Winning Space Race with Data Science

Muhammad Yasir
Date 4th Jan 2024
Github link:
https://github.com/yasir134/my_projects/tree/main/DataScientist_capstone

Outline

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

Executive Summary

- Summary of methodologies
  - Data collected through spaceX API & wikipedia page, SQL used to explore and plotly and folium used for visualizations & dashboards. GridSearchCV used to find best parameters for machine learning.

- Summary of all results
  - Machine learning models used on data are logistic regression, k-nearst neighbor, decision tree classifier. Accuracy results are between 66-96% more data needed to predict exact accuracy.

Introduction

- Project background and context
  - Space race has started, many companies trying to compete but spaceX is shining with cost effective results because spaceX landing its stage 1 rocket so it can be used again for further launches.
  - Space Y want to compete in space race too.
- Problems you want to find answers
  - spaceY needed to build a machine learning model which can predict stage 1 rocket landing safely, only than they can compete with spaceX.

Section 1

Methodology

Methodology

Executive Summary

- Data collection: spaceX API, coursera static url & wikipedia.org
- Data wrangling with python
- Exploratory data analysis (EDA) using visualization and SQL
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models
  - GridSearchCV used to find best hyperparameters for Decision tree, logistic regression and SVM.

Data Collection

SpaceX API, coursera static url and wikipedia.org used to collect data, links are:

https://api.spacexdata.com/v4/rockets/

https://api.spacexdata.com/v4/launchpads/

https://api.spacexdata.com/v4/payloads/

https://api.spacexdata.com/v4/cores/

https://api.spacexdata.com/v4/launches/past

https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/API_call_spacex_api.json

https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

Data Collection – SpaceX API

- Data imported from multiple sources like spaceX & wikipedia.
- spaceX columns used: flight number, Date, BoosterVersion, PayloadMass, Orbit, LaunchSite, Outcome, Flights, GridFins, Reused, Legs, LandingPad, Block, ReusedCount, Serial, Longitude, Latitude.

- Add the GitHub URL of the completed SpaceX API calls notebook (must include completed code cell and outcome cell), as an external reference and peer-review purpose

```
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}
```

Data Collection - Scraping

- Webscraped from wikipedia

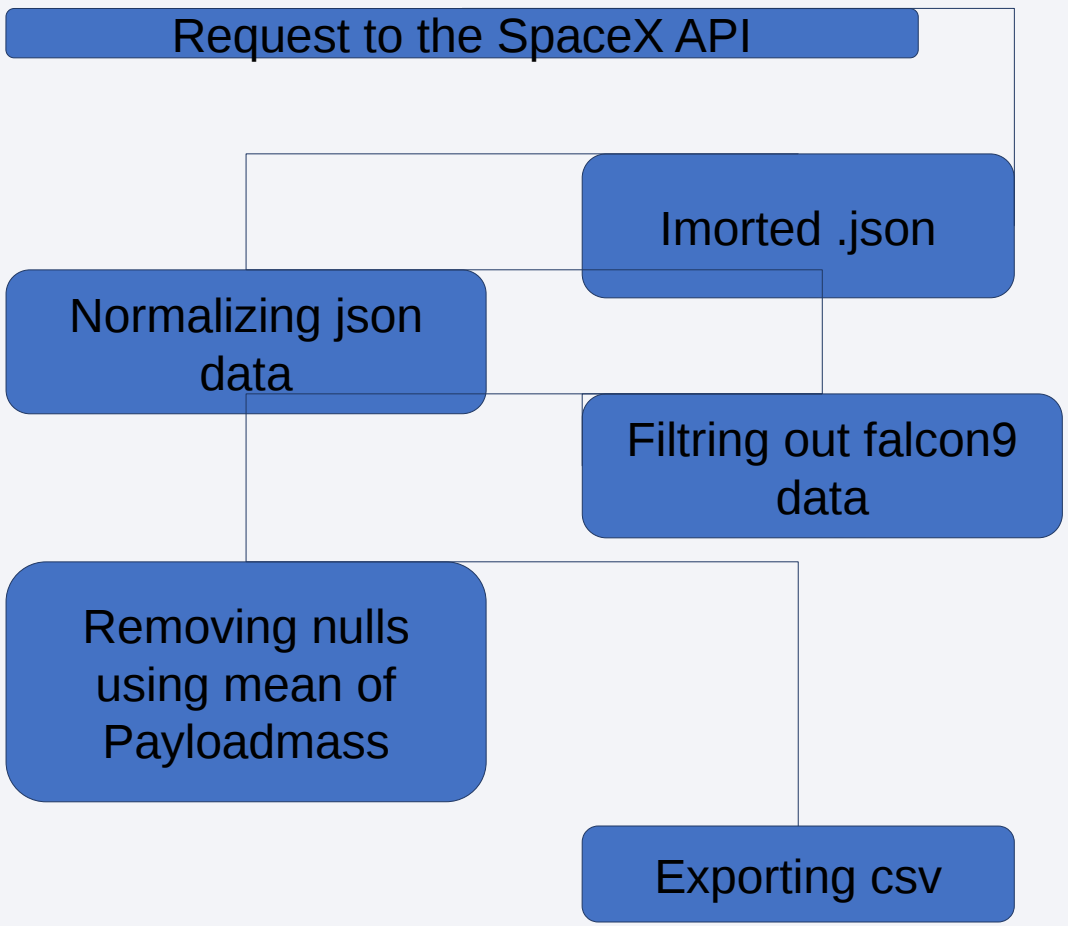
Link:
https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922

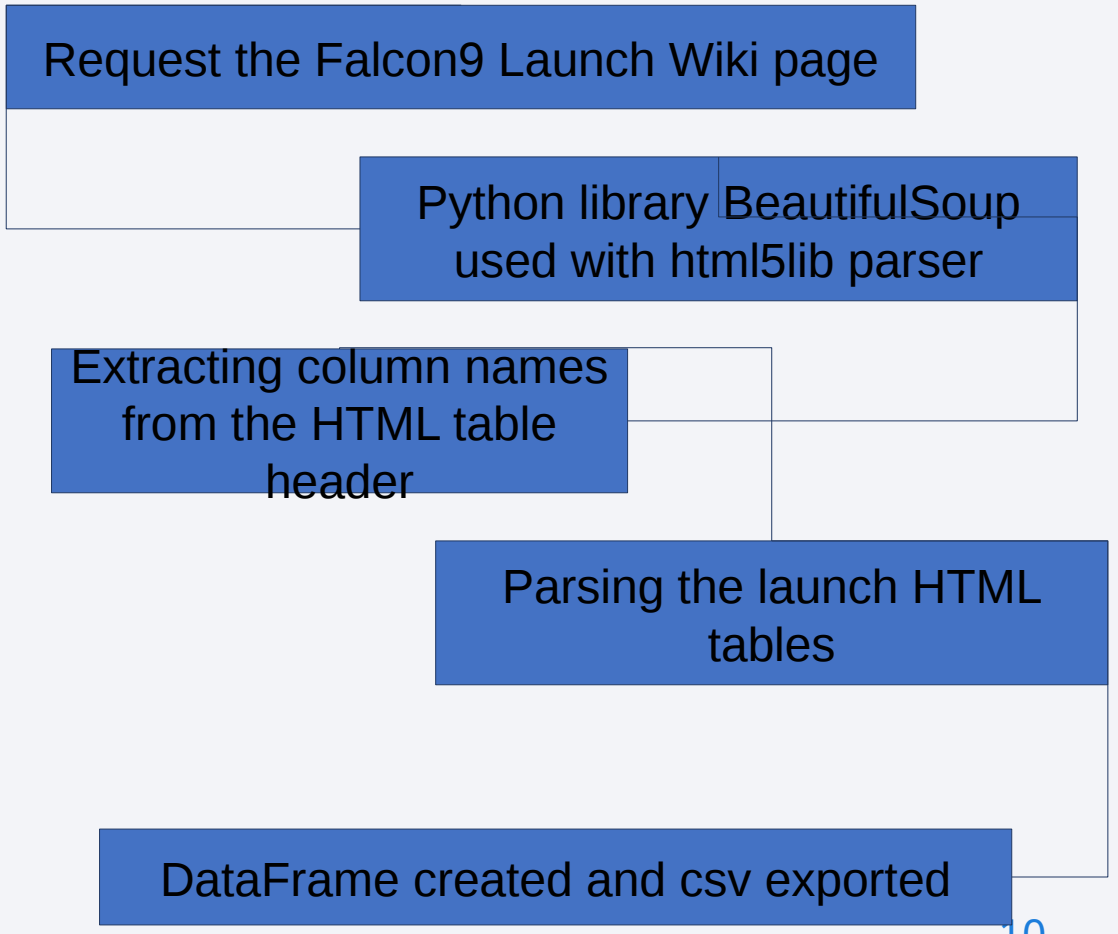
Columns used: Flight No. , Launch sit, Payload, Payload mass, Orbit, Customer, Launch outcome, Version Booster, Booster landing, Date, Time.

- Add the GitHub URL of the completed web scraping notebook, as an external reference and peer-review purpose

```
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']=[]
```

FlowCharts for spaceX and Wikipedia





Data Wrangling

Describing how data were processed

Creating a training label with landing outcomes where successful = 1 & failure = 0.

Outcome column has two components: ‘Mission Outcome’ ‘Landing Location’

New training label column ‘class’ with a value of 1 if ‘Mission Outcome’ is True and 0 otherwise.

Value Mapping:

True ASDS, True RTLS, & True Ocean – set to -> 1

None None, False ASDS, None ASDS, False Ocean, False RTLS – set to -> 0

Add the GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose

EDA with Data Visualization

Summarize what charts were plotted and why you used those charts

Column used for EDA : FlightNumber, PayloadMass, Orbit, LaunchSite, Flights, GridFins, Reused, Legs, 'LandingPad, Block, ReusedCount, Serial

Charts used for data Visualization: scatter point chart, bar chart for successful outcome, scatter plot for payload & orbit type relationships, line chart used for success yearly trend.

Add the GitHub URL of your completed EDA with data visualization notebook, as an external reference and peer-review purpose

EDA with SQL

CSV imported and loaded to ibm db database **SQL queries used to find:**

- names of the unique launch sites
- launch sites begin with the string 'CCA'
- total payload mass carried by boosters launched by NASA (CRS)
- average payload mass carried by booster version F9 v1.1
- date when the first succesful landing outcome in ground pad was acheived.
- names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
- total number of successful and failure mission outcomes
- names of the booster_versions which have carried the maximum payload mass
- records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
- Failures between the date 2010-06-04 and 2017-03-20, in descending order

Add the GitHub URL of your completed EDA with SQL notebook, as an external reference and peer-review purpose

Build an Interactive Map with Folium

Folium maps used to mark launch sited, successful landing and proximity example to nearest locations on map.

This helps to identify best location for launch sites.

Add the GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build a Dashboard with Plotly Dash

Plotly dash application contains pie & scatter chart.

- Pie chart shows success rate of launches by sites.

- Scatter point chart shows how successful landing varies on different site effected by payload mass and booster version types.

Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

Predictive Analysis (Classification)

Summarize how you built, evaluated, improved, and found the best performing classification model

- Selected training labels, standardize data & data splitting for training and testing
- Finding best Hyperparameter for SVM, sv=10 not working so used downsizing function used (next slide includes snaps) And classification Trees and Logistic regression.

You need present your model development process using key phrases and flowchart

Add the GitHub URL of your completed predictive analysis lab, as an external reference and peer-review purpose

SVM

```
parameters = {'kernel':('linear', 'rbf', 'poly', 'rbf', 'sigmoid'),
              'C': np.logspace(-3, 3, 5),
              'gamma':np.logspace(-3, 3, 5)}
svm = SVC()
```

✓ 0.0s

```
# making grid search object and fitting model
svm_cv = GridSearchCV(svm, parameters,cv=10)
svm_cv.fit(X_train, Y_train)
```

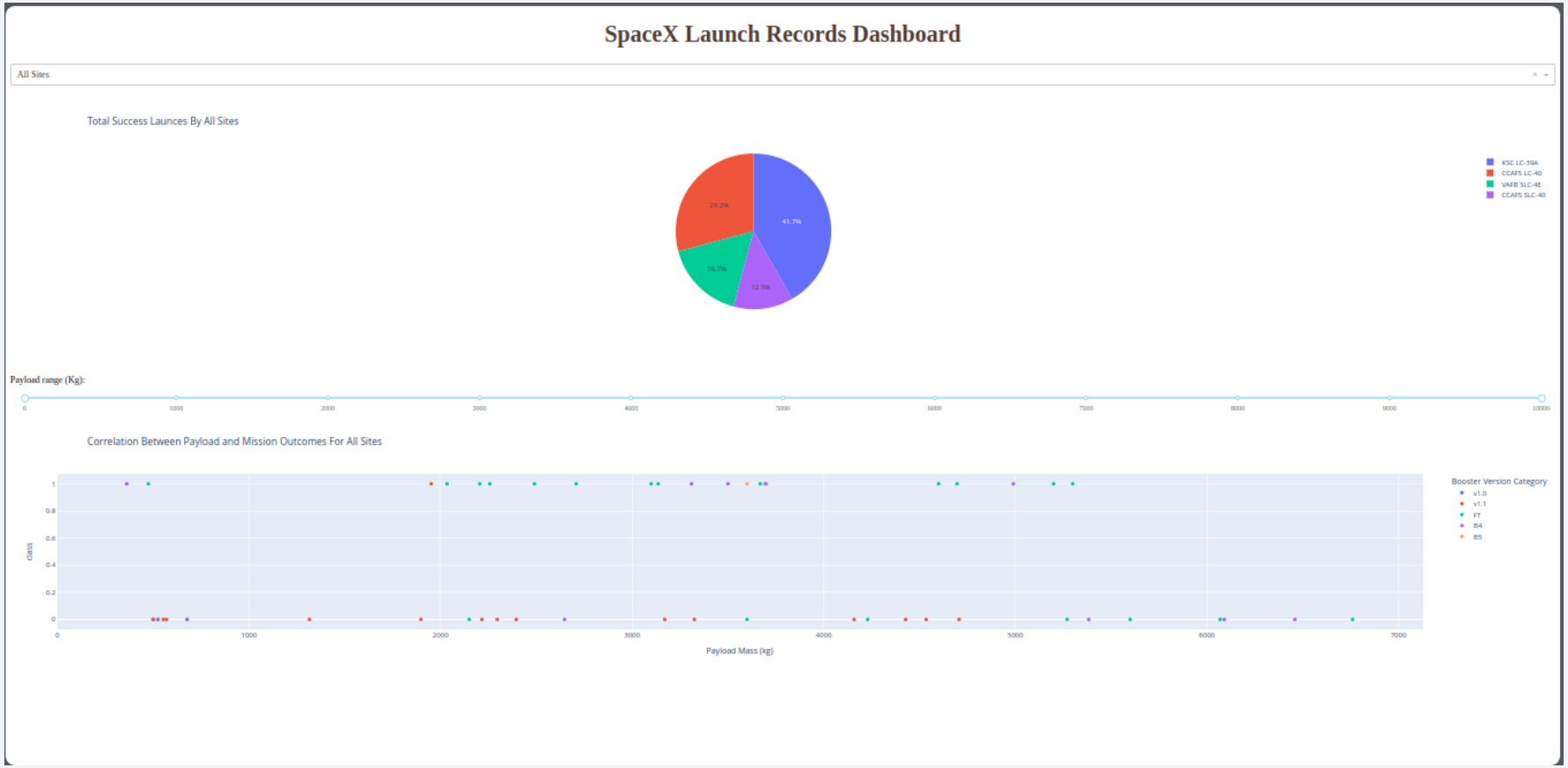
94m 11.4s

With cv 10 it took too much time for processing and no result, so fixed it with subset selection function as follows

```
parameters = {'kernel':('linear', 'rbf', 'poly', 'rbf', 'sigmoid'),
              'C': np.logspace(-3, 3, 5),
              'gamma':np.logspace(-3, 3, 5)}

# Reduce the size of the dataset or use a subset for the grid search
X_train_subset = X_train[:10]
Y_train_subset = Y_train[:10]

# Use a different cross-validation strategy
cv = StratifiedKFold(n_splits=5)

# Enable parallel processing
svm_cv = GridSearchCV(svm, parameters, cv=cv, n_jobs=-1)
svm_cv.fit(X_train_subset, Y_train_subset)
```

Results

- Dashboard preview shows accuracy of 57% to 73% by selection of launch site.
- CCAFS LC-40 is the most successful landing site.

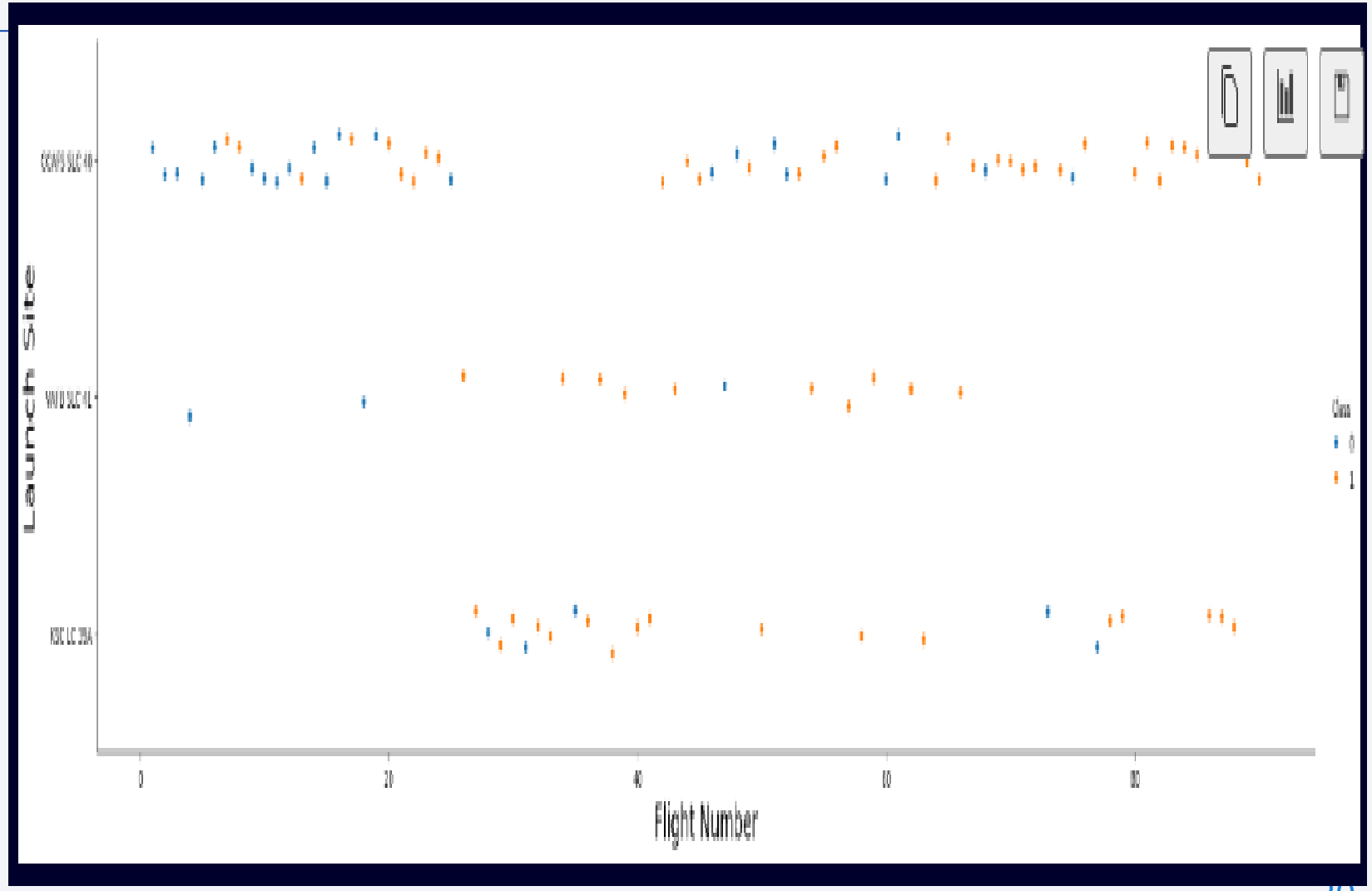


Section 2

Insights drawn from EDA

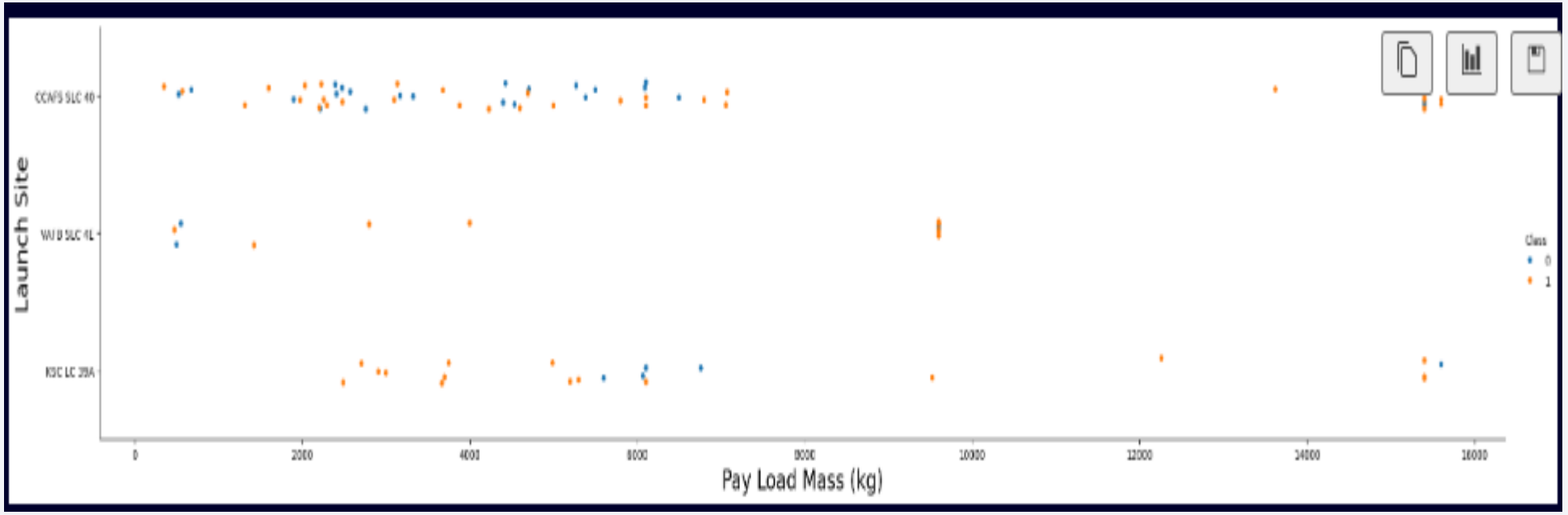
Flight Number vs. Launch Site

- Success rate increased as number of flights increased.



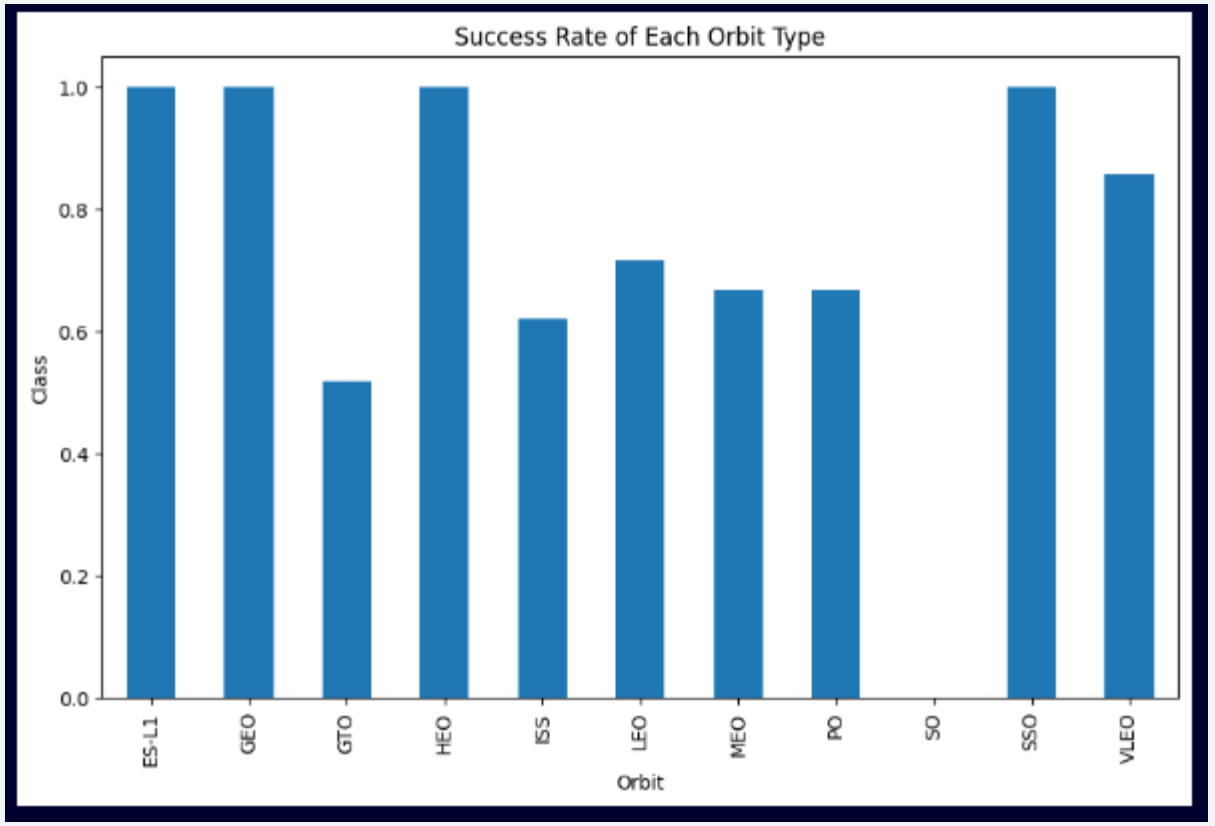
Payload vs. Launch Site

- More payload have higher success rate but over all no clear relationship between payload and launch site.



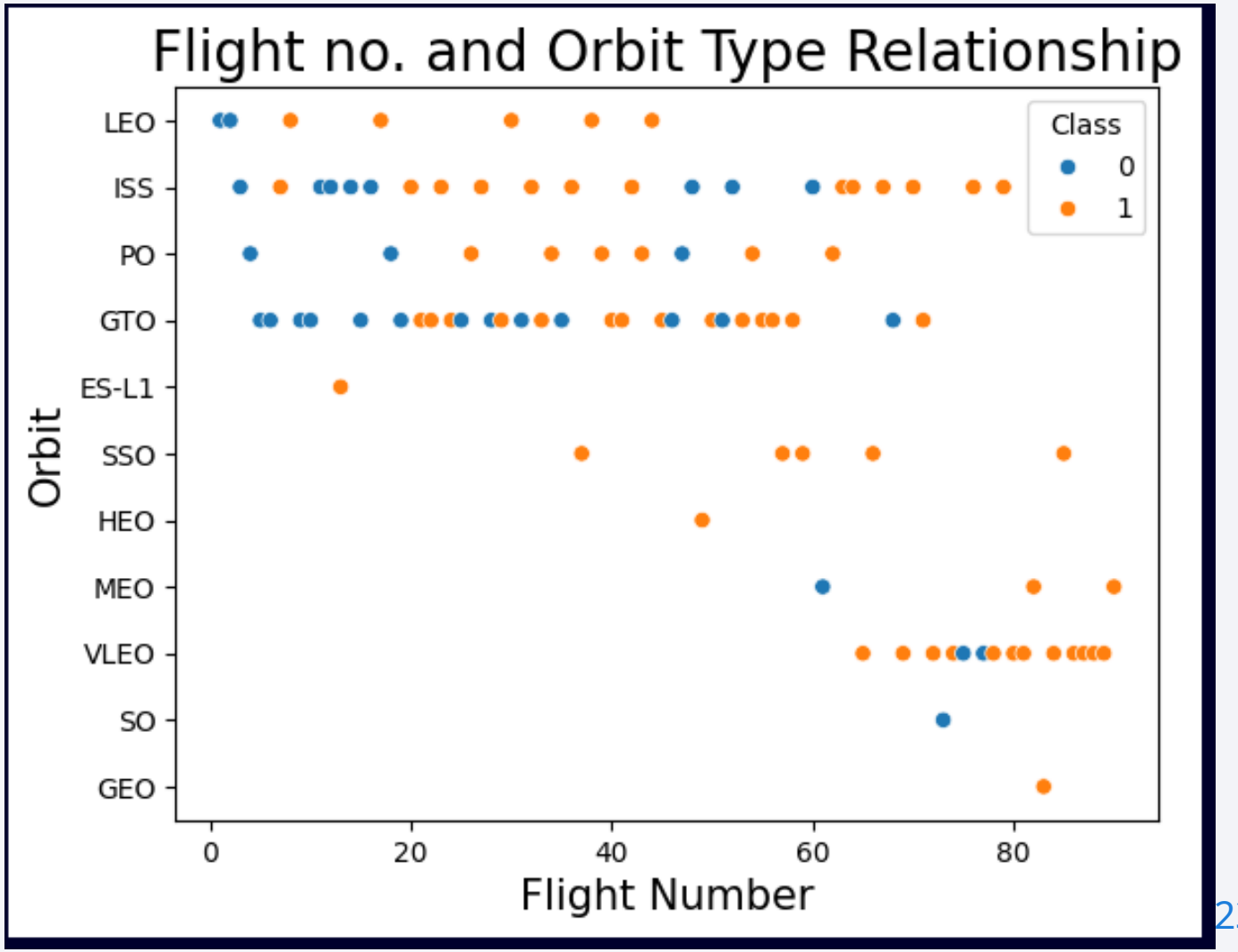
Success Rate vs. Orbit Type

- Orbit SSO, ES-11 AND GEO has more success rate as compare to others.



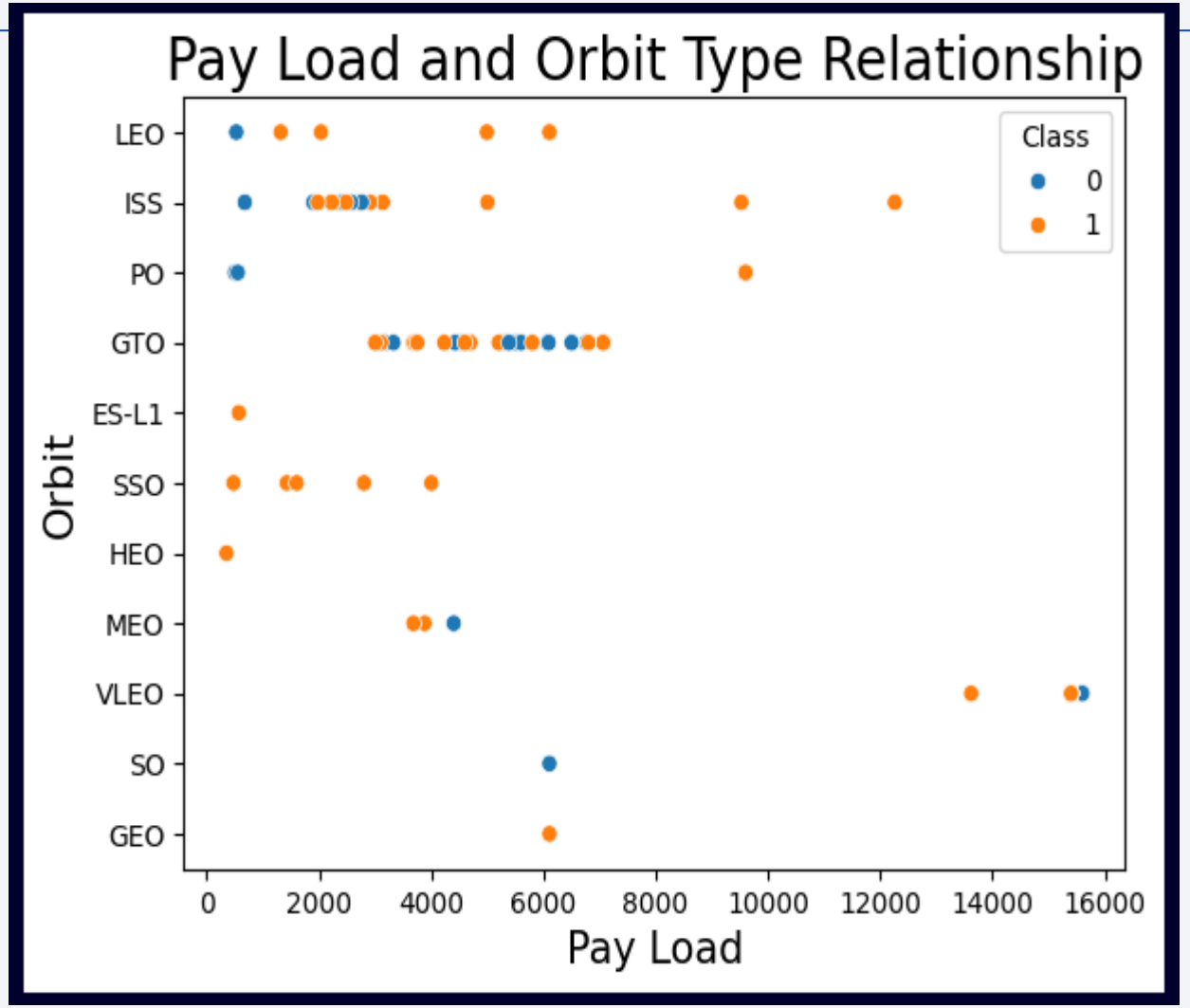
Flight Number vs. Orbit Type

- Looks like ortbit GTO dosen’t have relationship with payload.



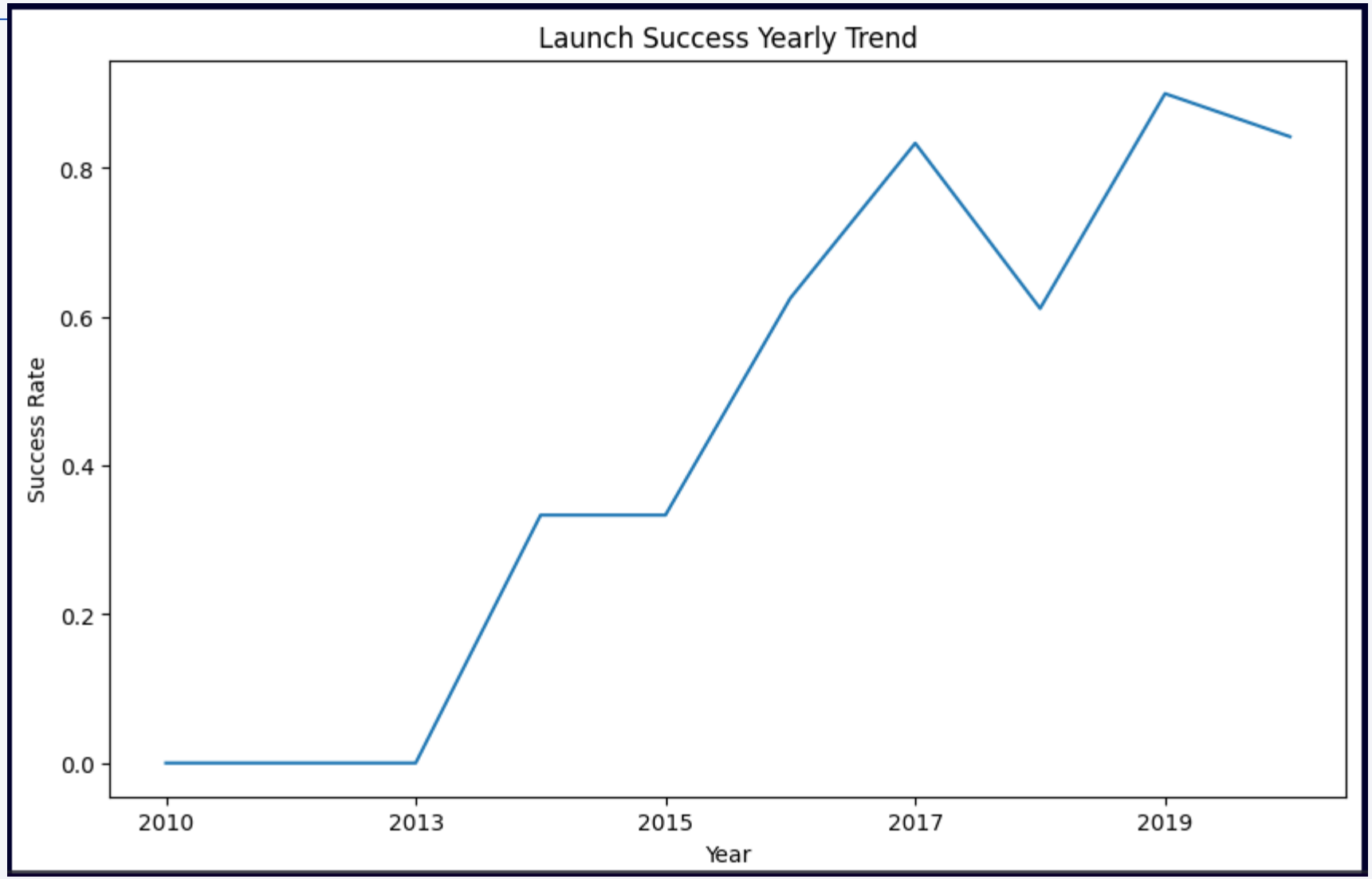
Payload vs. Orbit Type

- With heavy payload VLEO, PO, ISS has more success rate.



Launch Success Yearly Trend

- Since 2013 success rate goes up until 2017 than droped until 2018 and 2019 and onward success rate is about 80%



All Launch Site Names

With sql DISTINCT query we fetch uniqe site names from db.

Which are

CCAF LC-40, VAFB SLC-4E, KSC LC39A, AND CCAFS SLC-40.

```
%%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'

LIKE ‘CCA%’ query is used to get site names 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
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success
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
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success

Total Payload Mass

Sql query SUM(PAYLOAD_MASS__KG_) USED WHERE CUSTOMER IS ‘NASA(CRS)’

```
%%sql
SELECT SUM(PAYLOAD_MASS__KG_) FROM spacextbl WHERE Customer = 'NASA (CRS)' ;
```

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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

SQL AVG used to get average here

```
%%sql
SELECT AVG(PAYLOAD_MASS__KG_) FROM spacextbl WHERE Booster_Version LIKE 'F9 v1.1' ;
```

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

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

Sql minimum value query used to extract 1st successful landing.

```
%%sql
SELECT min(Date) AS min_date FROM spacextbl WHERE Landing_Outcome = 'Success (ground pad)';
```

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

min_date
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
%%sql
SELECT Booster_Version FROM spacextbl WHERE (PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 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

Total Number of Successful and Failure Mission Outcomes

Total number of successful and failure mission outcomes

Sql COUNT AND GROUP BY used here.

```
%%sql
SELECT Mission_Outcome, COUNT(Mission_Outcome) as counts from spacextbl group by Mission_Outcome;
```

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

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

+ Code + Markdown

Boosters Carried Maximum Payload

Names of the booster which have carried the maximum payload mass

Sql subquery used to get maximum payload

```
%%sql
SELECT 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

2015 Launch Records

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

```
%%sql
SELECT Date, Landing_Outcome, Booster_Version, Launch_Site FROM spacextbl WHERE Landing_Outcome LIKE '%Failure%' AND substr(Date, 0, 5) = '2015';
```

Python

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

Date	Landing_Outcome	Booster_Version	Launch_Site
2015-01-10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
2015-04-14	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

Ranking 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

```
%%sql
SELECT Landing_Outcome, COUNT(*) AS LandingCounts from spacextbl WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY COUNT(*) DESC;
```

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

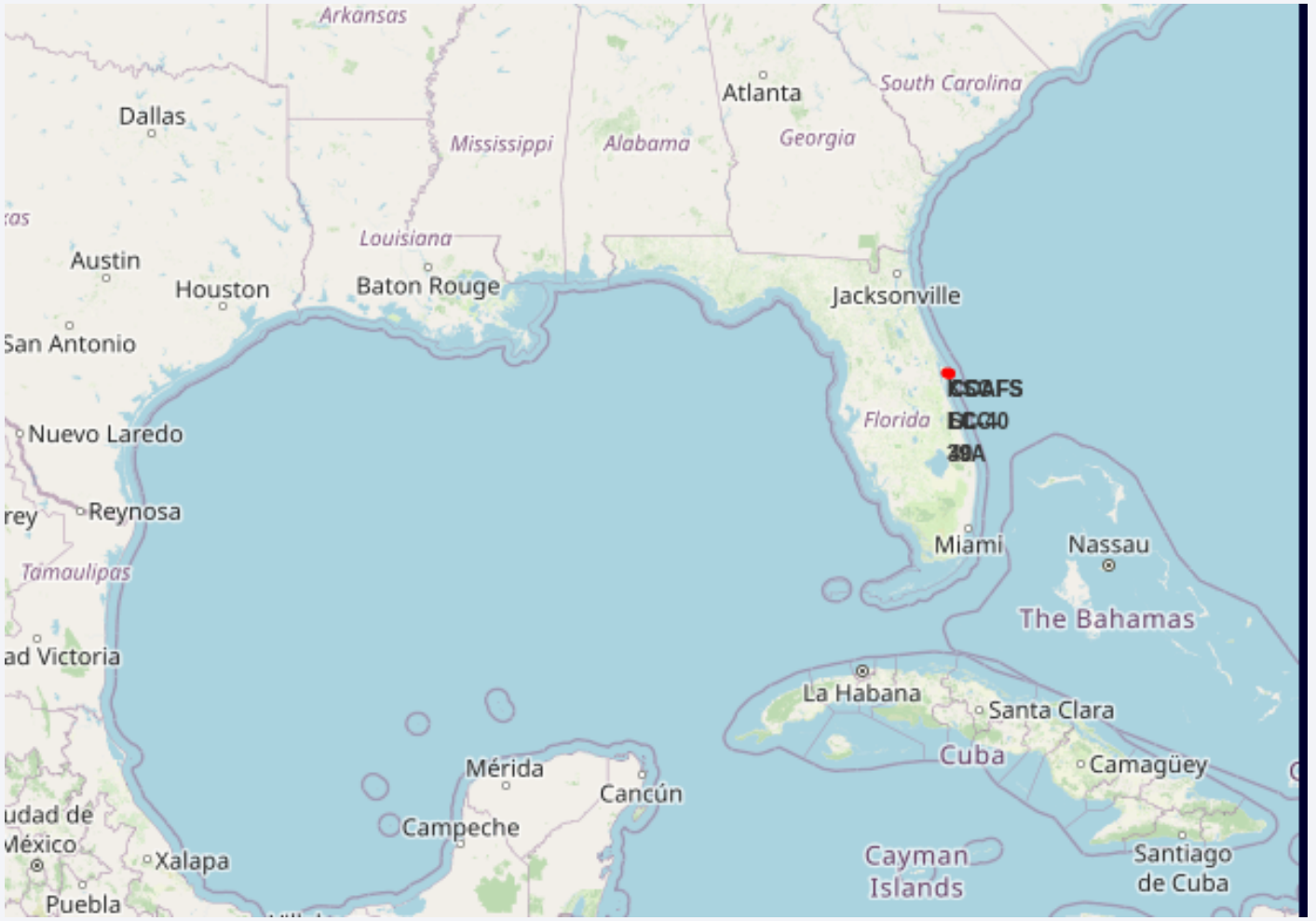
Landing_Outcome	LandingCounts
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Section 3

Launch Sites Proximities Analysis

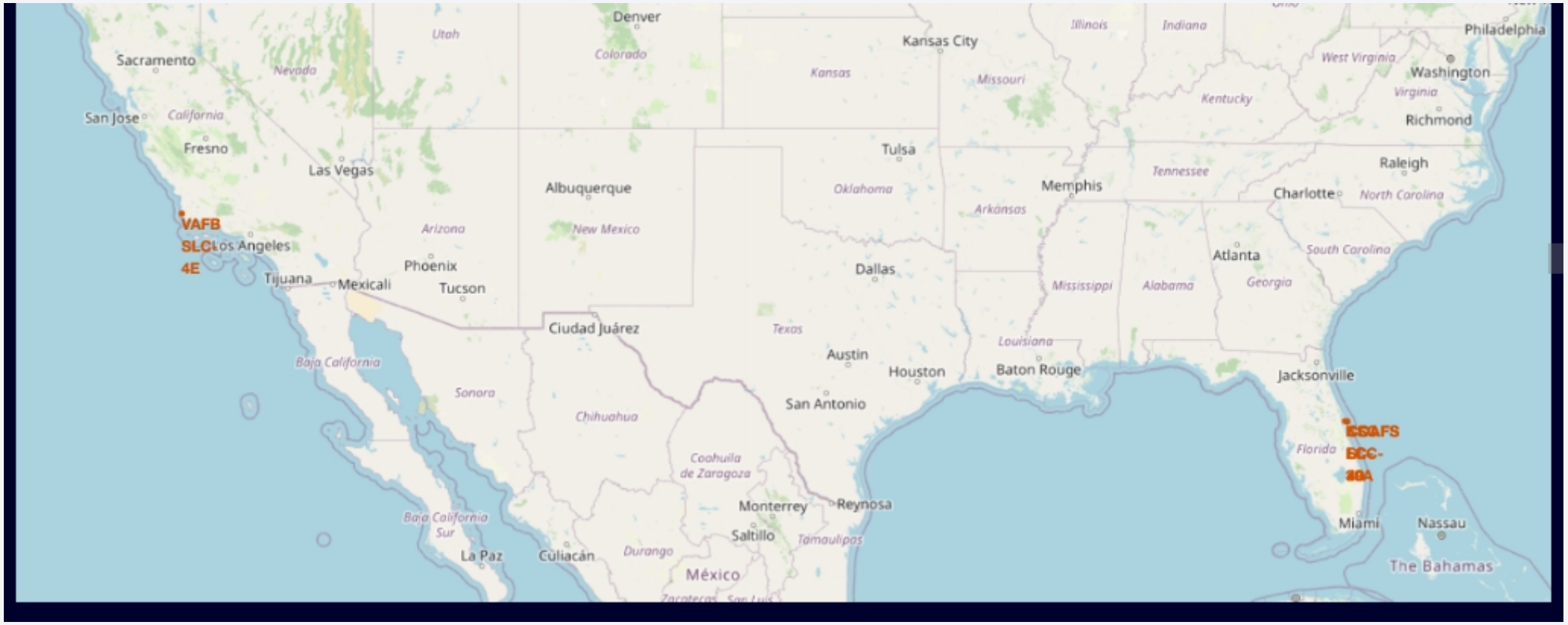
Launch Sites

Launch sites are near by costal line

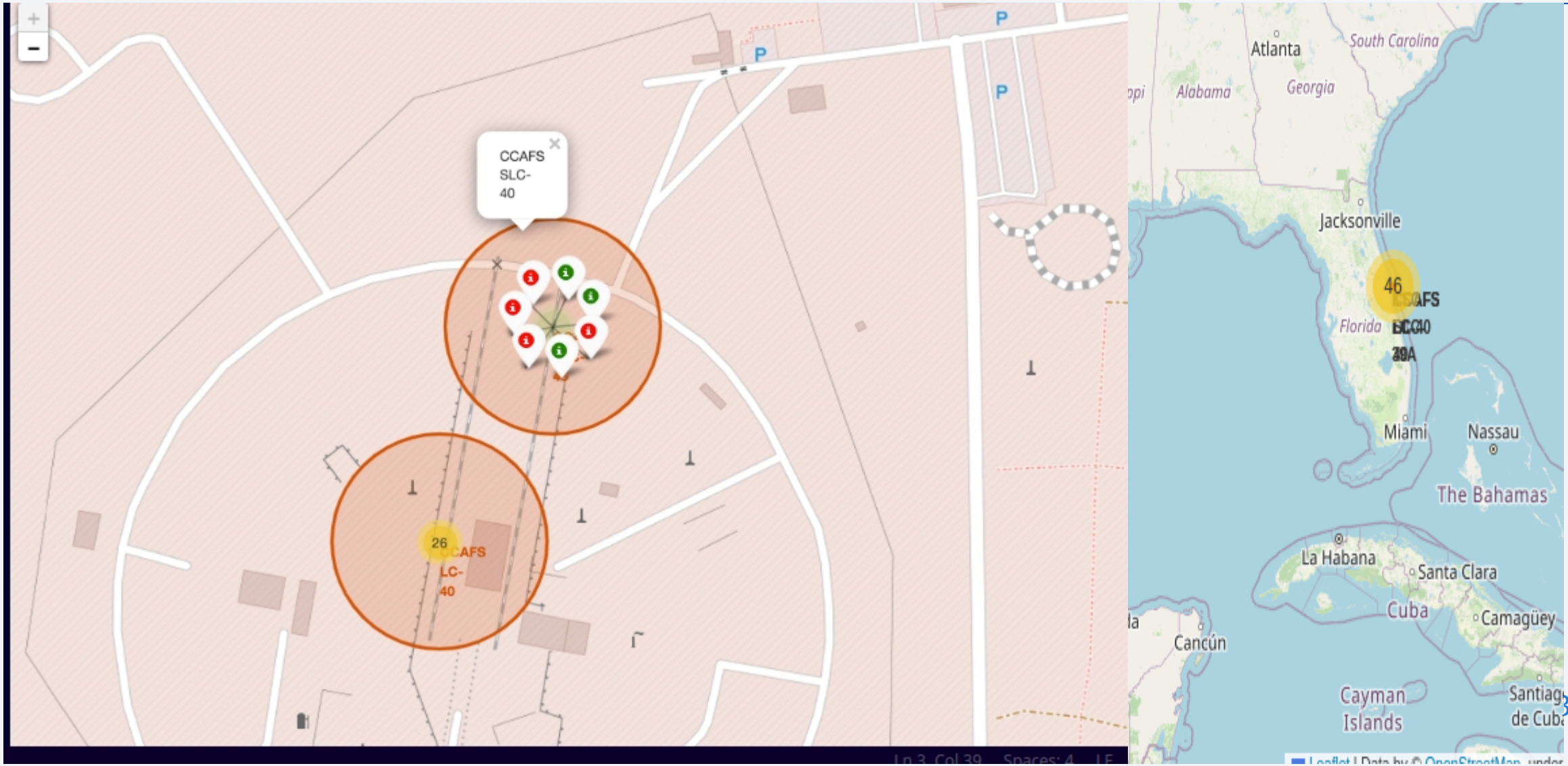


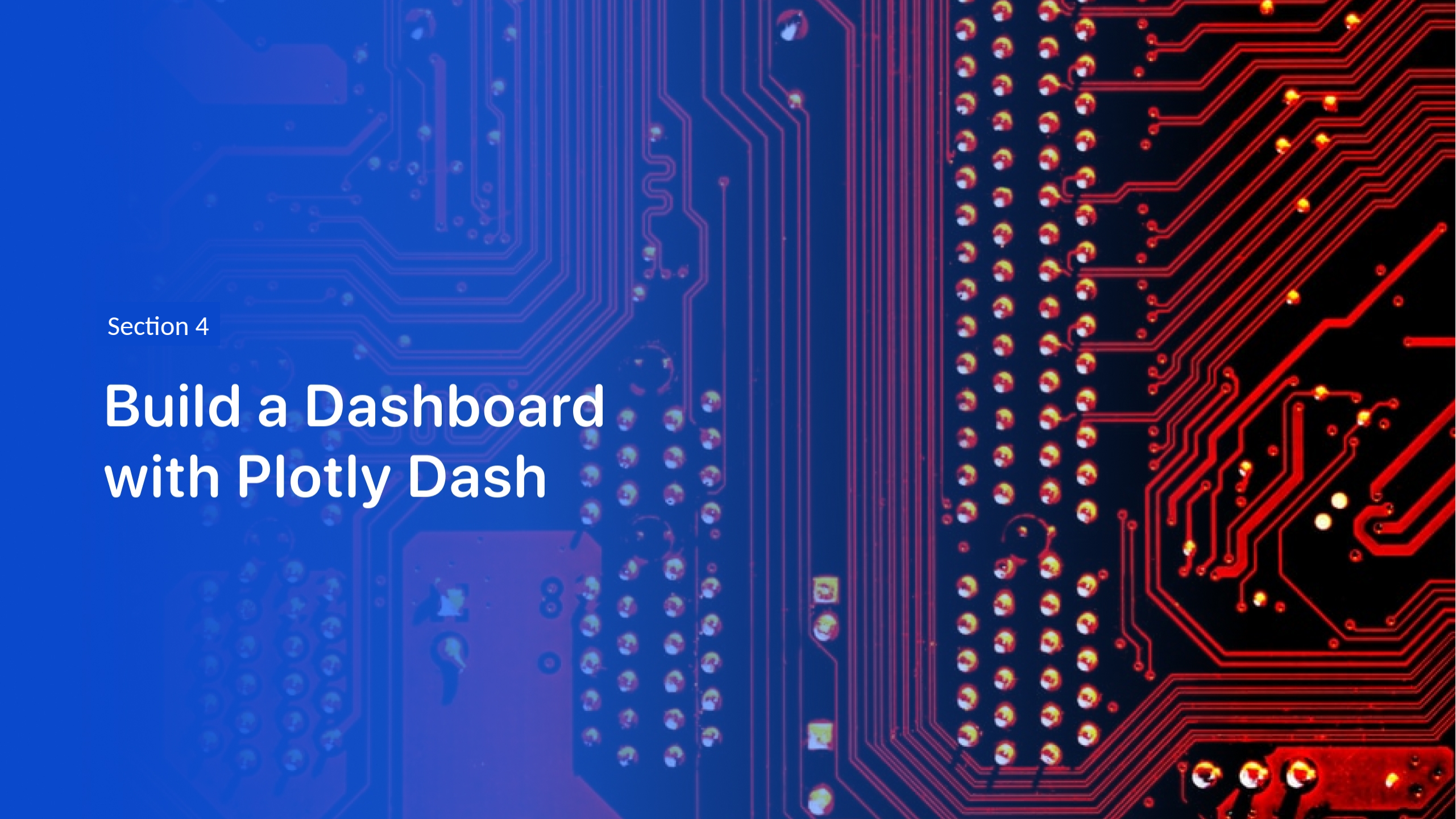
Site locations

All launch site near by costal line and away from facilities like railway stations, parks etc.



Distance to facilities

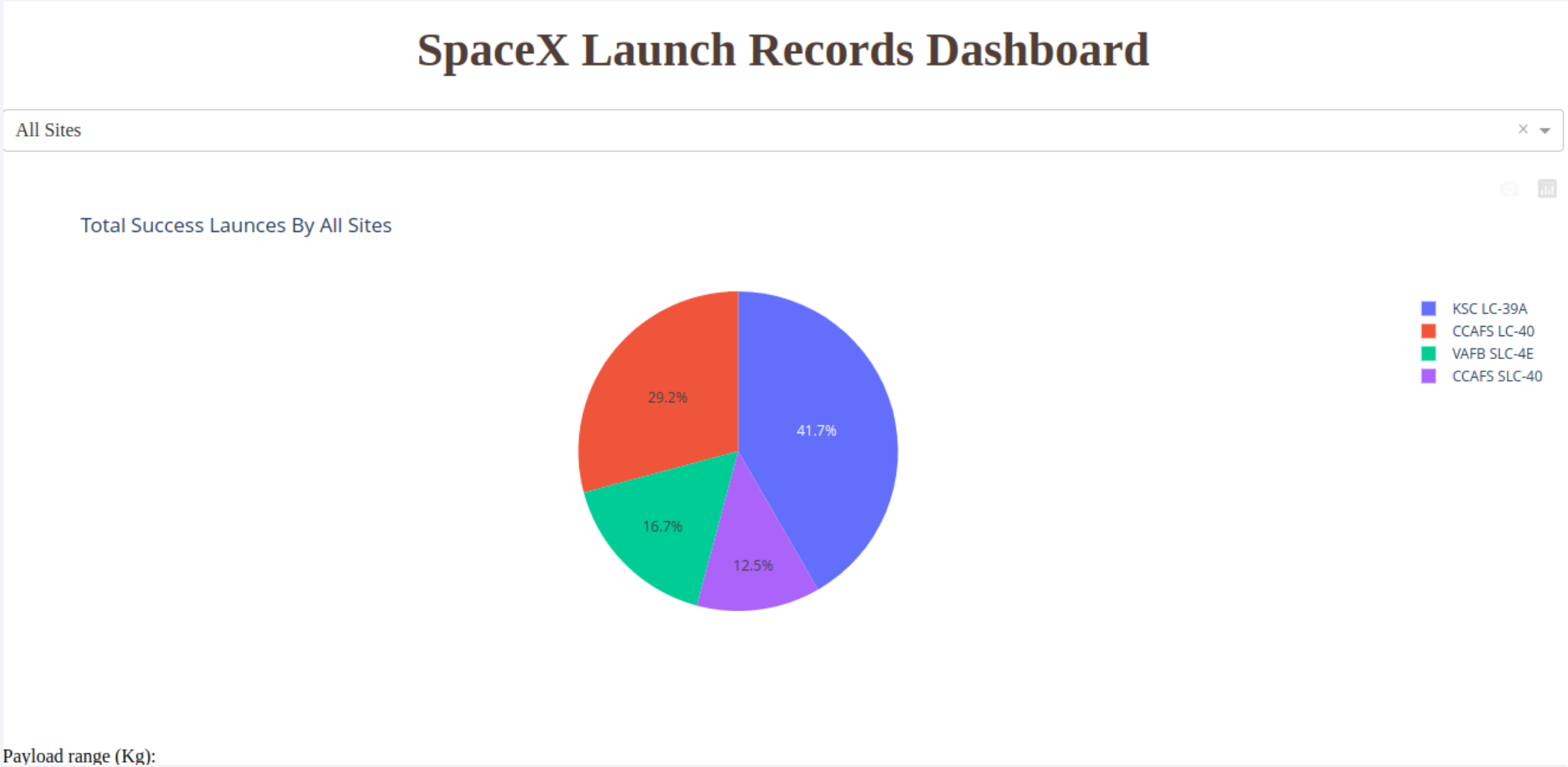




Section 4

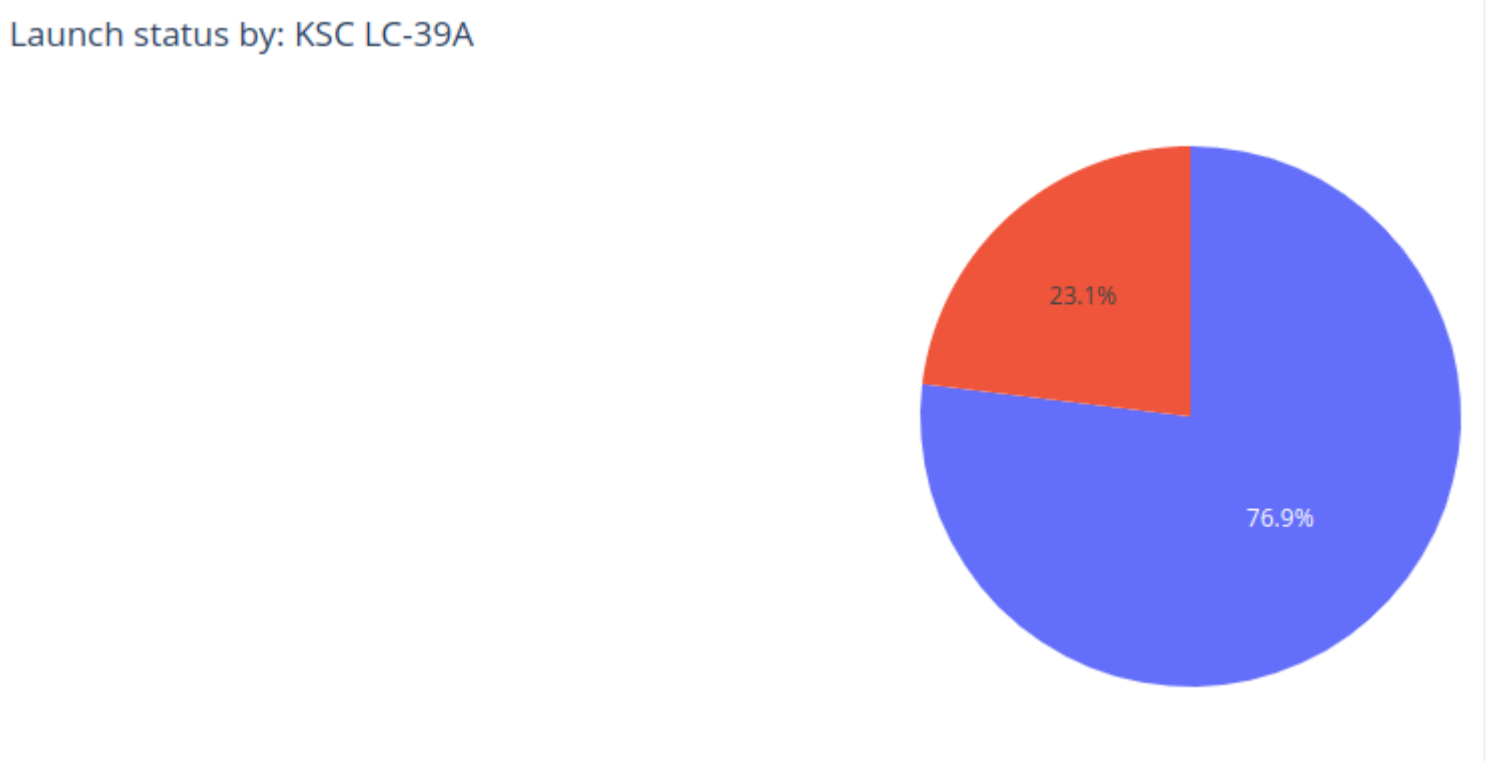
Build a Dashboard with Plotly Dash

KSC LC-39A is the most successful launch site

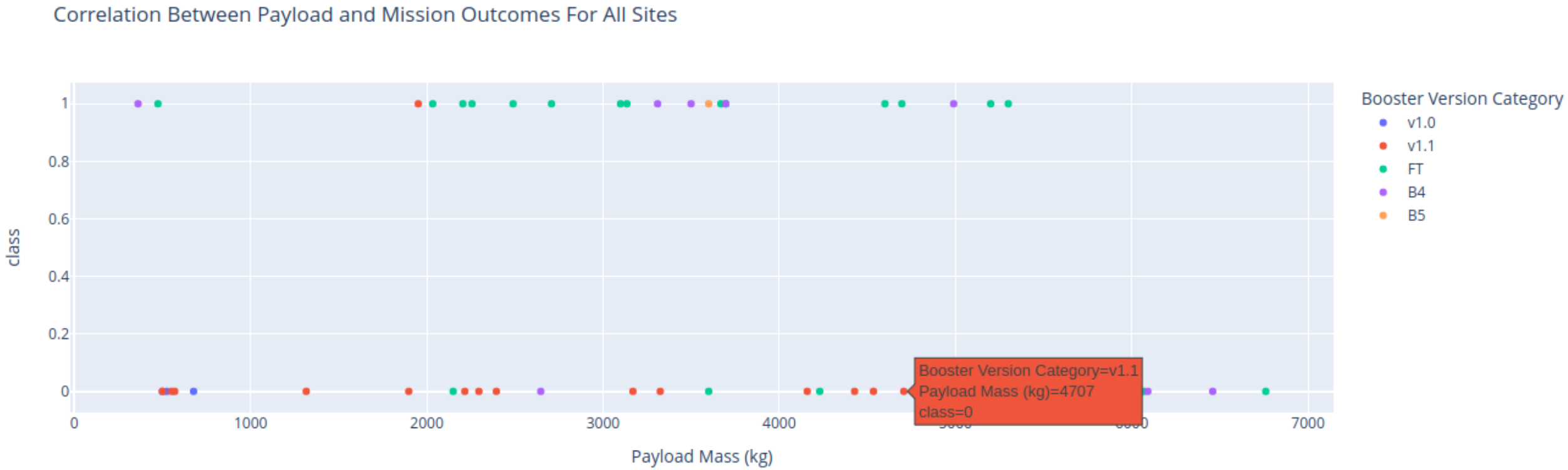


Best launch site

KSC LC-39A is the most successful launch site with 74% success rate



Correlation between payload and mission outcome for all sites



Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Accuracy score of SVM is highest by 90% decision tree and logistic regression is 80% and KNN on 66%

- SVM is best at this time.

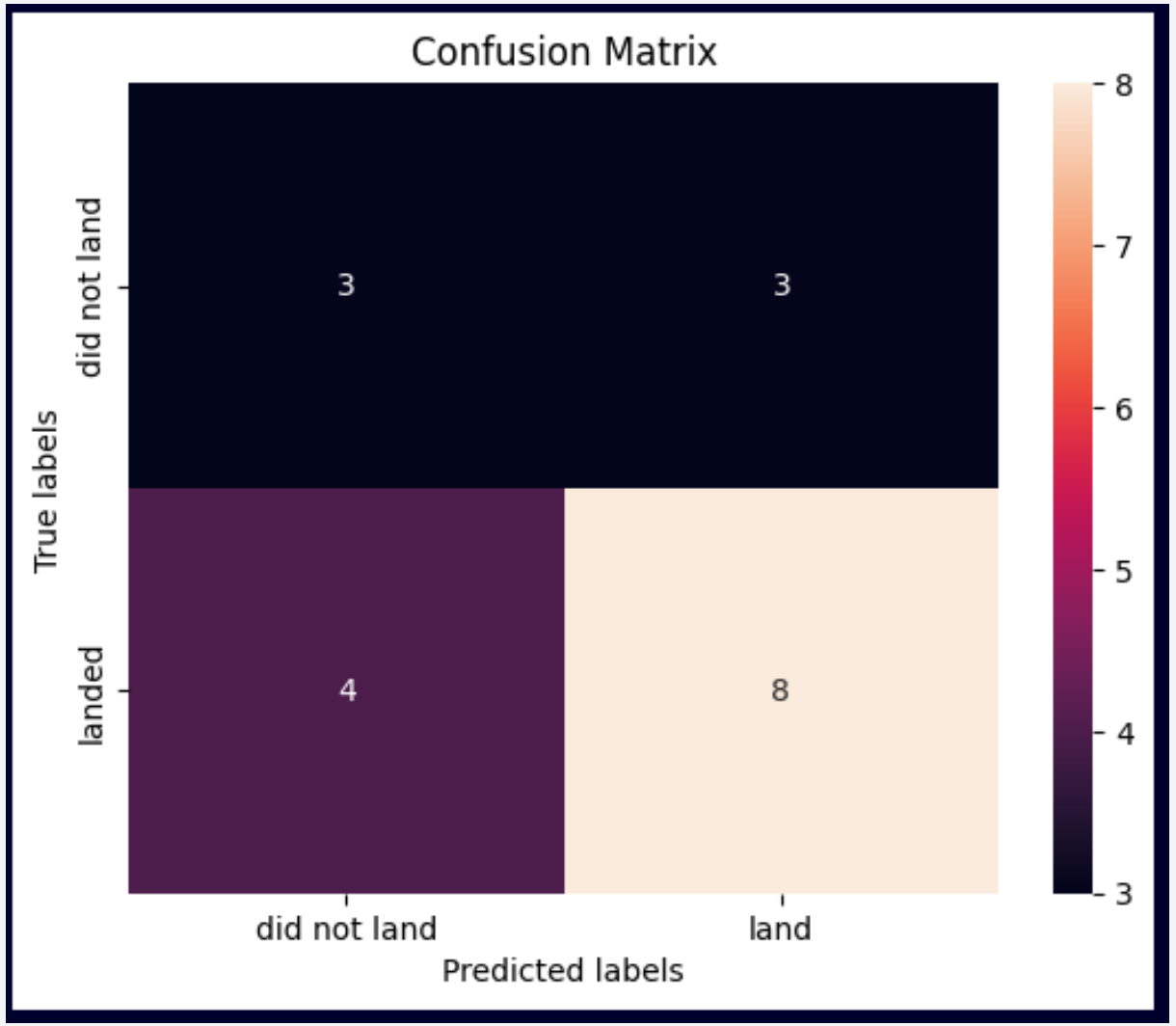
```
Model_Performance_df.sort_values(['Accuracy Score'], ascending = False, inplace=True)
Model_Performance_df
```

✓ 0.0s

	Algo Type	Accuracy Score	Test Data Accuracy Score
1	SVM	0.900000	0.666667
2	Decision Tree	0.889286	0.888889
0	Logistic Regression	0.819643	0.833333
3	KNN	0.664286	0.611111

Confusion Matrix

- These models predict successful landings.



Conclusions

As number of flights increased, success rate increases and after 2018 its around 80%

Orbit types ES-L1, GEO, HEO & VLEO have success rate of 100%

Launch sites are close to railways, highways and costal line but away from populated areas.

Dataset is too small to make reasonable predictions but SVM performs good when use with subset feature. Classification and decision tree is on 80% on this dataset.

...

Appendix

github

- https://github.com/yasir134/my_projects/tree/main/DataScientist_capstone

dashboard

- https://myasiriqbal-8050.theiadockernext-1-labs-prod-theiak8s-4-tor01.proxy.cognitiveclass.ai/

Thank you!