IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ali Khosravi
5 June 2022

Outline

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

Executive Summary

In this project we have used Wikipedia website as our main source of information to extract spaceX launch data.
Using API's and web scraping, we will collect and compile our the data into csv files and eventually using data wrangling techniques we will refine and transform the data into a data frame
Then we will use different tools, such as data visualisation, SQL querries, folium maps and plotly dashboards to obtain various forms of data with regards to SpaceX launches and how did they perform over time.

After all above analysis, we can clearly see that over time SpaceX tends to be more successful in their mission and has become capable of launching heavier payloads. We also can identify that there are 3 main orbits that SpaceX has been launching satellites to aside serving ISS. Also, looking at the location of launch sites, it is evident that SpaceX is only operates from coastal lines of the US.

Introduction

SpaceX business model pivots around the fact that they are capable of launching rockets into the space with substantially lower cost compare to other agencies. The main reason behind this huge success is their Falcon 9 first stage rocket lands back on earth in tact, enabling it to be reused for another launch after a minimum refurbishment.

Our main goal here is to identify key decisive factors in success of SpaceX attempts to land a rocket, as well as gain some insight about the overall process. At the end, we want to identify if we can find a model to predict future launches and identify pro's and con's of such model.

Section 1

Methodology

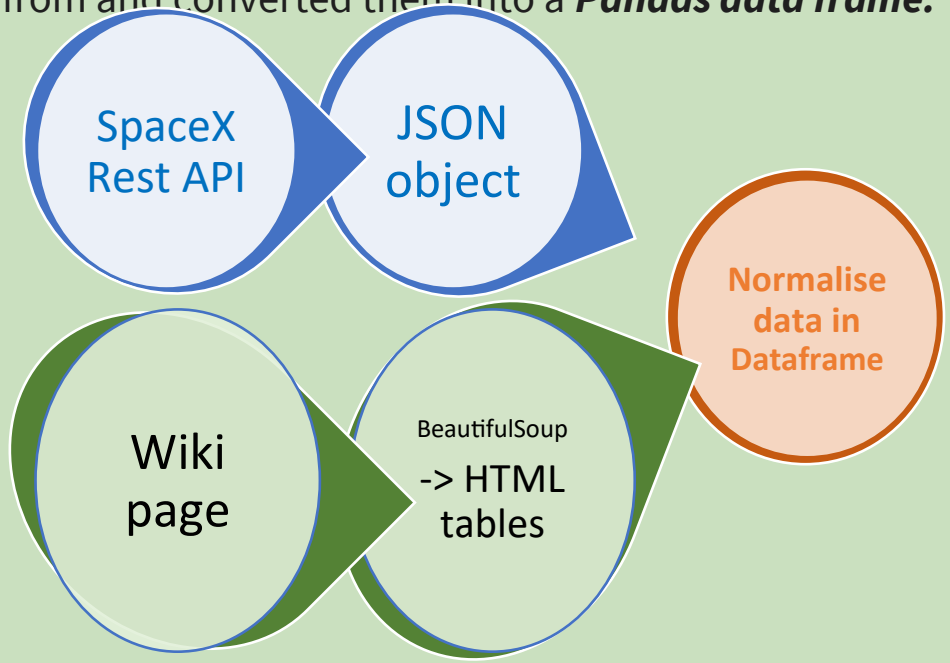
Methodology

Executive summary:

- Data collection methodology:
  - SpaceX rest API's
  - Web scraping using SpaceX Wikipedia website
- Data wrangling:
  - Cleaning data (removing nulls,etc.) and adding required columns,
  - One hot encoding for ML
- Exploratory data analysis (EDA) using visualisation and SQL
  - Using SQL queries in Jupyter notebooks
  - Using different plots and charts
- Interacting visual analytic
  - using folium maps to gain geographical insight
  - Using plotly dashboard to create an interactive reports
- Predictive analysis using classification models:
  - Split data into test and train datasets
  - sing grid search to identify best parameters for different ML models (LR,SVM,KNN,Classification Trees)
  - Running and evaluating each model to identify the best one.

Data Collection

- We used **SpaceX REST API** to gather SpaceX launch data. This API will give us data about launches, including information about the **rocket** used, **payload** delivered, **launch specifications**, **landing specifications**, and landing **outcome**.
- We will be working with the endpoint *api.spacexdata.com/v4/launches/past*. Then We will perform a get request using the requests library to obtain the launch data. Our response will be in the form of a JSON, specifically a list of **JSON objects**. Then, we convert this JSON to a ***dataframe***, using **json_normalize** function
- Another data source for obtaining Falcon 9 Launch data was **web scraping** related Wiki pages. we used **Python BeautifulSoup** package to web scrape some HTML tables that contain valuable Falcon 9 launch records. Then we **parsed the data** from and converted them into a ***Pandas data frame.***



Data Collection – SpaceX API

Data collection with SpaceX REST calls:

Obtaining Data:

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

response = requests.get(spacex_url)
```

Creating json object and normalising data

```
# Use json_normalize meethod to convert the json result into a dataframe
response = requests.get(static_json_url)
data=pd.json_normalize(response.json())
```

use the API to get info about the launches using the IDs given for each launch

```
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]

# We will remove rows with multiple cores because those are falcon rockets with 2 extra roc
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]

# Since payloads and cores are lists of size 1 we will also extract the single value in the
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])

# We also want to convert the date_utc to a datetime datatype and then extracting the date
data['date'] = pd.to_datetime(data['date_utc']).dt.date

# Using the date we will restrict the dates of the launches
data = data[data['date'] <= datetime.date(2020, 11, 13)]
data.head(2)
```

Call custom functions to grab info

```
# Call getBoosterVersion
getBoosterVersion(data)
```

Create a dict and turning it to 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}
```

- Github link:

https://github.com/akscb24/Coursera-Capstone-AK/blob/d2c04b41bed56565ff12497745f6767c3207873f/W1-1-Data%20Collection%20API%20Lab.ipynb

Data Collection - Scraping

Getting data from web using .get

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
response = requests.get(static_url)
```

Create a BautifulSoup object

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

Extract all column/variable names from the HTML table header

```
html_tables = soup.find_all(name='table')
column_names = []
for th in first_launch_table.find_all(name='th'):
    name = extract_column_from_header(th)
    if name is not None and len(name) > 0:

        column_names.append(name)
```

Create a data frame by parsing the launch HTML tables

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']

# Let's initial the launch_dict with ea
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']=[]
```

Fill up the ***launch_dict*** 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 collap
   # 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
```

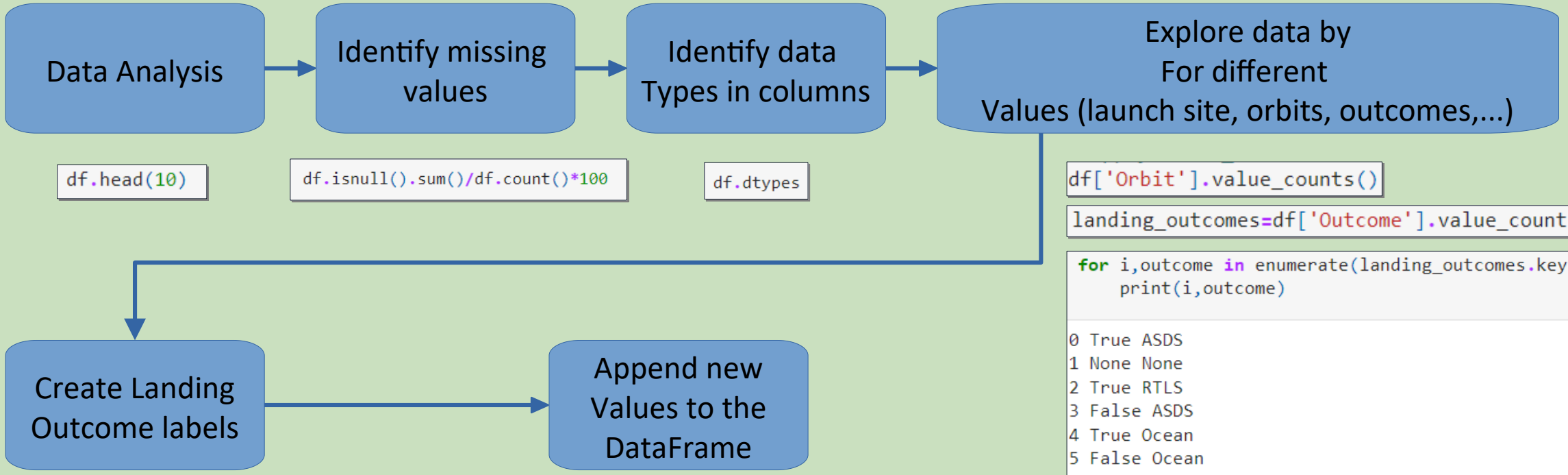
Convert to DataFrame

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

Git hub link

https://github.com/akscb24/Coursera-Capstone-AK/blob/6f773f608f6e97443e11e0a8018d8ec1a5411626/W1-2-Data Collection with Web Scraping lab.ipynb

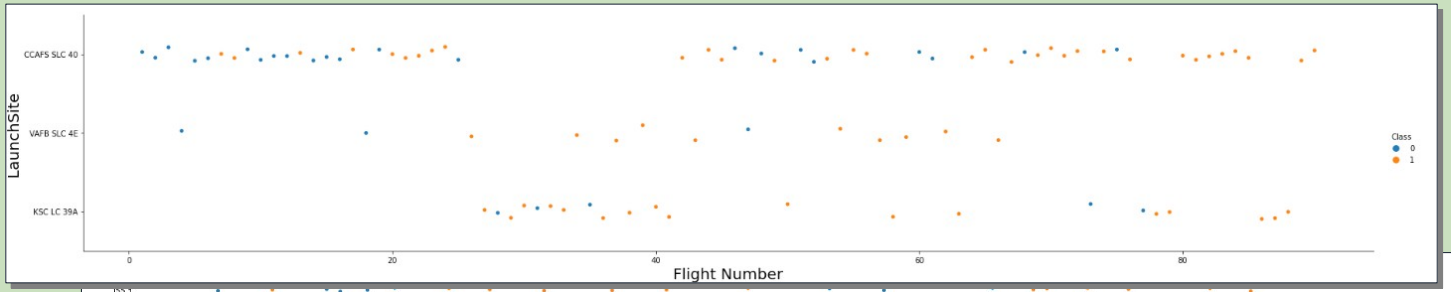
Data Wrangling

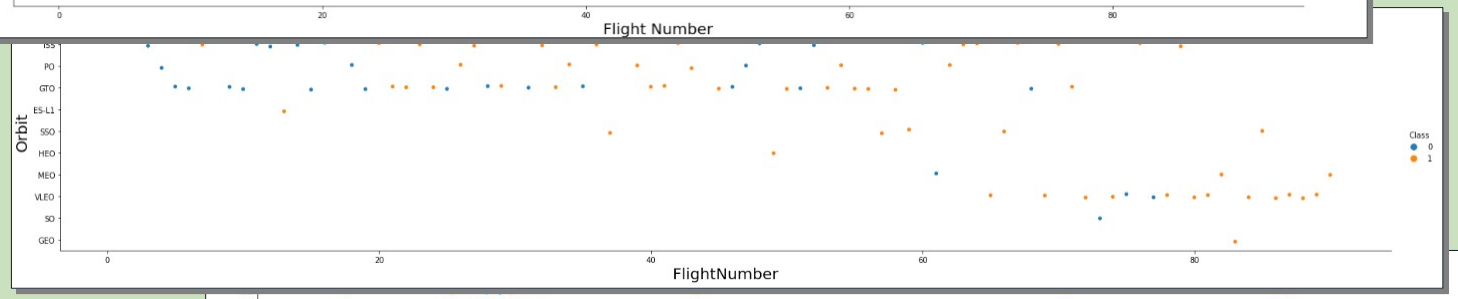


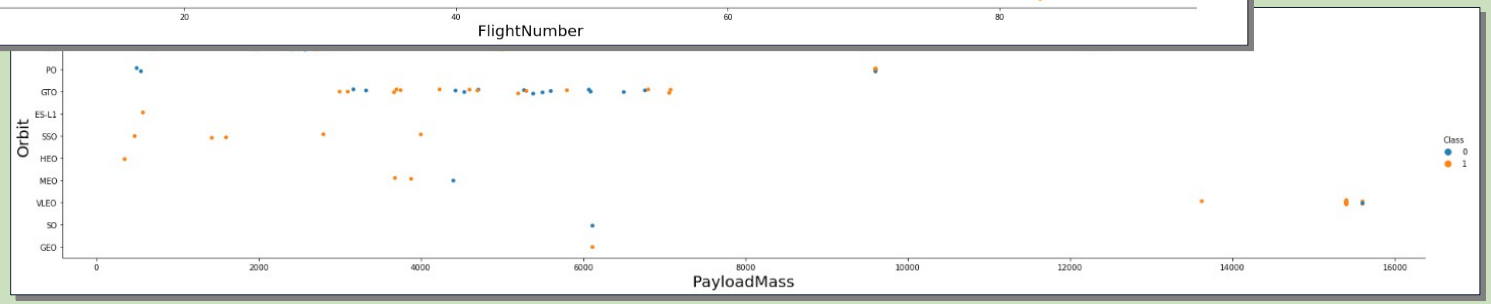
GitHub link: https://github.com/akscb24/Coursera-Capstone-AK/blob/f91f556fc7a1c8f9809f17cb14902e0a98a75904/W1-3-EDA lab- Data wrangling.ipynb

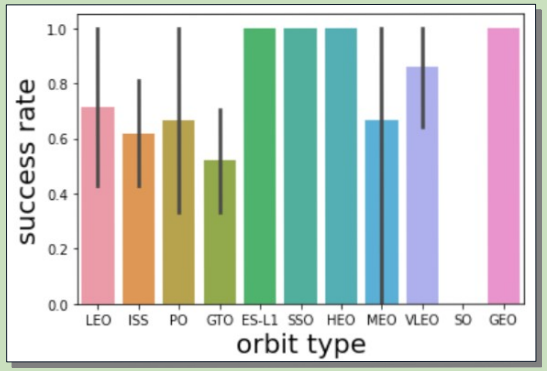
EDA with Data Visualization

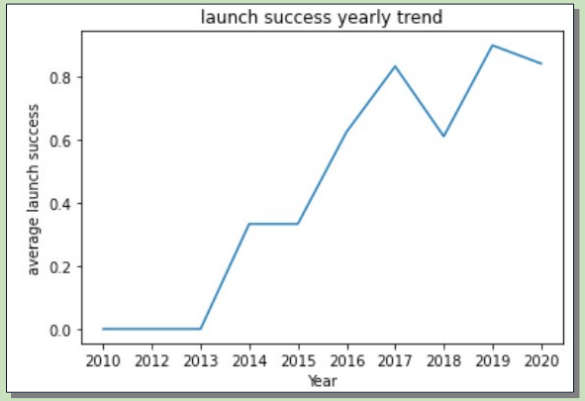
We will try to understand the data by trying different features such as yearly trend, outcome distribution by pairing different parameters (e.g. lunch pad, orbit, payload mass) to determine what attributes correlate with successful landing and which attributes can be set aside. To do this, we will resort to trend lines, bar charts and scatter graphs.











GitHub link: https://github.com/akscb24/Coursera-Capstone-AK/blob/115895f563eb2dd3f2a49beb7d9790b60e2c2896/W2-1- EAD data viz.ipynb

EDA with SQL

- Display the names of the unique launch sites in the space mission

```
select distinct "Launch_site" from SPACEXTBL
```

- Display 5 records where launch sites begin with the string 'CCA

```
select  * from SPACEXTBL where "Launch_site" like '%CCA%' limit 5
```

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

```
select sum("PAYLOAD_MASS__KG_") from SPACEXTBL where "Customer" like 'NASA (CRS)'
```

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

```
select avg("PAYLOAD_MASS__KG_") from SPACEXTBL where "Booster_Version" like '%F9 v1.1%'
```

- List the date when the first successful landing outcome in ground pad was achieved

```
select min("Date") from SPACEXTBL where "Landing _Outcome" like '%Success%'
```

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

```
select distinct "Booster_Version" from SPACEXTBL where "PAYLOAD_MASS__KG_" Between 4000 and 6000 and "Landing _Outcome"= 'Success (drone ship)'
```

- List the total number of successful and failure mission outcomes

```
select "Mission_Outcome", count("Mission_Outcome")
from SPACEXTBL
group by "Mission_Outcome" ;
```

Continued on next page...

EDA with SQL

- List the names of the booster_versions which have carried the maximum payload mass. Using a sub-query

```
select distinct "Booster_Version","PAYLOAD_MASS__KG_"
from SPACEXTBL
where "PAYLOAD_MASS__KG_" = (select max("PAYLOAD_MASS__KG_") from SPACEXTBL )
```

- 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.

```
select substr("Date", 4, 2) as 'month',"Landing _Outcome", "Booster_Version", "Launch_Site"
from SPACEXTBL
where substr("Date", 7, 4)='2015' and "Landing _Outcome"='Failure (drone ship)'
```

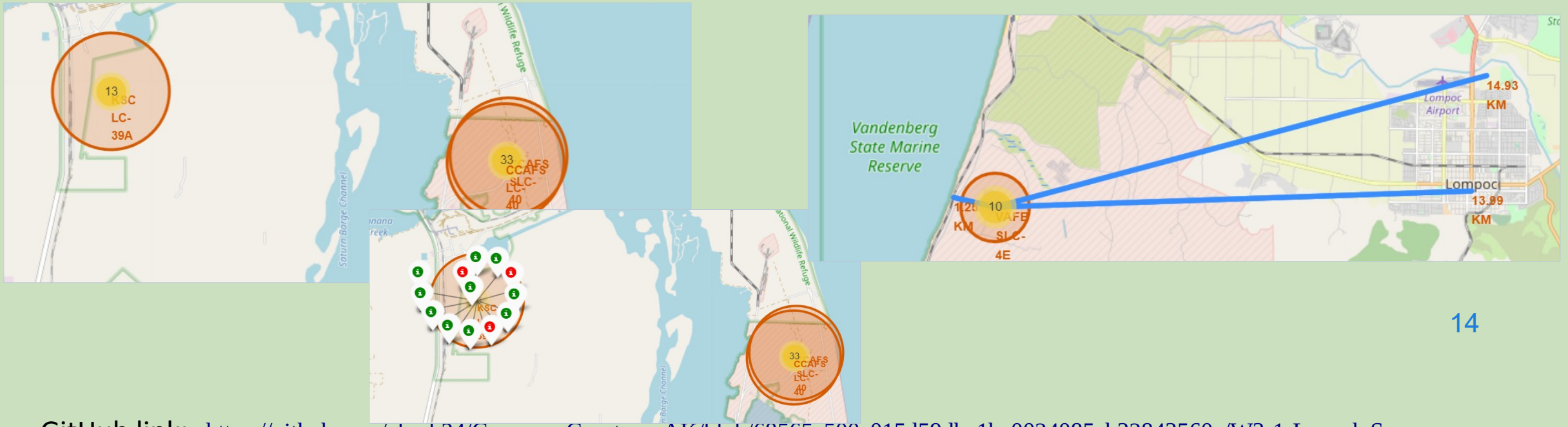
- Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

```
select "Landing _Outcome", count("Landing _Outcome")
from SPACEXTBL
where "Date" between '2001/02/25' and '2022/02/27'
Group by "Landing _Outcome"
order by count("Landing _Outcome") DESC;
```

GitHub link: https://github.com/akscb24/Coursera-Capstone-AK/blob/4aa0ac83ebcdcbc367b43e60ed8d5d60f046cb15/W2-2- EDA SQL Notebook.ipynb

Build an Interactive Map with Folium

- We have created circles with custom labels to show launch sites and their name.
- Then we have created a market cluster to contain several markers that are color coded based on landing outcome. These market clusters are also shown on each launch site.
- At the end, we have used lines and measurement tools to show the closet railway, road and city to our launch site.



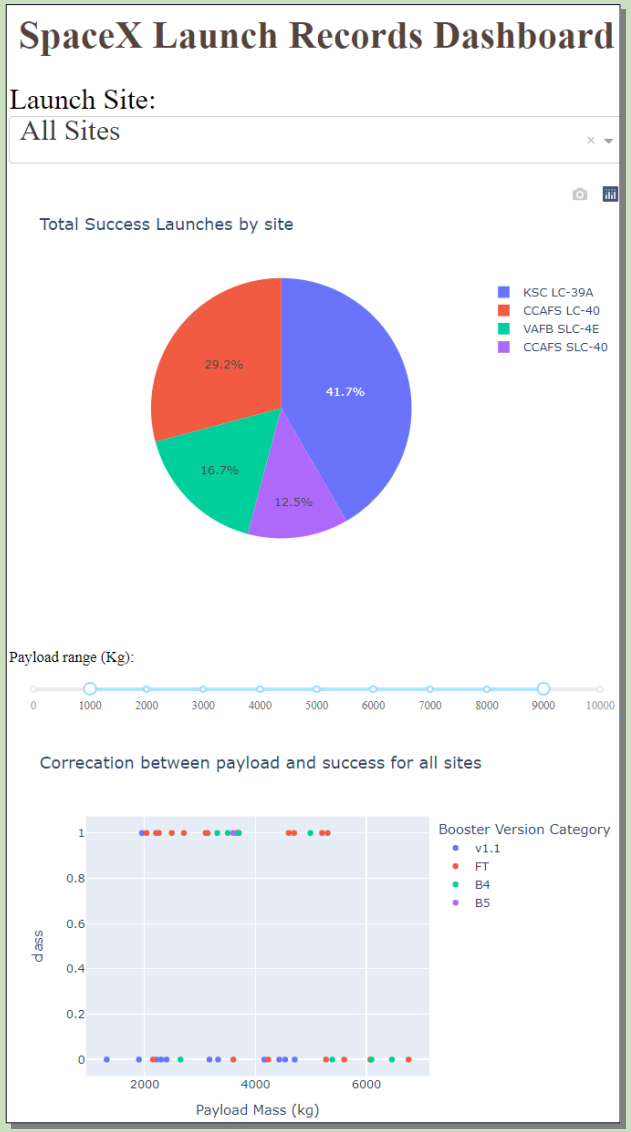
GitHub link: https://github.com/akscb24/Coursera-Capstone-AK/blob/68565e500a015d59dbc1be0024085ab32843560e/W3-1-Launch Sites Locations Analysis with Folium.ipynb

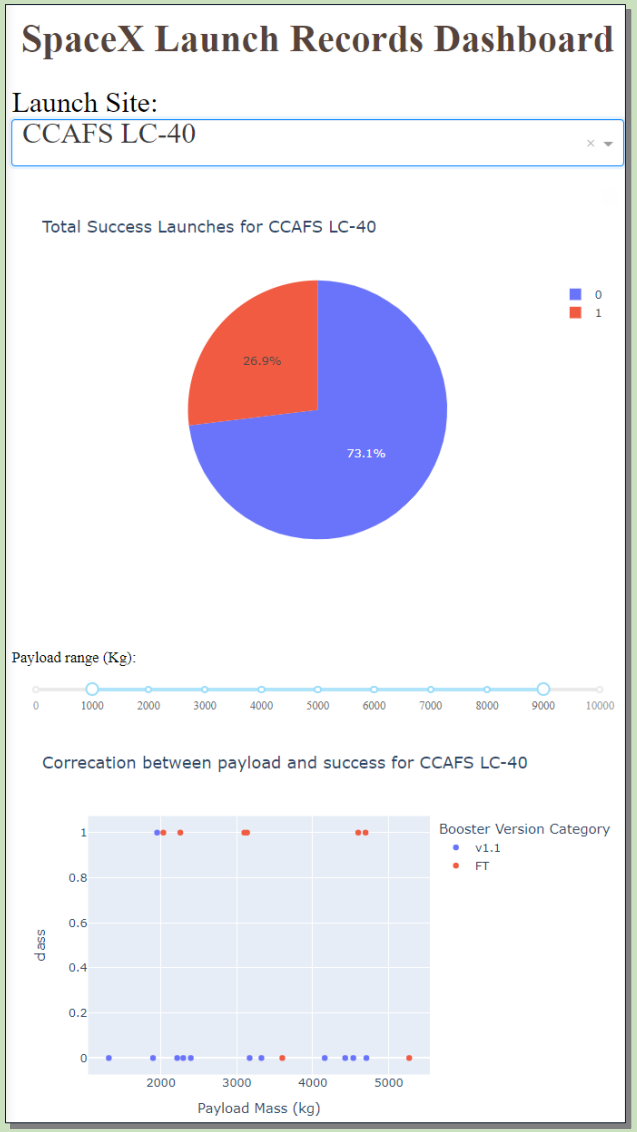
Build a Dashboard with Plotly Dash

- In the Plotly dashboard I have used pie chart and an scatter graph to represent clear visuals of the data.
- Pie charts make it much easier to have a good sense of dominant values within each category which in our case was success/failour.
- Scatter charts help us to correlate desired values (in this case launch success) to another parameter that we think can be a deriving factor (in this case payload)
- The data on these charts are narrowed down or summed up (taking all values) by using a drop down to select launch sites and an slider so we can set a limit on min and max payload for each launch.

Charts and link provided in the following slide...

Build a Dashboard with Plotly Dash





Link to GitHub:

https://github.com/akscb24/Coursera-Capstone-AK/blob/5c6dcf496fd7d3bf959f184a187174d9cad20b01/spacex_dash_app.py

Predictive Analysis (Classification)

Modify data frame (adding new column)

Standardise data

Split into training And test data set

SVM: Find the best Hyper-parameter

SVM: build model and evaluate Accuracy score

Clasf. tree: find the best Hyper-parameter

Clasf. Tree: build model and Evaluate Accuracy score

Log reg: Find the best Hyper-parameter

Log reg: build model and evaluate Accuracy score

KNN: Find the best Hyper-parameter

KNN: build model and evaluate Accuracy score

Identify the best model

```
transform = preprocessing.StandardScaler()

X = transform.fit_transform(X)
```

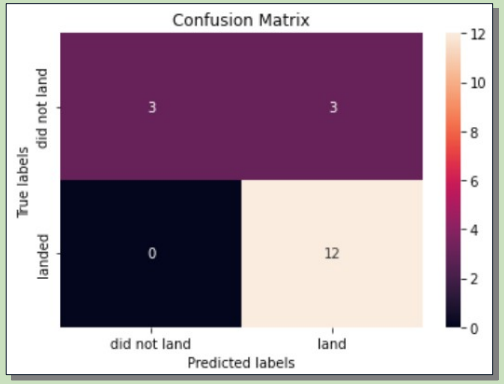
```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2, random_state=2)
```

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}
lr=LogisticRegression()

logreg_cv = GridSearchCV(lr,parameters,cv=10)

logreg_cv.fit(X_train, Y_train)
```

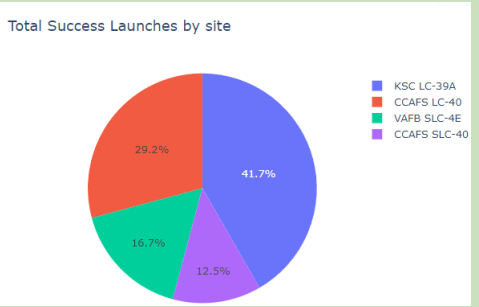
```
GridSearchCV(cv=10, estimator=LogisticRegression(),
             param_grid={'C': [0.01, 0.1, 1], 'penalty': ['l2'],
                         'solver': ['lbfgs']})
```

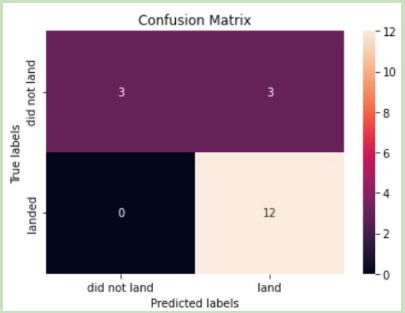


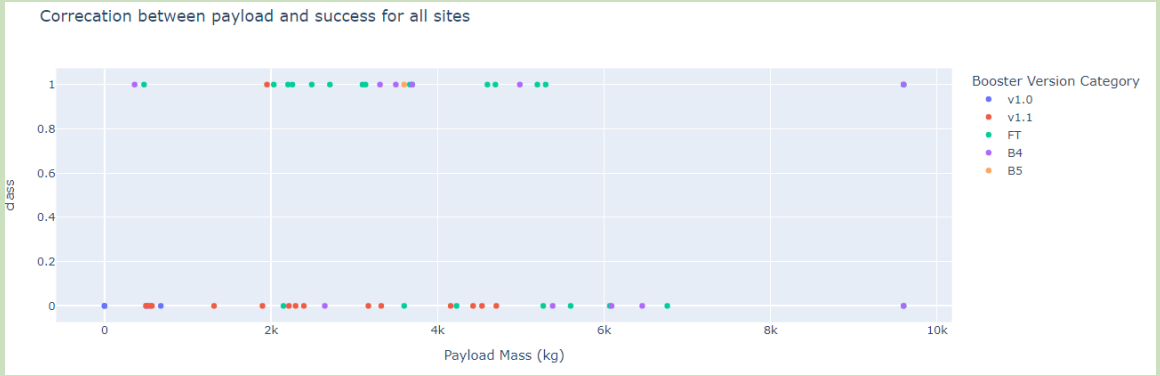
Link to GitHub: https://github.com/akscb24/Coursera-Capstone-AK/blob/7391e89c7aa61d35d9641f282a53a7ee2c9d9d86/W4 Machine Learning Prediction lab.ipynb

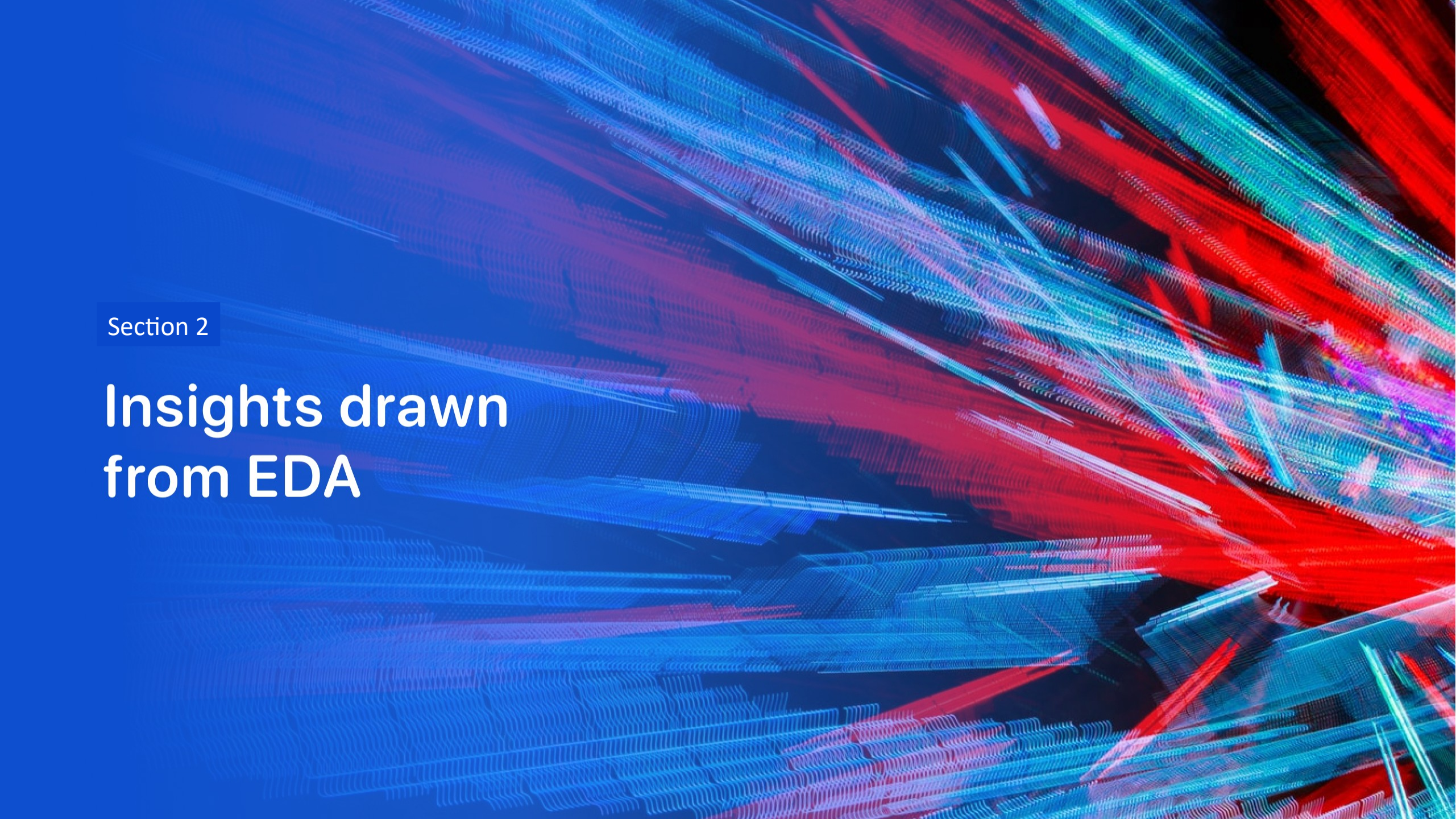
Results

- Exploratory data analysis results we can draw following conclusions:
  - Overall success rate has increased over time
  - Over time, SpaceX increased payload to max 16000kg most of which where successful
  - Even though success in launch sites vary, we can not correlate success rate to launch site as most of launches in the early days where carried out only from one site and those failures can be attributed to other factors like experience/technical knowledge.
  - The following orbit types had the most success: ES-L1, GEO, HEO, SSO. Considering number of launches, we can say GEO has more certainty.
- Interactive analytics demo in screenshots
  - According to the data, all payloads over 6 tons are launched via FT and B4 boosters, which B4 has proven to be more successful. However, FT seems to be more promising in payloads between 3-5tons
- Predictive analysis results
  - SVM provided best accuracy score which is 0.88.
  - according to the model, we have more certainty in predicting successful landing.

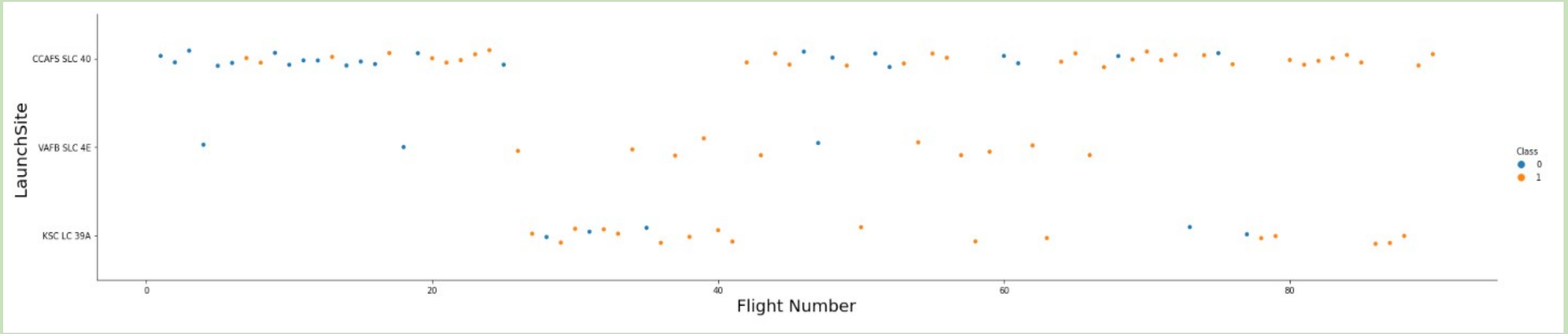







Section 2
Insights drawn from EDA

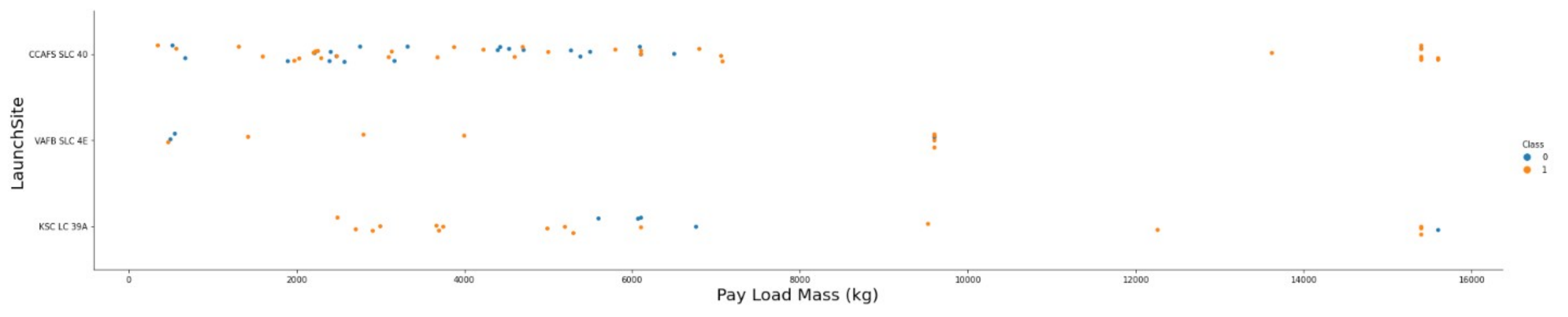
Flight Number vs. Launch Site



It appears that CCAFS SLC 40 is the most used launch site among the others, where VAFB SLC 4E is not used in most recent launches.

All last 5 launches from all sites were successful.

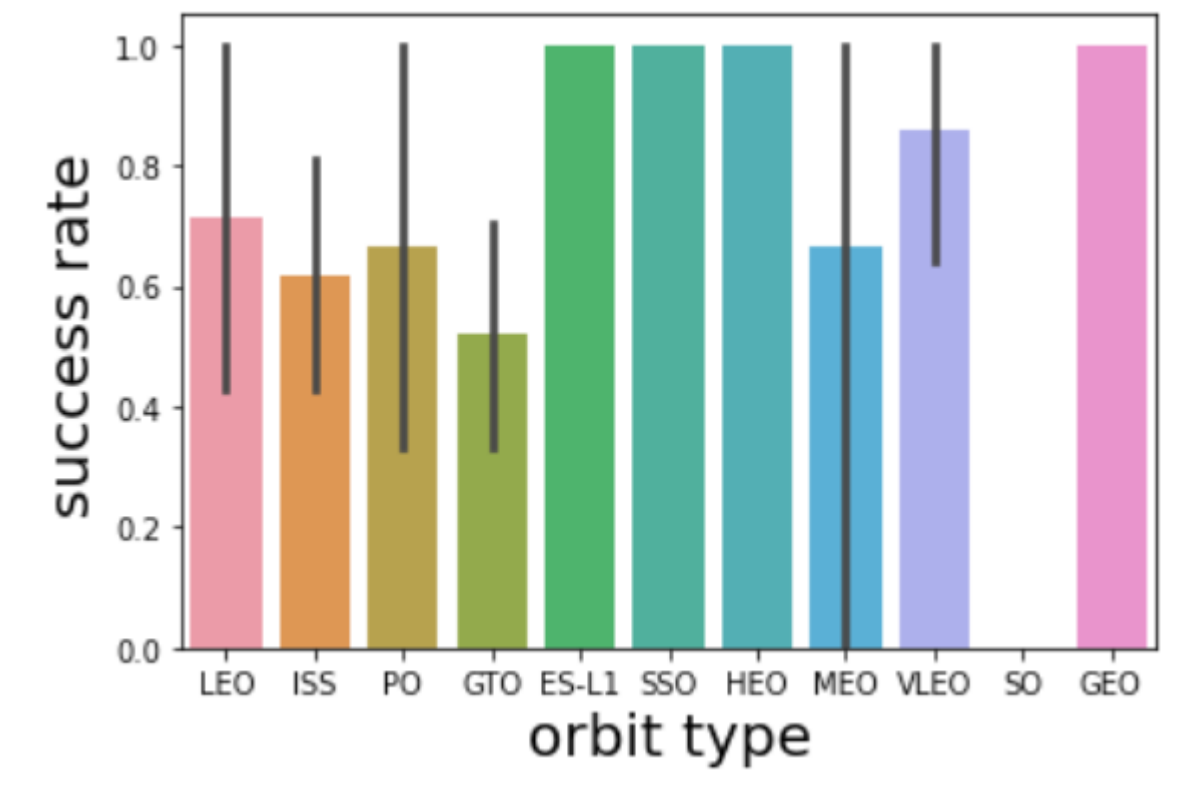
Payload vs. Launch Site



VAFB SLC 4E is only used for payloads under 10 tons

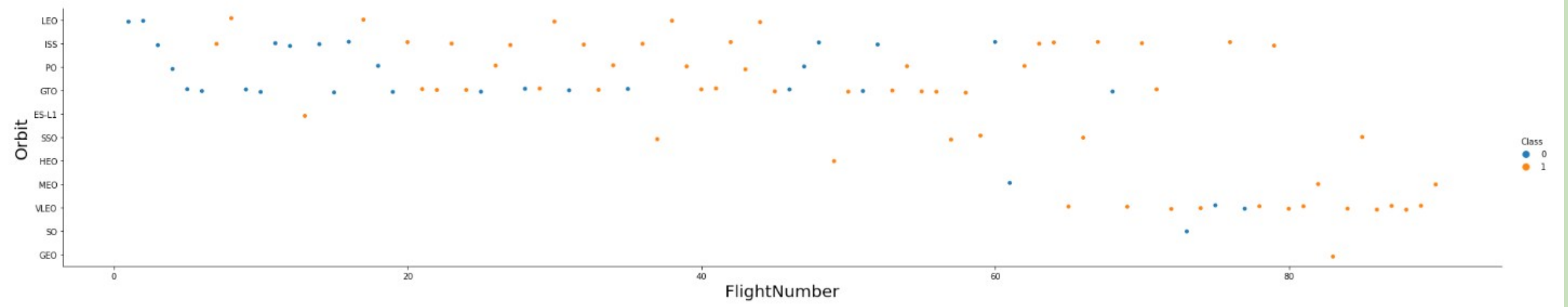
KSC LC is not used for any launches less than 2 tons in payload weight.

Success Rate vs. Orbit Type

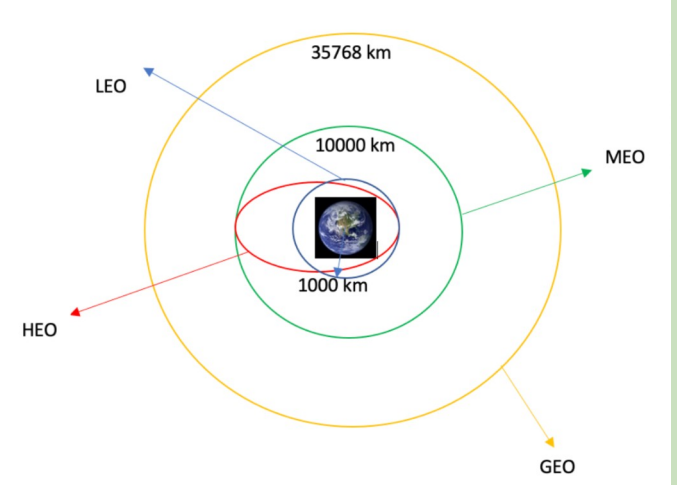


ES-L1, GEO, HEO, SSO. Considering number of launches, GEO has more certainty to be successful.

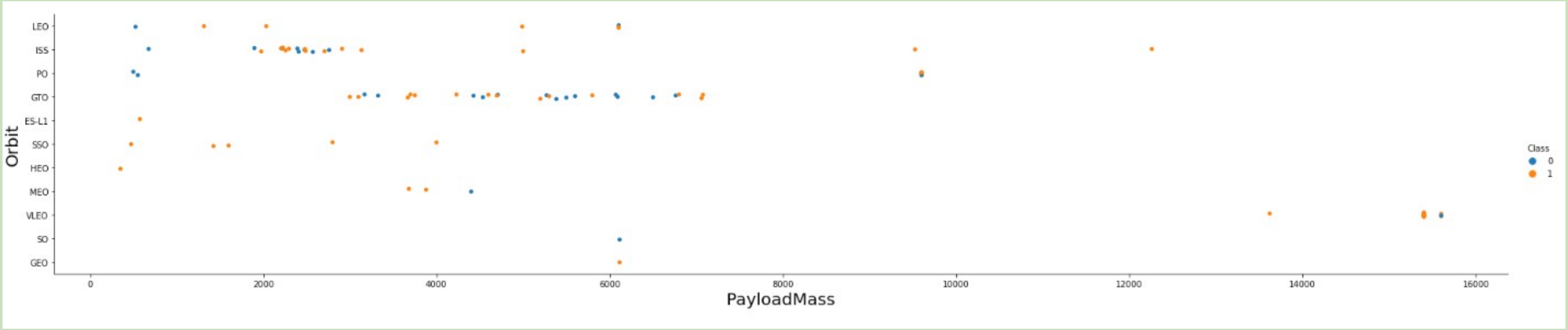
Flight Number vs. Orbit Type



- It seems that most recent launches are targeting VLEO and providing service to ISS
- In the early launches, LEO, GTO and ISS seem to be the most frequent targets



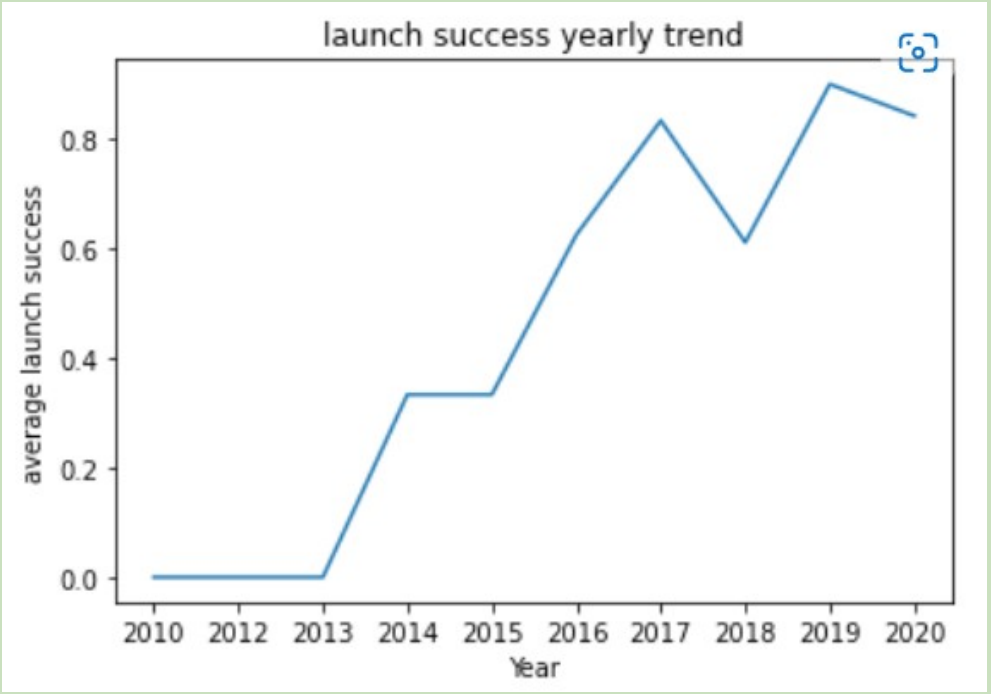
Payload vs. Orbit Type



- All payloads over 13 tons where aimed at VLEO
- GTO payloads are between 3-8 tons
- Most of payloads for ISS seems to be between 2-4 tons with few exceptions.

Launch Success Yearly Trend

- There was no success launches prior to 2013
- After 2013 there is an steady upward trend in launch success rates which can suggest improvement in experience and technolodgy.



All Launch Site Names

- ***Select distinct*** is used to remove duplicate values.

```
%%sql
select distinct "Launch_site" from SPACEXTBL
```

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

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

Launch Site Names Begin with 'CCA'

```
%%sql
select  * from SPACEXTBL where "Launch_site" like '%CCA%' limit 5
```

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

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	0:35:00	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

- Using **like** with **wildcard** (**%**) enables us to get the result we want

Total Payload Mass

```
%%sql
select sum("PAYLOAD_MASS__KG_") from SPACEXTBL where "Customer" like 'NASA (CRS)'
```

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

sum("PAYLOAD_MASS__KG_")
45596

- We used **sum** on payload mass column to get result we want

Average Payload Mass by F9 v1.1

```
%%sql
select avg("PAYLOAD_MASS__KG_") from SPACEXTBL where "Booster_Version" like '%F9 v1.1%'
```

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

avg("PAYLOAD_MASS__KG_")
2534.6666666666665

- Here we need to use **avg** as well as limit the data by using **like** syntax.

First Successful Ground Landing Date

```
%%sql
select min("Date") from SPACEXTBL where "Landing _Outcome" like 'Success (ground pad)'
```

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

min("Date")
01-05-2017

- We used **min** function on date column

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql
select distinct "Booster_Version"
from SPACEXTBL
where "PAYLOAD_MASS__KG_" Between 4000 and 6000 and "Landing _Outcome"= 'Success (drone ship)'
```

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

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

Using **between** we limit the result to the range we desire

Total Number of Successful and Failure Mission Outcomes

Using **group by** we can separate different outcomes.

```
%%sql
select "Mission_Outcome", count("Mission_Outcome")
from SPACEXTBL
group by "Mission_Outcome" ;
```

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

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

Boosters Carried Maximum Payload

- We need to used nested select to get the data we want

```
%%sql

select distinct "Booster_Version","PAYLOAD_MASS__KG_"
from SPACEXTBL
where "PAYLOAD_MASS__KG_" = (select max("PAYLOAD_MASS__KG_") from SPACEXTBL )
```

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

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

```
%%sql

select substr("Date", 4, 2) as 'month',"Landing _Outcome", "Booster_Version", "Launch_Site"
from SPACEXTBL
where substr("Date", 7, 4)='2015' and "Landing _Outcome"='Failure (drone ship)'
```

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

month	Landing _Outcome	Booster_Version	Launch_Site
01	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- Nested select, we use substr to extract month and year of launch dates

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

```
%%sql

select "Landing _Outcome", count("Landing _Outcome")
from SPACEXTBL
where "Landing _Outcome" LIKE '%success%' and (date between '04-06-2010' and '20-03-2017')
Group by "Landing _Outcome"
order by count("Landing _Outcome") DESC;
```

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

Landing _Outcome	count("Landing _Outcome")
Success	20
Success (drone ship)	8
Success (ground pad)	6

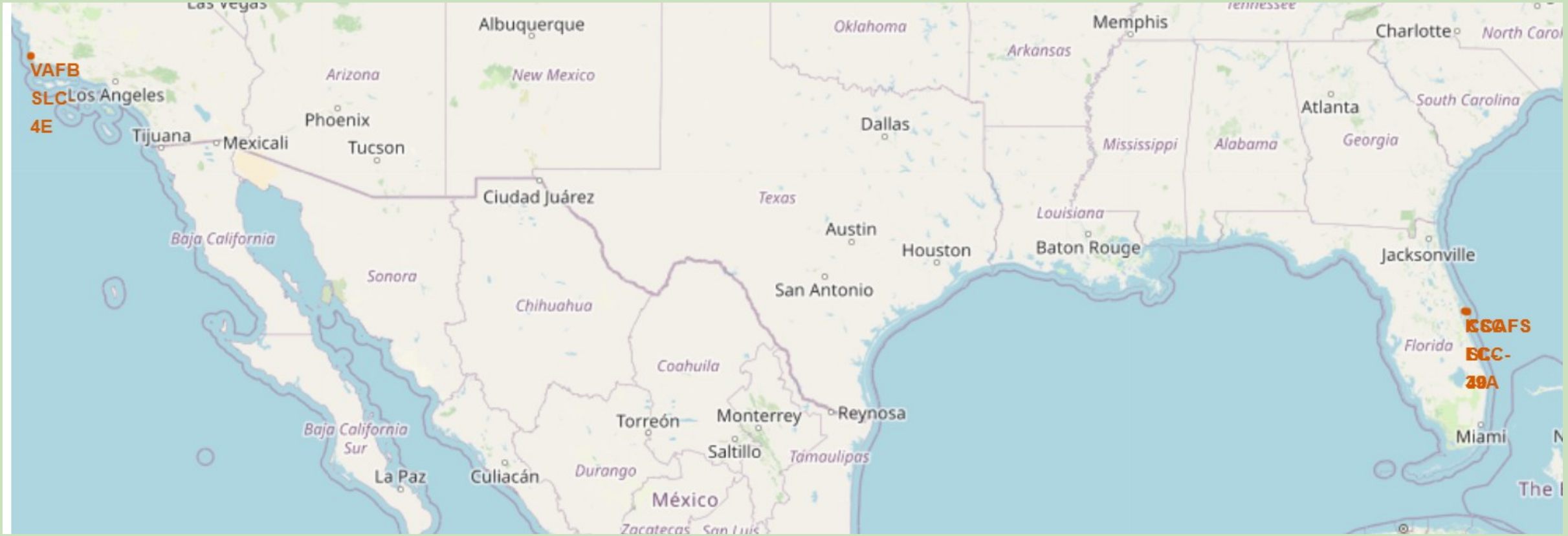
After selecting desired data, we have used **group by** and **order by** to sort our data

Section 3

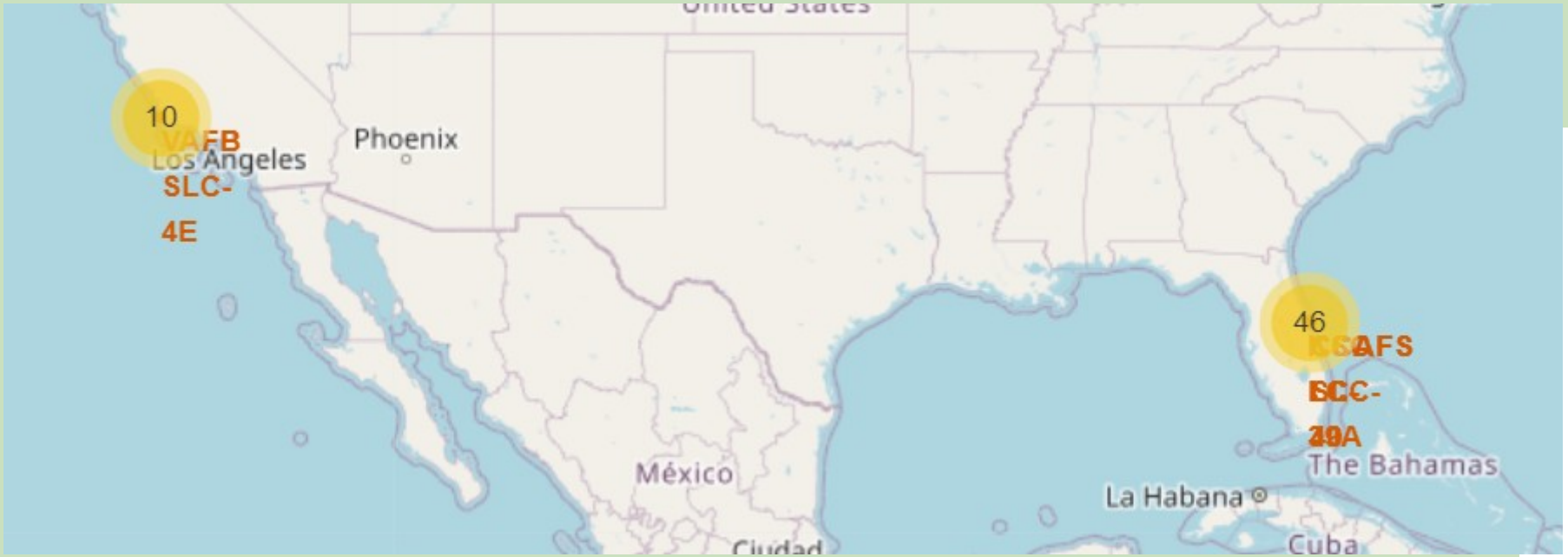
Launch Sites Proximities Analysis

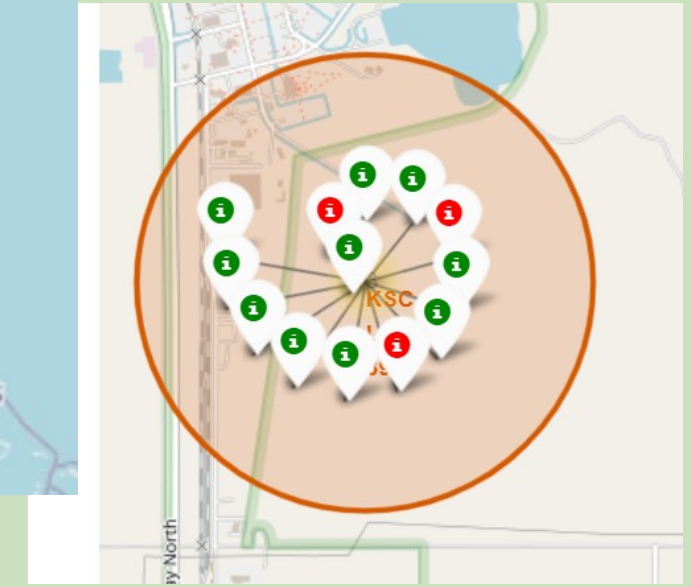
Mark all launch sites on a map

3 of 4 launch sites are located in Florida, out of which 2 are located in Cape Canaveral space force station.



success/failed launches for each site on the map

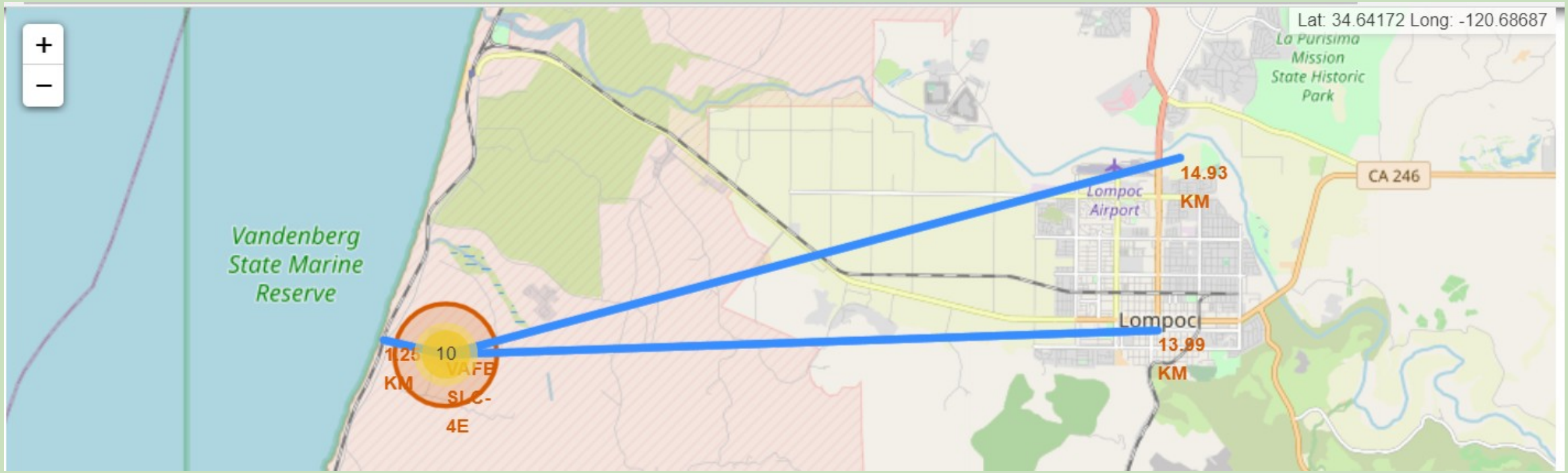


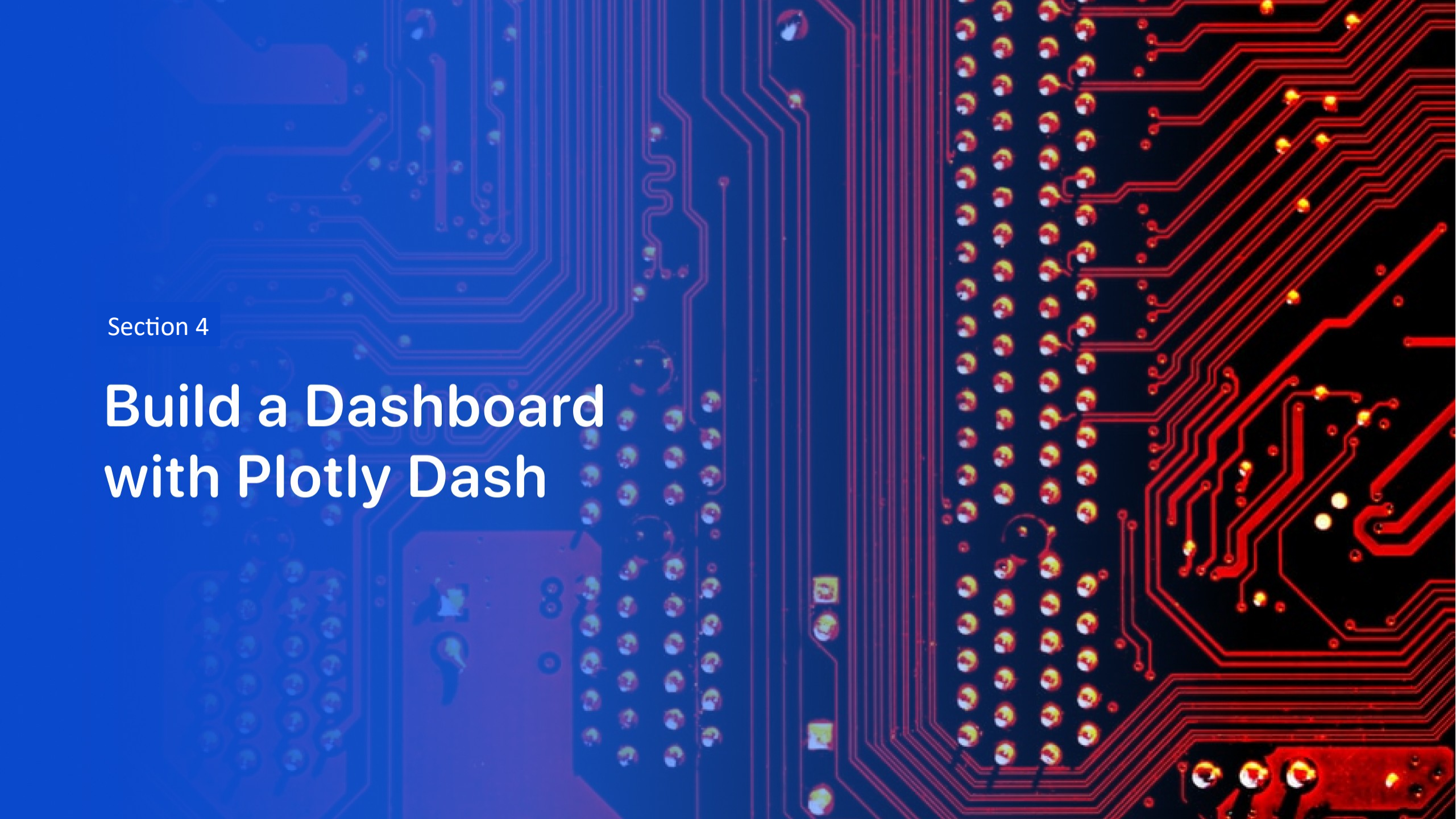


- Here we have used market clusters to give an impression of how many launches have been conducted from each station. As we zoom in, each marker would reveal the faith of a launch.

launch site proximity analysis

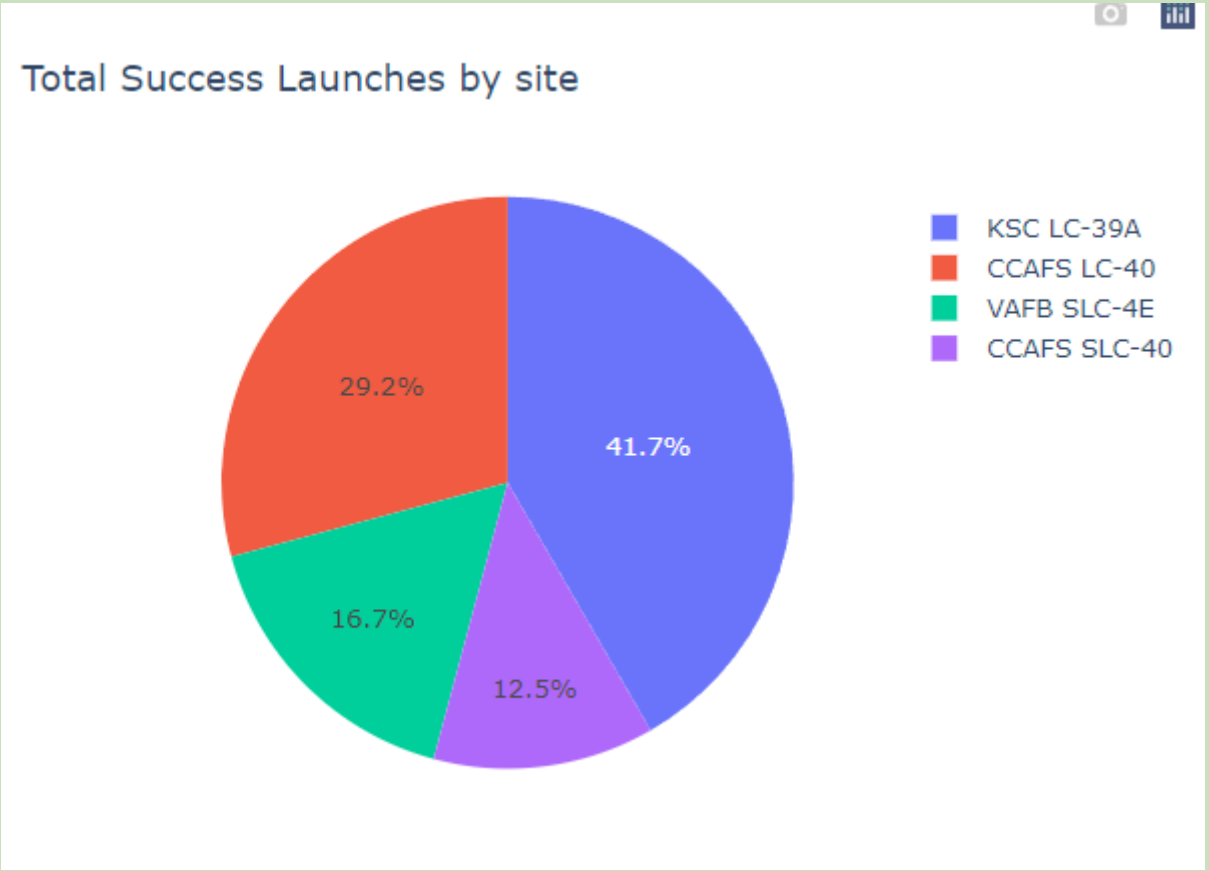
- Here we have equipped the map with mouse pointer coordinates, and different lines showing distance from various points of interest to our launch site (nearest city, railway, highway)




Section 4
Build a Dashboard with Plotly Dash

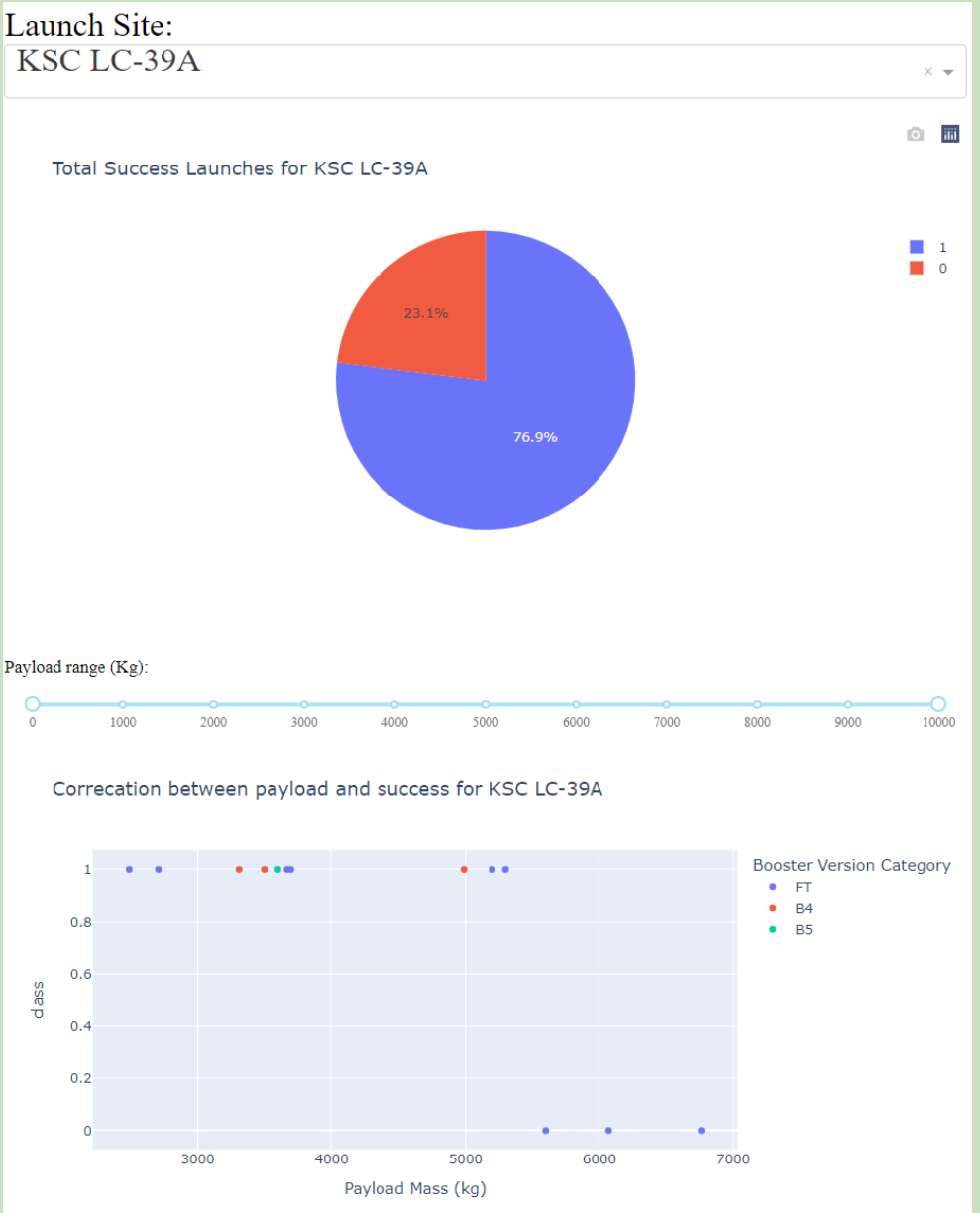
launch success count for all sites

- This pie chart provides immediate intuition about which sites are used most predominantly



launch site with highest launch success ratio

Comparing to other sites, KSC LC has the highest success rate.
This can be quickly identified using the scatter chart



Payload vs. Launch Outcome for all sites

Payload 0 to 5k

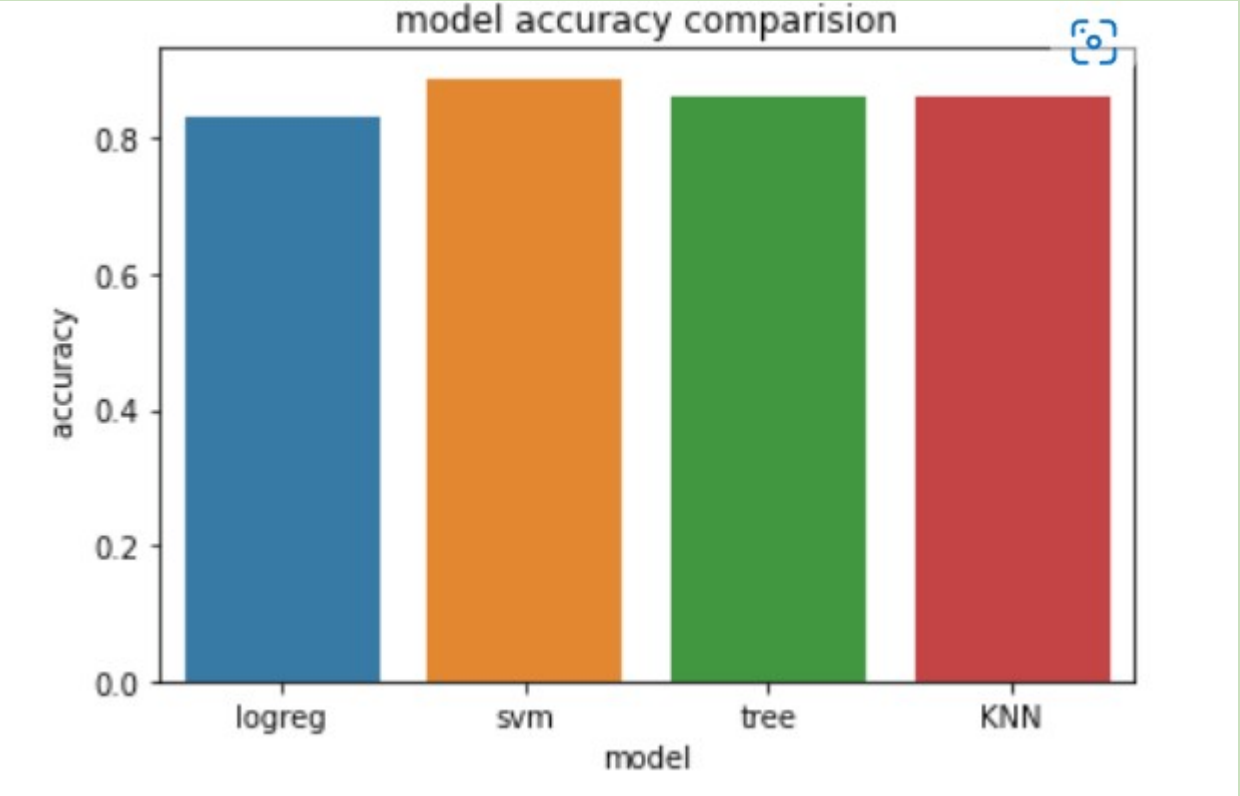
Payload 5k to 10k

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- SVM has higher accuracy



Confusion Matrix

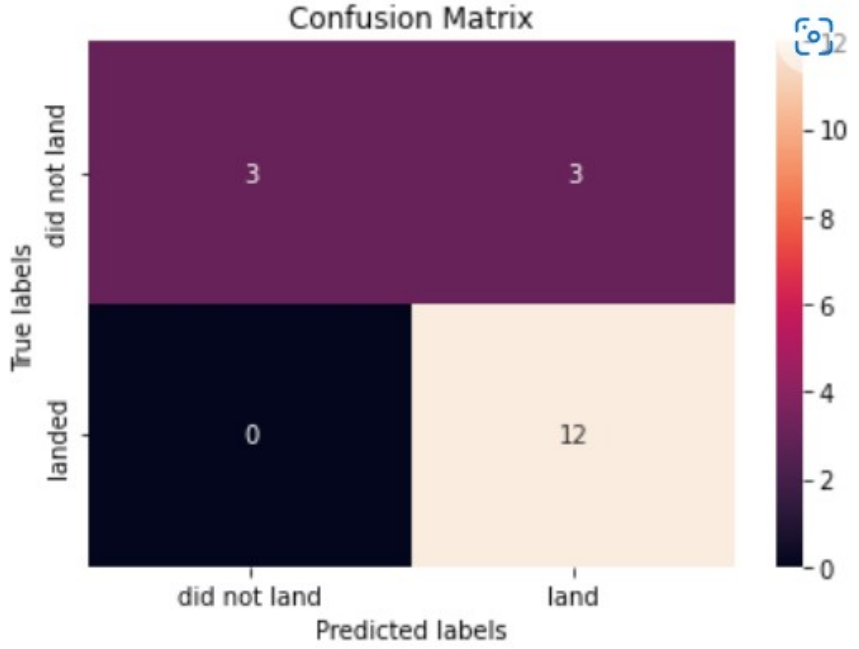
- I obtained the best results using SVM.
- Pros:
  - Zero false negative. Meaining that this model is good at predicting successful landing where landing was actually successful.
- Cons:
  - Model seems to be indecisive when it comes to false positive. Meaning that predictions of failure landings might not be reliable.

```
svm_cv_score = svm_cv.score(X_train,Y_train)
svm_cv_score
```

```
0.8888888888888888
```

We can plot the confusion matrix

```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

- As we move forward in time, launches tend to be more successful.
- As we move forward in time, we observe that SpaceX became capble of launching heavier payloads
- SVM model has the highest accuracy
- None of the models provided good predictions when outcome was failure (high false negatives)
- All launch sites are geographically quite close to the ocean
- KSC LC has the most success rate, but if we exclude the first few years, we can say that launch site might not determine success of the landing as we have observed that the last 5 launches for all sites were successful.
- It seems that most recent launches are targeting VLEO and providing service to ISS
- In the early launches, LEO, GTO and ISS seem to be the most frequent targets

Thank you!