IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Name : Yurgen Ivan Arias Colodro
Date : Febraury 2023

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

- Summary of methodologies

We use Data Collection with scraping , Wrangling Data using an API, Sampling Data, and Dealing with Nulls, Exploratory Data analysis, Data Visualization, SQL Queries and Predictive Analysis with the help of machine learning models such as SVM, logistic regression, tree of decision and KNN

- Summary of all results

The results will yield machine learning models capable of predicting if a launch will be successful, as well as interesting data that influences the result, such as the best places to launch and if the mass of the load is a factor related to the success rate.

Introduction

- Project background and context

The commercial space age is here, companies are making space travel affordable for everyone. Virgin Galactic is providing suborbital spaceflights. Rocket Lab is a small satellite provider. Blue Origin manufactures sub-orbital and orbital reusable rockets. Perhaps the most successful is SpaceX. SpaceX's accomplishments include: Sending spacecraft to the International Space Station. Starlink, a satellite internet constellation providing satellite Internet access. Sending manned missions to Space. One reason SpaceX can do this is the rocket launches are relatively inexpensive. SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upwards of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage.

- Problems we want to find answers

The job is to determine the price of each launch. We will do this by gathering information about Space X and creating dashboards for our team. We will also determine if SpaceX will reuse the first stage. Instead of using rocket science to determine if the first stage will land successfully, we will train a machine learning model and use public information to predict if SpaceX will reuse the first stage.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:

  The data was collected using an API, specifically the SpaceX REST API. This API will provide us with data about launches, including information about the rocket used, the payload delivered, the launch specifications, the landing specifications, and the landing outcome.

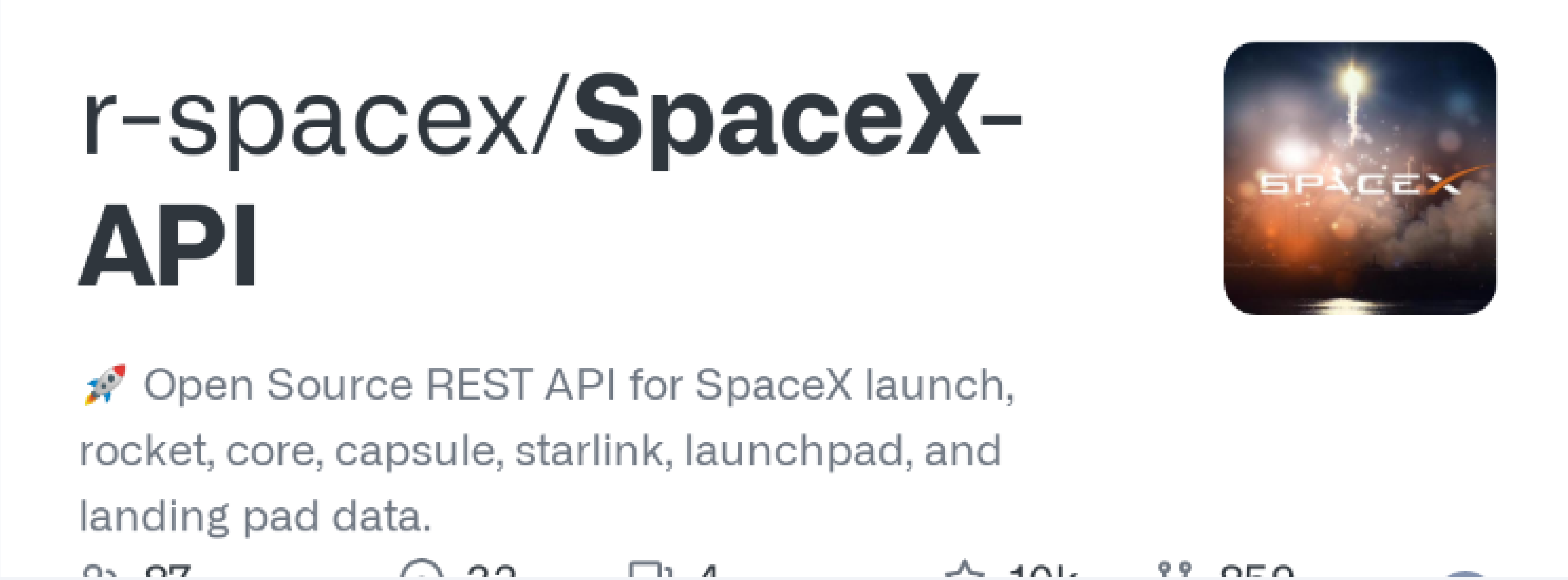
- Perform data wrangling

  We use: Wrangling Data using an API, Sampling Data, and Dealing with Nulls.

- 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

- The data was collected using an API, specifically the SpaceX REST API. This API will provide us with data about launches, including information about the rocket used, the payload delivered, the launch specifications, the landing specifications, and the landing outcome.



Data Collection – SpaceX API

- We use the method Get to obtaing the data from url:

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

- Use json_normalize meethod to convert the json result into a dataframe

```
In [11]: data = pd.json_normalize(response.json())
```

```
data.head()
```

Out[12]:

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	details	crew	ships	capsules	payloads	launchpad	flight_r
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Engine failure at 33 seconds and loss of vehicle	[]	[]	[]	[5eb0e4b5b6c3bb0006eeb1e1]	5e9e4502f5090995de566f86	
1	None	NaN	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 301, 'altitude': 289, 'reason': 'harmonic oscillation	Successful first stage burn and transition to second stage, maximum altitude 289 km, Premature engine	[]	[]	[]	[5eb0e4b6b6c3bb0006eeb1e2]	5e9e4502f5090995de566f86	

- Then we dilter the dataframe to only include `Falcon 9` launches l and the data frame will look something like:

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9.head()
```

8]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.632093
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004	-80.577366	28.561857

- Finally we treat the missing values using the median with the help of the .replace() method of the pandas library

```
# Calculate the mean value of PayloadMass column
mean = data_falcon9[["PayloadMass"]].mean()
print(mean)
# Replace the np.nan values with its mean value
data_falcon9 = data_falcon9.replace(np.nan,mean)
data_falcon9.isnull().sum()
```

- https://github.com/Yarias2106/Cousera_IBM_Data_Science_Professional_Certificate/blob/master/1.%20Spacex-data-collection-api.ipynb

Data Collection – Scraping

- We use the method Get to obtaing the data and create an object with the library Beautiful Soup

```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
response = requests.get(static_url).text
```

```
In [5]: soup = BeautifulSoup(response)
```

- Create a data frame by parsing the launch HTML tables

```
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
```

After traversing the object with the help of the find_all() method we will get our data frame:

```
df=pd.DataFrame(launch_dict)
df.head()
```

t[59]:

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success\n	F9 v1.0	Success\n	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0	Success	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0	Success	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success\n	F9 v1.0	Success\n	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success\n	F9 v1.0	Success\n	1 March 2013	15:10

https://github.com/Yarias2106/Cousera_IBM_Data_Science_Professional_Certificate/blob/fae9f5b51520ef74b703e9a168e3f22d50dc48ee/2.%20Webscraping.ipynb

Data Wrangling

- In the data set, there are several different cases where the booster did not land successfully. Sometimes a landing was attempted but failed due to an accident; for example, ***True Ocean*** means the mission outcome was successfully landed to a specific region of the ocean while ***False Ocean*** means the mission outcome was unsuccessfully landed to a specific region of the ocean. ***True RTLS*** means the mission outcome was successfully landed to a ground pad ***False RTLS*** means the mission outcome was unsuccessfully landed to a ground pad. ***True ASDS*** means the mission outcome was successfully landed on a drone ship ***False ASDS*** means the mission outcome was unsuccessfully landed on a drone ship.

```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_1.csv")
df.head(10)
```

]:

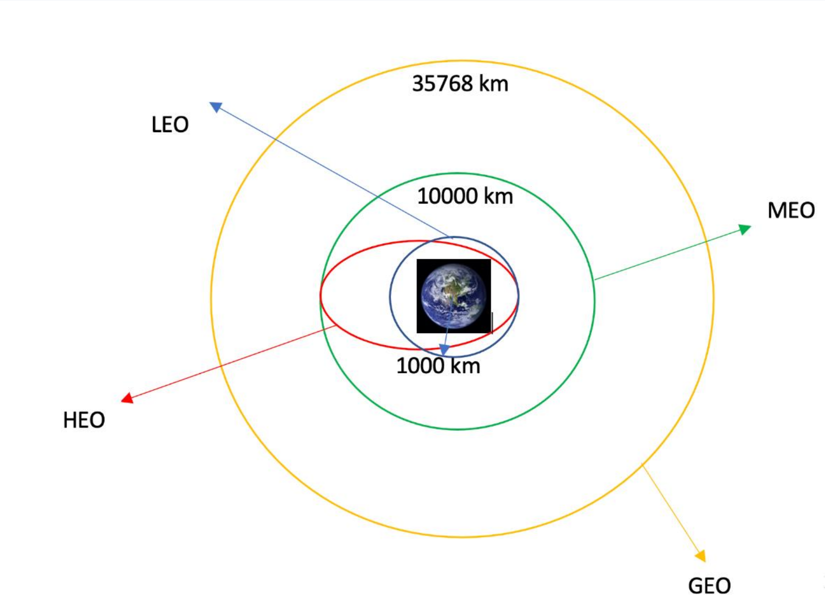
	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005	-80.577366	28.561857
6	7	2014-04-18	Falcon 9	2296.000000	ISS	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1006	-80.577366	28.561857
7	8	2014-07-14	Falcon 9	1316.000000	LEO	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1007	-80.577366	28.561857
8	9	2014-08-05	Falcon 9	4535.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1008	-80.577366	28.561857
9	10	2014-09-07	Falcon 9	4428.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1011	-80.577366	28.561857

- The data contains several Space X launch facilities, Next, let's see the number of launches for each site.

```
df["LaunchSite"].value_counts()
```

```
5]:  CCAFS SLC 40    55
     KSC LC 39A      22
     VAFB SLC 4E     13
     Name: LaunchSite, dtype: int64
```

- Each launch aims to an dedicated orbit:




```
df["Orbit"].value_counts()
```

```
[6]: GTO       27
     ISS       21
     VLEO      14
     PO         9
     LEO        7
     SSO        5
     MEO        3
     ES-L1      1
     HEO        1
     SO         1
     GEO        1
     Name: Orbit, dtype: int64
```

- Now we are going to create a new column called class in the data frame where we will indicate that a mission was successful (1) or if it failed (0).

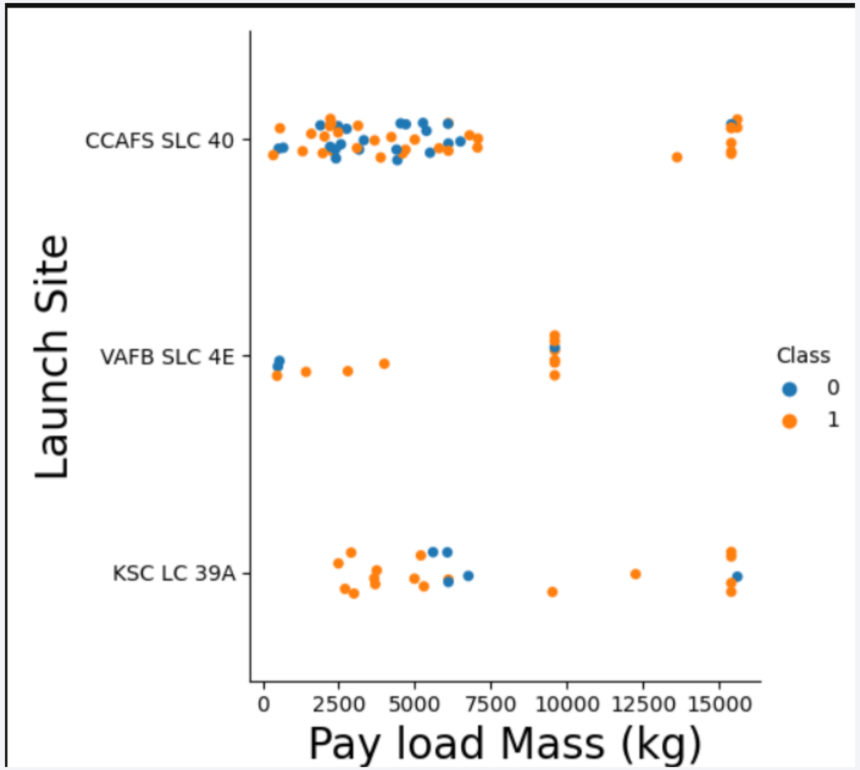
```
landing_outcomes = df["Outcome"].value_counts()
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
landing_class=[]
for i in df["Outcome"]:
 if i in bad_outcomes:
  landing_class.append(0)
 else:
  landing_class.append(1)

df['Class']=landing_class
df.head(5)
```

Serial	Longitude	Latitude	Class
B0003	-80.577366	28.561857	0
B0005	-80.577366	28.561857	0
B0007	-80.577366	28.561857	0
B1003	-120.610829	34.632093	0
B1004	-80.577366	28.561857	0

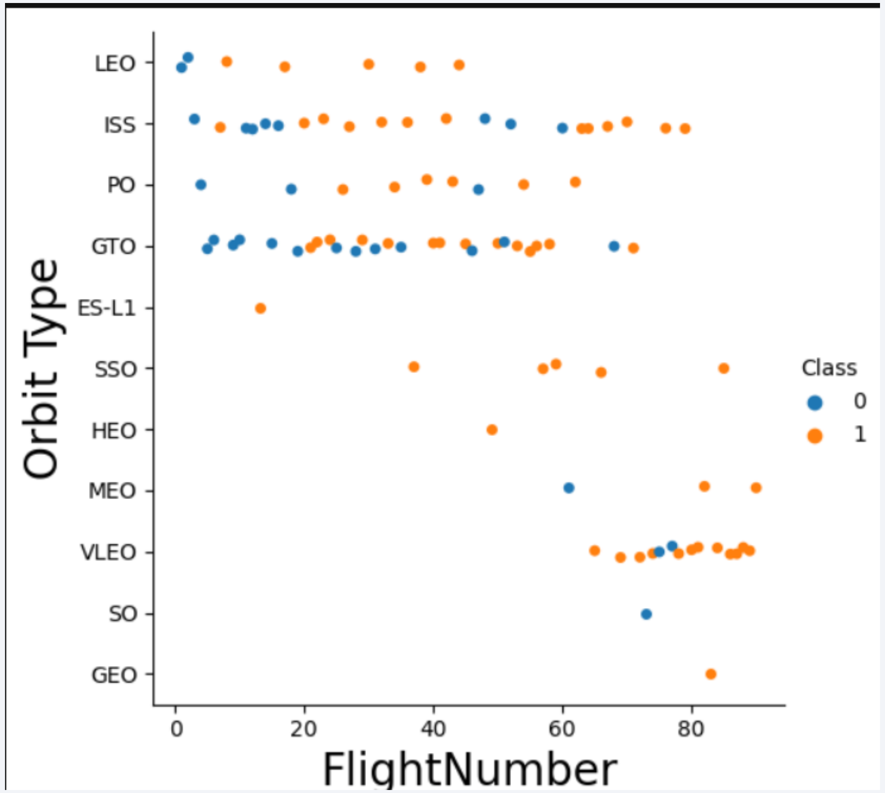
- https://github.com/Yarias2106/Cousera_IBM_Data_Science_Professional_Certificate/blob/b839c5fe4ccdc49805595b6e348ad259af4d5ccd/3.%20Data%20Wrangling.ipynb

EDA with Data Visualization

We can observe Payload Vs. Launch Site We will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

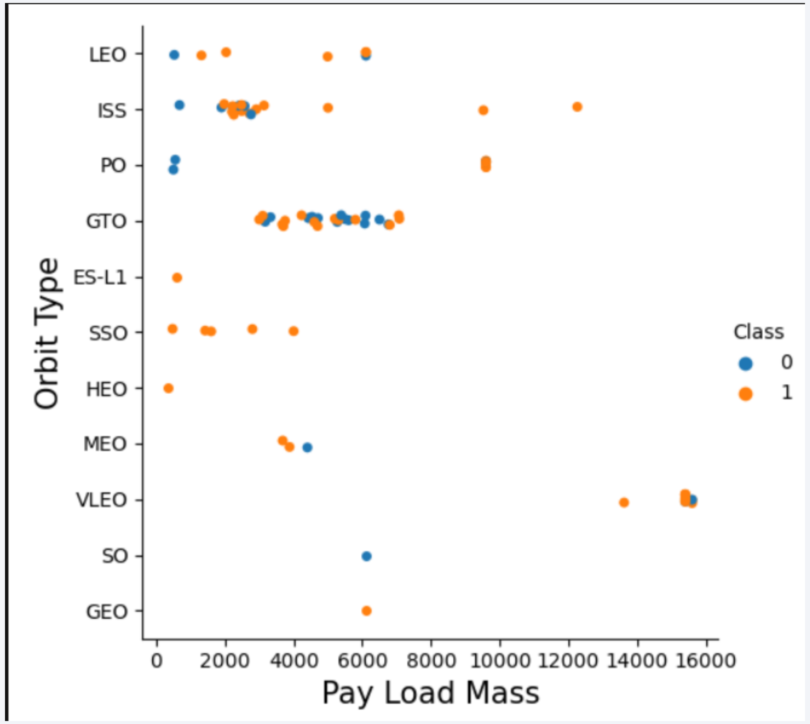


We see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

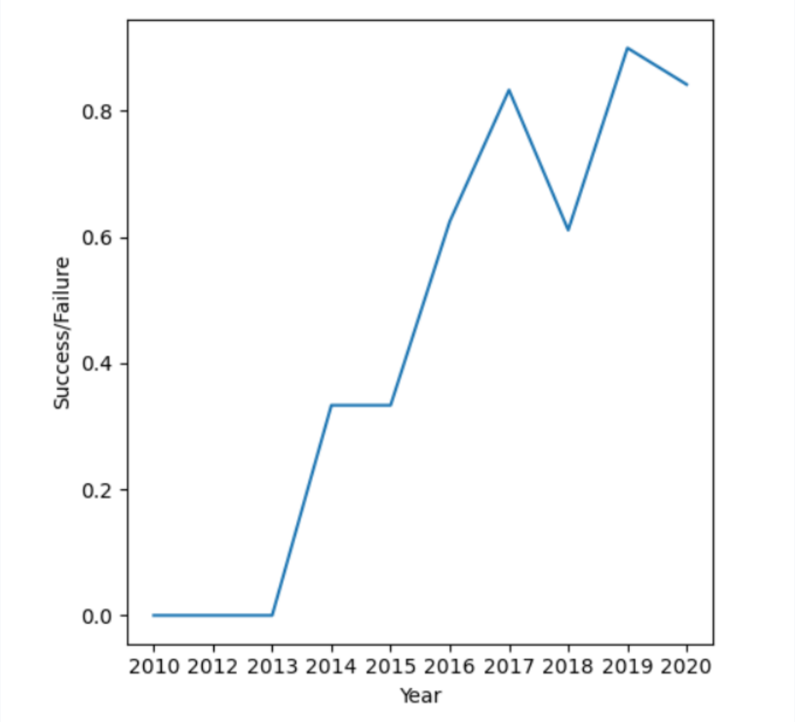


EDA with Data Visualization

- 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.



We can observe that the sucess rate since 2013 kept increasing till 2020.



https://github.com/Yarias2106/Cousera_IBM_Data_Science_Professional_Certificate/blob/121c31863950681c0e973414108607b526b1f55c/4%20Eda-Data%20Visualization.ipynb

EDA with SQL

Summarize the SQL queries you performed

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

```
%sql Select distinct "launch_Site" from "SPACEXTBL"
```

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

```
%sql Select * from "SPACEXTBL" where "launch_site" like "CCA%" limit 5
```

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

```
%sql select sum("PAYLOAD_MASS__KG_") from "SPACEXTBL" where "Customer" = "NASA (CRS)"
```

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

```
%sql select avg("PAYLOAD_MASS__KG_") from "SPACEXTBL"
```

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

```
%sql select min("Date") from "SPACEXTBL"  where "Mission_Outcome"= "Success"
```

- 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"  from "SPACEXTBL" where "Landing _Outcome" = "Success (drone ship)" and "PAYLOAD_MASS__KG_" > 4000 and "PA
```

EDA with SQL

Summarize the SQL queries you performed

- List the total number of successful and failure mission outcomes

```
%sql select count(MISSION_OUTCOME) from SPACEXTBL GROUP BY MISSION_OUTCOME;
```

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

```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

- 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

```
%sql SELECT substr(Date, 4, 2) ,MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE from "SPACEXTBL" where substr(Date,7,4)='2015'
```

- Rank the count of successful landing_outcomes between the date 04-06-2010 and 20-03-2017 in descending order.

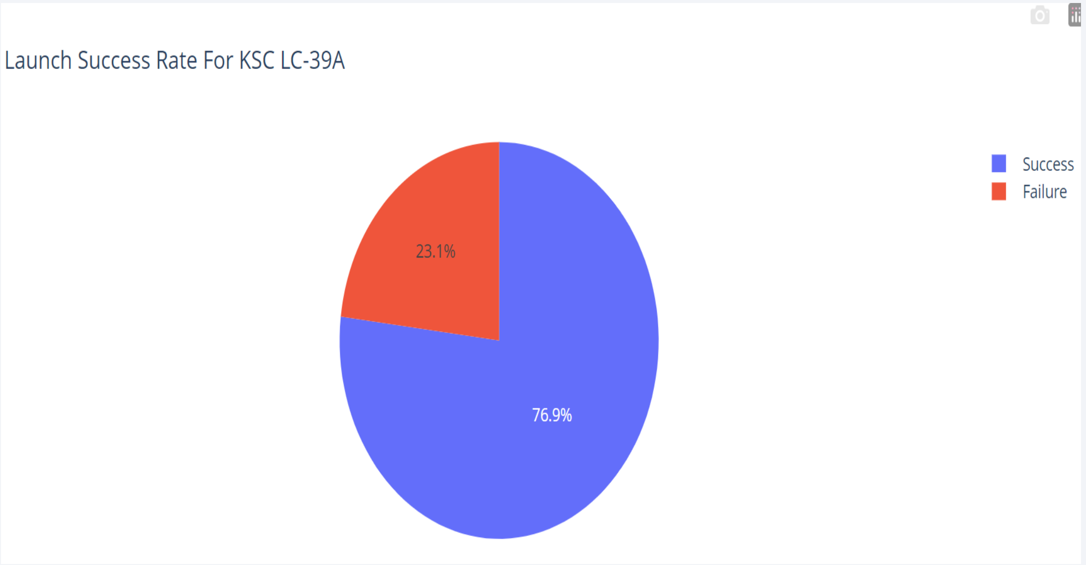
```
%sql SELECT "Landing _Outcome" from SPACEXTBL WHERE "Date" between '04-06-2010 ' and '20-03-2017' ORDER BY DATE DESC;
```

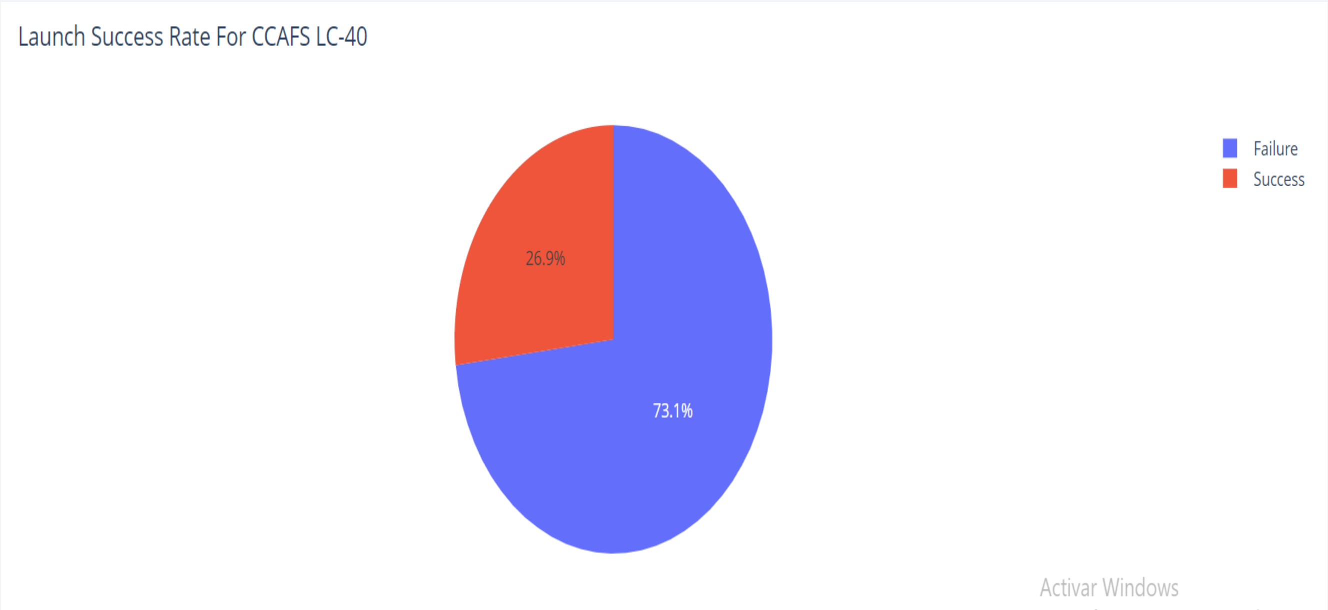
- https://github.com/Yarias2106/Cousera_IBM_Data_Science_Professional_Certificate/blob/2f89bc9a96b756c82a32a6fd4b986734c7dd1e53/5.%20Eda-SQL.ipynb

Build a Dashboard with Plotly Dash

We can see that the Kennedy Space Center Launch Complex 39A (KSC LC-39A) field was more successful with a success rate of 76.9% and a failure rate of 23.1%.

For its part, the Cape Canaveral Launch Complex 40 (CAFS LC-40) obtained a higher failure rate with 73.1 and only success of 26.9%.





https://github.com/Yarias2106/Cousera_IBM_Data_Science_Professional_Certificate/blob/d1d18bfd88938198272fc83c17d5c9ec4a87e754/7.%20Spacex_dashboard_app.py

Predictive Analysis (Classification)

To create the machine learning models, different machine learning algorithms were used, such as logistic regression, SVM, and Classification Trees.

1, The first thing we do is import the corresponding libraries and also obtain the data that we will use to train the models.

2. Rhen we split our data into training data and test data,

3. To obtain the best parameters we use GridSearchCV to train the models with different parameters and get the parameters that yield the best results.

4. Finally we evaluate the models with the scoring methods and the confusion matrix to know how well our models are working.

https://github.com/Yarias2106/Cousera_IBM_Data_Science_Professional_Certificate/blob/c78a139859f6eddf83b112620931651575fc3ec5/8.%20Machine%20Learning%20Prediction%20lab.ipynb

Results

- Interactive analytics demo in screenshots

```
data.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flight
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	

```
transform = preprocessing.StandardScaler()
```

```
X = transform.fit_transform(X)
X
```

```
array([[-1.71291154e+00, -1.94814463e-16, -6.53912840e-01, ...,
        -8.35531692e-01,  1.93309133e+00, -1.93309133e+00],
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size = 0.2, random_state = 2)
```

```
parameters ={'C':[0.01,0.1,1],
             'penalty':['l2'],
             'solver':['lbfgs']}
```

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
logreg_cv = GridSearchCV(lr,
                         param_grid = parameters,
                         cv=10)
logreg_cv.fit(X_train, Y_train)
```

```
GridSearchCV
estimator: LogisticRegression
    LogisticRegression
```

- Predictive analysis results

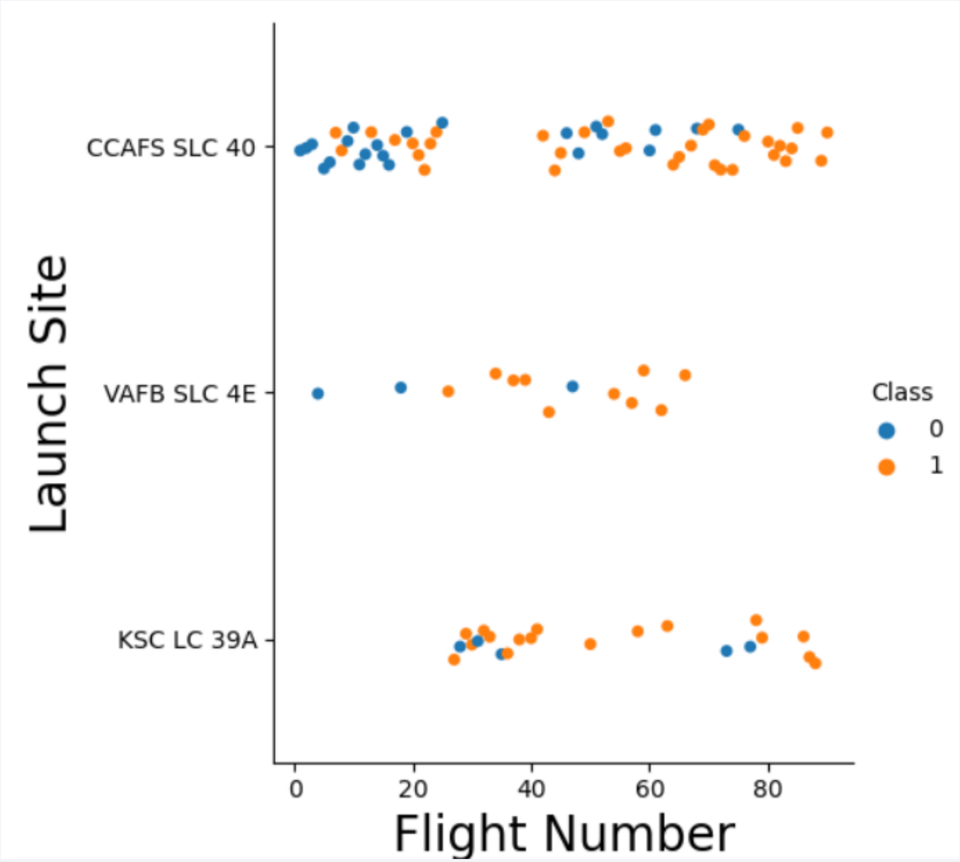
Scores on test data for each method

Logistic Regression: 0.83
SVM: 0.83
Decision Tree: 0.83
KNN: 0.83

Section 2

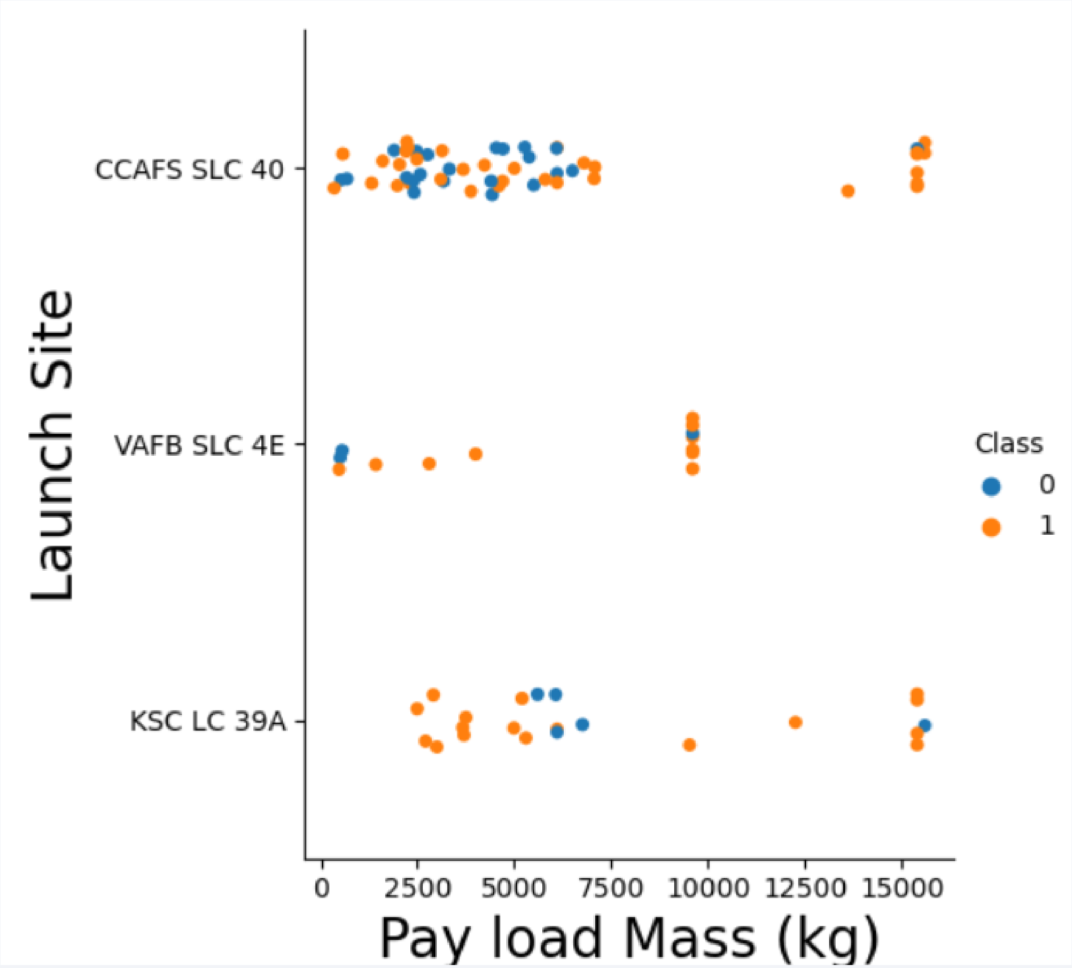
Insights drawn from EDA

Flight Number vs. Launch Site



We see that at the CCAFS SLC 40 launch site at first there is not much success and as the number of flights increases, the number of successful launches increases, a similar case to the VAFB SLC 4E launch site, KSC LC 39A does not seem to have a direct relationship with the number of flights and

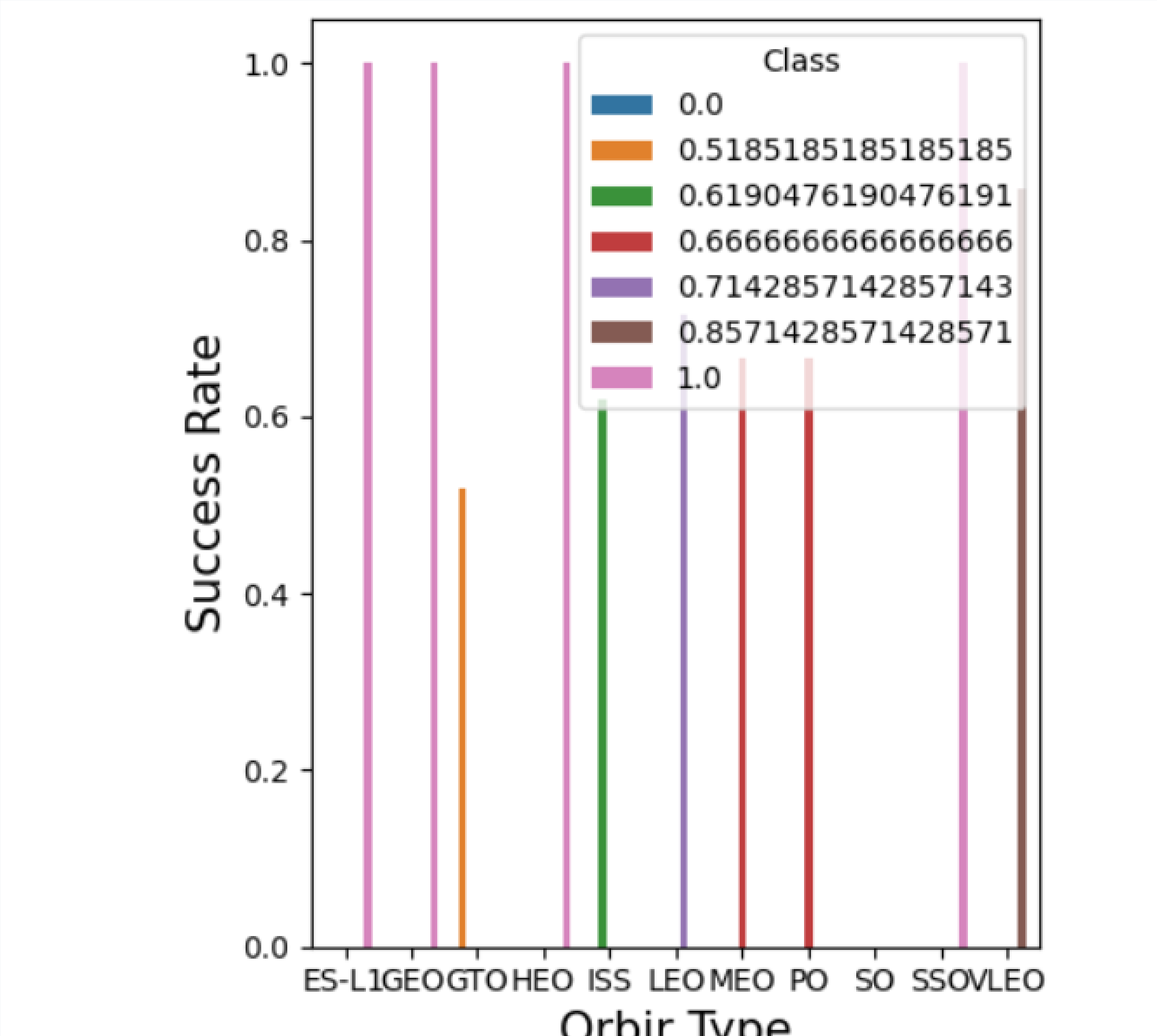
Payload vs. Launch Site



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

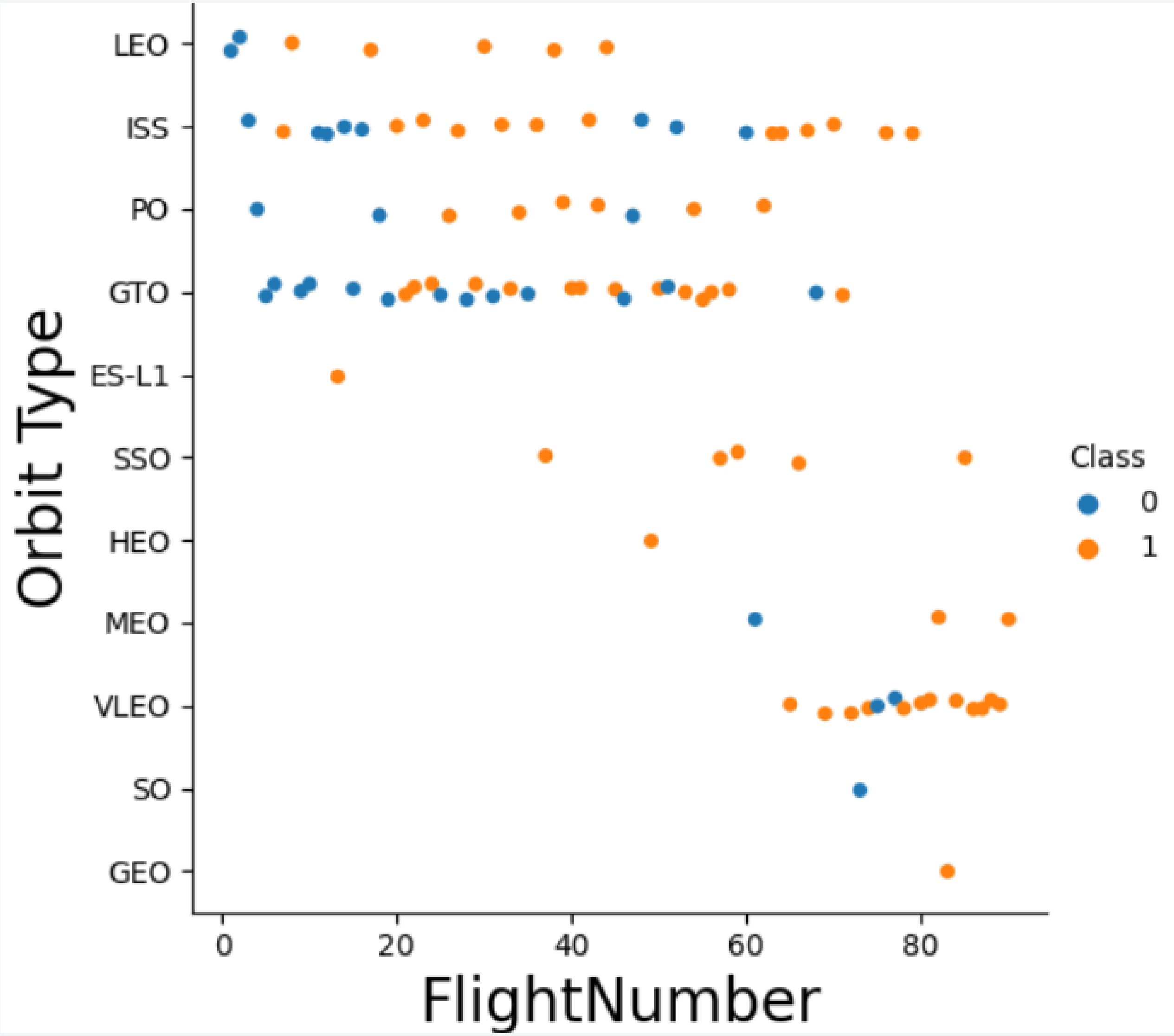
We can also see that it seems that the more payload mass there is, the success rate also increases.

Success Rate vs. Orbit Type



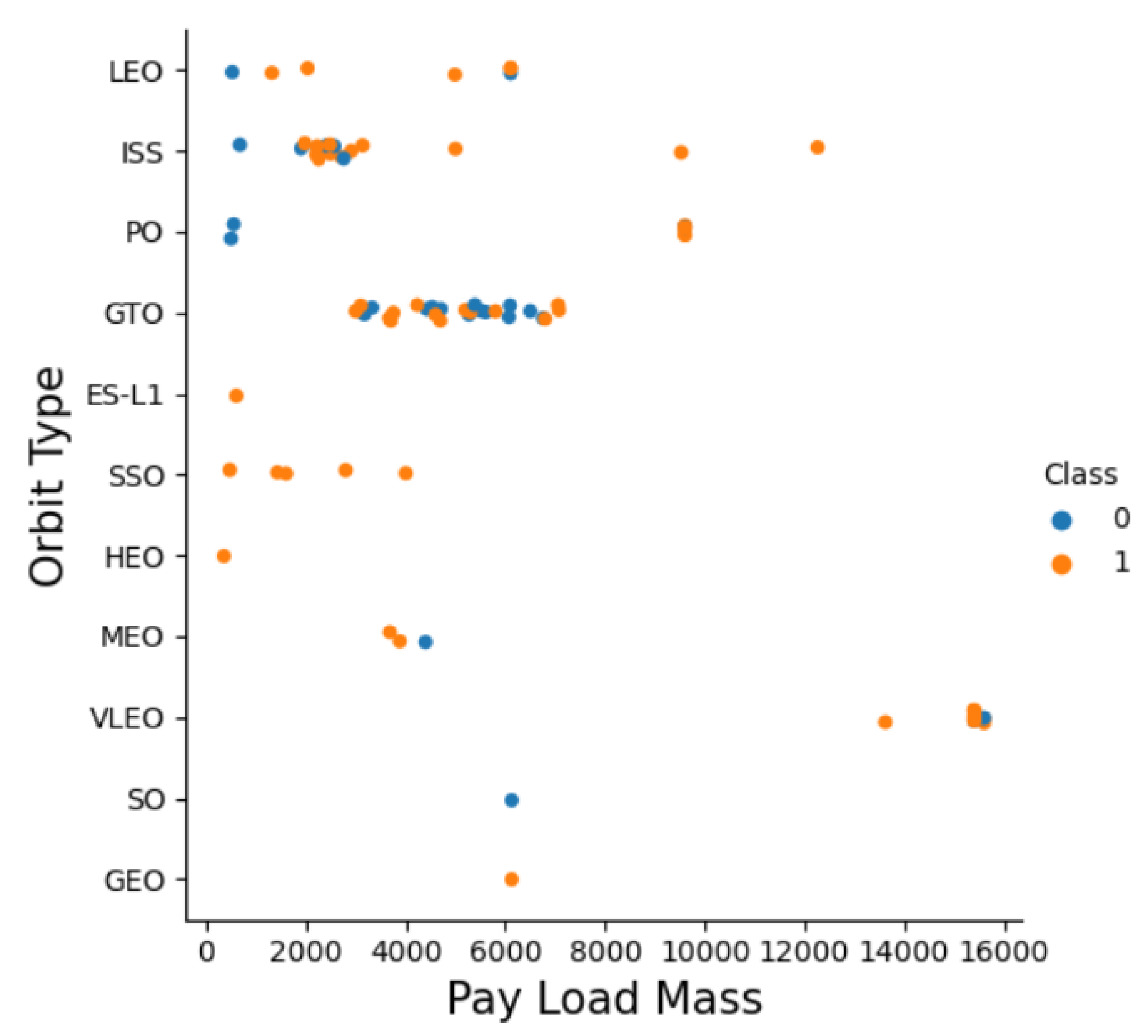
The most successful orbits are ES-L1, GEO, ISS and SSOV, the orbit with the lowest success rate with only 51% in GTO, the other orbits we can say have a decent success rate.

Flight Number vs. Orbit Type



We 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.

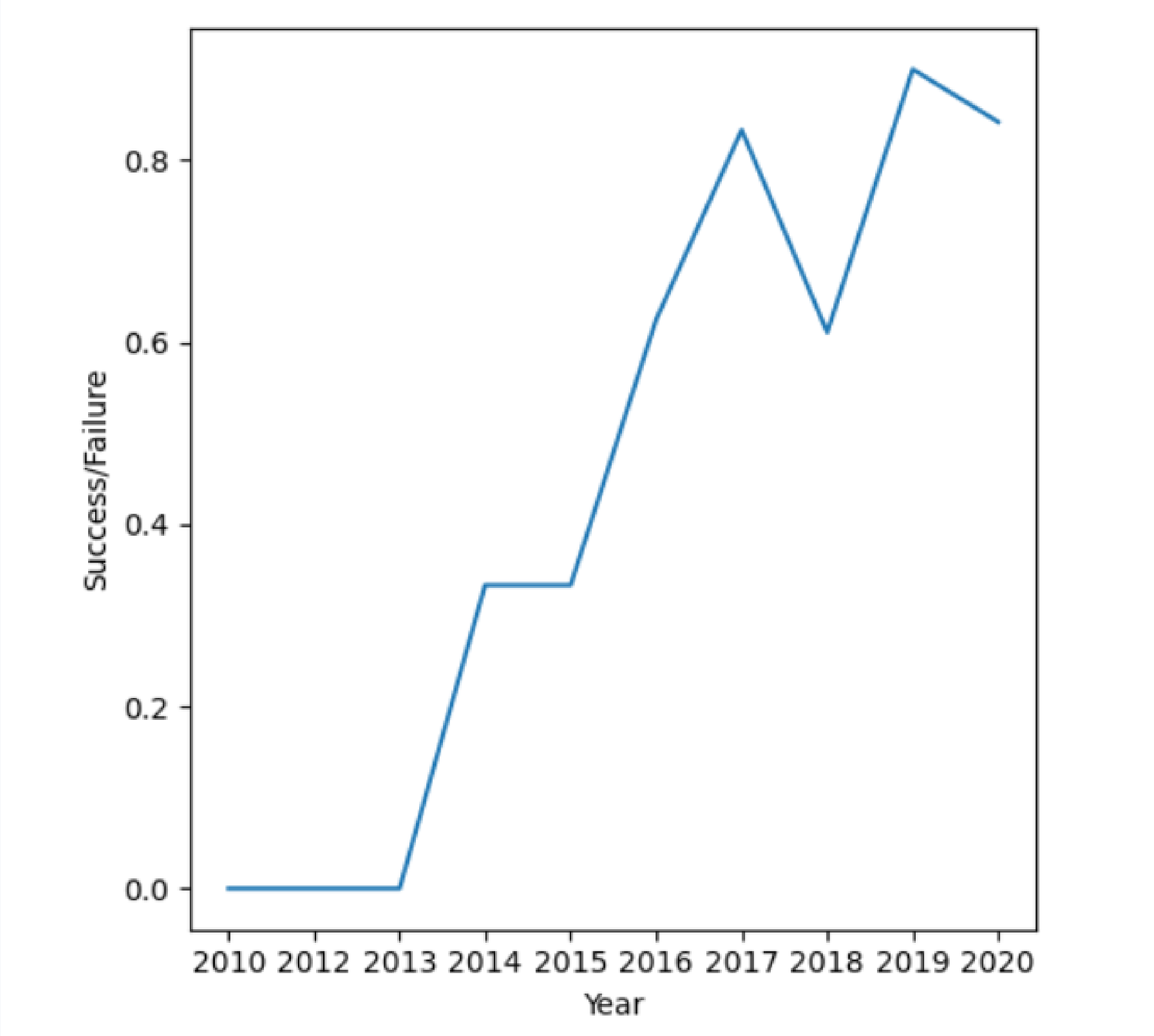
Payload vs. Orbit Type



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



We can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

Find the names of the unique launch sites

```
%sql Select distinct "launch_Site" from "SPACEXTBL"
```

Query result

Out[6]:

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

We use the function “distinct” to capture only uniques names

Launch Site Names Begin with 'CCA'

Find 5 records where launch sites begin with `CCA`

```
%sql Select * from "SPACEXTBL" where "launch_site" like "CCA%" limit 5
```

Query result

Out[21]:

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

The function “like” help us to filter sites begin with specific letters and “limit” to show only the count specific that we want

Total Payload Mass

Calculate the total payload carried by boosters from NASA

```
%sql select sum("PAYLOAD_MASS__KG_") from "SPACEXTBL" where "Customer" = "NASA (CRS)"
```

Query result

```
Out[8]:
```

sum("PAYLOAD_MASS__KG_")
45596

We use “sum” to calculate the sum total of the a column

Average Payload Mass by F9 v1.1

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

```
%sql select avg("PAYLOAD_MASS__KG_") from "SPACEXTBL"
```

Query result

avg("PAYLOAD_MASS__KG_")
6138.287128712871

We use “avg” to calculate automatically de average of a column

First Successful Ground Landing Date

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

```
%sql select min("Date") from "SPACEXTBL"  where "Mission_Outcome"= "Success"
```

Query result

min("Date")
01-03-2013

The fuction “min” help us to capture the less value of the a column

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

```
%sql select "Booster_Version"  from "SPACEXTBL" where "Landing _Outcome" = "Success (drone ship)" and "PAYLOAD_MASS__KG_" > 4000 and "PAYLOAD_MASS__KG
```

Query result

Out[36]:

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

We use “and” to concatenate different logical clauses in this greater and lesser case (">" , "<")

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

```
%sql select count(MISSION_OUTCOME) from SPACEXTBL GROUP BY MISSION_OUTCOME;
```

Query result

```
Done.
Out[38]:
```

count(MISSION_OUTCOME)
1
98
1
1

The fuction “count” show the amount of variables of a column

Boosters Carried Maximum Payload

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

```
%sql select BOOSTER_VERSION from SPACEXTBL where PAYLOAD_MASS__KG_=(select max(PAYLOAD_MASS__KG_) from SPACEXTBL)
```

Query result

Out[39]:

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

Here we use a subquery that help to find the maximun payload

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, 4, 2) ,MISSION_OUTCOME,BOOSTER_VERSION,LAUNCH_SITE from "SPACEXTBL" where substr(Date,7,4)='2015'
```

Query result

Out[48]:

substr(Date, 4, 2)	Mission_Outcome	Booster_Version	Launch_Site
01	Success	F9 v1.1 B1012	CCAFS LC-40
02	Success	F9 v1.1 B1013	CCAFS LC-40
03	Success	F9 v1.1 B1014	CCAFS LC-40
04	Success	F9 v1.1 B1015	CCAFS LC-40
04	Success	F9 v1.1 B1016	CCAFS LC-40
06	Failure (in flight)	F9 v1.1 B1018	CCAFS LC-40
12	Success	F9 FT B1019	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

```
%sql SELECT "Landing _Outcome" from SPACEXTBL WHERE "Date" between '04-06-2010 ' and '20-03-2017' ORDER BY DATE DESC;
```

Query result

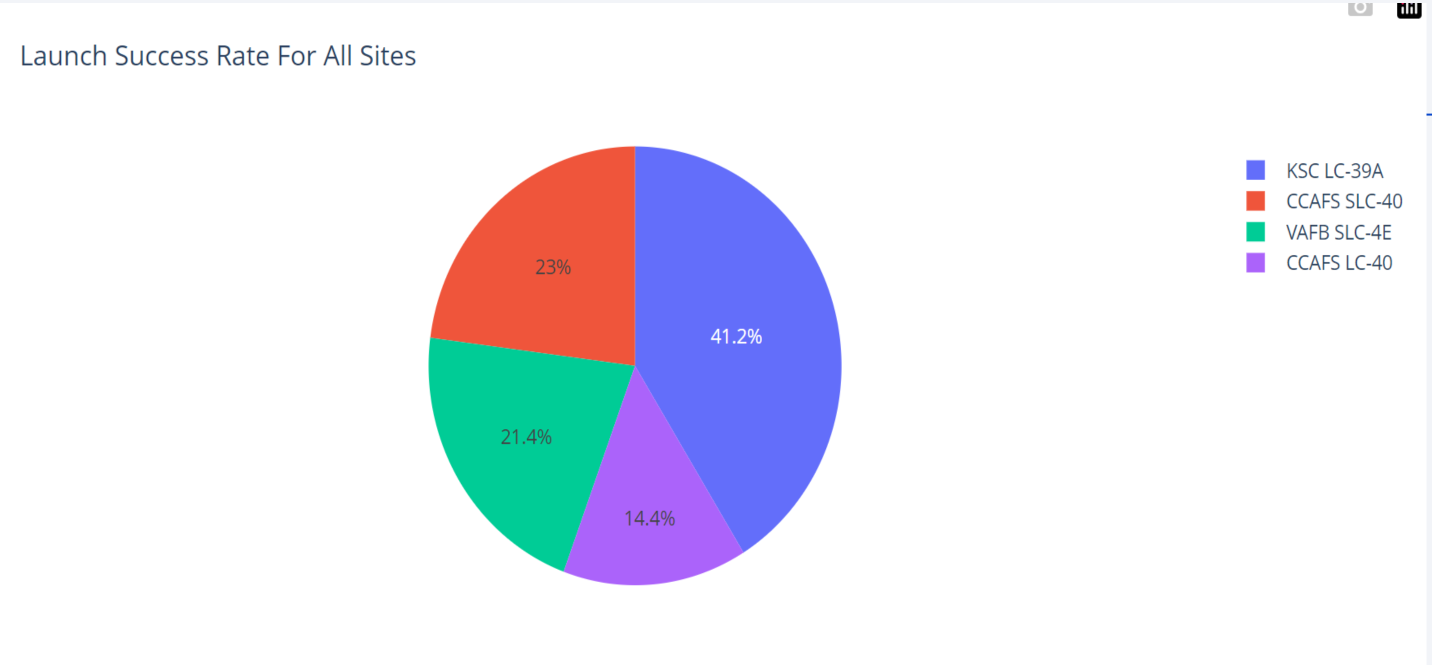
Out[58]:

Landing _Outcome
Success (ground pad)
No attempt
Success
Success
Success (ground pad)
Success (drone ship)
Controlled (ocean)
Failure
Success

Section 5

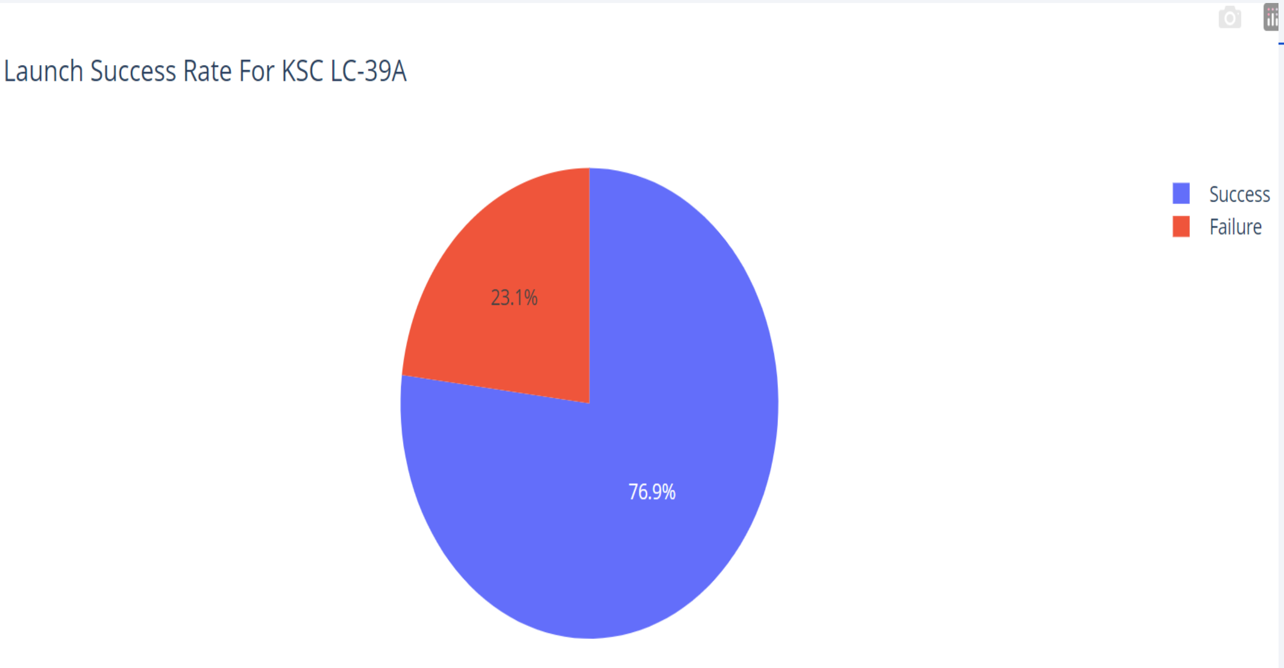
Build a Dashboard with Plotly Dash

Launch success count for all site



We observed that KSC LC-39A obtained the highest success rate with 41.2% the least was CCAFS-40 with only 14.4%

Piechart for the launch site with highest launch success ratio

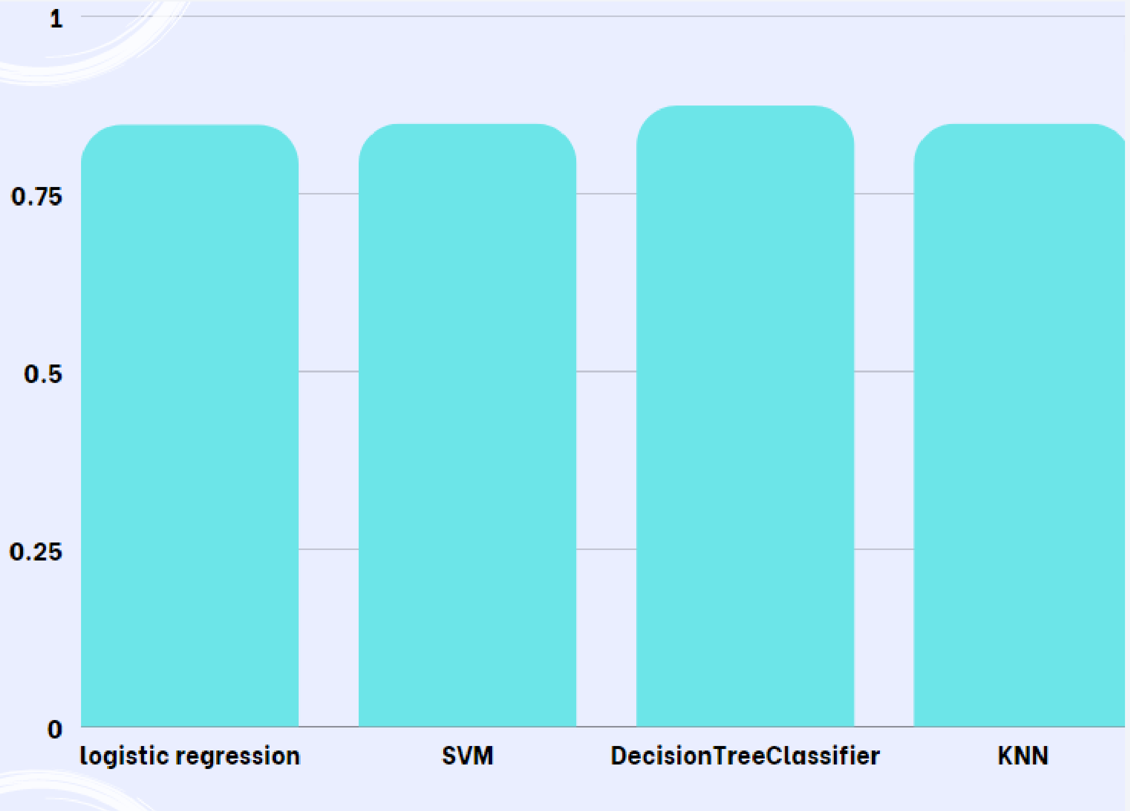


KSC LC-39A is the site that assures us a higher success rate we should choose this place more often for future launches

Section 6

Predictive Analysis (Classification)

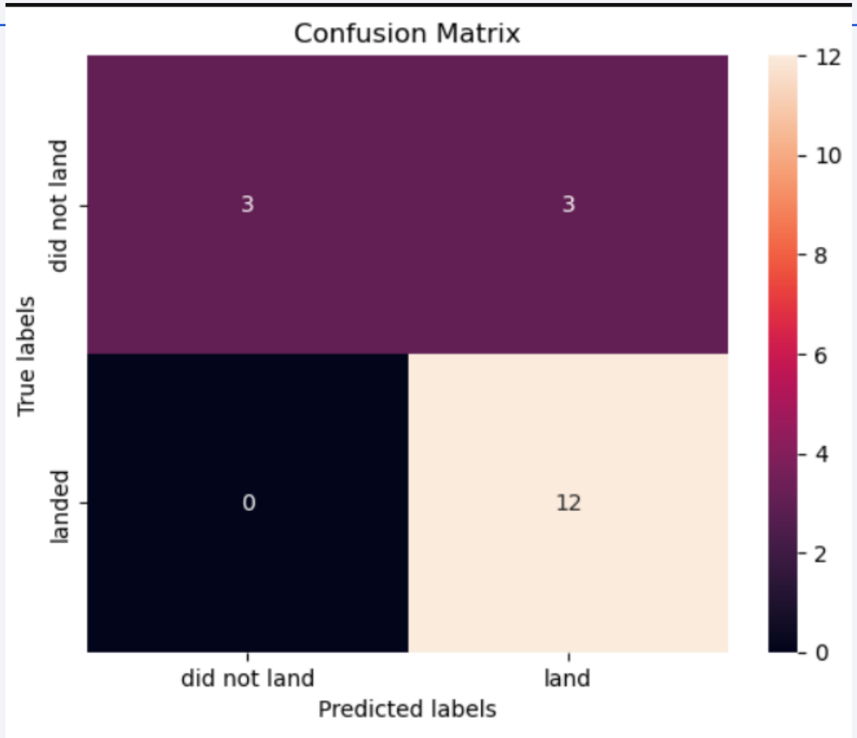
Classification Accuracy



logistic re	0.846
SVM	0.848
DecisionT	0.873
KNN	0.848

- The model that have the highest classification accuracy is DecisionTreeClassifier with 87,3%

Confusion Matrix



We have 15 correct predictions and 3 incorrect predictions with the model of machine learning

Conclusions

- All the prediction models obtained good scores, around 83% , highlighting the decision trees a little more with 87%
- KSC LC-39A is the site that assures us a higher success rate we should choose this place more often for future launches

Thank you!