IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Sheriff Said Elahl>
<1st of December, 2021>

Outline

Executive Summary

In this research we are trying to determine the best model to predict if a certain SpaceX flight will have a successful landing on Earth based on different flight characteristics.

As summarized result of the whole data investigation through this research we came to the conclusion that there is an association between SpaceX landing success and many aspects, however the strongest correlations are found with flight number, launching site location, payload mass and orbit. We also came to conclusion that based on the public SpaceX data available, the best model to predict the landing status is the Decision Tree Algorithm.

Introduction

Space is no longer far away and the competition in both governmental and private sectors is very hot. One of the critical aspects in this race is the cost of launch where recently SpaceX made a huge breakthrough by landing the first stage of their rockets back on earth. So as a player in this game, it would be a competitive advantage if we can predict whether a certain flight that is going to be launched soon will have successful landing based on its main features. Consequently, the flight parameters can be modified to bring the highest assurance for successful landing.

many questions are on table for such a target, such as:

- what are the most relevant features to success rate and what is the relation between them?
- What is the best Machine Learning (ML) algorithm that can accomplish the prediction task with optimum results?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

  1- Collecting from SpaceX – Rest API.

  2- Collecting by web scrapping from Wikipedia.

- Perform data wrangling
  - Replacing missing values with mean values or dropping irrelevant data.
  - Using one hot encoding to process the final features into ML model inputs.
- 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

1- We used SpaceX API to collect the first part of data.

2- we used BeautifulSoup library to scrap some extra data from Wikipedia.

Data Collection – SpaceX API

- URL of the required object is saved.
- Get request with output as response.
- Response is normalized.

GitHub link:

https://github.com/SherifSaidElahl/IBMCapstone/blob/master/Data%20Collection%20API.ipynb

```
In [12]: static_json_url='https://cf-courses-data.s3.us.cloud-object-storage.appdom
```

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

```
In [13]: response.status_code
Out[13]: 200
```

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

```
In [22]: # Use json_normalize meethod to convert the json result into a dataframe
response3 = requests.get(static_json_url).json()
data = pd.json_normalize(response3)
```

```
In [23]: # Get the head of the dataframe
data.head()
```

Out[23]:

	static_fire_date_utc	static_fire_date_unix	tbd	net	window	rocket	success	details	c
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Engine failure at 33 seconds and loss of vehicle	
1	None	NaN	False	False	0.0	5e9d0d95eda69955f709d1eb	False	Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine shutdown	

Data Collection - Scraping

- URL of the required object is saved.
- Get request with output as response.
- Response is converted to HTML.
- HTML is inserted into Beautifulsoup.
- Required tables are collected

GitHub link:

https://github.com/SherifSaidElahl/IBMCapstone/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb

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

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

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

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

```
# Use soup.title attribute
print(soup.title)
```

Data Wrangling

- Missing values were checked.
- columns dtypes were checked.
- Unique site names are collected.
- Column ‘landing_class’ is introduced to represent the status of each landing.

Pad	Block	ReusedCount	Serial	Longitude	Latitude	Class
NaN	1.0	0	B0003	-80.577366	28.561857	0
NaN	1.0	0	B0005	-80.577366	28.561857	0
NaN	1.0	0	B0007	-80.577366	28.561857	0
NaN	1.0	0	B1003	-120.610829	34.632093	0
NaN	1.0	0	B1004	-80.577366	28.561857	0

GitHub link:

https://github.com/SherifSaidElahl/IBMCapstone/blob/master/EDA.ipynb

EDA with Data Visualization

In visualization we had different plots such as:

1. scatter plots:
    - FlightNumber vs. PayloadMass while result as class.
    - scatter plots to represent the relation of FlightNumber vs LaunchSite while result as class.
    - scatter plots to represent the relation of Launch Site vs. PayloadMass while result as class.
    - FlightNumber and Orbit while result as class.
    - Payload vs. Orbitwhile result as class.
2. Bar Chart to represent the relation of success rate of each orbit.
3. Line Cart to represent relation of year and success rate.

GitHub link:

https://github.com/SherifSaidElahl/IBMCapstone/blob/master/EDA%20with%20Data%20Visualization.ipynb

EDA with SQL

we performed the below queries:

1. %sql select distinct Launch_Site from SPACEXDATASET
2. %sql select * from SPACEXDATASET where Launch_Site like 'CCA%' limit 5
3. %sql select sum(payload_mass__kg_) from SPACEXDATASET where customer = 'NASA (CRS)‘
4. %sql select avg(payload_mass__kg_) from SPACEXDATASET where Booster_Version = 'F9 v1.1‘
5. %sql select min(Date) from SPACEXDATASET where landing__outcome = 'Success (ground pad)‘
6. %sql select booster_version from SPACEXDATASET where landing__outcome = 'Success (drone ship)' and (payload_mass__kg_ > 4000 and payload_mass__kg_ < 6000)
7. %sql select count(*) from SPACEXDATASET where mission_outcome = 'Success‘
8. %sql select distinct booster_version from SPACEXDATASET where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXDATASET)
9. %sql select Date, landing__outcome, booster_version, launch_site from SPACEXDATASET where landing__outcome = 'Failure (drone ship)' and Date like '%2015%‘
10. %sql select landing__outcome, count(*) as count from SPACEXDATASET where (Date >= '06-04-2010' and Date <= '03-20-2017') group by landing__outcome

GitHub Link:

https://github.com/SherifSaidElahl/IBMCapstone/blob/master/EDA%20with%20SQL.ipynb

Build an Interactive Map with Folium

- we built a folium map for the world.
- we added markers and circles to launch locations.
- we grouped location into clusters.
- and then we differentiated the landing status by color where red is failure and green is success.
- we identified some land markers like reailways and costs.
- we determined if launch locations are close of each of them.

GitHub:

https://github.com/SherifSaidElahl/IBMCapstone/blob/master/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

Build a Dashboard with Plotly Dash

- we made a plotly Dash board.
- we added to the board a drop down to select Lanuch site.
- we added a Pie Chart to represent success rate for the selected site.
- we added a range slider to input payload mass.
- and added a scatter plot to show success rate with payload and site.

GitHub:

https://github.com/SherifSaidElahl/IBMCapstone/blob/master/SpaceX
Dash_Coursera.py

Predictive Analysis (Classification)

- We split the data using train test split.
- we used Grid Search to get the best paramaters.
- we used different models (LR, SVM, D. Tree and KNN)
- we built a confusion matrix to each one and score for each one.
- based on the confusion matrix we get what is the best model to be implemented.

- GitHub:
- https://github.com/SherifSaidElahl/IBMCapstone/blob/master/Machine%20Learning%20Prediction.ipynb

Results

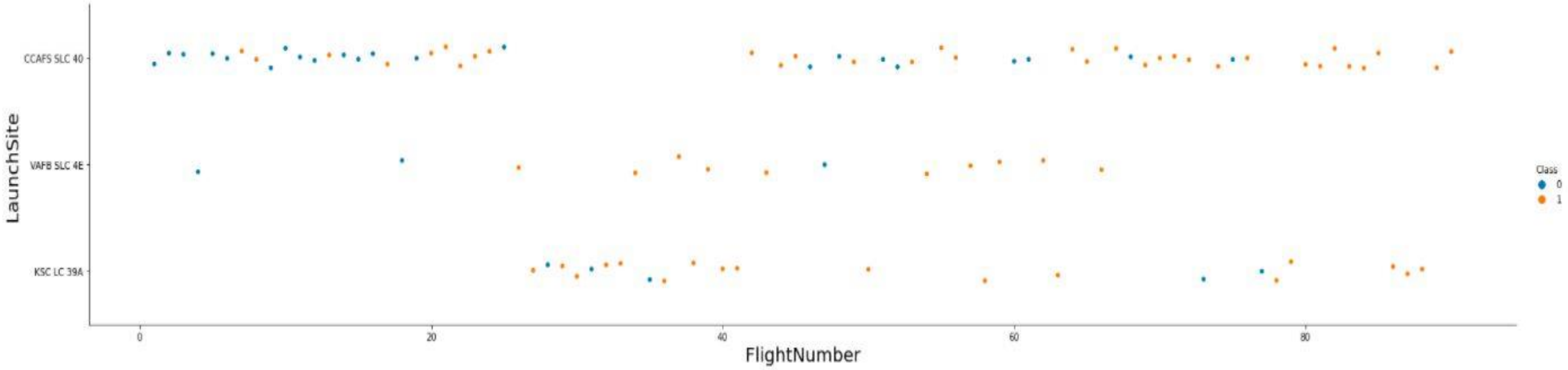
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

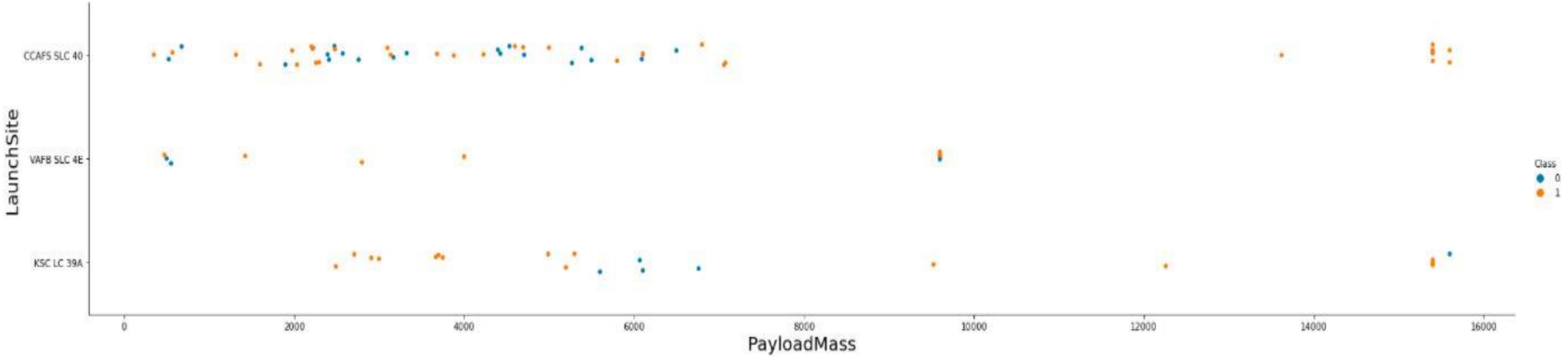
Flight Number vs. Launch Site

In [9]:
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("FlightNumber",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



Payload vs. Launch Site

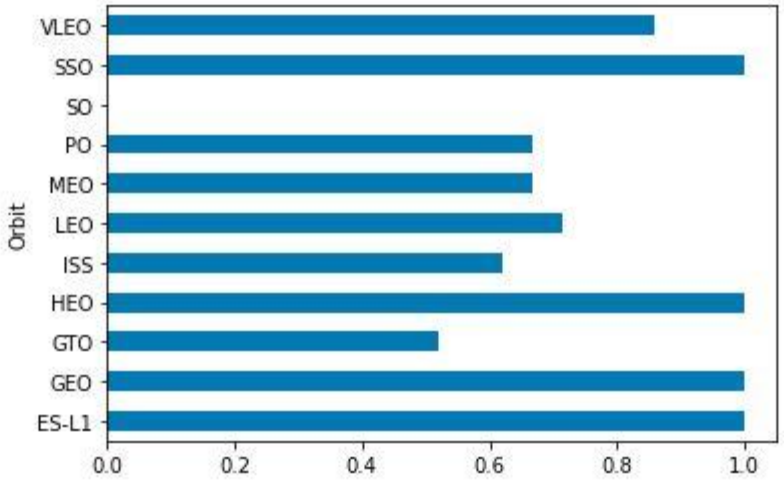
```
[10]: # Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("LaunchSite",fontsize=20)
plt.show()
```



Success Rate vs. Orbit Type

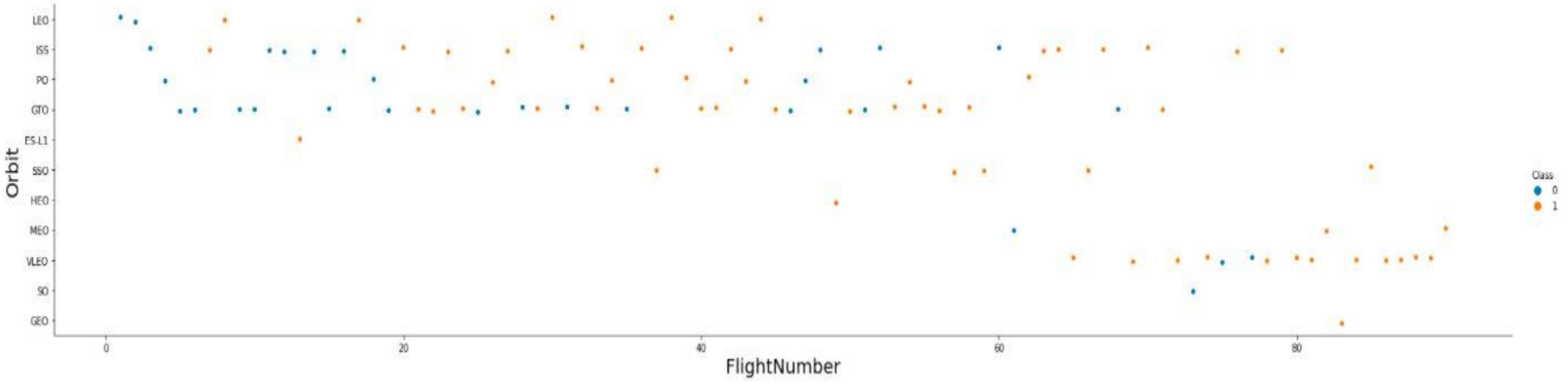
```
In [18]: # HINT use groupby method on Orbit column and get the mean of Class column
         df_group_by = df.groupby(['Orbit'])['Class'].mean()
         df_group_by.plot(kind='barh')
```

```
Out[18]: <AxesSubplot:ylabel='Orbit'>
```



Flight Number vs. Orbit Type

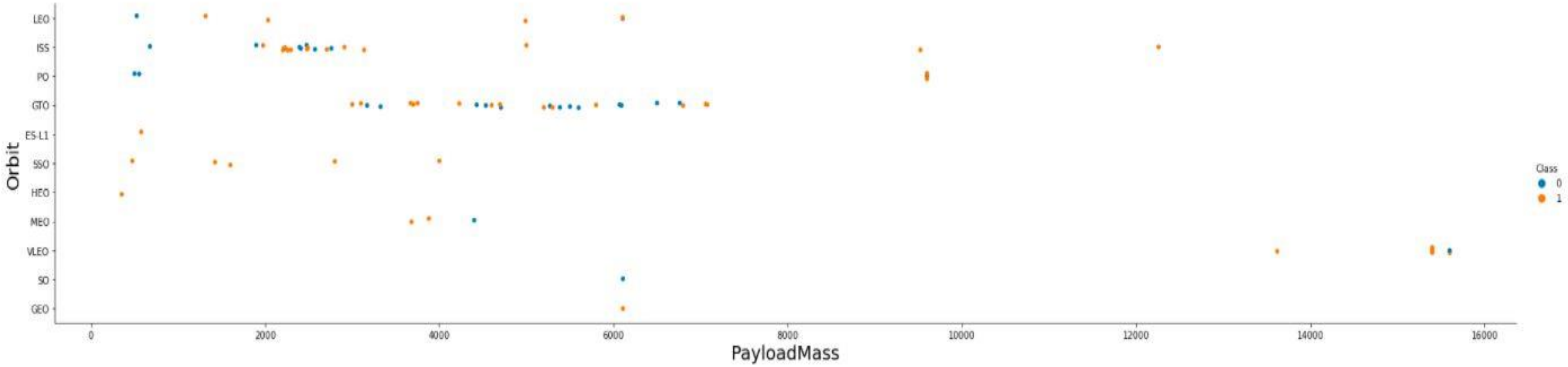
```
In [19]: # Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
         sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
         plt.xlabel("FlightNumber",fontsize=20)
         plt.ylabel("Orbit",fontsize=20)
         plt.show()
```



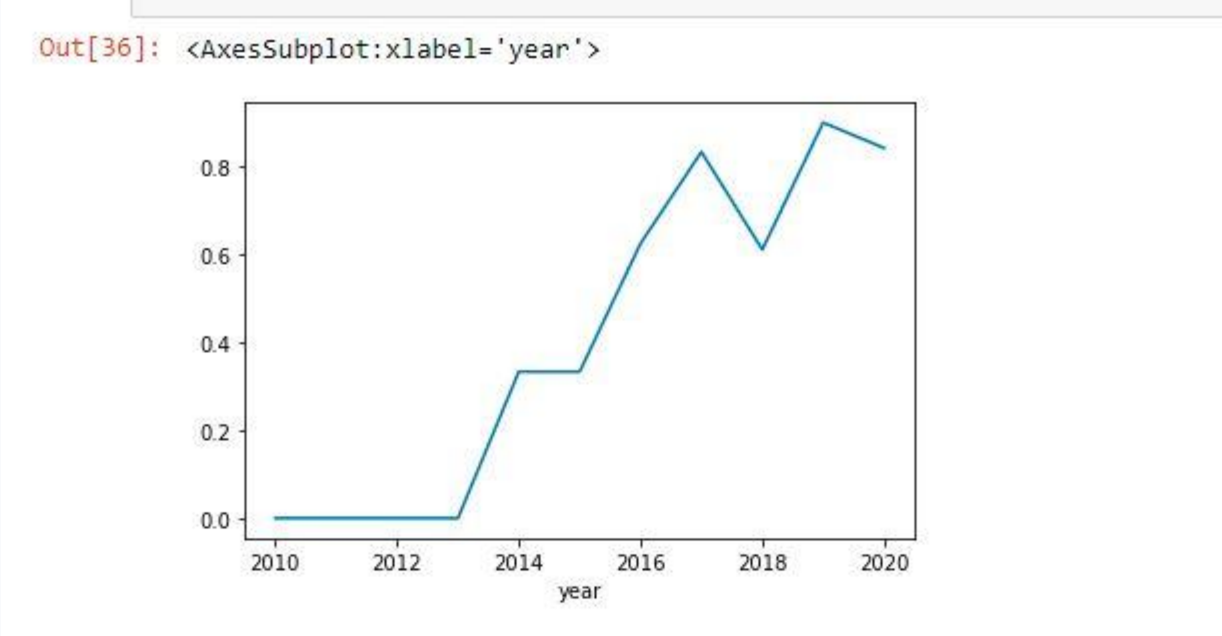
Payload vs. Orbit Type

In [21]:
```
# Plot a scatter point chart with x axis to be Payload and y axis to be the Orbit, and hue to be the class value

sns.catplot(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.xlabel("PayloadMass",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



Launch Success Yearly Trend



All Launch Site Names

Done.

Out[5]:

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

Launch Site Names Begin with 'CCA'

```
In [15]: %sql select * from SPACEXDATASET where Launch_Site like 'CCA%' limit 5
```

```
 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[15]:

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	None	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	None	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	07:44:00	F9 v1.0 B0005	CCAFS LC-40	None	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	None	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	None	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

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

In [19]:
```
%sql select sum(payload_mass__kg_) from SPACEXDATASET where customer = 'NASA (CRS)'
```

```
 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[19]:

1
45596

Average Payload Mass by F9 v1.1

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

```
In [21]: %sql select avg(payload_mass__kg_) from SPACEXDATASET where Booster_Version = 'F9 v1.1'

 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.

Out[21]:
```

1
2928

First Successful Ground Landing Date

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

Hint:Use min function

In [29]:
```
%sql select min(Date) from SPACEXDATASET where landing__outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[29]:

1
2015-12-22

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

In [34]:
```
%sql select booster_version from SPACEXDATASET where landing__outcome = 'Success (drone ship)' and (payload_mass__kg_ > 4000 and payload_mass__kg_ < 6000)
```

```
 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[34]:

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

Total Number of Successful and Failure Mission Outcomes

Task 7

List the total number of successful and failure mission outcomes

In [42]:
```
%sql select count(*) from SPACEXDATASET where mission_outcome = 'Success'
```

```
 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[42]:

1
99

Boosters Carried Maximum Payload

```
In [48]: %sql select distinct booster_version from SPACEXDATASET where payload_mass__kg_ = (select max(payload_mass__kg_) from SPACEXDATASET)

 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.

Out[48]:
```

booster_version
F9 B5 B1048.4
F9 B5 B1048.5
F9 B5 B1049.4
F9 B5 B1049.5
F9 B5 B1049.7
F9 B5 B1051.3
F9 B5 B1051.4
F9 B5 B1051.6
F9 B5 B1056.4
F9 B5 B1058.3
F9 B5 B1060.2
F9 B5 B1060.3

2015 Launch Records

```
In [57]: %sql select Date, landing__outcome, booster_version, launch_site from SPACEXDATASET where landing__outcome = 'Failure (drone ship)' and Date like '%2015%'
```

```
 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[57]:

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

In [76]:
```
%sql select landing__outcome, count(*) as count from SPACEXDATASET where (Date >= '06-04-2010' and Date <= '03-20-2017') group by landing__outcome
```

```
 * ibm_db_sa://tyr08267:***@764264db-9824-4b7c-82df-40d1b13897c2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:32536/BLUDB
Done.
```

Out[76]:

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

Section 4

Launch Sites Proximities Analysis

Launch Sites Map

- as shown, sites are grouped into two clusters

Launch Sites Map

- by clicking on the cluster we can see there are two sites.
- each site contains several launch flights.
- successful launches is in green while failure is in red.

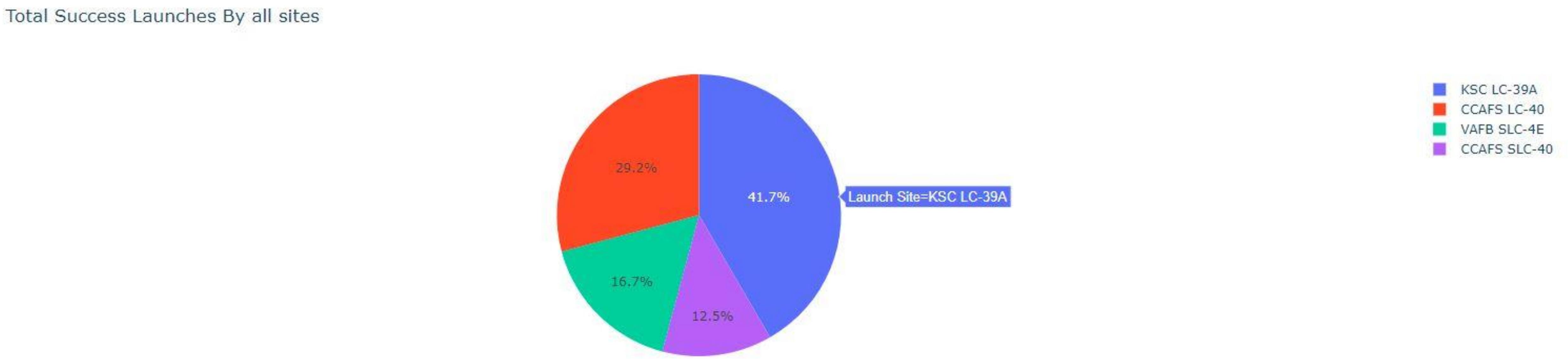
Launch Sites Map

- Distance is measured to coast line.

Section 5

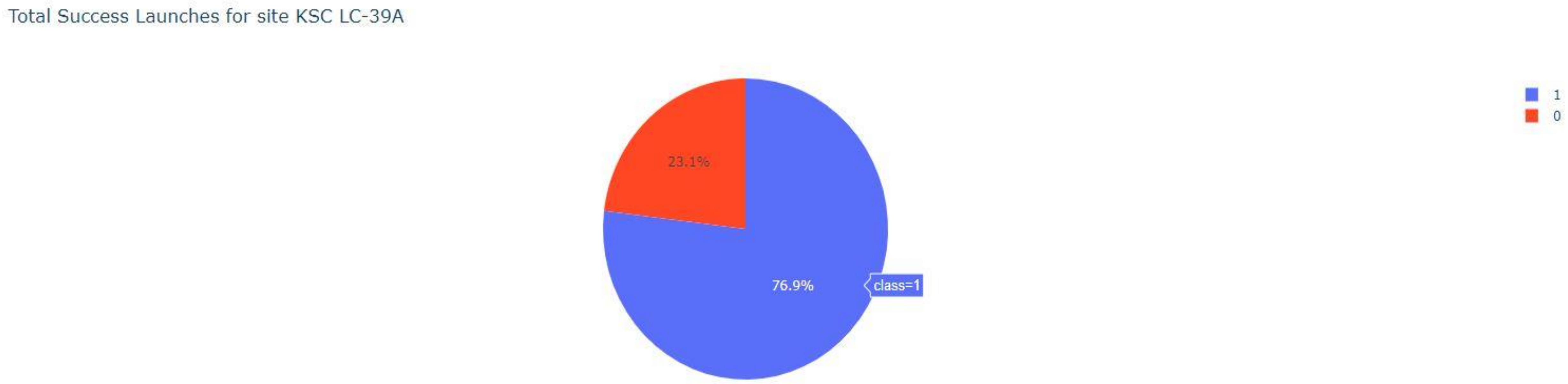
Build a Dashboard with Plotly Dash

Success Rate Distribution



Highest site with successful lunch sites is KSC-LC-39A

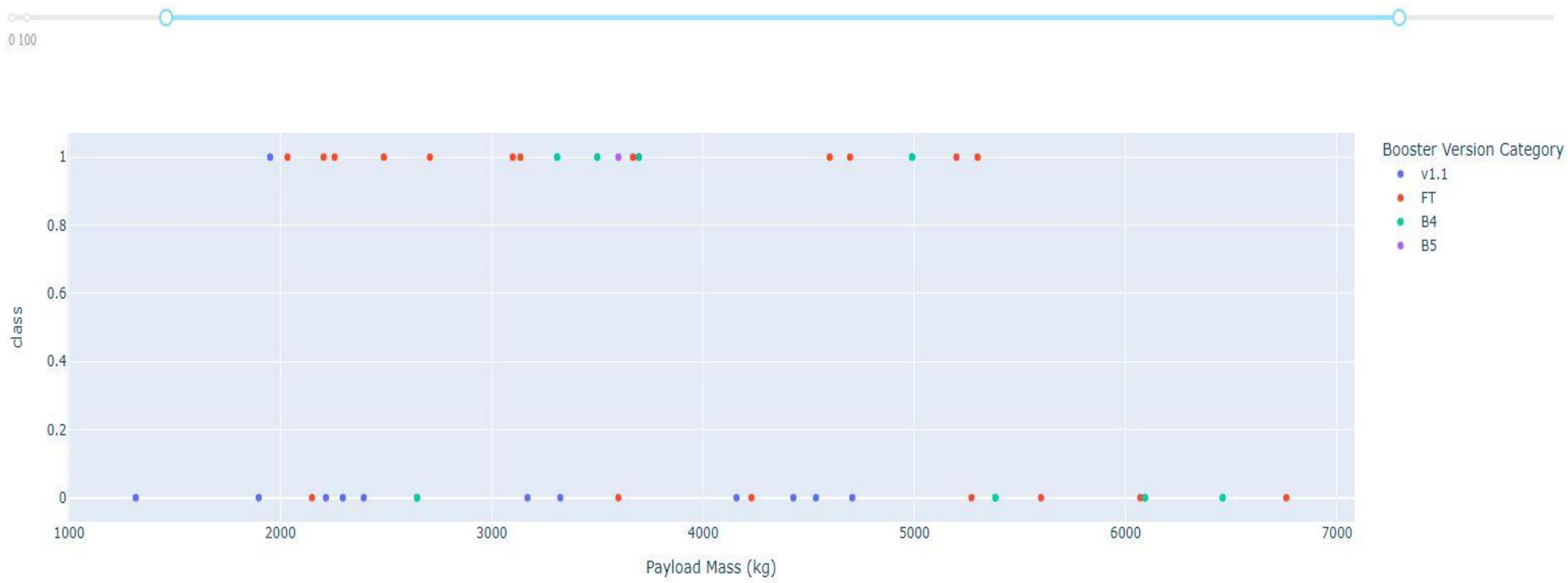
KSC LC 39A Site Success Rate



• highest site of success has the rate of 76.9%

Payload Vs Class with different Booster Versions

Payload range (Kg):



Section 6

Predictive Analysis (Classification)

Confusion Matrix

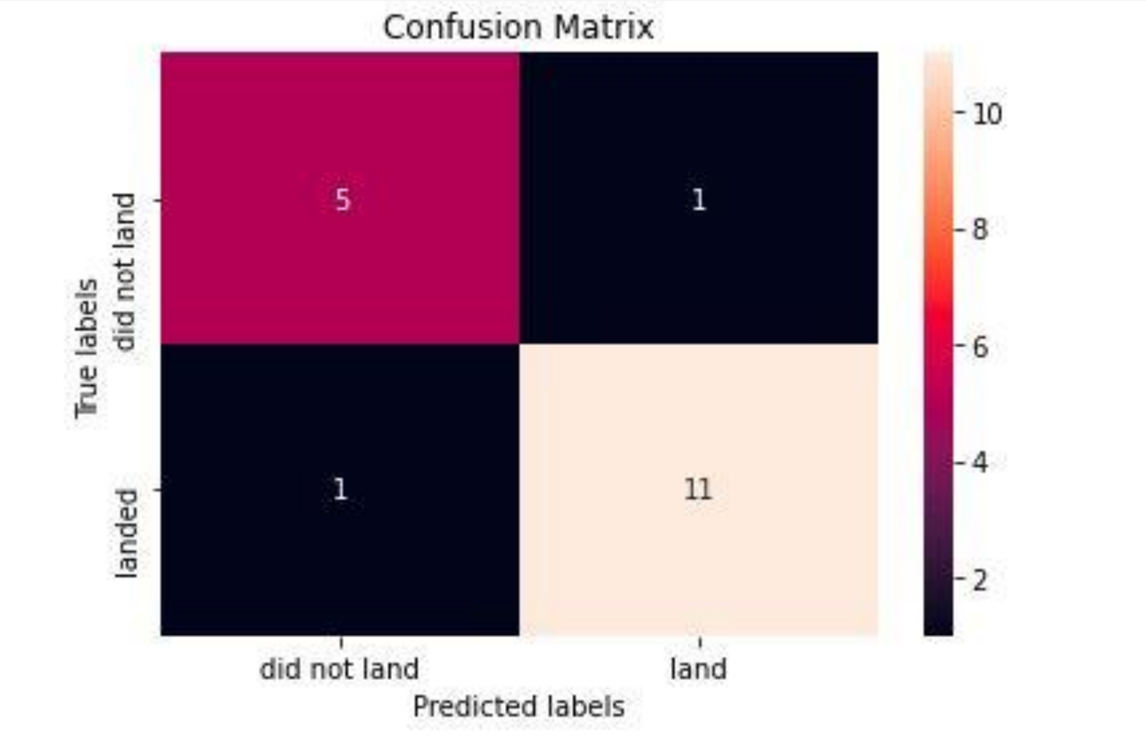
D. Tree Model Confusion Matrix.

as shown:

true positive is very high.

true negative is accurate.

only 1 false positive and 1 false negative



Conclusions

1. Successful landing is strongly correlated with flight number, with higher numbers associated higher success rate.
2. Successful landing is highest on KSC LC 39A Site.
3. Booster FT version is generally the best performing.
4. Orbits (SSO, GEO, ES-L1 and HEO) are the highest orbits with successful rate.
5. Decision Tree Algorithm is the best prediction model

Thank you!