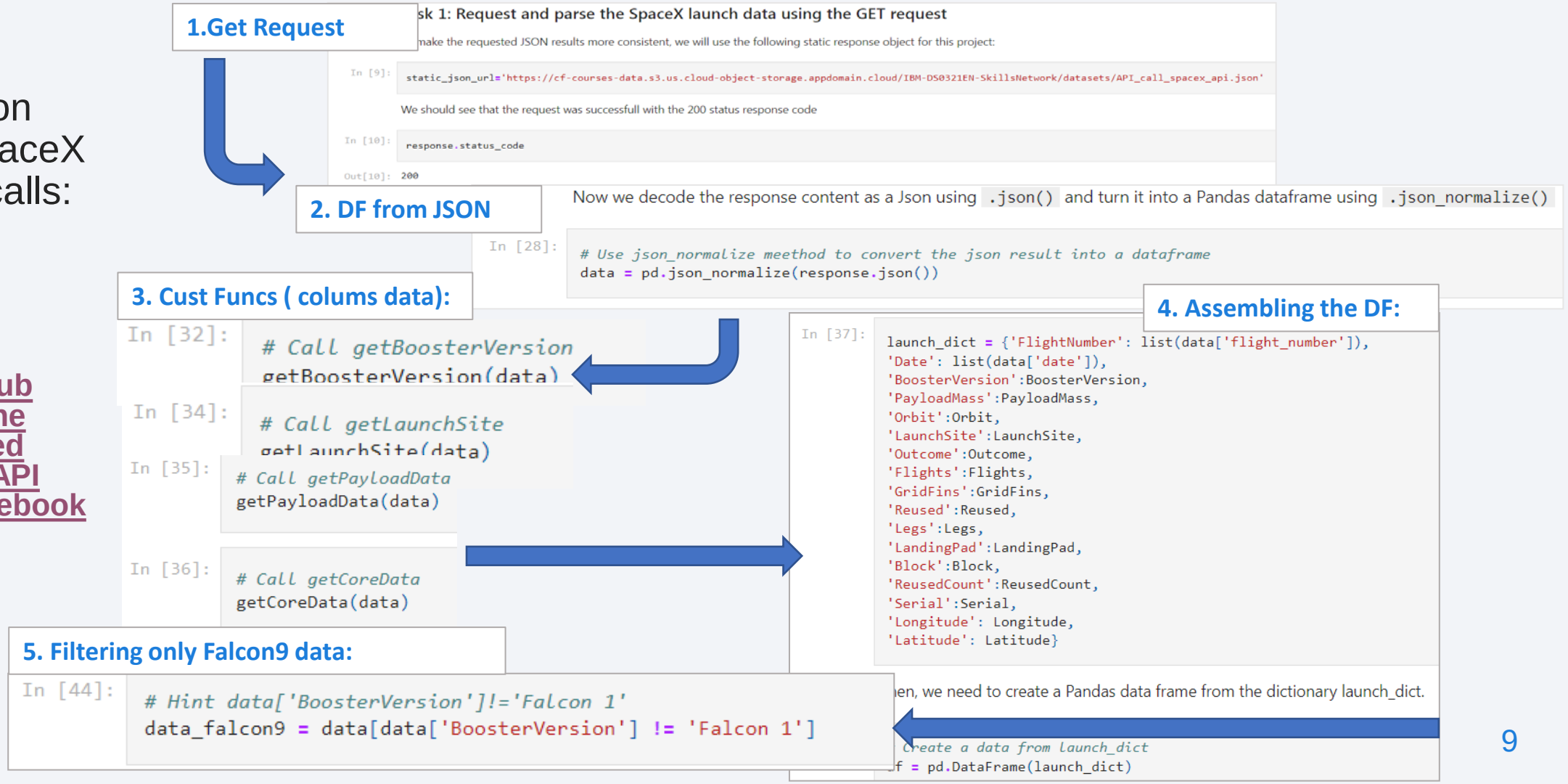
- A. Collect data via REST API from SpaseX
- B. Web Scraping of the SpaceX Launches page in Wikipedia



Data Collection – SpaceX API

- Data collection with SpaceX REST calls:

- **>> GitHub URL of the completed SpaceX API calls notebook**



Data Collection - Scraping

- Web scraping process:

1. Request the Falcon9 Launch HTML page

```
In [57]:
# use requests.get() method with the provided static_url
# assign the response to a object
data = requests.get(static_url).text
```

2. Create ButifullSoup:

```
soup = BeautifulSoup(data, 'lxml')
```

3. Find Tables:

```
html_tables = soup.find_all('table')
```

4. Get Columns Names

```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

for text in first_launch_table.find_all('th'):
    name = extract_column_from_header(text)
    if ((name != None) and (len(name) > 0)):
        column_names.append(name)
    else:
        pass
```

5. Create Dict and fill with data:

```
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
launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
```

```
#Extract each table
for table_number, table in enumerate(soup.find_all('table', "wikitabl
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number correspondi
        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 ke
            launch_dict["Flight No."].append(flight_number)

            datatimelist = date_time(row[0])
```

6. Dict -> DF - > CSV:

```
# Now the df created without returning error
df=pd.DataFrame(launch_dict)
```

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

- >>> GitHub URL of the completed SpaceX Scraping notebook

Data Wrangling

1. Check for NaN and Which columns are numerical and categorical :

```
In [23]: df.isnull().mean()
```

```
In [24]: df.dtypes
```

2. Calculate the number of launches on each site:

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

3. Calculate the number and occurrence of each orbit:

```
df.Orbit.value_counts()  # or :
df['Orbit'].value_counts()
```

4. Calculate the number and occurence of mission outcome per orbit type:

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

5. Create a landing outcome label from Outcome column:

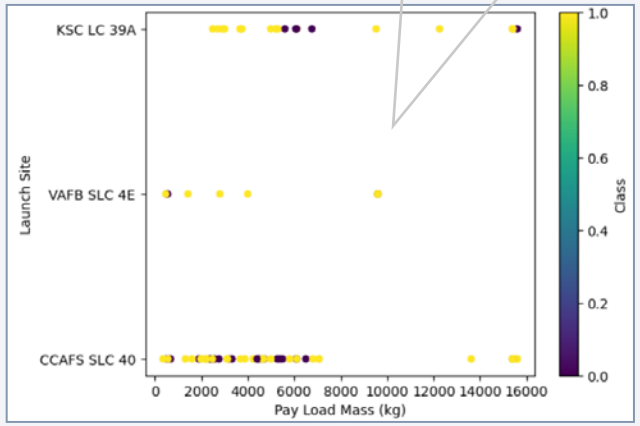
```
In [58]:
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class =[]
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
       landing_class.append(0)
    else:
        landing_class.append(1)
print(landing_class[0:10])
print(len(landing_class))
```

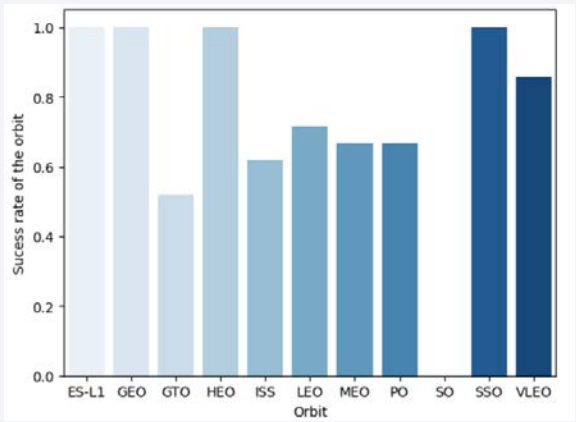
```
In [56]: df['Class']=landing_class
```

>>> GitHub URL of completed Data Wrangling notebook

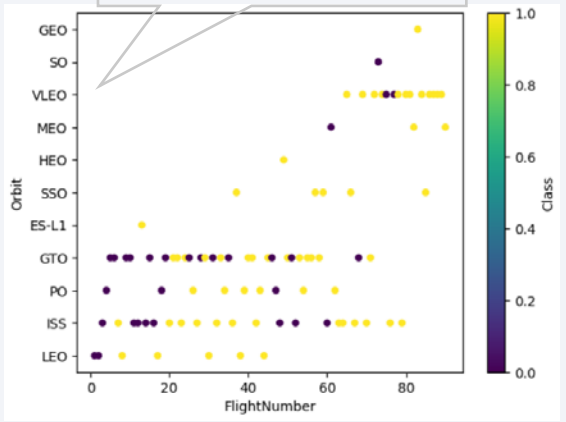
EDA with Data Visualization

For the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000)

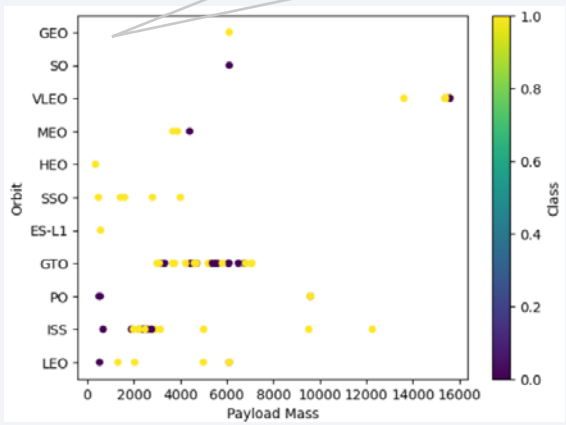


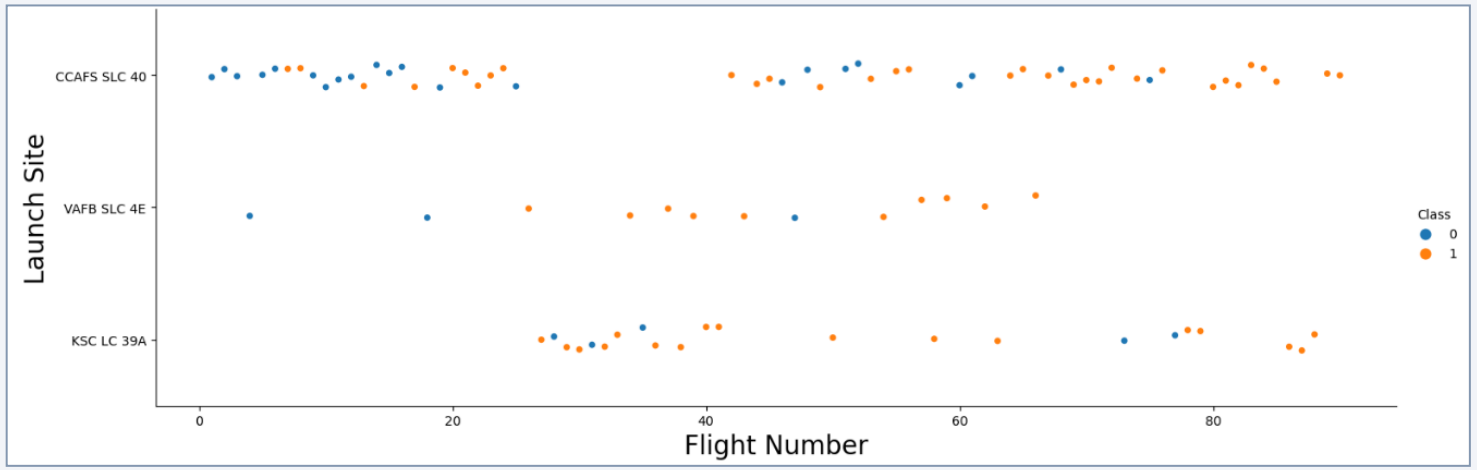


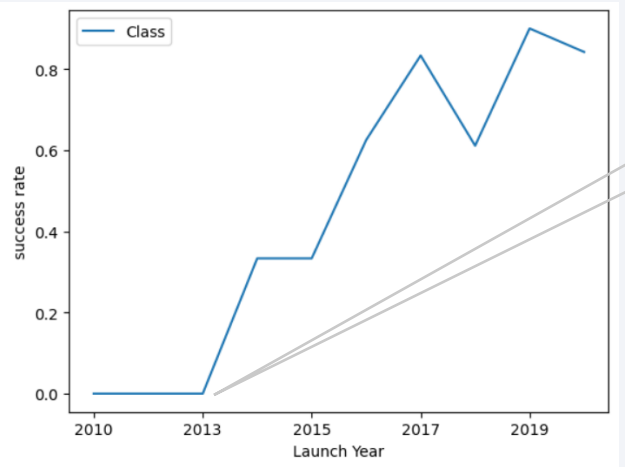
in the LEO orbit the Success relates to the number of flights. No relationship between flight number when in GTO orbit



With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.







Sucess rate since 2013 kept increasing till 2020

>>> GitHub URL of completed EDA with Data Withualization notebook

EDA with SQL

SQL queries performed:

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

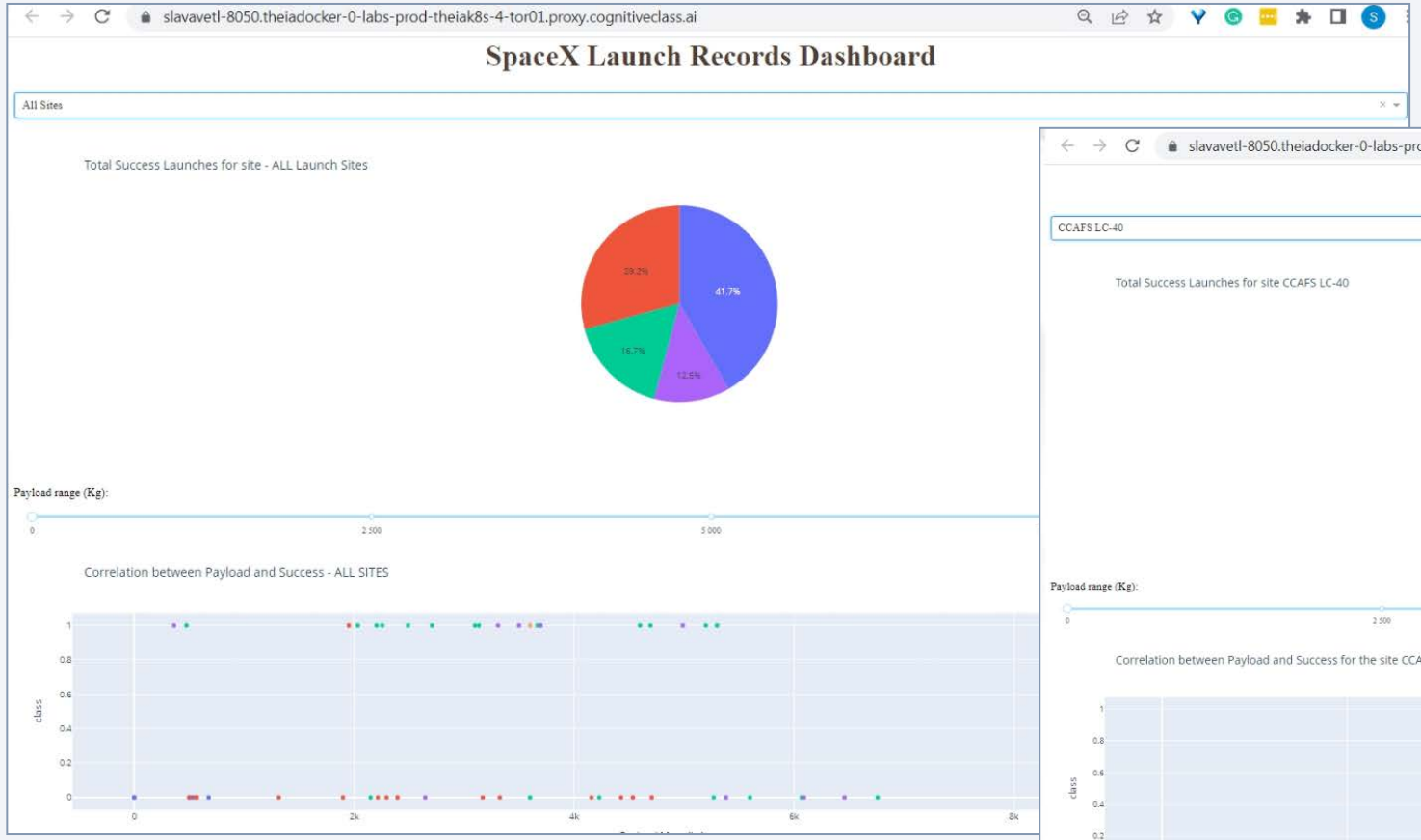
>>> GitHub URL of completed EDA with SQL notebook

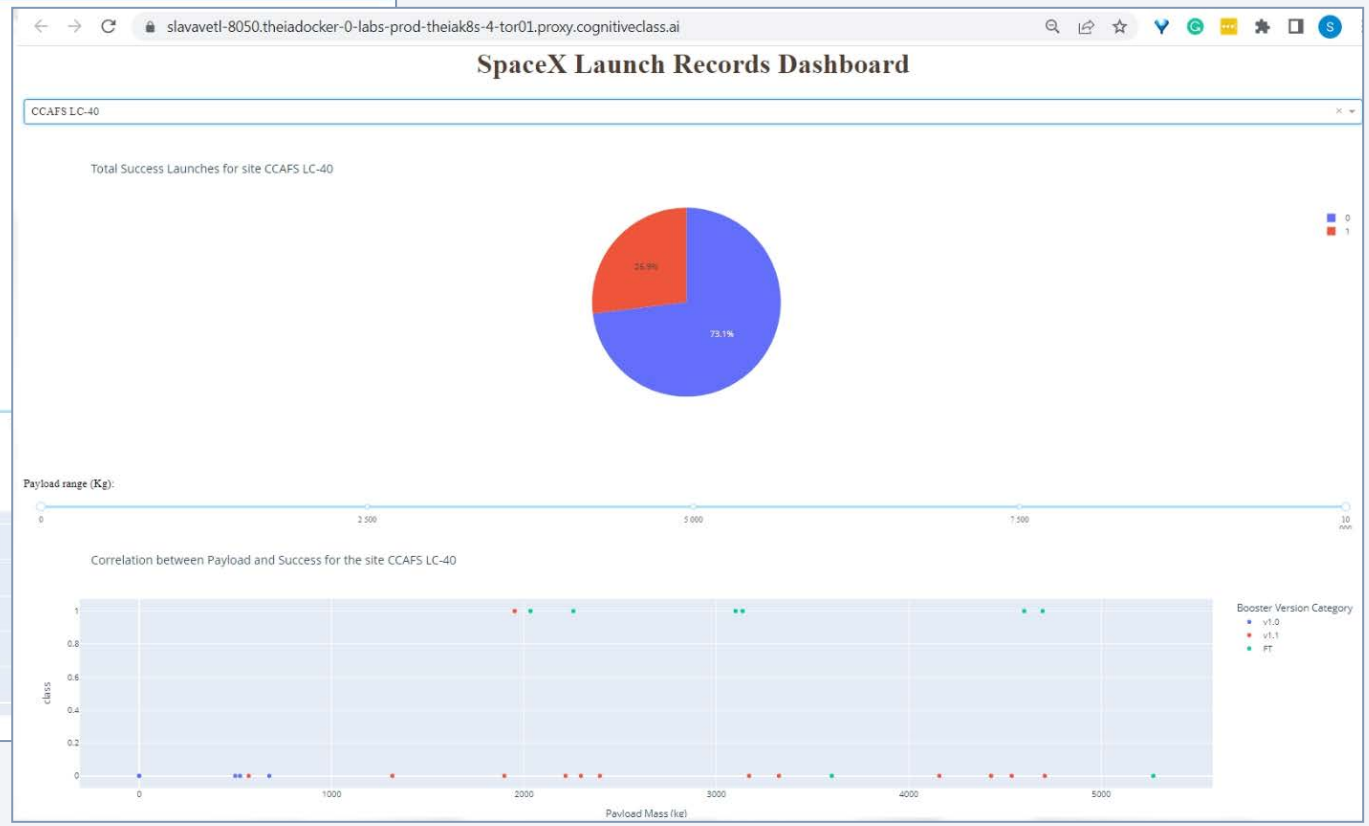
Build an Interactive Map with Folium

- Used folium.Circle() to add a highlighted circle area with a text label on a specific coordinate.
- Used folium.Marker() to create marks on the map
- Used folium.Icon() to create icons on the map
- Used markerCluster() to to simply a map containing many markers
- Used folium.PolyLine() to create a line (polynomial) between points

>>> GitHub URL of completed Map with Folium notebook

Build a Dashboard with Plotly Dash





[>>> GitHub URL to Spacex Dash_app_Done.py]

[>>> GitHub URL to ZIP with .py and print_screens of the Dashboard]

Predictive Analysis (Classification)

- For each of : Logistic Regression, SVM, KNN, DecisionTree, the following steps have been performed:

1. **Model Building (create column for the Class; standardize the data; split into train and test sets; fit the model with set of parameters using GridSearchCV)**

2. **Evaluating the model (using the best hyper-parameters selected by GridSearchCV - calculate accuracies; Calculate Confusion Matrixes; Plot the results)**

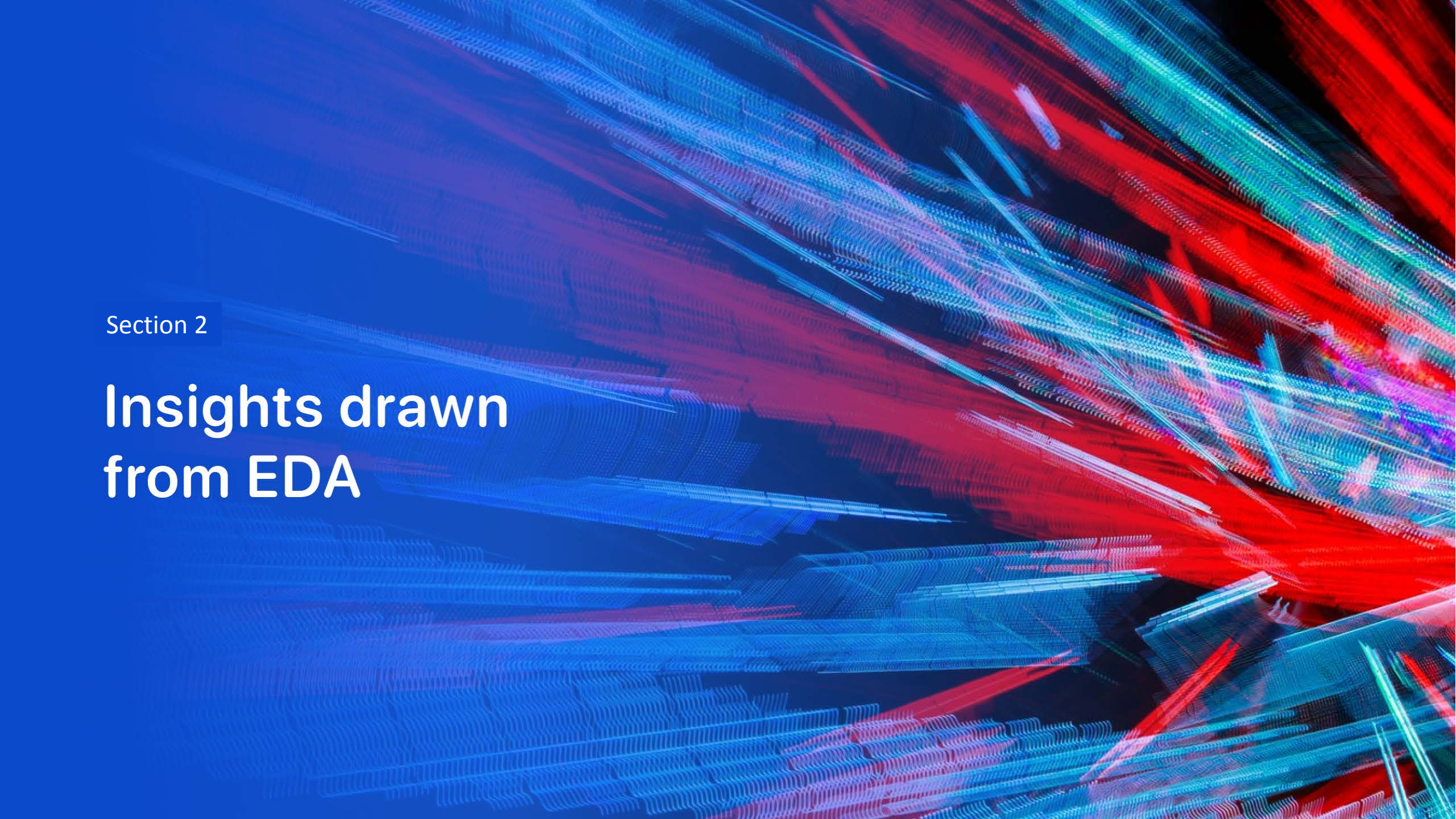
- and finally:

3. **Evaluate and select the best performing model (analyze and find the model with highest accuracy , finally choose optimal model)**

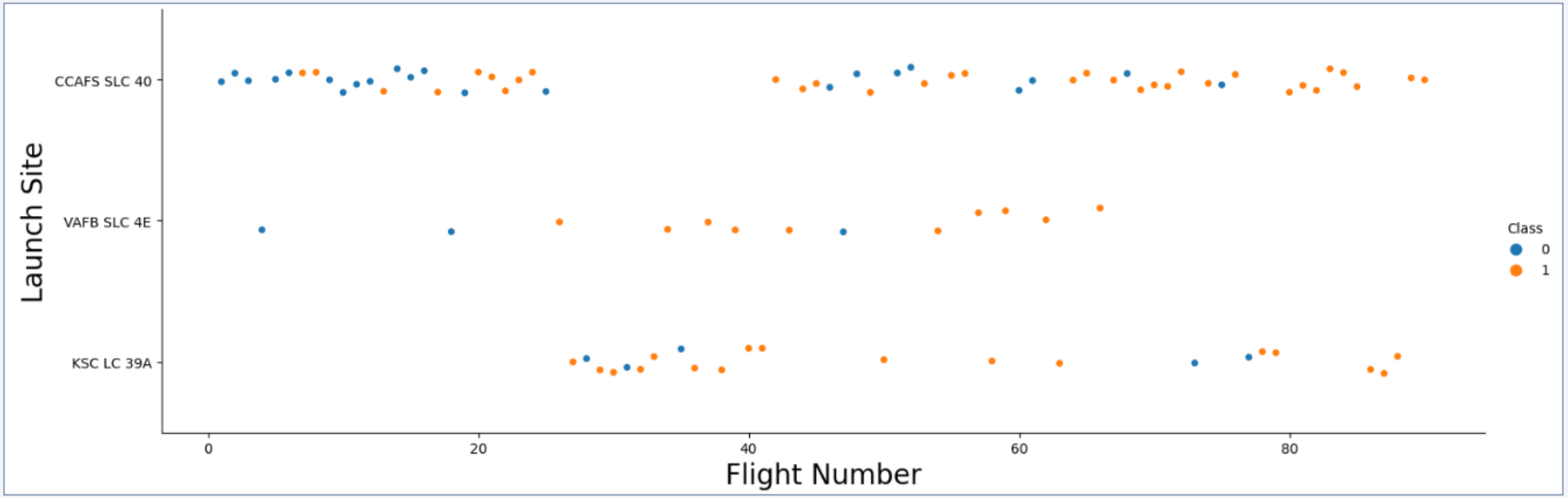
>>> GitHub URL to Predictive Analysis Notebook

Results

- Exploratory data analysis results
  - Orbits SSO, GEO, HEO,ES L1 have the highest success rates
  - KSC LC 39A is the best performing launch site
  - Low weighted payload -- > better success rate, and vice versa
- Interactive analytics demo in screenshots
  - See this info by the link: >> link to Slide 14
- Predictive analysis results
  - The results of all of the four models are close to each other. Slightly better performs the Tree model, so it is preferable to use it.

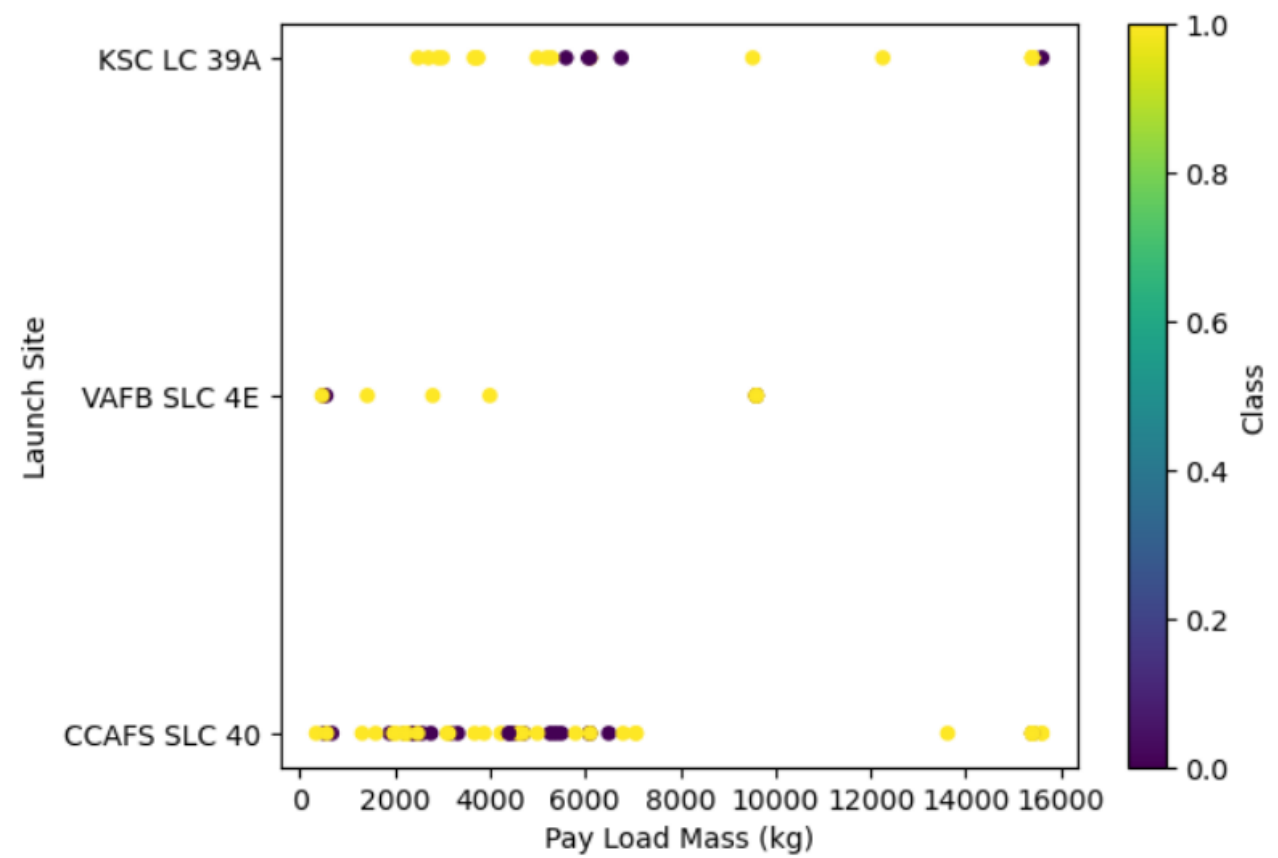

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



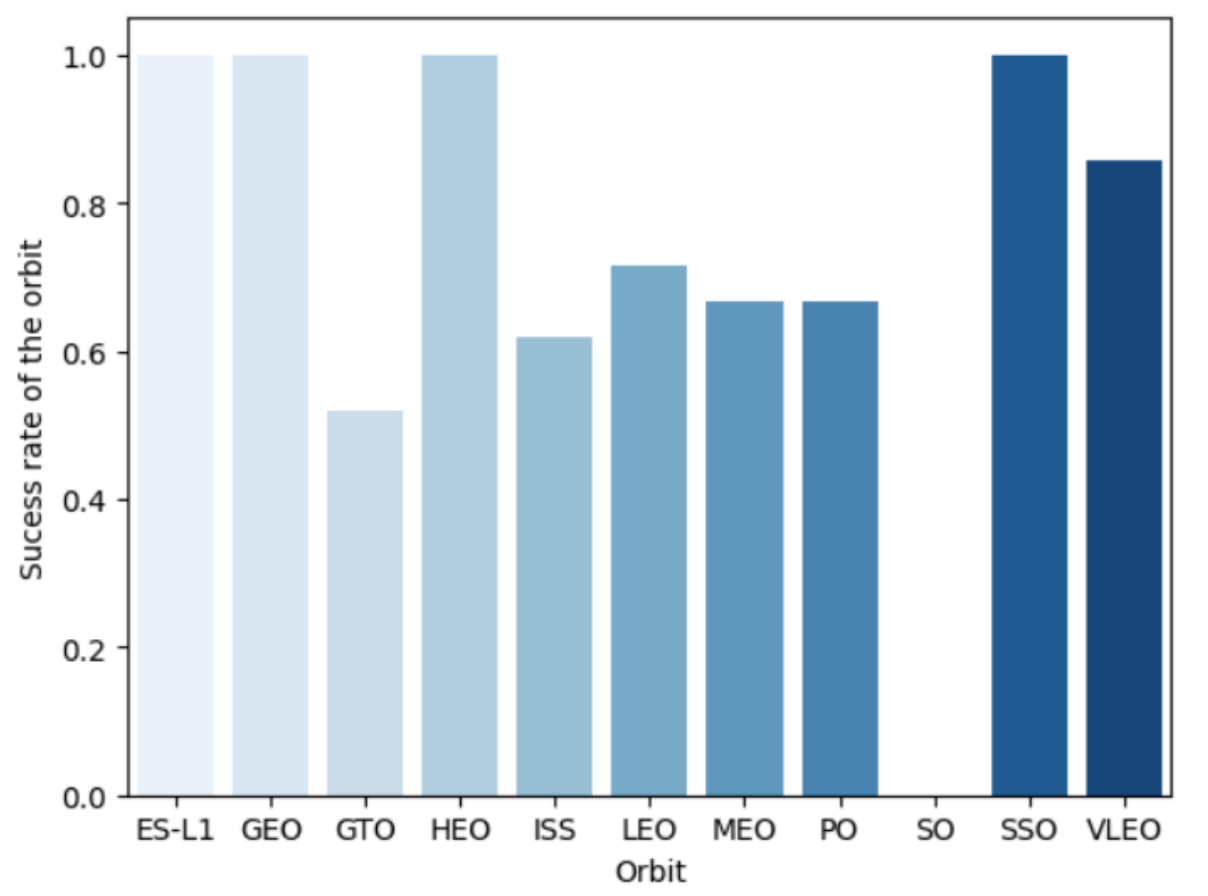
- The most intensively used launch site is CCAFS SLC 40. The least used one is VAFB SLC 4E.
- We could range the sites by decreasing total success rate of launches as:
  - VAFB SLC 4E,
  - KSL LC 39A
  - CCAFS SLC 40

Payload vs. Launch Site



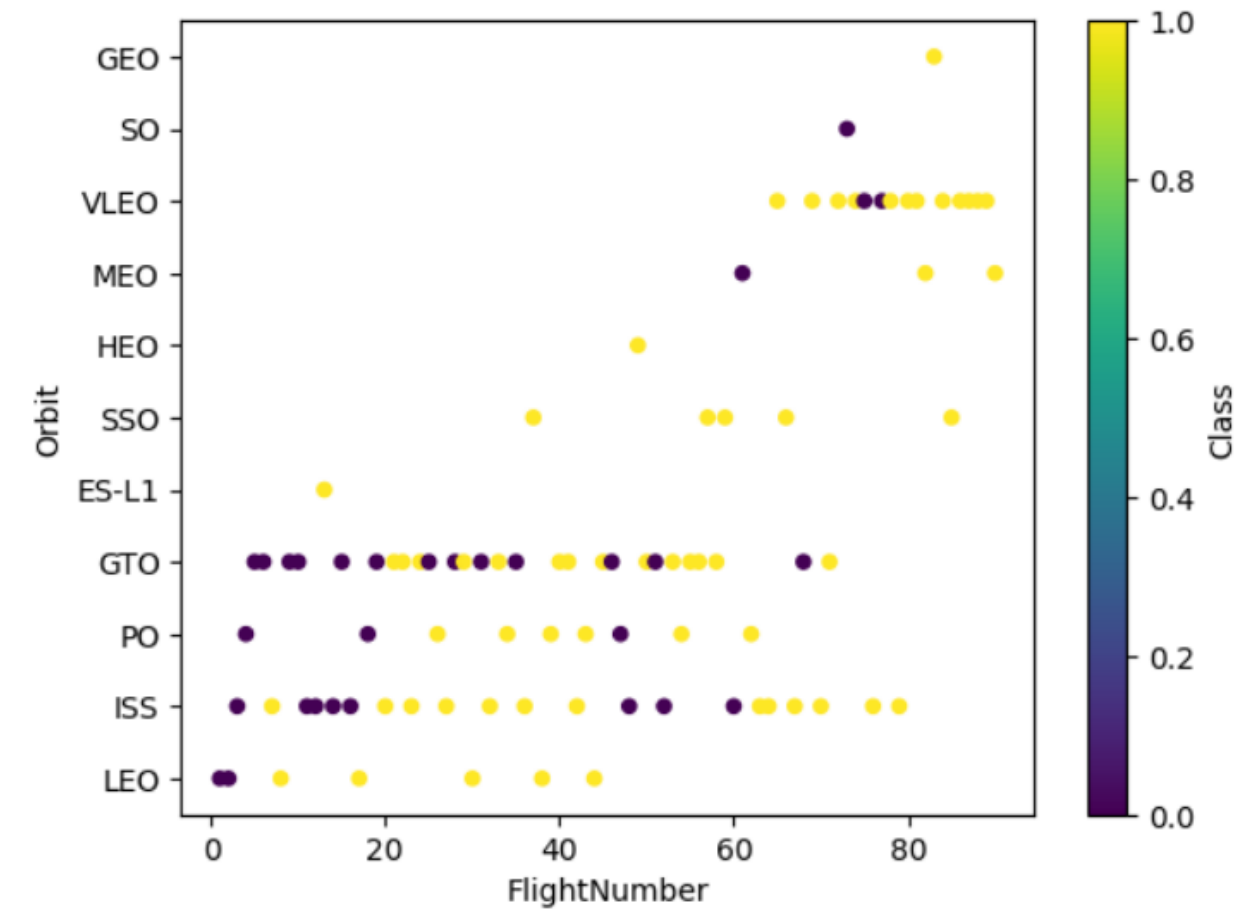
- At the VAFB-SLC launch site there are no rockets launched for heavy payload mass(greater than 10000).
- CCAFS SLC 40 was specialized on launches with payloads less than 7000
- Launches with heavy payloads(>8000) are rare but have higher success rate.

Success Rate vs. Orbit Type



- The highest success rates are with the following orbits: ES-L1, GEO, HEO, SSO.
- The lowest success rates are with the following orbits: GTO, ISS
- SO – orbit type has 0 success rate

Flight Number vs. Orbit Type

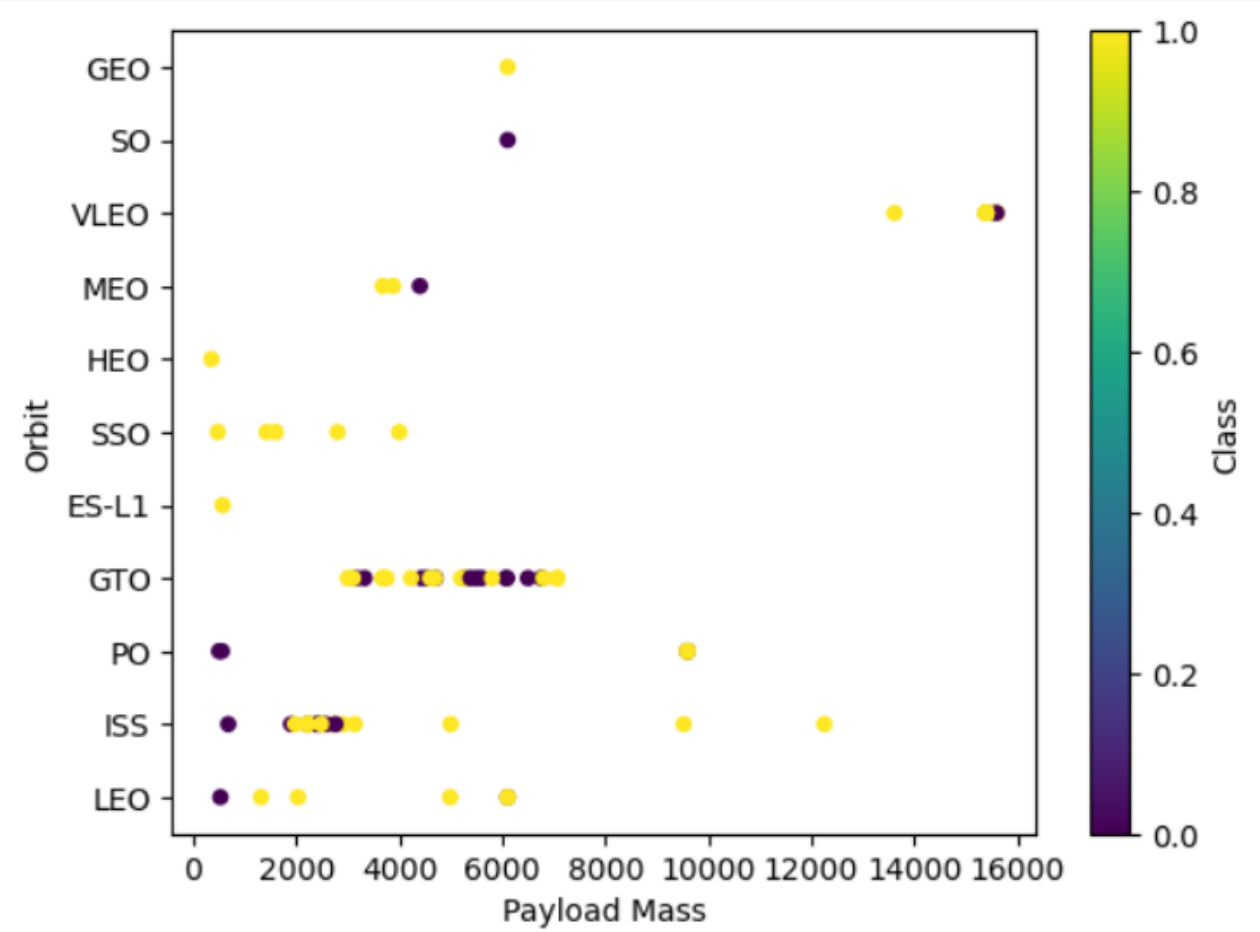


With passing of time (for latest flights numbers) we see:

a) the general increase in success rate, and

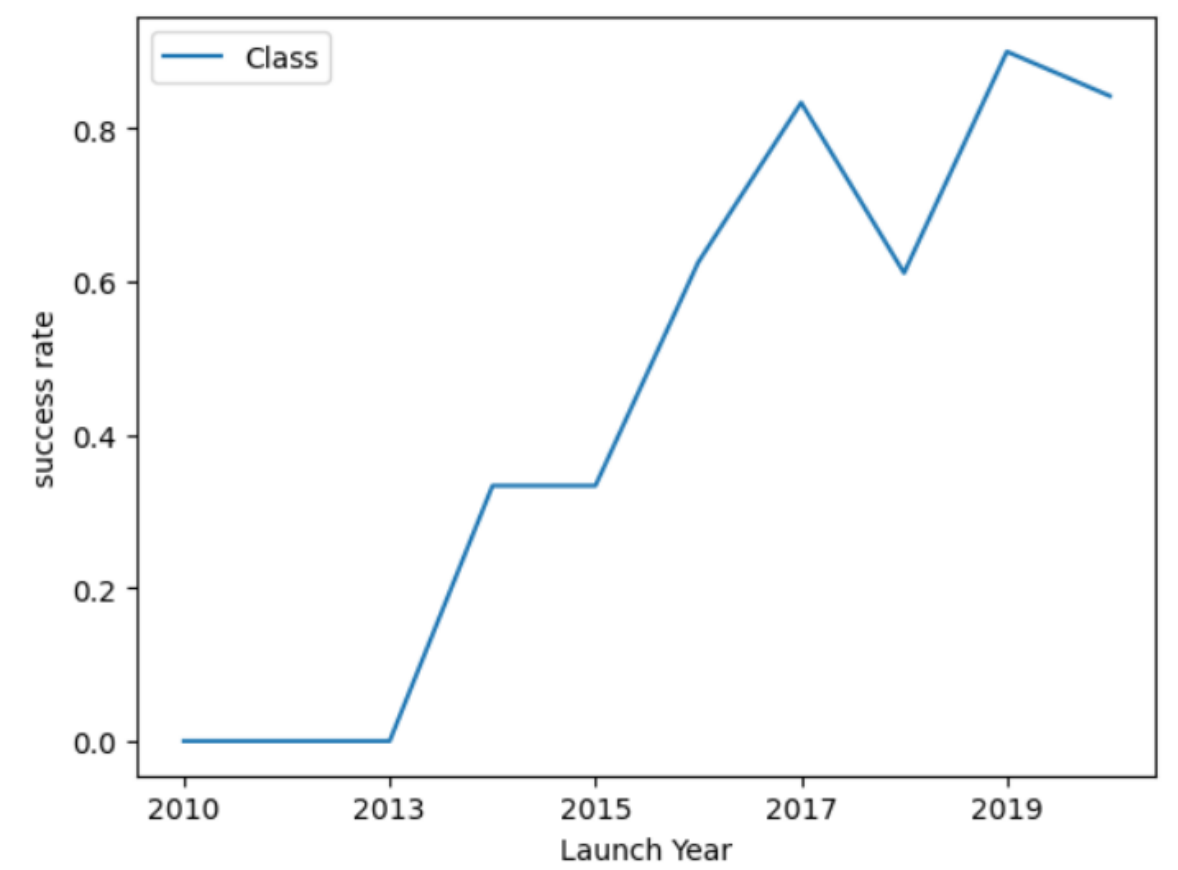
b) shifting in number of launches towards VLEO orbit type.

Payload vs. Orbit Type



- VLEO orbit used exclusively for high payloads(>13 000)
- MEO, SSO, LEO and ES L1 - exclusively for low payloads(<5 000)
- Higher payloads (>8 000), in general, have significantly higher success rates.

Launch Success Yearly Trend



- Starting from 2013 we can see vivid trend of increase in success rates with passing time
- Within the period 2017 to 2019 the growth somehow stabilized at the level of roughly 80%

All Launch Site Names

```
In [21]: %sql SELECT UNIQUE launch_site FROM SPACEX;
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-
Done.
```

Out[21]:

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

Launch Site Names Begin with 'CCA'

In [25]:
```
%sql select * from SPACEX where launch_site like 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1oc
Done.
```

Out[25]:

DATE	time_utc_	booster_version	launch_site	payload
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2

Total Payload Mass

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

In [30]:
```
%sql select sum(payload_mass__kg_) as total_payload_mass_NASA from SPACEX where CUSTOMER like '%CRS%';
```

```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdoma
Done.
```

Out[30]:

total_payload_mass_nasa
48213

Average Payload Mass by F9 v1.1

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

In [32]:
```
%sql select avg(payload_mass__kg_) as avg_payload_mass_F9_v1_1 from SPACEX where booster_version like '%F9 v1.1%';
```

```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:3119
Done.
```

Out[32]:

avg_payload_mass_f9_v1_1
2534

First Successful Ground Landing Date

```
In [36]: %sql select min(DATE) as first_success_ground from SPACEX where landing__outcome = 'Success (ground pad)';
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cl
Done.
Out[36]:
```

first_success_ground
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

In [40]:
```
%sql select booster_version from SPACEX where landing__outcome = 'Success (drone ship)' and (payload_mass__kg_ BETWEEN 4000 AND 6000);
```

```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[40]:

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

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [51]:
```
%sql select mission_outcome, count(*) from SPACEX group by mission_outcome;
```

```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kc
Done.
```

Out[51]:

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

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

In [66]:
```
%sql select booster_version, payload_mass__kg_ from SPACEX where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEX);
```

```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[66]:

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

2015 Launch Records

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

In [81]:
```
%sql select landing__outcome, booster_version, launch_site, DATE from SPACEX \
      where landing__outcome = 'Failure (drone ship)' and DATE between '2015-01-01' and '2015-12-31';
```

```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdom
Done.
```

Out[81]:

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

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

In [85]:

```
%sql select landing__outcome, count(landing__outcome) as num from \
       (select * from SPACEX where DATE between '2010-06-04' and '2017-03-20')\
        group by landing__outcome\
          order by num DESC;
```

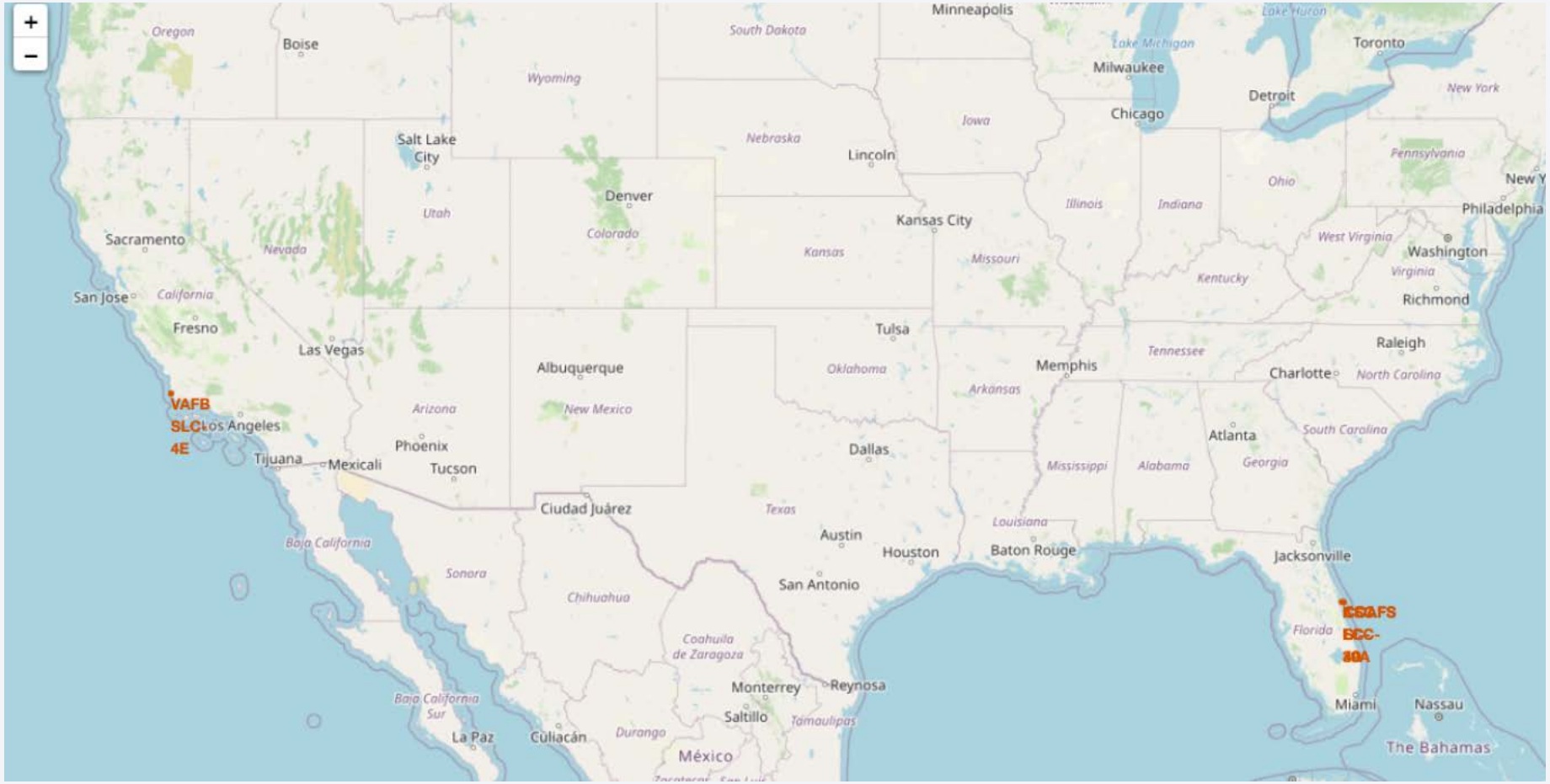
```
 * ibm_db_sa://fct98098:***@0c77d6f2-5da9-48a9-81f8-86b520b87518.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31198/bludb
Done.
```

Out[85]:

landing__outcome	num
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

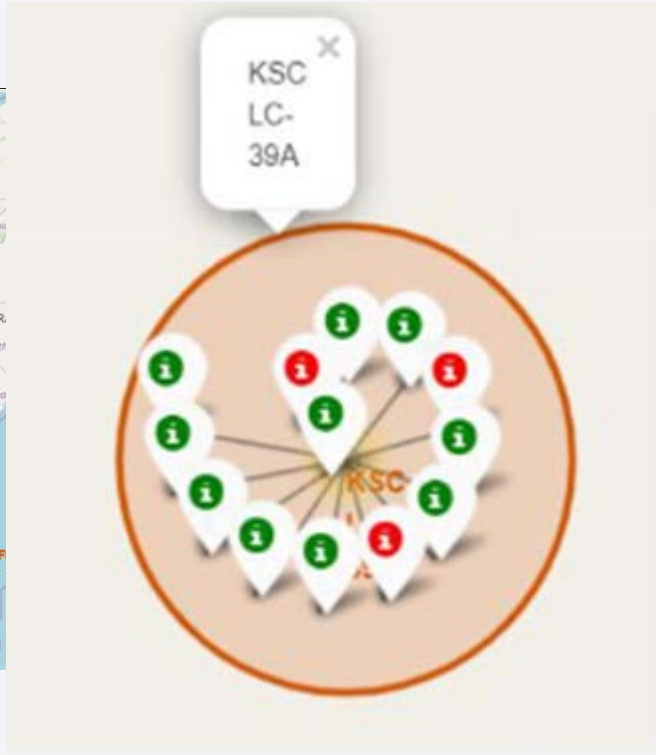

Section 3
Launch Sites
Proximities Analysis

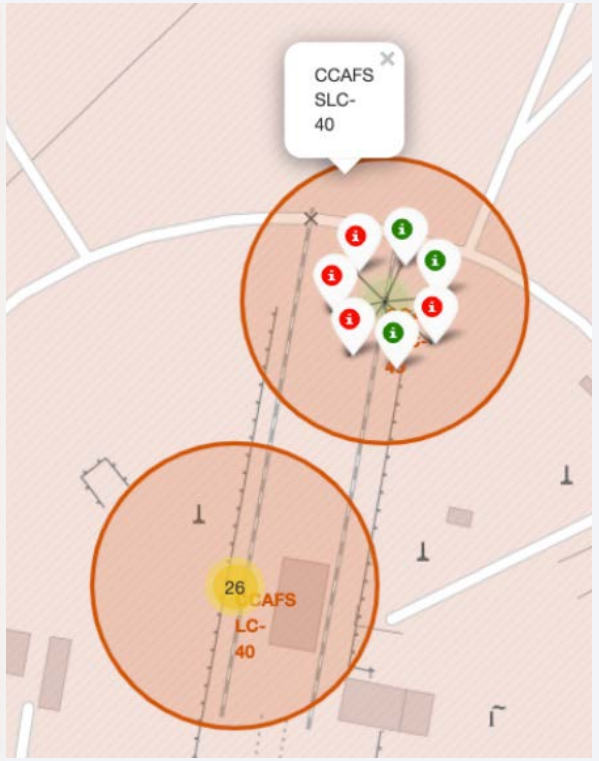
Map with marked Launch Sites



- Launch sites are located in USA and near Earth's equator to take optimum advantage of the Earth's substantial rotational speed.
- They are situated close to coast line for transportation and safety reasons.

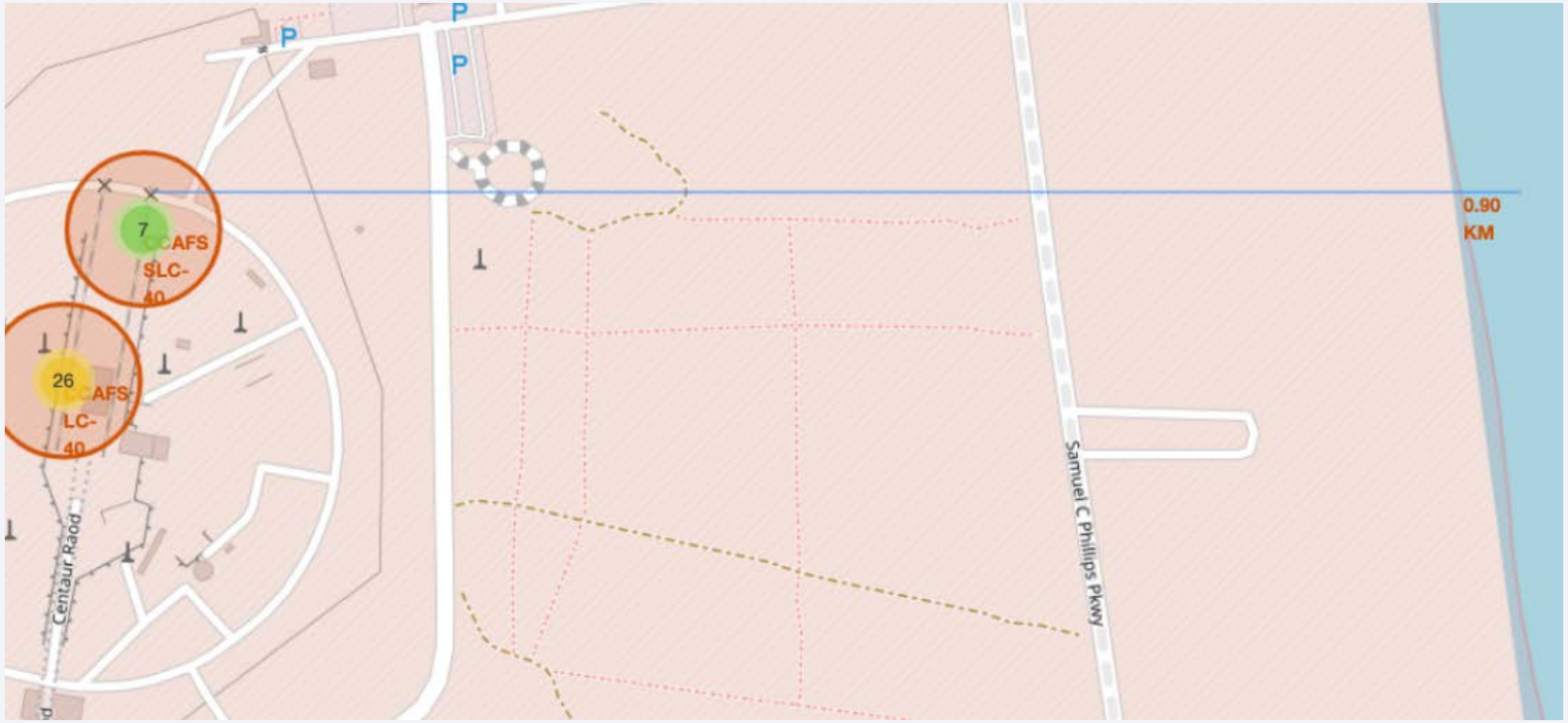
Markers with color labels to depict launch sites





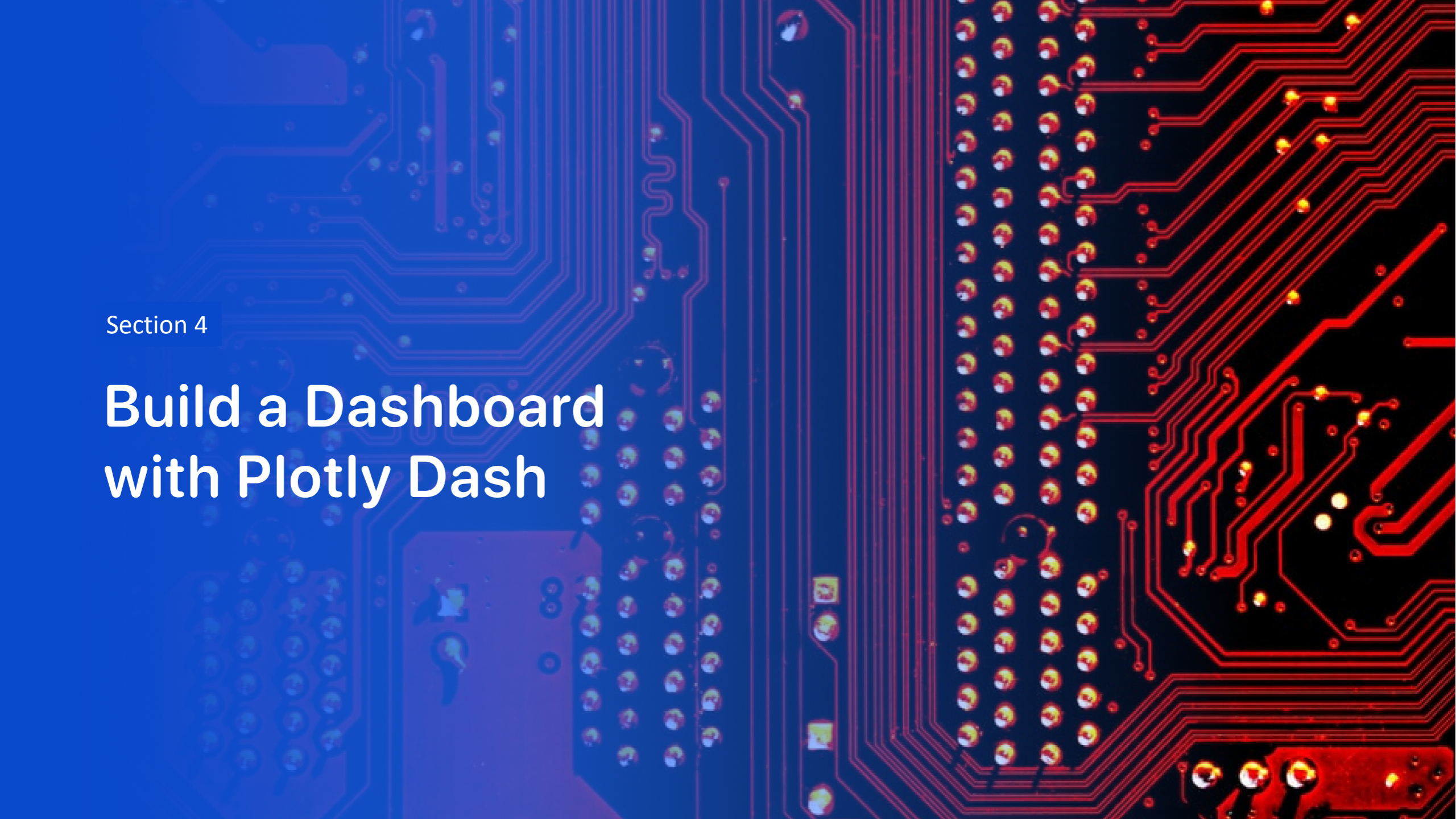
- **Green**-colored markers reflect Success outcome of the launch
- **Red**-colored markers indicate Failure of the launch

Launch Site distance to landmarks

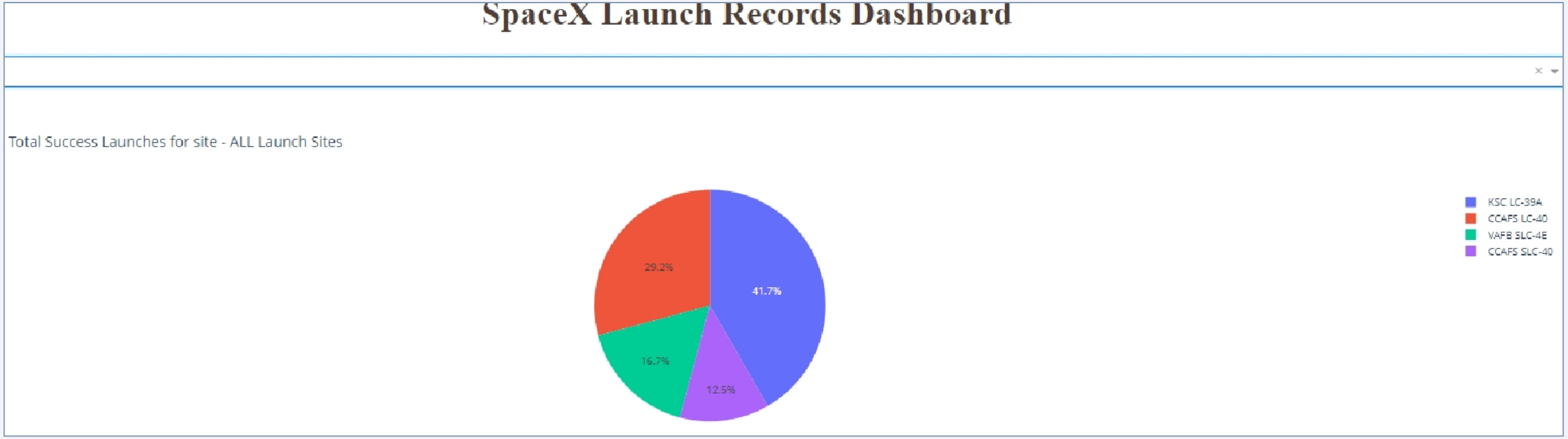


distance_highway : 0.5834695366934144
distance_railway : 1.2845344718142522
distance_city : 51.434169995172326

- Launch sites are situated in close proximity to
  - Coast lines
  - Highways and railways
- At the same time - rather far from cities

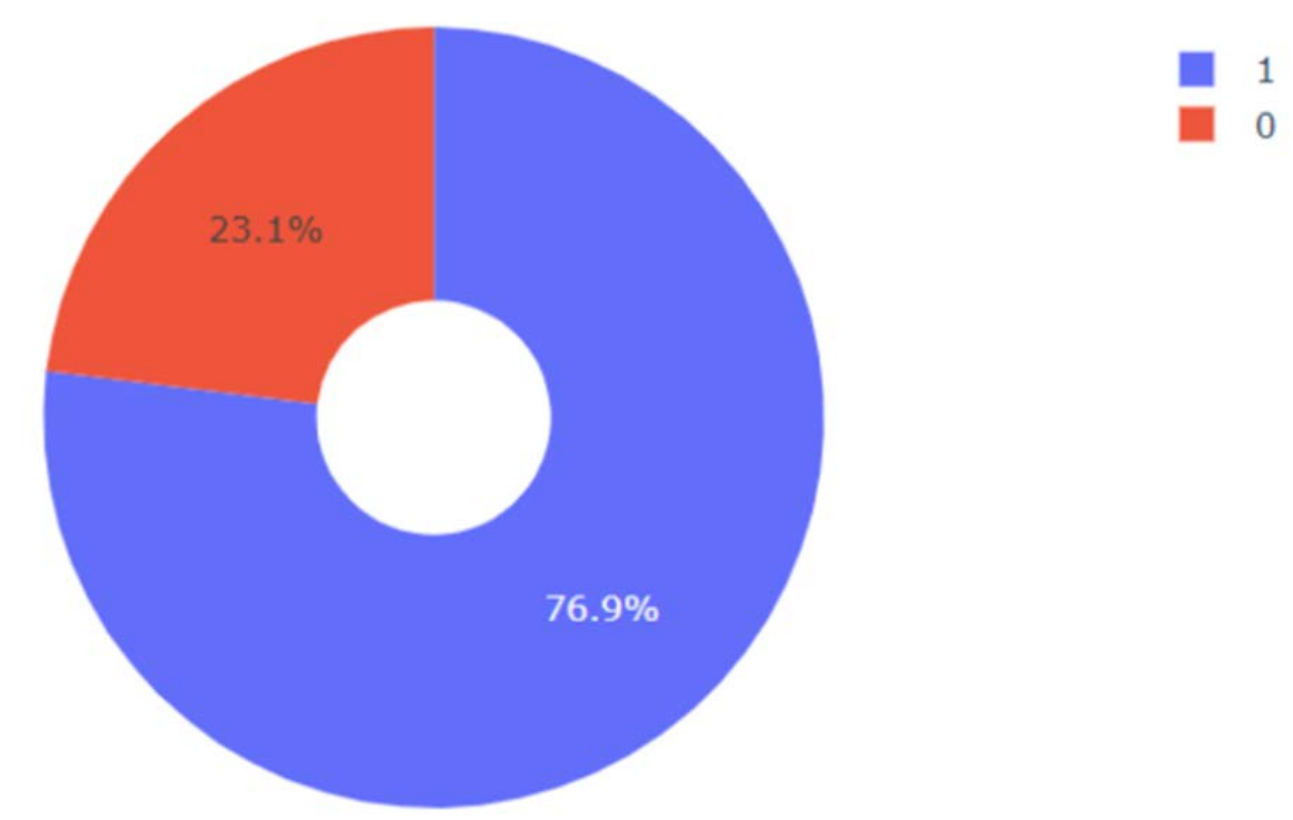

Section 4
Build a Dashboard with Plotly Dash

Pie chart: Success rate per each launch site



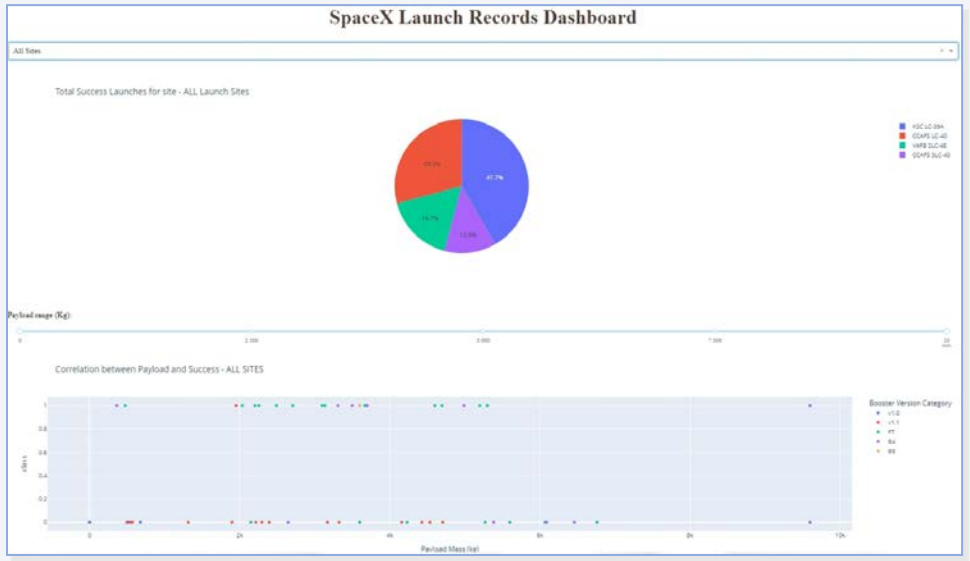
It is seen clearly that KSC LC39 has the highest number of success launches while CCAFS SLC40 - lowest.

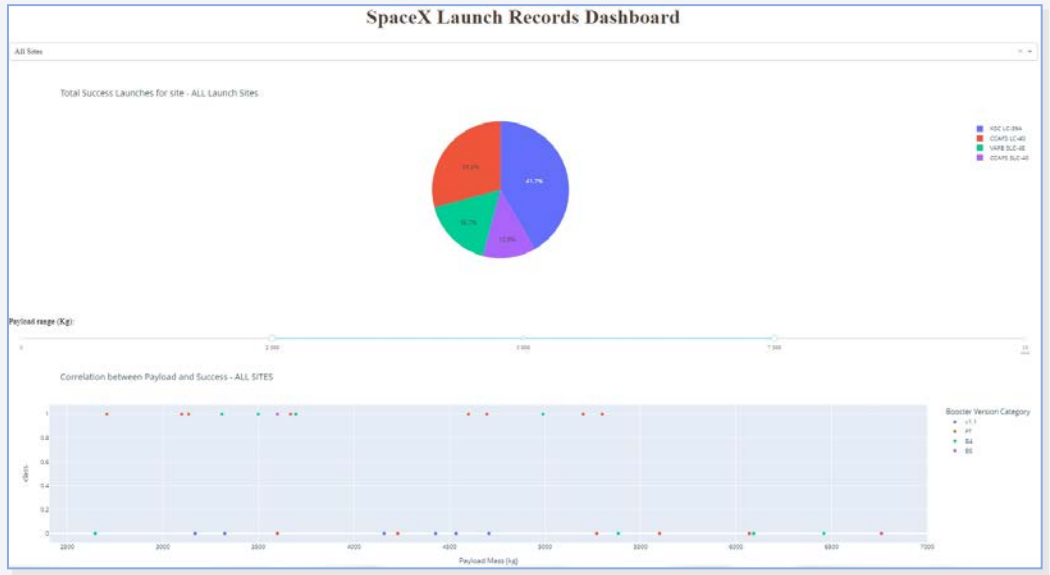
“Drill down” to details of the site with the highest success rate

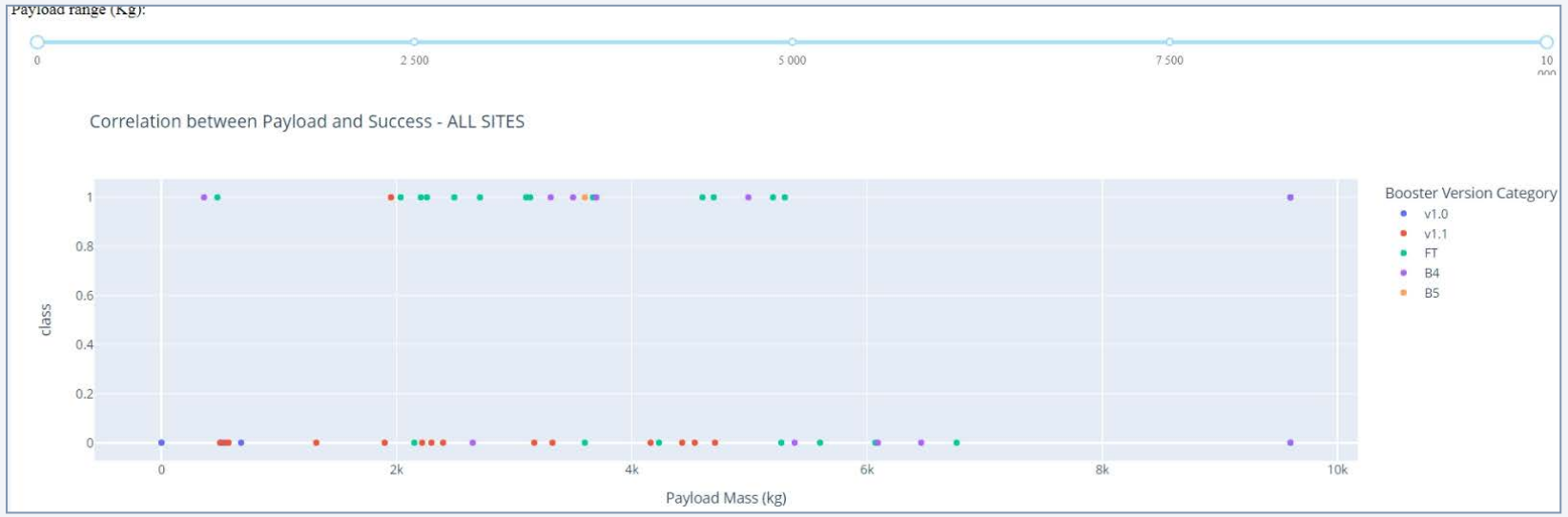


KSC LC39 has the highest success rate 76.9% and failure rate of 23.1%

Full Dashboard with Scatter plot of Payload vs Launch, with range slider for payload







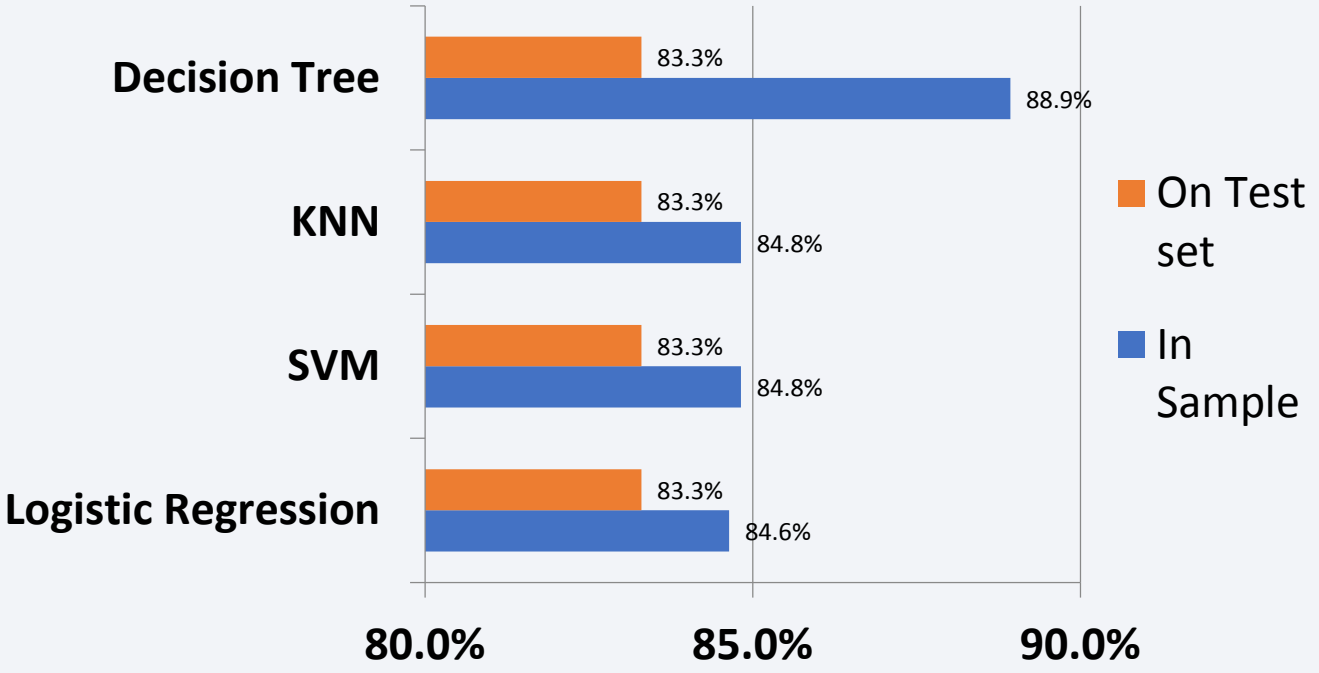
We can say that:

- All launches with payloads 5.5k to 9k were totally unsuccessful
- On all other ranges there is no clear correlation between payload and success

Section 5

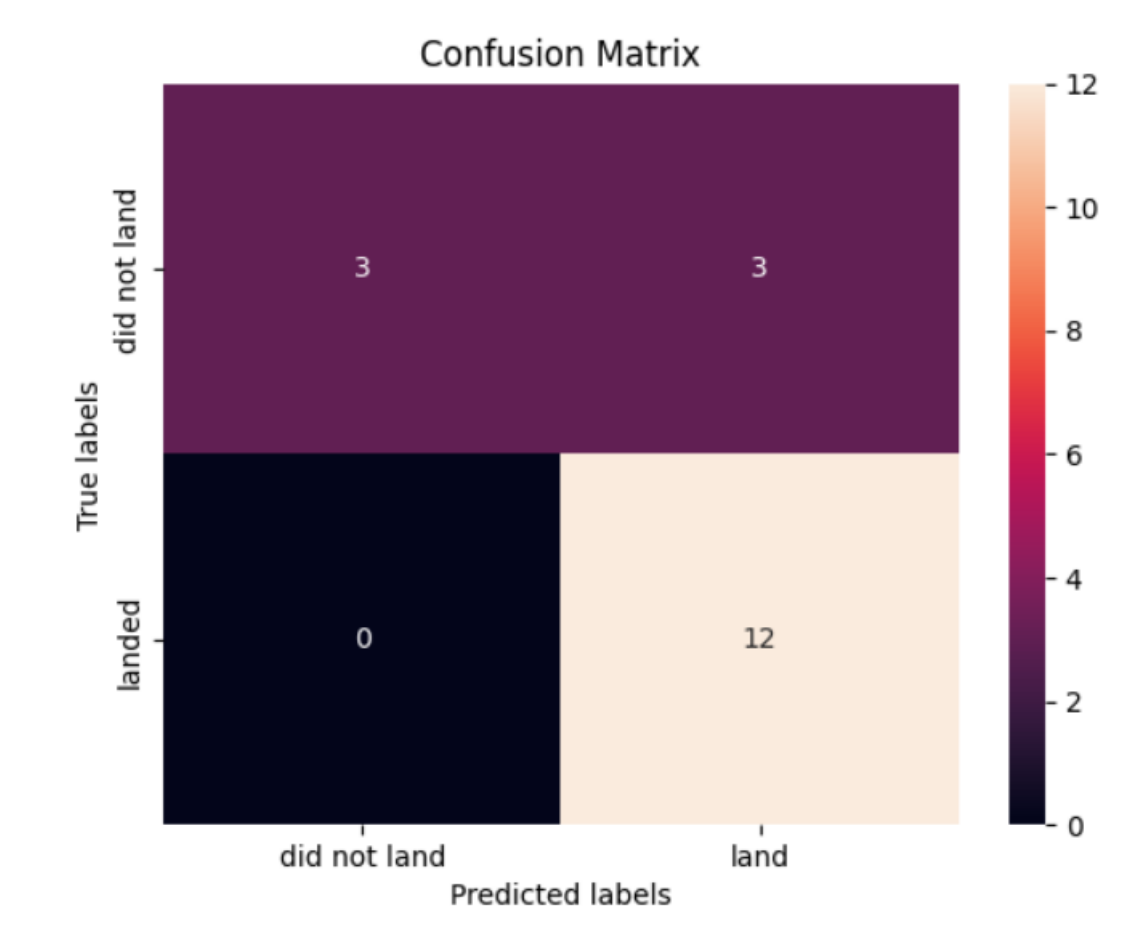
Predictive Analysis (Classification)

Classification Accuracy



- On test accuracy results are the same for all the four
- Accuracy in learning sample better for Decision Tree model

Confusion Matrix for Decision Tree Model



The confusion matrix for the decision tree classifier indicates that the model is able to predict target classes (launch outcomes).

It performs strongly for true positives, at the same time it is not sound for false positives (i.e could predict success for real failure cases)

Conclusions

- Success rates have been on rise from 2013 and achieved the plateau of ~ 80% in to 2020
- VLEO, ES-L1, GEO, HEO, SSO, VLEO orbit types have the highest success rate.
- The larger the number of flights performed at a particular launch site, the higher the current success rate at that site
- The developed ML predicting models are able to statistically soundly predicts outcomes. The Decision tree model is slightly better then others in terms of accuracy. So it is an optimum model for this task

Appendix

- Relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, etc… can be reached at the author's GitHub directory by the >>> URL.

Thank you!