IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Ahmed Mohamed
21-May-2022

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

Methodologies

- Data Collection
  - Through API
  - with Web Scraping
- Data Wrangling
- Exploratory Data Analysis
  - With SQL
  - with Data Visualization
- Interactive Visual Analytics
  - With Folium
  - With Plotly Dash
- Machine Learning Prediction

Summary of all results and EDA

- Exploratory Data Analysis result
- Interactive analytics in screenshots
- Predictive Analytics result

Introduction

Project background and context

- Space X 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 Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

Problems you want to find answers

- Reasons of a successful landing?
- Connection between features that makes a landing successful?
- How to increase the success rate of the rocket?

GitHub Repo

https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/tree/master

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

- Data was collected using two methods:

- Using the official SpaceX API

  Data was collected by requesting it from the API and receiving a JSON response

  The JSON is then converted into a Pandas DataFrame using `.json_normalize()` method.

- And by using BeautifulSoup for web scraping of a Wikipedia page

  The HTML table that had the data needed was scraped and extracted

  Then the HTML table was converted into a Pandas DataFrame and saved as a .csv

Data Collection – SpaceX API

- The SpaceX RESTful API was used to collect, clean the data and perform some wrangling to it.
- https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/Data%20Collection%20with%20API.ipynb

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

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

Check the content of the response

```
print(response.text[0:1000])
```


```
[{"fairings":{"reused":false,"recovery_attempt":false,"recovered":false,"ships":[]},"links":{"patc
h":{"small":"https://images2.imgbox.com/3c/0e/T8iJcSN3_o.png","large":"https://images2.imgbox.com/
40/e3/GypSkayF_o.png"},"reddit":{"campaign":null,"launch":null,"media":null,"recovery":null},"flic
kr":{"small":[],"original":[]},"presskit":null,"webcast":"https://www.youtube.com/watch?v=0a_00nJ_
Y88","youtube_id":"0a_00nJ_Y88","article":"https://www.space.com/2196-spacex-inaugural-falcon-1-ro
```


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

```
data
```

	static_fire_date_utc	static_fire_date_unix	net	window	rocket	success	failures	deta
0	2006-03-17T00:00:00.000Z	1.142554e+09	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 33, 'altitude': None, 'reason': 'merlin engine failure'}]	Eng failure at secor and loss vehi
1	None	NaN	False	0.0	5e9d0d95eda69955f709d1eb	False	[{'time': 301, 'altitude': 289, 'reason': 'harmonic oscillation	Success first sta burn a transition seco sta maxim altitude Premat

```
# Hint data['BoosterVersion']!='Falcon 1'
data_falcon9 = launch_df[(lambda x : x.BoosterVersion == 'Falcon 9')]
```

```
data_falcon9.info
```

```
<bound method DataFrame.info of     FlightNumber        Date BoosterVersion  PayloadMass Orbit
LaunchSite  \
4              6  2010-06-04       Falcon 9          NaN   LEO  CCSFS SLC 40
5              8  2012-05-22       Falcon 9        525.0   LEO  CCSFS SLC 40
6             10  2013-03-01       Falcon 9        677.0   ISS  CCSFS SLC 40
7             11  2013-09-29       Falcon 9        500.0    PO   VAFB SLC 4E
8             12  2013-12-03       Falcon 9       3170.0   GTO  CCSFS SLC 40
..           ...         ...            ...          ...   ...           ...
89           102  2020-09-03       Falcon 9      15600.0  VLEO    KSC LC 39A
90           103  2020-10-06       Falcon 9      15600.0  VLEO    KSC LC 39A
```

```
# Calculate the mean value of PayloadMass column
PayloadMass_mean = data_falcon9['PayloadMass'].mean()

# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] =  data_falcon9['PayloadMass'].replace(np.nan, PayloadMass_m
```

```
C:\Users\sario\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWithCopyWarnir
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
xing.html#returning-a-view-versus-a-copy
  """
```

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

```
FlightNumber       0
Date               0
BoosterVersion     0
PayloadMass        0
Orbit              0
LaunchSite         0
Outcome            0
Flights            0
```

Data Collection - Scraping

- Web scrapping to a Wikipedia page that had Falcon 9 launch records was done with BeautifulSoup.
- The table was parsed and converted into a pandas dataframe and saved as a csv file.
- https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/Data%20Collection%20with%20Web%20Scraping.ipynb

In [3]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_
```

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

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

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

In [4]:
```
# 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`

In [5]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response tex
soup = BeautifulSoup(response.text, "html.parser")
```

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

In [6]:
```
# Use soup.title attribute
soup.title
```

Out[6]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

In [7]:
```
# 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')
```

In [9]:
```
type(html_tables)
```

Out[9]:
```
bs4.element.ResultSet
```

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

In [64]:
```
# Let's print the third table and check its content
first_launch_table = html_tables[3]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date andtime ()
```

In [79]:
```
df=pd.DataFrame(launch_dict)
df
```

Out[79]:

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX
1	2	CCAFS	Dragon	0	LEO	NASA (COTS)\nNRO
2	3	CCAFS	Dragon	525 kg	LEO	NASA (COTS)
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA (CRS)
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA (CRS)

In [80]:
```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data Wrangling

- In this step, exploratory data analysis was done on the dataset, and we calculated:
  - number of launches on each site
  - number and occurrence of each orbit
  - number and occurrence of mission outcome per orbit type
  - We also created a landing outcome label from Outcome column

- https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/Data%20Wrangling.ipynb

In [5]:
```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

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

Use the method `.value_counts()` to determine the number and occurrence of each orbit in the column `Orbit`

In [6]:
```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

Out[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
```

Use the method `.value_counts()` on the column `Outcome` to determine the number of `landing_outcomes` .Then assign it to a variable landing_outcomes.

In [22]:
```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()
landing_outcomes
```

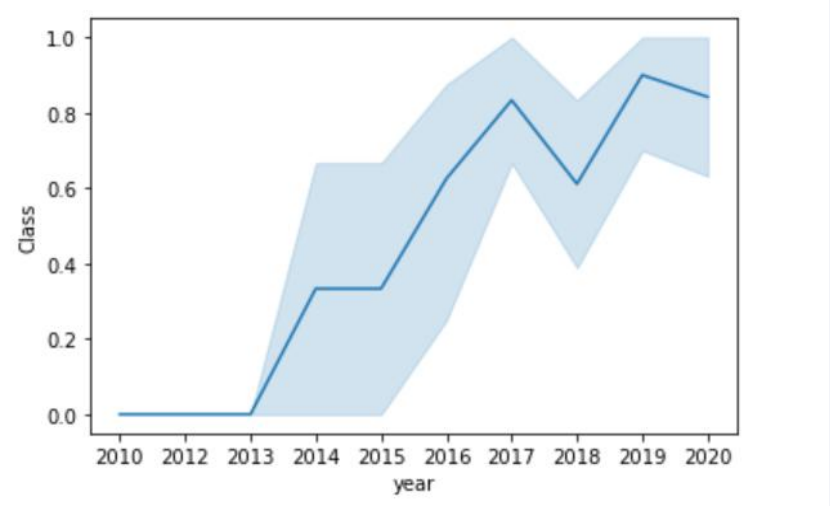
Out[22]:
```
True ASDS      41
None None      19
True RTLS      14
False ASDS      6
True Ocean      5
False Ocean     2
None ASDS       2
False RTLS      1
Name: Outcome, dtype: int64
```

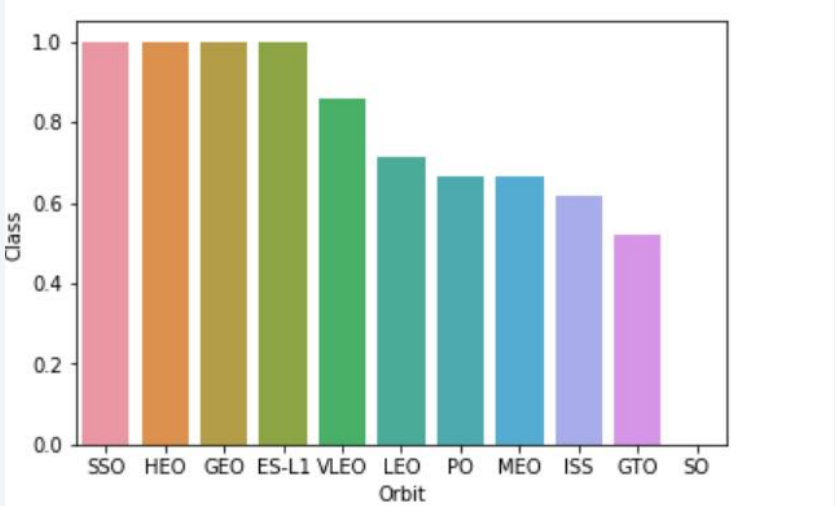
EDA with Data Visualization

- EDA was performed on the data by visualizing the relationship between

  - Flight number and launch site
  - Payload and launch site
  - Success rate of each orbit type
  - Flight number and orbit type
  - Payload and orbit type
  - Launch success yearly trends

- https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/EDA%20with%20Data%20Visualization.ipynb





Build an Interactive Map with Folium

- All launch sites were marked, and map objects like markers, circles, lines were added to mark the failure or success of launches for each site on Folium map.
- Success/Fail launches for each site on the map was marked.
- Launch outcomes feature 0 for failure, and 1 for success was added.
- Distances between a launch site to its proximities were calculated. Below questions were answered for example:

✓ Are launch sites in close proximity to railways?

✓ Are launch sites in close proximity to highways?

✓ Are launch sites in close proximity to coastline?

✓ Do launch sites keep certain distance away from cities?

https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/Interactive%20Visual%20Analytics%20with%20Folium%20lab.ipynb

EDA with SQL

- SpaceX dataset was loaded from a PostgreSQL database within the Jupyter notebook.
- SQL queries were written to find out:

- Names of the unique launch sites in the space mission - 5 Records where launch sites begin with the string 'CCA'

- Total payload mass carried by boosters launched by NASA (CRS) - Average payload mass carried by booster version F9 v1.1

- Date when the first successful landing outcome in ground pad was achieved

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

- Total number of successful and failure mission outcomes

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

- Failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

- 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

- https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/EDA%20with%20SQL.ipynb

Build a Dashboard with Plotly Dash

- An interactive dashboard with Plotly Dash was build
- Also Pie Charts showing the total launches by a certain site were drawn
- Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different booster version was drawn as well.
- https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/Interactive%20Visual%20Analytics%20with%20Plotly%20Dash%20lab.ipynb

Predictive Analysis (Classification)

- The data was loaded with Pandas and NumPy, then Standardized, and split into training and testing sets.
- Four machine learning models were build (logistic regression, support vector machine, decision tree classifier, k nearest neighbors)
- The hyperparameters were tuned using GridSearchCV.
- `accuracy_score()` was used as the metric for the models.
- The best performing classification model was found.
- https://github.com/Ahmed-M-Osman/IBM-Capstone-Project/blob/master/Machine%20Learning%20Prediction%20lab.ipynb

Results

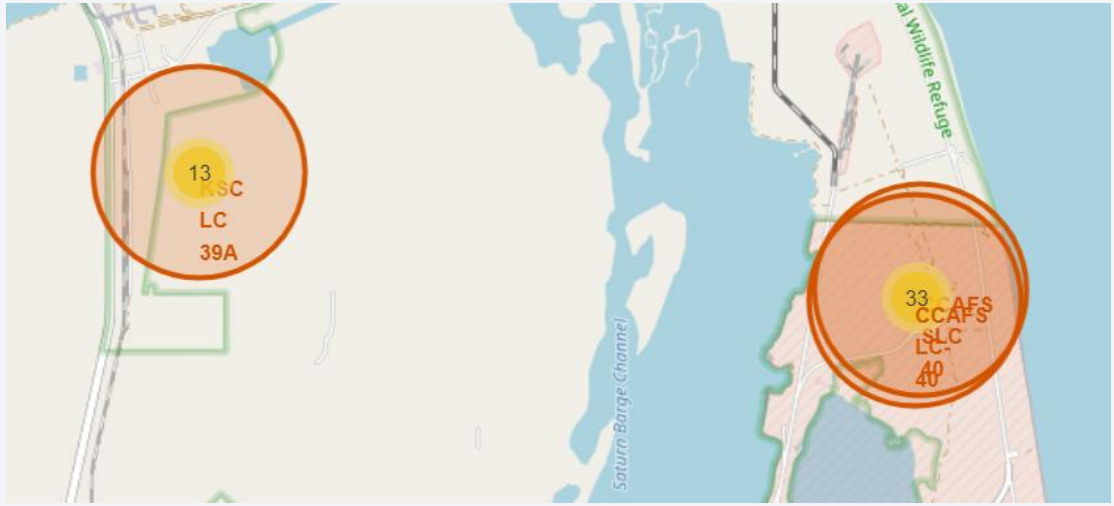
Exploratory Data Analysis

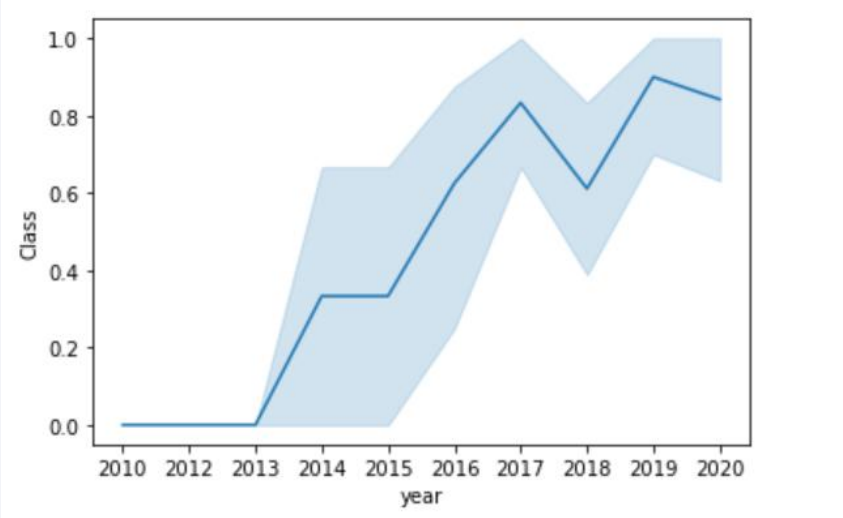
First, let's read the SpaceX dataset into a Pandas dataframe and print its summary

3]:
```
df=pd.read_csv("https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/datase
# If you were unable to complete the previous lab correctly you can uncomment and load this csv
# df = pd.read_csv('https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DS0701EN-Ski
df.head(5)
```

Out[3]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	S
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B



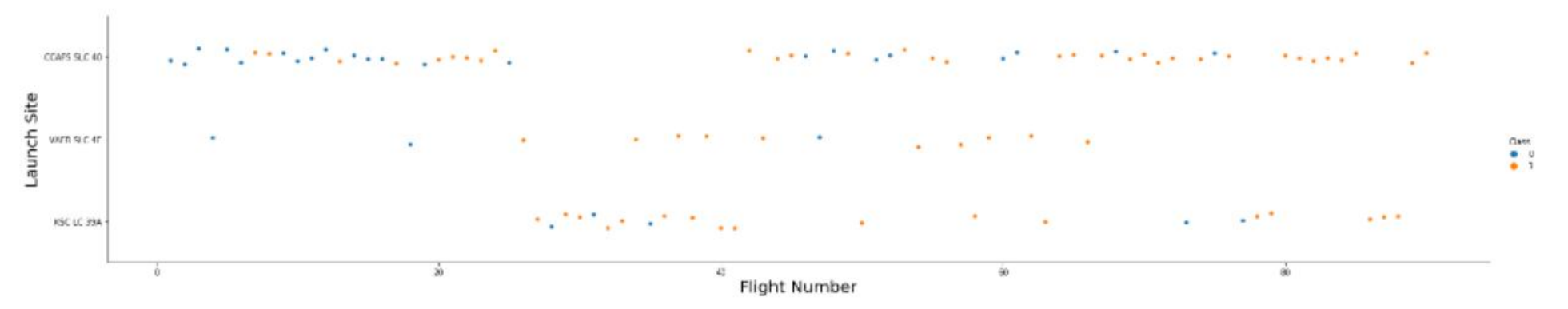


Section 2

Insights drawn from EDA

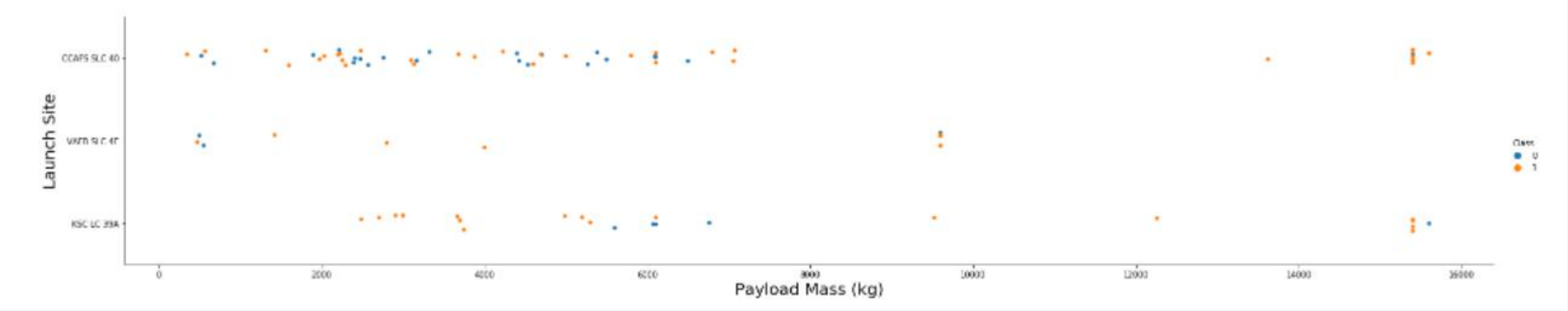
Flight Number vs. Launch Site

- From the plot, it was found that the larger the flight amount at a launch site, the greater the success rate at a launch site.
- Also, launch site CCAFS LC-40 had the most flights and the most success rate.



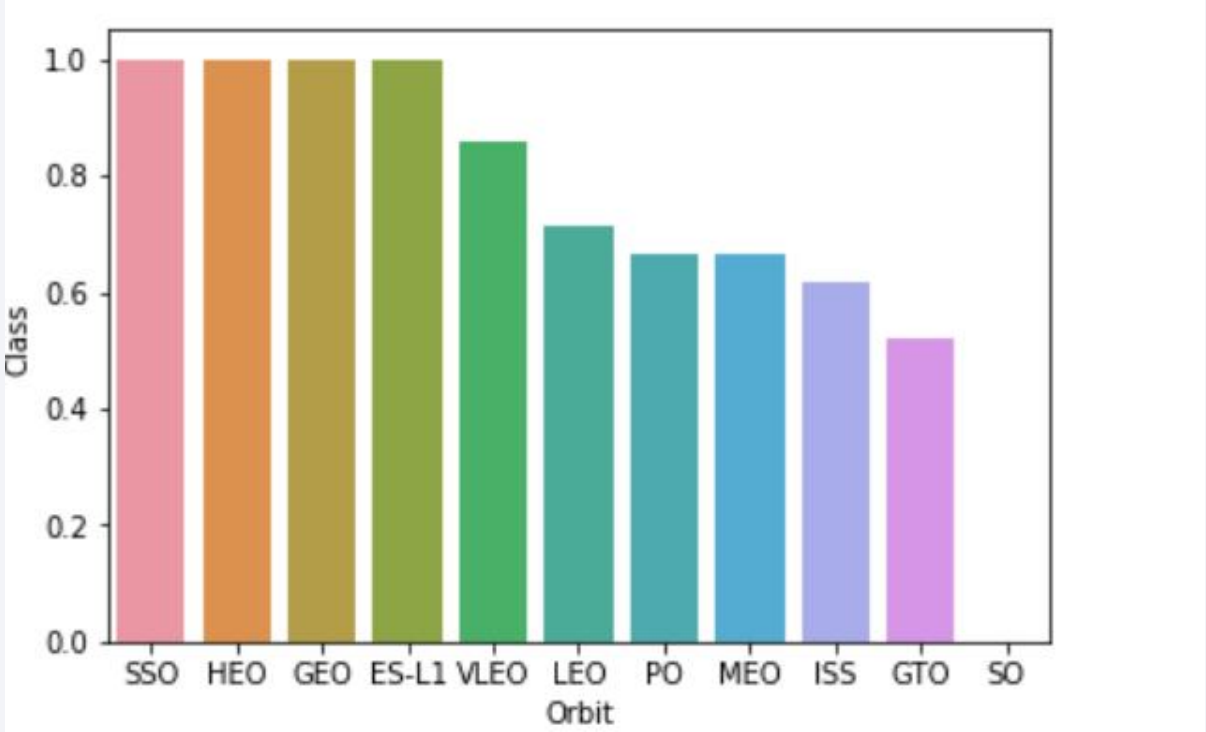
Payload vs. Launch Site

- The greater the mass of the payload the higher the success rate
- Launch site VAFB SLC-4E doesn't launch heavy payloads > 10000



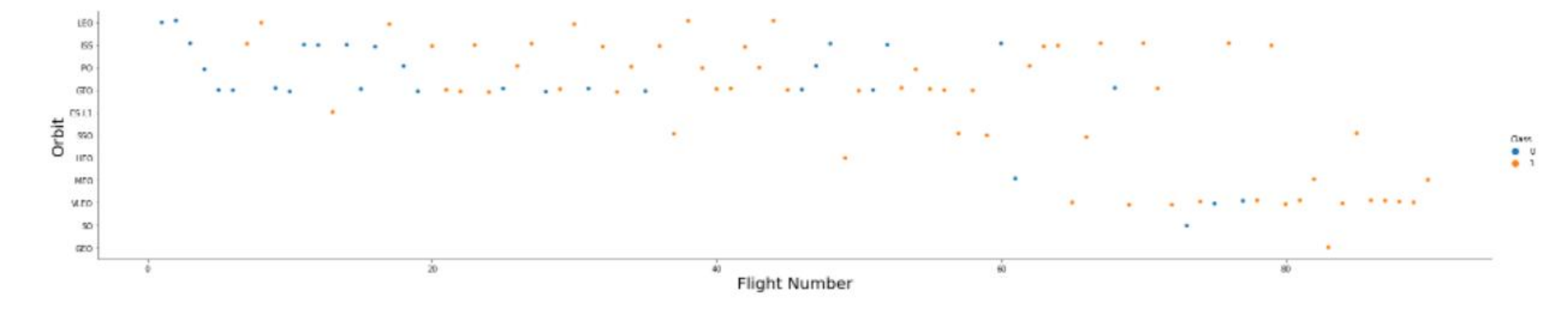
Success Rate vs. Orbit Type

- From the plot, it was shown that SSO, HEO, GEO ES-L1, VLEO had the most success rate.



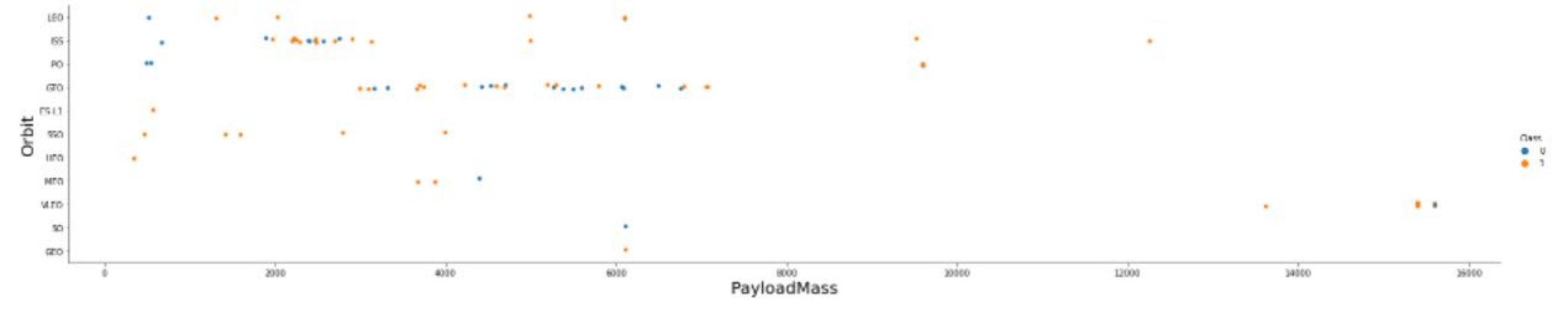
Flight Number vs. Orbit Type

- The plot below shows the Flight Number vs. Orbit type. It was observed that in the LEO orbit, success is related to the number of flights whereas in the GTO orbit, there is no relationship between flight number and the orbit.



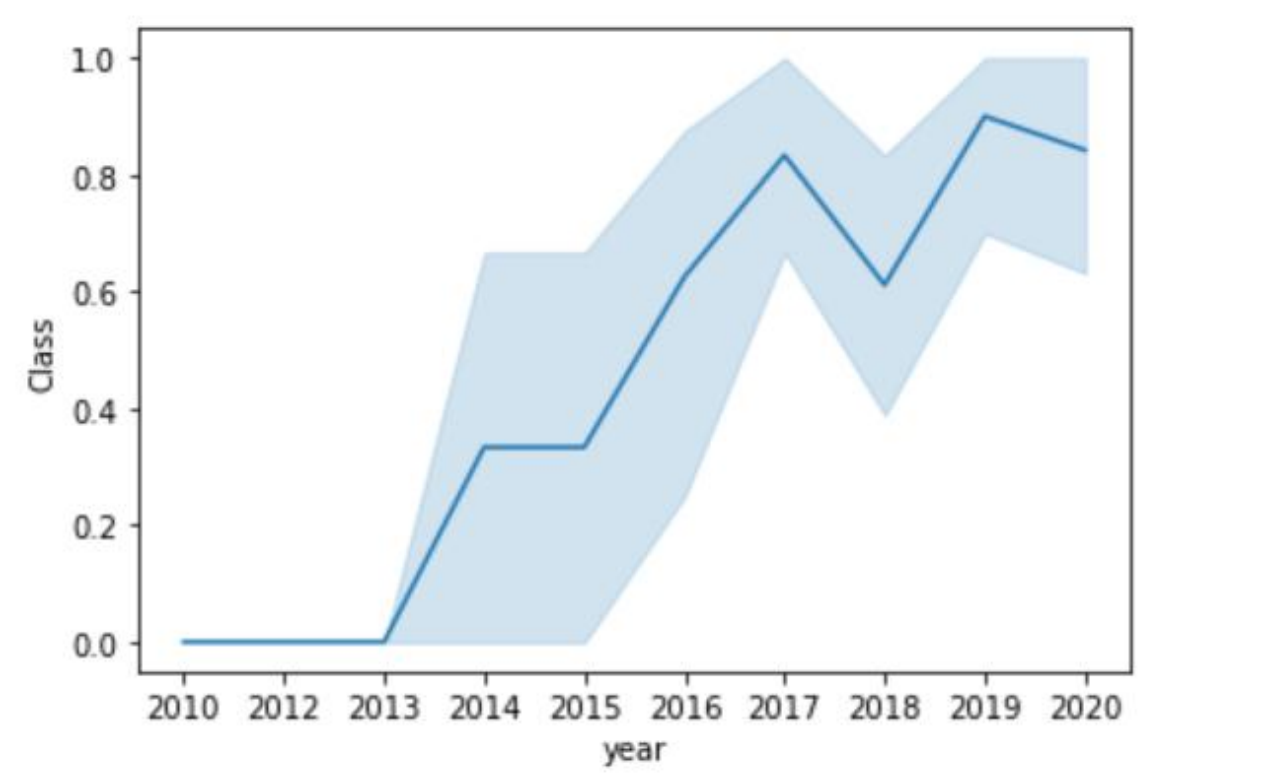
Payload vs. Orbit Type

- It was observed that with heavy payloads, the successful landing are more for Polar, LEO and ISS orbits.
- 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

- It was observed that the sucess rate since 2013 kept increasing till 2020



All Launch Site Names

- The key word **DISTINCT** was used to show only unique launch sites from the SpaceX data.

Task 1

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

In [8]:
```
%sql SELECT DISTINCT LAUNCH_SITE FROM SPACEXTBL
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa
pdomain.cloud:30875/bludb
Done.
```

Out[8]:

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

Launch Site Names Begin with 'CCA'

- A query with the **LIKE** keyword was used to display records where launch sites begin with `CCA` and **LIMIT** was used to show only 5 records

Task 2

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

```
]: %sql SELECT * FROM SPACEXTBL WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

ut[4]:

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

- The total payload carried by boosters from NASA (CRS) was calculated as 45596 using the query below

Task 3

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

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE Customer LIKE 'NASA (CRS)'
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98
Done.
```

8]:

1
45596

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1 was calculated as 2928.4 as shown below

Task 4

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

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION = 'F9 v1.1'
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.
Done.
```

6]:

1
2928

First Successful Ground Landing Date

- It was observed that the date of the first successful landing outcome on ground pad was the 22nd December 2015 as shown below

Task 5

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

Hint:Use min function

```
%sql SELECT MIN(Date) FROM SPACEXTBL WHERE LANDING__OUTCOME = 'Success (ground pad)'
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g
Done.
```

8]:

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- The **WHERE** clause was used to filter for boosters which have successfully landed on drone ship and applied the **AND** condition to determine successful landing with payload mass greater than **4000** but less than **6000** as shown below

Task 6

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

```
ct booster_version from spacextbl where LANDING__OUTCOME = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

t[8]:

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

Total Number of Successful and Failure Mission Outcomes

- The wildcard **LIKE '%'** was used to filter for WHERE MISSION_OUTCOME was a success or a failure, as shown below

Task 7

List the total number of successful and failure mission outcomes

```
%sql select count(MISSION_OUTCOME) AS success_Outcoms from spacextbl where MISSION_OUTCOME like 'Success%'
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cl
Done.
```

4]:

success_outcoms
100

```
%sql select count(MISSION_OUTCOME) AS failure_Outcoms from spacextbl where MISSION_OUTCOME like 'Failure%'
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cl
Done.
```

3]:

failure_outcoms
1

Boosters Carried Maximum Payload

- it was determined the booster that have carried the maximum payload using a subquery in the **WHERE** clause and the **MAX()** function, as shown below

Task 8

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)
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875
Done.
```

16]:

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

- A combinations of the **WHERE** clause, **LIKE**, **AND**, and **BETWEEN** conditions were used to filter for failed landing outcomes in drone ship, their booster versions, and launch site names for year 2015, as shown below

Task 9

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

```
G__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE from spacextbl where LANDING__OUTCOME = 'Failure (drone ship)' and year(DATE) = 2015
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

18]:

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- Landing outcomes and the **COUNT** of landing outcomes were selected from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20. Then **GROUP B** clause was applied to group the landing outcomes and the **ORDER BY** clause was applied to order the grouped landing outcome in descending order.

Task 10

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

```
DING__OUTCOME, count(LANDING__OUTCOME) as landing_outcome_count from spacextbl where DATE between '2010-06-04' and '2017-03-2
```

```
 * ibm_db_sa://fyd37602:***@98538591-7217-4024-b027-8baa776ffad1.c3n41cmd0nqnrk39u98g.databases.appdomain.cloud:30875/bludb
Done.
```

2]:

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

Section 3

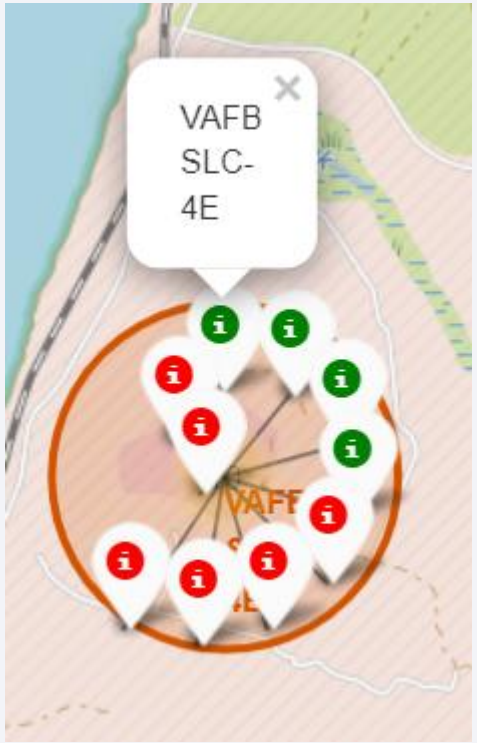
Launch Sites Proximities Analysis

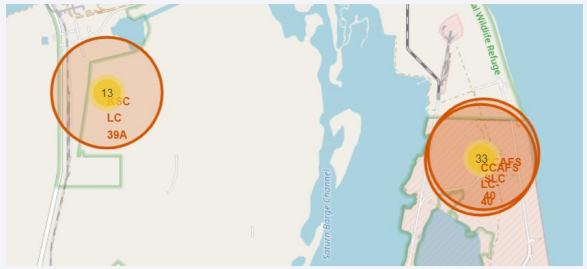
Global launch sites

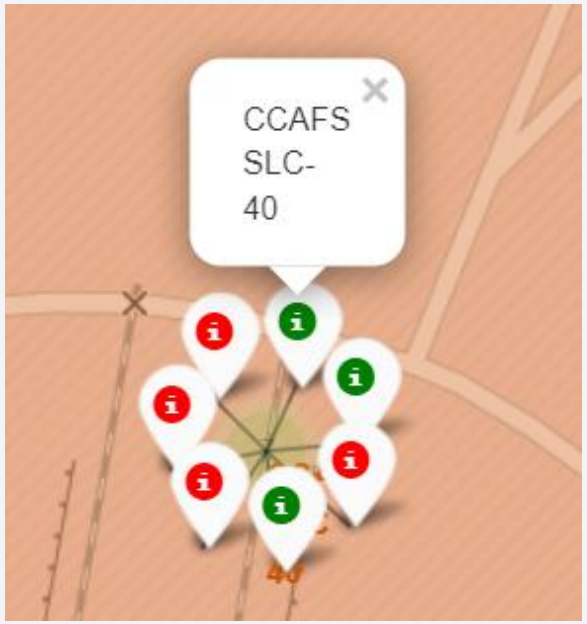
It is shown that SpaceX launch sites are located in USA only, in Florida and California

Launch sites with colors labels for success/failure

California



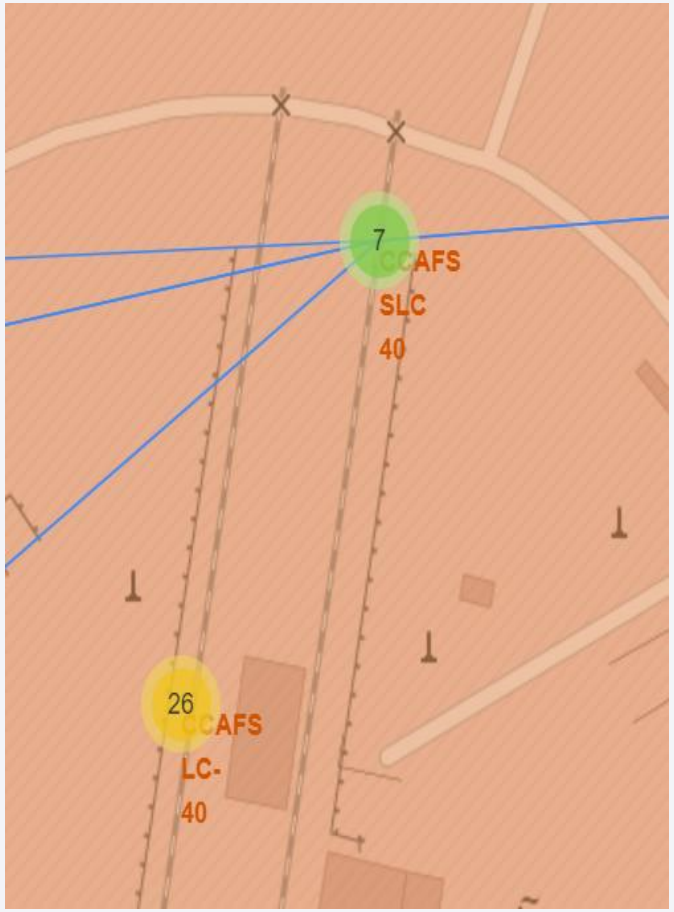




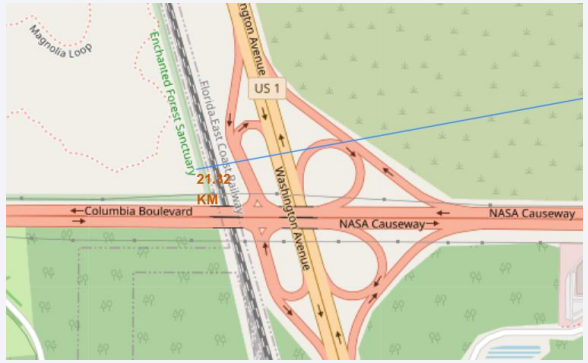
Florida

Green == Success
Red === Failure

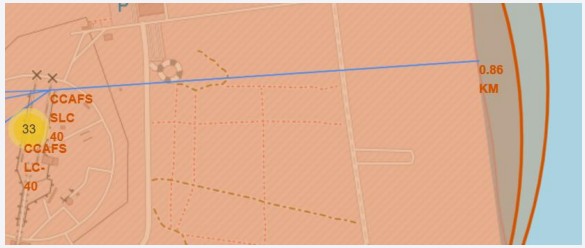
Important distances from launch site



Launch site

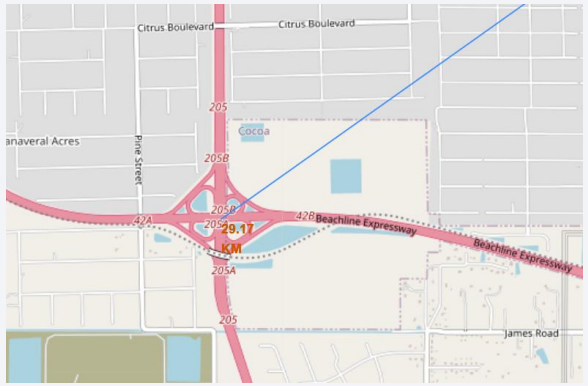


Distance to railway



Distance to coastline

Distance to city



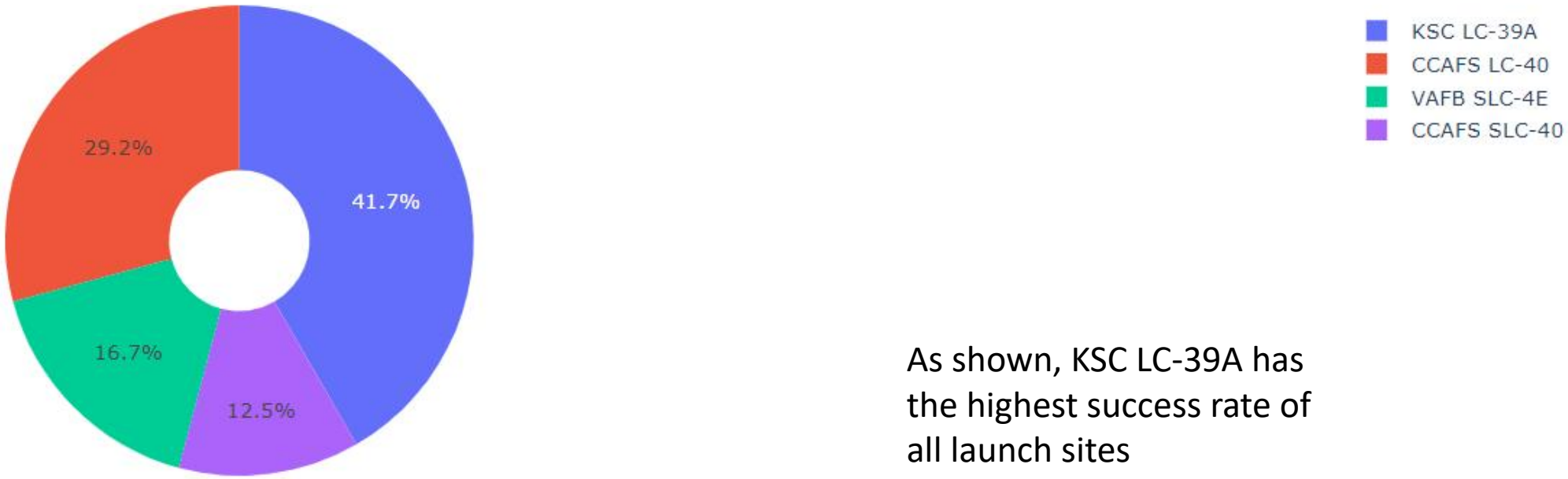
Distance to highway

- 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***

Section 4

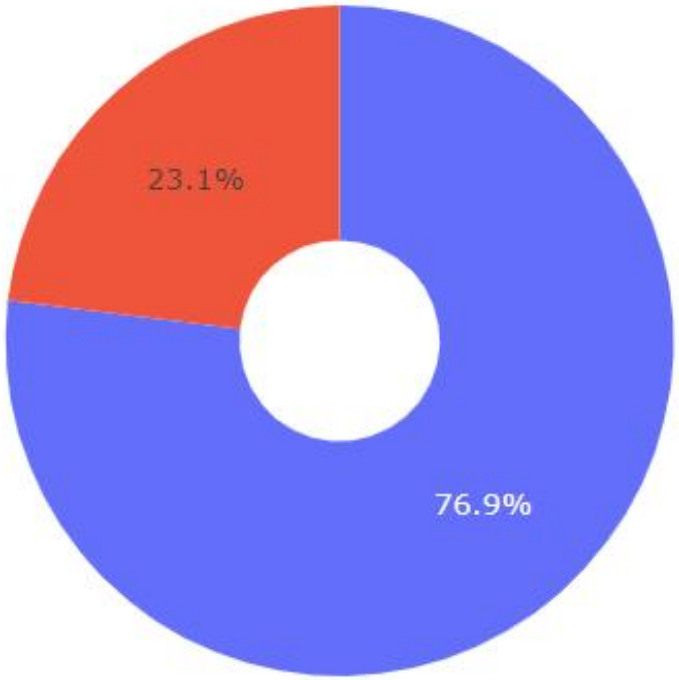
Build a Dashboard with Plotly Dash

Success rate of each launch site in a pie chart



As shown, KSC LC-39A has the highest success rate of all launch sites

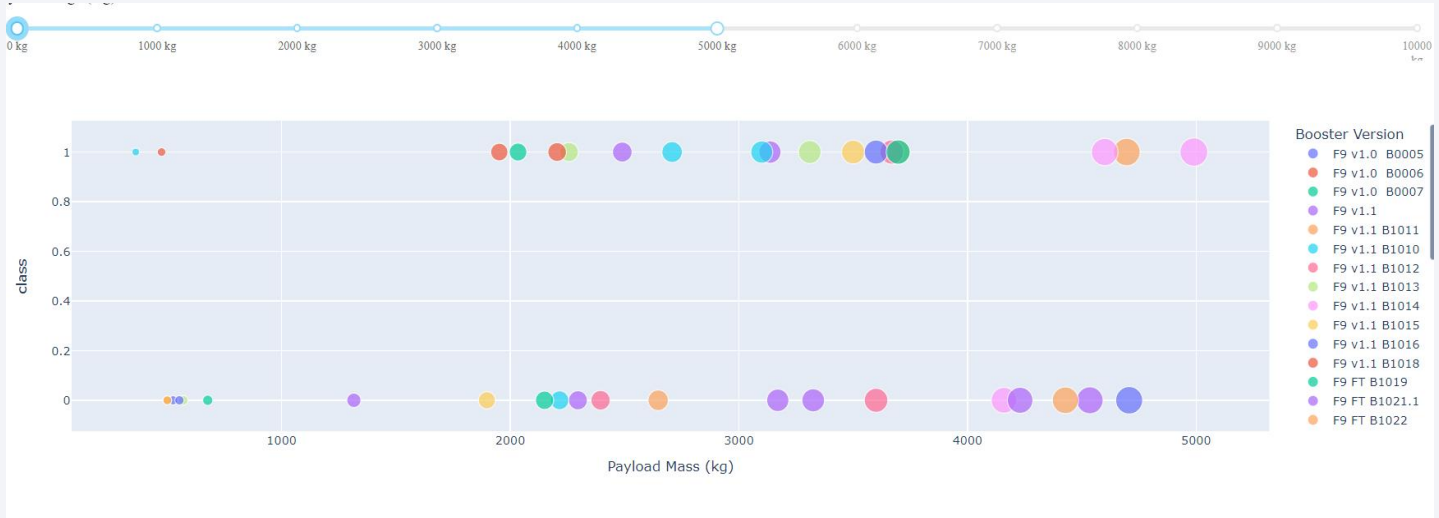
A pie chart showing the launch site with the highest success rate



KSC LC-39A has a success rate of 76.9% while its failure rate is only 23.1%

Scatter plot for success rate by payload mass

It was observed that payloads less than 5000 kg has higher success rate than those above it

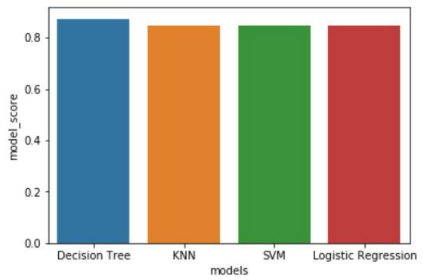


Section 5

Predictive Analysis (Classification)

Classification Accuracy

```
grouped = df.groupby(by=['models'])['model_score'].mean().sort_values(ascending=False).reset_index()
sns.barplot(x="models",y="model_score",data=grouped)
plt.show()
```

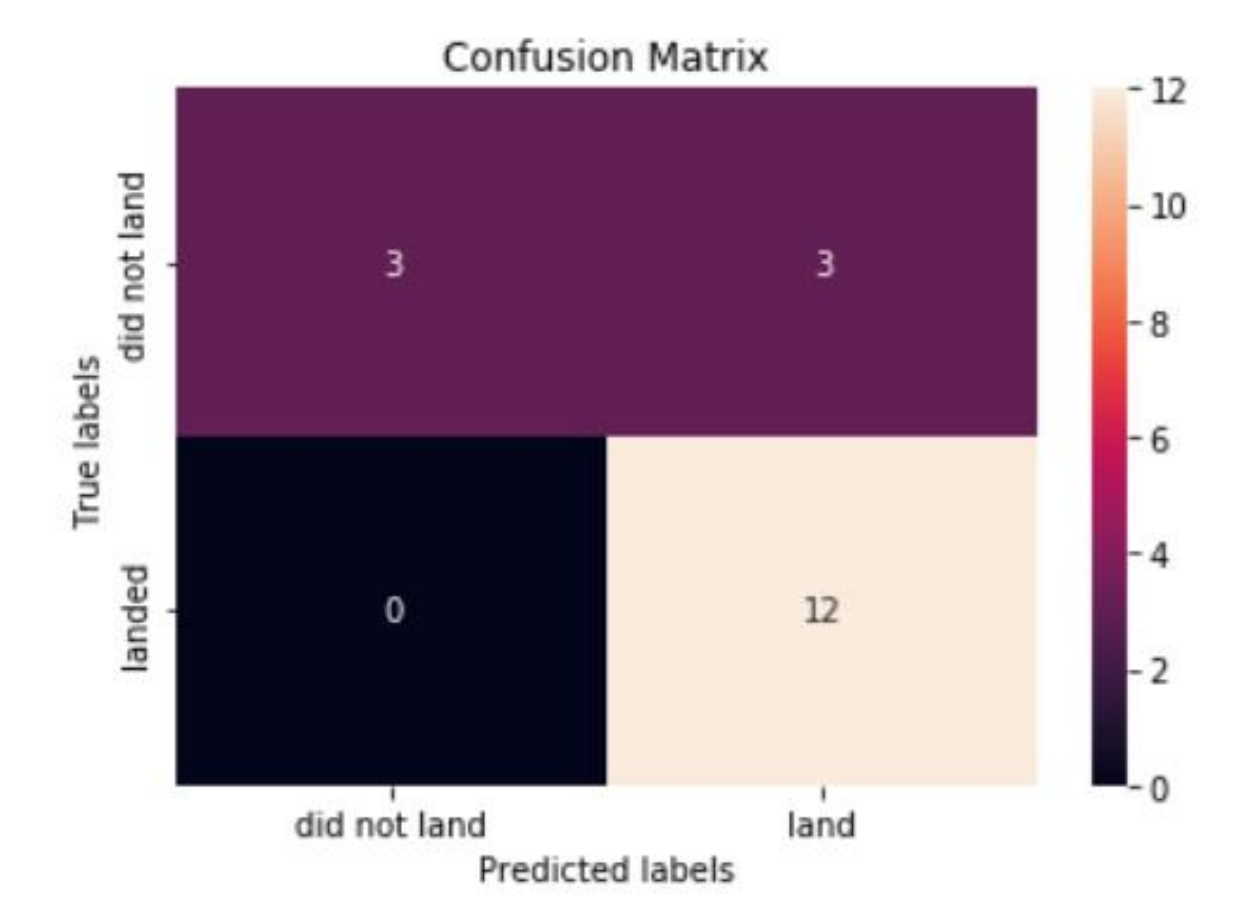


The decision tree classifier was found to be the model with the highest classification accuracy

```
models_score = [lr_model.best_score_, svm_model.best_score_,  tree_model.best_score_, knn_model.best_score_]
model_params = [lr_model.best_params_, svm_model.best_params_, tree_model.best_params_, knn_model.best_params_]
models = ["Logistic Regression", "SVM", "Decision Tree", "KNN"]

print("Logistic Reg.'s Accuracy: ", models_score[0])
print("SVM's Accuracy: ", models_score[1])
print("DecisionTrees's Accuracy: ", models_score[2])
print("KNN's Accuracy: ", models_score[3])
print("-------------------------------------------------------------\n")

index_max = np.argmax(models_score)
best_model = models[index_max]
max_score = models_score[index_max]
best_params = model_params[index_max]

print("Best method is: {} \nWith an accuracy of: {} \nBest parameters are: {}"
      .format(best_model, max_score, best_params))
```

```
Logistic Reg.'s Accuracy:  0.8464285714285713
SVM's Accuracy:  0.8482142857142856
DecisionTrees's Accuracy:  0.8732142857142856
KNN's Accuracy:  0.8482142857142858
-------------------------------------------------------------

Best method is: Decision Tree
With an accuracy of: 0.8732142857142856
Best parameters are: {'criterion': 'gini', 'max_depth': 6, 'max_features': 'auto', 'min_samples_leaf': 2, 'min_samples_spli
t': 5, 'splitter': 'random'}
```

Confusion Matrix



As shown in the confusion matrix, the model has a false positive problem. But in general it performed good.

Conclusions

It was concluded that:

- The more flight number a launch site has, it relates proportionally with its success rate.
- The rate of success has been on the rise from 2013 till 2020.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the highest success rate.
- KSC LC-39A had the most successful launches of any sites.
- The best machine learning algorithm was found to be the Decision tree classifier.

Thank you!