IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sirikamon Nation
29/11/2022

Outline

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

Executive Summary

- Summary of methodologies
  - Data Collection via API and Web Scraping
  - Data Wrangling
  - Exploratory Data Analysis (EDA) with SQL
  - EDA with Data Visualization
  - Building a Data Visualization (map) with Folium
  - Building an Interactive Dashboard with Plotly Dash
  - Predictive Analysis (Classification)
- Summary of all results
  - EDA results
  - Interactive Maps and Dashboard
  - Predictive results

Introduction

- Project background and context
  - The purpose of this project is to predict if the Falcon 9 first stage with land successfully. SpaceX advertises on its website that Falcon 9 rocket launches with a cost of 62 million dollars when other providers cost upward of 165 million dollars each. This 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.
- Problems you want to find answers
  - What are the factor for successful landing?
  - What are the relationship between each variables on success landing result?
  - What are the condition that will allow SpaceX to achieve the best landing success rate?

Section 1

Methodology

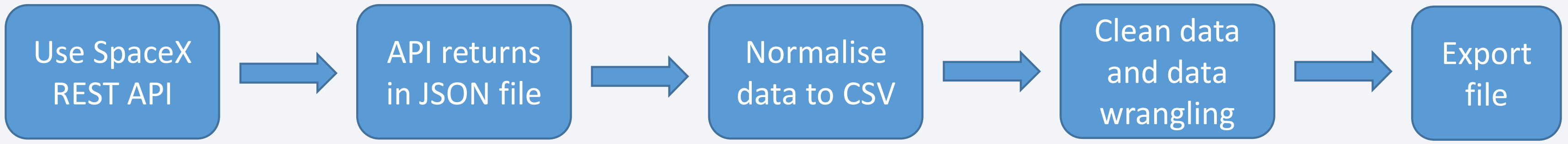
Methodology

Executive Summary

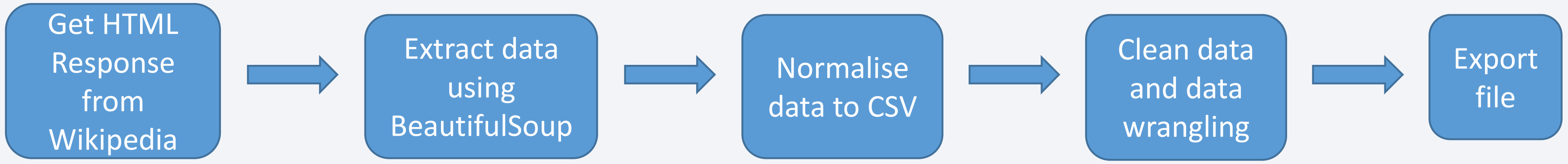
- Data collection methodology:
  - SpaceX REST API
  - Web Scrapping from Wikipedia
- Perform data wrangling
  - Dropping unnecessary columns
  - One Hot Encoding for classification model
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - LR, KNN, SVM and DT models have been built and evaluated for the best classification

Data Collection

- Datasets are collected from the SpaceX REST API and Web Scrapping Wikipedia
  - The information obtained by the API are rocket used, payload delivered, launch specifications, landing specifications and landing outcome.
    - The SpaceX REST API URL is api.spacexdata.com/v4



  - The information obtained by Web Scrapping is Falcon 9 launch data, using BeautifulSoup
    - URL is https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922


Get HTML Response from Wikipedia
Extract data using BeautifulSoup
Normalise data to CSV
Clean data and data wrangling
Export file

Data Collection – SpaceX API

1. Getting Respond from API

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

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

2. Converting Response to JSON file

```
response = requests.get(static_json_url)
response.json()
data = pd.json_normalize(response.json())
```

3. Transforming data

```
getBoosterVersion(data)
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

4. Creating dictionary and dataframe

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

5. Creating dataframe

```
df = pd.DataFrame({key:pd.Series(value,dtype='object')for key, value in launch_dict.items()})
```

6. Filter the dataframe

```
data_falcon9 = df.loc[df['BoosterVersion']!='Falcon 1']
```

7. Dealing with missing value

```
data_falcon9.isnull().sum()
```

```
PayloadMass_mean = data_falcon9['PayloadMass'].mean()

data_falcon9['PayloadMass'].replace(np.nan, PayloadMass_mean, inplace=True)
```

8. Export to CSV file

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

Click here for code

Data Collection - Scraping

1. Getting Response from HTML

```
response = requests.get(static_url)
response.status_code
print(response.content)
```

2. Creating BeautifulSoup Object

```
BeautifulSoup = BeautifulSoup(response.text, 'html5lib')
```

3. Finding tables

```
html_tables = BeautifulSoup.find_all(name='table')
```

4. Getting column names

```
col_list = first_launch_table.find_all('th')
for i in col_list:
    col_name = extract_column_from_header(i)
    if col_name != None and len(col_name) > 0:
        column_names.append(col_name)
```

5. Creating dictionary

```
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']=[]
```

6. Adding data to keys

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(BeautifulSoup.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()
```

See completed code from link

7. Creating dataframe from dictionary

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

8. Export to CSV file

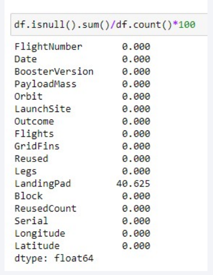
```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Click here for code

Data Wrangling

- In the dataset, there are several cases where the booster did not land successfully
  - True Ocean, True RTLS, True ASDS mean the mission has been successfully
  - False Ocean, False RTLS, False ASDA mean the mission was failure
- Convert those out comes into Training Labels with 1 means the booster successfully landed, 0 means it was unsuccessful

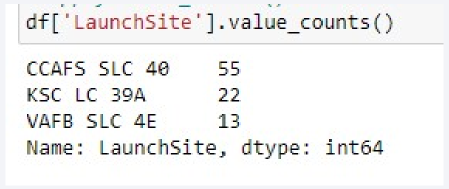
1. Check null values



```
df.isnull().sum()/df.count()*100
```

```
FlightNumber       0.000
Date               0.000
BoosterVersion     0.000
PayloadMass        0.000
Orbit              0.000
LaunchSite         0.000
Outcome            0.000
Flights            0.000
GridFins           0.000
Reused             0.000
Legs               0.000
LandingPad        40.625
Block              0.000
ReusedCount        0.000
Serial             0.000
Longitude          0.000
Latitude           0.000
dtype: float64
```

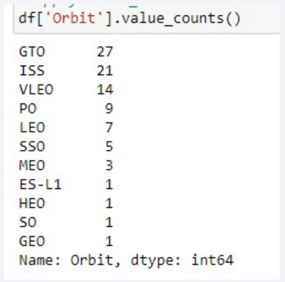
2. Calculate the number of launches on each site



```
df['LaunchSite'].value_counts()
```

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

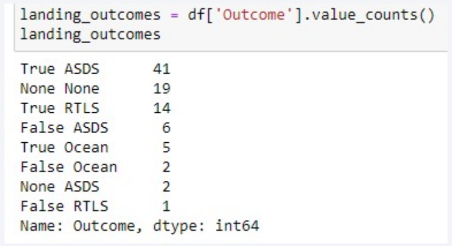
3. Calculate the number and occurrence of each orbit



```
df['Orbit'].value_counts()
```

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

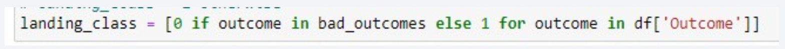
4. Calculate the number and occurrence of mission outcome per orbit type



```
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

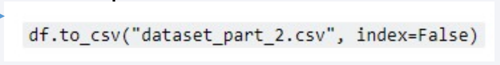
```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

5. Create a landing outcome label from outcome column



```
landing_class = [0 if outcome in bad_outcomes else 1 for outcome in df['Outcome']]
```

6. Export to CSV file



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

Click here for code

EDA with Data Visualization

Charts were plotted as follow:

- **Scatter plot**
  - Flight Number vs Payload
  - Flight Number vs Launch Site
  - Payload vs Launch Site
  - Flight Number vs Orbit type
  - Payload vs Orbit type

Scatter plot can determine whether or not two variables have a relationship or correlation.

- **Bar Graph**
  - Success rate vs Orbit type

Bar graphs used to represent the relationship between numeric and categoric variables.

- **Line Graph**
  - Launch success trend

Line graphs show how the values of variables change overtime and their trend.

Click here for code

EDA with SQL

- SQL queries performed include:
  - 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
  - List the records which will display the month names, failure landing_outcomes in drone ship, booster versions, launch_site for the months in year 2015
  - Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order

Click here for code

Build an Interactive Map with Folium

- Folium map object is a map centered on NASA Johnson Space Center at Houson, Texas
  - Blue circle at NASA Johnson Space Center's coordinate with label showing it name, using **folium.Circle** and **folium.map.Marker**
  - Red circles at each launch site coordinates with label showing launch site name, using **folium.Circle**, **folium.Marker** and **folium.features.DivIcon**
  - Grouping of points in a cluster to display multiple and different information for the same coordinates, using **MarkerCluster**
  - Markers to show successful (green) and unsuccessful (red) landings, using **folium.Marker** and **folium.Icon**
  - Add mouse position on the map to get coordinate for a mouse over the point on the map, using **MousePosition**
  - Markers to show distance between launch site to key location (coastline, railway, highway and city) using **folium.DivIcon**, **folium.map.Marker** and **folium.PolyLine**
- These objects are created in order to understand better the problem and the data. We can show all launch sites easily, their surroundings and the number of successful landing.

Click here for code

Build a Dashboard with Plotly Dash

- Dashboard contains the following plots/graphs and interactions:
    - **Dropdown** which allows a user to choose the launch site or all launch sites (dash_core_components.Dropdown)
    - **Pie chart** which shows the total success and the total failure for the launch site chosen with the dropdown component (plotly.express.pie)
    - **Rangeslider** which allows a user to select a payload mass in a fixed range (dash_core_components.RangeSlider)
    - **Scatter chart** which shows the relationship between two variables, in particular Success vs Payload Mass (plotly.express.scatter)

Click here for code

Predictive Analysis (Classification)

- Data Preparation
  - Load dataset
  - Normalize data
  - Split data into training and test sets
- Model Preparation
  - Selection of machine learning algorithms
  - Set parameters for each algorithm to GridSearchCV
  - Training GridSearchModel models with training dataset
- Model Evaluation
  - Get best hyperparameters for each type of model
  - Calculate accuracy for each model with test dataset
  - Plot Confusion Matrix
- Model Comparison
  - Comparison of models according to their accuracy
  - The model with the best accuracy will be chosen

Click here for code

Results

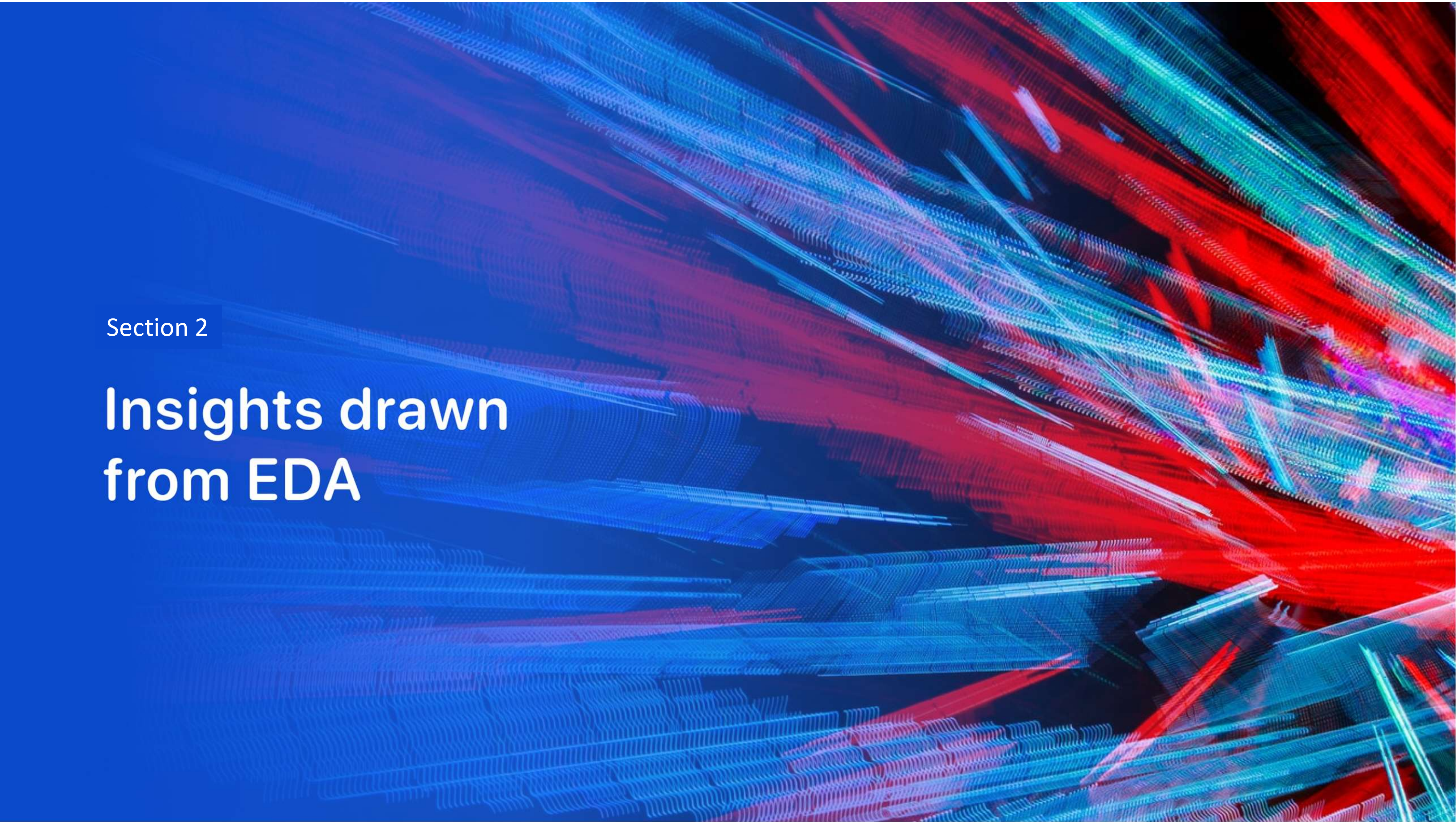
Exploratory data analysis results

Interactive analytics demo in screenshots

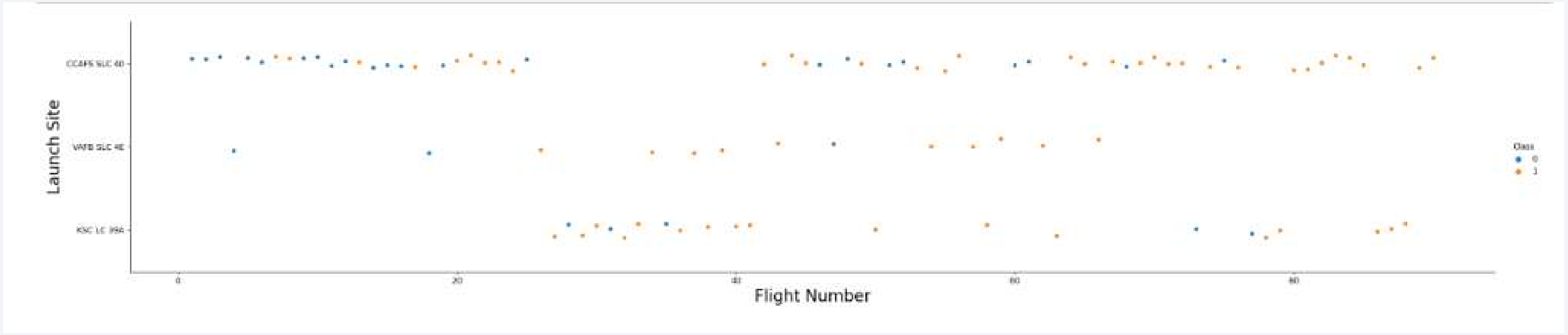
Predictive analysis results

Section 2

Insights drawn from EDA

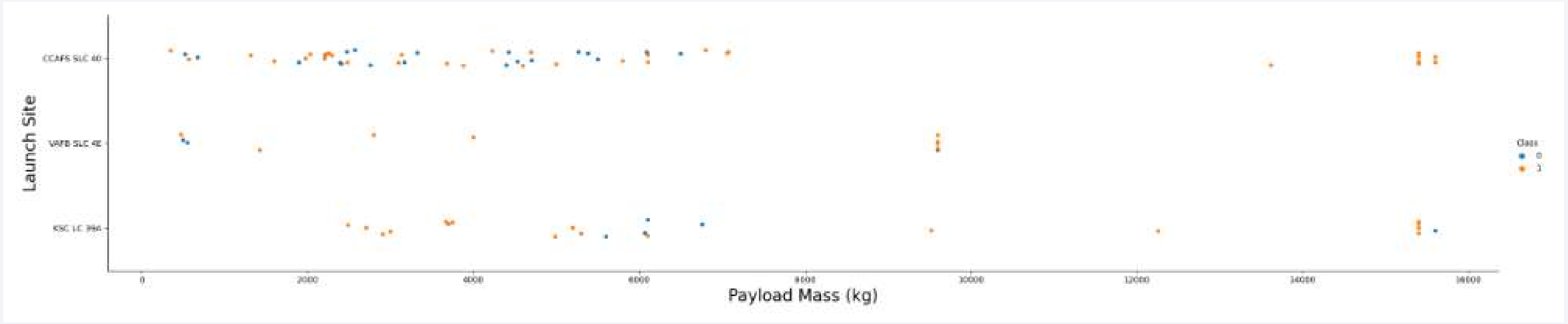


Flight Number vs. Launch Site



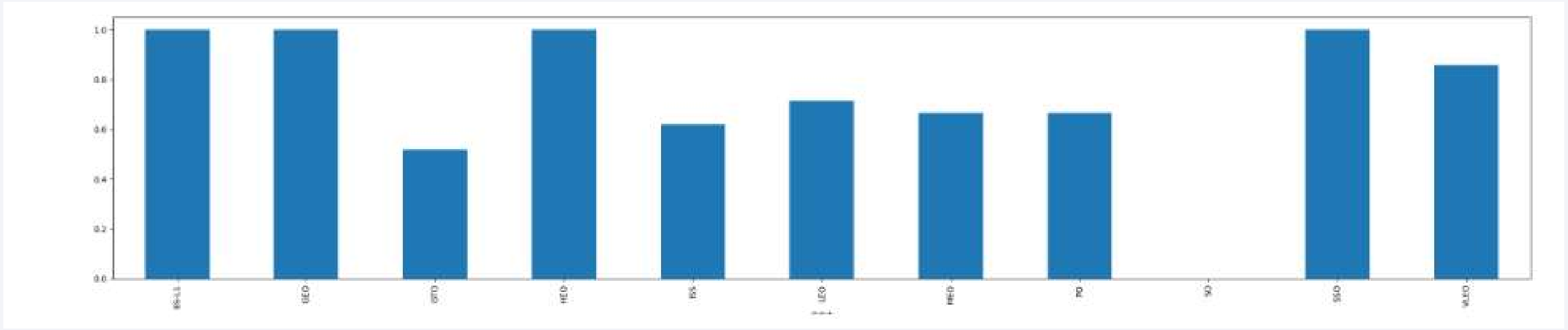
For each site, the success rate is increasing.

Payload vs. Launch Site



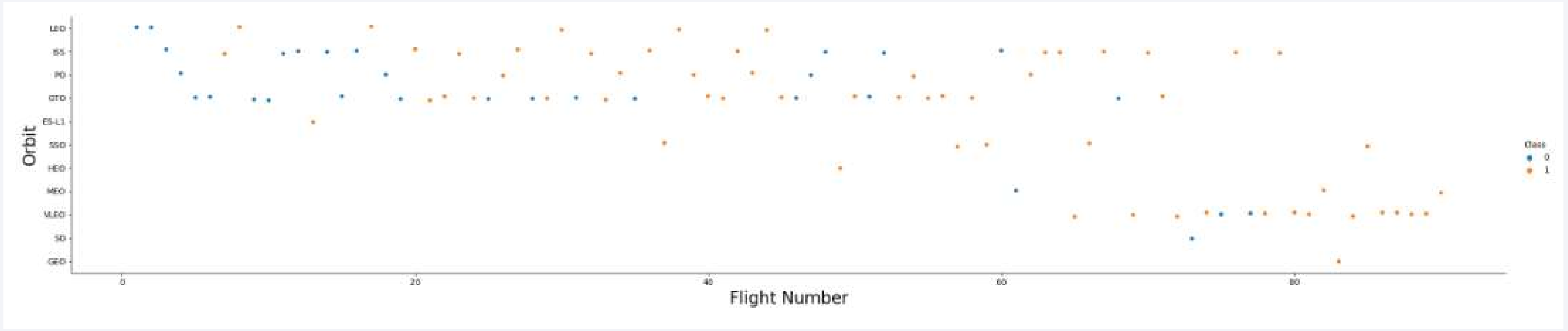
The VAFB-SLC launch site, there are no rockets launched for heavy payload mass that greater than 10000 kg.

Success Rate vs. Orbit Type



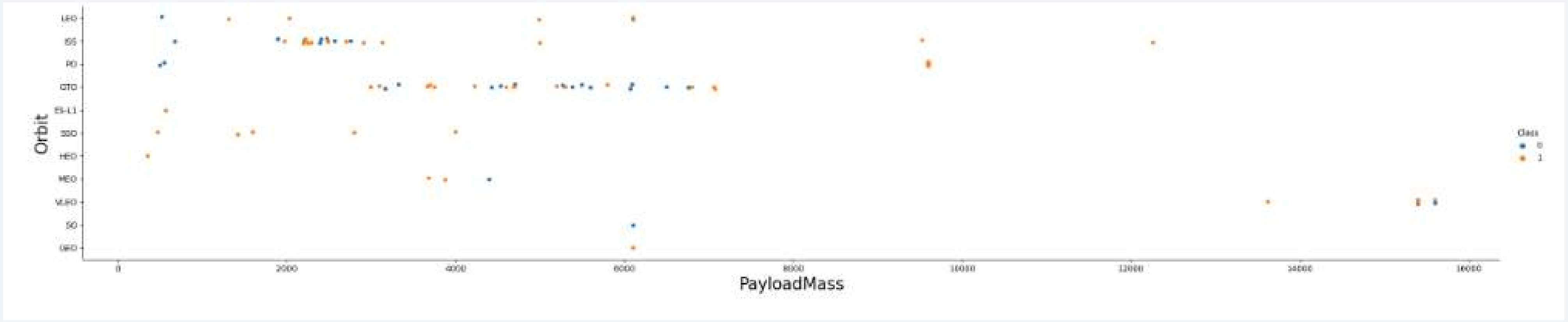
The Orbit Type ES-L1, GEO, HEO and SSO have high success rate.

Flight Number vs. Orbit Type



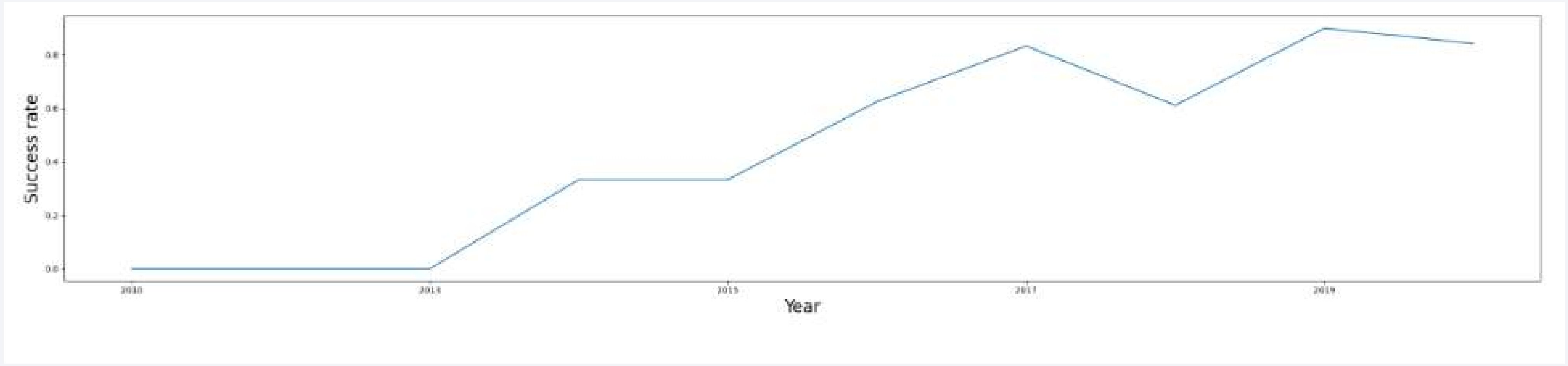
The success in the LEO orbit appears related to the number of flights, on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. Orbit Type



- There are more successful landing rate with heavy payloads for Polar, LEO and ISS orbits.
- It is hard to distinguish well for GTO orbit as both successful and unsuccessful landing rate are both appeared.

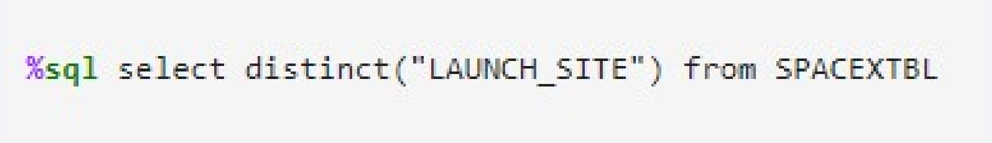
Launch Success Yearly Trend



The success rate kept increasing since 2013 till 2020.

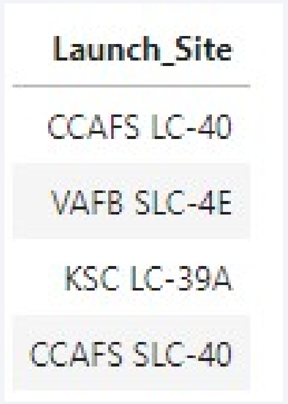
All Launch Site Names

SQL Query



```
%sql select distinct("LAUNCH_SITE") from SPACEXTBL
```

Result



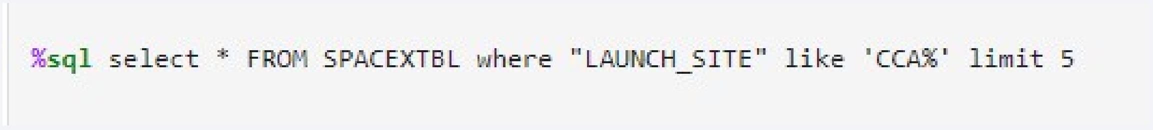
Launch_Site
CCAFS LC-40
VAFB SLC-4E
KSC LC-39A
CCAFS SLC-40

Explanation

Using **distinct** in query can get unique value

Launch Site Names Begin with 'CCA'

SQL Query

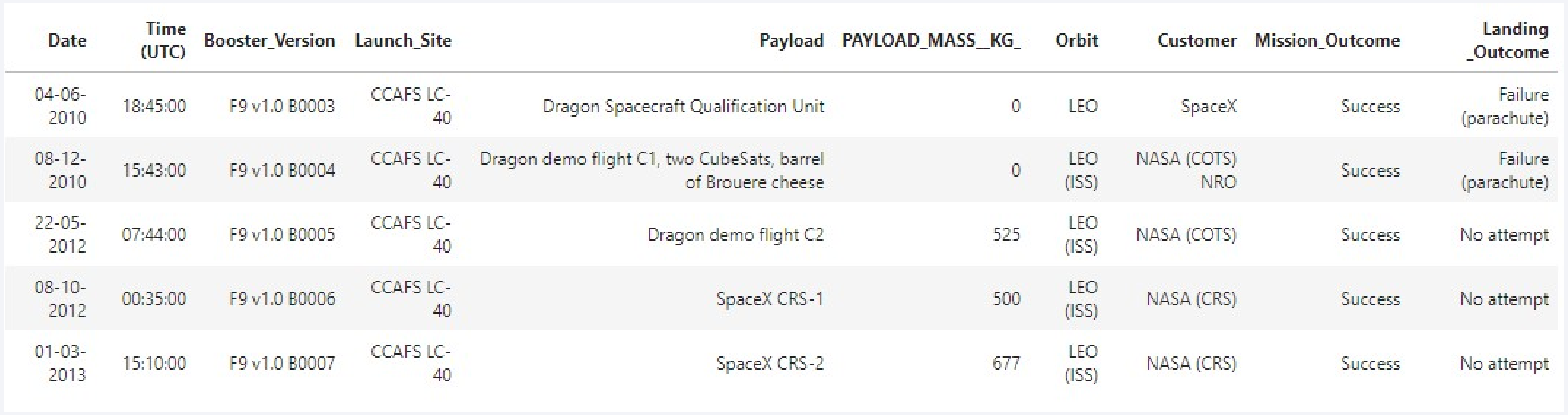


```
%sql select * FROM SPACEXTBL where "LAUNCH_SITE" like 'CCA%' limit 5
```

Explanation

- **where** function followed by **like** function filters launch sites that contain substring CCA
- **Limit 5** for showing 5 rows of record

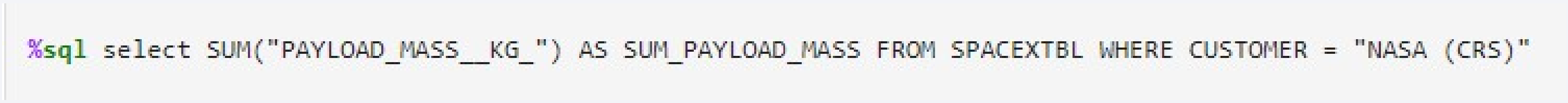
Result



Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	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)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

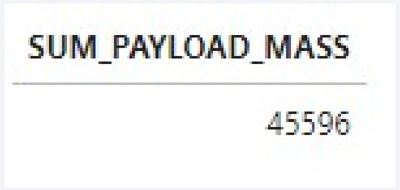
Total Payload Mass

SQL Query



```
%sql select SUM("PAYLOAD_MASS__KG_") AS SUM_PAYLOAD_MASS FROM SPACEXTBL WHERE CUSTOMER = "NASA (CRS)"
```

Result



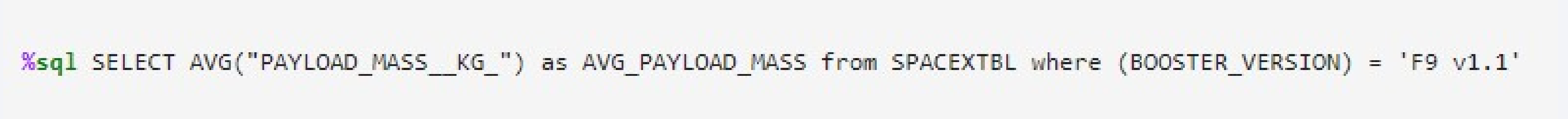
SUM_PAYLOAD_MASS
45596

Explanation

Using SUM to get the sum of all payload masses and using WHERE function to get customer in NASA (CRS).

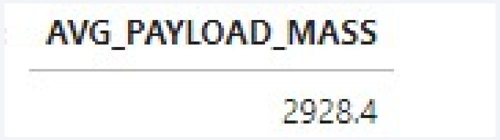
Average Payload Mass by F9 v1.1

SQL Query



```
%sql SELECT AVG("PAYLOAD_MASS__KG_") as AVG_PAYLOAD_MASS from SPACEXTBL where (BOOSTER_VERSION) = 'F9 v1.1'
```

Result



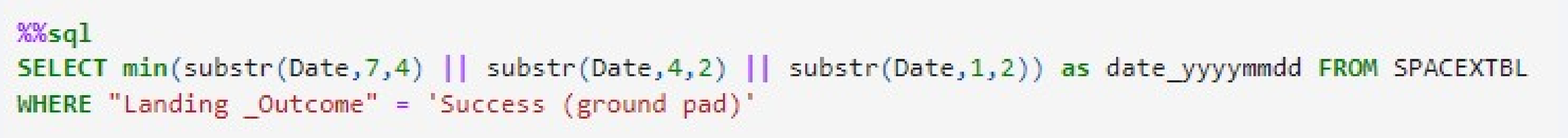
AVG_PAYLOAD_MASS
2928.4

Explanation

Using **AVG** returns the average of all payload masses where the booster version contains the substring F9 v1.1

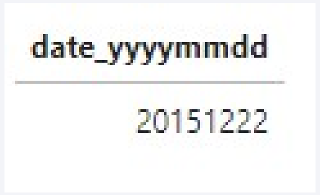
First Successful Ground Landing Date

SQL Query



```
%%sql
SELECT min(substr(Date,7,4) || substr(Date,4,2) || substr(Date,1,2)) as date_yyyymmdd FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (ground pad)'
```

Result



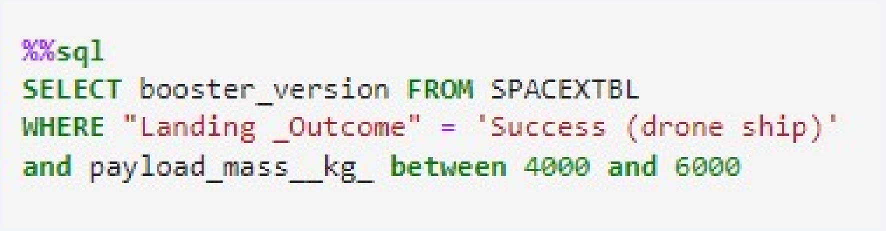
date_yyyymmdd
20151222

Explanation

To get the first successful date by using min because the first date is the same with the minimum date.

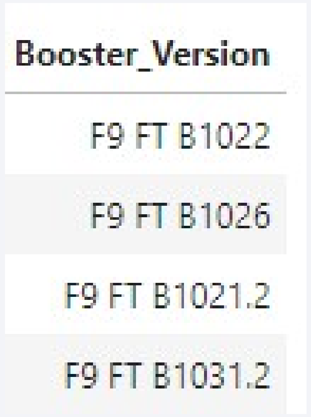
Successful Drone Ship Landing with Payload between 4000 and 6000

SQL Query



```
%%sql
SELECT booster_version FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Success (drone ship)'
and payload_mass__kg_ between 4000 and 6000
```

Result



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

Explanation

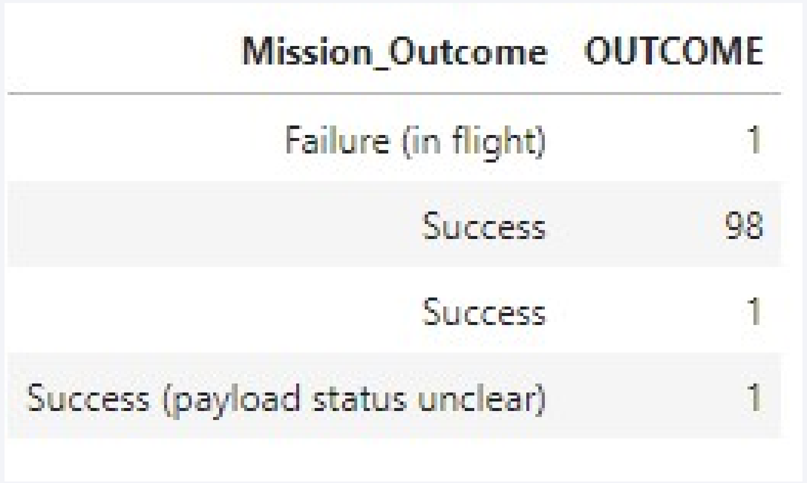
WHERE and **AND** function filter the dataset which is the booster version where landing outcome was successful and payload mass is between 4000 and 6000 kg.

Total Number of Successful and Failure Mission Outcomes

SQL Query

```
%sql SELECT MISSION_OUTCOME, COUNT(MISSION_OUTCOME) AS OUTCOME FROM SPACEXTBL GROUP BY MISSION_OUTCOME
```

Result



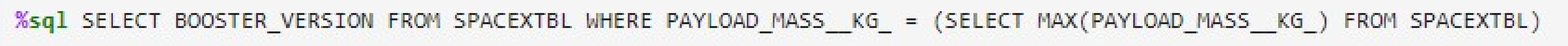
Mission_Outcome	OUTCOME
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Explanation

COUNT function in conjunction with **GROUP BY** characterize data under various groupings. A combination of same values (on a column) will be treated as an individual group.

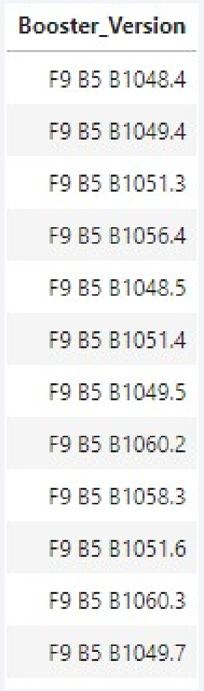
Boosters Carried Maximum Payload

SQL Query



```
%sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
```

Result



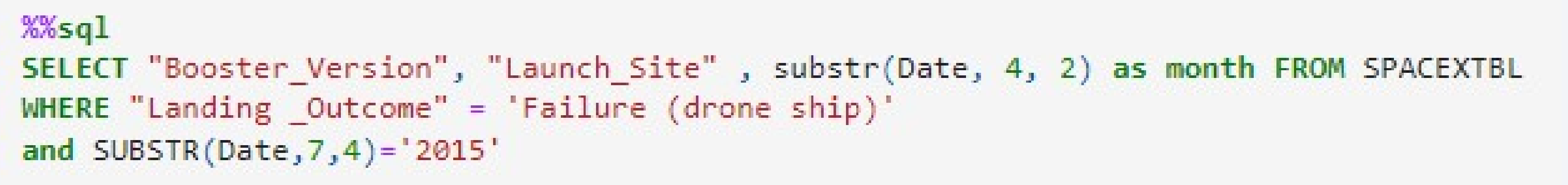
Booster_Version
F9 B5 B1048.4
F9 B5 B1049.4
F9 B5 B1051.3
F9 B5 B1056.4
F9 B5 B1048.5
F9 B5 B1051.4
F9 B5 B1049.5
F9 B5 B1060.2
F9 B5 B1058.3
F9 B5 B1051.6
F9 B5 B1060.3
F9 B5 B1049.7

Explanation

The main query uses subquery results and returns booster version with the heaviest payload mass. The subquery filters data (the heaviest payload mass) by using **WHERE** function and **MAX** function.

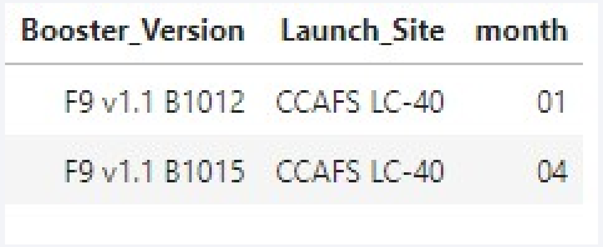
2015 Launch Records

SQL Query



```
%%sql
SELECT "Booster_Version", "Launch_Site" , substr(Date, 4, 2) as month FROM SPACEXTBL
WHERE "Landing _Outcome" = 'Failure (drone ship)'
and SUBSTR(Date,7,4)='2015'
```

Result



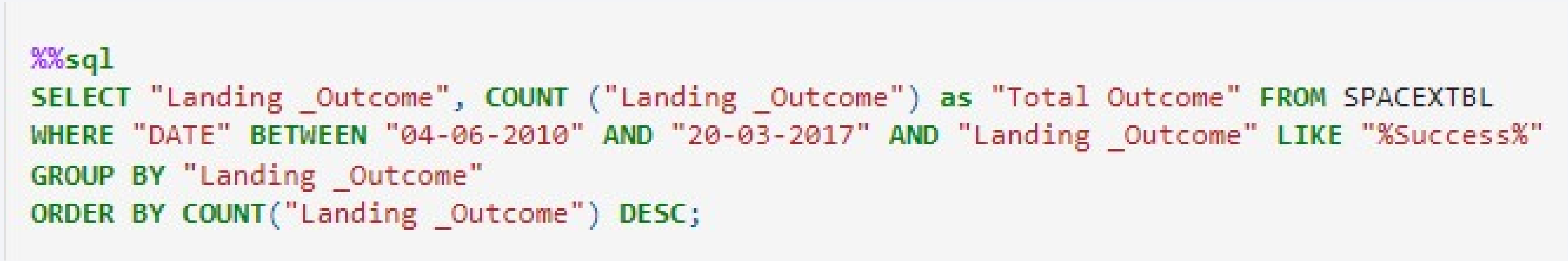
Booster_Version	Launch_Site	month
F9 v1.1 B1012	CCAFS LC-40	01
F9 v1.1 B1015	CCAFS LC-40	04

Explanation

This query returns month, booster version and launch site where landing was unsuccessful and landing date took place in 2015. **Substr** function process date in order to take month or year. Substr(Date, 4 , 2) shows month. Substr(Date,7,4) shows year.

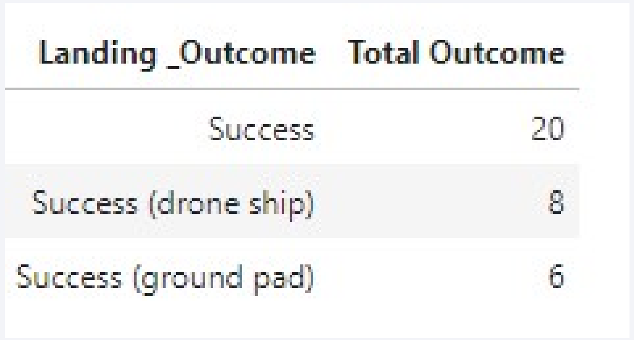
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

SQL Query



```
%%sql
SELECT "Landing _Outcome", COUNT ("Landing _Outcome") as "Total Outcome" FROM SPACEXTBL
WHERE "DATE" BETWEEN "04-06-2010" AND "20-03-2017" AND "Landing _Outcome" LIKE "%Success%"
GROUP BY "Landing _Outcome"
ORDER BY COUNT("Landing _Outcome") DESC;
```

Result



Landing _Outcome	Total Outcome
Success	20
Success (drone ship)	8
Success (ground pad)	6

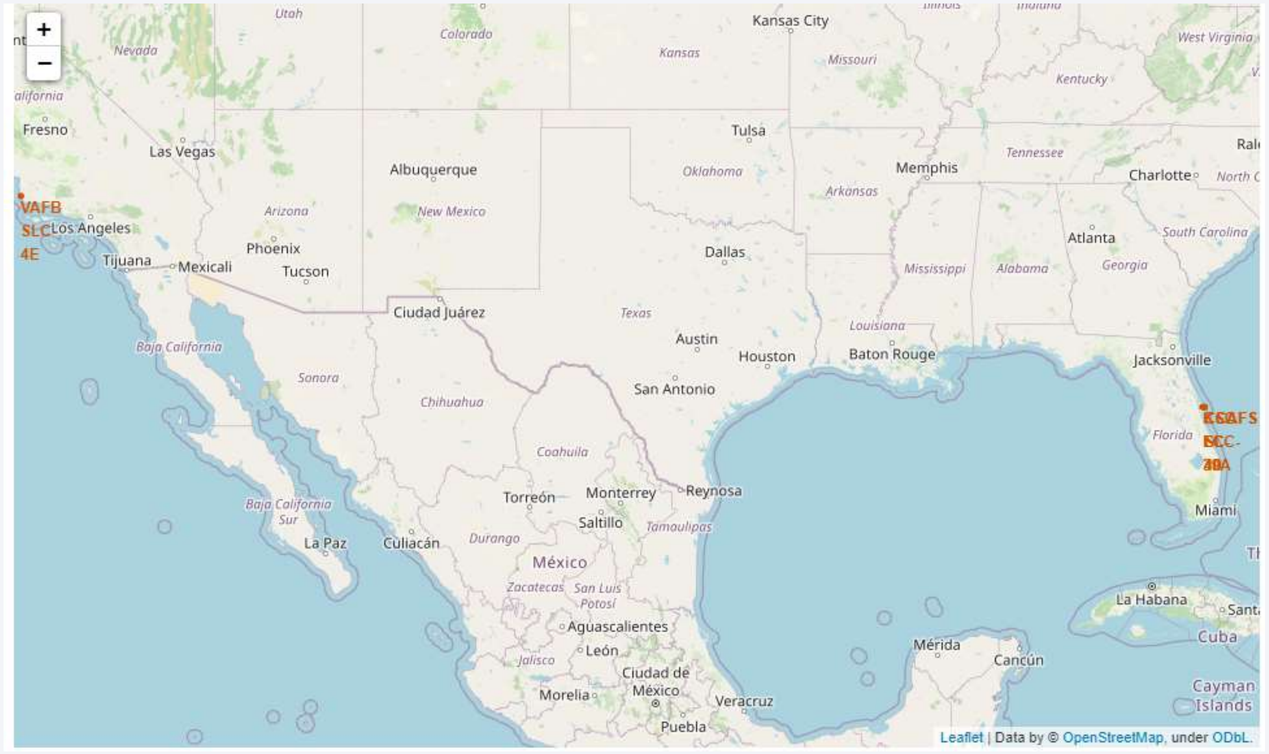
Explanation

This query returns landing outcomes and their count where mission was success between 04/06/2010 and 20/30/2017. The GROUP BY function groups the results by landing outcome and ORDER BY COUNT DESC returns results in decreasing order.

Section 3

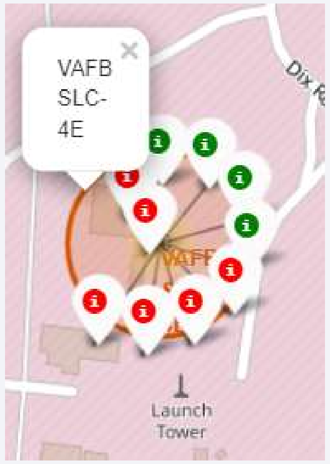
Launch Sites Proximities Analysis

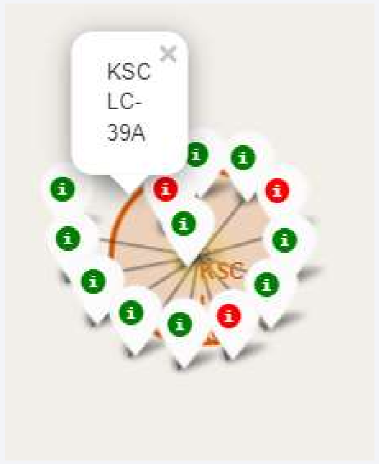
All Launch Site Location

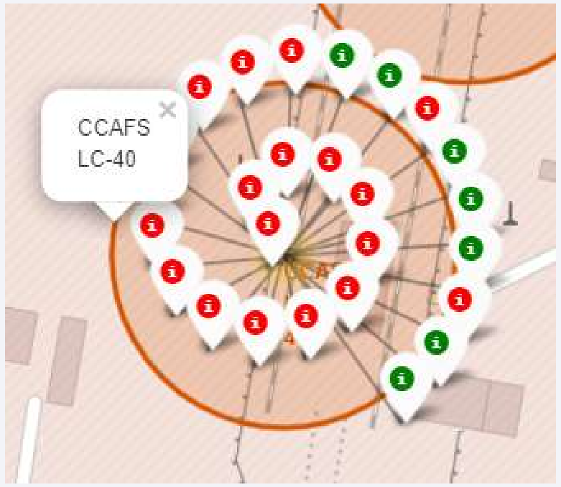


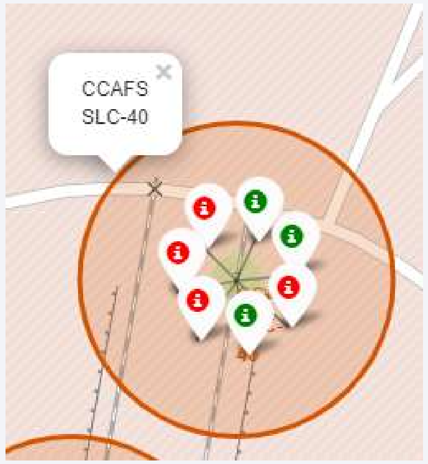
All launch sites are in proximity to the Equator line and they are very close proximity to the coast

Colour-labeled markers with outcomes



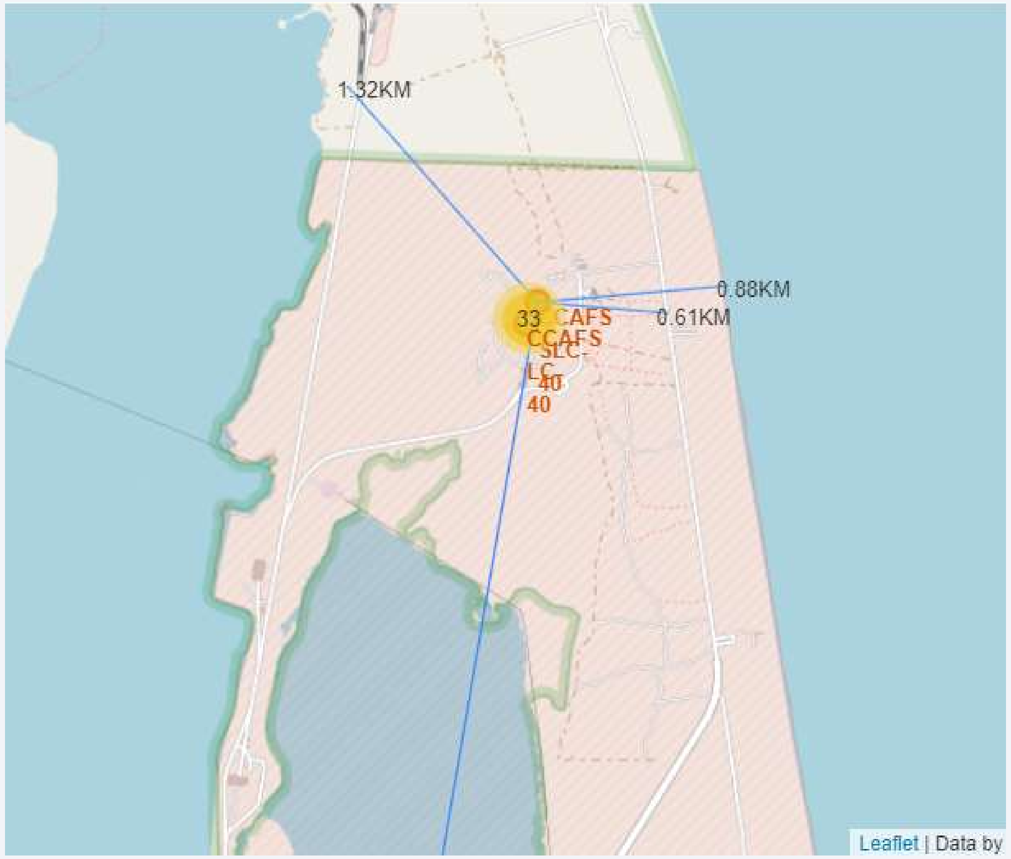




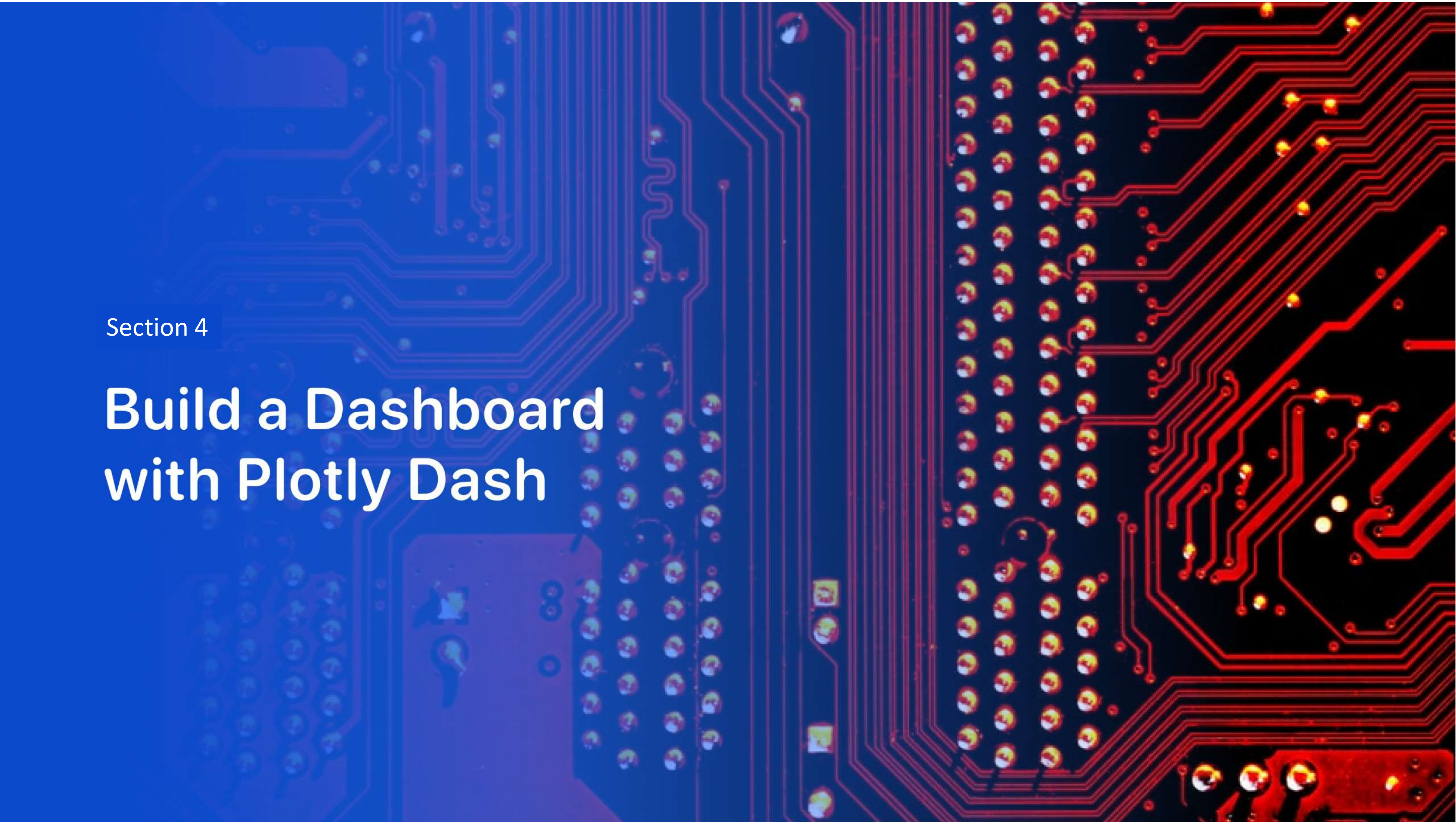


Green marker represents successful launch and Red marker represents unsuccessful launch. The launch site which have relatively high success rates is **KSC LC-39A**.

Distances between CCAFS SCL-40 to its proximities



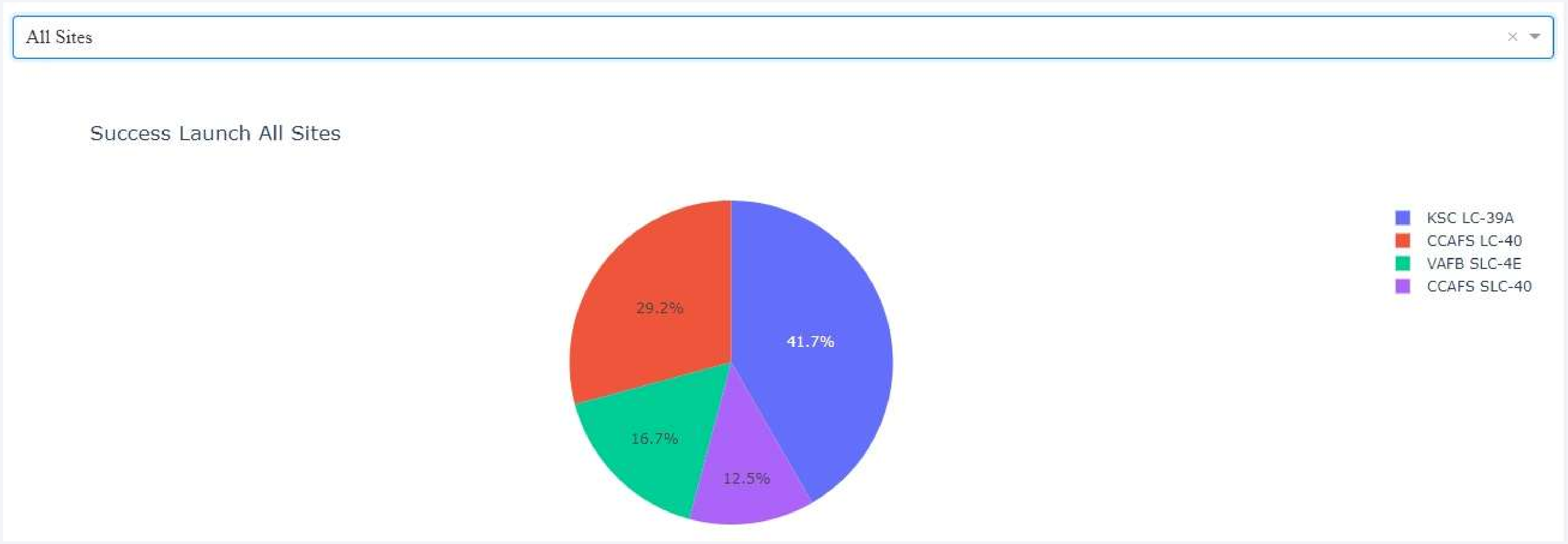
- CCAFS SCL-40 launch site are in close proximity to railways (1.32 km), highways (0.61 km) and coastline (0.88 km)
- This launch site keep distance away from city such as Melbourne



Section 4

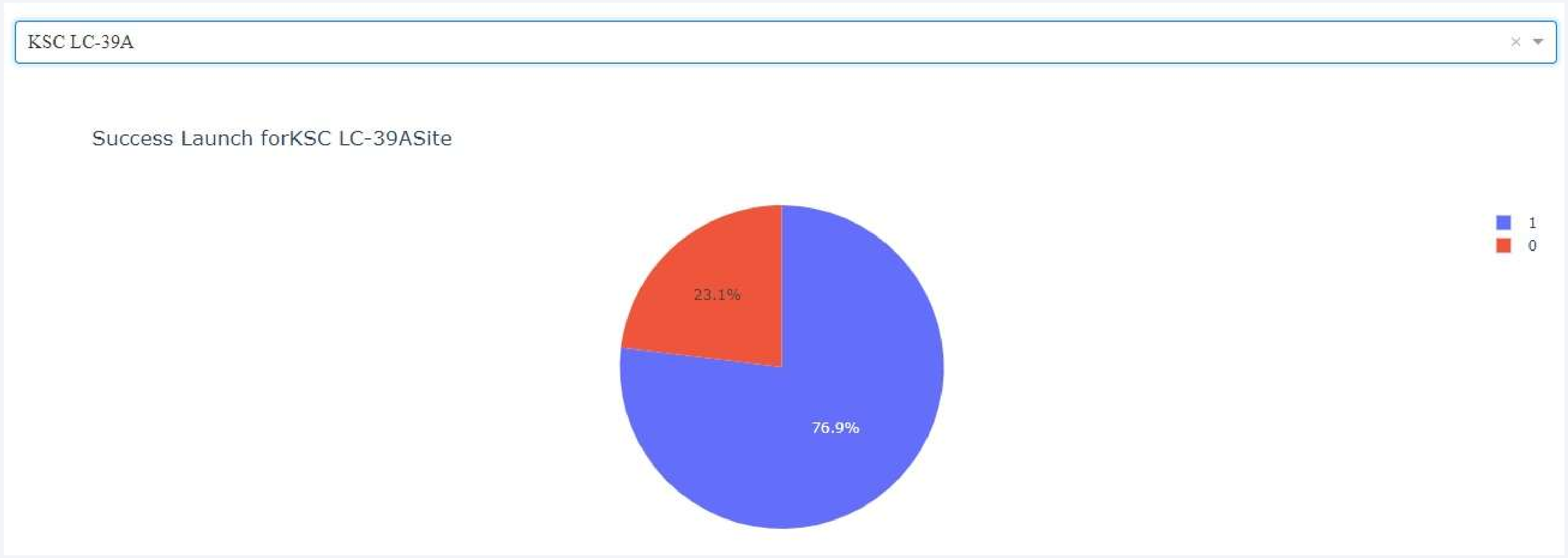
Build a Dashboard with Plotly Dash

Total success for all sites


All Sites

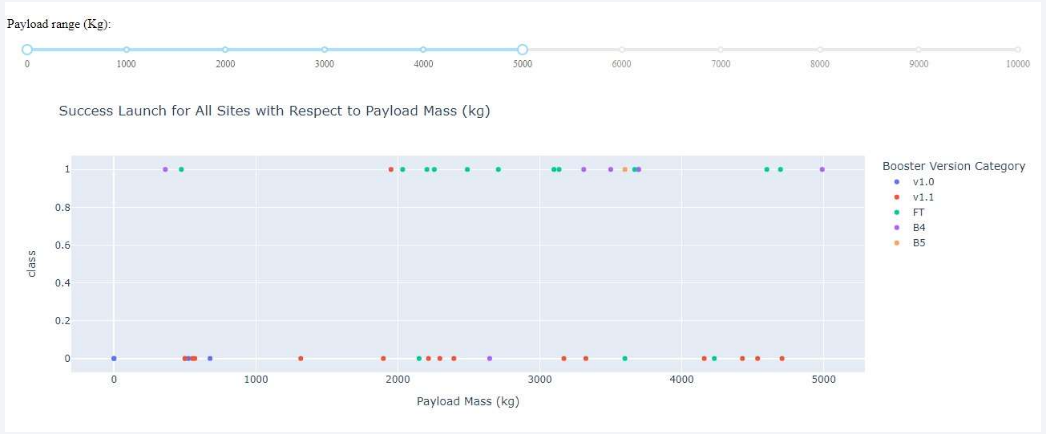
- KSC LC-39A has the highest launch success rate
- CCAFS SLC-40 has the lowest launch success rate

Launch Success rate for KSC LC-39A

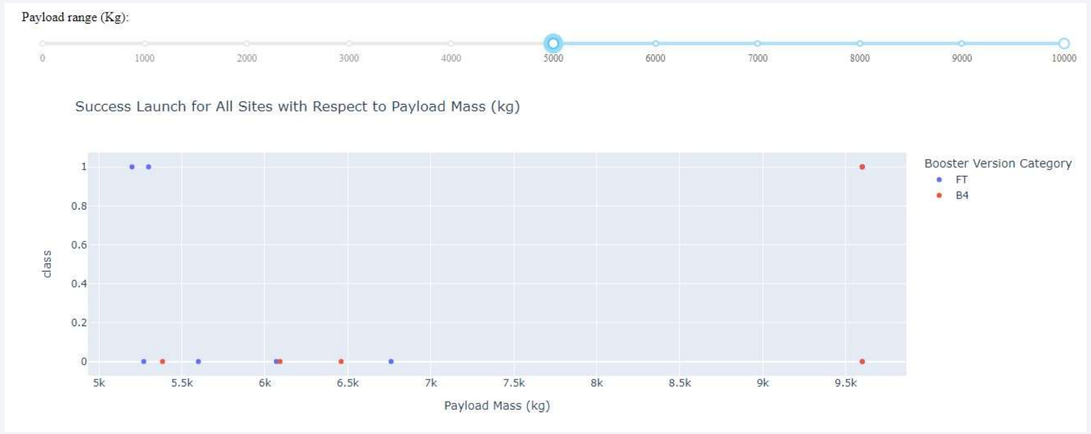


KSC LC-39A has achieved 76.9% success rate and 23.1% failure rate

Payload Mass vs Launch outcome for all sites



Low weight payload: 0 – 5000 kg



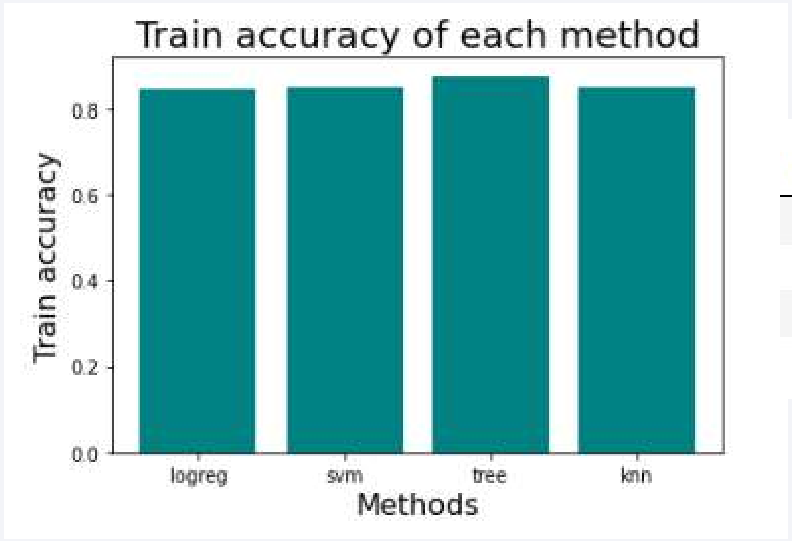
Heavy weight payload: 5000 - 10000 kg

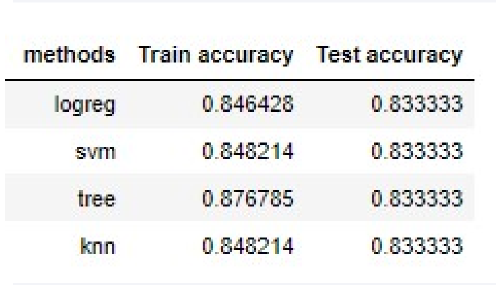
- The low weight payload has the highest launch success rate than the heavy weight payload.
- FT booster version has the largest launch success rate.

Section 5

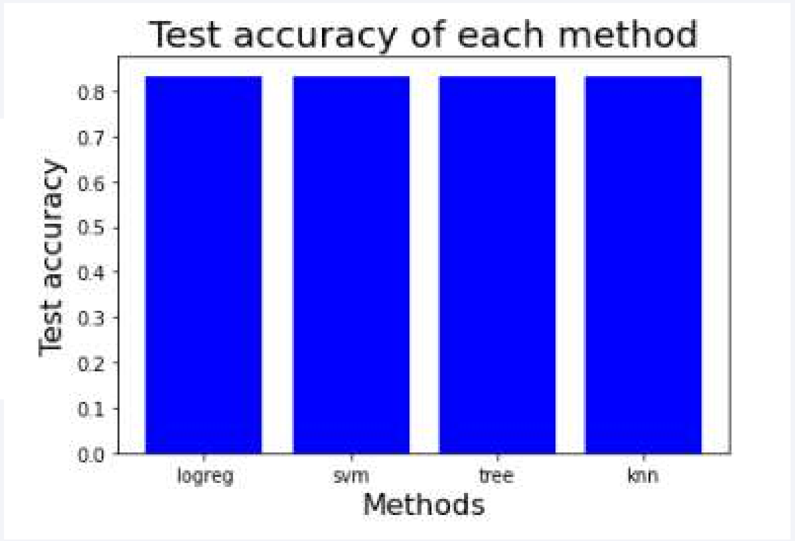
Predictive Analysis (Classification)

Classification Accuracy





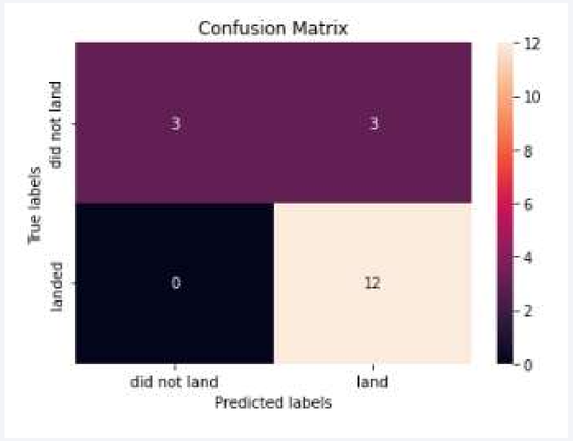
methods	Train accuracy	Test accuracy
logreg	0.846428	0.833333
svm	0.848214	0.833333
tree	0.876785	0.833333
knn	0.848214	0.833333



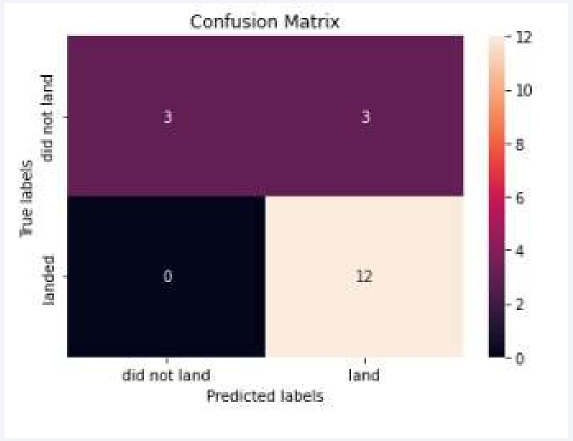
In all methods have the same accuracy in test, so all methods should perform well. However, in Tree method has the highest accuracy in train so it can be the preferred method.

Confusion Matrix

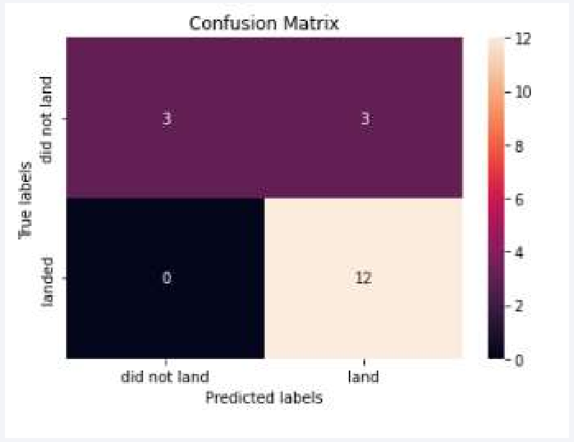
Logistic regression



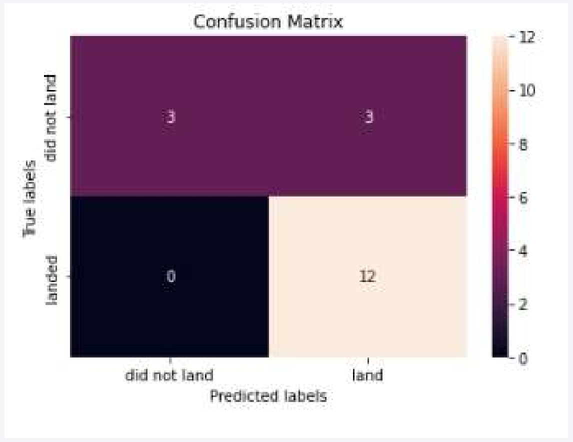
SVM



Decision Tree



KNN



Each methods have the same test accuracy and the Confusion Matrixes are identical.

Conclusions

- The orbit with the high success rate is ES-L1, GEO, HEO and SSO.
- The low weight payload (0 – 5000 kg) has the highest launch success rate than the heavy weight payload (5000 – 10000 kg).
- With the heavy payload, Polar, LEO and ISS orbits have more success landing rate.
- SC LC-39A has the highest launch success rate in all the sites which is 76.9%.
- All algorithms are good to use as they have identical test accuracy, however the Decision Tree Algorithm is chosen as the best model because it has a better train accuracy.

Appendix

- All the codes, SQL queries, chart and Notebook output for this project can be found from the link below.

Coursera Applied Data Science Capstone

Thank you!