IBM Developer
SKILLS NETWORK

TheDataMultiverse

Winning Space Race With Data Science

Isaac Dada – 28/03/22

OUTLINE

- EXECUTIVE SUMMARY
- INTRODUCTION
- METHODOLOGY
- RESULTS
- CONCLUSION
- APPENDIX

EXECUTIVE SUMMARY

The capstone project for IBM Data Science Certification is aimed at experimenting using all that was learnt from all the courses covered. The data used in this report was collected through API, and web scrapping. The resource was stored in a database interacted with using SQL. Data Wrangling was used for grouping and aggregations. Interactive Maps with Folium was also used for visualization of landing and launching areas. The final process was creating models for prediction of landing failures or successes.

INTRODUCTION

The goal of this project is to predict if the Falcon 9 first stage will successfully land. This project uses SpaceX as a case study. SpaceX advertises Falcon 9 rocket launches on their website at 62 million dollars, whereas other companies charge upwards of 165 million dollars apiece; much of the savings is due to SpaceX's ability to reuse the first stage. As a result, we attempt to predict whether the first stage will land in order to calculate the cost of a launch. The project's ultimate purpose is to make the knowledge obtained from this exploratory data analysis and predictive model usable if another company wishes to compete with SpaceX for a rocket launch.

METHODOLOGY

- Executive Summary
- Data collection methodology: The data used in this report was collected through SpaceX rest API, and web scrapping
- Perform data wrangling: Pandas and NumPy were used for all data wrangling - grouping and aggregations.
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models

DATA COLLECTION

THE DATA USED IN THIS REPORT WAS COLLECTED THROUGH SPACEX REST API, AND WEB SCRAPPING USING BEAUTIFUL SOUP – A PYTHON MODULE

API RESULT

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPac
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	None
...	...	...	...	...	...	...	...	...	...	...	...	...
89	86	2020-09-03	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3032383ecb6bb234e7ca
90	87	2020-10-06	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3032383ecb6bb234e7ca
91	88	2020-10-18	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3032383ecb6bb234e7ca
92	89	2020-10-24	Falcon 9	15600.0	VLEO	CCSFS SLC 40	True ASDS	3	True	True	True	5e9e3033383ecbb9e534e7cc

BEAUTIFUL SOUP PROCESSING

```
<tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Vers
r/>Booster</a> <sup class="reference" id="cite_ref-booster_11-0"><a href="#cite_note-booster-11">[b]</a></sup>
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-0"><a href="#cite_note-Dragon-12">[c]</a></sup>
</th>
<th scope="col">Payload mass
</th>
<th scope="col">Orbit
</th>
<th scope="col">Customer
</th>
<th scope="col">Launch<br/>outcome
</th>
```

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

```
data = pd.json_normalize(response.json())
```

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

Request to the SpaceX API → Convert json result into data frame → Clean Data and Create Dictionaries → Assign Dictionaries to Data Frame and apply appropriate filters

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...
89	86	2020-09-03	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	2	True	True	True
90	87	2020-10-06	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	3	True	True	True
91	88	2020-10-18	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	6	True	True	True

GitHub

DATA COLLECTION – SPACEX API

https://github.com/idada29/IBM-Capstone-Project/blob/master/Data%20Collection%20using%20API.ipynb

Data Collection - Scraping

TheDataMultiverse

GitHub

https://github.com/idada29/IBM-Capstone-Project/blob/master/Data%20Collection%20using%20Webscrapping.ipynb

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

Pull response from URL

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

Create BeautifulSoup Object

```
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
```

Extract tables Column

Create a data frame by parsing the launch HTML tables

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

Data frame to . CSV

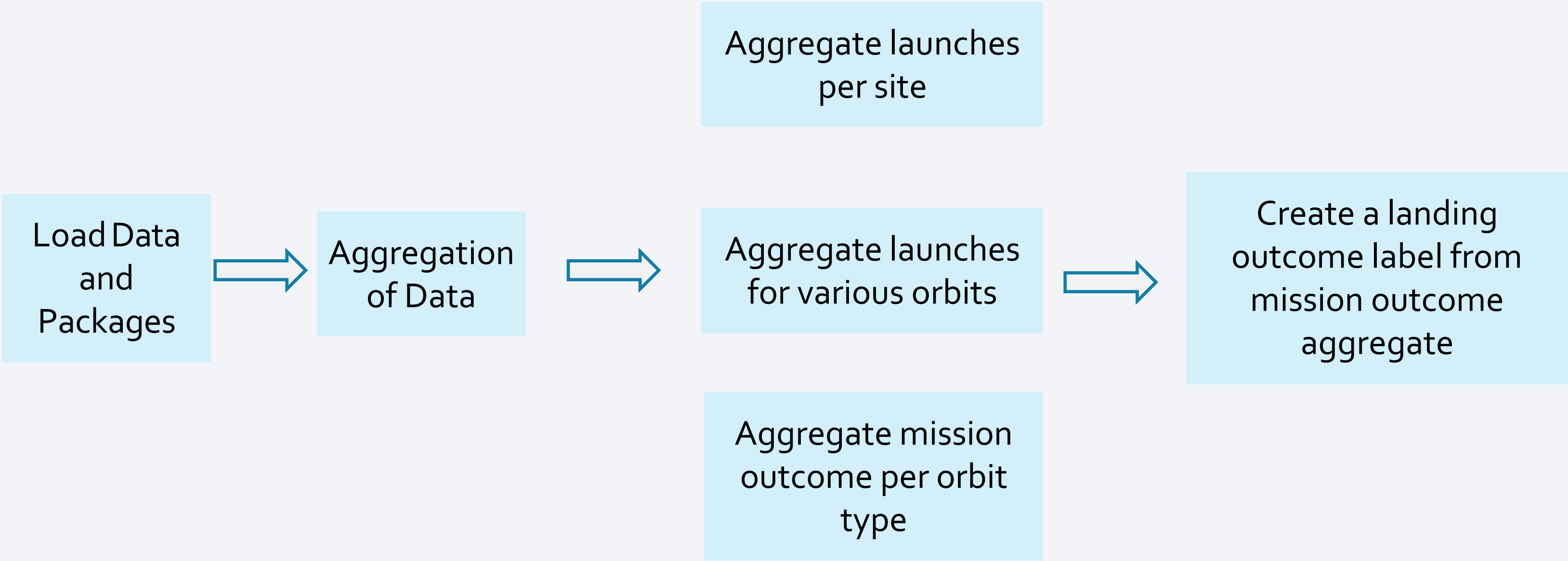
	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857
1	2	2012-05-22	Falcon 9	525.000000	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

TheDataMultiverse

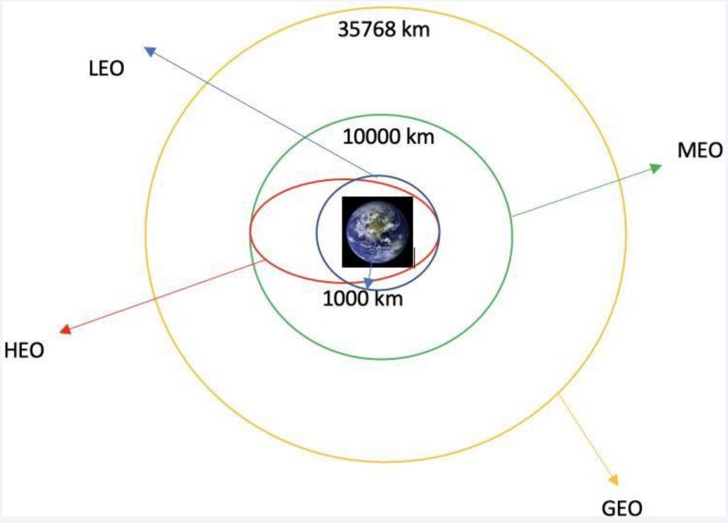
Data Wrangling

https://github.com/idada29/IBM-Capstone-Project/blob/master/EDA%20-%20Data%20Wrangling.ipynb

Data wrangling involves processing the data in various formats like - merging, grouping, concatenating etc. for the purpose of analyzing or getting them ready to be used with another set of data. There have been multiple instances where the booster has failed to land. A landing may have been attempted but failed due to an accident; for example, True Ocean denotes a successful landing to a specific region of the ocean, whereas False Ocean denotes an unsuccessful landing to a specific region of the ocean. True RTLS indicates that the mission's results were successfully landed on a ground pad. False RTLS indicates that the mission was not successfully landed on a ground pad. The mission conclusion was successfully landed on a drone ship, according to true ASDS. The mission outcome was unsuccessfully landed on a drone ship if the ASDS was false. We turned the results into Training Labels, with 1 indicating success and 0 indicating failure.



Each launch aims to a dedicated orbit, and here are some common orbit types:



EDA WITH DATA VISUALIZATION

Scatter Graphs Drawn:

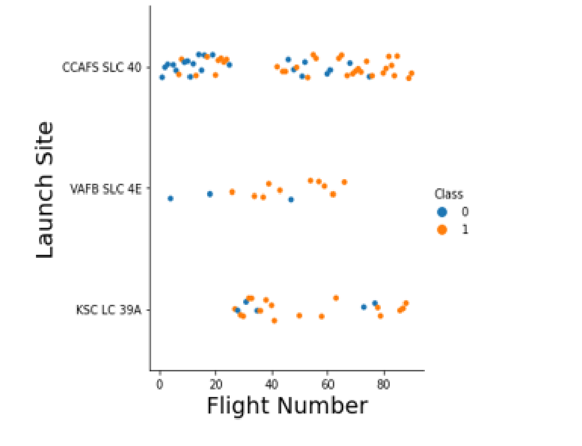
- Payload and Flight Number
- Launch Site and Flight Number
- Payload and Launch Site
- Flight Number and Orbit Type
- Payload and Orbit Type

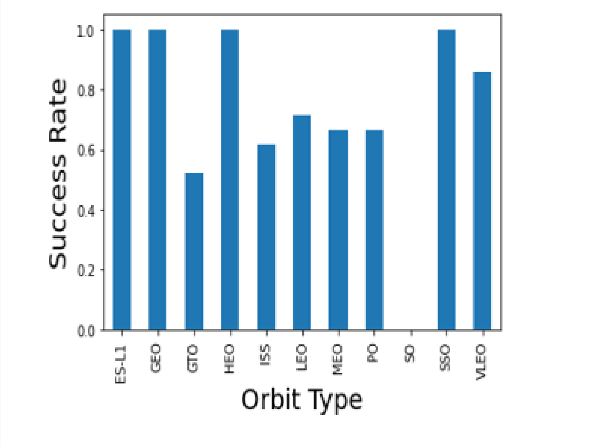
Bar Graph Drawn:

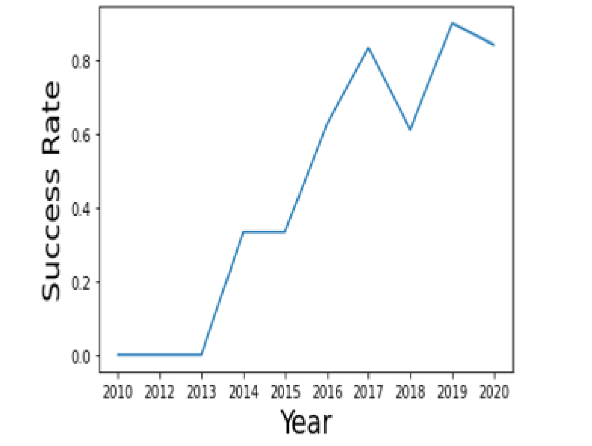
- Visualizing the relationship between success rate of each orbit type

Line Graph Drawn:

- Launch Success Yearly Trend







GitHub

https://github.com/idada29/IBM-Capstone-Project/blob/master/EDA%20with%20Visualization.ipynb

EDA WITH SQL

https://github.com/idada29/IBM-Capstone-Project/blob/master/EDA%20with%20SQL.ipynb

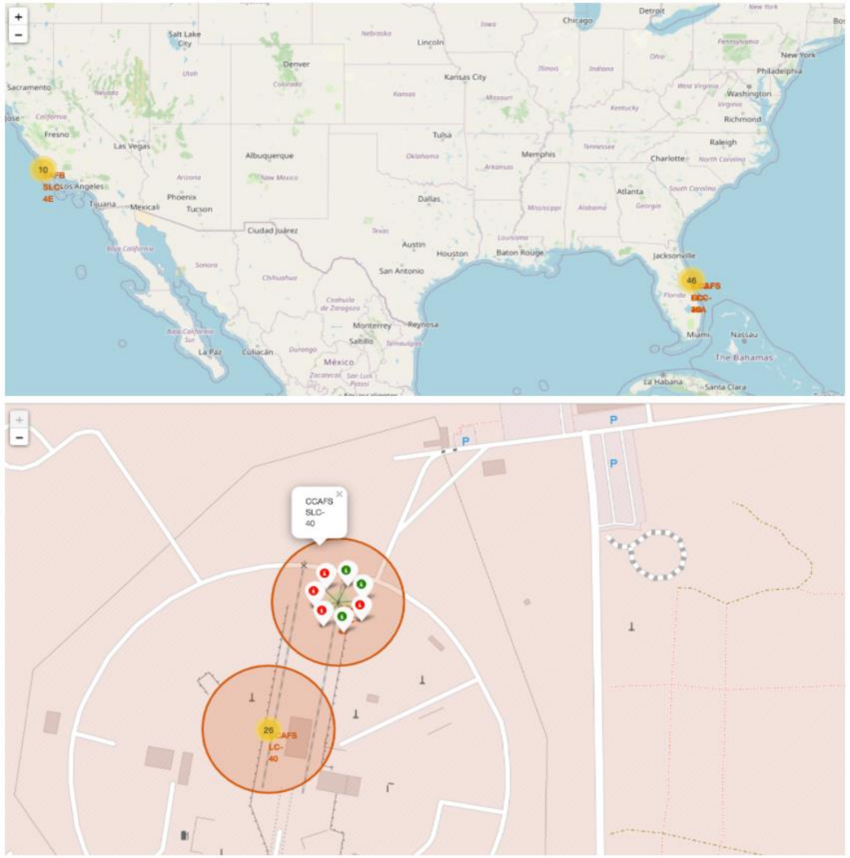
We performed SQL queries to gather information from given dataset:

- Displaying the names of the unique launchsit esin the space mission
- Display 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 F9vl.l
- 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 greaterthan4000 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 failed landing outcomes in drone ship, their booster versions, and launch site names for the year 2015
- Ranking 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 * FROM SPACEXTBL
WHERE (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://yvv08229:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

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



Map Objects, Codes and Results

- Map Marker - folium.Marker() — Map object to make a mark on map.
- Icon Marker - folium.Icon() — Create an icon on map.
- Circle Marker - folium.Circle() — Create a circle where Marker is being placed.
- Polyline - folium.Polyline() — Create a line between points.
- Marker Cluster Object - MarkerCluster() — Simplify the map containing many markers with the same coordinate.
- AntPath - folium.plugins.AntPath() — Create an animated line between points.

https://github.com/idada29/IBM-Capstone-Project/blob/master/Interactive%20Visual%20Analytics%20with%20Folium.ipynb

GitHub

BUILD AN INTERACTIVE MAP WITH FOLIUM

Graphs and Justification

Pie Chart showing the total launches by a certain site/all sites

Chart Justification:

Display relative proportions of multiple classes of data.

Size of the circle can be made proportional to the total quantity it represents.

Scatter Graph showing the relationship with Outcome and Payload Mass (Kg) for the different Booster Versions

Chart Justification:

It shows the relationship between two variables

It is the best method to show you a non-linear pattern.

Plots and interactions

1. Dash and it's components - *import* dash
2. Pandas - import pandas as pd — Fetching values from CSV and creating a dataframe
3. Plotly - import plotly.express as px — Plot the graphs with interactive plotly library
4. Dropdown - dcc.Dropdown() — Create a dropdown for launch sites
5. Rangeslider - dcc.RangeSlider() — Create a range slider for Payload Mass range selection
6. Pie Chart - px.pie() — Creating the Pie graph for Success percentage display
7. Scatter Chart - px.scatter() — Creating the Scatter graph for correlation display

GitHub https://github.com/idada29/IBM-Capstone-Project/blob/master/spacex_dash_app.py

Build a Dashboard with Plotly Dash

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.

EVALUATING MODEL

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

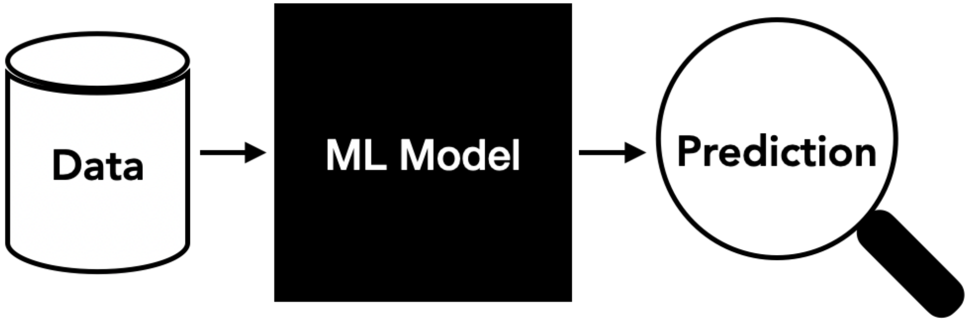
IMPROVING MODEL

- 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.

https://github.com/idada29/IBM-Capstone-Project/blob/master/Machine%20Learning%20Prediction.ipynb



Predictive Analysis (Classification)

RESULTS

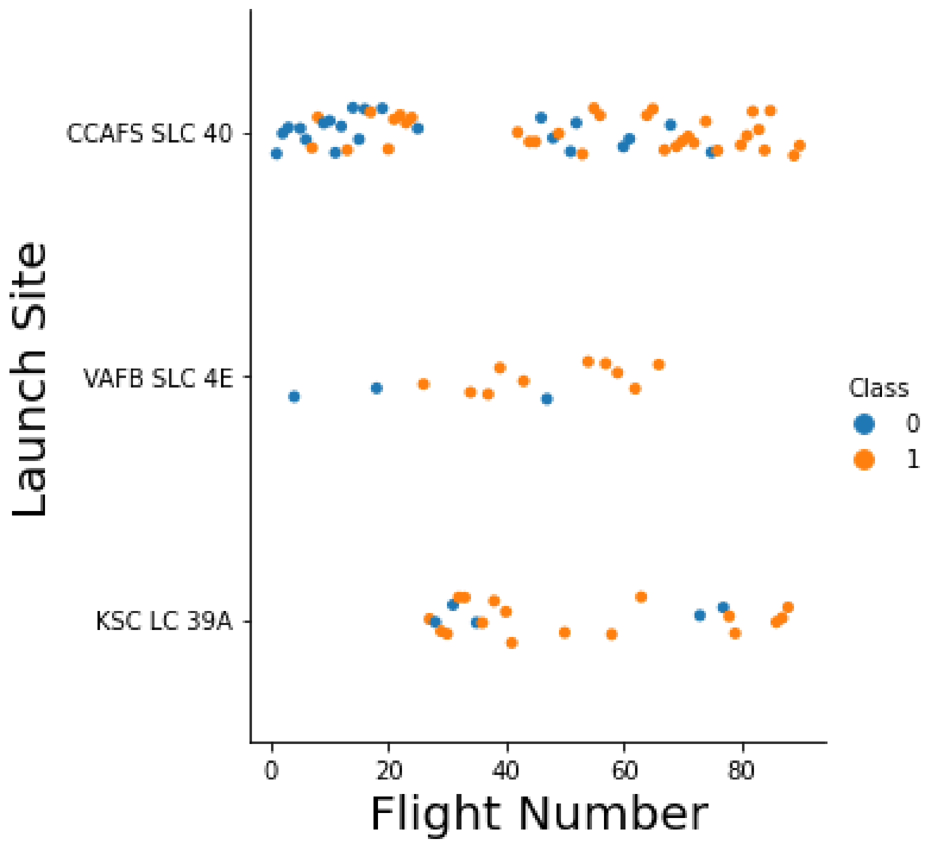
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

FLIGHT NUMBER VS. LAUNCH SITE

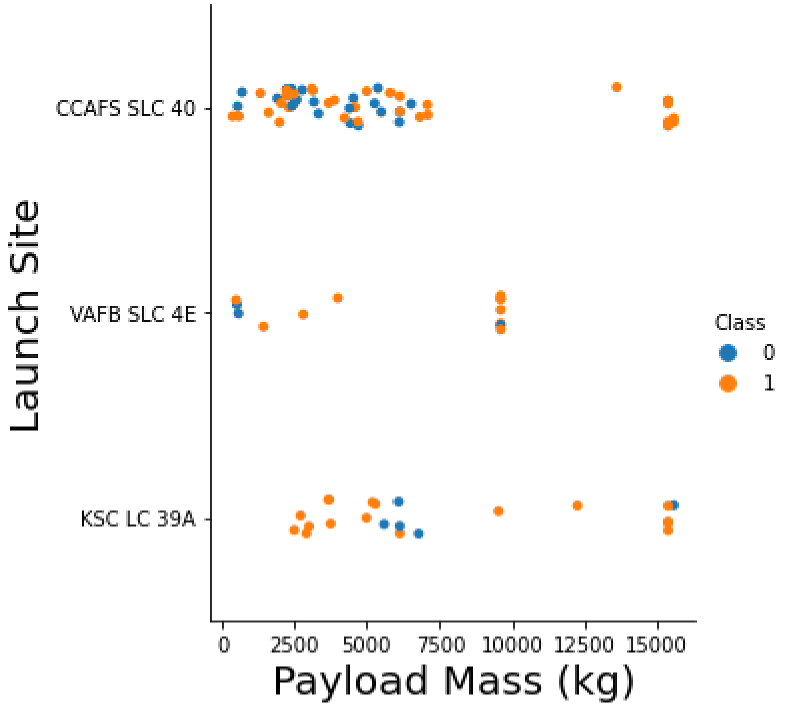
I. There are higher success rate for flights launching attempts within the range if 60-80 for CCAFS SLC 40
II. Approximately 76.9% of VAFB SLC 4E landed successfully.
III. KSC LC 39A has the highest minimum flight number launched amongst the three launch site.



TheDataMultiverse

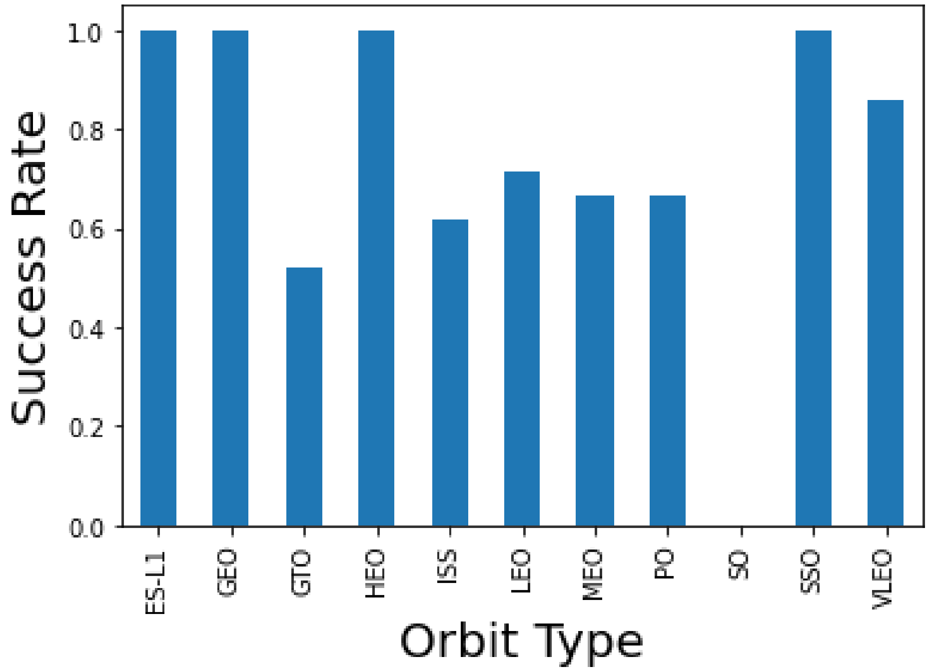
PAYLOAD VS. LAUNCH SITE

I. There are no rockets launched in VAFB SLC 4E with heavy payload mass (greater than 10000).
II. More unsuccessfully experienced in CCAFS SLC 40 compared to higher payload mass with more success rate.
III. Most unsuccessful launch in KSC LC 39A has a payload mass between 5000 -7500

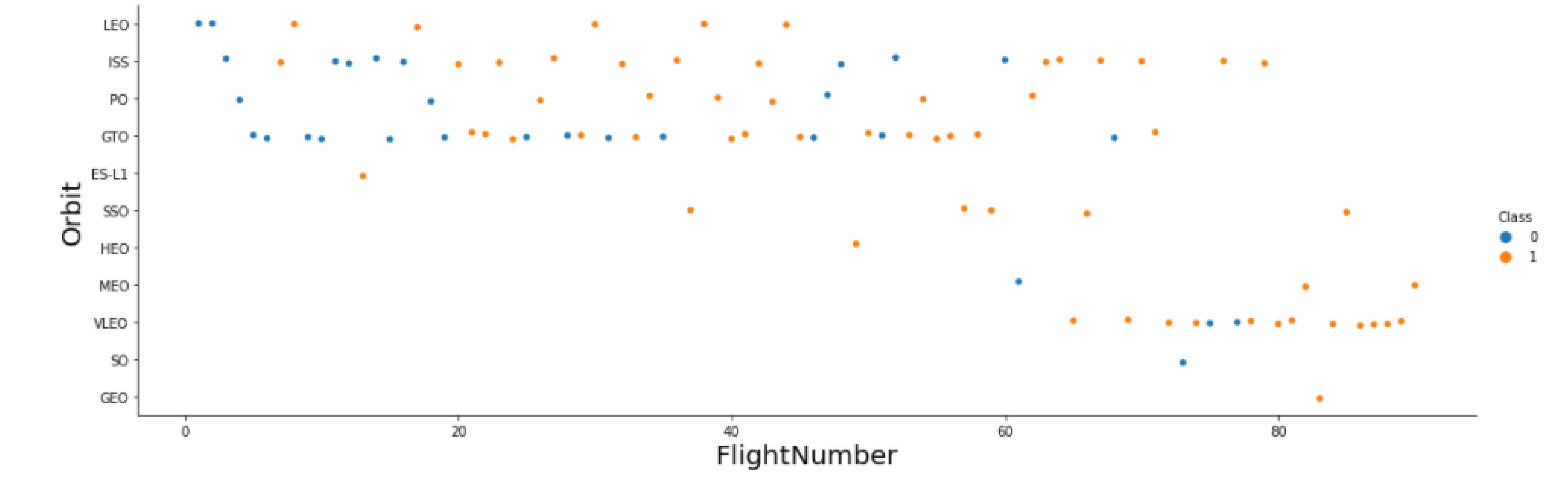


SUCCESS RATE VS. ORBIT TYPE

I. There is maximum success rate with ES.L1,GEO,HEO,SSO orbits
II. Least landing success rate with flights to SO orbit with 0% chance.
III. There is a total of 11 orbit type to monitor within the SpaceX flight exposition.



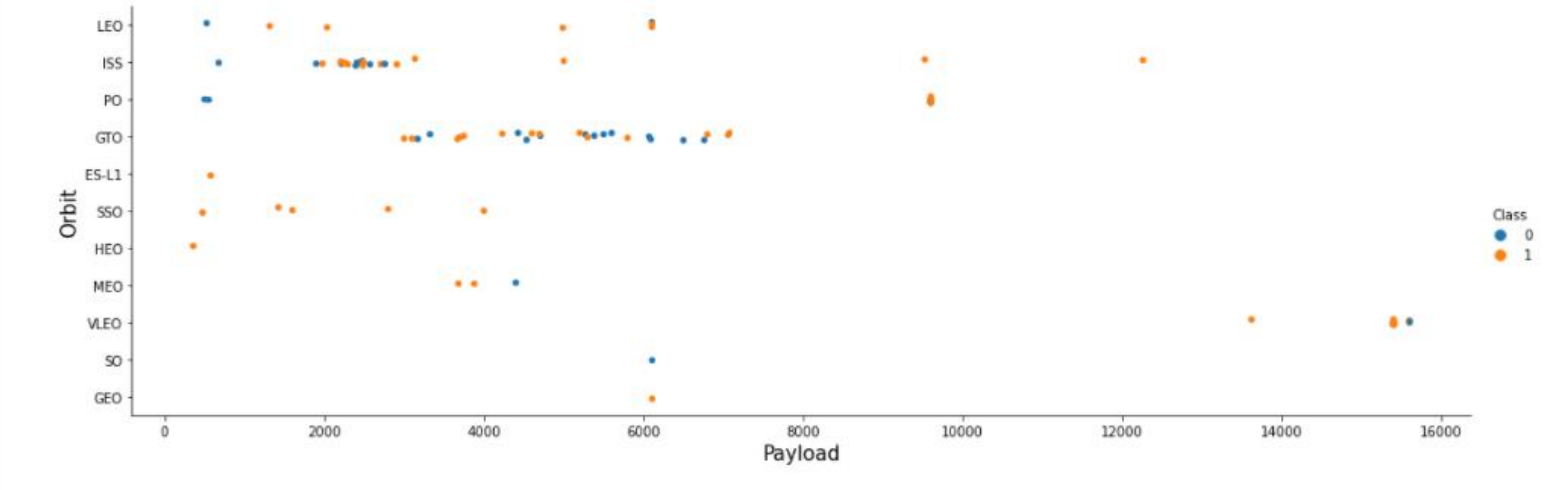
Flight Number vs. Orbit Type



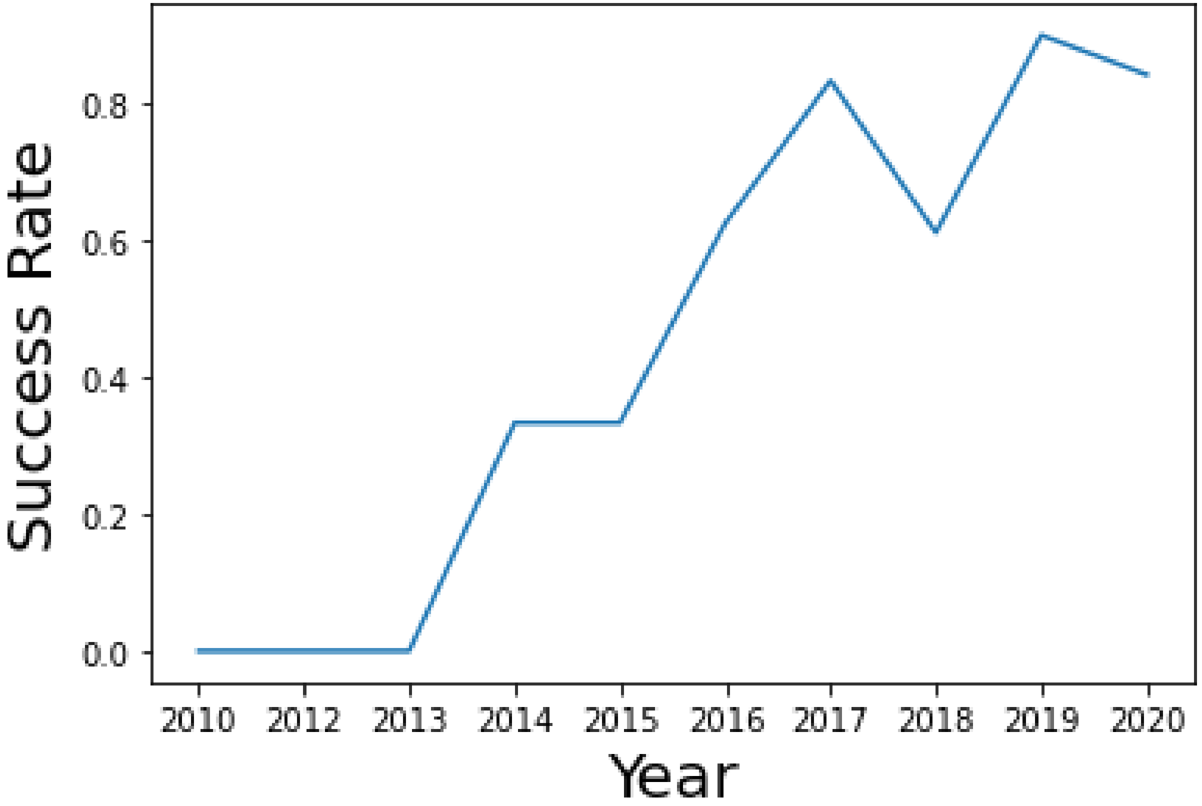
You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

TheDataMultiverse

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

1. The success rate since 2013 kept increasing till 2020
2. The most consistent stable rise of successful landing occurred between 2015-2017

All Launch Site Names

```
%sql select Unique(LAUNCH_SITE) from SPACEXTBL;
```

```
 * ibm_db_sa://[redacted]:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

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

QUERY EXPLAINATION

Using the word ***Unique*** in the query means that it will only show Unique values in the ***Launch_Site*** column from ***SPACEXTBL***

Launch Site Names Begin with 'CCA'

```
%%sql SELECT * FROM SPACEXTBL
WHERE (LAUNCH_SITE) LIKE 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://[redacted]:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

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

The word LIMIT 5 in the query signifies that it will only return the first 5 entries from SPACEXTBL, and the LIKE keyword has the words 'CCA %' as a wild card. The final % indicates that the Launch Site name must begin with CCA.

Total Payload Mass

```
%%sql select
      SUM(PAYLOAD_MASS__KG_) as payloadmass
      from SPACEXTBL
      WHERE Customer = 'NASA (CRS)';
```

```
 * ibm_db_sa://[redacted]***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

]:

payloadmass
45596

QUERY EXPLAINATION

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)***

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_) as payloadmass
      from SPACEXTBL
      WHERE (Booster_Version) LIKE 'F9 v1.1';
```

```
 * ibm_db_sa://[redacted]:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

1]:

payloadmass
2928

QUERY EXPLAINATION

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***

First Successful Ground Landing Date

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

```
%sql select min(DATE) from SPACEXTBL where Landing__Outcome = 'Success (ground pad)';
```

```
 * ibm_db_sa://[redacted]:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

2]:

1
2015-12-22

QUERY EXPLAINATION

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)***

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_ BETWEEN 4000 and 6000;
```

```
 * ibm_db_sa://[redacted]:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

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

QUERY EXPLAINATION

Select only Booster_Version . The WHERE clause filters the dataset to Landing_Outcome = Success (drone ship)

The WHERE clause specifies additional filter conditions
WHERE LANDING__OUTCOME='Success (drone ship)' and PAYLOAD_MASS__KG_ BETWEEN 4000 and 6000;

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

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

```
 * ibm_db_sa://[redacted]:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

mission_outcome	missionoutcomes
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

QUERY EXPLAINATION

Select Mission_outcome and count of the mission outcome. GROUP BY mission_outcome;

Boosters Carried Maximum Payload

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

SELECT the Booster_version from the table.
WHERE clause introduces a sub query which selects just the max payload_mass_kg from the table

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

```
 * ibm_db_sa://[redacted]***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

boosterversion
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

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

```
%sql select date, booster_version, launch_site, landing__outcome from spacextbl where landing__outcome ='Failure (drone ship)' and year(date) ='2015';
```

```
 * ibm_db_sa://[redacted]**@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

1]:

DATE	booster_version	launch_site	landing__outcome
2015-01-10	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
2015-04-14	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

SELECT date, booster_version, launch_site, landing__outcome from the table.
WHERE clause introduces a filter which selects just landing__outcome = 'Failure (drone ship)' and year(date) ='2015'

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

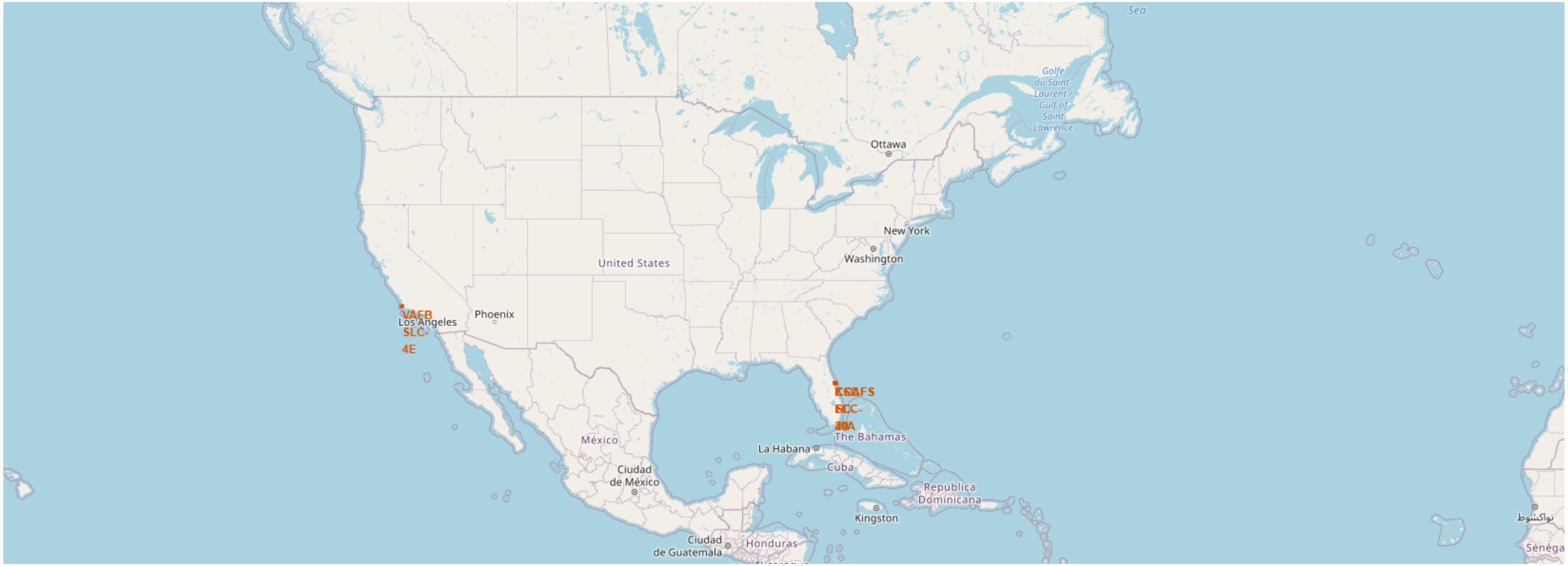
```
%%sql  SELECT LANDING__OUTCOME, COUNT(LANDING__OUTCOME) AS COUNTING FROM SPACEXTBL
        WHERE DATE >= '2010-06-04' AND DATE <= '2017-03-20'
        GROUP BY LANDING__OUTCOME
        ORDER BY COUNTING DESC
        ;
```

```
 * ibm_db_sa://[redacted]:***@824dfd4d-99de-440d-9991-629c01b3832d.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:30119/bludb
Done.
```

landing__outcome	counting
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

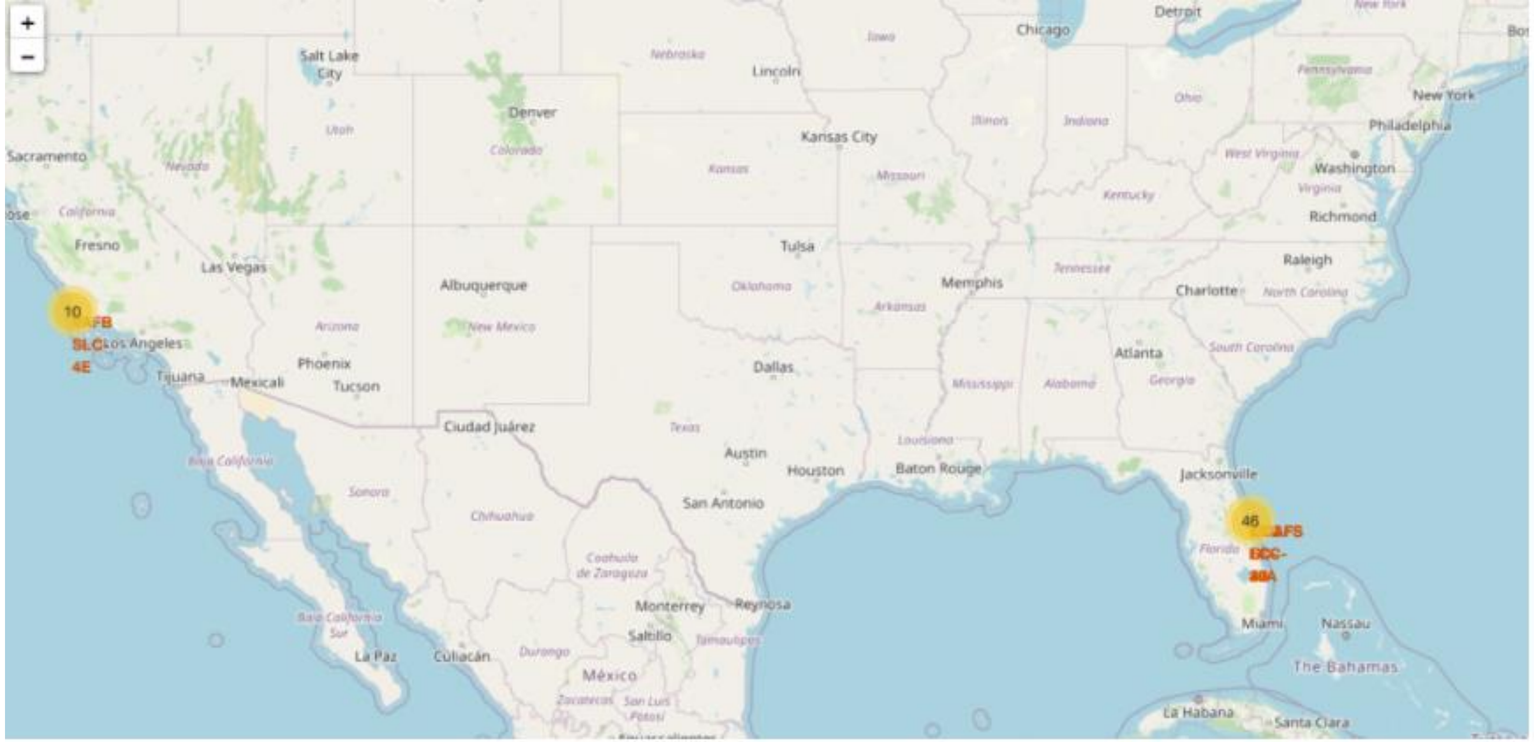
SELECT landing outcome, count landing__outcome as counting from the table.
WHERE clause introduces a filter which selects a date limit between 2010 to 2017, grouped by the landing outcome and the order of the counting was set to descending order.

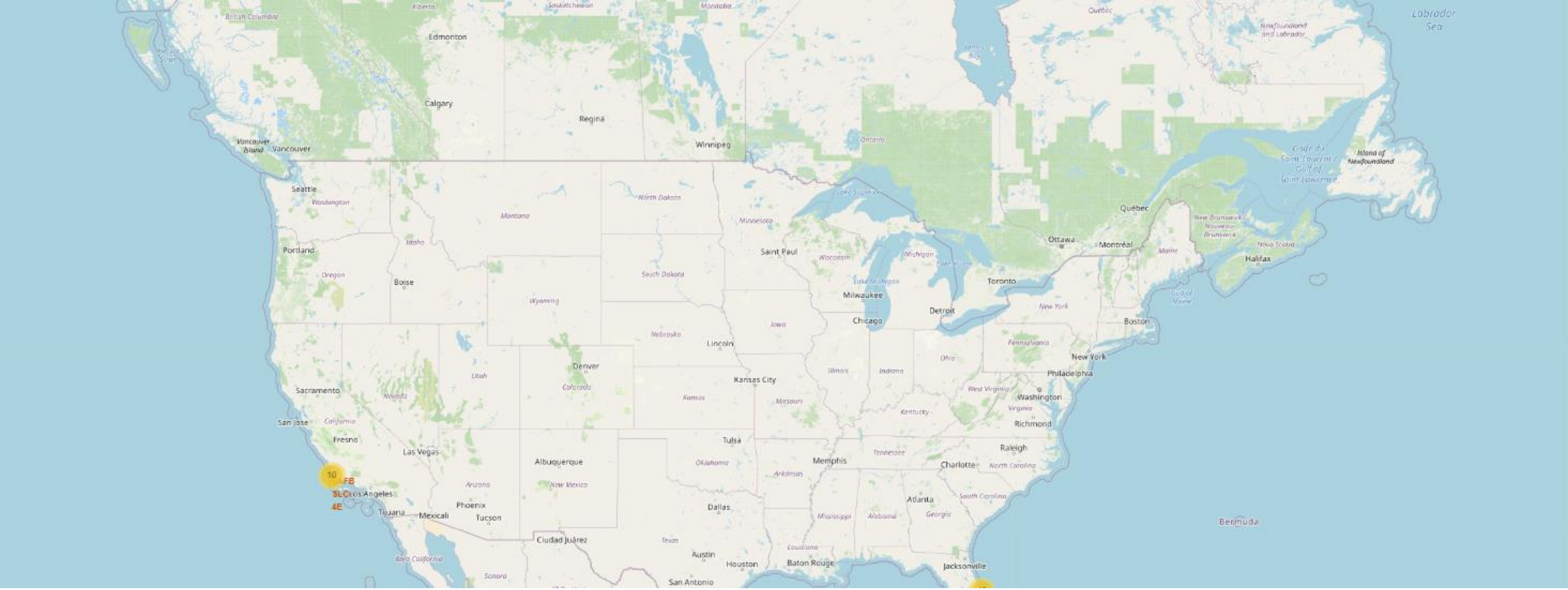

TheDataMultiverse
Section 3



FOLIUM MAP SCREENSHOT 1

FOLIUM MAP SCREENSHOT 2

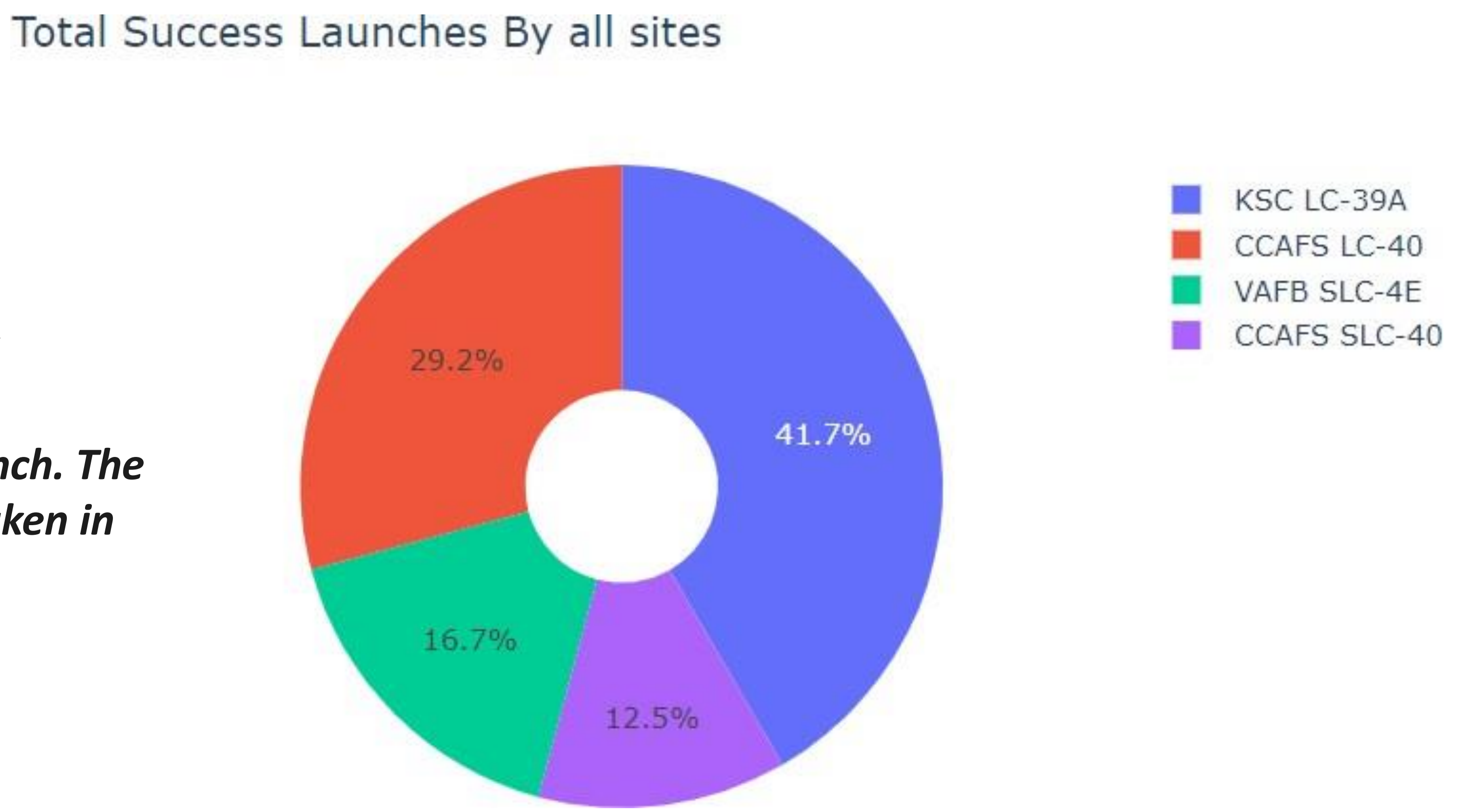




FOLIUM MAP SCREENSHOT 3

Section 4

Build a Dashboard with Plotly Dash



KSC LC-39A had the most successful launches from all the sites with approximately 42% of the total launch. The least successful launch was undertaken in the CCAFS SLC-40 launch site

VISUALIZATION OF ALL LAUNCH SUCCESS COUNT

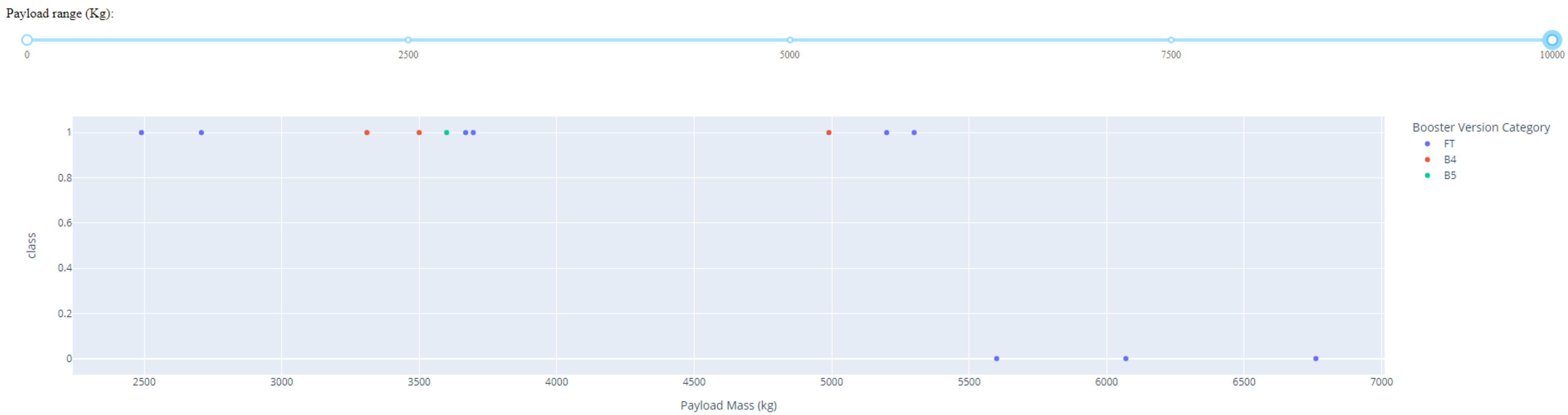
SpaceX Launch Records Dashboard

KSC LC-39A

Total Launch for a Specific Site

The success rate in KSC LC-39A launch site is approximately 77 percent and failures are around 23 percent.

LAUNCH SITE SUCCESS RATIO



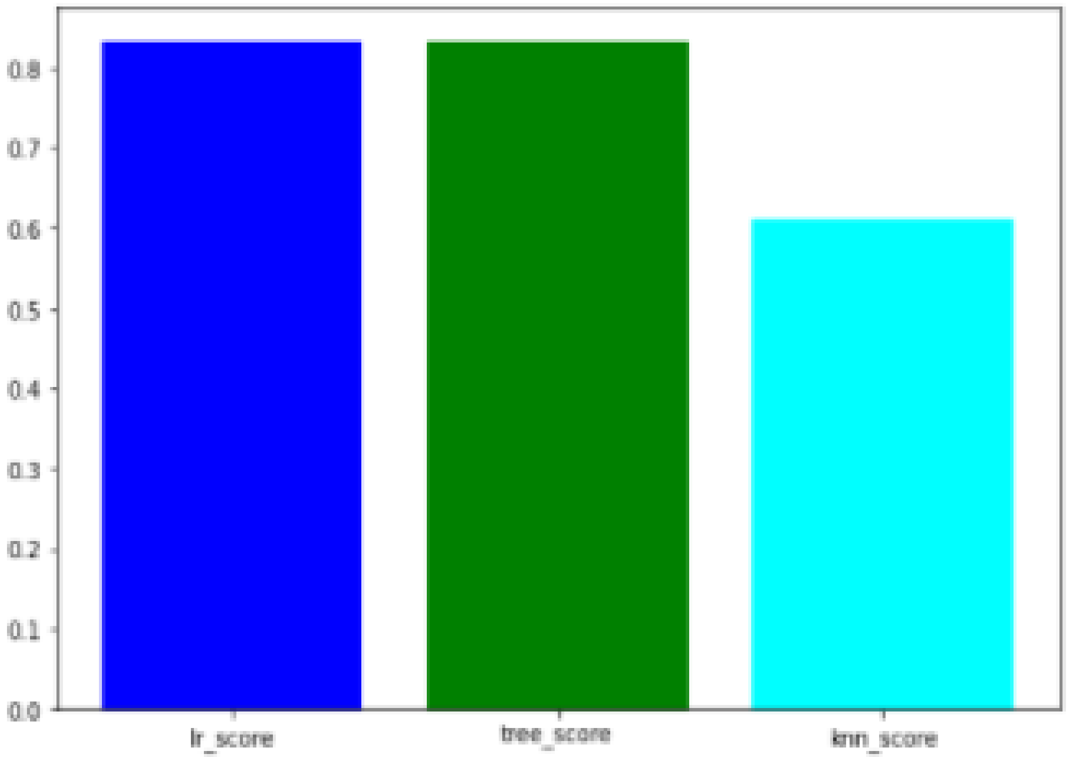
Lesser payload mass has higher success rate as compared to higher weighted payload

PAYLOAD VS. LAUNCH OUTCOME SCATTER PLOT

Section 5

Predictive Analysis (Classification)

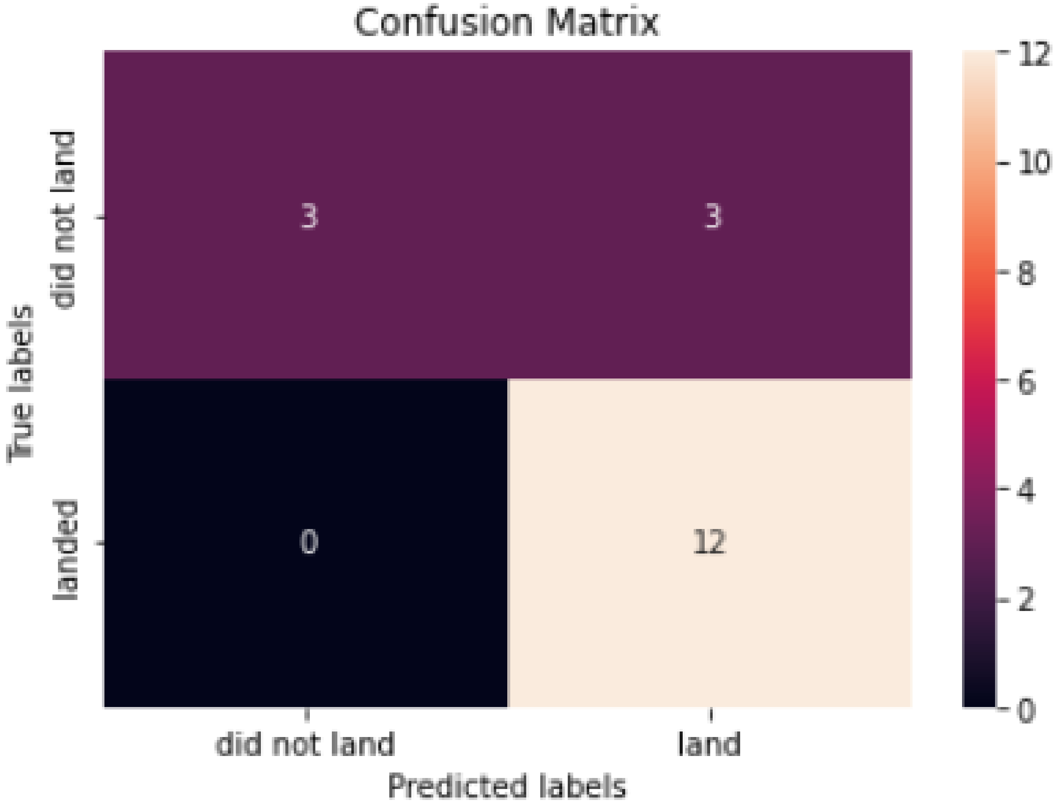
CLASSIFICATION ACCURACY



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

Visualize the built model accuracy for all built classification models, in a bar chart

The logistic regressor had the best accuracy on the classification of successes or failures



CONFUSION MATRIX FOR LOGISTIC REGRESSOR

Examining the confusion matrix, we see that the regressor distinguishes between the different classes. The major challenge is with the false prediction on launches that didn't land but was classified as "landed"

CONCLUSIONS

1. Investment of launched SpaceX for KSC LC-39A launch site is more likely to yield as the success rate is high

2. Using lesser payload mass on the launch increases the possibility for success as compared to higher weighted payload

3. Historically drone ships have had high success ratio on landing compared to other landing sites. More of such landing sites should be considered.

4. There are higher success rate for flights launching attempts within the range if 60-80 for CCAFS SLC 40

Appendix

```
# Create an app layout
app.layout = html.Div(children=[html.H1('SpaceX Launch Records Dashboard',
                                        style={'textAlign': 'center', 'color': '#503D36',
                                               'font-size': 40}),
                                # TASK 1: Add a dropdown list to enable Launch Site selection
                                # The default select value is for ALL sites
                                dcc.Dropdown(id='site-dropdown',
                                             options=[
                                                      {'label': 'ALL SITES', 'value': 'ALL'},
                                                      {'label': 'CCAFS LC-40', 'value': 'CCAFS LC-40'},
                                                      {'label': 'VAFB SLC-4E', 'value': 'VAFB SLC-4E'},
                                                      {'label': 'KSC LC-39A', 'value': 'KSC LC-39A'},
                                                      {'label': 'CCAFS SLC-40', 'value': 'CCAFS SLC-40'}
                                                  ],
                                             value='ALL',
                                             placeholder="Select a Launch Site here",
                                             searchable=True),
                                html.Br(),

                                # TASK 2: Add a pie chart to show the total successful launches count for all sites
                                # If a specific launch site was selected, show the Success vs. Failed counts for the site
                                html.Div(dcc.Graph(id='success-pie-chart')),
                                html.Br(),

                                html.P("Payload range (Kg):"),
                                # TASK 3: Add a slider to select payload range
                                dcc.RangeSlider(id='payload-slider',
                                                min=0,max=10000,step=1000,
                                                value=[min_payload,max_payload],
                                                marks={0: '0', 2500:'2500', 5000:'5000',
```

Thank you!