IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Hafizuddin Bin Aminuddin

03 July 2023

Outline

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

Executive Summary

- Summary of methodologies
    - Data collection
    - Data wrangling
    - Exploratory Data Analysis with data visualization
    - Exploratory Data Analysis with SQL
    - Building interactive map with Folium
    - Building interactive dashboard with Plotly Dash
    - Predictive analysis using machine learning

- Summary of all results
    - Exploratory Data Analysis results
    - Interactive analytic results
    - Predictive analysis results

Introduction

Project background and context

SpaceX

- Advertises Falcon 9 rocket launches on its website
- A cost of 62 million dollars as compared to other providers at 165 million dollars
- Savings due to feasibility of reusing first stage.

Problems you want to find answers

- What factors affecting the success rate for rocket landing?
- To predict the feasibility of SpaceX Falcon 9 landing successfully.

Section 1

Methodology

Methodology

Executive Summary

Data collection methodology

- Using SpaceX API
- Web scraping from Wikipedia

Perform data wrangling

- Implementation of one-hot encoding to categorical features

Perform exploratory data analysis (EDA)

- Using Visualisation
- Using SQL

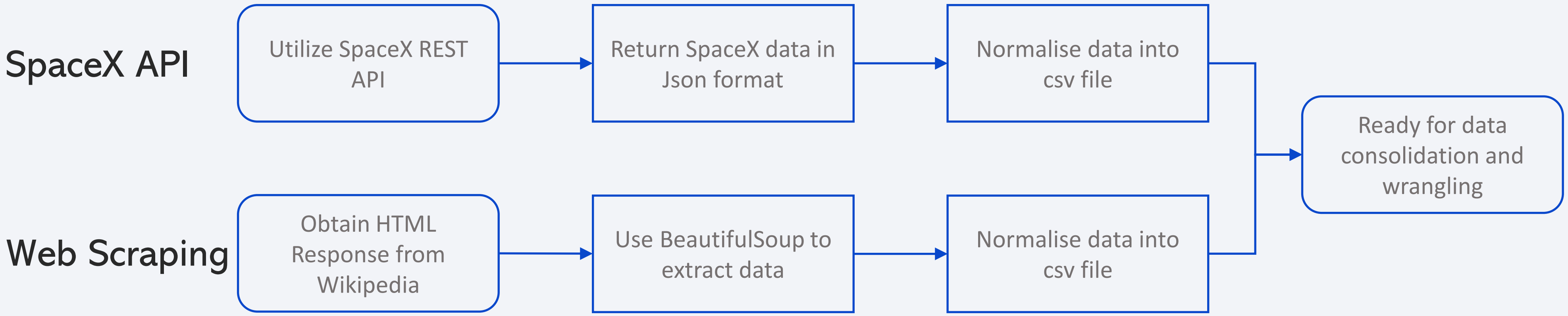
Perform interactive visual analytics

- Using Folium
- Using Plotly Dash

Perform predictive analysis

- Using classification models
- Build, tune and evaluate classification models

Data Collection



SpaceX API

- Obtain request to SpaceX API
- Convert content into Json format with .json() function and correspondingly into pandas dataframe with .json_normalize()
- Clean the data, checked for missing values and substitute them where applicable.

Web Scraping

- Use BeautifulSoup to extract the data for Falcon 9 from Wikipedia.
- Extract as an HTML table, then parse and transform to a pandas dataframe.

Data Collection – SpaceX API

Obtain request for rocket launch data using API

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

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

Convert json result into pandas dataframe using json_normalize

```
In [12]: # Use json_normalize meethod to convert the json result into a dataframe
         data = pd.json_normalize(response.json())
```

Perform data checking, filling missing values where applicable

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

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

# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'].fillna(value=meanPayloadMass, inplace=True)

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

Link to the notebook as below:

https://github.com/Hafiz4869/ibm_project/blob/0cf6138215f2282a81a536a7fadeac076e7245e6/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

Perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
In [16]:
# use requests.get() method with the provided static_url
# assign the response to a object

responsehttp = requests.get(static_url)
```

Create from the HTML response a BeautifulSoup object

```
In [17]:
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content

soup = BeautifulSoup(responsehttp.content, 'html.parser')
```

Extract all column/variable names from the HTML table header

```
In [19]:
# 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')
```

Create a data frame by parsing the launch HTML tables

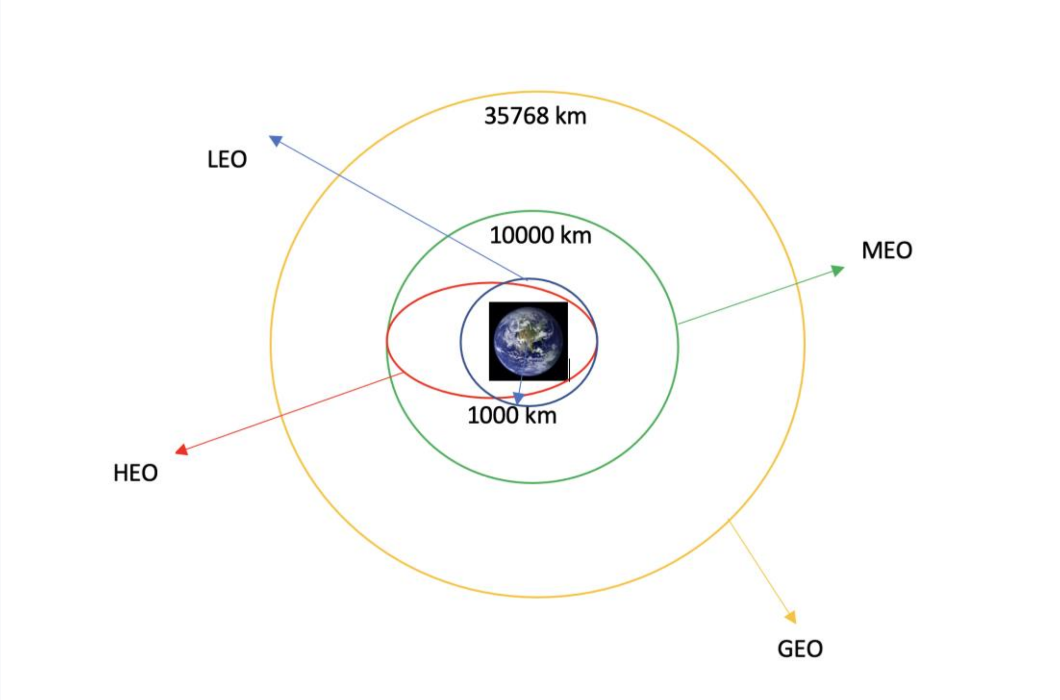
```
In [23]:
launch_dict= dict.fromkeys(column_names)

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

# Let's initial the launch_dict with each value to be an empty list
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'] = []
# Added some new columns
launch_dict['Version Booster']=[]
launch_dict['Booster landing']=[]
launch_dict['Date']=[]
launch_dict['Time']=[]
```

```
In [25]:
df=pd.DataFrame(launch_dict)
```

Link to the notebook as below:

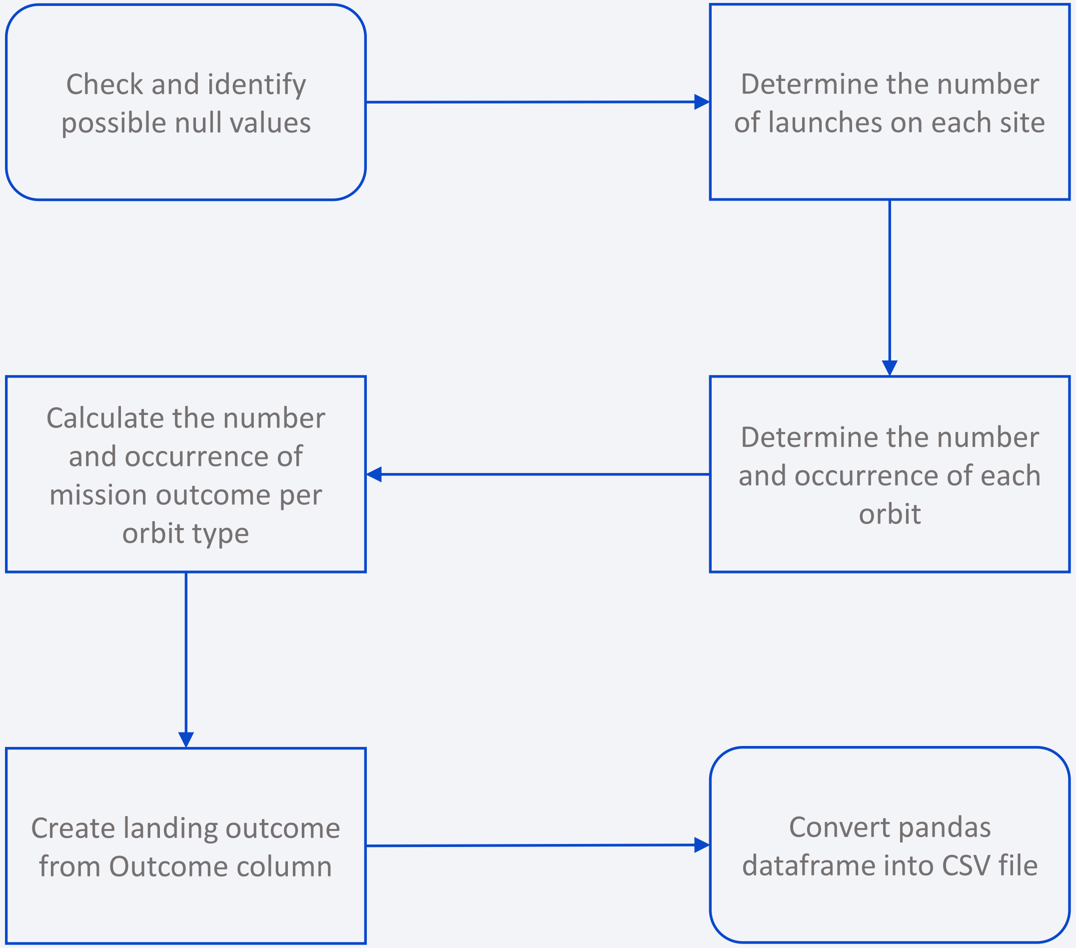
https://github.com/Hafiz4869/ibm_project/blob/0cf6138215f2282a81a536a7fadeac076e7245e6/jupyter-labs-webscraping.ipynb

Data Wrangling

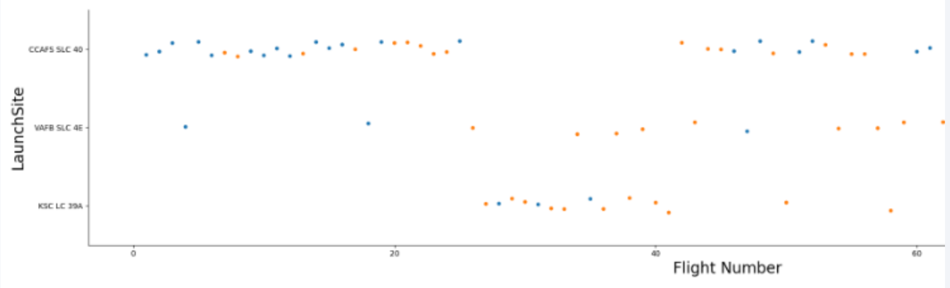


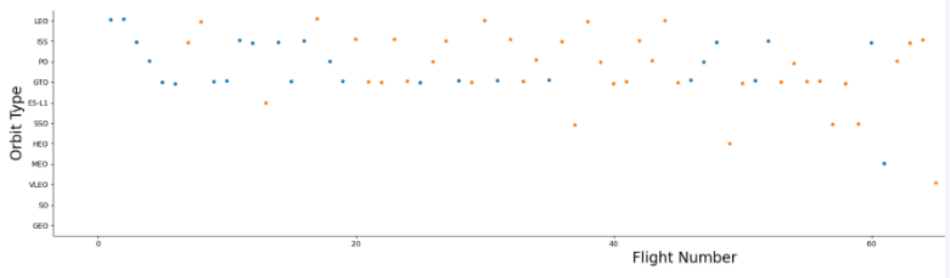
Link to the notebook as below:

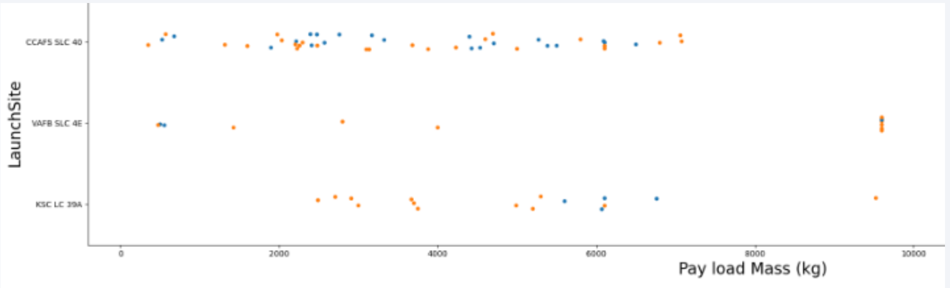
https://github.com/Hafiz4869/ibm_project/blob/0cf6138215f2282a81a536a7fadeac076e7245e6/IBM-DS0321EN-SkillsNetwork_labs_module_1_L3_labs-jupyter-spacex-data_wrangling_jupyterlite.jupyterlite.ipynb

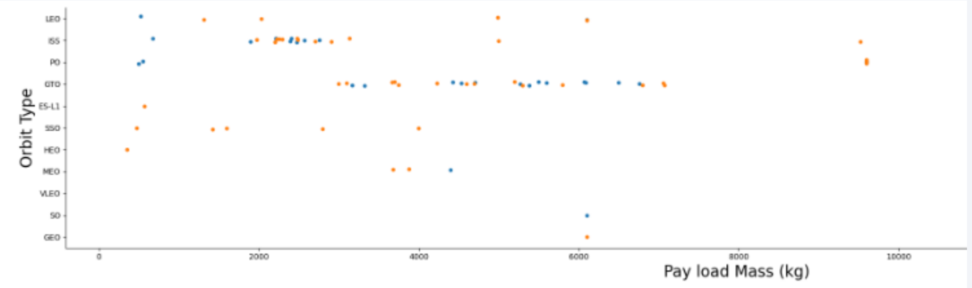


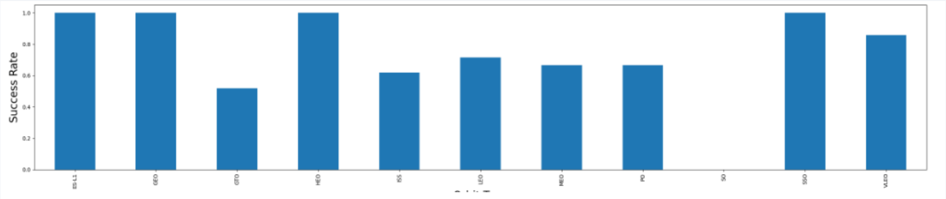
EDA with Data Visualization

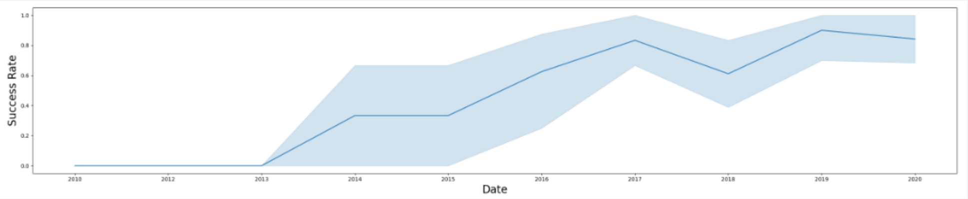












- Scatter plot used to visualize correlation between flight number, orbit type and payload mass.
- Bar chart used to visualize correlation between orbit type and success rate
- Line graph used to visualize the yearly trend against success rate

https://github.com/Hafiz4869/ibm_project/blob/0cf6138215f2282a81a536a7fadeac076e7245e6/IBM-DS0321EN-SkillsNetwork_labs_module_2_jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

EDA with SQL

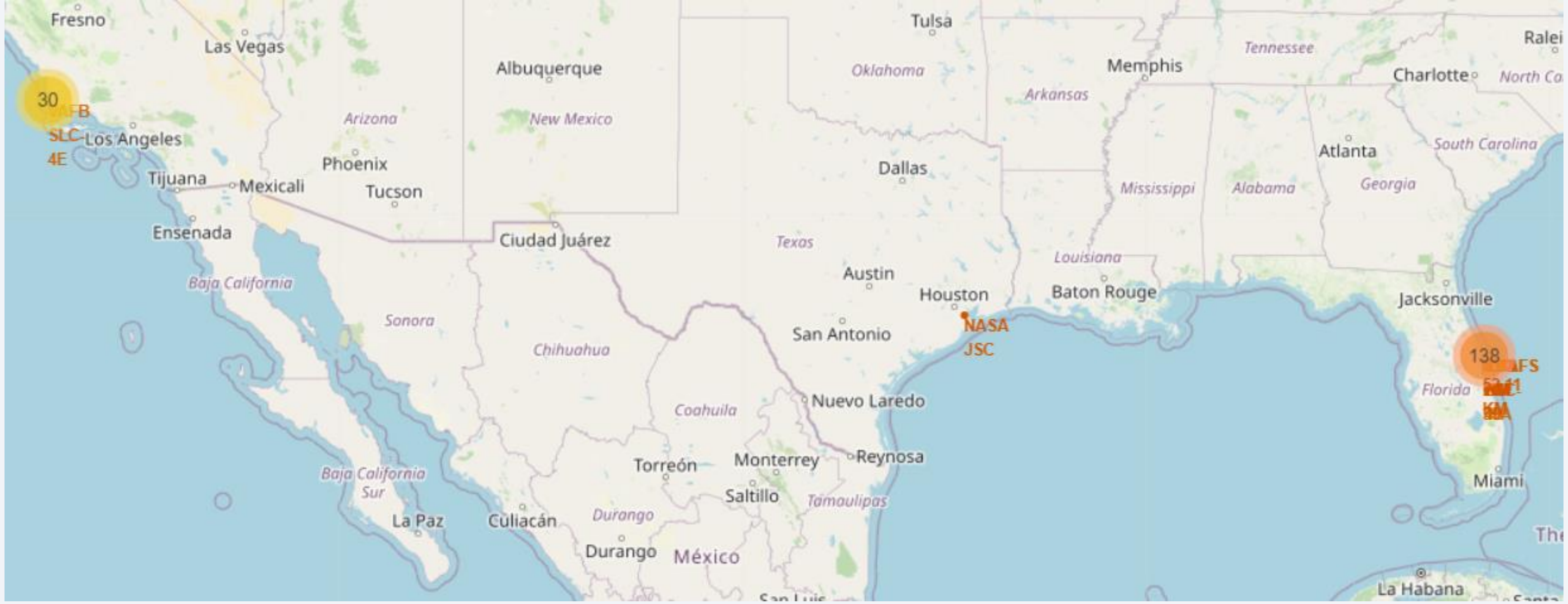
The queries initiated for EDA via SQL:

- The names of unique launch sites in the space mission.
- Records where launch sites begin with 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.
- List of names for the boosters which have success in drone ship and have payload mass greater than 4,000 but less than 6,000.
- Total number of successful and failure mission outcomes.
- Names of the booster version which have carried the maximum payload mass.
- Records for months, failed landing outcome in drone ship with their responding booster version and launch site.
- Ranking the landing outcomes in descending order within the stipulated period.

Link to the notebook as below:

https://github.com/Hafiz4869/ibm_project/blob/0cf6138215f2282a81a536a7fadeac076e7245e6/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium



Map objects such as markers, circles, lines are added into the map with the objective of **determining the optimal location for launching sites**.

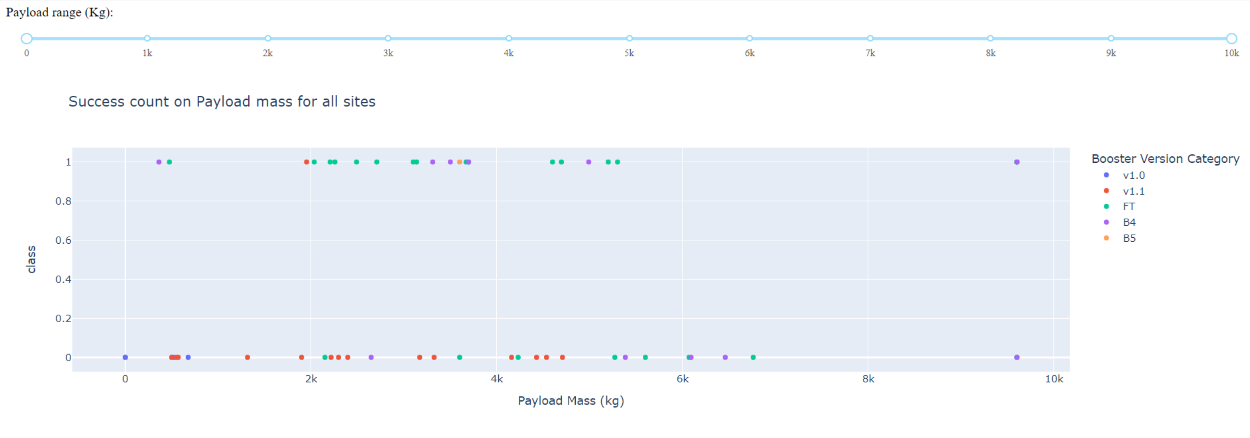
Colour-labeled markers enable to determine location with high potential for success rate for rocket landing.

Link to the notebook as below:
https://github.com/Hafiz4869/ibm_project/blob/0cf6138215f2282a81a536a7fadeac076e7245e6/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

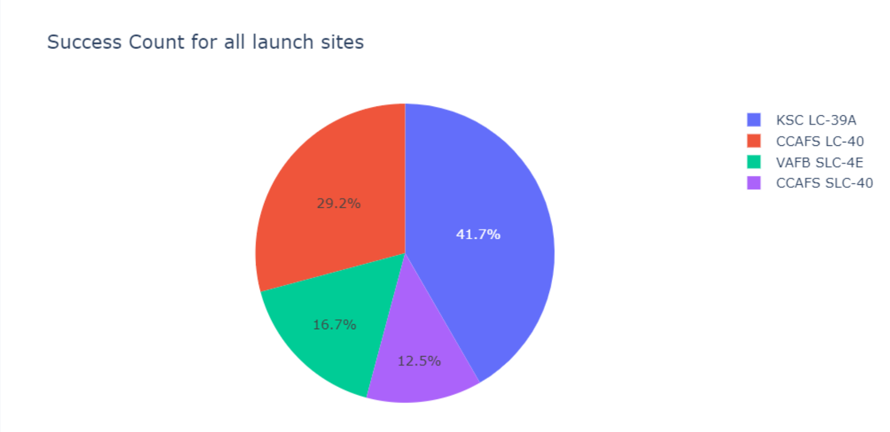
Interactive Dashboard

Scatter plot



Able to observe the correlation between Payload Mass (kg) and success count outcome for different booster version.

Pie chart

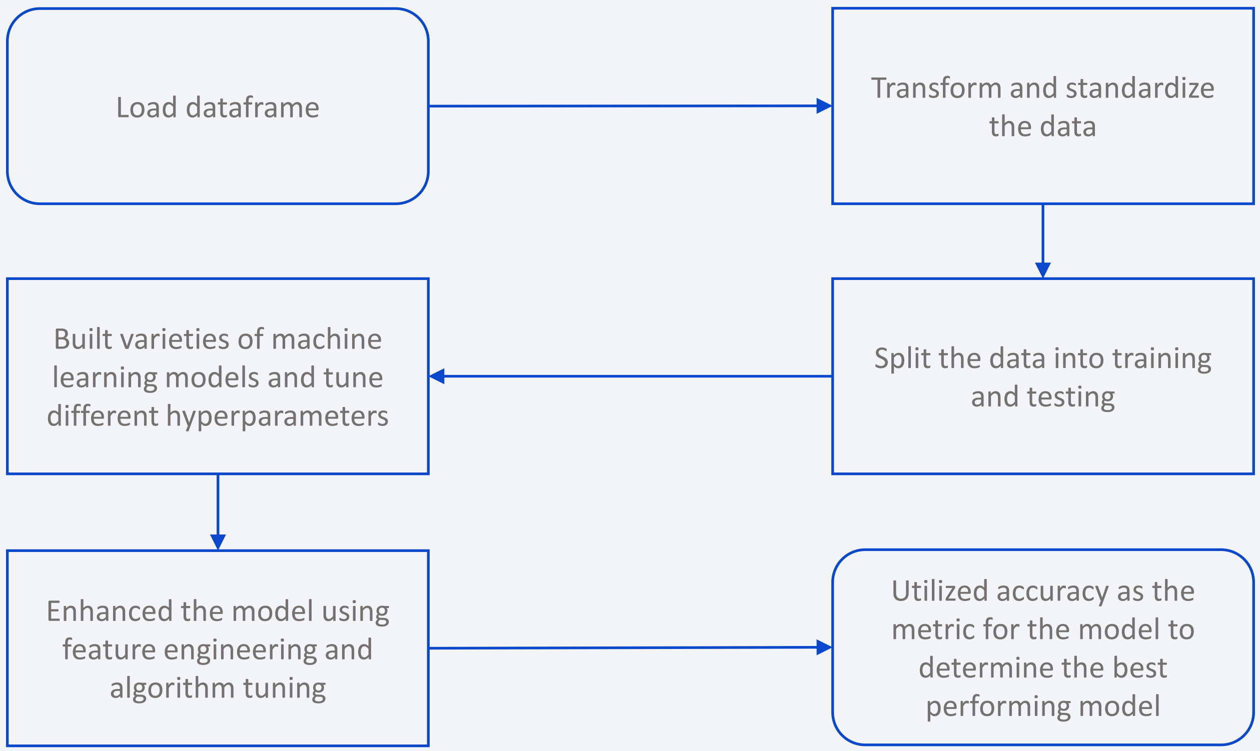


Comparison of success counts launch based on different sites.

Link to the notebook as below:
https://github.com/Hafiz4869/ibm_project/blob/fef661c391fc1c4cc359478fd68c462b2b711b5c/spaceX_app

Predictive Analysis (Classification)

Model Development Process



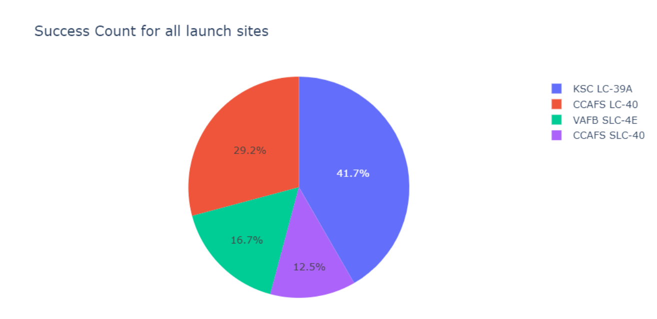
Link to the notebook as below:
https://github.com/Hafiz4869/ibm_project/blob/fef661c391fc1c4cc359478fd68c462b2b711b5c/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

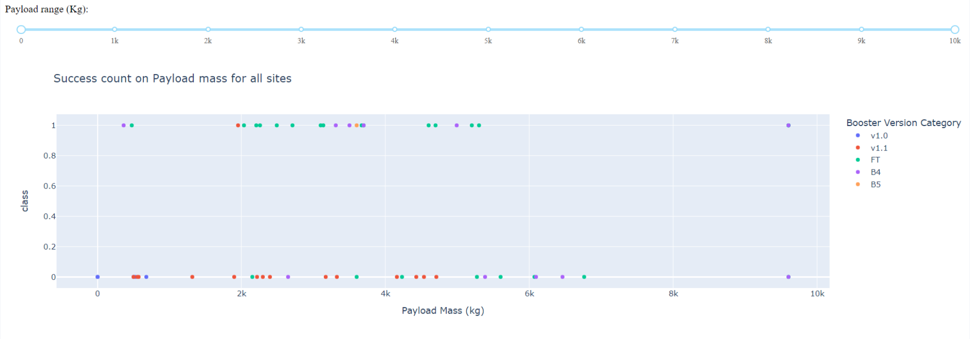
Results

Exploratory data analysis results

- Low payload mass has a higher success rate than the high payload mass.
- Both KSC LC – 39A and VAFB SLC 4E have the best success rate for launching.
- The orbit type which has high success rate are ES-L1, GEO, HEO, and SSO.

Interactive analytics demo in screenshots





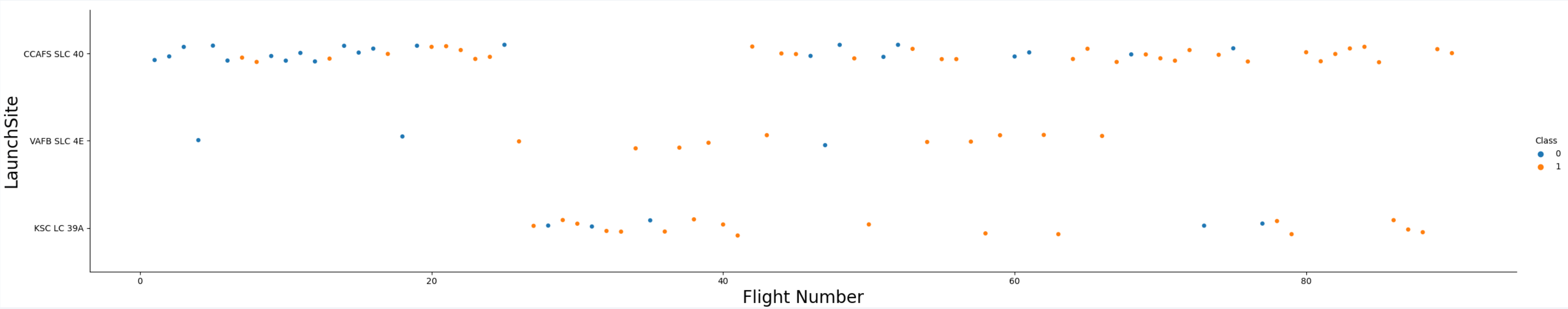
Predictive analysis results

- Logistic regression, SVM, Decision Tree and KNeigbhour Classifier have the same accuracy for test data.
- Decision Tree has the best accuracy for train data.

Section 2

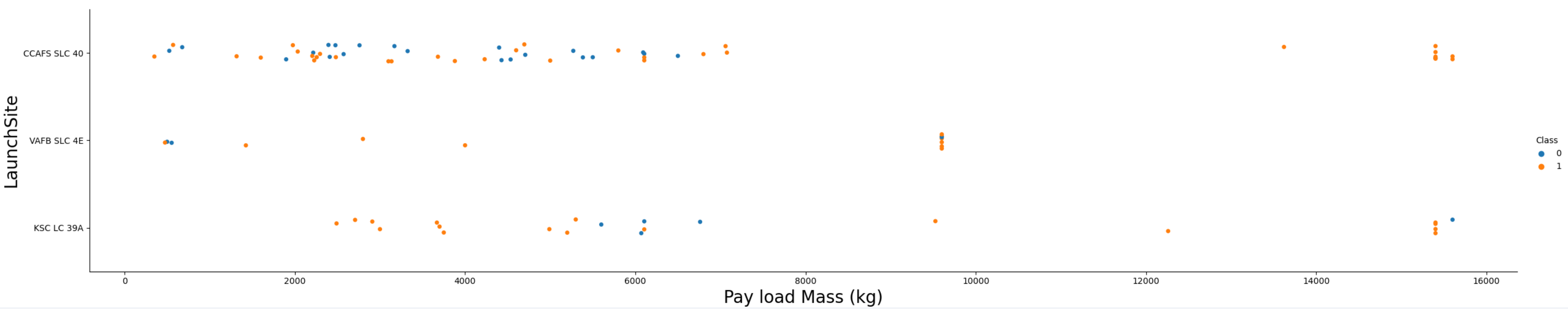
Insights drawn from EDA

Flight Number vs. Launch Site



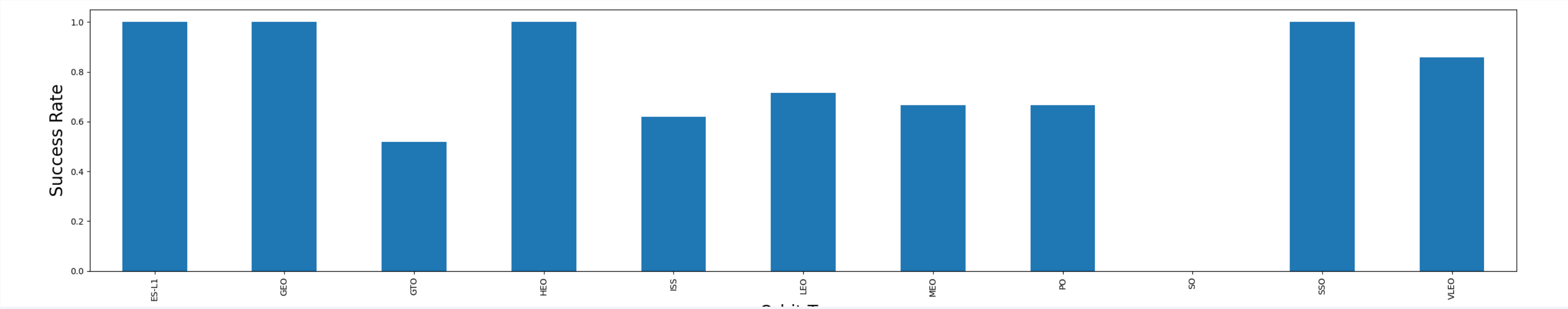
- Based on the scatter plot, **the higher the flight number, the higher success rate for landing**.
- Both KSC LC – 39A and VAFB SLC 4E have the best success rate for launching.

Payload vs. Launch Site



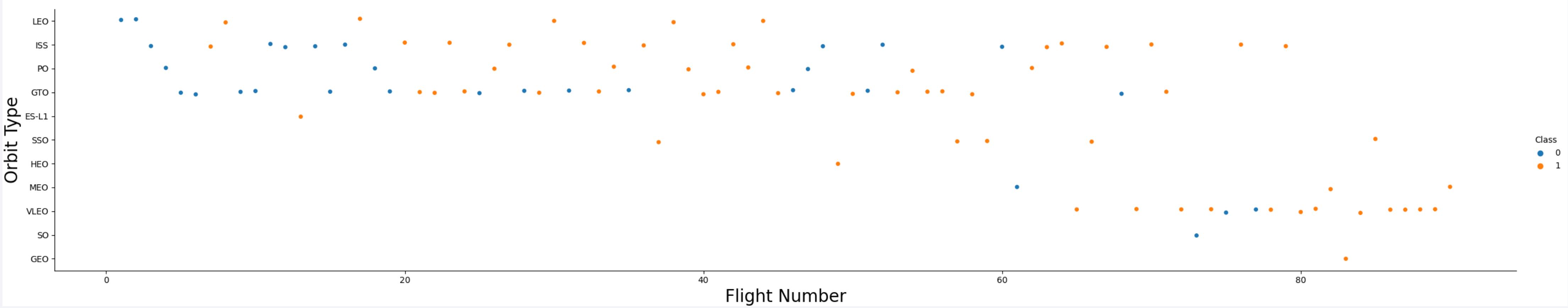
- KSC LC – 39A has a high success rate for landing if the pay load mass is less than 6,000 kg.
- There is no rocket launching for pay load mass greater than 10,000 kg for VAFB-SLC.

Success Rate vs. Orbit Type



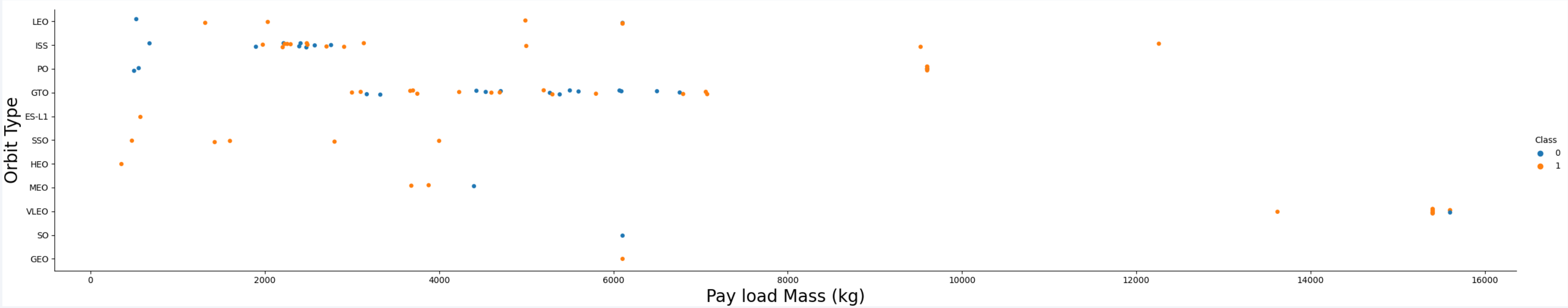
The orbit type which has **high success rate** are **ES-L1, GEO, HEO, and SSO**.

Flight Number vs. Orbit Type



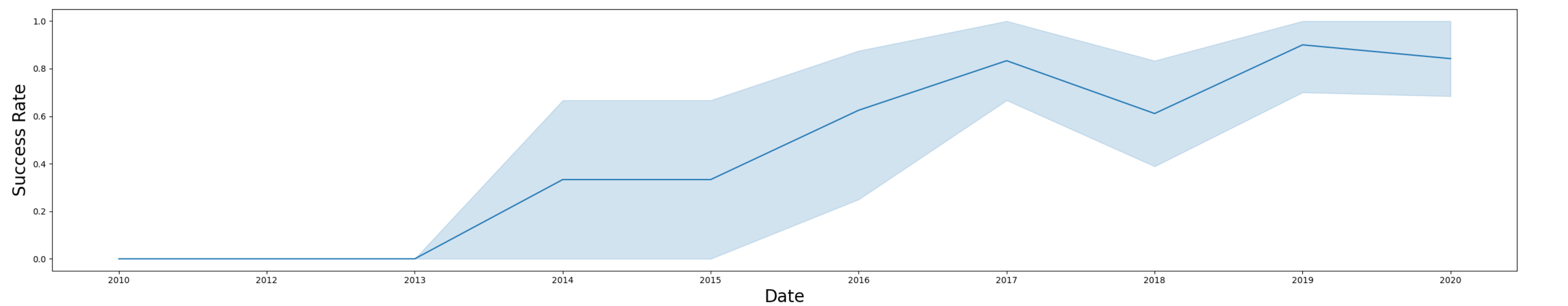
- **LEO orbit** exhibits a relationship with the flight number. **The higher the flight number, the higher the success rate**.
- There is **no correlation** observed for **GEO orbit with the flight number**.

Payload vs. Orbit Type



- For **LEO, ISS and PO orbits**, it is observed that **the higher pay load mass results in higher success rate**.

Launch Success Yearly Trend



- There is an **increasing trend for success rate** starting from year 2013 until year 2020 based on the line graph shown above.

All Launch Site Names

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

```
In [7]: %sql SELECT DISTINCT Launch_site FROM SPACEXTBL
```

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

Out[7]:

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

%sql SELECT DISTINCT Launch_site FROM SPACEXTBL

- The keyword used to display the unique launch sites from the SpaceX data is **DISTINCT**.

Launch Site Names Begin with 'CCA'

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

In [8]:
```
%sql SELECT * FROM SPACEXTBL WHERE Launch_site LIKE 'CCA%' LIMIT 5
```

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

Out[8]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outc
06/04/2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0.0	LEO	SpaceX	Success	Failure (paracl
12/08/2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0.0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (paracl
22/05/2012	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525.0	LEO (ISS)	NASA (COTS)	Success	No att
10/08/2012	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500.0	LEO (ISS)	NASA (CRS)	Success	No att
03/01/2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677.0	LEO (ISS)	NASA (CRS)	Success	No att

%sql SELECT * FROM SPACEXTBL WHERE Launch_site LIKE 'CCA%' LIMIT 5

- The keyword used to display the launch sites begin with ‘CCA’ is **LIKE**.
- The keyword used to show 5 records is **LIMIT**.

Total Payload Mass

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

```
In [14]: %sql SELECT sum(payload_mass__kg_) as TOTAL_PAYLOAD_MASS FROM SPACEXTBL WHERE customer = 'NASA (CRS)'

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

Out[14]:

TOTAL_PAYLOAD_MASS
45596.0

%sql SELECT sum(payload_mass__kg_) as TOTAL_PAYLOAD_MASS FROM SPACEXTBL WHERE customer = 'NASA (CRS)'

Average Payload Mass by F9 v1.1

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

```
In [15]: %sql SELECT avg(payload_mass__kg_) AS AVERAGE_PAYLOAD_MASS FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'

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

Out[15]:

AVERAGE_PAYLOAD_MASS
2928.4

%sql SELECT avg(payload_mass__kg_) AS AVERAGE_PAYLOAD_MASS FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'

First Successful Ground Landing Date

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

Hint:Use min function

In [13]:
```
%sql SELECT min(DATE) FROM SPACEXTBL WHERE landing_outcome = 'Success (ground pad)'
```

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

Out[13]:

min(DATE)
01/08/2018

%sql SELECT min(DATE) FROM SPACEXTBL WHERE landing_outcome = 'Success (ground pad)'

- The first successful landing outcome in ground pad was achieved on 1 August 2018.

Successful Drone Ship Landing with Payload between 4000 and 6000

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

In [17]:
```
%sql SELECT booster_version FROM SPACEXTBL WHERE landing_outcome = 'Success (drone ship)'\
and payload_mass__kg_ between 4000 and 6000
```

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

Out[17]:

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

%sql SELECT booster_version FROM SPACEXTBL WHERE landing_outcome = 'Success (drone ship)'\ **and** payload_mass__kg_ between 4000 **and** 6000

- The keyword used to filter for boosters that have successfully landed on drone ship is **WHERE**.
- The keyword used to determine successful landing with payload mass greater than 4,000 but less than 6,000 after the filter with WHERE is **AND**.

Total Number of Successful and Failure Mission Outcomes

List the total number of successful and failure mission outcomes

In [18]:
```
%sql SELECT mission_outcome, COUNT(mission_outcome) FROM SPACEXTBL GROUP BY mission_outcome
```

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

Out[18]:

Mission_Outcome	COUNT(mission_outcome)
None	0
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

%sql SELECT mission_outcome, COUNT(mission_outcome) FROM SPACEXTBL GROUP BY mission_outcome

- **GROUP BY** is used to determine the type of mission outcome.
- The total number of **successful mission outcome is 100**.
- The total number of **failure mission outcome is 0**.

Boosters Carried Maximum Payload

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

In [19]:
```
%sql SELECT booster_version, payload_mass__kg_ FROM SPACEXTBL\
WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)
```

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

Out[19]:

Booster_Version	PAYLOAD_MASS__KG_
F9 B5 B1048.4	15600.0
F9 B5 B1049.4	15600.0
F9 B5 B1051.3	15600.0
F9 B5 B1056.4	15600.0
F9 B5 B1048.5	15600.0
F9 B5 B1051.4	15600.0
F9 B5 B1049.5	15600.0
F9 B5 B1060.2	15600.0
F9 B5 B1058.3	15600.0
F9 B5 B1051.6	15600.0
F9 B5 B1060.3	15600.0
F9 B5 B1049.7	15600.0

%sql SELECT booster_version, payload_mass__kg_ FROM SPACEXTBL\ WHERE payload_mass__kg_ = (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)

- **MAX()** function is used in subquery of WHERE clause to determine the booster version that carried maximum payload mass.

2015 Launch Records

List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.

Note: SQLLite does not support monthnames. So you need to use substr(Date, 4, 2) as month to get the months and substr(Date,7,4)='2015' for year.

In [29]:

```
%sql SELECT substr(Date,4,2) as month, DATE, booster_version, launch_site, [Landing_Outcome] FROM SPACEXTBL\
WHERE [Landing_Outcome] = 'Failure (drone ship)'\
AND substr(Date,7,4)='2015'
```

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

Out[29]:

month	Date	Booster_Version	Launch_Site	Landing_Outcome
10	01/10/2015	F9 v1.1 B1012	CCAFS LC-40	Failure (drone ship)
04	14/04/2015	F9 v1.1 B1015	CCAFS LC-40	Failure (drone ship)

%sql SELECT substr(Date,4,2) as month, DATE, booster_version, launch_site, [Landing_Outcome] FROM SPACEXTBL\ WHERE [Landing_Outcome] = 'Failure (drone ship)'\ AND substr(Date,7,4)='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.

In [30]:

```
%sql SELECT [Landing_Outcome], COUNT(*) AS count_outcomes FROM SPACEXTBL\
WHERE DATE between '04-06-2010' and '20-03-2017' GROUP BY [Landing_Outcome] ORDER BY count_outcomes DESC
```

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

Out[30]:

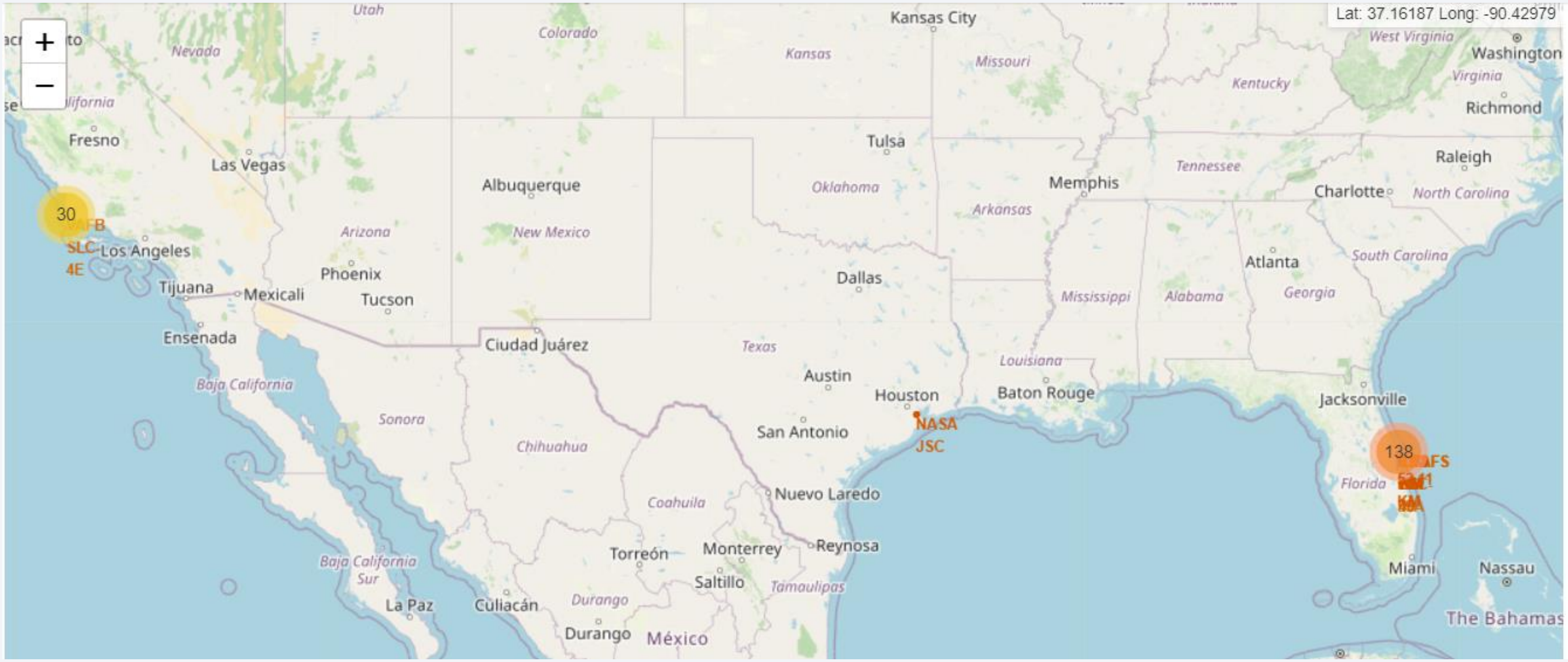
Landing_Outcome	count_outcomes
Success	20
No attempt	10
Success (drone ship)	8
Success (ground pad)	7
Failure (drone ship)	3
Failure	3
Failure (parachute)	2
Controlled (ocean)	2
No attempt	1

%sql SELECT [Landing_Outcome], COUNT(*) AS count_outcomes FROM SPACEXTBL\ WHERE DATE between '04-06-2010' **and** '20-03-2017' GROUP BY [Landing_Outcome] ORDER BY count_outcomes DESC

Section 3

Launch Sites Proximities Analysis

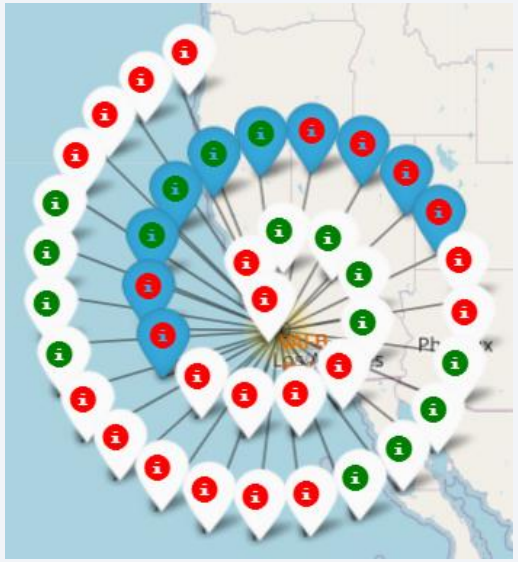
All Launch Sites Location on a Global Map



- SpaceX launch sites in the United States shown with circle marker are in:
  - Florida
  - California
- NASA Johnson Space Center is marked without circle marker.

Color-labeled Launch Outcomes

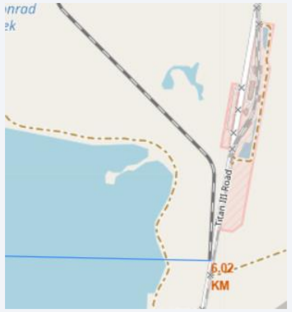
Florida Launch Sites



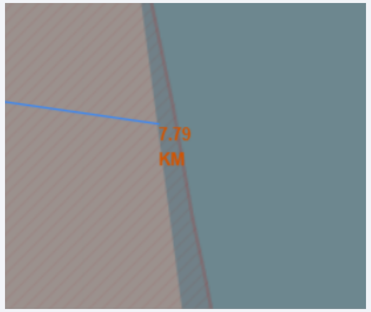
California Launch Site

- Green marker indicates successful launches.
- Red marker refers to failure launches.

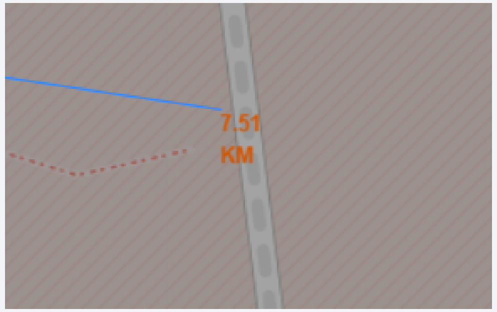
Launch Sites Distance to Landmarks



Distance to railway



Distance to coastline



Distance to highway

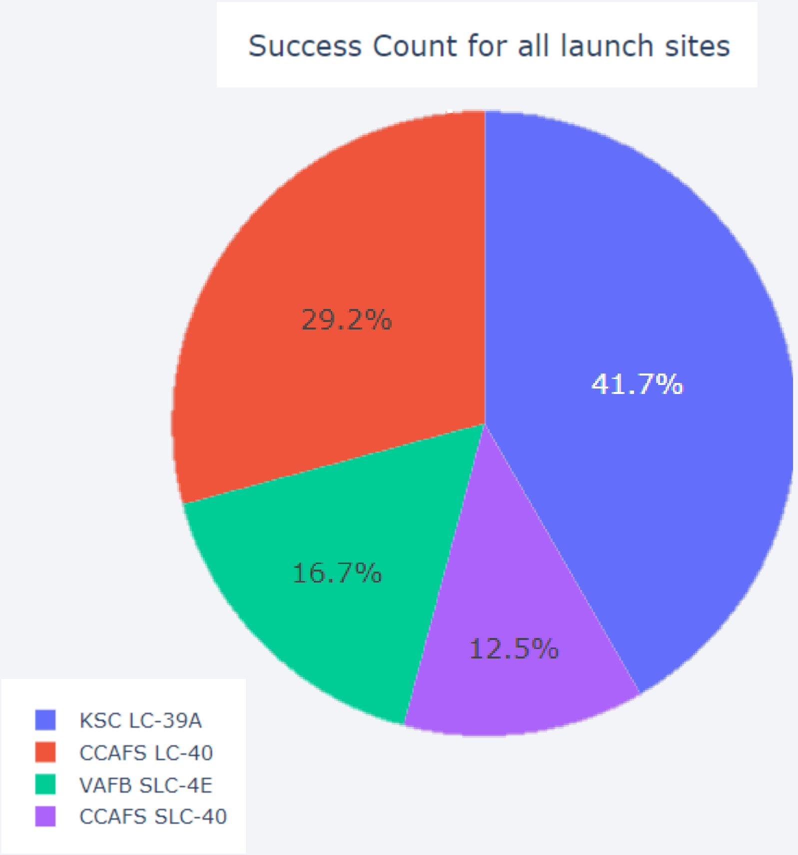
Distance to cities

- 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.
- Are launch sites in close proximity to cities? Yes.

Section 4

Build a Dashboard with Plotly Dash

SpaceX Success Launches



KSC LC-39A

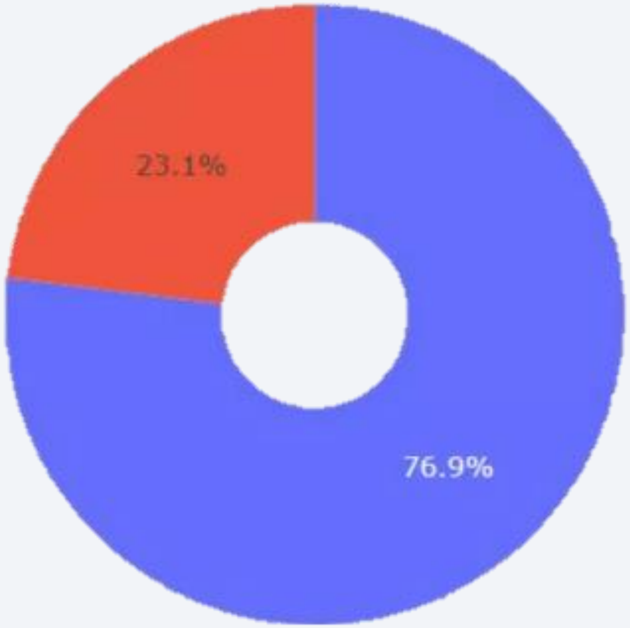
has among all sites the **most prosperous** launches.

KSC LC-39A Launch Ratio

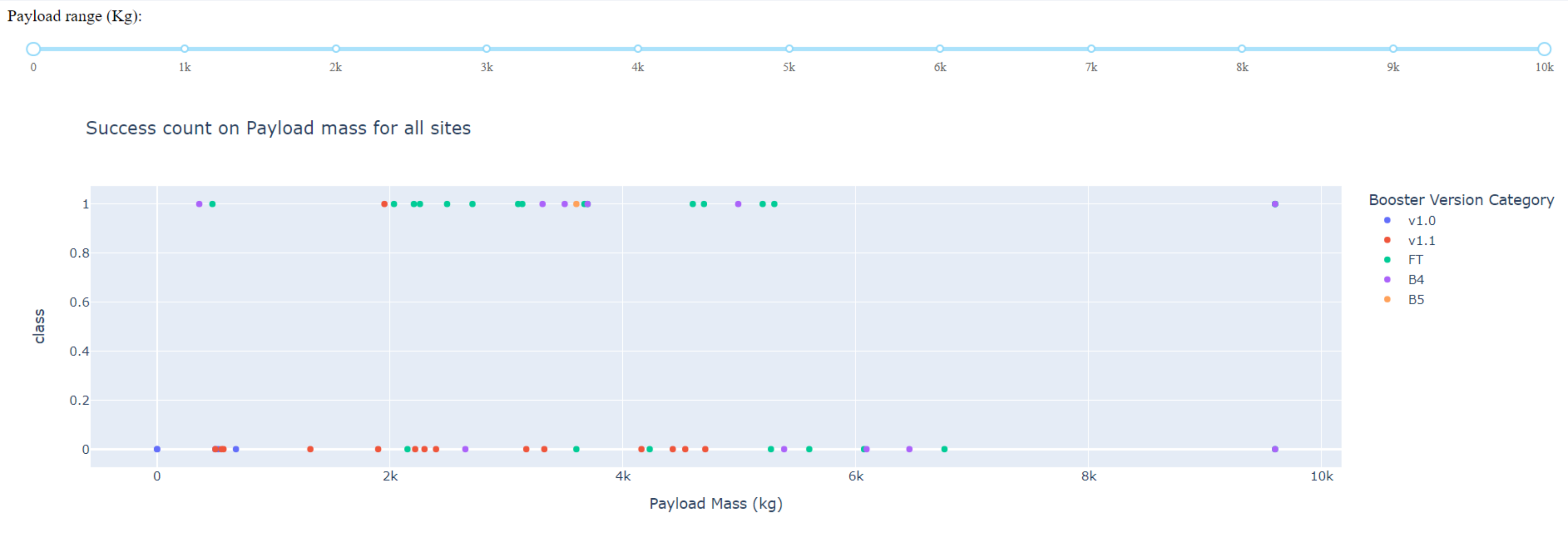
Most prosperous launches.

76.9% **success rate.**

23.1% **failure rate.**



Payload vs Launch Outcome



0 kg – 4,000 kg

success rate > failure rate

4,000 kg – 8,000 kg

success rate < failure rate

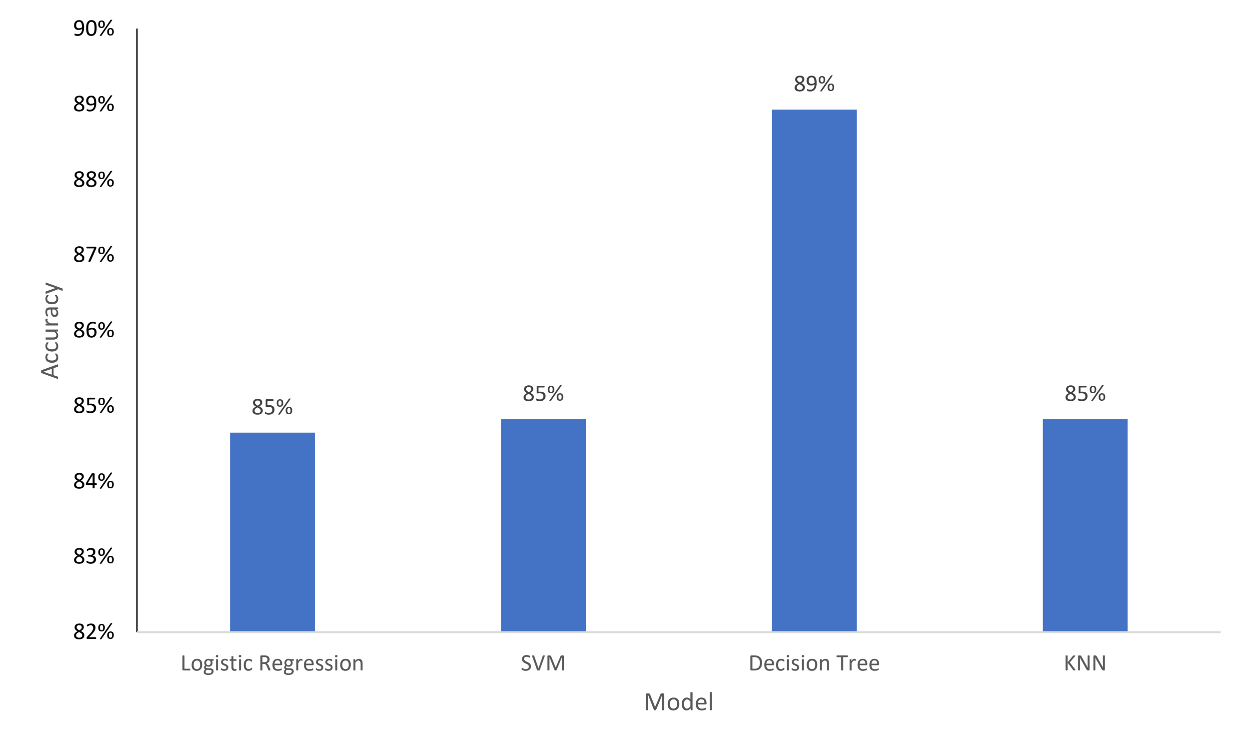
FT Booster Version

has the **most success** rate.

Section 5

Predictive Analysis (Classification)

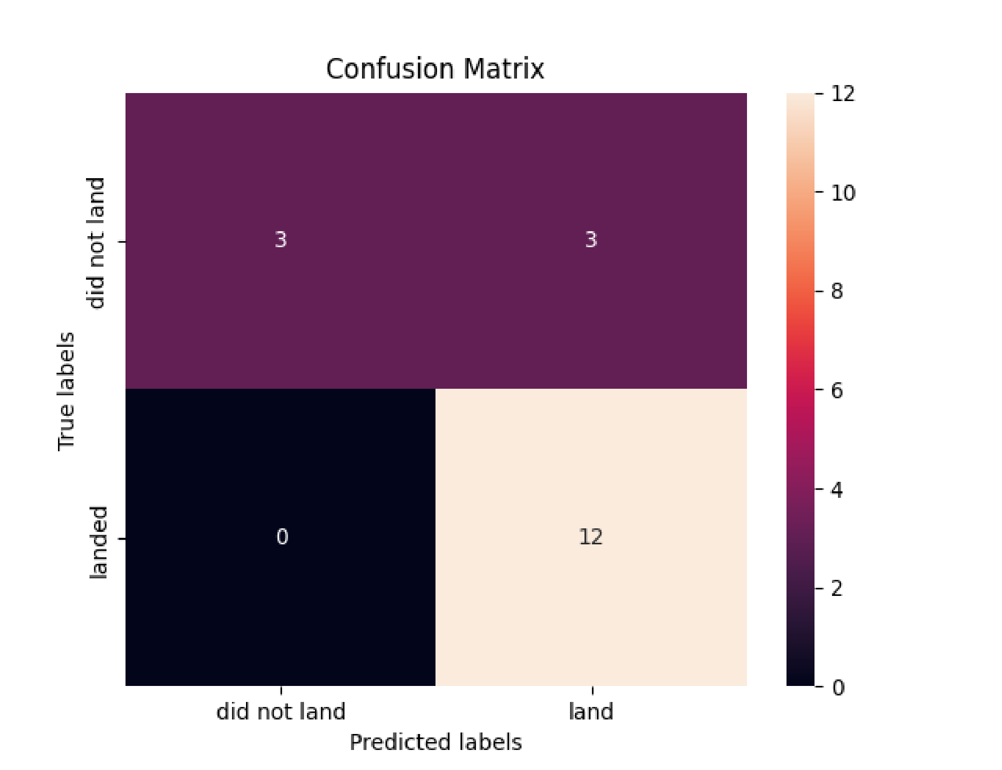
Classification Accuracy



Decision Tree

has the **highest** classification accuracy

Confusion Matrix



Decision Tree

- Confusion matrix shown for decision tree on the left figure display the capability of the classifier to differentiate between different classes.
- One notable problem observed in the matrix is the **false positives** where actual unsuccessful landing is determined as successful landing by the classifier.

Conclusions

- Low payload mass has a higher success rate than the high payload mass.
- Both KSC LC – 39A and VAFB SLC 4E have the best success rate for launching.
- Increasing trend of success rate from 2013 until 2020.
- The orbit type which has high success rate are ES-L1, GEO, HEO, and SSO.
- Logistic regression, SVM, Decision Tree and KNeigbhour Classifier have the same accuracy for test data.
- Decision Tree has the best accuracy for train data.

Thank you!