Winning Space Race with Data Science

Bárbara Morais

03/10/2021

IBM Developer SKILLS NETWORK

OUTLINE

- Executive Summary
- Introduction
- Methodology
- Results
  - Visualization – Charts
  - Dashboard
- Discussion
  - Findings & Implications
- Conclusion

EXECUTIVE SUMMARY

- Methodology
  - Data Collection
  - Data Wrangling
  - EDA with Data Visualization
  - EDA with SQL
  - Building an interactive map with Follium
  - Building a Dashboard with Plotly Dash
  - Predictive Analysis (Classification)

- Results
  - Exploratory data analysis
  - Interactive analysis
  - Predictive analysis

INTRODUCTION

- Project background and context

  We predict if the Falcon 9 first stage will land successfully. SpaceX advertises Falcon 9 rocket lauches on its website, with a cost of 62 million dollars; while other provides 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.

- Problems that need solving
  - What influences if the rocket will land successfully?
  - The effect each relationshipwith certain rocket variables will impact in determining the success rate of a successful landing?
  - What conditions does SpaceX have to achieve to get the best results and ensure the best rocket success landing rate?

METHODOLOGY

- Data Collection
  - SpaceX API
  - WebScrapping from Wikipedia (using BeautifulSoup)
- Data Wrangling
  - Enconding data field for Machine Learning and dropping irrelevant columns
- EDA (visualization and SQL)
  - Scatter Graphs and Bar Graphs to show relationships between variables.
- Folium and Plotly Dash
  - Performed interactive visual analytics
- Classification Models
  - Performed predictive analysis

Data Collection - SpaceX API

Normalize data into flat
data file such as .csv

1 .Getting Response from API

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
response = requests.get(spacex_url)
```

2. Converting Response to a .json file

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

3. Apply custom functions to clean data

```
getLaunchSite(data)
getPayloadData(data)
getCoreData(data)
```

```
getBoosterVersion(data)
```

[GitHub URL to Notebook]

4. Assign list to dictionary then dataframe

```
launch_dict = {'FlightNumber': list(data['flight_number']),
'Date': list(data['date']),
'BoosterVersion':BoosterVersion,
'PayloadMass':PayloadMass,
'Orbit':Orbit,
'LaunchSite':LaunchSite,
'Outcome':Outcome,
'Flights':Flights,
'GridFins':GridFins,
'Reused':Reused,
'Legs':Legs,
'LandingPad':LandingPad,
'Block':Block,
'ReusedCount':ReusedCount,
'Serial':Serial,
'Longitude': Longitude,
'Latitude': Latitude}
```

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

5. Filter dataframe and export to flat file (.csv)

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Data Collection – Web Scrapping

1 .Getting Response from HTML

```
page = requests.get(static_url)
```

2. Creating BeautifulSoup Object

```
soup = BeautifulSoup(page.text, 'html.parser')
```

3. Finding tables

```
html_tables = soup.find_all('table')
```

4. Getting column names

```
column_names = []
temp = soup.find_all('th')
for x in range(len(temp)):
    try:
     name = extract_column_from_header(temp[x])
     if (name is not None and len(name) > 0):
        column_names.append(name)
    except:
     pass
```

5. Creation of dictionary

```
launch_dict= dict.fromkeys(column_names)

# Remove an irrelvant column
del launch_dict['Date and time ( )']


launch_dict['Flight No.'] = []
launch_dict['Launch site'] = []
launch_dict['Payload'] = []
launch_dict['Payload mass'] = []
launch_dict['Orbit'] = []
launch_dict['Customer'] = []
launch_dict['Launch outcome'] = []
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

Data Collection – Web Scrapping

6. Appending data to keys

```
extracted_row = 0
#Extract each table
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
```

7. Converting dictionary to dataframe

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

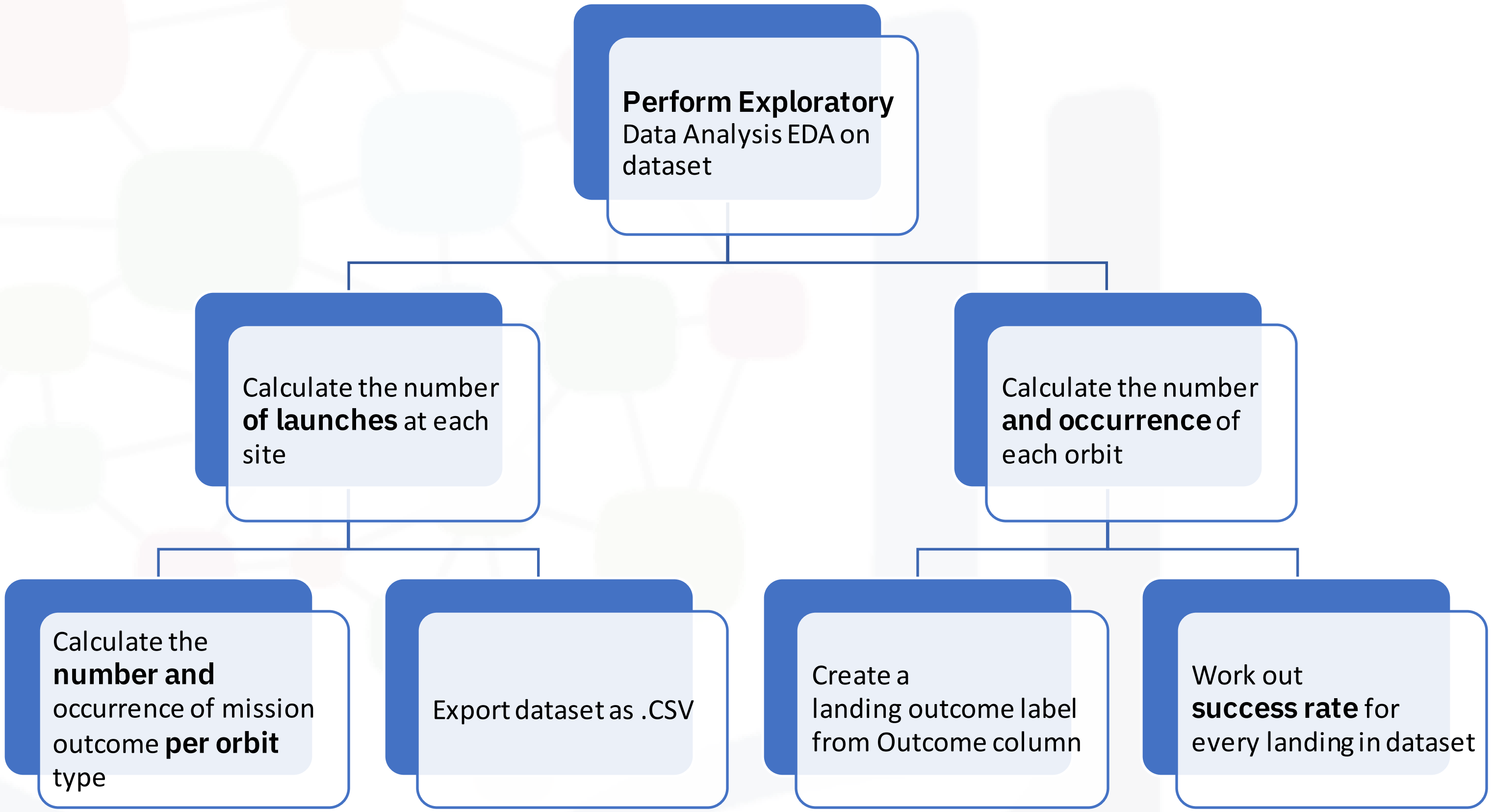
8. Dataframe to .CSV

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

GitHub URL to Notebook

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 and False ASDS means the mission outcome was unsuccessfully landed on a drone ship. We mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.



GitHub URL to Notebook

EDA with Data Visualization

Scatter Graphs being drawn:

1. Flight Number VS. Payload Mass
2. Flight Number VS. Launch Site
3. Payload VS. Launch Site
4. Orbit VS. Flight Number
5. Payload VS. Orbit Type
6. Orbit VS. Payload Mass

Scatter plots show how much one variable is affected by another. The relationship between two variables is called their correlation . Scatter plots usually consist of a large body of data.

Bar Graph being drawn: Mean VS. Orbit

A bar diagram makes it easy to compare sets of data between different groups at a glance.
The graph represents categories on one axis and a discrete value in the other. The goal is to show the relationship between the two axes. Bar charts can also show big changes in data over time.

Line Graph being drawn: Success Rate VS. Year

Line graphs are useful in that they show data variables and trends very clearly and can help to make predictions about the results of data not yet recorded.

GitHub URL to Notebook

EDA with SQL

Performed SQL queries to gather information about the dataset,

- Displaying the names of the unique launch sites in the space mission
- Displaying 5 records where launch sites begin with the string 'CCA'
- Displaying the total payload mass carried by boosters launched by NASA (CRS)
- Displaying average payload mass carried by booster version F9 v1.1
- Listing the date where the successful landing outcome in drone ship was achieved.
- Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
- Listing the total number of successful and failure mission outcomes
- Listing the names of the booster_versions which have carried the maximum payload mass.
- Listing the records which will display the successful landing_outcomes in ground pad, booster versions, launch_site for the months in year 2015
- Ranking the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.

[GitHub URL to Notebook]

Building an interactive map with Follium

- To visualize the Launch Data into an interactive map: We took the Latitude and Longitude Coordinates at each launch site, and added a Circle Marker around each launch site with a label of the name of the launch site.

- We assigned the dataframe launch_outcomes(failures, successes) to classes 0 and 1 with Green and Red markers on the map in a MarkerCluster()

- Using MousePosition we calculated the distance from the Launch Site to various landmarks to find various trends about what is around the Launch Site to measure patterns. Lines are drawn on the map to measure distance to landmarks.

- Some trends in which the Launch Site is situated in:

    - Are launch sites in close proximity to railways? No
    - Are launch sites in close proximity to highways? No
    - Are launch sites in close proximity to coastline? Yes
    - Do launch sites keep certain distance away from cities? Yes

GitHub URL to Notebook

Building a Dashboard with Plotly Dash

A. Pie Chart showing the total launches by a certain site/all sites
   i. - display relative proportions of multiple classes of data.
   ii. - size of the circle can be made proportional to the total quantity it represents.

B. Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions
   i. - It shows the relationship between two variables.
   ii. - It is the best method to show you a non linear pattern.
   iii. - The range of data flow, maximum and minimum value, can be determined.
   iv. - Observation and reading are straightforward.

GitHub Link to source code

Predictive Analysis (Classification)

BUILDING MODEL
- Load our dataset into NumPy and Pandas
- Transform Data
- Split our data into training and test data sets
- Check how many test samples we have
- Decide which type of machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our dataset.

GitHub URL to Notebook

EVALUATING MODEL
- Check accuracy for each model
- Get tuned hyperparameters for each type of algorithms
- Plot Confusion Matrix

IMPROVING MODEL
- Feature Engineering
- Algorithm Tuning

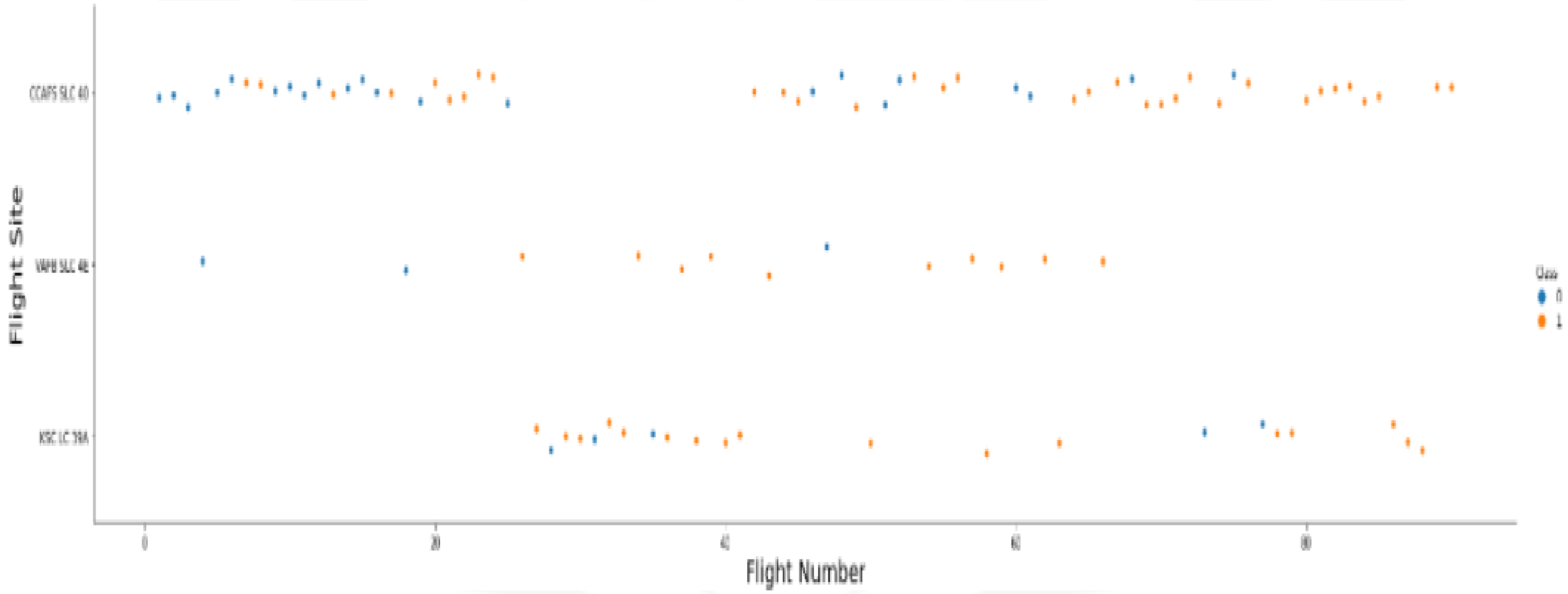
FINDING THE BEST PERFORMING CLASSIFICATION MODEL
- The model with the best accuracy score wins the best performing model
- In the notebook there is a dictionary of algorithms with scores at the bottom of the notebook.

RESULTS

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

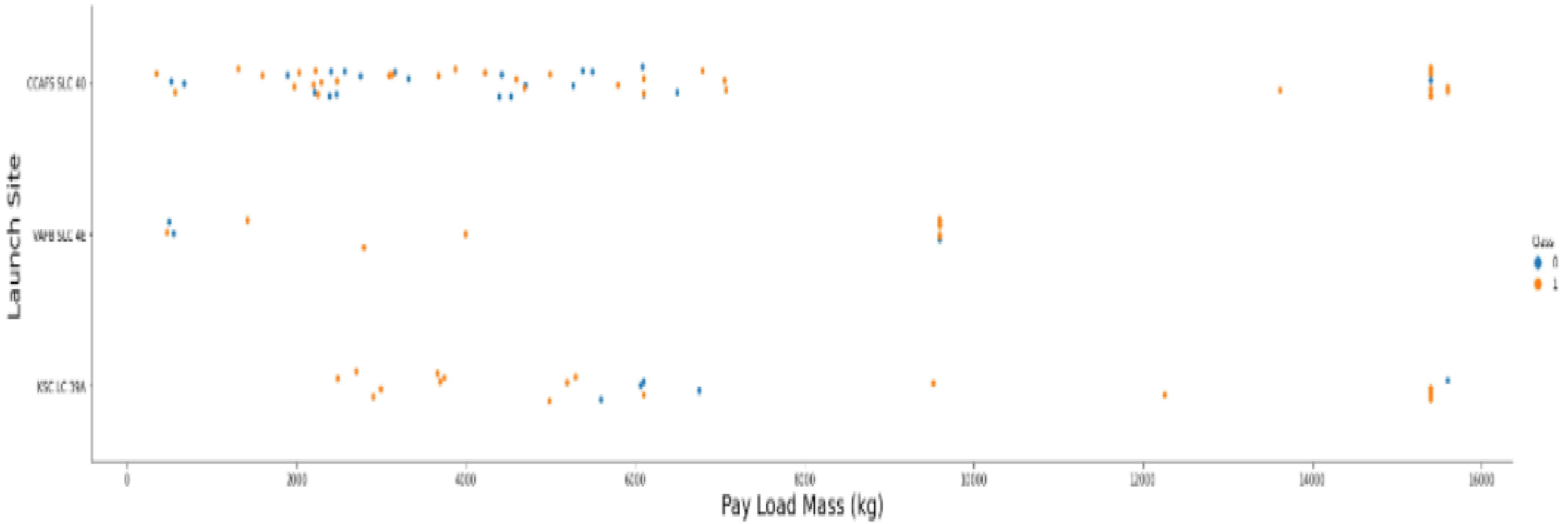
Flight Number vs. Launch Site

The more amount of flights at a launch site the greater the success rate at a launch site.



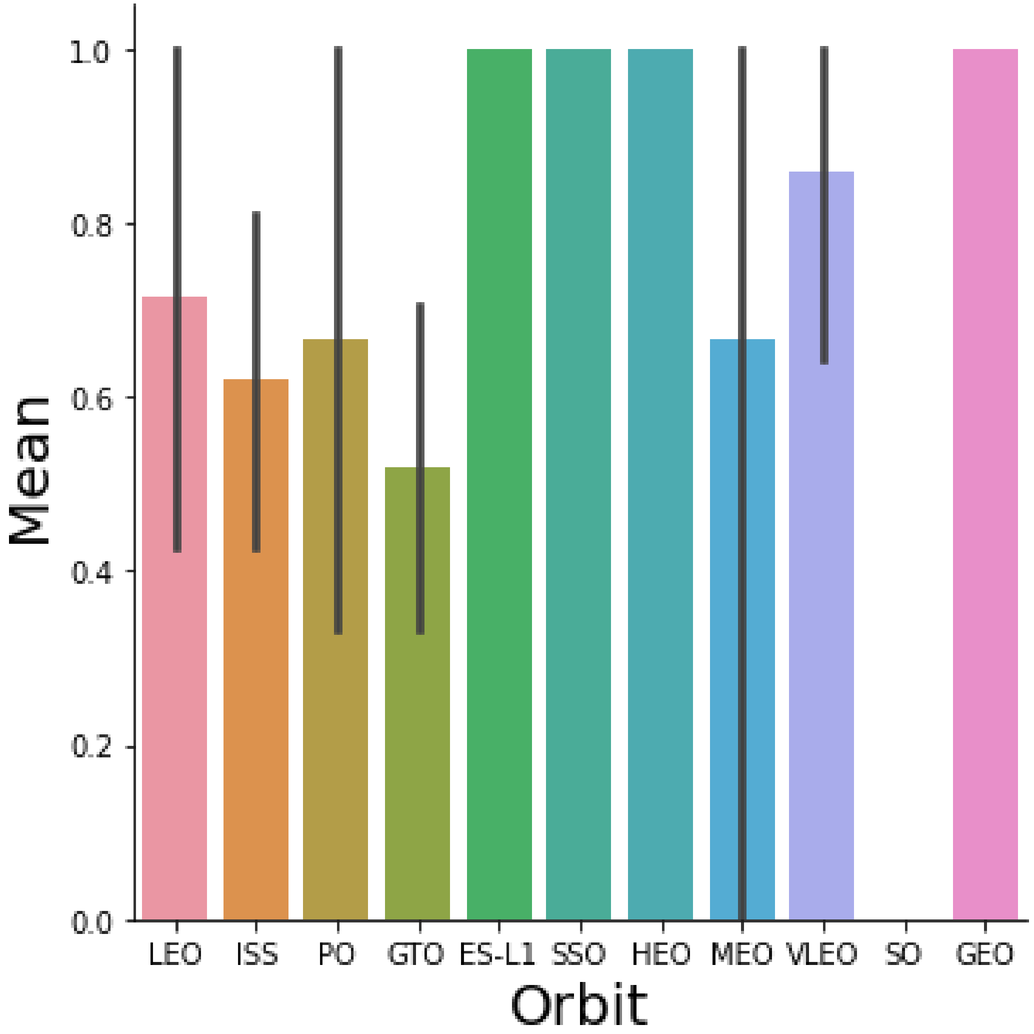
Payload vs. Launch Site

The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket. There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependant on Pay Load Mass for a success launch.



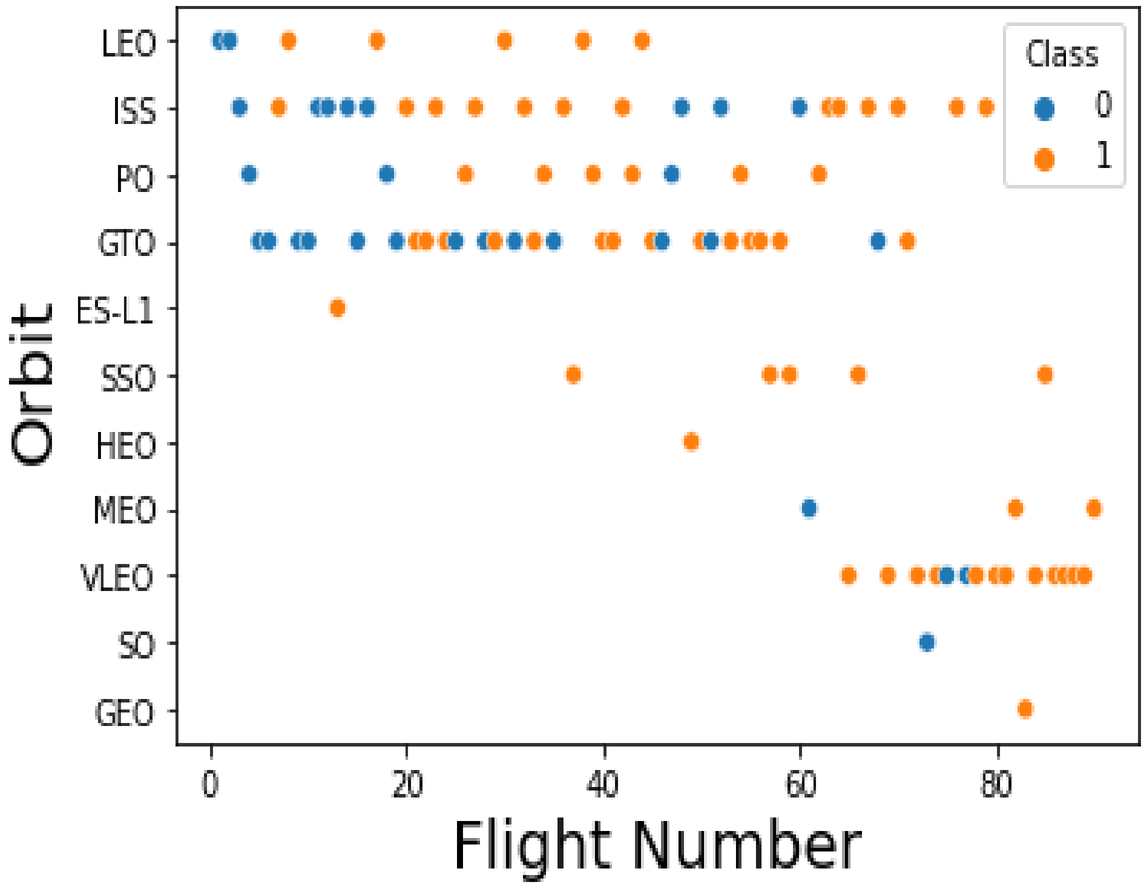
Success Rate vs. Orbit Type

Orbit GEO,HEO,SSO,ES-L1
has the best Success Rate



Flight Number vs. Orbit Type

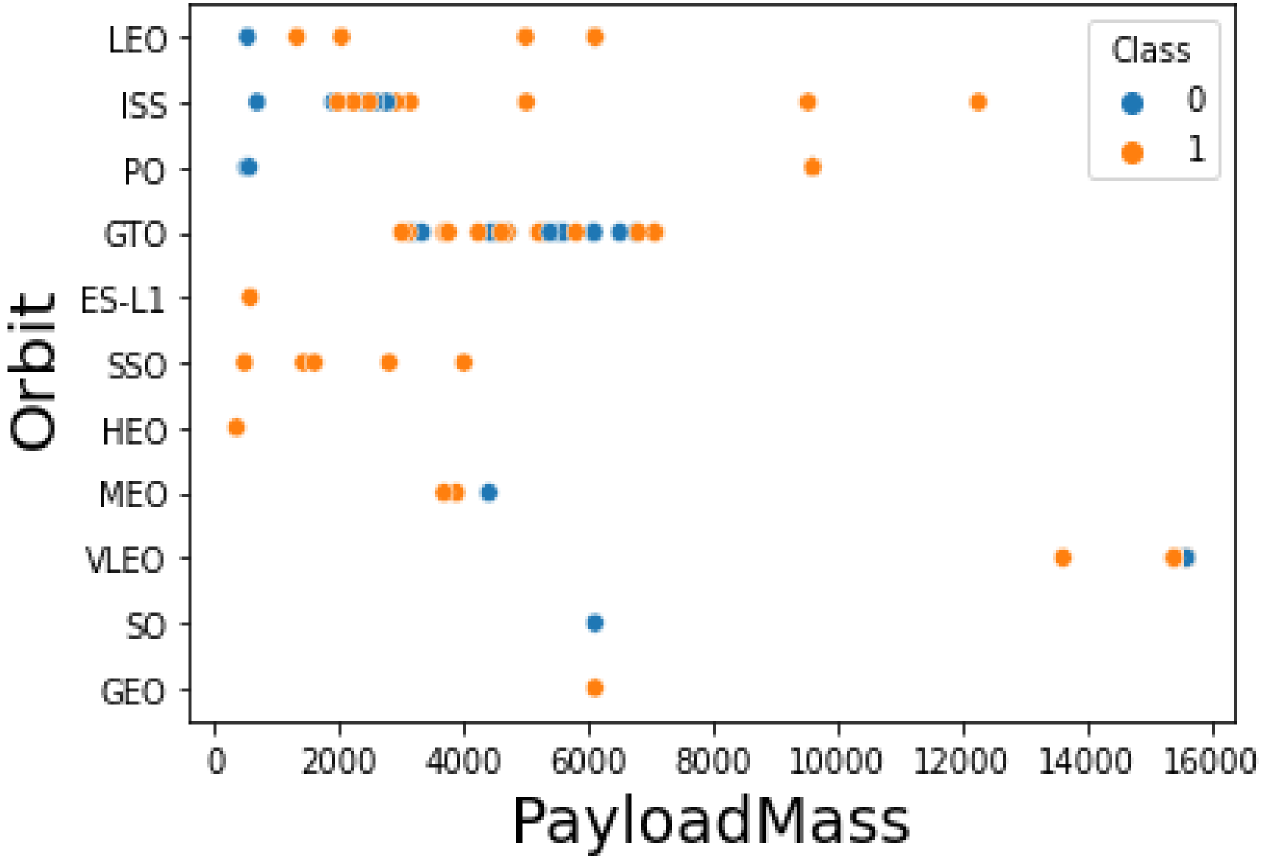
You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.



Payload vs. Orbit Type

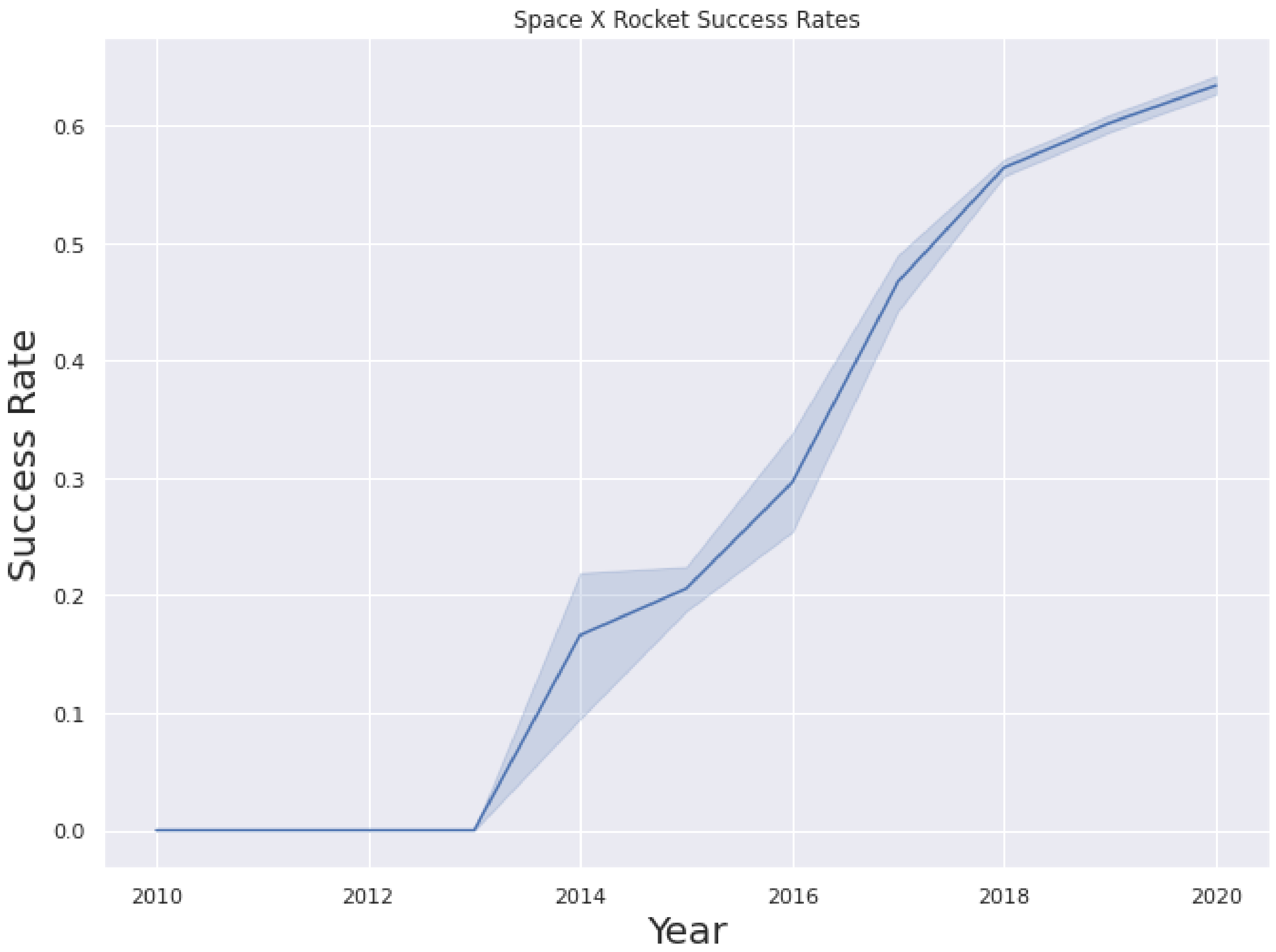
You should observe that Heavy payloads

have a negative influence on GTO orbits and posit
GTO and Polar LEO (ISS) orbits.



Launch Success Yearly Trend

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



IBM Developer

SKILLS NETWORK

All Launch Site Names

Using the word DISTINCT in the query means that it will only show Unique values in the Launch_Site column from SpaceX.

Unique Launch Sites
CCAFS LC-40
CCAFS SLC-40
CCAFS SLC-40
KSC LC-39A
VAFB SLC-4E

Launch Site Names Begin with 'CCA'

Using the word LIMIT 5 in the query means that it will only show 5 records from SpaceX and LIKE keyword has a wild card with the words "CCA%" (the percentage in the end suggests that the Launch_Site name must start with CCA).

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	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	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
2012-10-08	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- Using the function SUM summates the total in the column PAYLOAD_MASS_KG_
- The WHERE clause filters the dataset to only perform calculations on Customer NASA (CRS)

Total Payload Mass
45596

Average Payload Mass by F9 v1.1

Using the function AVG works out the average in the column PAYLOAD_MASS_KG_

The WHERE clause filters the dataset to only perform calculations on Booster_version F9 v1.1

Average Payload Mass
2928

First Successful Ground Landing Date

Using the function MIN works out the minimum date in the column Date

The WHERE clause filters the dataset to only perform

calculations on Landing_Outcome = Success (ground pad)

Date which first Successful landing outcome in ground pad was acheived
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

Selecting only Booster_Version

The WHERE clause filters the dataset to Landing_Outcome = Success (drone ship)

The AND clause specifies additional filter conditions Payload_MASS_KG 4000 AND Payload_MASS_KG 6000

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

Total Number of Successful and Failure Mission Outcomes

Using the function COUNT works out the amount

The WHERE clause filters the dataset to only perform calculations on Mission_Outcome

(Success or Failure)

Successful_Mission_Outcomes	Failure_Mission_Outcomes
100	1

Boosters Carried Maximum Payload

Using the word SELECT in the query means that it will show values in the Booster_Version column from SpaceX

The WHERE clause filters the dataset to
Payload_Mass__Kg_

And the MAX in the subquery
Payload_Mass__Kg_ means its arranging the dataset into the maximum values

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

2015 Launch Records

The WHERE clause filters the dataset to landing_outcome, and LIKE to filter the Failure.

DATE LIKE puts the value of 2015

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

Function WHERE filters landing__outcome and LIKE (Success or Failure; AND (DATE between)

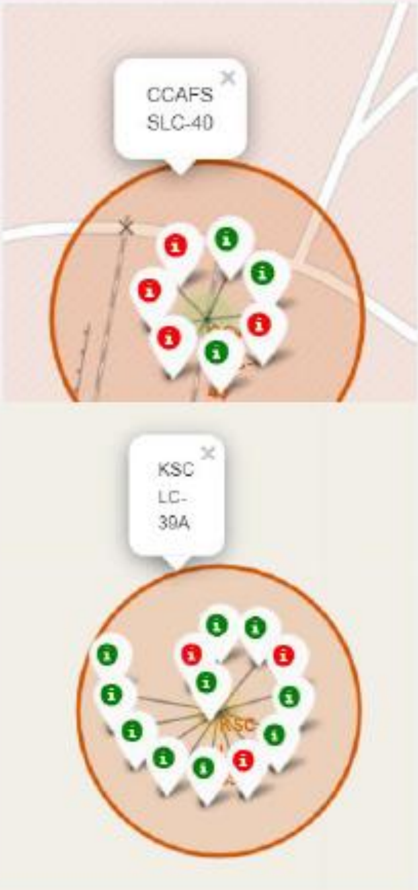
DESC means its arranging the dataset into descending order

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2017-02-19	14:39:00	F9 FT B1031.1	KSC LC-39A	SpaceX CRS-10	2490	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2017-01-14	17:54:00	F9 FT B1029.1	VAFB SLC-4E	Iridium NEXT 1	9600	Polar LEO	Iridium Communications	Success	Success (drone ship)
2016-08-14	05:26:00	F9 FT B1026	CCAFS LC-40	JCSAT-16	4600	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-07-18	04:45:00	F9 FT B1025.1	CCAFS LC-40	SpaceX CRS-9	2257	LEO (ISS)	NASA (CRS)	Success	Success (ground pad)
2016-05-27	21:39:00	F9 FT B1023.1	CCAFS LC-40	Thaicom 8	3100	GTO	Thaicom	Success	Success (drone ship)
2016-05-06	05:21:00	F9 FT B1022	CCAFS LC-40	JCSAT-14	4696	GTO	SKY Perfect JSAT Group	Success	Success (drone ship)
2016-04-08	20:43:00	F9 FT B1021.1	CCAFS LC-40	SpaceX CRS-8	3136	LEO (ISS)	NASA (CRS)	Success	Success (drone ship)
2015-12-22	01:29:00	F9 FT B1019	CCAFS LC-40	OG2 Mission 2 11 Orbcomm-OG2 satellites	2034	LEO	Orbcomm	Success	Success (ground pad)

DATE	time__utc_	booster_version	launch_site	payload	payload_mass__kg_	orbit	customer	mission_outcome	landing__outcome
2016-06-15	14:29:00	F9 FT B1024	CCAFS LC-40	ABS-2A Eutelsat 117 West B	3600	GTO	ABS Eutelsat	Success	Failure (drone ship)
2016-03-04	23:35:00	F9 FT B1020	CCAFS LC-40	SES-9	5271	GTO	SES	Success	Failure (drone ship)
2016-01-17	18:42:00	F9 v1.1 B1017	VAFB SLC-4E	Jason-3	553	LEO	NASA (LSP) NOAA CNES	Success	Failure (drone ship)
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)
2015-01-10	09:47:00	F9 v1.1 B1012	CCAFS LC-40	SpaceX CRS-5	2395	LEO (ISS)	NASA (CRS)	Success	Failure (drone ship)
2010-12-08	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)
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)

All launch sites' location markers on a global map

Color-labeled launch outcomes

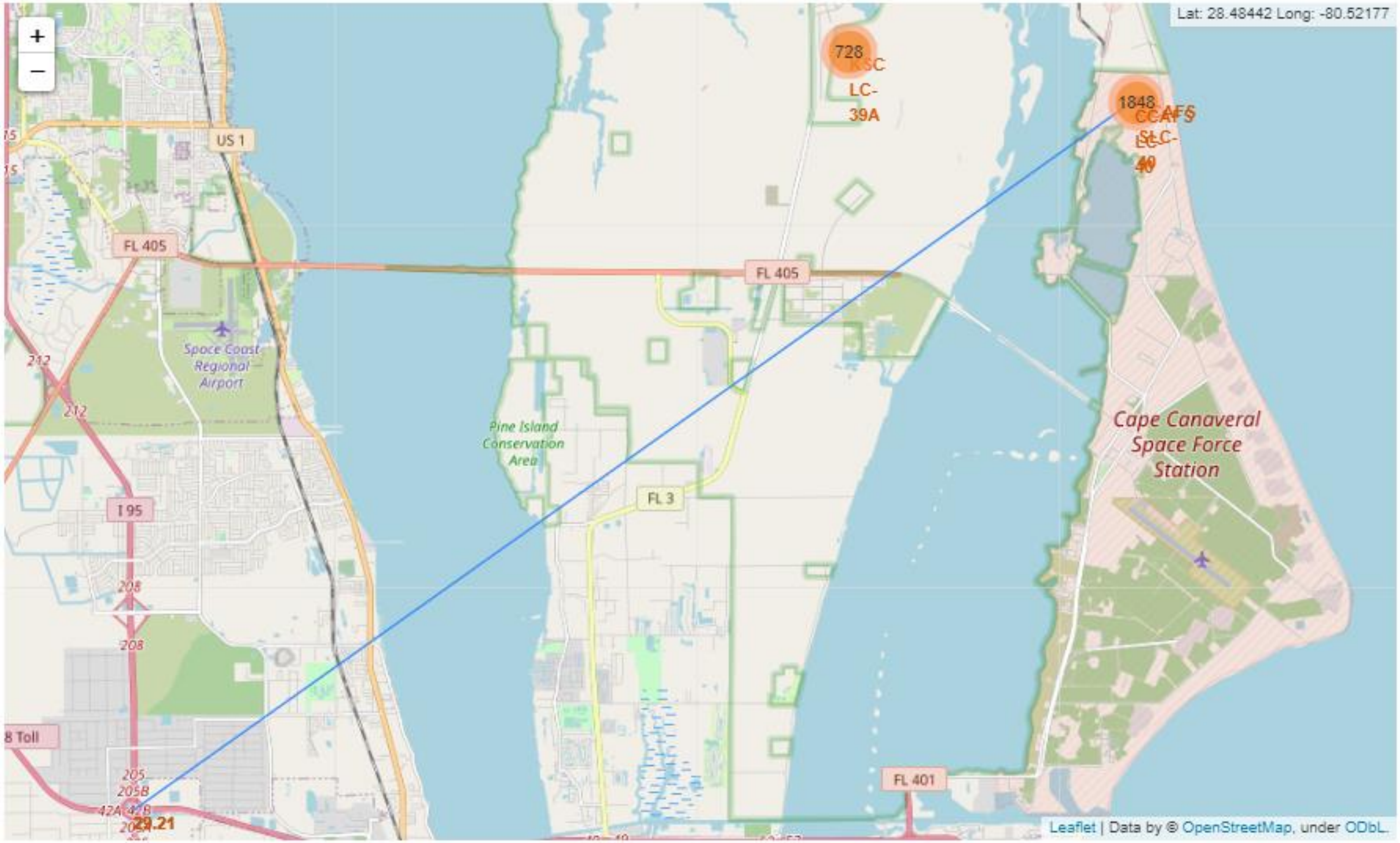


Florida Launch Sites

Green Marker shows successful Launches and Red Marker shows Failures

California Launch Site

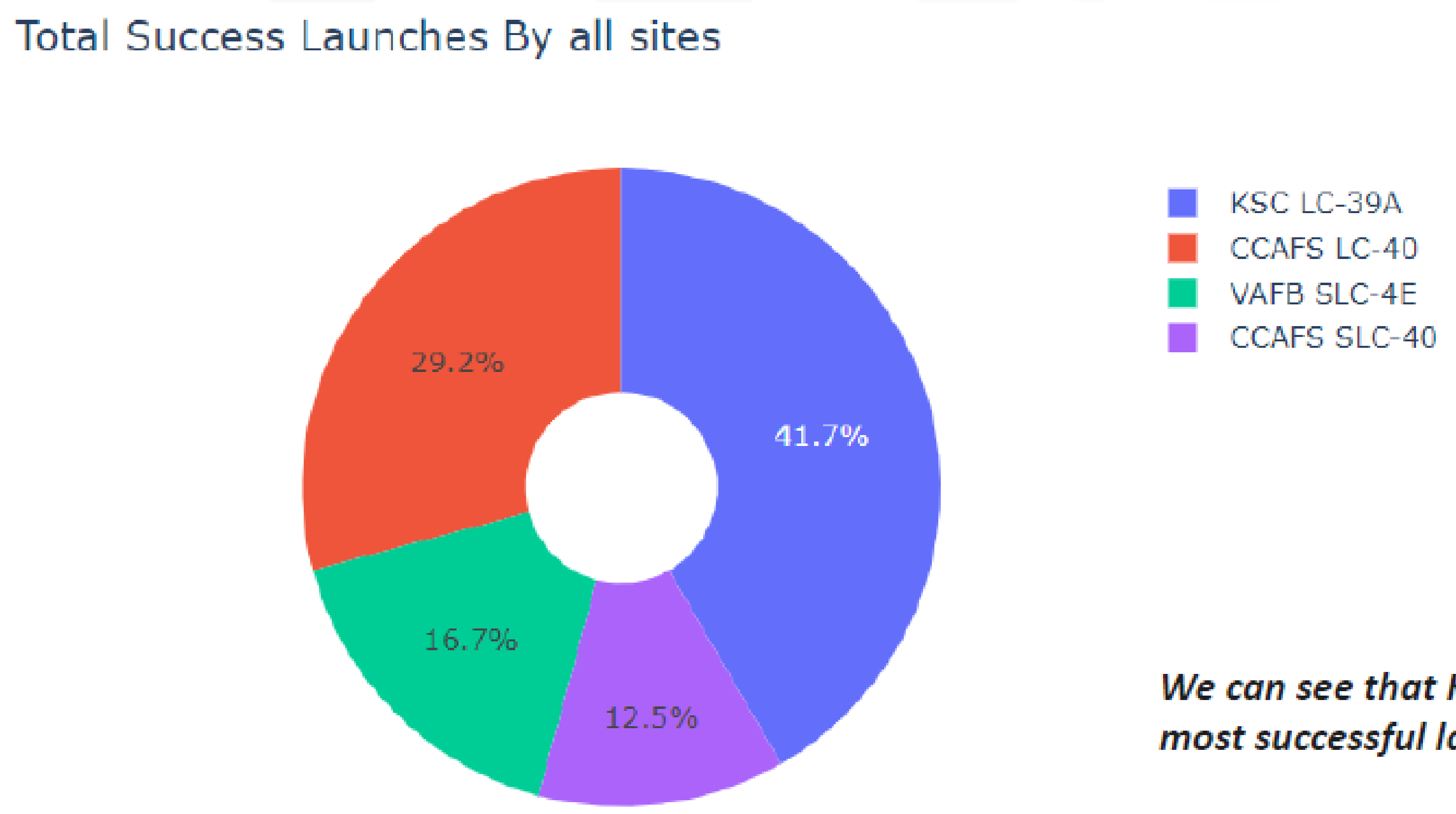
Selected launch site to railway



IBM Developer

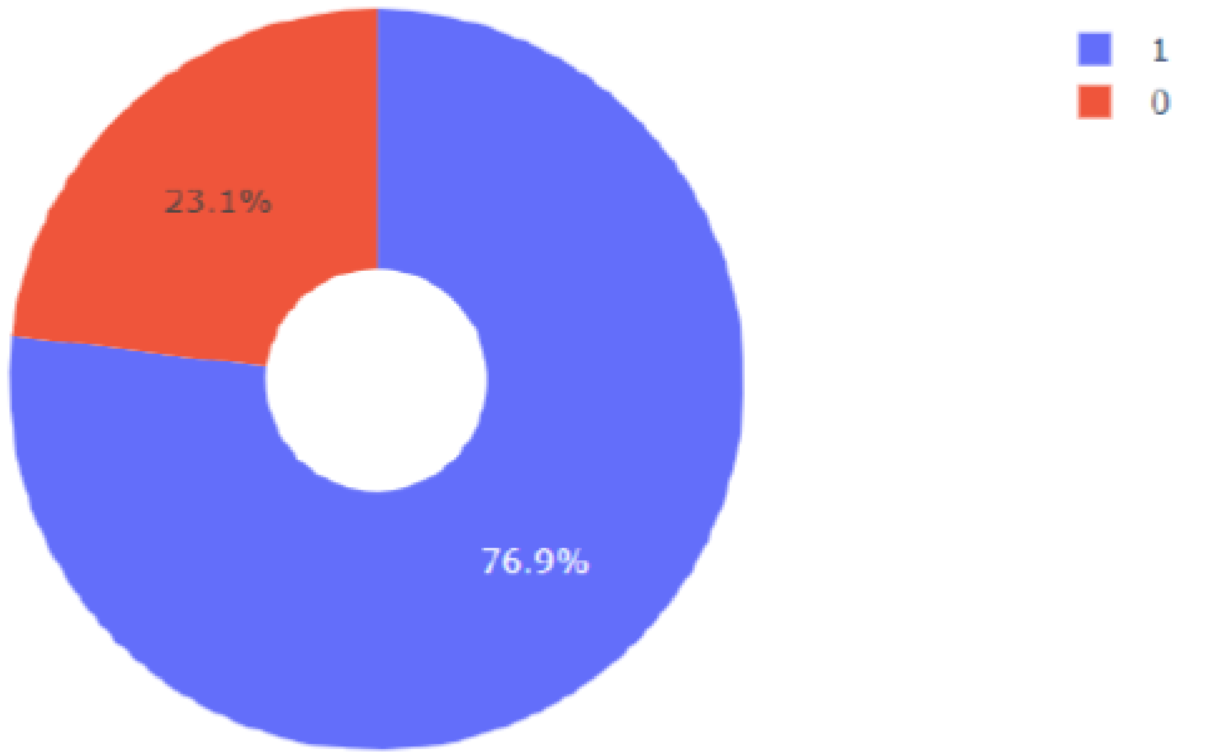
SKILLS NETWORK

Launch success count for all sites, in a piechart



We can see that KSC LC-39A had the most successful launches from all the sites

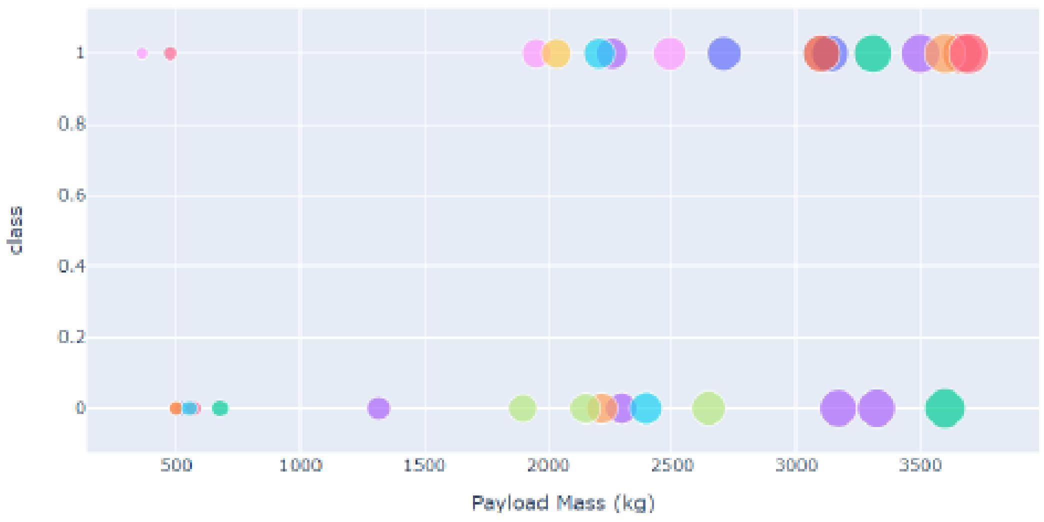
Piechart for the launch site with highest launch success ratio



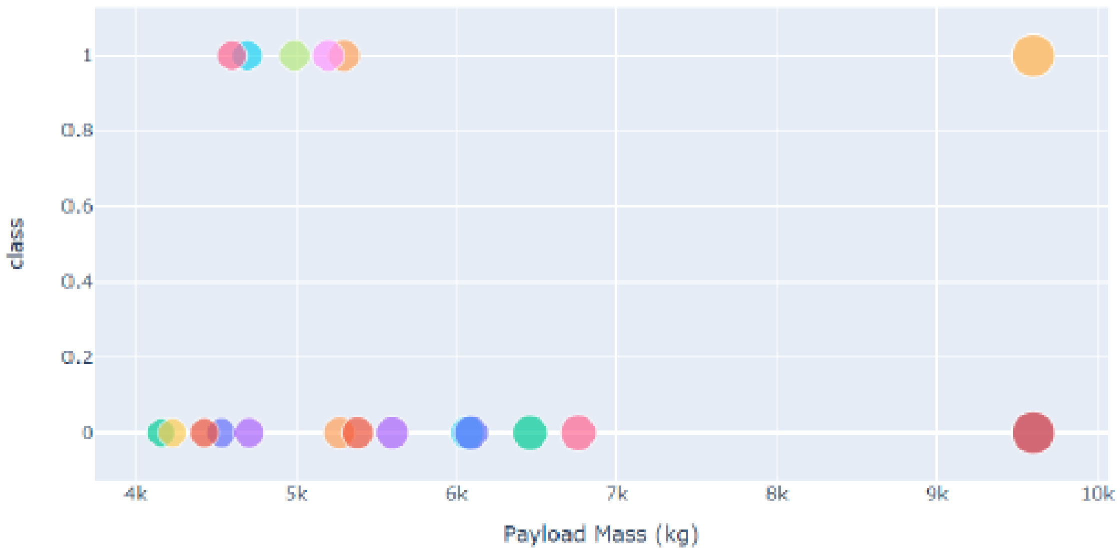
KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate

Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

Low Weighted Payload 0kg – 4000kg



Heavy Weighted Payload 4000kg – 10000kg



We can see the success rates for low weighted payloads is higher than the heavy weighted payloads

Classification Accuracy

As you can see our accuracy is extremely close but we do have a winner its down to decimal places! using this function

```
import operator
methods={'logistic regression': logreg_cv.score(X_test, Y_test), 'svm': svm_cv.score(X_test, Y_test), 'tree': tree_cv.score(X_te
st, Y_test), 'knn': knn_cv.score(X_test, Y_test)}
print('Method perfoms best: ', max(methods.items(), key=operator.itemgetter(1))[0])
```

```
Method perfoms best:  logistic regression
```

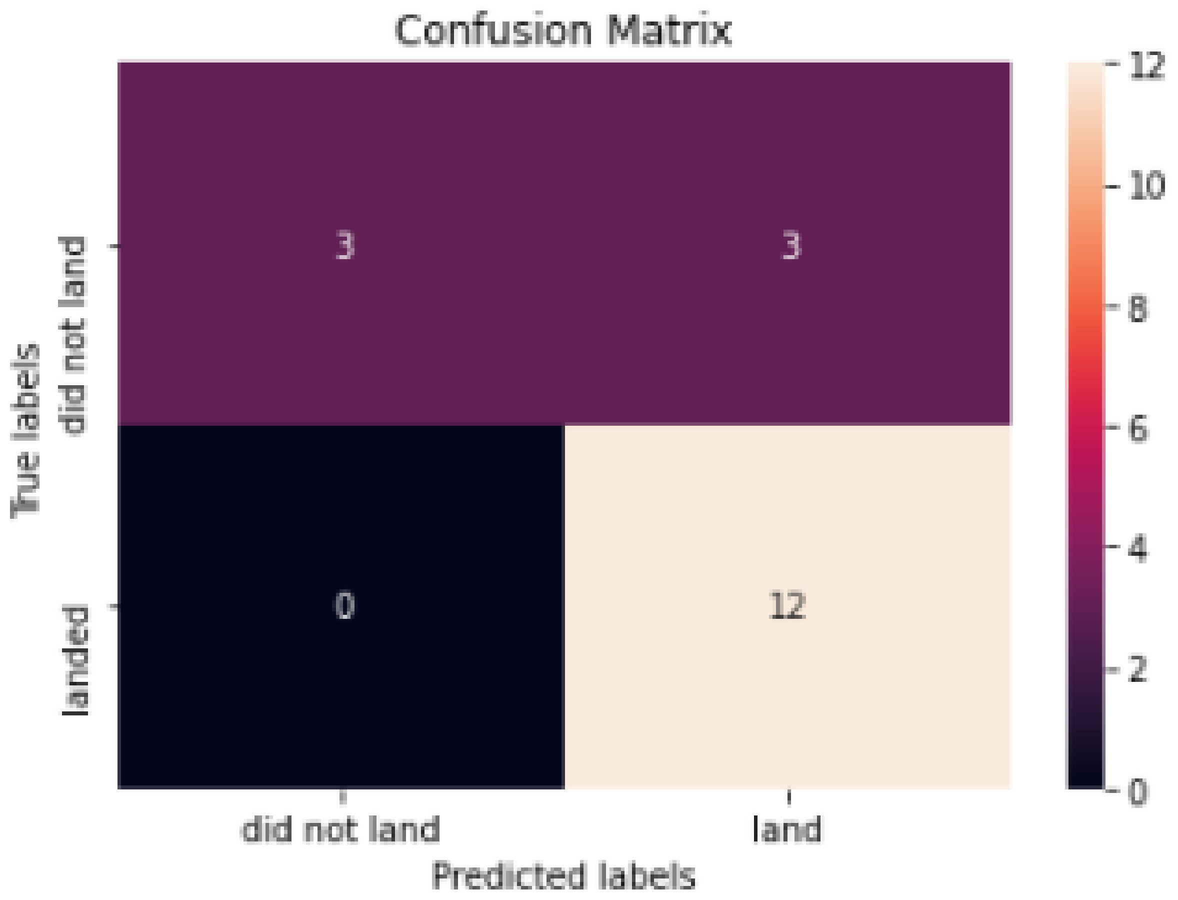
After selecting the best hyperparameters for the classifier using the validation data, we achieved 83.33% accuracy on the test data.

```
In [13]: logreg_cv.score(X_test, Y_test)

Out[13]: 0.8333333333333334
```

Confusion Matrix

Examining the confusion matrix, we see that Logistic Regression can distinguish between the different classes.



CONCLUSION

- The Logistic Regression is the best for Machine Learning for this dataset
- Low weighted payloads perform better than the heavier payloads
- The success rates for SpaceX launches is directly proportional time in years they will eventually perfect the launches
- We can see that KSC LC 39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES L1 has the best Success Rate

Thank you!