IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Asher Maor
08/14/2022

Outline

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

Executive Summary

- Summary of methodologies
  - Two types of data collection:
    - Via API
    - WEB scraping
  - Data Wrangling
  - Analyzing data with:
    - SQL
    - Via Data Visualization (with Dashboard and Folium Maps)
  - Implementing Machine Learning classifiers to predict successful outcome.
- Summary of all results
  - Successful outcome with accuracy of 94% is obtained using Decision Tree Classifier.

Introduction

- Project background and context
- SpaceX launches annually dozen of Falcon 9 rockets SpaceY would like to estimate launch outcomes if it will enter into competetion with SpaceX.
- Problems you want to find answers
  - Find correlation between all independent variables.
  - Find all correlation between successful launches and all available data.
  - What is the best model that predicts launch outcome so we can rely on it on our decision.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- Describe how data sets were collected.
- You need to present your data collection process use key phrases and flowcharts

Data Collection – SpaceX API

- Get data by request, normalization and filtering

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

Collect Data

In [ ]:
```
response = requests.get(spacex_url)
```

Convert into Dataframe

In [ ]:
```
data=pd.json_normalize(response.json())
```

Cleaning and preparing

In [ ]:
```
data['date'] = pd.to_datetime(data['date_utc']).dt.date
data = data[data['date'] <= datetime.date(2020, 11, 13)]
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
isFalcon9=data2['BoosterVersion']!='Falcon 1'
data_falcon9=data2[isFalcon9]
```

Data Collection - Scraping

- Scraping using Beautiful soup, parsing the table, converting to DataFrame and saving to csv

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

Scraping using Beautiful Soup

In [ ]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [ ]:
```
response = requests.get(static_url)
soup = BeautifulSoup(response.text,"html.parser")
```

Parsing

In [ ]:
```
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
            extracted_row += 1
            # Flight Number value

            # TODO: Append the flight_number into launch_dict with key `Flight No.`

            if not isinstance(launch_dict['Flight No.'], list):
            # If type is not list then make it list
              launch_dict['Flight No.'] = [launch_dict['Flight No.']]
            launch_dict['Flight No.'].append(flight_number)
            ...
```

Converting to DataFrame and Saving

In [ ]:
```
df=pd.DataFrame(launch_dict)
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

- Exploratory data analysis was performed
- Data loaded and cleaned from null values, calculated required values and created outcomes table.
- GitHub URL of your completed data wrangling related notebooks, as an external reference and peer-review purpose

Data loading and cleaning missing values

```
In [ ]: df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_pa
```

```
In [ ]: df.isnull().sum()/df.count()*100
```

Calculating # of occurencies at each orbit and # of launches at each site

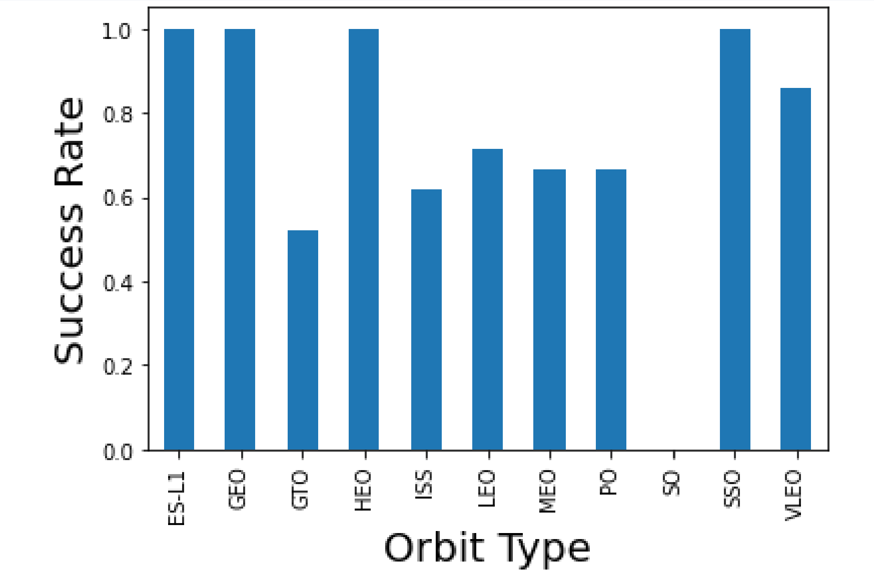
```
In [ ]: df['LaunchSite'].value_counts()
        df['Orbit'].value_counts()
```

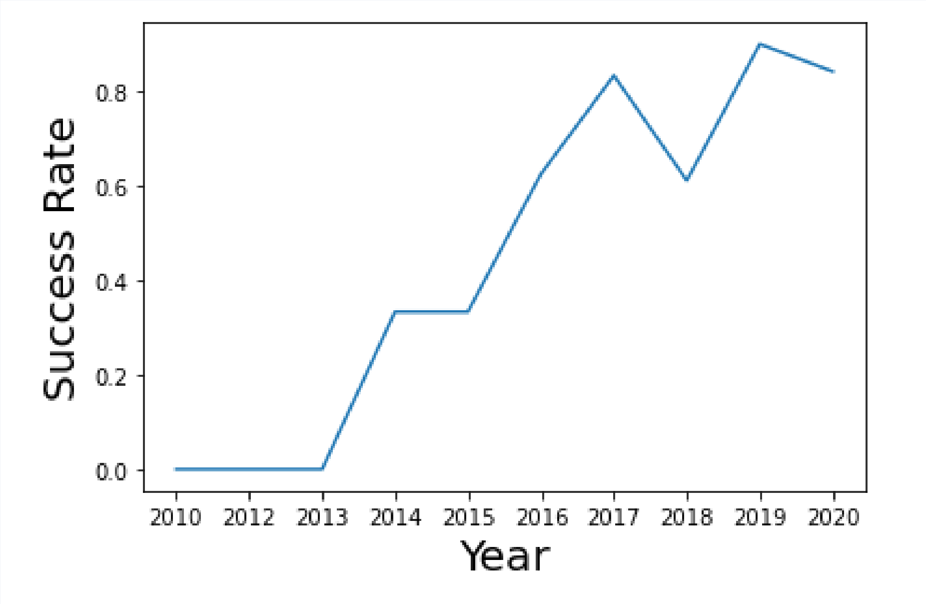
Creating outcomes labels and saving to cvs

```
In [ ]: for i,outcome in enumerate(landing_outcomes.keys()):
            print(i,outcome)
        bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
        landing_class = df['Outcome'].replace({'False Ocean': 0, 'False ASDS': 0, 'None None': 0, 'None ASDS': 0, 'False RTLS': 0, 'True
        df['Outcome'] = df['Outcome'].astype(int)
        df.to_csv("dataset_part_2.csv", index=False)
```

EDA with Data Visualization

- Right chart shows Success Rate vs. Orbit Type
- Left chart shows Success Rate vs. Year. One may notice that success rate grows with time
- GitHub URL





EDA with SQL

We have worked in EDA using MagicSQL to process the data. Here is the list of main queries performed:

- -The names of unique launch sites in the space mission.
- -The total payload mass carried by boosters launched by NASA (CRS)
- -The average payload mass carried by booster version F9 v1.1
- -The total number of successful and failure mission outcomes and their corresponding time and date
- -The failed landing outcomes in drone ship, their booster version and launch site names and their corresponding time and date
- GitHub URL

Build an Interactive Map with Folium

All launch sites have been mapped, including additional objects such as markers, circles, lines to mark the success or failure of launches for each site on the folium map.

- Launch outcomes (failure or success) to assigned to class 0 and 1.(0 for failure, 1 for success).
- Using the color-labeled marker clusters, high success rate
- launch sites were identified.
- We calculated the distances between a launch site to its cities, coastlines, railways.
- It clear that a lot of cities in proximites but still pretty far and certain distance must be kept.
- GitHub URL of your completed interactive map with Folium map, as an external reference and peer-review purpose

Build a Dashboard with Plotly Dash

Plots were generated to answer next questions (answers inside links)

- Which site has the largest successful launches?
- Which site has the highest launch success rate?
- Which payload range(s) has the highest launch success rate?
- Which payload range(s) has the lowest launch success rate?
- Which F9 Booster version (`v1.0`, `v1.1`, FT, B4, B5, etc.) has the highest
- launch success rate?
- Add the GitHub URL of python code.

Predictive Analysis (Classification)

- Data were loaded using panda and transformed using StandardScaler. Then, it was split into training and testing sets.
- Various machine learning models were simulated with different tuned hyperparameters for best accuracy determination.
- Best accuracy classification model was obtained.
- GitHub URL

Results

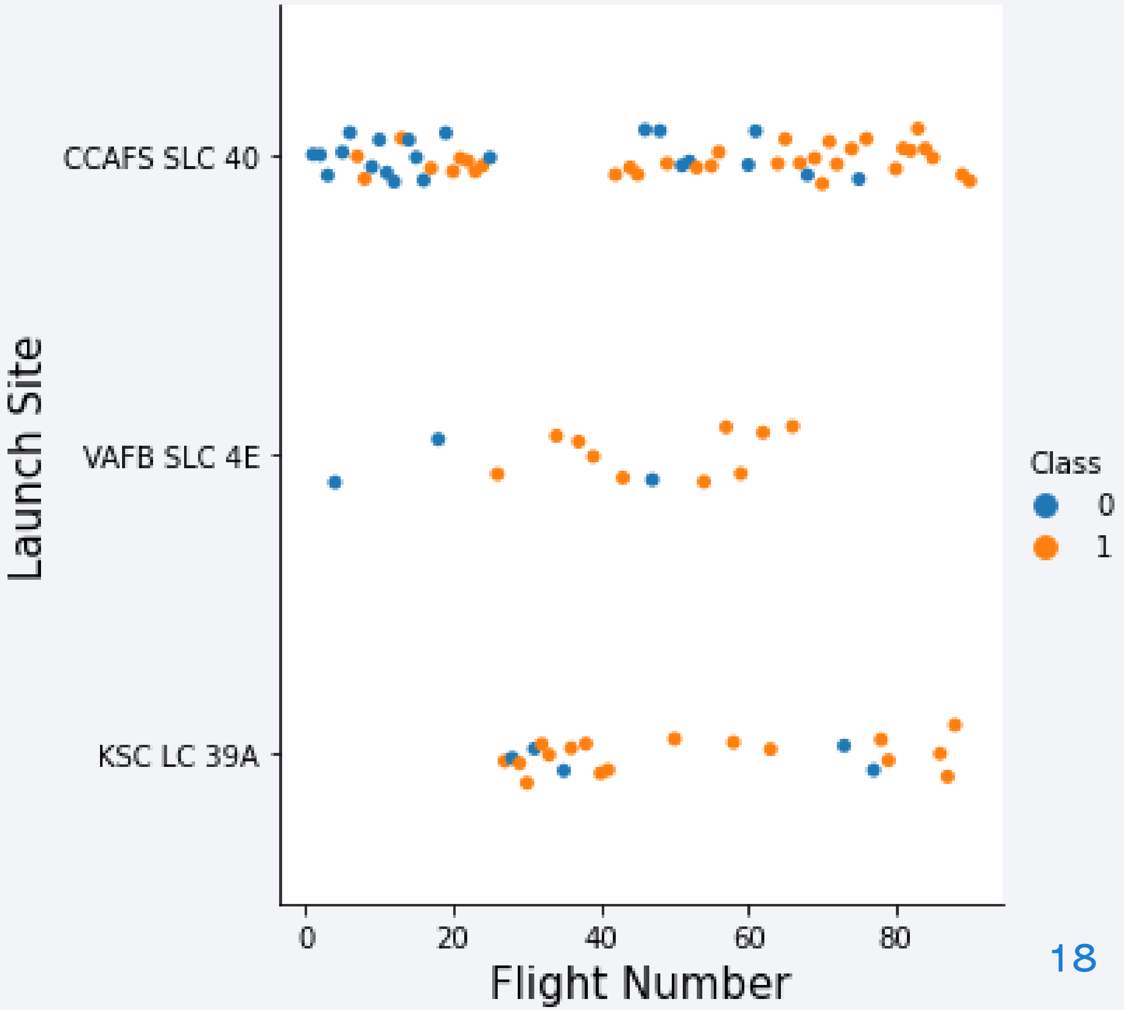
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

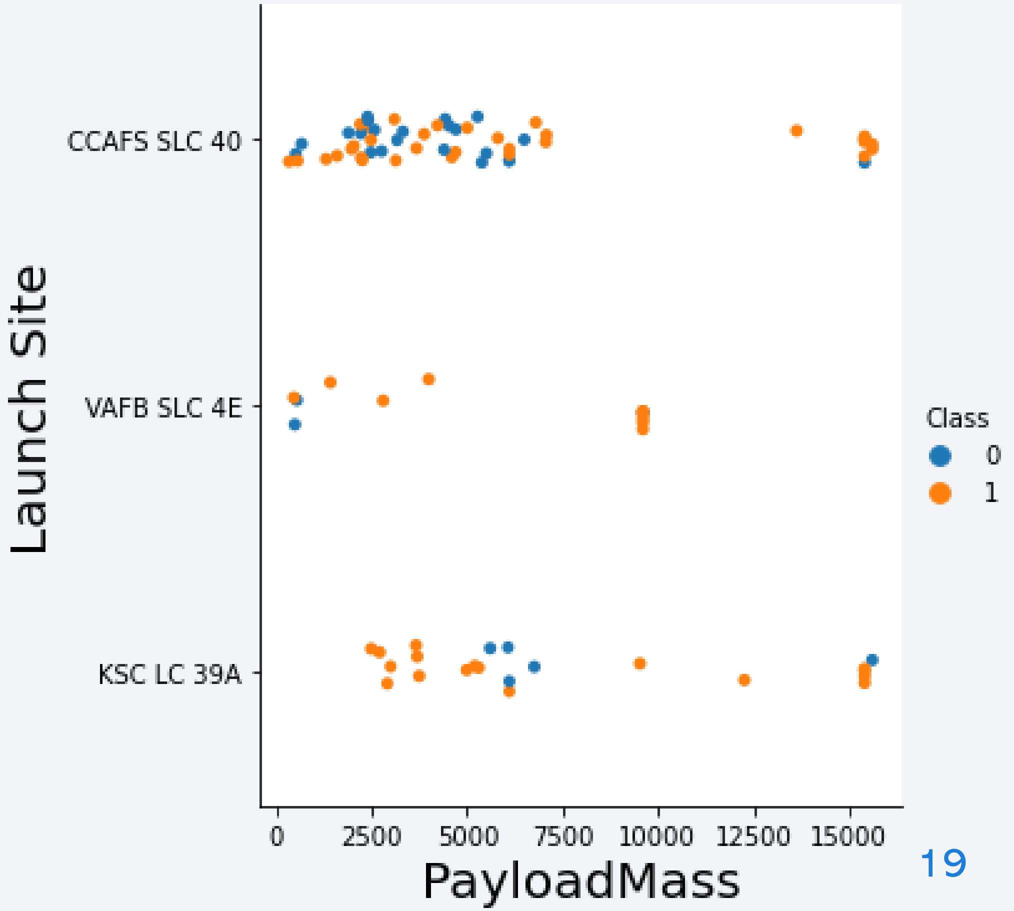
Flight Number vs. Launch Site

- Findings:
  - For CCAFS SLC 40 and VAFB SLC 4E, launch outcomes improve with the time.
  - KSC LC 39A keeps roughly the same number of successive lunched



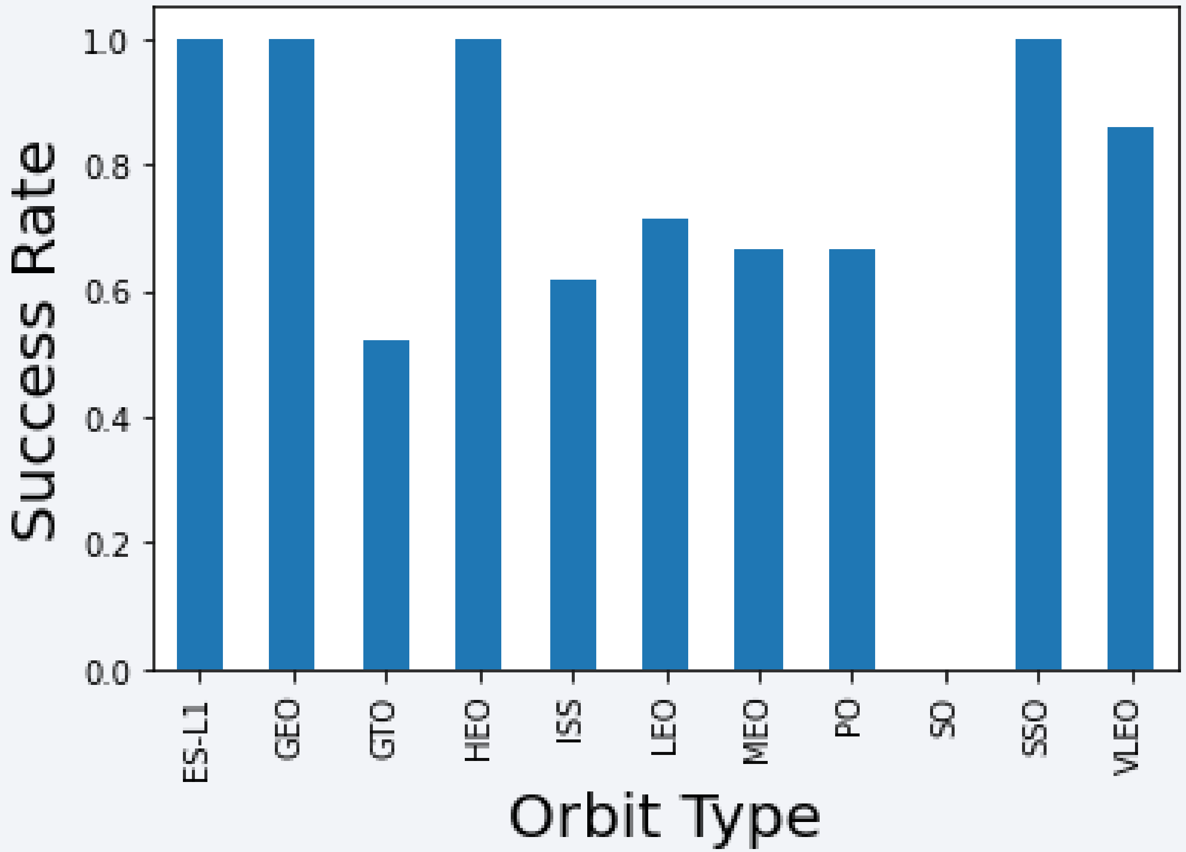
Payload vs. Launch Site

- Findings:
  - For CCAFS SLC 40, heavy boosters have high successive rate.
  - VAFB9 SLC4E site (only for small and middle size boosters) and KSC LC39A (besides 6000ks) gives good launch outcomes for almost all Payload Masses.



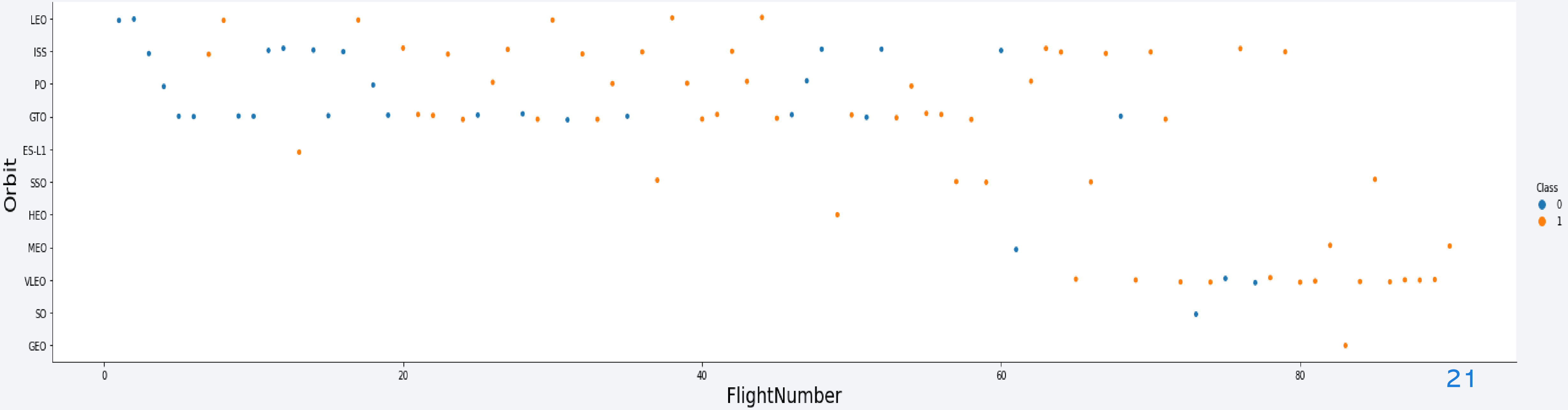
Success Rate vs. Orbit Type

- Findings:
  - ES-L1, GEO, HEO and SSO are success rate orbit types.
  - Worst orbit type is SO.
  - GTO orbit has the same number of failure and success outcomes (says nothing)



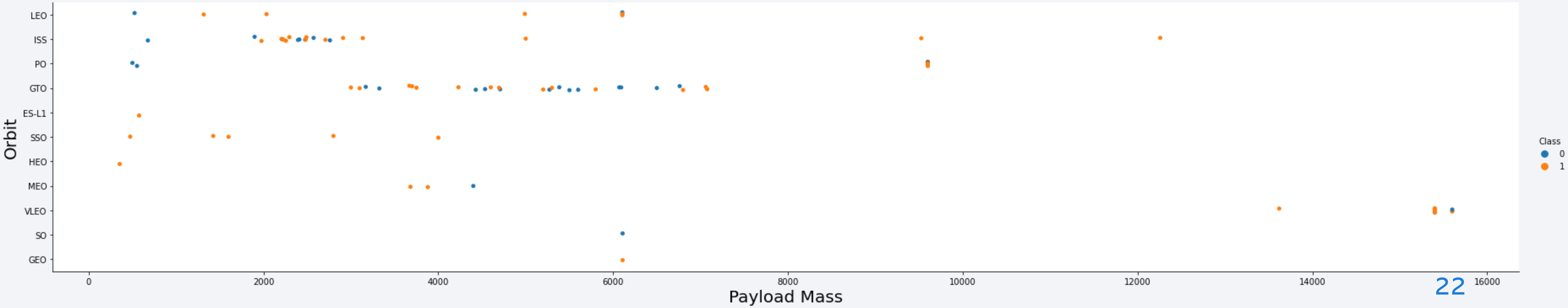
Flight Number vs. Orbit Type

- LEO orbit improves with time
- ISS does not have any relationship



Payload vs. Orbit Type

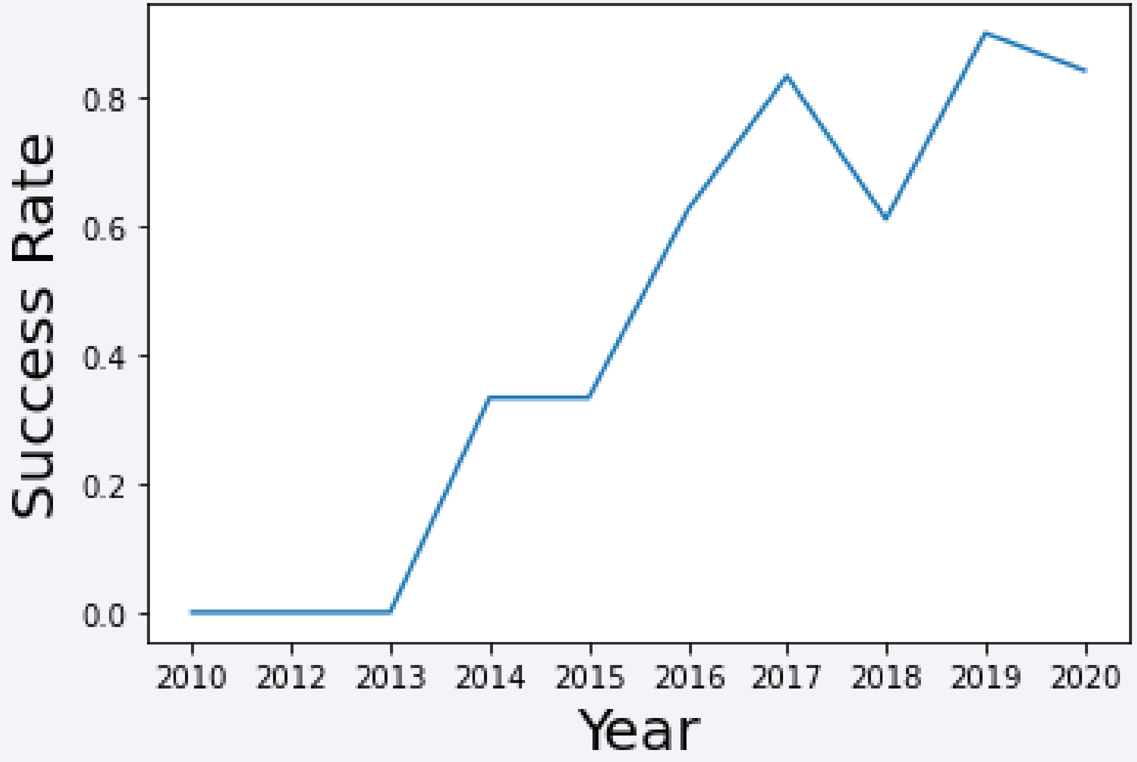
- SSO is successive up to 4000 kg boosters
- GFO has not relationship to kind of booster (in terms of success rate)



Launch Success Yearly Trend

- It may be clearly seen that success rate has grown with years

- Looks like rate of growth gets smaller



All Launch Site Names

- Distinct Launch_site is selected by query from table called SPACEXTBL2

```
In [163]: %%sql
          select distinct Launch_Site from SPACEXTBL2;

             sqlite:///my_data1.db
             sqlite:///my_data2.db
           * sqlite:///my_data3.db
          Done.
```

Out[163]:

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

Launch Site Names Begin with 'CCA'

- Here, only Launch_site that begins with 'CCA%' (in SQL) are dispayed (max 5 rows)

In [164]:
```
%%sql
select Launch_Site
from SPACEXTBL2
where Launch_Site like 'CCA%'
LIMIT 5
```

```
   sqlite:///my_data1.db
   sqlite:///my_data2.db
 * sqlite:///my_data3.db
Done.
```

Out[164]:

LAUNCH_SITE
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- Here, the query is specified so only NASA (CRS) as customer. Overall mass is calculated with sum()

```
In [165]: %%sql
          select sum(PAYLOAD_MASS__KG_)
          from SPACEXTBL2
          WHERE Customer='NASA (CRS)'

             sqlite:///my_data1.db
             sqlite:///my_data2.db
           * sqlite:///my_data3.db
          Done.

Out[165]: sum(PAYLOAD_MASS__KG_)
                           45596
```

Average Payload Mass by F9 v1.1

- Average Payload mass (in kg) is calculated by avg() in select part only for Booster_version of F9 v1.1.

```
In [166]: %%sql
select AVG(PAYLOAD_MASS__KG_)
from SPACEXTBL2
WHERE Booster_Version='F9 v1.1'

   sqlite:///my_data1.db
   sqlite:///my_data2.db
 * sqlite:///my_data3.db
Done.
```

Out[166]:

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- First date (early date) calculated using min() in select part of query

```
In [169]: %%sql
          select min(Date)
          from SPACEXTBL2
          WHERE Landing__Outcome='Success (ground pad)'

             sqlite:///my_data1.db
             sqlite:///my_data2.db
           * sqlite:///my_data3.db
          Done.

Out[169]:
```

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- Query of Booster_Version is defined by range of PayLoad (in kg) between 4 and 6 tons.

```
In [170]: %%sql
select Booster_Version
from SPACEXTBL2
WHERE Landing__Outcome='Success (drone ship)' and PAYLOAD_MASS__KG_>4000 and PAYLOAD_MASS__KG_<6000

   sqlite:///my_data1.db
   sqlite:///my_data2.db
 * sqlite:///my_data3.db
Done.
```

Out[170]:

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

Total Number of Successful and Failure Mission Outcomes

- Count(*) is used to calculate 1's, and all other are calculated by using sub-query (using not like)

In [172]:
```
%%sql
select count(*) as success, (select count(*) from SPACEXTBL2 where mission_outcome not like 'Success%') as failures
from SPACEXTBL2
where mission_outcome like 'Success%'
```

```
   sqlite:///my_data1.db
   sqlite:///my_data2.db
 * sqlite:///my_data3.db
Done.
```

Out[172]:

success	failures
100	1

Boosters Carried Maximum Payload

- The query is performed using group by Booster_version

In [173]:
```
%%sql
select Booster_Version,max(PAYLOAD_MASS__KG_)
from SPACEXTBL2
GROUP BY Booster_Version ORDER BY max(PAYLOAD_MASS__KG_) DESC
```

```
   sqlite:///my_data1.db
   sqlite:///my_data2.db
 * sqlite:///my_data3.db
Done.
```

Out[173]:

BOOSTER_VERSION	max(PAYLOAD_MASS__KG_)
F9 B5 B1060.3	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1056.4	15600
F9 B5 B1051.6	15600
F9 B5 B1051.4	15600
F9 B5 B1051.3	15600
F9 B5 B1049.7	15600
F9 B5 B1049.5	15600

2015 Launch Records

- Here, substr() was used to assign Month name to date of failure (drone ship) landing outcomes for all dates beginning with '2015%' (in SQL)

In [199]:
```
%%sql
select substr('JanFebMarAprMayJunJulAugSepOctNovDec', 1 + 3*strftime('%m', Date), -3) as Month, Landing__Outcome, Booster_Versior
from SPACEXTBL2
where Landing__Outcome='Failure (drone ship)' and Date like '2015%'
```

```
   sqlite:///my_data1.db
   sqlite:///my_data2.db
 * sqlite:///my_data3.db
Done.
```

Out[199]:

Month	LANDING__OUTCOME	BOOSTER_VERSION	LAUNCH_SITE	DATE
Jan	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40	2015-01-10
Apr	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40	2015-04-14

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

- Ranking was performed by using count(*) for Landing_Outcome that has a word "Success" in value and limited within range of dates.

In [202]:
```
%%sql
select Landing__Outcome,count(*)
from SPACEXTBL2
where (Date >'2010-06-04') AND (Date < '2017-03-20') and Landing__Outcome LIKE '%Success%'
group by Landing__Outcome
order by count(*) DESC
```

```
   sqlite:///my_data1.db
   sqlite:///my_data2.db
 * sqlite:///my_data3.db
Done.
```

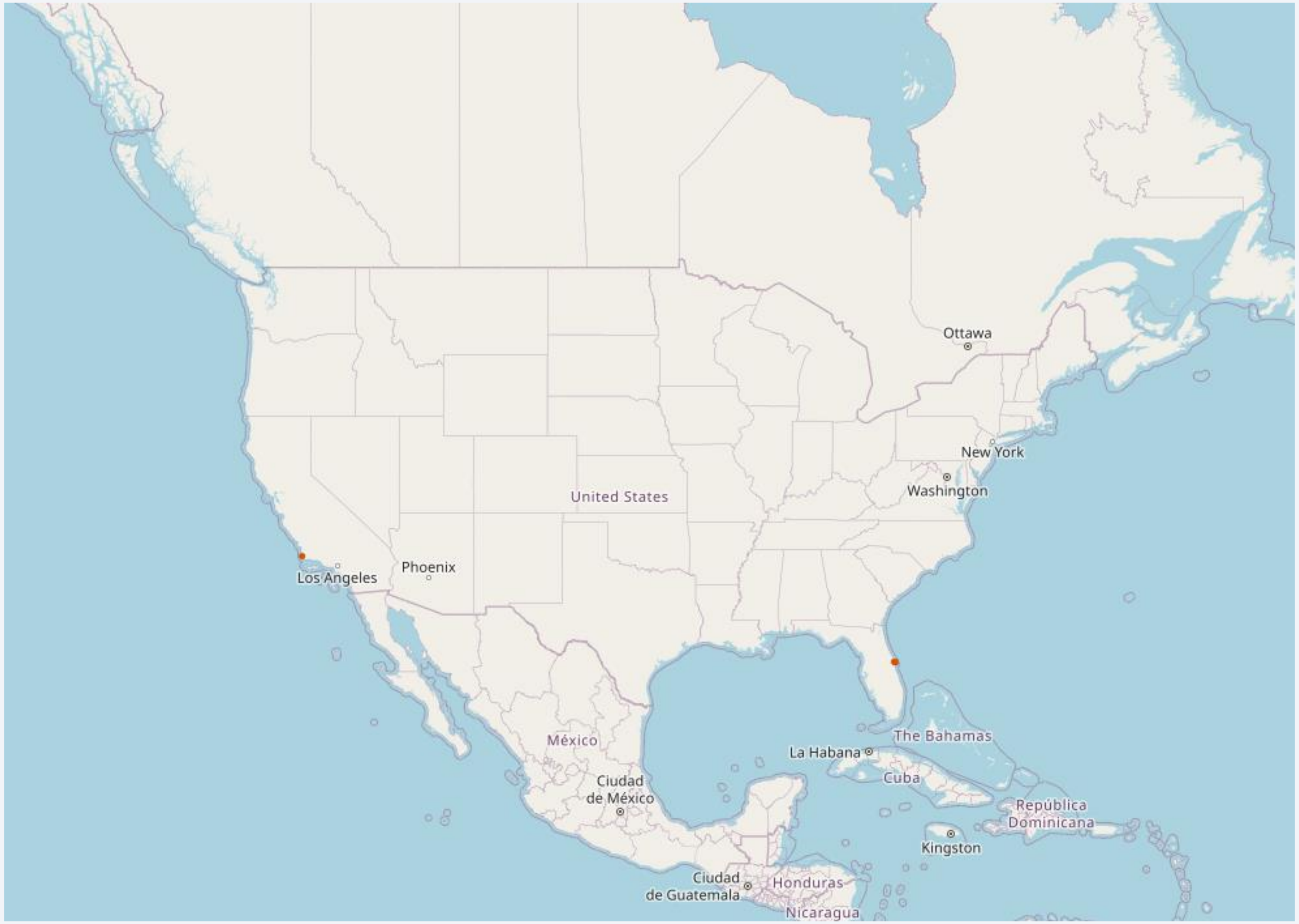
Out[202]:

LANDING__OUTCOME	count(*)
Success (drone ship)	5
Success (ground pad)	3

Section 3

Launch Sites Proximities Analysis

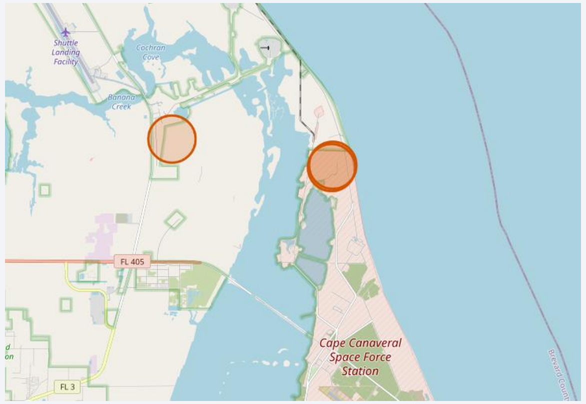
Launch Sites Location

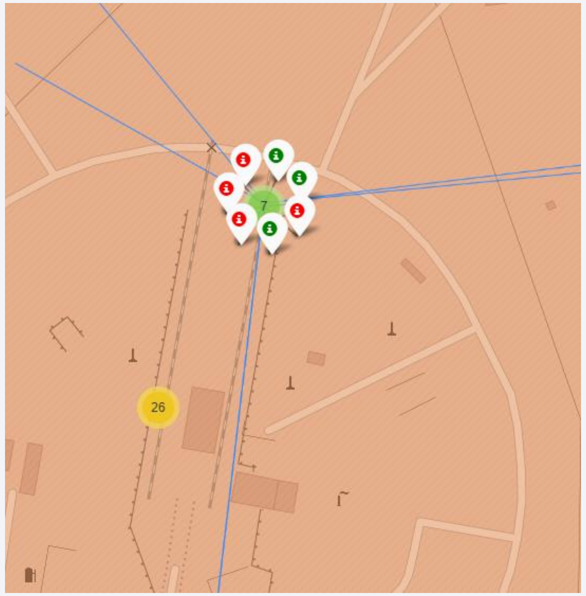


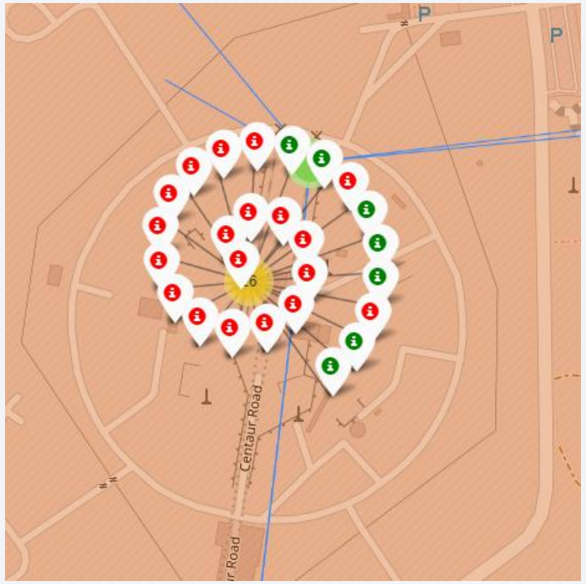
- SpaceX launch sited located in proximity to Pacific and Atltantic coastlines.

Launch Site and their corresponding success rated

- The sites are encircled with circle object. When zoom in, one can find success rates for each launch site by clicking on it.

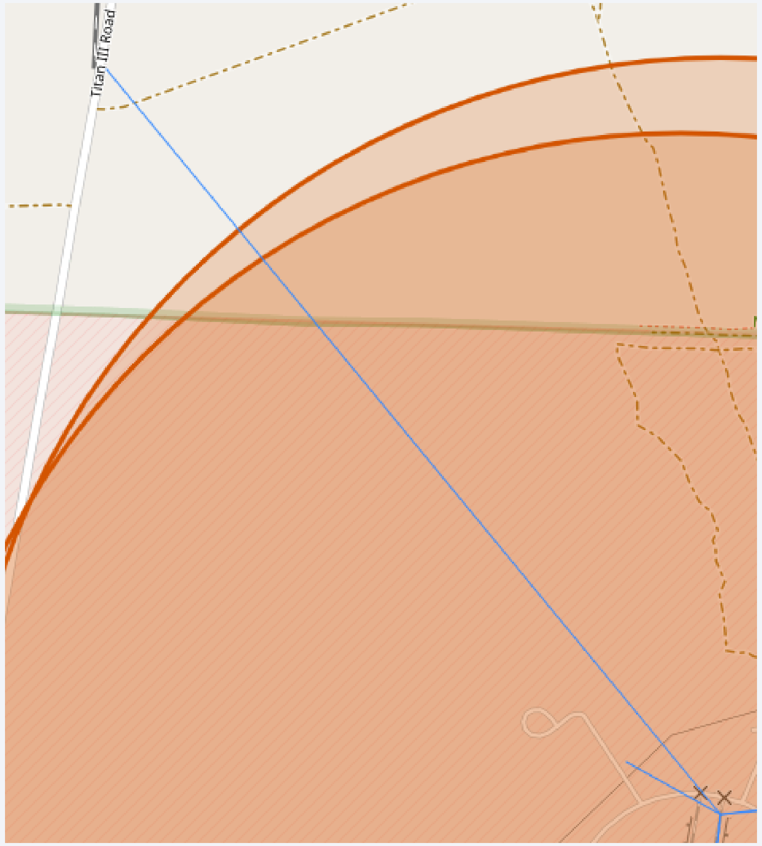






Proximity of launch sites to Landmarks

- One may find that launch sites located in relatively far distance from important Landmarks.

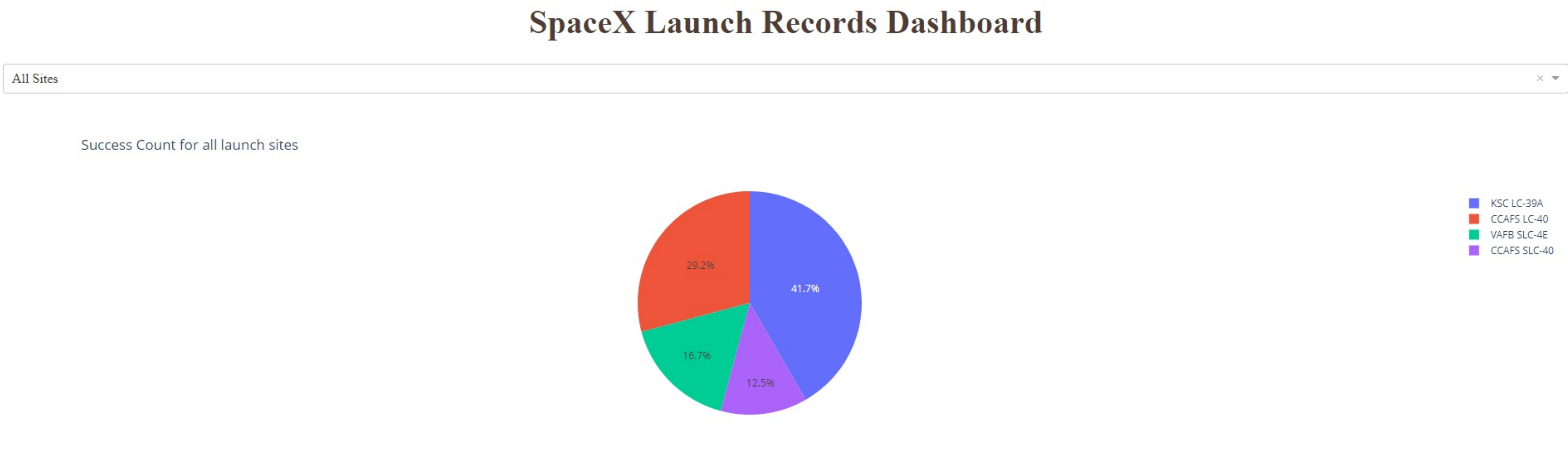


Section 4

Build a Dashboard with Plotly Dash

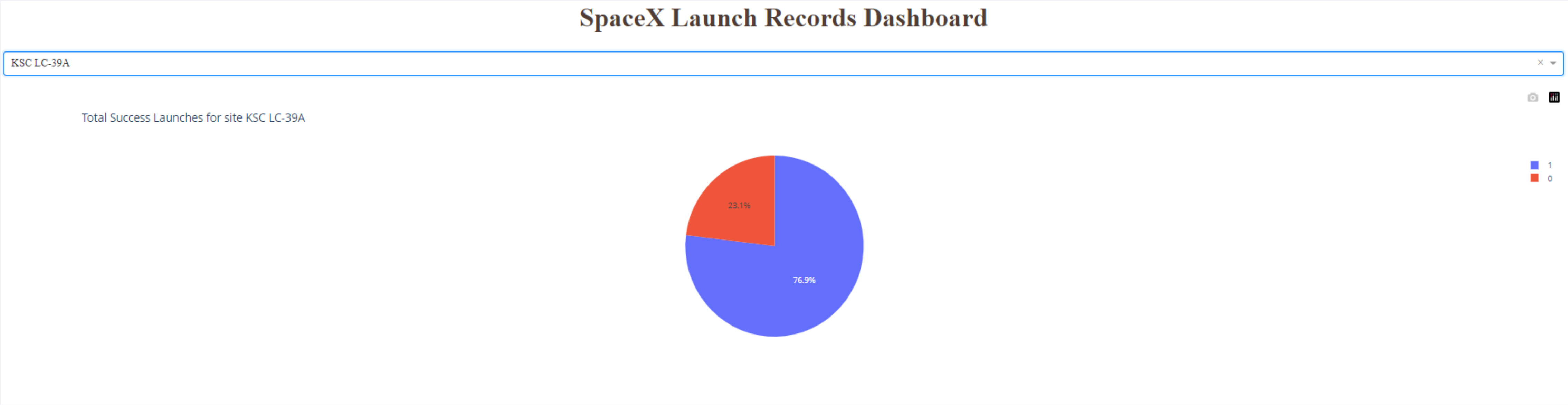
Pie chart shows percentage of successful outcomes for each launch site

- KSC LC-39A has most successful number of launches



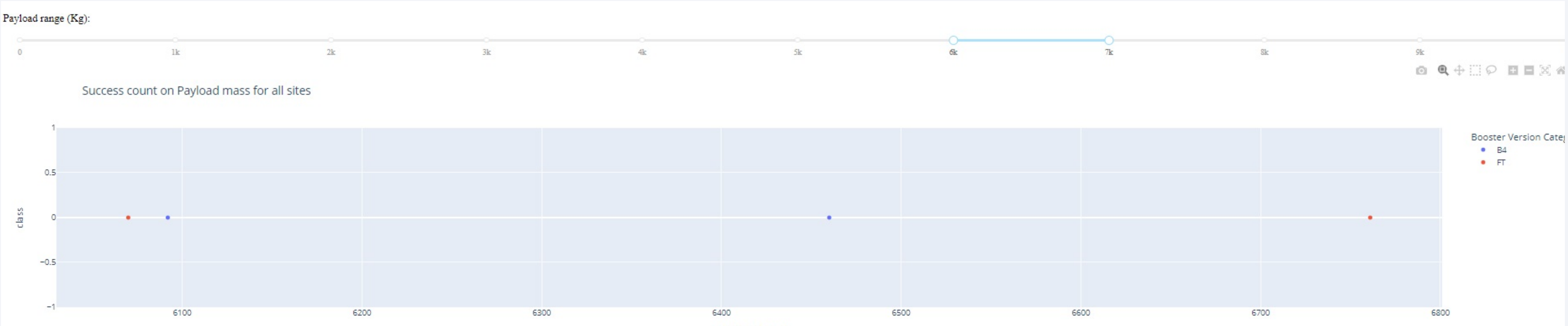
Most successful site

- KSC LC39A is most successful site with 76.9%



Less successive range of PayLoad

One may see that 6-7tons payloads are worst in terms of success outcome

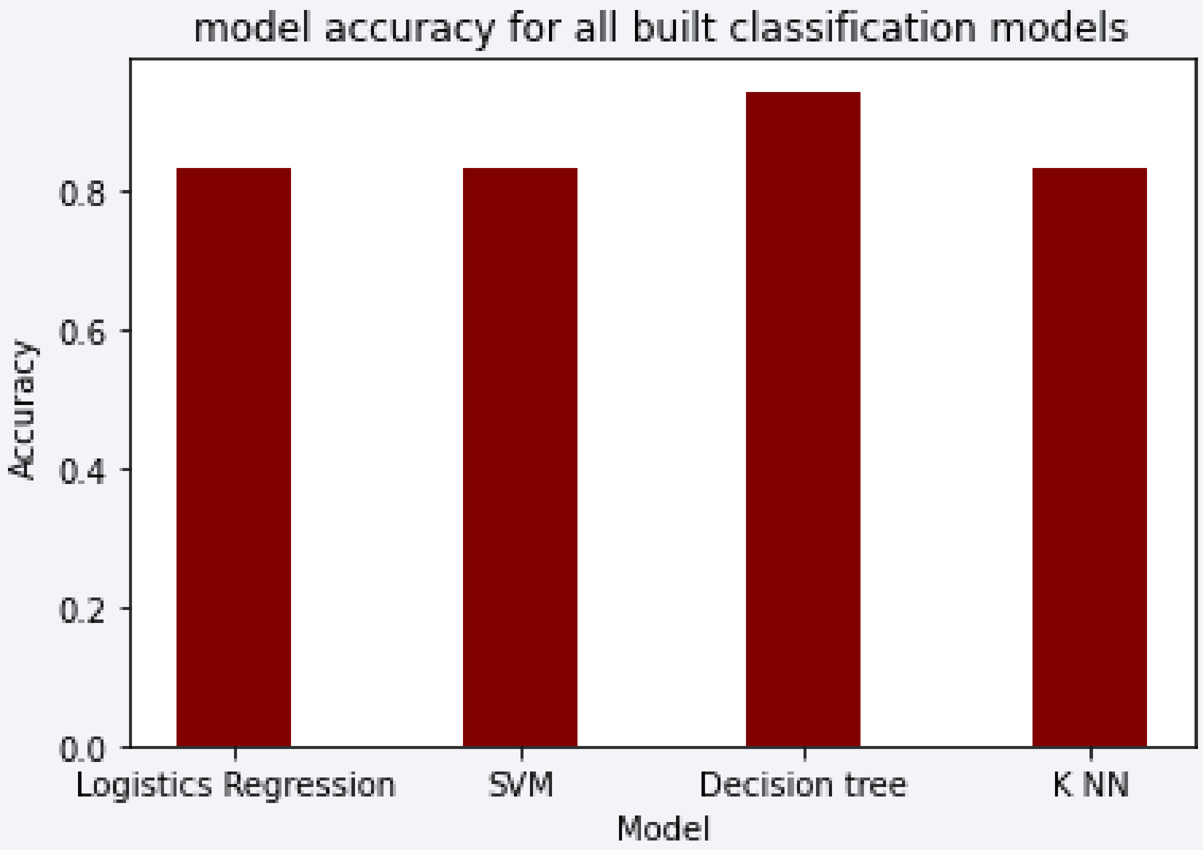


Section 5

Predictive Analysis (Classification)

Classification Accuracy

- Decision tree is of highest accuracy while all other show similar values



Confusion Matrix

- The confusion matrix of the best performing model is of Decision Tree classifier.
- The major problem is with False-Negative (3).
- The best accuracy value is 0.94.

Conclusions

- For CCAFS SLC 40, heavy boosters have high successive rate.
- ES-L1, GEO, HEO and SSO are success rate orbit types.
- Launch Site are located in not very close proximity to important landmarks
- Successful rate improves with the time
- Decision tree classifier gives best results.

Thank you!