IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Name>
<Date>

Outline

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

Executive Summary

- Summary of methodologies
  - SpaceX Data Collection using SpaceX API
  - SpaceX Data Collection with Web Scraping
  - SpaceX Data Wrangling -
  - SpaceX Exploratory Data Analysis using SQL
  - Space-X EDA DataViz Using Python Pandas and Matplotlib
  - Space-X Launch Sites Analysis with Folium-Interactive Visual Analytics and Ploty Dash
  - SpaceX Machine Learning Landing Prediction
- Summary of all results
  - EDA results
  - Interactive Visual Analytics and Dashboards
  - Predictive Analysis(Classification)

Introduction

In this capstone, we will predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket launches 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 SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

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

Description of how SpaceX Falcon9 data was collected.

- Data was first collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API. This was done by first defining a series helper functions that would help in the use of the API to extract information using identification numbers in the launch data and then requesting rocket launch data from the SpaceX API url.

- Finally to make the requested JSON results more consistent, the SpaceX launch data was requested and parsed using the GET request and then decoded the response content as a Json result which was then converted into a Pandas data frame.

- Also performed web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled List of Falcon 9 and Falcon Heavy launches of the launch records are stored in a HTML. Using BeautifulSoup and request Libraries, I extract the Falcon 9 launch HTML table records from the Wikipedia page, Parsed the table and converted it into a Pandas data frame

Data Collection

To make the requested JSON results more consistent, we will use the following static response object for this project:

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

We should see that the request was successfull with the 200 status response code

```
response.status_code
```

```
200
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

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

Using the dataframe `data` print the first 5 rows

```
# Get the head of the dataframe
data.head()
```

Data collected using SpaceX API (a RESTful API) by making a get request to the SpaceX API

Here is the the GitHub URL of the completed SpaceX API calls notebook https://github.com/Aabdouni/testrepoIBM/blob/Applied-Data-Science-Capstone-Project/1%20-%20Spacex-data-collection-api.ipynb

Data Collection - Scraping

Performing web scraping to collect Falcon 9 historical launch records from a Wikipedia page titled `List of Falcon 9 and Falcon Heavy launches`

https://en.wikipedia.org/wiki/List_of_Falcon_9_and_Falcon_Heavy_launches

Here is the the GitHub URL of the completed web scraping notebook https://github.com/Aabdouni/testrepoIBM/blob/Applied-Data-Science-Capstone-Project/2%20-%20Spacex-webscraping.ipynb

To keep the lab tasks consistent, you will be asked to scrape the data from a snapshot of the `List of Falcon 9 and Falcon Heavy launches` Wikipage updated on `9th June 2021`

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

Next, request the HTML page from the above URL and get a `response` object

TASK 1: Request the Falcon9 Launch Wiki page from its URL

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

Create a `BeautifulSoup` object from the HTML `response`

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(response.content, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

```
# Use soup.title attribute
soup.title
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

TASK 2: Extract all column/variable names from the HTML table header

Next, we want to collect all relevant column names from the HTML table header

Let's try to find all tables on the wiki page first. If you need to refresh your memory about `BeautifulSoup`, please check the external reference link towards the end of this lab

```
# 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')
```

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

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

Data Wrangling

Creating a Pandas DF from the collected data.Cleaning and selecting the Falcon 9 launches record, then dealt with the missing data values in the LandingPad and PayloadMass columns.

Replacing missing data values of the PayloadMass column with the mean value of column.

Performing Exploratory Data Analysis (EDA) to find some patterns in the data and determine what would be the label for training supervised models

Here is the the GitHub URL of the completed wrangling notebook https://github.com/Aabdouni/testrepoIBM/blob/Applied-Data-Science-Capstone-Project/3%20-%20Spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

TASK 4: Create a landing outcome label from Outcome column

Using the `Outcome`, create a list where the element is zero if the corresponding row in `Outcome` is in the set `bad_outcome`; otherwise, it's one. Then assign it to the variable `landing_class`:

```
# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

This variable will represent the classification variable that represents the outcome of each launch. If the value is zero, the first stage did not land successfully; one means the first stage landed Successfully

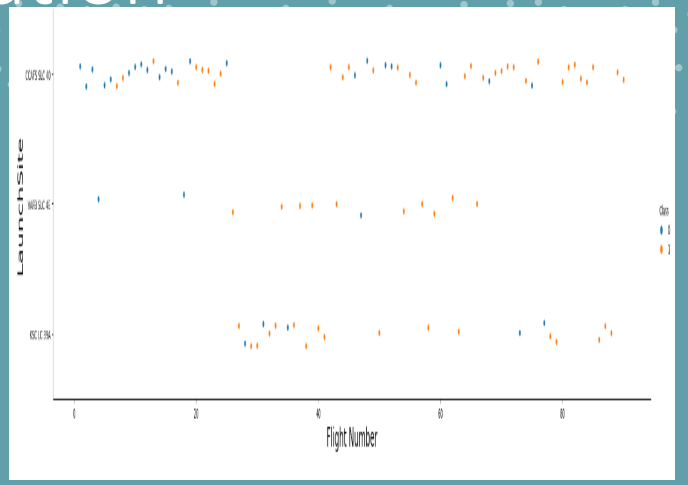
```
df['Class']=landing_class
df[['Class']].head(8)
```

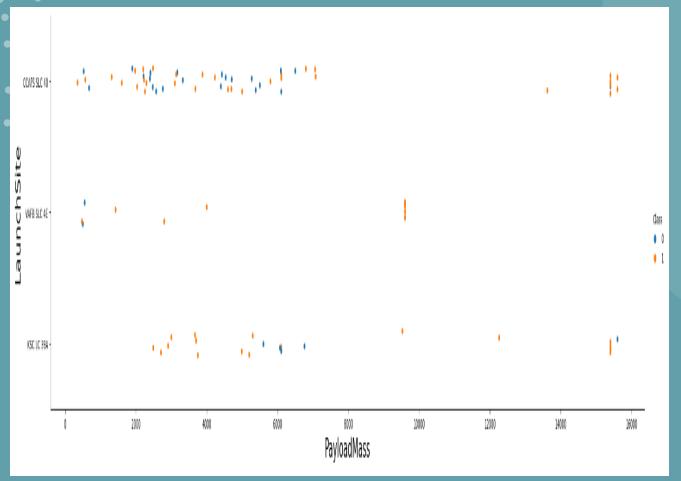
	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

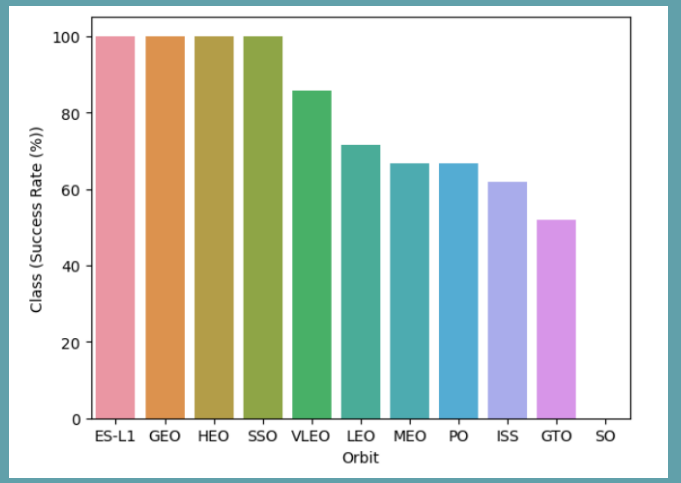
EDA with Data Visualization

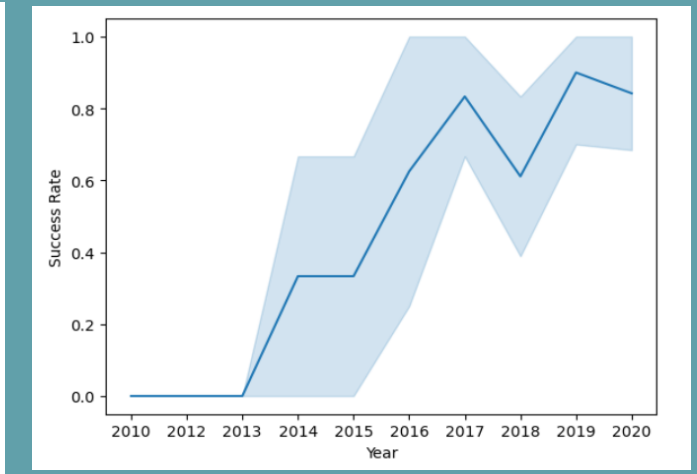
Used different charts to understand the relation between multiple columns : scatter plot, Bar chart and line plot

Here is the the GitHub URL of the completed EDA with data visualization notebook https://github.com/Aabdouni/testrepoIBM/blob/Applied-Data-Science-Capstone-Project/5%20-%20Spacex-labs-eda-dataviz.ipynb.jupyterlite.ipynb









EDA with SQL

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

```
%sql SELECT Distinct Launch_Site FROM SPACEXTBL
```

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

```
%sql SELECT sum(PAYLOAD_MASS__KG_) as "Total payload mass" FROM SPACEXTBL WHERE Customer like "%NASA (CRS)%"
```

List the date when the first succesful landing outcome in ground pad was acheived.

```
%sql SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome like 'Success (ground pad)'
```

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

```
%sql SELECT * FROM SPACEXTBL WHERE Launch_Site like "CCA%" LIMIT 5
```

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Average payload mass" FROM SPACEXTBL WHERE Booster_Version like 'F9 v1.1'
```

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

```
%sql SELECT  Booster_Version as " Booster Version" FROM SPACEXTBL WHERE Landing_Outcome like "Success (drone ship)" \
AND (PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000 )
```

List the total number of successful and failure mission outcomes

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

Here is the the GitHub URL of the completed EDA with data visualization notebook https://github.com/Aabdouni/testrepoIBM/blob/Applied-Data-Science-Capstone-Project/4%20-Spacex-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

Created folium map to marked all the launch sites, and created map objects such as markers, circles, lines to mark the success or failure of launches for each launch site.

- Created a launch set outcomes.

Here is the the GitHub URL of the completed interactive map with Folium map https://github.com/Aabdouni/testrepoIBM/blob/Applied-Data-Science-Capstone-Project/6%20-%20Spacex%20-%20Launch_site_location.jupyterlite.ipynb

Predictive Analysis (Classification)

- • To build :

  Creating a Pandas Dataframe from the dataset

  Creating a NumPy array from the column Class in data

  Creating training and testing sets

- To evaluate :

  Using SVM, Classification Trees, k nearest neighbors

  and Logistic Regression method;

- To improve :

  Using the method score to calculate the accuracy on the test data for each model and plotted a confussion matrix for each using the test and predicted outcomes.

Predictive Analysis (Classification)

To found the best performing classification classifying model :

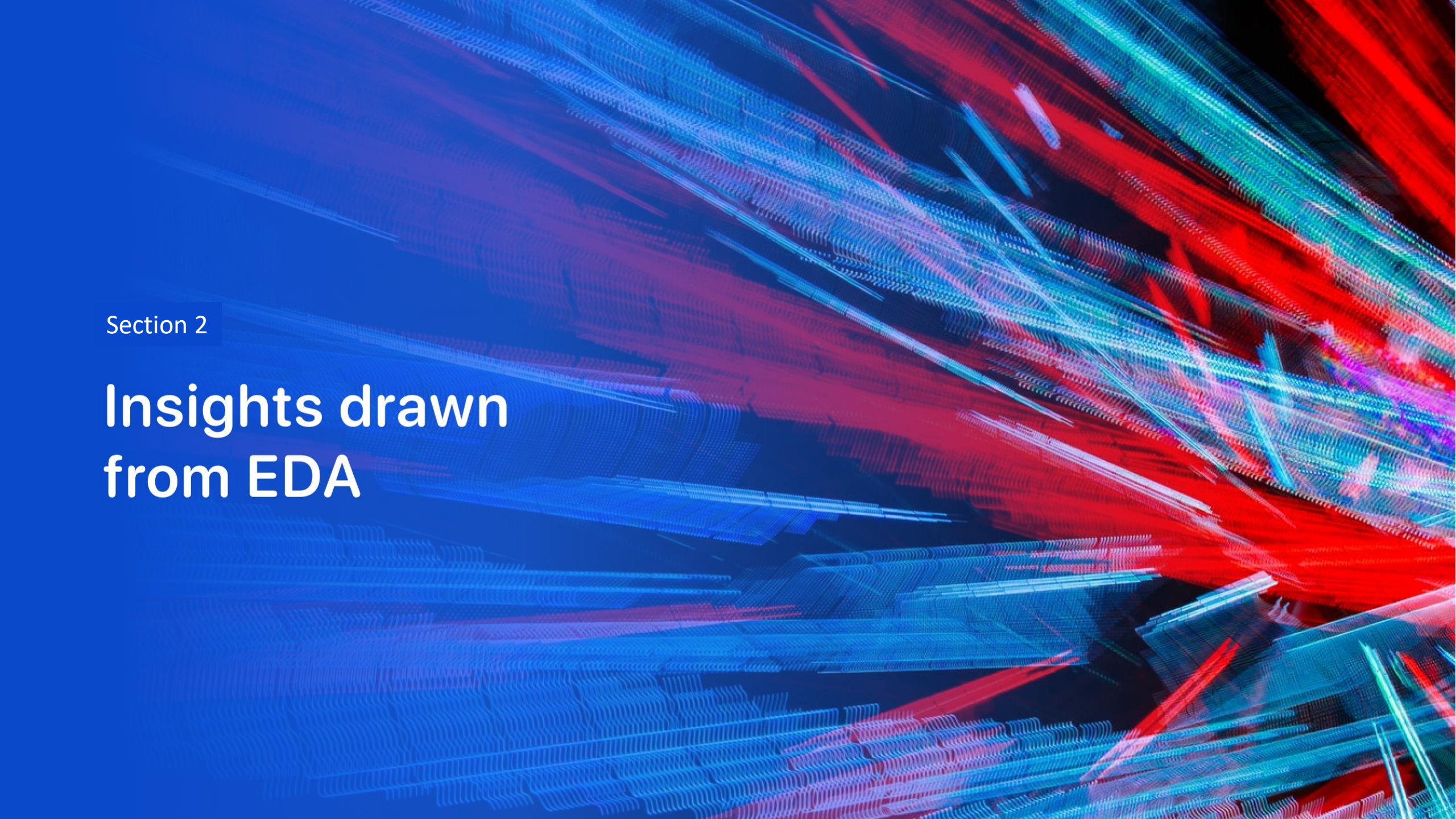
The table resume the test data accuracy score for each of the methods

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.888889
KNN	0.833333

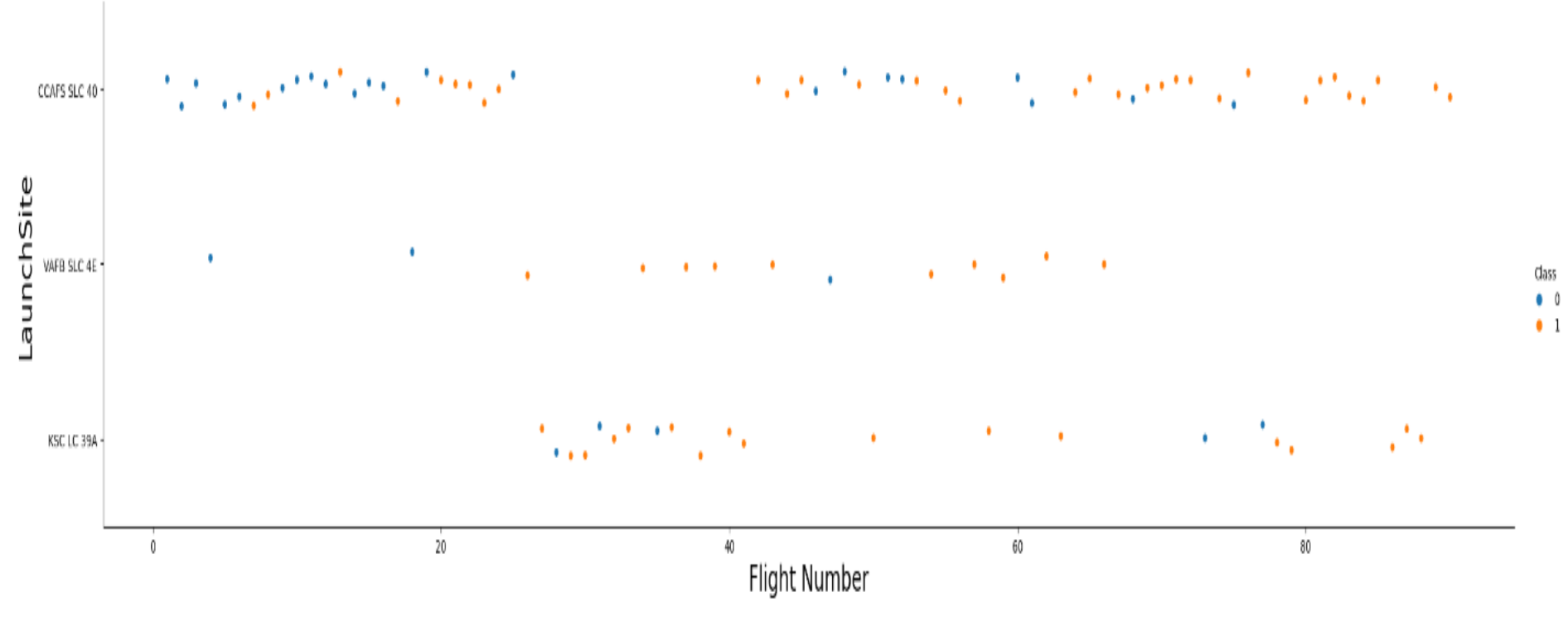
Here is the the GitHub URL of the predictive analysis lab
https://github.com/Aabdouni/testrepoIBM/blob/Applied-Data-Science-Capstone-Project/8%20-%20Spacex_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

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

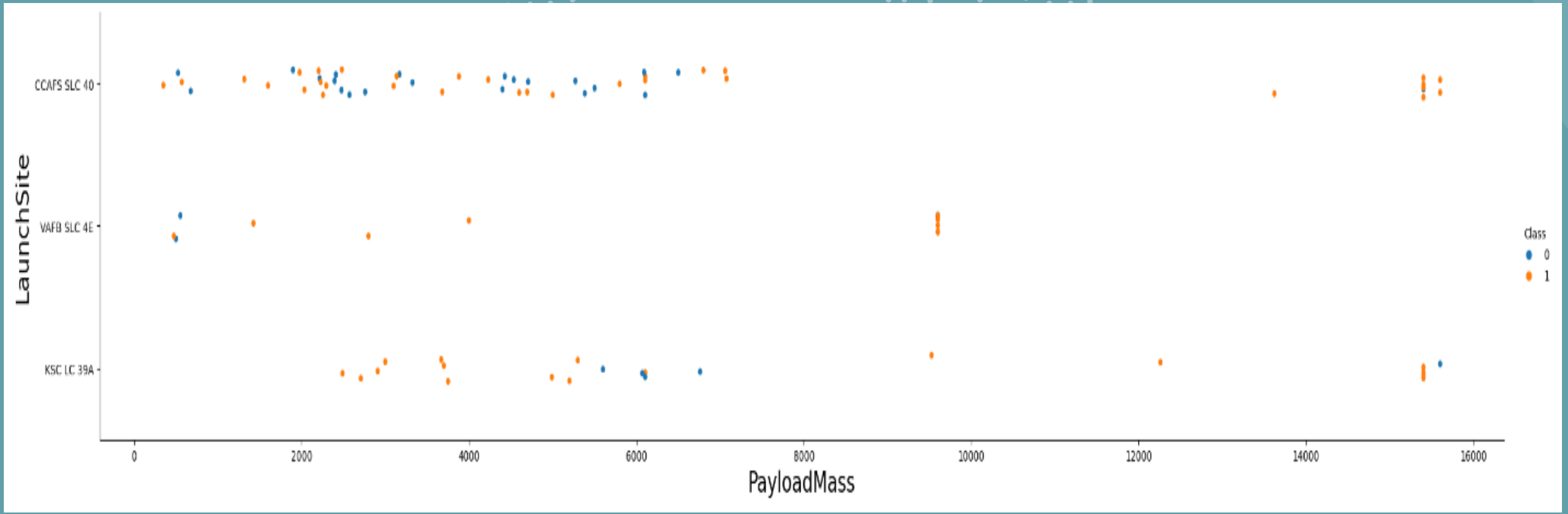

Section 2
Insights drawn from EDA

Flight Number vs. Launch Site



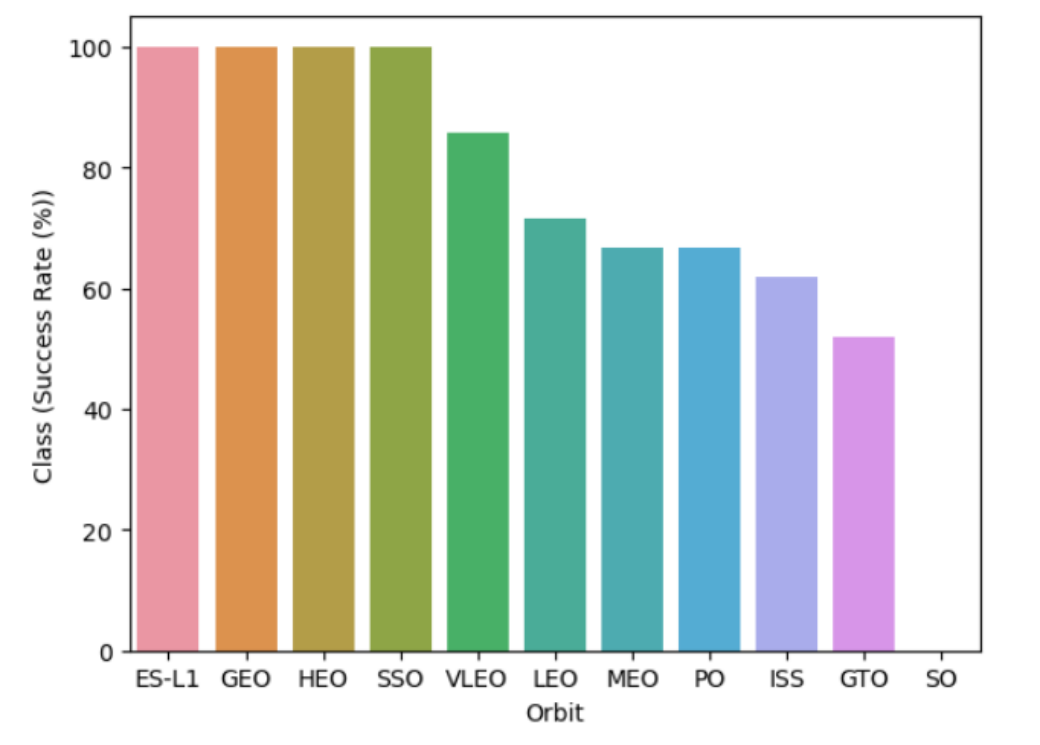
As you can see CCAFS SLC 40 needed more launch to be successful, the others needed less.

Payload vs. Launch Site



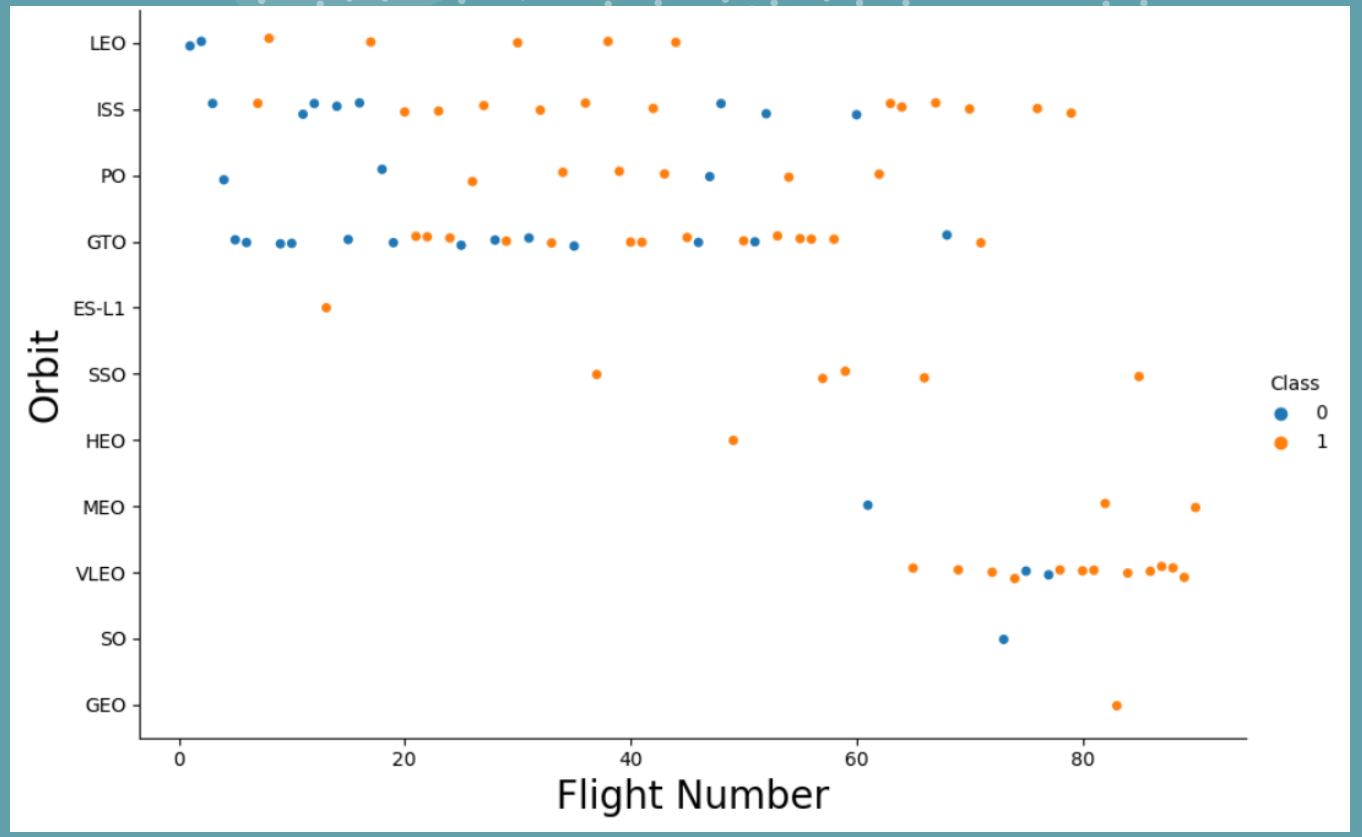
Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

The success rate of each orbit type



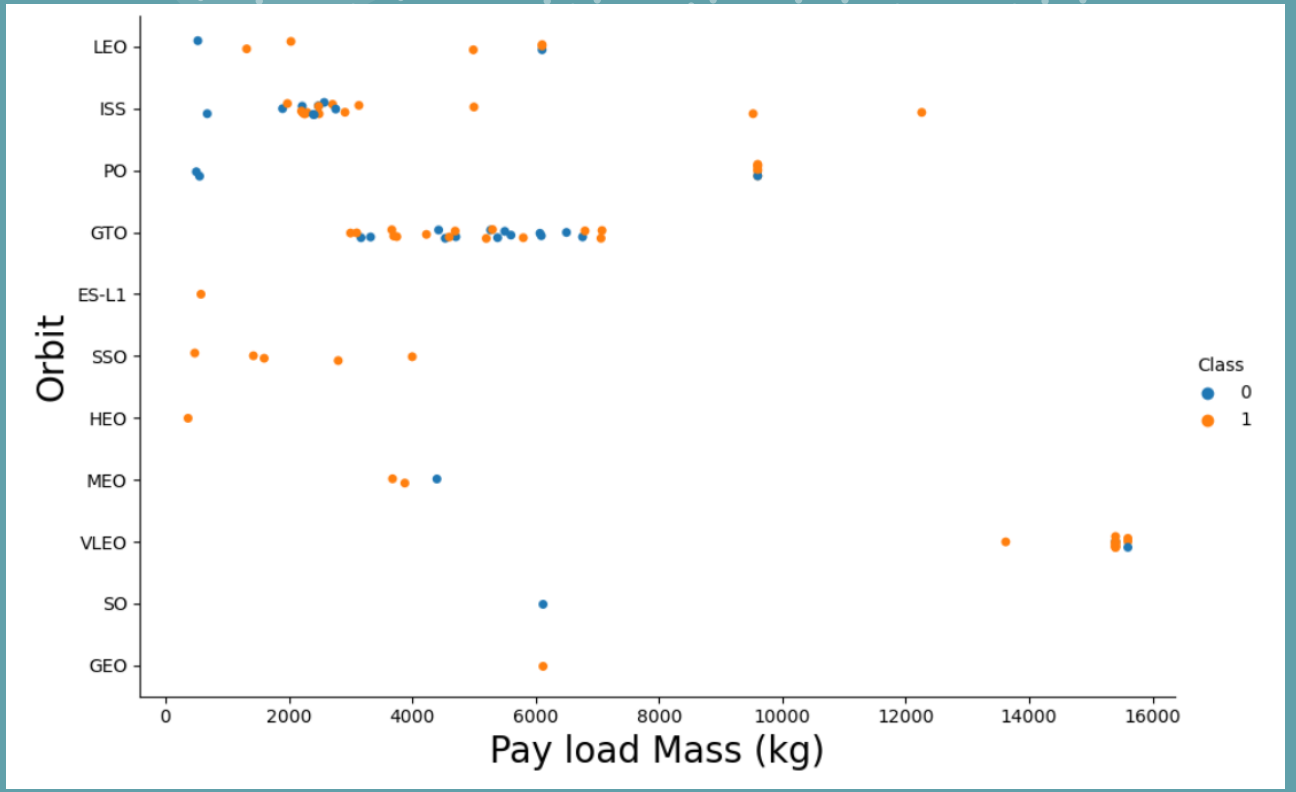
The bar chart is clear, ES-L1 GEO HHEO and SSO have the highest success rate 100%.

Flight Number vs. orbit type



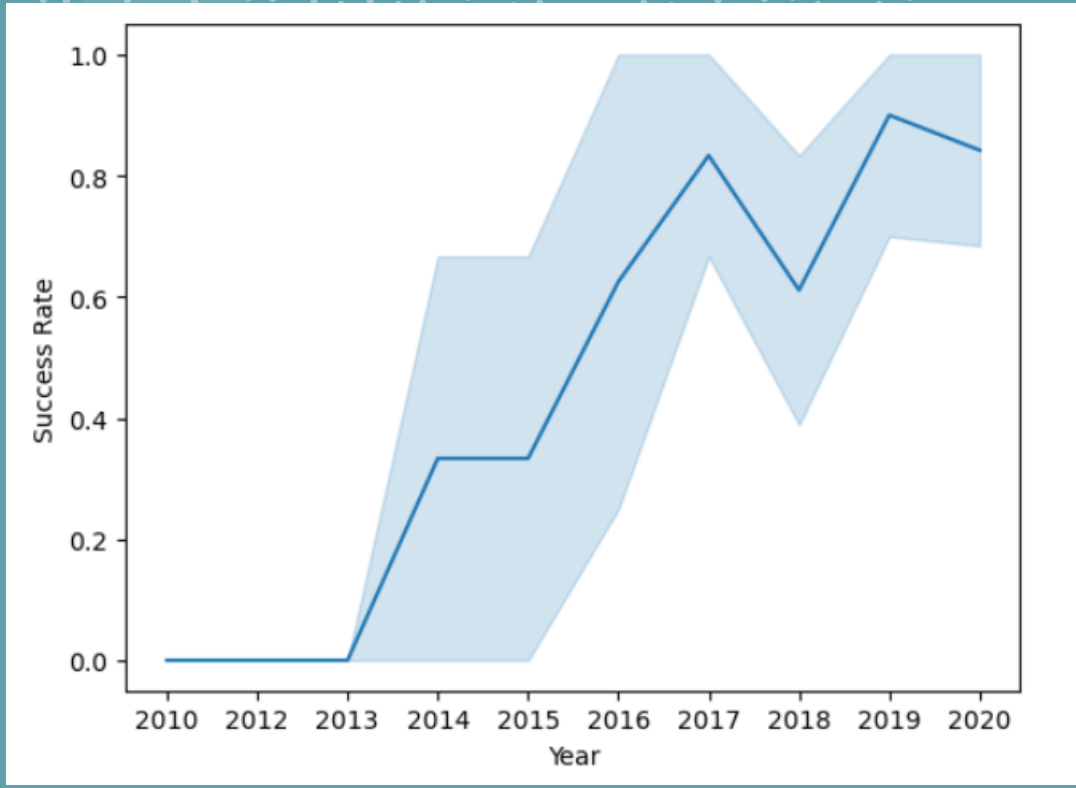
You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

Payload vs. orbit type



- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here.

Launch success Yearly Trend



You can observe that the success rate since 2013 kept increasing till 2020

All Launch site names

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

```
%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

Find the names of the unique launch sites by using the ‘DISTINCT’ statement to return the unique valune Launch_Site (name of th sites) from the table SPACEXTBL.

5 records where Launch site names begin with ‘CCA’

Display 5 records where launch sites begin with the string '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
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
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 (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

To find the 5 records you have to filter the table Launch_site with the ‘Like’ statement and ‘%’ wildcard, and ‘LIMIT 5’ to have 5 records.

Total Payloads Mass

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

```
%sql SELECT sum(PAYLOAD_MASS__KG_) as "Total payload mass" FROM SPACEXTBL WHERE Customer like "%NASA (CRS)%"
```

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

Total payload mass
48213

To find the total mass carried by booster launched by Nasa (CRS), we have to used the SUM statement and to filter the table Launch_site with the ‘Like’ statement and argument “NASA (CRS)L%”.

Average Payload Mass by F9 v1.1

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) as "Average payload mass" FROM SPACEXTBL WHERE Booster_Version like 'F9 v1.1'
```

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

Average payload mass
2928.4

To find the average payload mass carried by booster version F9 v1.1, we have to use the AVG statement and to filter the table SPACEXTBL with the ‘Like’ statement and argument “F9 v1.1” argument for the column Booster_Version.

First Successful Ground Landing Date

List the date when the first succesful landing outcome in ground pad was acheived.

Hint:Use min function

```
%sql SELECT MIN(Date) FROM SPACEXTBL WHERE Landing_Outcome like 'Success (ground pad)'
```

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

MIN(Date)
2015-12-22

To find the date when the first successful landing outcome in ground was achieved.We used the MIN statement and to filter the table SPACEXTBL with the ‘Like’ statement and ‘Success (ground pad)’ argument for the column Landing_Outcome

Successful Drone Ship Landing with Payload between 4000 and 6000

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

```
%sql SELECT  Booster_Version as " Booster Version" FROM SPACEXTBL WHERE Landing_Outcome like "Success (drone ship)" \
AND (PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000 )
```

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

Booster Version
F9 FT B1022
F9 FT B1026
F9 FT B1021.2
F9 FT B1031.2

Here we used the AND,> and < operators and separators ( ) to filter the records.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

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

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

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

Here we used the ‘COUNT’ operator to count the number of records, and ‘Group By’ operator to group the count by MISSION_OUTCOME.

Boosters Carried Maximum Payload

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

```
%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

Here we used the ‘DISTINCT’ and ‘MAX’ operators to select the record where PAYLOAD_MASS__KG_ are the maximum values od the table SPACEXTBL.

2015 Launch Records

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.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

```
%sql SELECT SUBSTR(Date, 0,5) AS Year, SUBSTRING(Date, 6, 2) AS Month, * FROM SPACEXTBL WHERE Landing_Outcome like "Failure (drone ship)" AND Year = '2015'
```

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

Year	Month	Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2015	10	2015-10-01	09:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2015	04	2015-04-14	20:10:00	F9 v1.1 B1015	CCAFS LC-40	SpaceX CRS-6	1898	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)

Here we used the ‘SUBSTR’ operator to extract the year and the month from the Date, also ‘Like’ statement and ‘AND’ operator to filter the record from the SPACEXTBL.

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.

```
%sql SELECT Landing_Outcome, COUNT(Landing_Outcome) AS TOTAL_NUMBER FROM SPACEXTBL  \
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing_Outcome ORDER BY TOTAL_NUMBER DESC
```

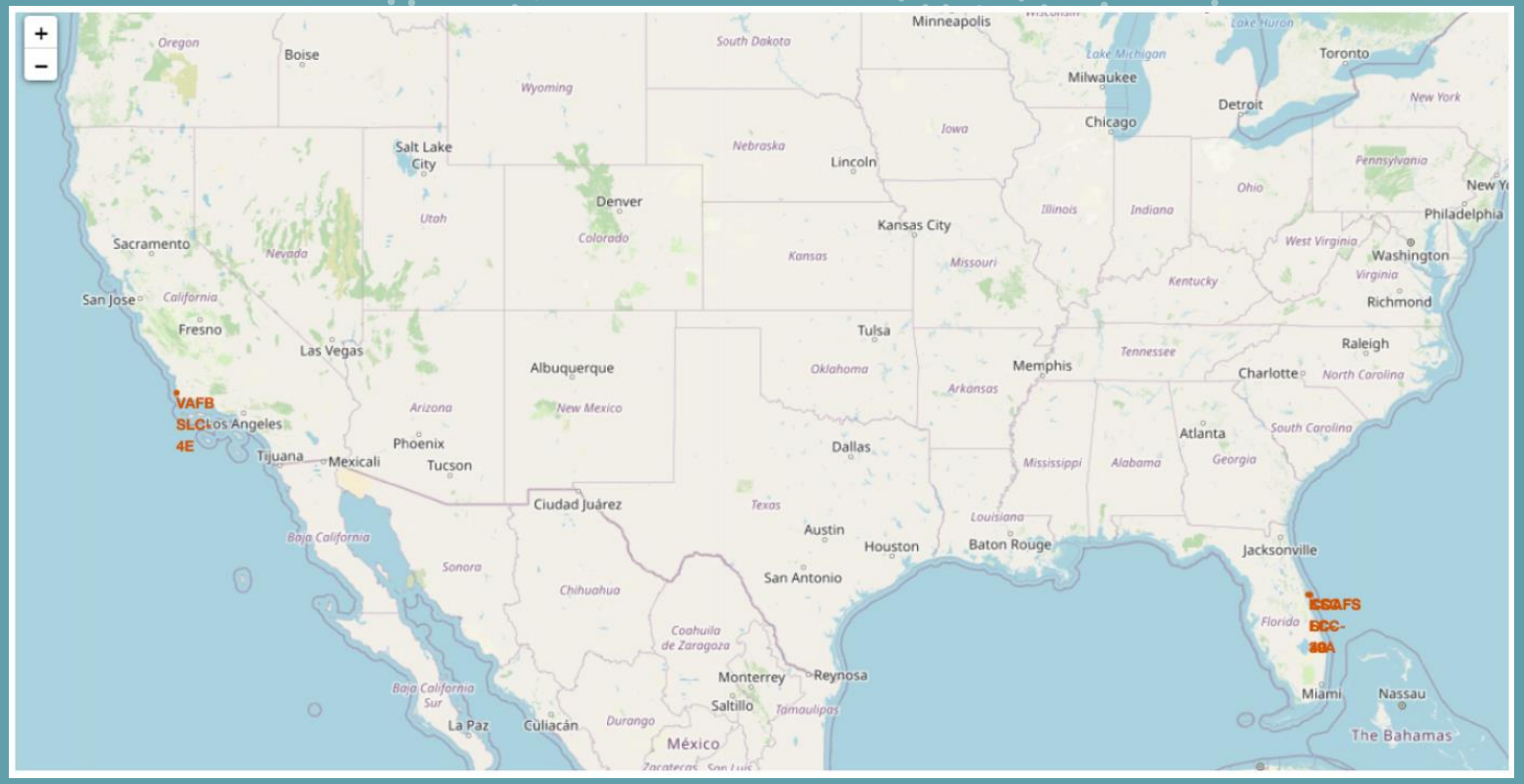
```
 * sqlite:///my_data1.db
Done.
```

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

Here we used the ‘BETWEEN’ operator to select records between 2 values, and ‘ORDER BY’ to define an order of results list.


Section 3
Launch Sites
Proximities Analysis

Launch sites

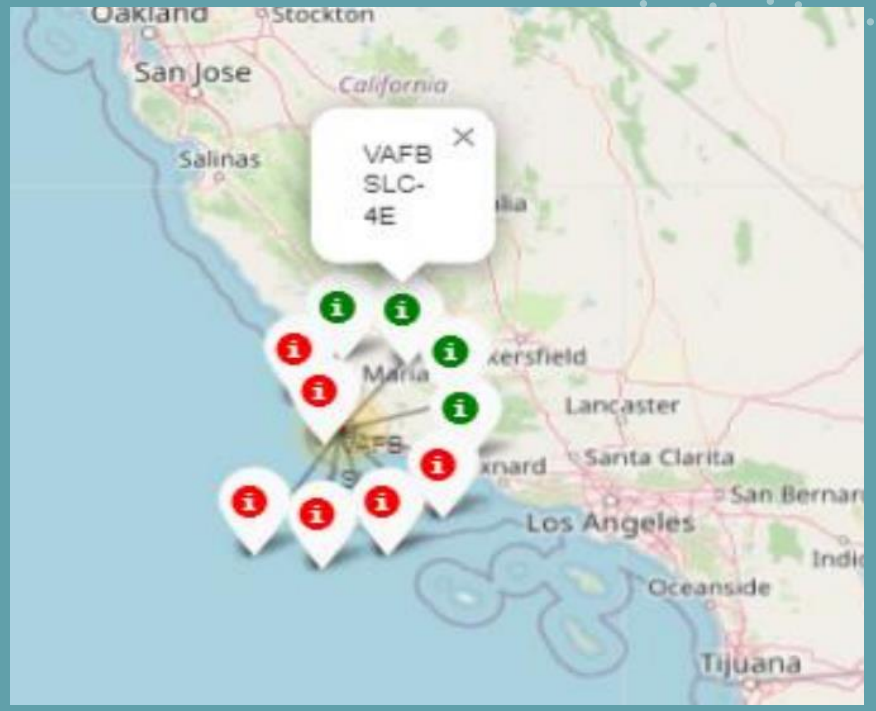


In the West USA we have the launch site VAFB SLC-42 near the Vandeberg Space force Base in California and the others launch sites in the East USA near the Cape Canaveral Space Force Station in Florida.

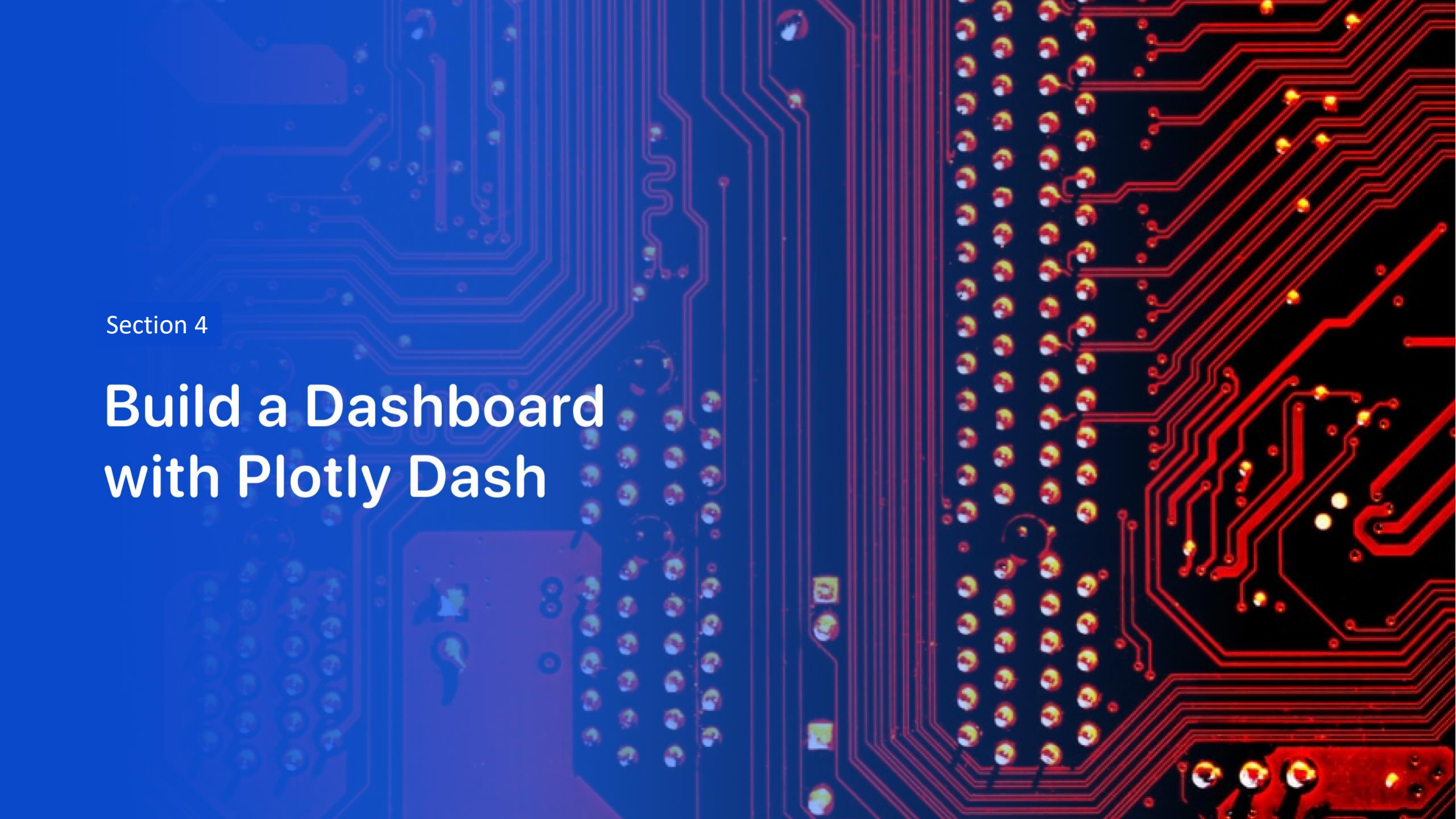
Launch sites in Florida

As you see, in Florida, the launch site KSC LC-39A has the high success rates compared to CCAFS SLC-40 & CCAFS LC-40.

Launch site in California

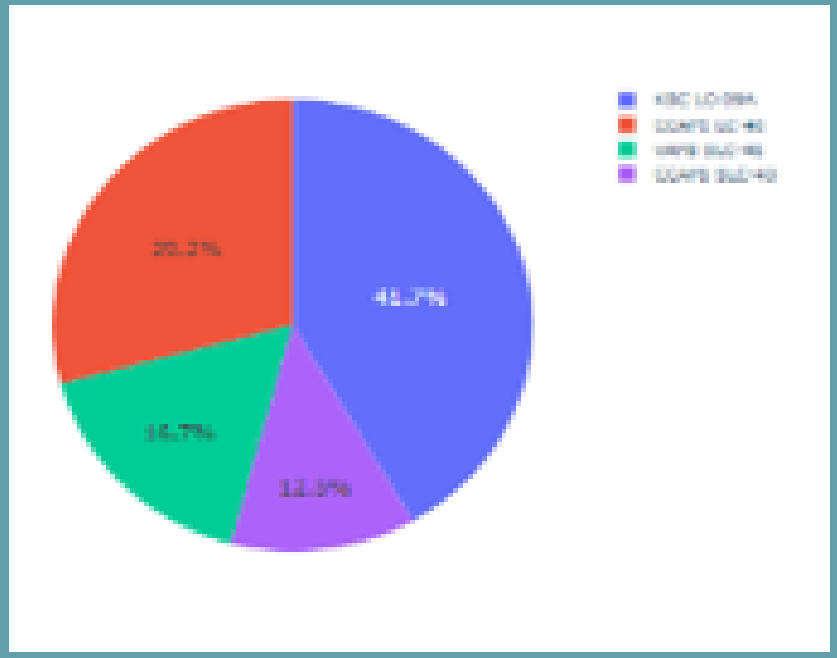


In California, the Launch site VAFB SLC-4E has the lower success rates 4/10 compared to all launch sites.


Section 4
Build a Dashboard with Plotly Dash

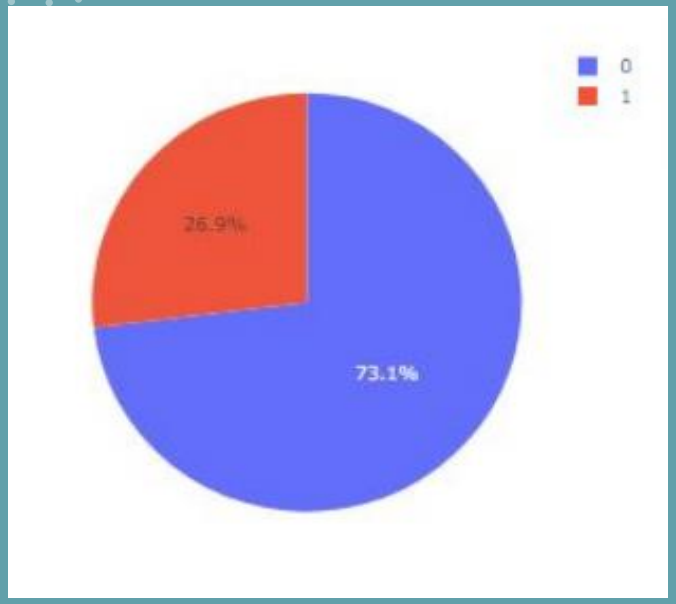
Launch success count for all sites

Launch site KSC LC-39A has the highest launch success rate at 42% followed by CCAFS LC-40 at 29%, VAFB SLC-4E at 17% and lastly launch site CCAFS SLC-40 with a success rate of 13%

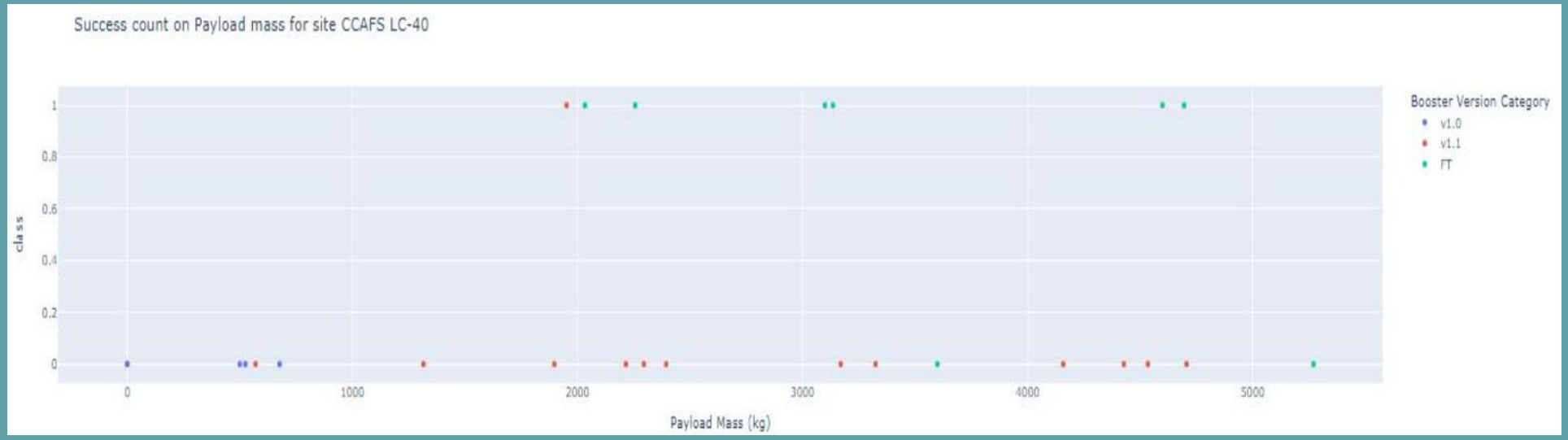


The launch site with highest launch success ratio

Launch site CCAFS SLC-40 classed 2nd for all sites, but it has the highest launch success ratio with 73,1% success compared to 26,9% failed launched.



Payload vs. Launch Outcome scatter plot for all sites



We can see that the booster version FT has the largest success rate from a payload mass of >2000kg

Section 5

Predictive Analysis (Classification)

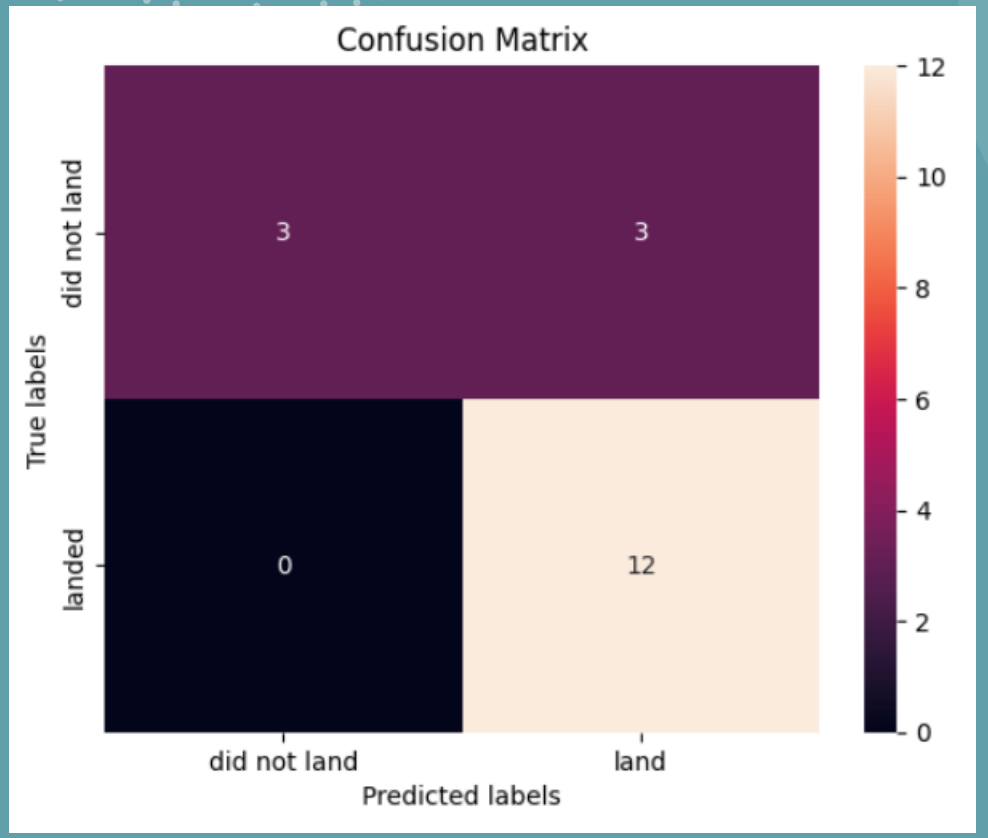
Classification Accuracy

- In this figure you can see that Decision Tree has the best Accuracy 0.888889 instead 0,.833333 for the others.

	0
Method	Test Data Accuracy
Logistic_Reg	0.833333
SVM	0.833333
Decision Tree	0.888889
KNN	0.833333

Confusion Matrix

- In this analsing we used Using Logistic Regression, Support Vector Machine, Decision Tree, K Nearest Neighboor and finally we constate that we have the same Confusion Matrix result.



Conclusion

- To compare success rates of launch sites, we can see CCAFS LC-40 has a lowest success rate 60 %, while KSC LC-39A and VAFB SLC 4E has a success rate of 77%.
- To compare the flight number for each launches sites. The success rate for the VAFB SLC 4E launch site is 100% after the Flight number 50. Both KSC LC 39A and CCAFS SLC 40 have a 100% success rates after 80th flight
- If you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).
- Orbits ES-L1, GEO, HEO & SSO have the highest success rates at 100%, with SO orbit having the lowest success rate at ~50%. Orbit SO has 0% success rate.
- LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit
- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS. However for GTO we cannot distinguish this well as both positive landing rate and negative landing(unsuccessful mission) are both there here
- Anf finally the sucess rate since 2013 kept increasing till 2020.

Thank you!