IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Orlando Jesús Herrera Ruiz

September, 2022

Outline

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

Executive Summary

- Summary of methodologies
    - Data collection using API
    - Data collection using Website Scraping
    - Data wrangling
    - Data Analysis using SQL
    - Data Visualization
    - Visual Analytics with Folium
    - Machine Learning Predictions
- Summary of all results
    - Predictive analytics result

Introduction

- Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. Therefore if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch. In this lab, you will collect and make sure the data is in the correct format from an API. The following is an example of a successful and launch.

- Problems you want to find answers

1. What will determine the successful landing of the rockets?
2. What conditions are necessary to have success launches and landings?

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping this link of wikipedia
- Perform data wrangling
  - Watching, Decoding, and Cleaning Data
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Machine learning algorithms with Python

Data Collection

The data was collected:

- Using get & request and SpaceX API
- Decoding and normalizing data with Pandas .json() and json_normalize()
- Data was cleaned, removing null data or fill missed values
- Data table from wikipedia was transformed in dataframe

Data Collection – SpaceX API

- Using api from SpaceX

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)
```

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

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

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

```
response.status_code
```

```
200
```

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

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

- Click here and open the file

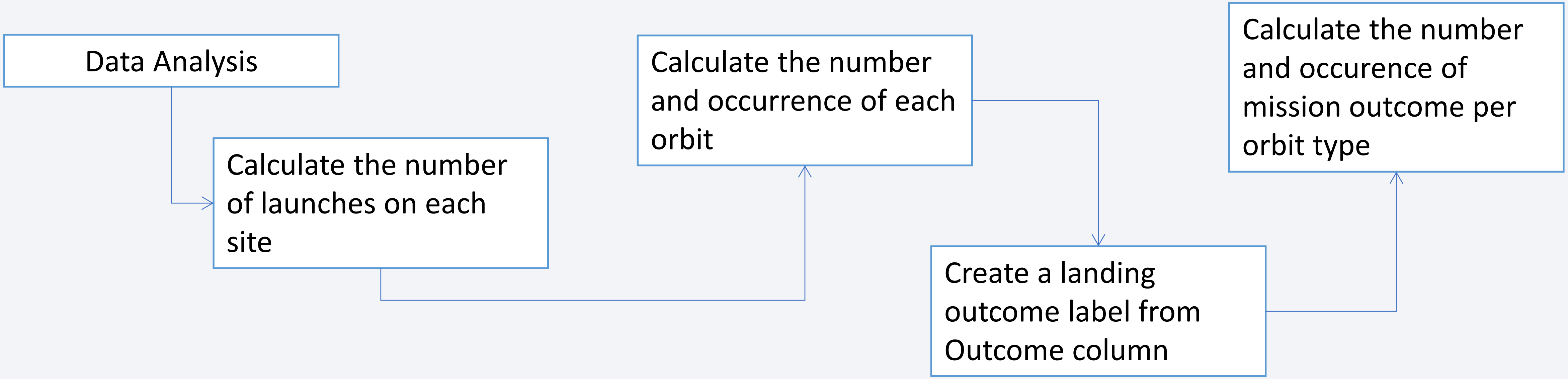
Data Collection - Scraping

- Webscrapping and BeautifulSoup
- The table was converted in a Pandas dataframe
- You can visit the file, [click here]()

```python
headings = []
for key,values in dict(launch_dict).items():
    if key not in headings:
        headings.append(key)
    if values is None:
        del launch_dict[key]

def pad_dict_list(dict_list, padel):
    lmax = 0
    for lname in dict_list.keys():
        lmax = max(lmax, len(dict_list[lname]))
    for lname in dict_list.keys():
        ll = len(dict_list[lname])
        if  ll < lmax:
            dict_list[lname] += [padel] * (lmax - ll)
    return dict_list

pad_dict_list(launch_dict,0)

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

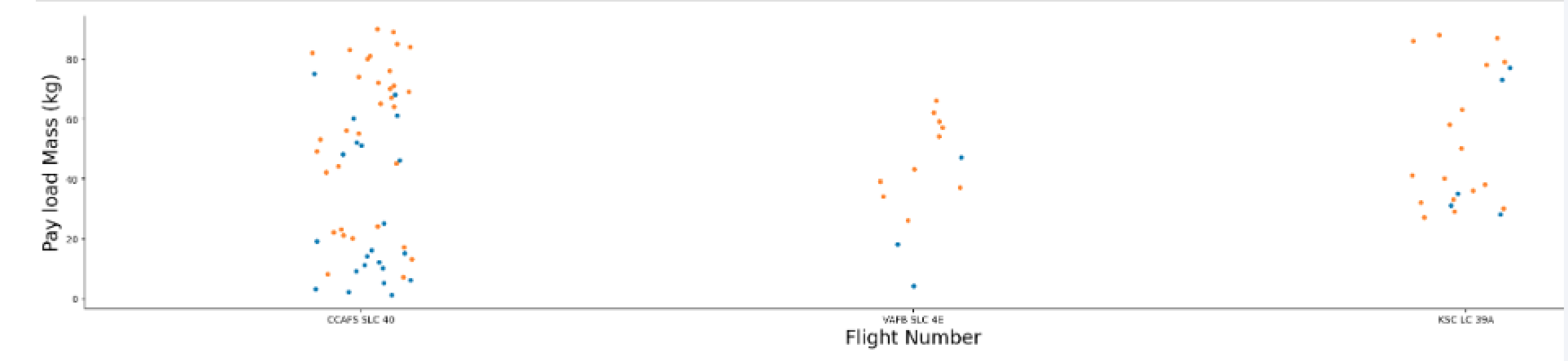
	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	<generator object Tag._all_strings at 0x7ff1d8...	Success\n	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	<generator object Tag._all_strings at 0x7ff1d8...	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	<generator object Tag._all_strings at 0x7ff1d8...	Success	F9 v1.0B0005.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	<generator object Tag._all_strings at 0x7ff1d8...	Success\n	F9 v1.0B0006.1	No attempt	8 October 2012	00:35

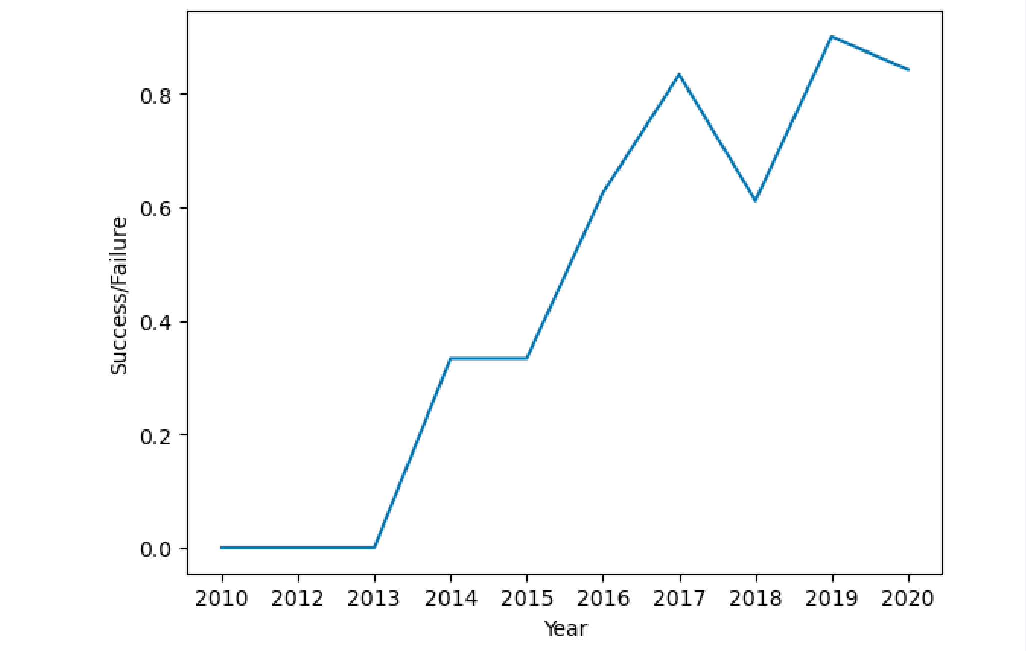
Data Wrangling



- The complete notebook is in the next link

EDA with Data Visualization





[The complete notebook is the next link]()

EDA with SQL

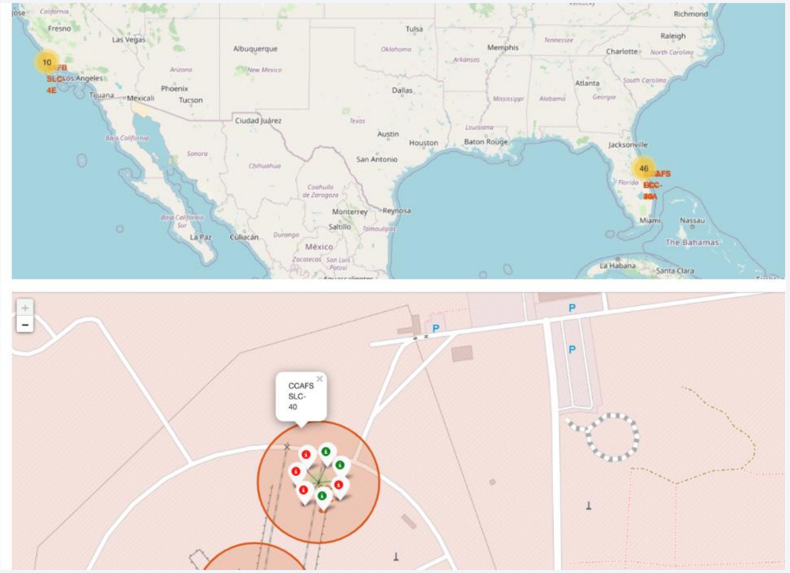
SQL queries insights:

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

Complete list of queries clicking here

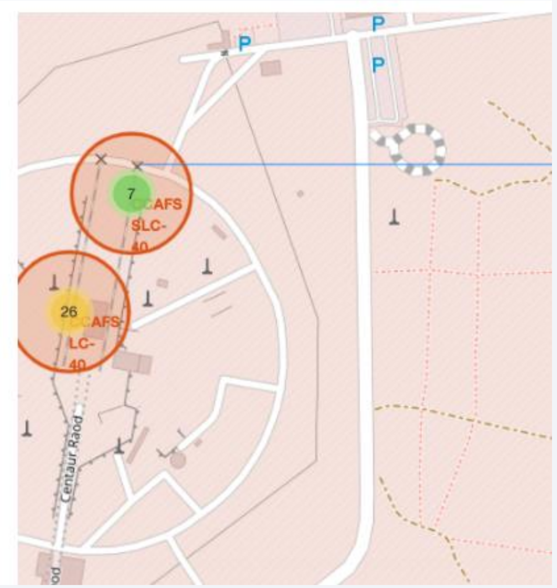
Build an Interactive Map with Folium

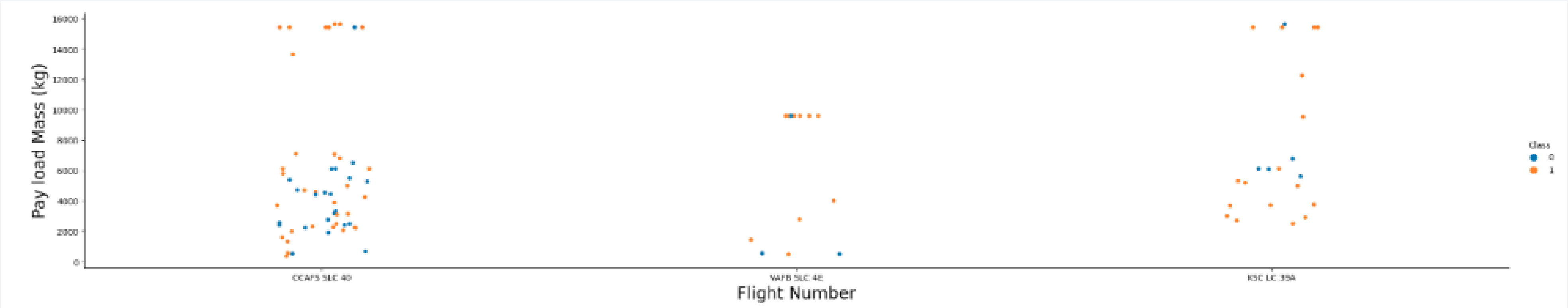
- Sites were marked, We added map objects: circles & lines to mark the success or failure of launches for each site on the map.
- 0 Class represent failure (red points), 1 class represent success
- Identification of which launch site have high probability success rate
- The distance between a launch site was calculated to its proximities.



Build a Dashboard with Plotly Dash

- Interactive dashboard with Plotly dash
- Mark all launch sites on a map
- Calculate the distances between a launch site to its proximities
- The link to the notebook is [here]





Predictive Analysis (Classification)

- Numpy and pandas for transform the data, after that split the data into training and testing.
- Different ML models were built
- We use accuracy to measure the best performance of the algorithms

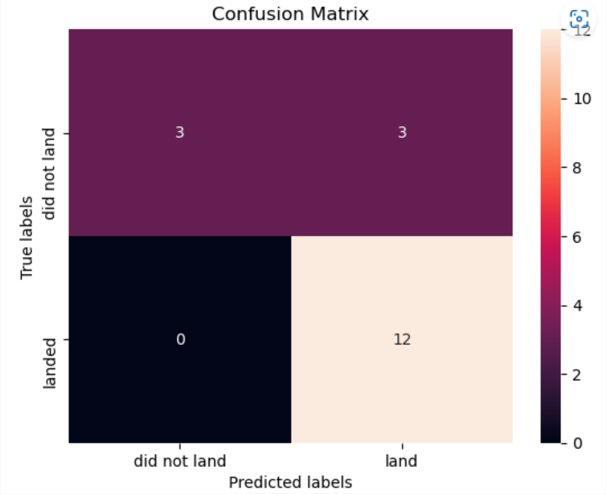
```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.7222222222222222
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

If you want to see the complete notebook, please click next link

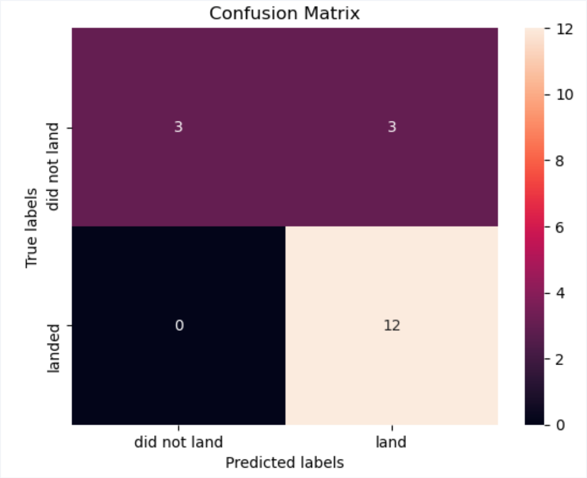
Results

	FlightNumber	PayloadMass	Flights	Block	ReusedCount	Orbit_ES-L1	Orbit_GEO	Orbit_GTO	Orbit_HEO	Orbit_ISS	...	Serial_B1058	Serial_B1059	Ser
0	1.0	6104.959412	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
1	2.0	525.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
2	3.0	677.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	...	0.0	0.0	
3	4.0	500.000000	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
4	5.0	3170.000000	1.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
85	86.0	15400.000000	2.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
86	87.0	15400.000000	3.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	1.0	0.0	
87	88.0	15400.000000	6.0	5.0	5.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
88	89.0	15400.000000	3.0	5.0	2.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	
89	90.0	3681.000000	1.0	5.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	

90 rows × 83 columns



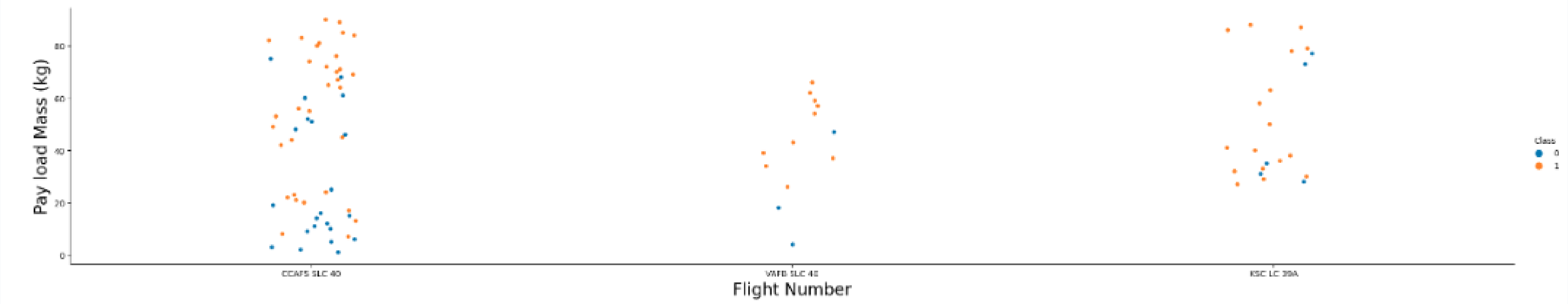

```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.7222222222222222
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```



Section 2

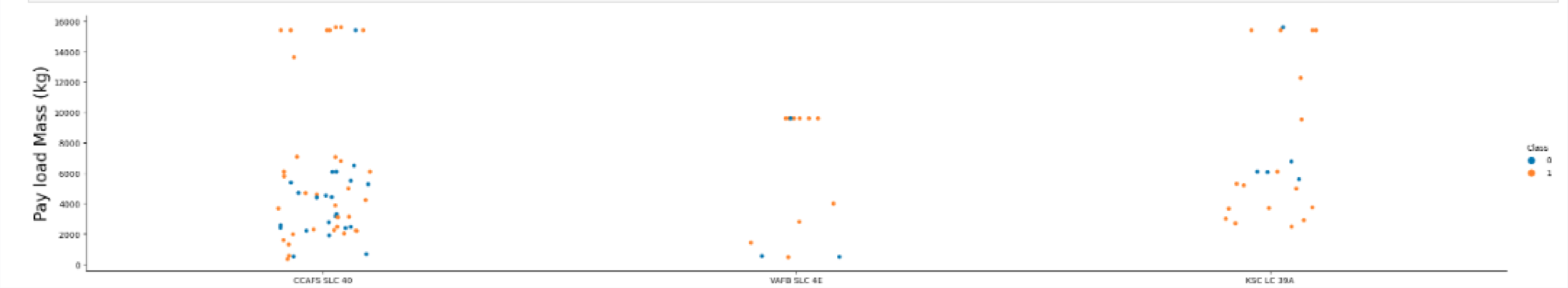
Insights drawn from EDA

Flight Number vs. Launch Site



Payload vs. Launch Site

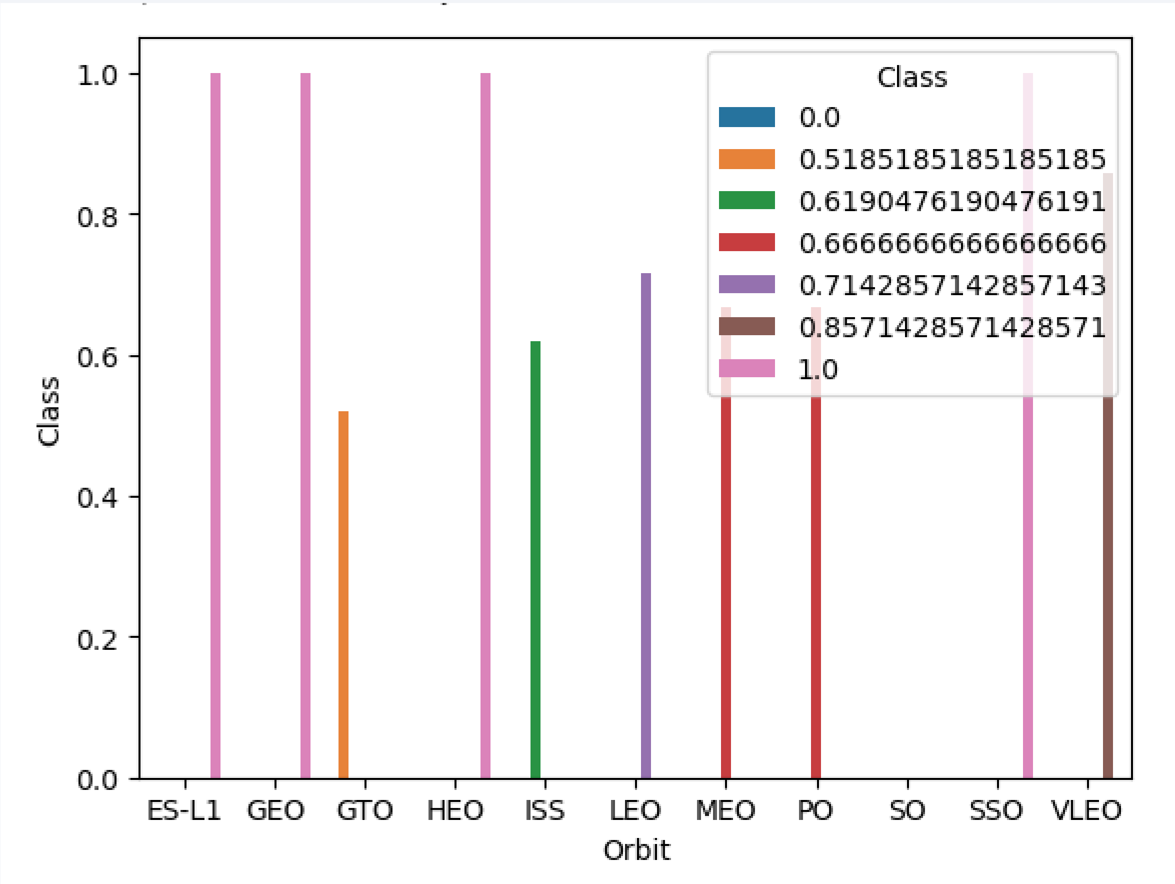
CCAFS higher rate of success



VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

Success Rate vs. Orbit Type

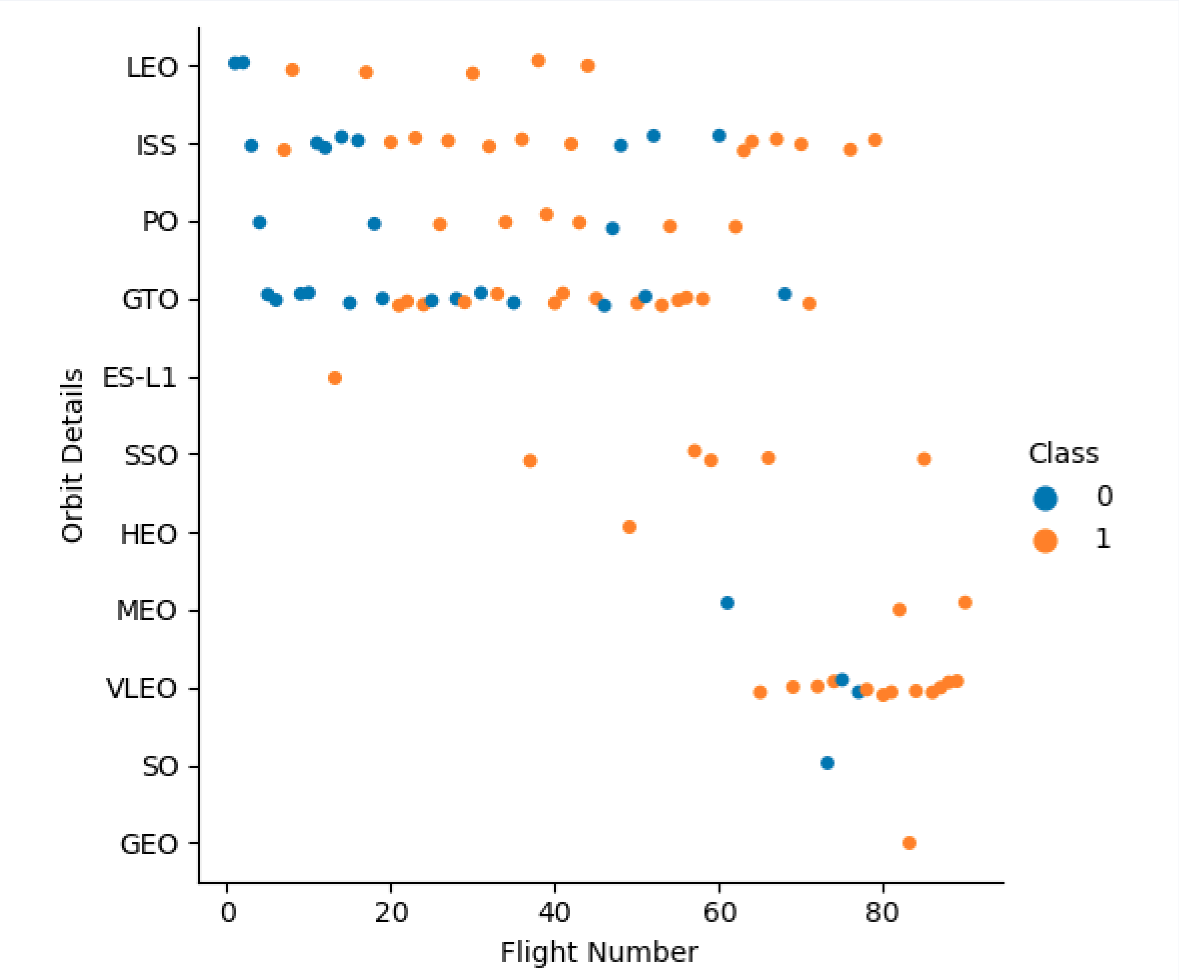
ES-L1, GEO, HEO, SSO, VLEO had the most success rate.



Flight Number vs. Orbit Type

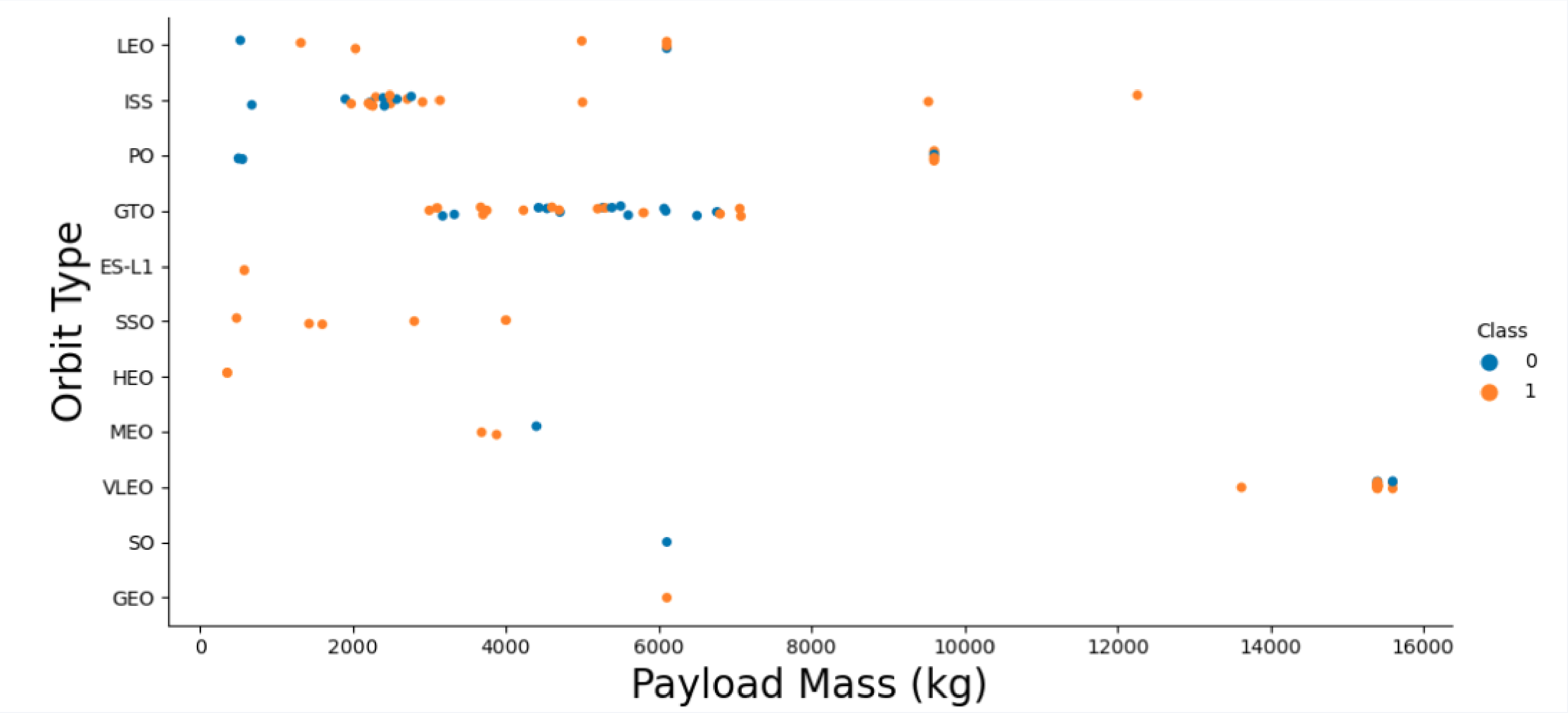
After the half flights, the success was higher

ES-L1, SSO and HEO was 100% of success



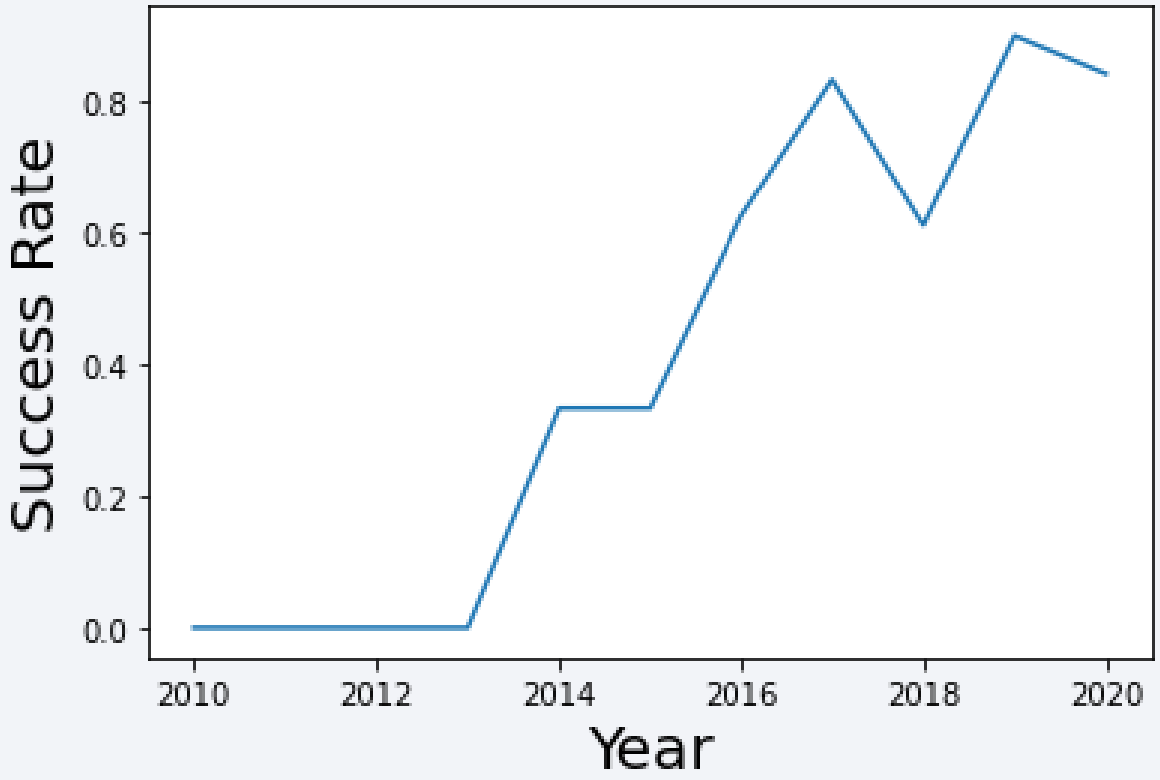
Payload vs. Orbit Type

PO, ISS, VLEO with high success heavy payload



Launch Success Yearly Trend

More experience is synonym of high success rate



All Launch Site Names

DISTINCT clause to show only unique launch sites from the SpaceX data.

```
[7]: %sql select distinct Launch_Site from spacexdata
 * sqlite:///my_data1.db
Done.
```

[7]:

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

Launch Site Names Begin with 'CCA'

```
[9]: %sql select * from spacexdata where Launch_Site like 'CCA%' limit 5
 * sqlite:///my_data1.db
Done.
```

[9]:

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

“LIKE” clause to select “CCA” launch sites

Total Payload Mass

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

```
%sql select sum(PAYLOAD_MASS__KG_) from spacexdata where Customer='NASA (CRS)'
```

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

sum(PAYLOAD_MASS__KG_)
45596

“SUM” clause to calculate the total

Average Payload Mass by F9 v1.1

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

```
%sql  select avg(PAYLOAD_MASS__KG_) from spacexdata where Booster_Version='F9 v1.1'
```

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

avg(PAYLOAD_MASS__KG_)
2928.4

“AVG” clause to calculate the average payloadmass

First Successful Ground Landing Date

```
%sql select min(Date) from spacexdata where [Landing _Outcome] LIKE '%Success (ground pad)%'
```

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

min(Date)
01-05-2017

We observed the first successful landing outcome on ground pad was 01/05/2017

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select distinct Booster_Version from spacexdata where [Landing _Outcome]='Success (drone ship)' and PAYLOAD_MASS__KG_ between 4000 and 6000
```

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

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

We used the WHERE clause to filter successfully landed on drone ship

BETWEEN clause to determine successful landing with payload mass greater than 4000 but less than 6000

Total Number of Successful and Failure Mission Outcomes

```
%sql select substr(Mission_Outcome,1,7) as Mission_Outcome, count(*) from spacexdata  group by 1
```

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

Mission_Outcome	count(*)
Failure	1
Success	100

GROUP clause split two groups “Failure” and “Success”

COUNT clause calculate all rows with the previous condition

Boosters Carried Maximum Payload

```
%sql select distinct Booster_Version from spacexdata where PAYLOAD_MASS__KG_ = (select max(PAYLOAD_MASS__KG_) from spacexdata)
```

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

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

MAX clause take the maximum payload mass value in that column

DISTINCT clause take unique values, not repeat

2015 Launch Records

```
%sql select distinct [Landing _Outcome], Booster_Version, Launch_Site from spacexdata where [Landing _Outcome]='Failure (drone ship)'
```

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

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
Failure (drone ship)	F9 v1.1 B1017	VAFB SLC-4E
Failure (drone ship)	F9 FT B1020	CCAFS LC-40
Failure (drone ship)	F9 FT B1024	CCAFS LC-40

WHERE clause is the condition need to be validate

DISTINCT clause take unique values, not repeat

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

```
ql select [Landing _Outcome], count(*) from spacexdata where Date between '04-06-2010' and '20-03-2017' group by [Landing _Outcome] order by 2 desc
```

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

Landing _Outcome	count(*)
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	6
Failure (drone ship)	4
Failure	3
Controlled (ocean)	3
Failure (parachute)	2
No attempt	1

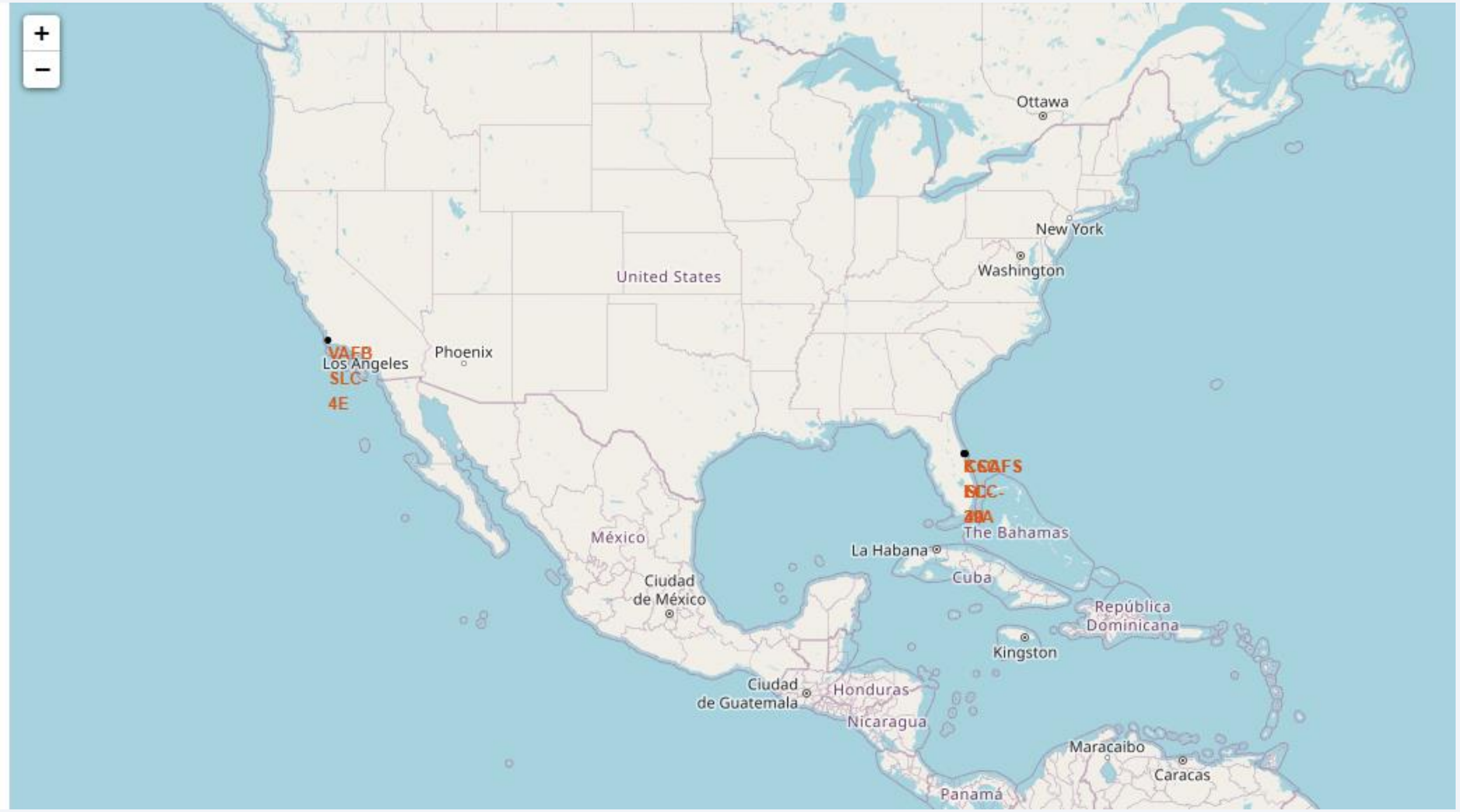
20 Success landings outcome

Comparing dates with BETWEEN clause and DESC clause show results in descending order

Section 3

Launch Sites Proximities Analysis

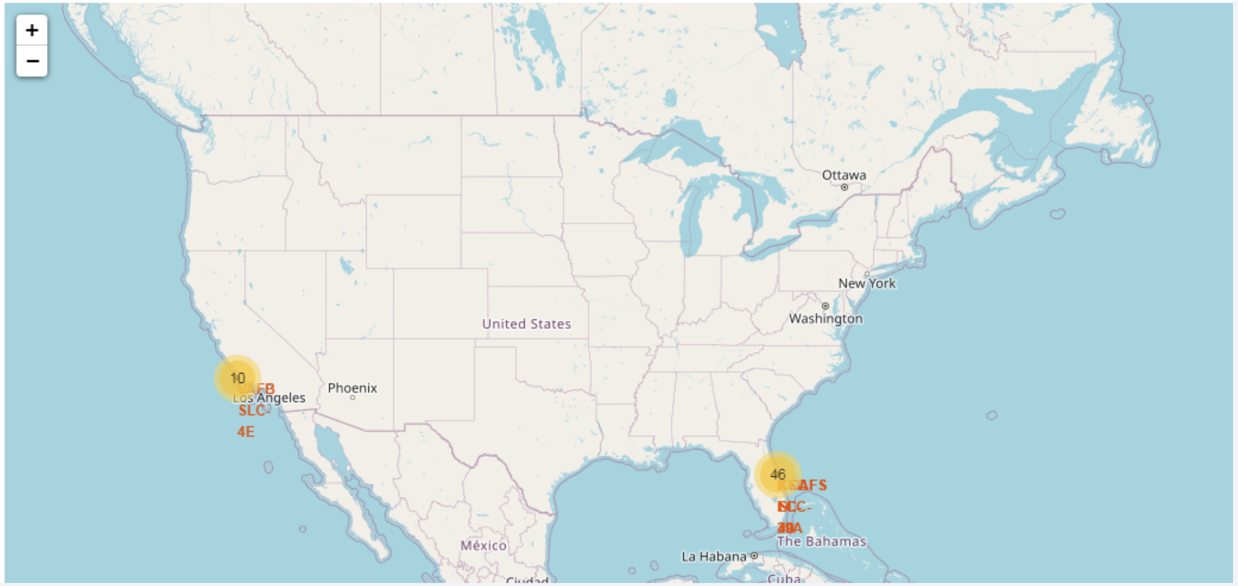
All launch sites global map markers

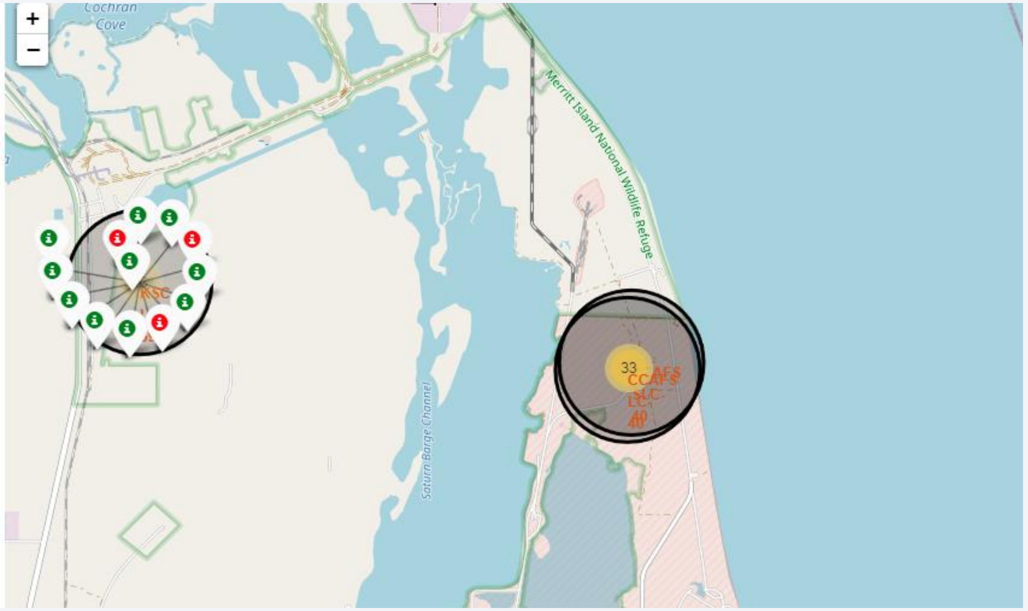


All launches are from Florida and California

Markers showing launch sites

```
for index, row in spacex_df.iterrows():
    # create and add a Marker cluster to the site map
    coordinate = [row['Lat'], row['Long']]
    folium.map.Marker(coordinate, icon=folium.Icon(color='white',icon_color=row['marker_color'])).add_to(marker_cluster)
site_map
```





Green is succeed, red is a fail.
Zoom in to show the map with all succeeded and failure launches

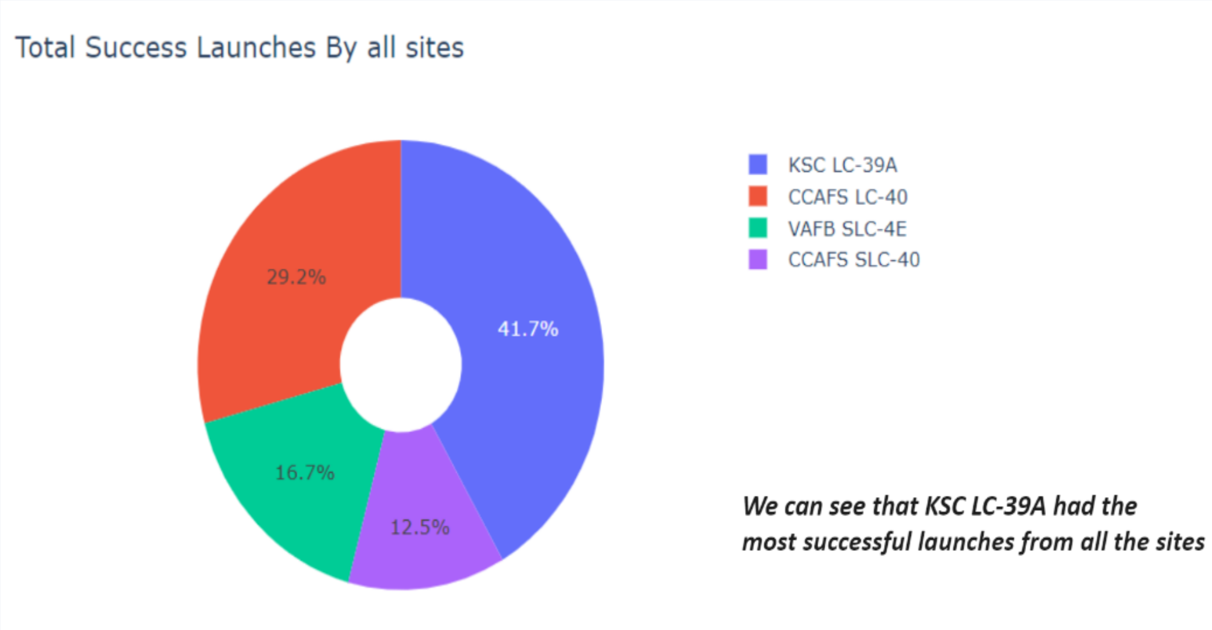
Launch Site distance

A blue line show the distance from one poin to the Florida coast. In this case: 0.9km.

Section 4

Build a Dashboard with Plotly Dash

Total success launches group by all sites



Section 5

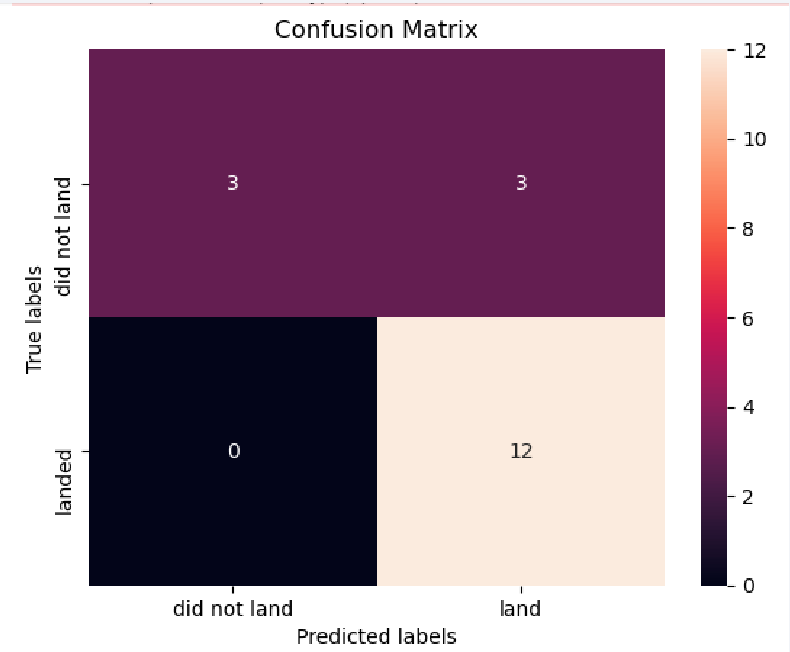
Predictive Analysis (Classification)

Classification Accuracy

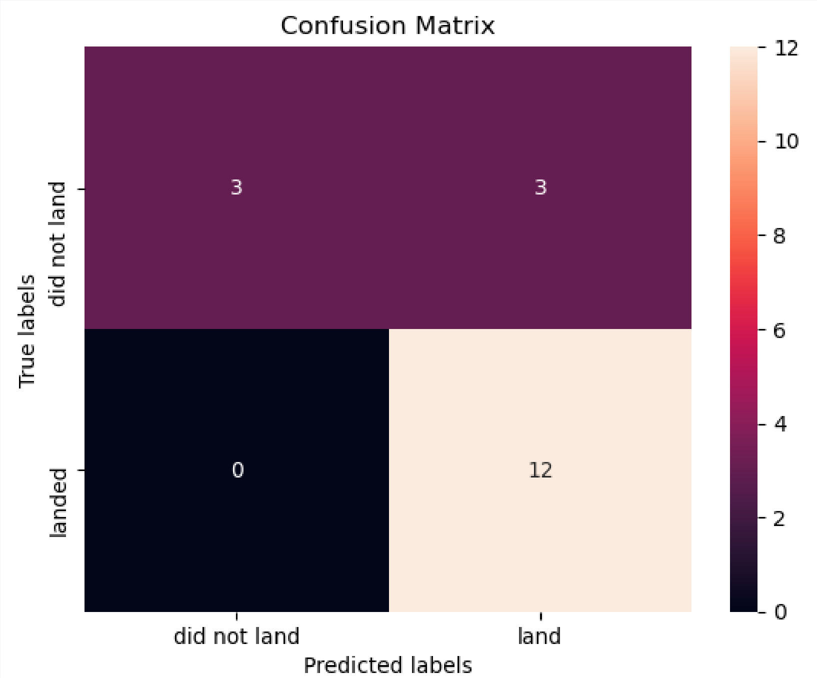
```
Accuracy for Logistics Regression method: 0.8333333333333334
Accuracy for Support Vector Machine method: 0.8333333333333334
Accuracy for Decision tree method: 0.7222222222222222
Accuracy for K nearsdt neighbors method: 0.8333333333333334
```

Logistic Regression, SVM & KN have a high accuracy metric

Confusion Matrix



Logistic Regression



SVM

Conclusions

- ES-L1, GEO, HEO, SSO, VLEO orbits, had a very high success rate
- Logistic Regression, SVM & KN show a very high accuracy metric, so we can work with them
- KSC LC-39A is the most successful site for launches
- If the flight amount at a launch site is large, the probability of success rate will be high

Thank you!