IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Fan Huang>
<08/06/2023>

Outline

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

Executive Summary

- Main goal is to make prediction on SpaceX Falcon 9 first stage landing
- Primary steps:
  - Data collection, Data Wrangling, and Data formatting
  - EDA(Exploratory data analysis)
  - Interactive data visualization with Folium and Plotly Dash
  - Machine Learning such as Classification Trees, Logistic Regression and SVM

Introduction

- Space X advertises Falcon 9 rocket launch on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. In this lab, you will create a machine learning pipeline to predict if the first stage will land given the data from the preceding labs.
- Thus, making a prediction model to alleviate such heavy cost is demanding
- The challenging part is how to find valid features and make an accurate prediction

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - SapceX API
  - Web scraping from Wiki
- Perform data wrangling
  - Cleaning and combing Json file by multiple functions to DataFrame
  - Extracting and Parse HTML table by using BeautifulSoup
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Grid searching with crossing validation on a range of different parameters and methods

Data Collection

- SpaceX API
  - The API source can be find from https://api.spacexdata.com/v4/rockets/.
  - Data from the API have many types BoosterVersion and other uncorrelated features
  - Missing data is also presented and need to be imputed or deleted if data is large.

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None

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

```
FlightNumber        0
Date                0
BoosterVersion      0
PayloadMass         5
Orbit               0
LaunchSite          0
Outcome             0
Flights             0
GridFins            0
Reused              0
Legs                0
LandingPad         26
Block               0
ReusedCount         0
Serial              0
Longitude           0
Latitude            0
dtype: int64
```

Using Python API to get URL and fetch and decoding the file to your working directory

Convert retrieved data to Pandas Data Frame

With Pandas API, filtering data only including Falcon 9 and impute data

Data Collection – SpaceX API

- Here is Data is extracted and address is below
- IBM-Applied-Data-Science-Capstone-Project for review: https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/jupyter-labs-spacex-data-collection-api.ipynb)

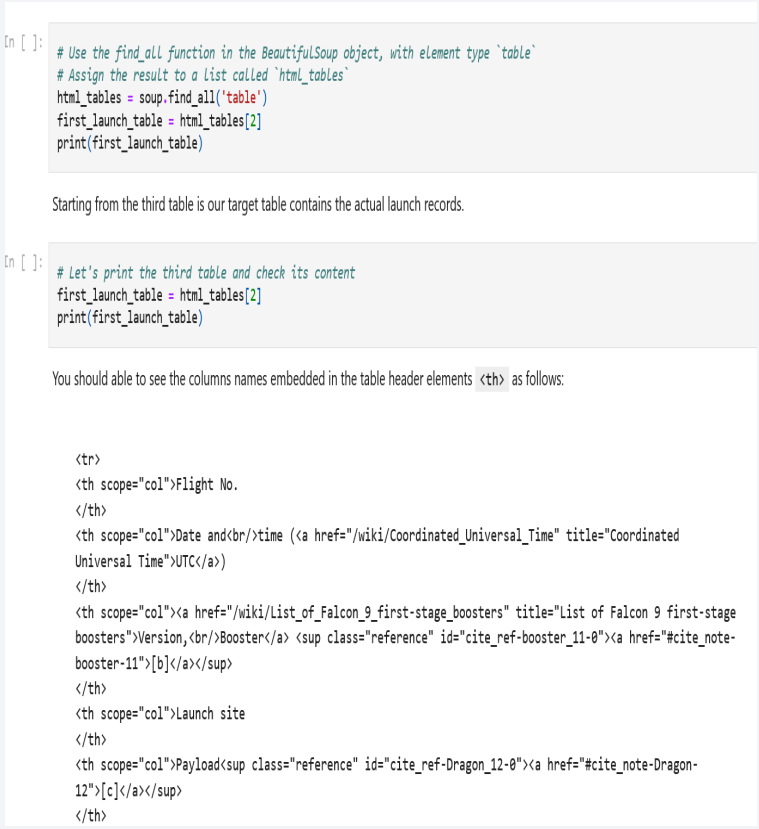
Requesting to access SpaceX API

Decode the downloaded content as Json file

Convert Json to Data Frame and iterate accessing to API through matching IDs until all data are obtained

Data Collection - Scraping

- Data is scraped from following: https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launch&oldid=1027686922. Process is given by flow chart
- GitHub address: https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/jupyter-labs-webscraping.ipynb



```
In [ ]:
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = soup.find_all('table')
first_launch_table = html_tables[2]
print(first_launch_table)
```

Starting from the third table is our target table contains the actual launch records.

```
In [ ]:
# Let's print the third table and check its content
first_launch_table = html_tables[2]
print(first_launch_table)
```

You should able to see the columns names embedded in the table header elements `<th>` as follows:

```
<tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated
Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage
boosters">Version,<br/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#cite_note-
booster-11">[b]</a></sup>
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-
12">[c]</a></sup>
</th>
```

Fetching html data from Wiki URL

Instantiate Beautiful Soup object and parse html data into it

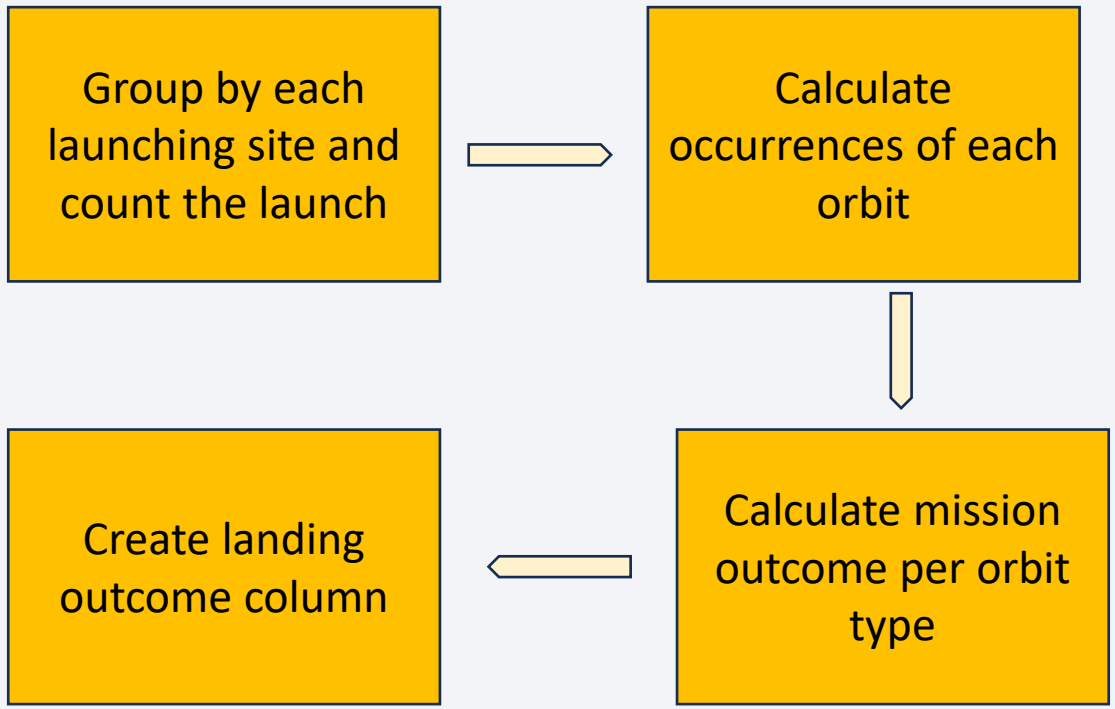
Search for the third table where data contains actual records with soup method find all

Create a dictionary object with column names and fill all necessary data by looping each table rows

Convert files to CSV data sheet

Data Wrangling

- After extracting and combining table content into Data Frame, imputation and One-hot encoding to categorical features are performed
- Adding new 'Class' column 1 as success and 0 as fail
- The final shape of the dataset is (90,83)
- https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb



EDA with Data Visualization

- Plots
  - Scatter plot: such as Flight Number vs Payload mass and Payload vs Orbit type. Scatter plot creating a direct view of relationship between two variables
  - Bar chart: shows the success rate of each orbit
  - Line plot: shows the success rate along with date
- GitHub URL: https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

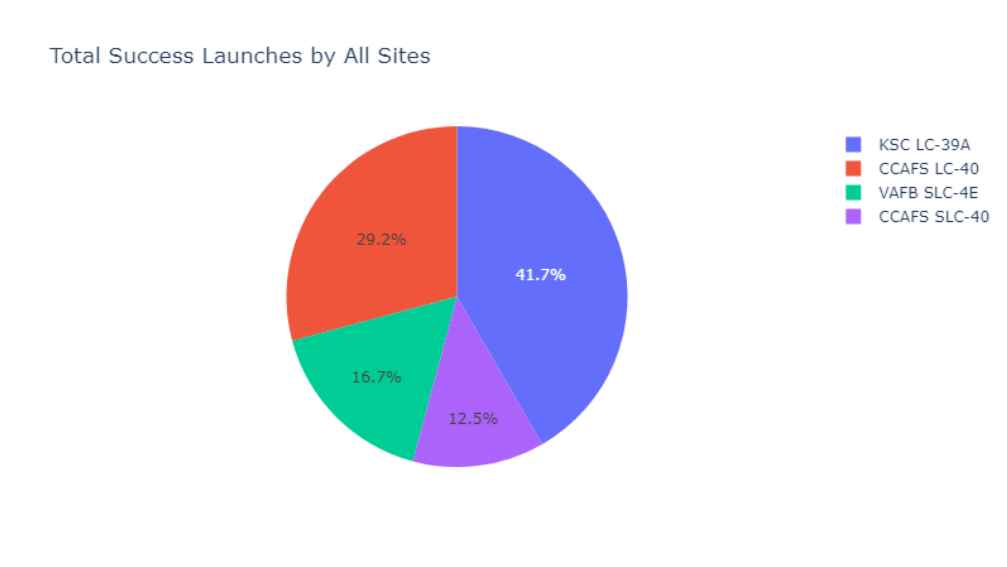
- Following SQL queries are performed:
  - **Display the names of the unique launch sites in the space mission**
  - **Display 5 records where launch sites begin with the string 'CCA'**
  - **Display the total payload mass carried by boosters launched by NASA (CRS)**
  - **Display average payload mass carried by booster version F9 v1.1**
  - **List the date when the first succesful landing outcome in ground pad was acheived.**
  - **List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000**
  - **List the total number of successful and failure mission outcomes**
  - **List the names of the booster_versions which have carried the maximum payload mass. Use a subquery**
  - **List the 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 landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order.**
- GitHub URL: https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Use Folium Markers to show the space X launch sites and landmarks such as railways, highways, coastlines and cities nearby
- To prevent deadly accident and damage may occur during launching
- Use Polylines to connect the distance between launching sites and important objects
- Maker tags such as Red and Green differentiate the fail and success of launching
- GitHub URL: https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/lab_jupyter_launch_site_location.jupyterlite.ipynb

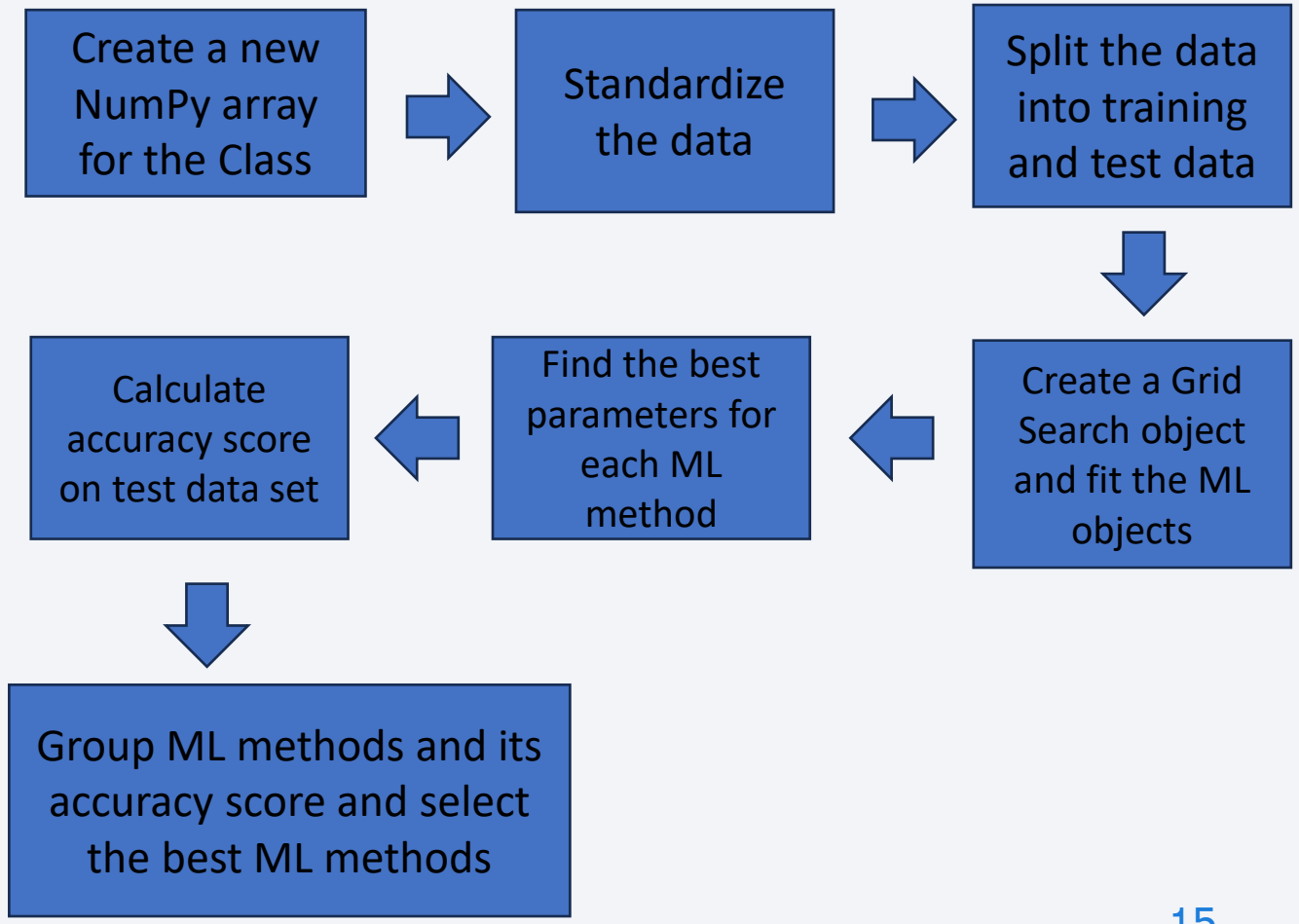
Build a Dashboard with Plotly Dash

- Pie charts to show the percentage success rate of each launching sites
- Scatter charts to visualize correlation between Payload and Success for each launching sites
- GitHub URL: https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/spacex_dash_app.ipynb



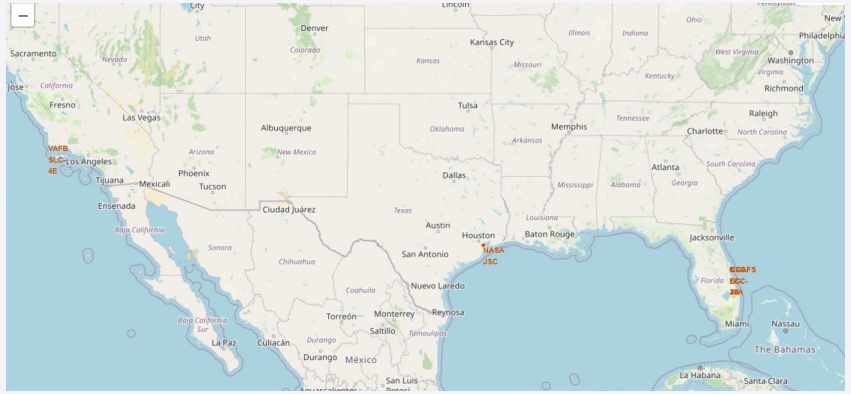
Predictive Analysis (Classification)

- Scikit-learn is prominent Machine Learning API with wide range of predictive methods, tunning parameters such as Grid Searching and efficient implementation as pipeline
- GitHub URL: https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project/blob/main/Space%20X%20Machine%20Learning%20Prediction.ipynb



Results

- Exploratory data analysis results:
  - We see that as the flight number increases, the first stage is more likely to land successfully. The payload mass is also important; it seems the more massive the payload, the less likely the first stage will return
  - We see that different launch sites have different success rates. CCAFS LC-40, has a success rate of 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%
- Interactive analytics demo in screenshots



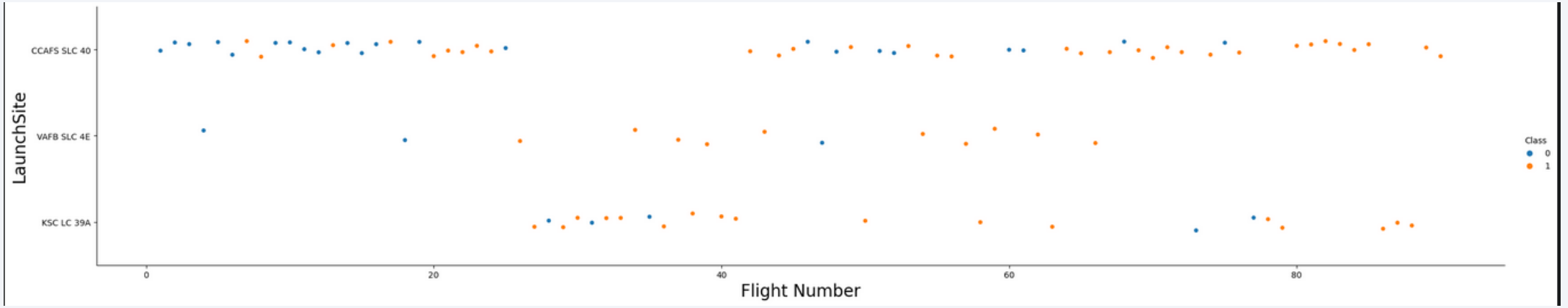
- Predictive analysis results
  - 83.33% is the best test score on ML methods: SVM, Classification Trees, Logistic Regression and KNN

Section 2

Insights drawn from EDA

Flight Number vs. Launch Site

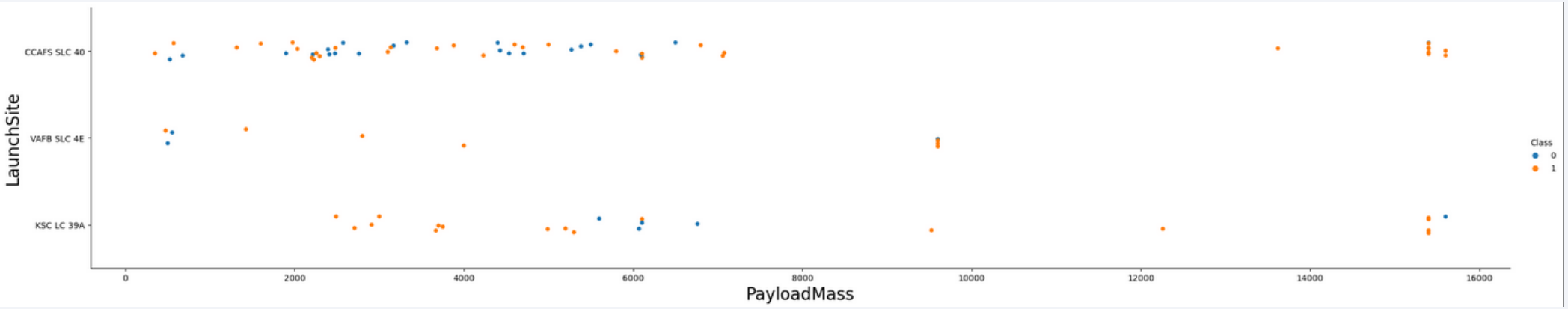
- Show a scatter plot of Flight Number vs. Launch Site



- It appears that there are more successful launch as the flight numbers increased. launch site CCAFS SLC 40 has the most number of launch.

Payload vs. Launch Site

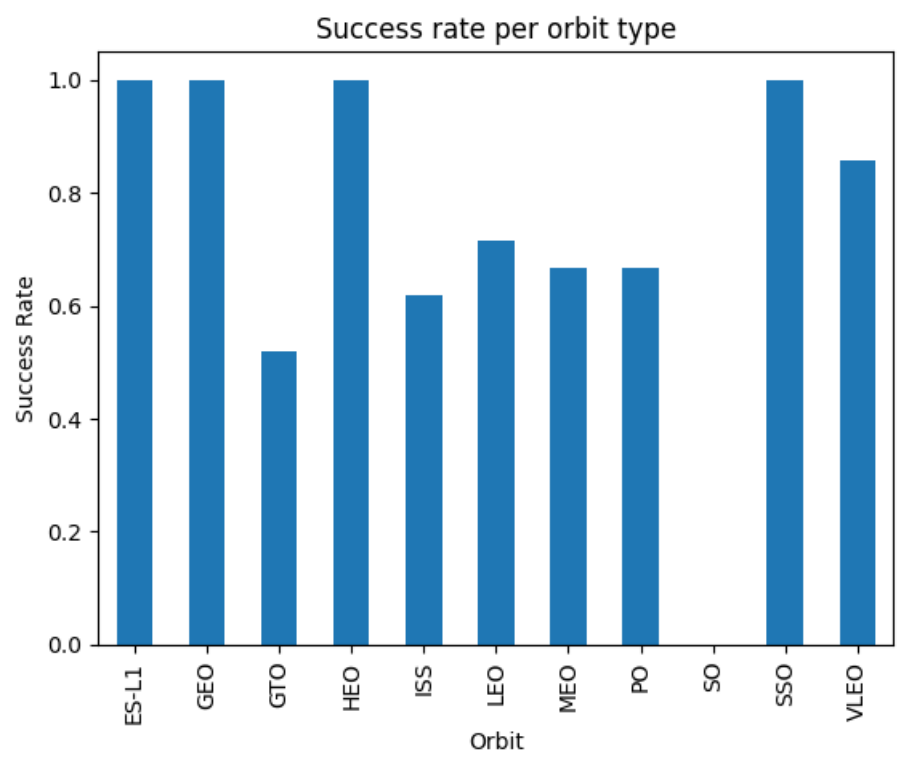
- Show a scatter plot of Payload vs. Launch Site



- Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launch site there are no rockets launch for heavy payload mass(greater than 10000).

Success Rate vs. Orbit Type

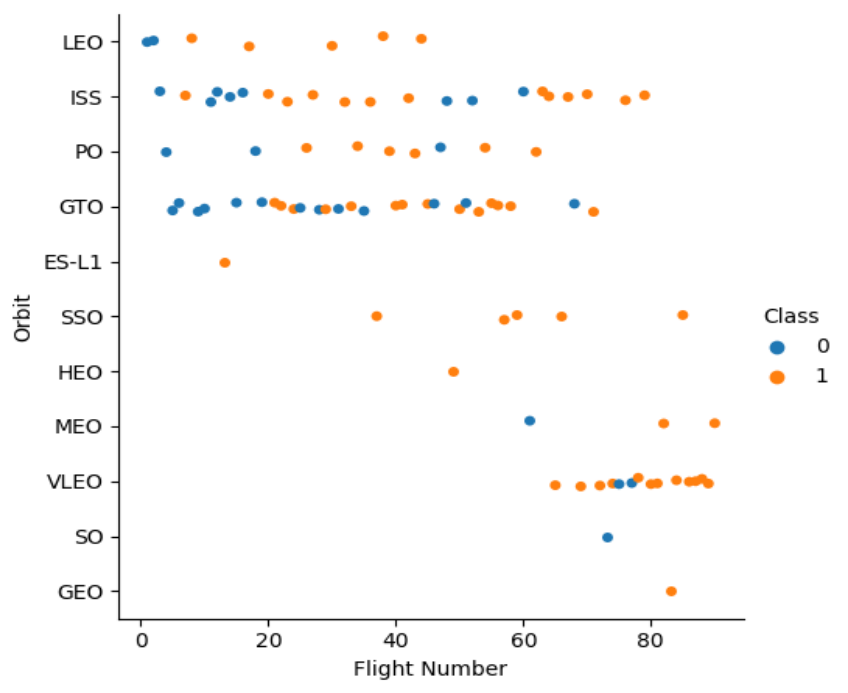
- Show a bar chart for the success rate of each orbit type



- The highest success rate of each orbit are ES-L1, GEO, HEO, SSO

Flight Number vs. Orbit Type

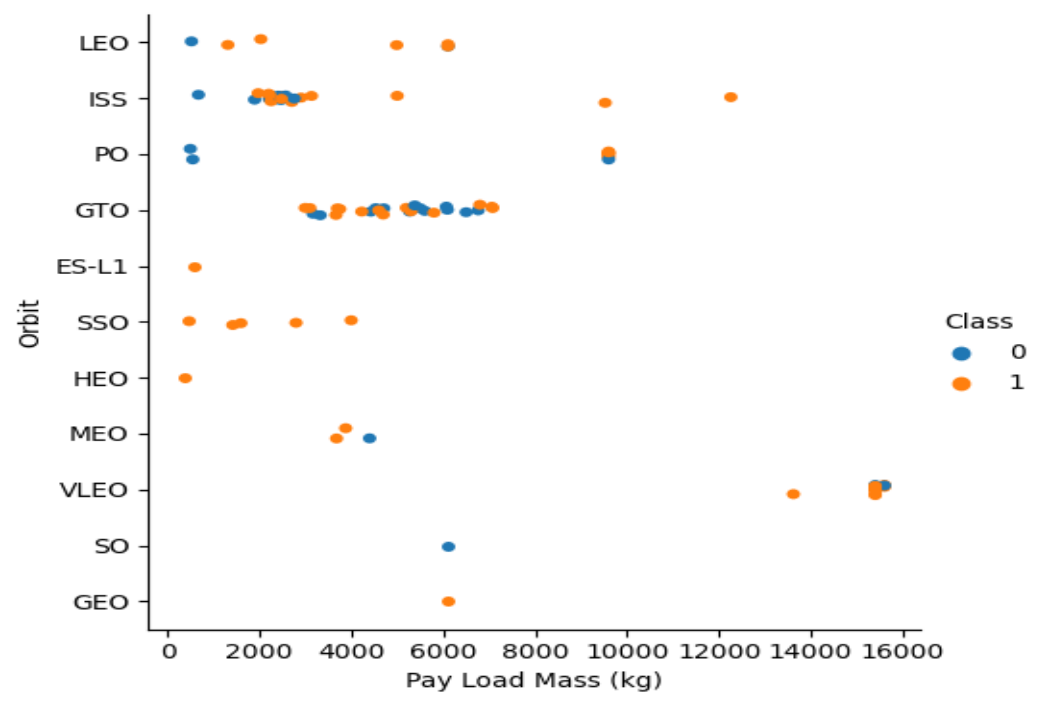
- Show a scatter point of Flight number vs. Orbit type



- LEO orbit has success launching increased as the Flight Number increases but GTO has no relationship with Flight Number

Payload vs. Orbit Type

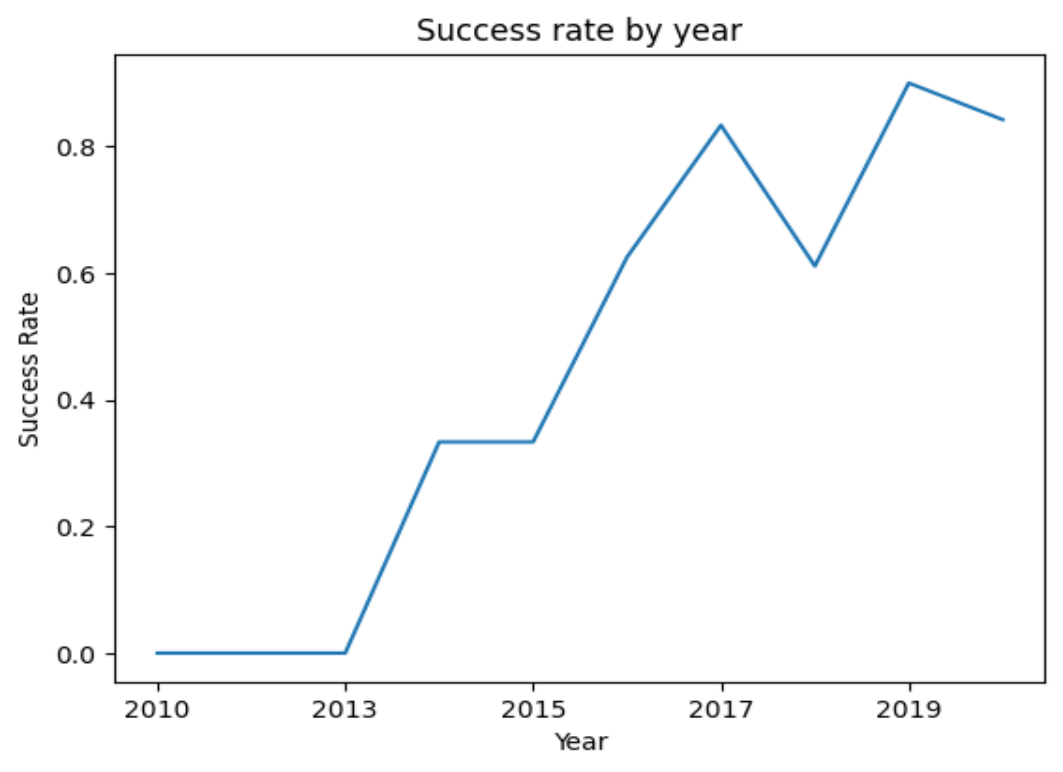
- Show a scatter point of payload vs. orbit type



- Polar, LEO and ISS has positive relationship with Pay Load Mass
- GTO doesn't have clear relationship with Pay Load Mass
- SSO has successful launching between 0 and 4000kg

Launch Success Yearly Trend

- Show a line chart of yearly average success rate



- you can see that the success rate since 2013 kept increasing till 2020

All Launch Site Names

- Find the names of the unique launch sites
  - CCAFS LC-40
  - CCAFS SLC-40
  - KSC LC-39A
  - VAFB SLC-4E

```
In [12]: %sql select distinct Launch_Site from SPACEXTABLE ;
 * sqlite:///my_data1.db
Done.
Out[12]:
```

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

- Showing distinct names querying from the database

Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

```
In [45]: %sql select *  from SPACEXTABLE WHERE Launch_Site like 'CCA%' limit 5
 * sqlite:///my_data1.db
Done.
```

Out[45]:

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

- Fetching and querying name contain with ‘CCA’ from the Database

Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
[35]: #%sql select PAYLOAD_MASS__KG_ from SPACEXTABLE where Launch_Site like '%NASA%'
      %sql select sum(PAYLOAD_MASS__KG_) from SPACEXTABLE where Customer like '%NASA (CRS)%'

       * sqlite:///my_data1.db
      Done.
[35]: sum(PAYLOAD_MASS__KG_)
      ----------------------
                       48213
```

- Total payload from customer NASA(CRS) is 48213

Average Payload Mass by F9 v1.1

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

```
In [34]: %sql select avg(PAYLOAD_MASS__KG_) from SPACEXTABLE where booster_version like '%F9 v1.1%'
 * sqlite:///my_data1.db
Done.
Out[34]: avg(PAYLOAD_MASS__KG_)
         2534.6666666666665
```

- The average payload mass carried by booster F9 v1.1 is around 2534

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
In [54]: %sql select "Date",Landing_Outcome  from SPACEXTABLE where Landing_Outcome like '%Success (ground pad)%' order by Date limit
 * sqlite:///my_data1.db
Done.
Out[54]:
```

Date	Landing_Outcome
2015-12-22	Success (ground pad)

- The dates of the first successful landing is 2015 12 22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
In [55]: %sql select Booster_Version, Landing_Outcome, PAYLOAD_MASS__KG_ from SPACEXTABLE \
         where Landing_Outcome like '%Success (drone ship)%' and PAYLOAD_MASS__KG_ between 4000 and 6000;
 * sqlite:///my_data1.db
Done.
```

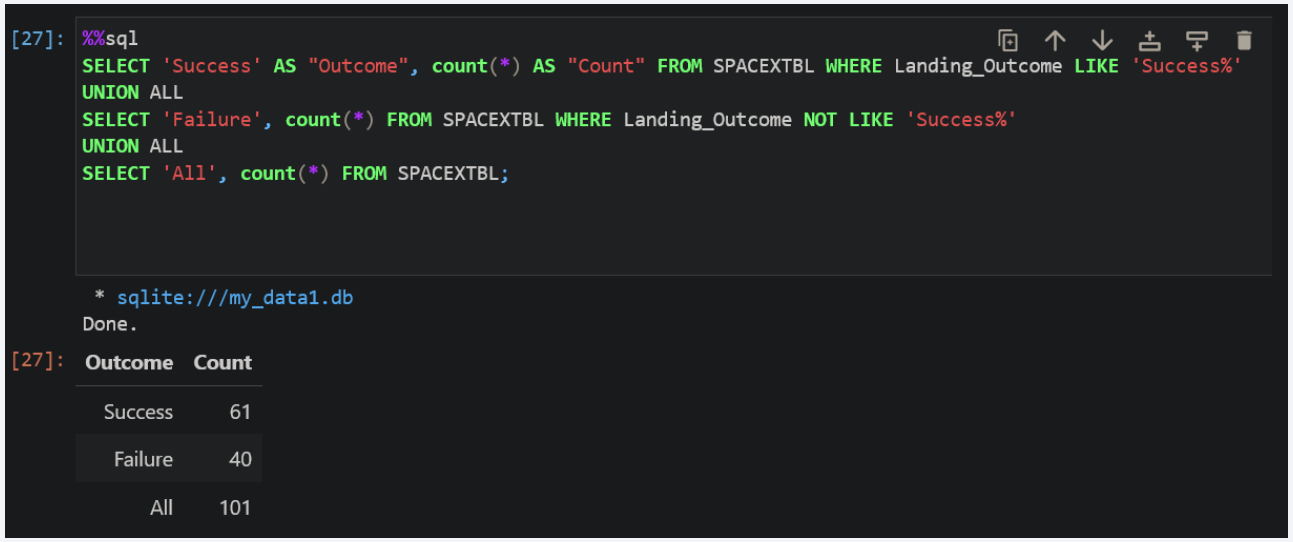
Out[55]:

Booster_Version	Landing_Outcome	PAYLOAD_MASS__KG_
F9 FT B1022	Success (drone ship)	4696
F9 FT B1026	Success (drone ship)	4600
F9 FT B1021.2	Success (drone ship)	5300
F9 FT B1031.2	Success (drone ship)	5200

- The booster versions that have successfully landed on drone ship between 4000 and 6000 are F9 FT B1022, F9 FT B1026, F9 FT B1021.2, F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes



```
[27]: %%sql
SELECT 'Success' AS "Outcome", count(*) AS "Count" FROM SPACEXTBL WHERE Landing_Outcome LIKE 'Success%'
UNION ALL
SELECT 'Failure', count(*) FROM SPACEXTBL WHERE Landing_Outcome NOT LIKE 'Success%'
UNION ALL
SELECT 'All', count(*) FROM SPACEXTBL;

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

[27]:

Outcome	Count
Success	61
Failure	40
All	101

- Total 101 missions and successful missions count as 61 and failed missions count as 40

Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass

```
[67]: %sql select booster_version, PAYLOAD_MASS__KG_ from SPACEXTABLE \
      where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from SPACEXTABLE)
       * sqlite:///my_data1.db
      Done.
```

[67]:

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

- The list of booster versions have carried the maximum payload 15600kg is showing above

2015 Launch Records

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

```
%sql select substr(Date, 6, 2) As "Month", substr(Date,1,4) As year , Landing_Outcome, Booster_Version, Launch_Site\
from SPACEXTABLE\
where substr(Date,1,4)='2015' and Landing_Outcome = 'Failure (drone ship)'
```

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

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

- The failure of landing booster in 2015 are F9 v.1.1 B1012 and F9 v1.1 B1015 from same launch site CCAFS LC-40

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

- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

```
[94]: %sql select "Date", Landing_Outcome from SPACEXTABLE \
      where Landing_Outcome in ('Failure (drone ship)','Success (ground pad)') and "Date" between '2010-06-04' and '2017-03-20' \
      order by "Date" desc
```

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

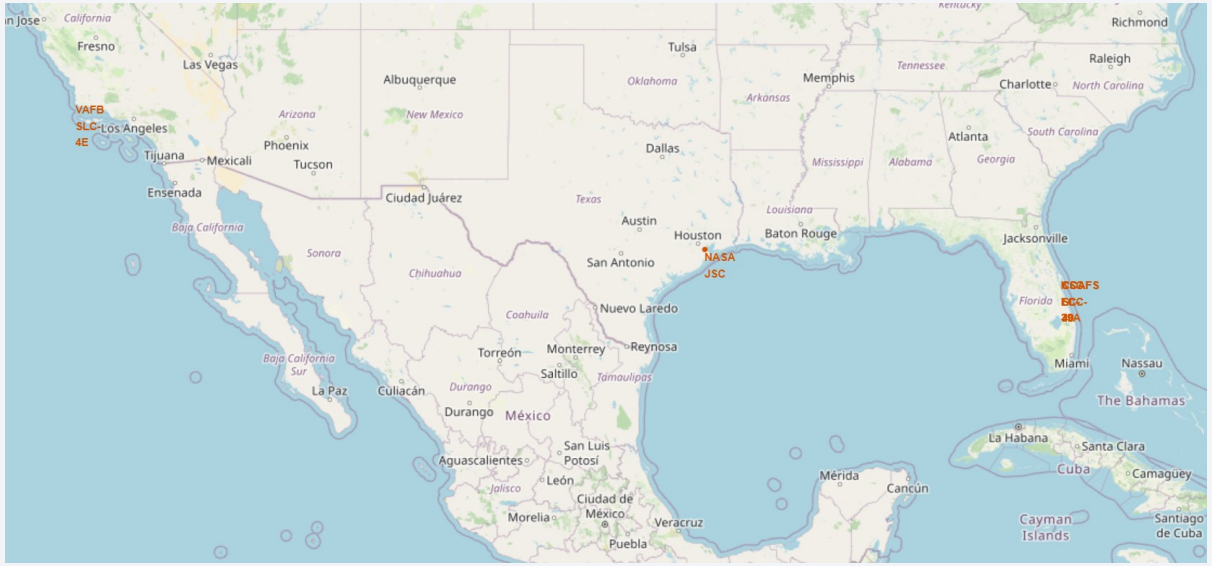
[94]:

Date	Landing_Outcome
2017-03-06	Success (ground pad)
2017-02-19	Success (ground pad)
2017-01-05	Success (ground pad)
2016-07-18	Success (ground pad)
2016-06-15	Failure (drone ship)
2016-04-03	Failure (drone ship)
2016-01-17	Failure (drone ship)
2015-12-22	Success (ground pad)
2015-10-01	Failure (drone ship)
2015-04-14	Failure (drone ship)

- The number of successful landings are increasing since 2015

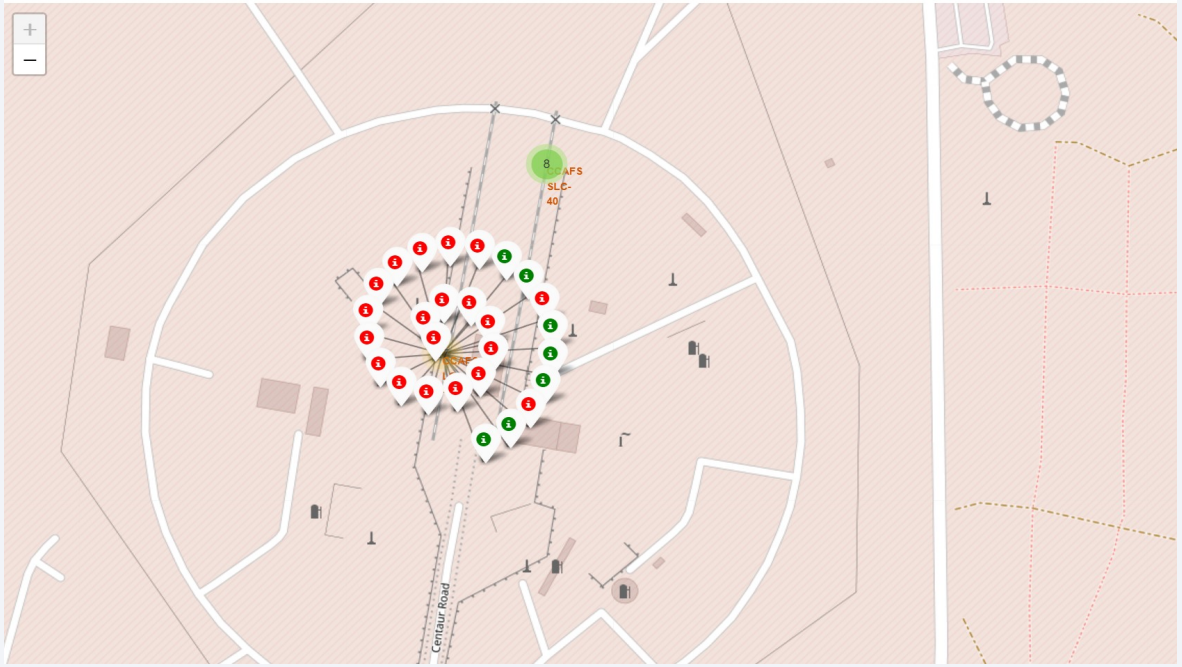

Section 3
Launch Sites
Proximities Analysis

Launch Site Locations



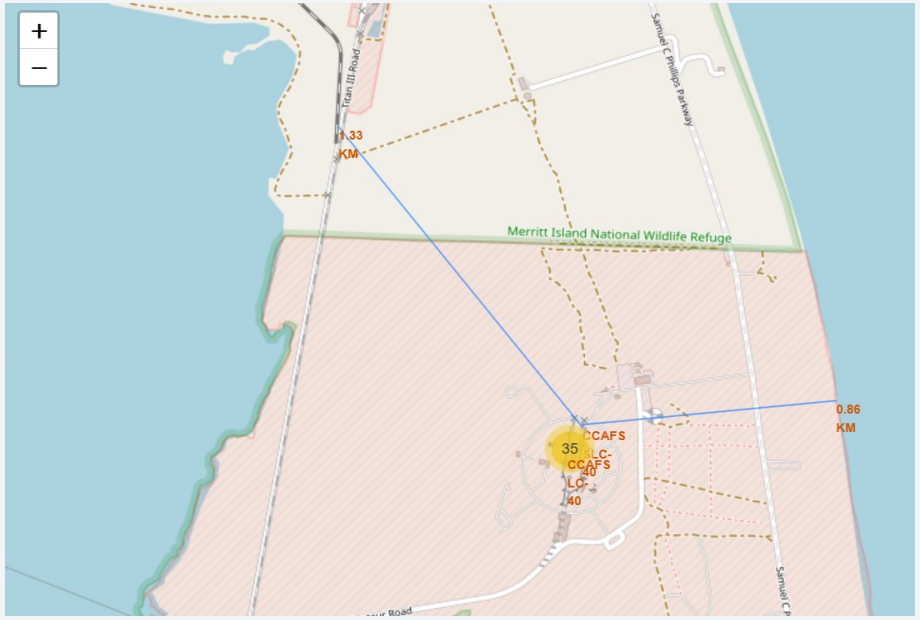
- All launch sites are very close to coastline for massive water resources and transportation near the launching sites.

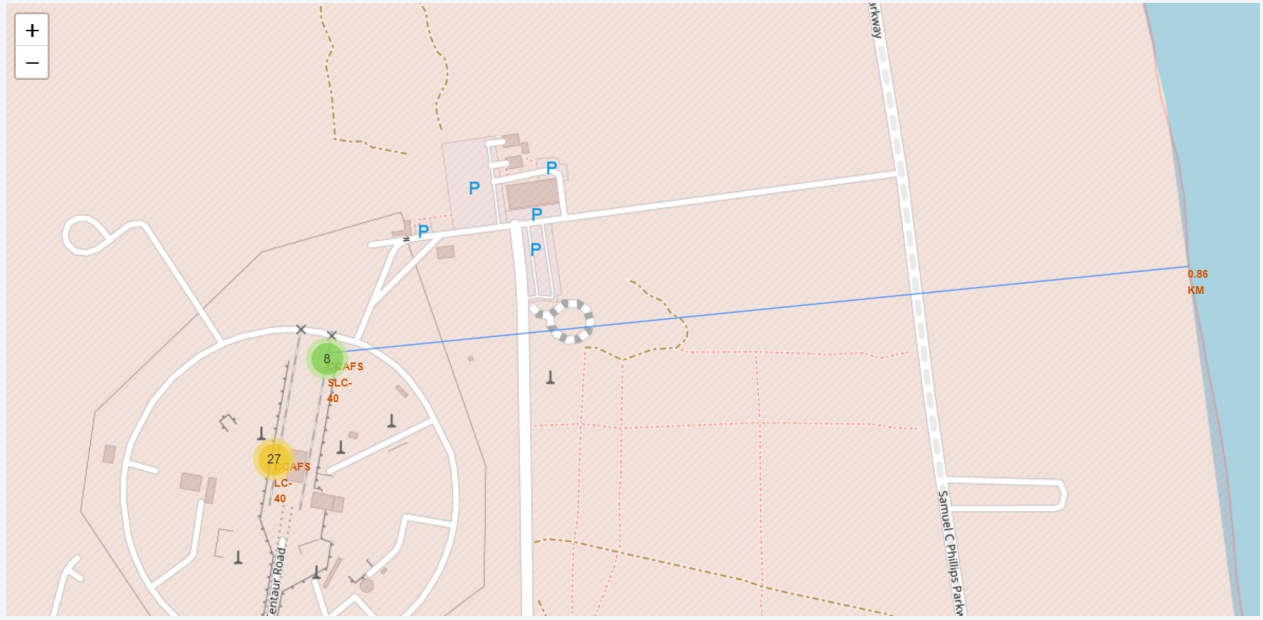
Success Rate of Rocket launch



- The graph shows the RED tag as failed launch and Green tag as successful launch from launching site CCAFS LC-40

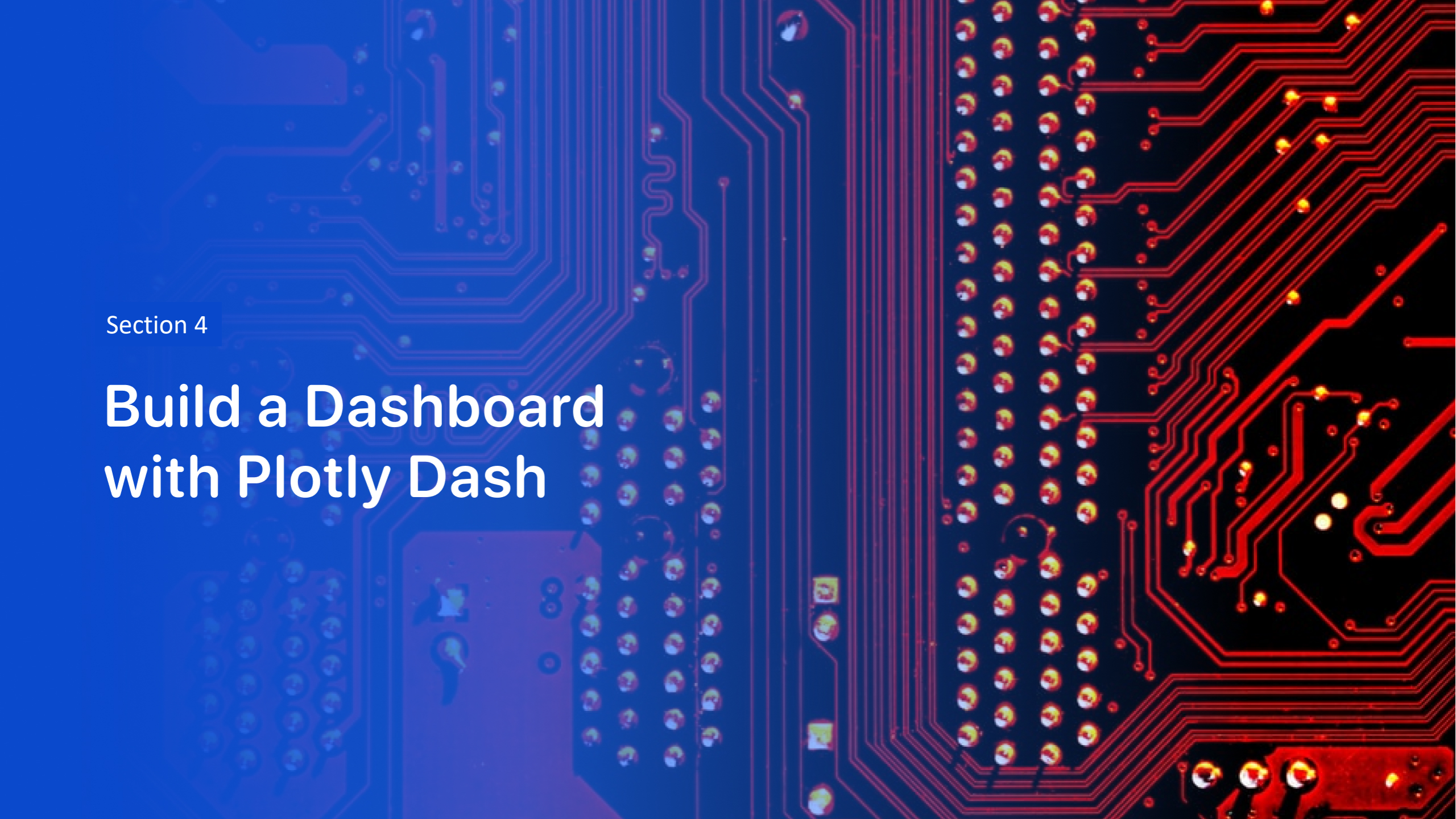
Distance to Launch Sites



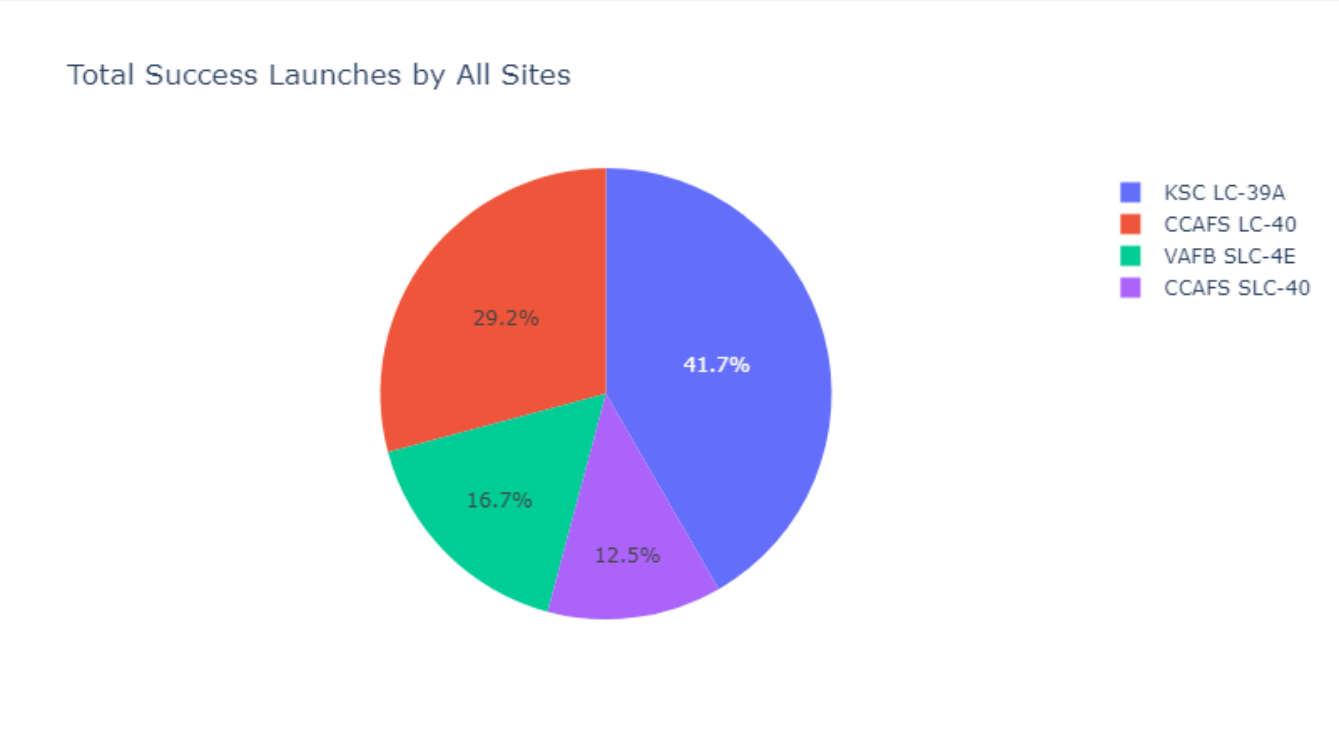


- launch site to its proximities such as railway, coastline with distance calculated and displayed 1.33km from railway and 0.68km from coastline

- The reason Launching site near railway is to take advantage of transportation as well as water resources

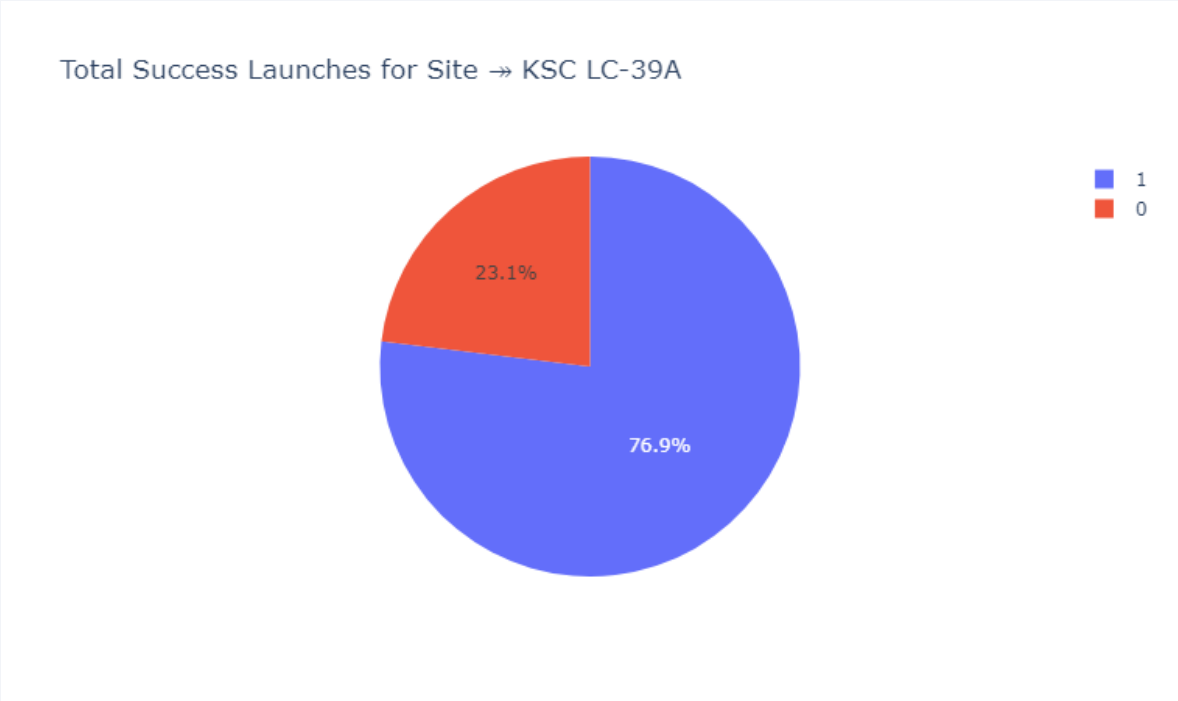

Section 4
Build a Dashboard with Plotly Dash

Successful launch by Site



- From the pie chart, KSC LC-39A has the most successful launch

KSC LC-39A successful rate



- KSC LC-39A has 76.9% successful rate, which is the highest among other launch sites

Payload Mass vs Launch Success for All Sites

- It shows payload mass between 2000kg and 4000kg has the highest success rate

Section 5

Predictive Analysis (Classification)

Classification Accuracy

Out[28]:

	ML Method	Accuracy Score (%)
0	Support Vector Machine	83.333333
1	Logistic Regression	83.333333
2	K Nearest Neighbour	83.333333
3	Decision Tree	83.333333

In [20]:

```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier()

tree_cv = GridSearchCV(estimator=tree, cv=10, param_grid=parameters).fit(X_train, Y_train)
```

In [21]:

```
print("tuned hpyerparameters :(best parameters) ",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)
```

```
tuned hpyerparameters :(best parameters)  {'criterion': 'entropy', 'max_depth': 8, 'max_features': 'sqrt', 'min_samples_leaf
': 2, 'min_samples_split': 10, 'splitter': 'best'}
accuracy : 0.8892857142857142
```

- Since all methods have 83.33% accuracy on test data, it is really hard to choose the model solely on accuracy score
- If we look at the training score, Decision Tree Classifier has 88.9% accuracy, it is reasonable to choose it as the desirable model for future prediction.

Confusion Matrix

- From the confusion matrix, we can see false positive has 3 and True negative has 0 and correct to predict successful landing is 12 and correct to predict failed landing is 3.
- Based on the confusion matrix, Decision Tree Classifier is a solid model

Conclusions

- From the launch site locations, we know that they are all near coastlines and transportation routes are also close. It gives more utilities and advantages
- KSC LC-39A has the highest launch success rate among all other sites
- Since 2015, the successful rate of rocket landing is increased result in raised flight number
- Payload mass from 2000kg and 4000kg from rocket has more successful launch than other range of Payload mass
- All Machine Learning method logistic regression, SVM, Tree classifiers and KNN are doing very well based on both confusion matrix and accuracy score.

Appendix

- https://github.com/FanHuang321/-IBM-Applied-Data-Science-Capstone-Project

Thank you!