Winning the Space Race With Data Science

Heath Barnett

October 16, 2023

Outline

Executive Summary

Summary of Methodologies

- Data Collection with API and Web Scraping
- Data Wrangling and Formatting
- Exploratory Data Analysis
  - SQL
  - Visualizations
    - Jupyter Notebooks
    - Folium Maps
    - Dashboard with Plotly Dash
- Classification with Machine Learning

Summary of Results

- Exploratory Data Analysis
  - Query Results
  - Graphs
- Interactive Folium Map
- Interactive Dashboard
- Machine Learning Predictions for several models
  - Decision Tree
  - k-Nearest Neighbors
  - Logistic Regression
  - Support Vector

Introduction

Predicting whether the Falcon 9 first stage will land successfully is a complex task that involves various factors, including:

- Payload
- Orbits
- Booster Types
- Geographical Information of Launch Sites
- Success and Failure Rates

While SpaceX has made significant advancements in rocket reusability and has successfully landed many Falcon 9 first stages, there are still risks involved in each launch.

To make an accurate prediction of the success rate for future versions of Falcon 9, we will need to employ a variety of data science methodologies and machine learning techniques.

Section 1

Methodology

Data Collection – SpaceX API

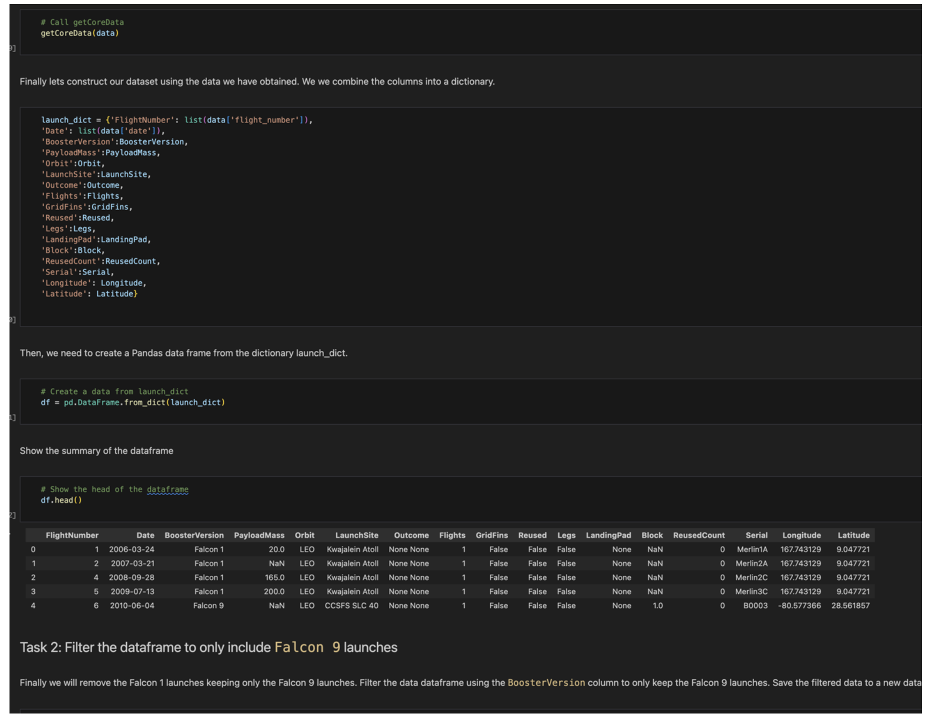
Python package request to get content.

Response content imported as a JSON file.

JSON data converted to Pandas dataframe.

Cleaned data for missing values, replacing with averages.

Filtered data in various ways to ensure integrity.



```
# Call getCoreData
getCoreData(data)
```

Finally lets construct our dataset using the data we have obtained. We we combine the columns into a dictionary.

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

Then, we need to create a Pandas data frame from the dictionary launch_dict.

```
# Create a data from launch_dict
df = pd.DataFrame.from_dict(launch_dict)
```

Show the summary of the dataframe

```
# Show the head of the dataframe
df.head()
```

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin1A	167.743129	9.047721
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2A	167.743129	9.047721
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin2C	167.743129	9.047721
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN	0	Merlin3C	167.743129	9.047721
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857

Task 2: Filter the dataframe to only include `Falcon 9` launches

Finally we will remove the Falcon 1 launches keeping only the Falcon 9 launches. Filter the data dataframe using the `BoosterVersion` column to only keep the Falcon 9 launches. Save the filtered data to a new data

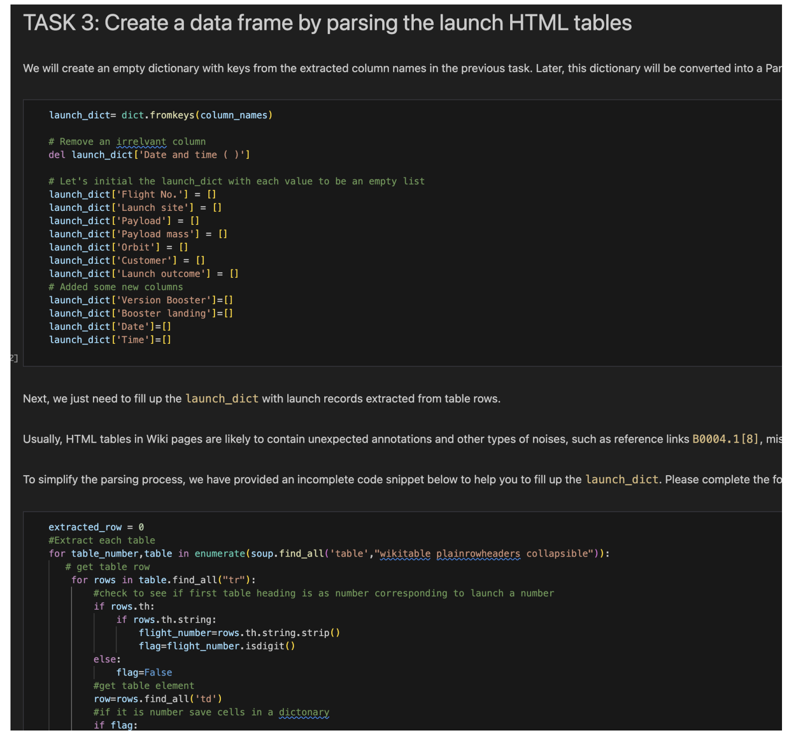
https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-spacex-data-collection-api.ipynb

Data Collection - Scraping

Web Scraping with Beautiful Soup

Wikipedia data for Falcon 9 launch records.

Results converted to pandas dataframe.



TASK 3: Create a data frame by parsing the launch HTML tables

We will create an empty dictionary with keys from the extracted column names in the previous task. Later, this dictionary will be converted into a Par

```
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']=[]
```

Next, we just need to fill up the `launch_dict` with launch records extracted from table rows.

Usually, HTML tables in Wiki pages are likely to contain unexpected annotations and other types of noises, such as reference links `B0004.1[8]`, mis

To simplify the parsing process, we have provided an incomplete code snippet below to help you to fill up the `launch_dict`. Please complete the fo

```
extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
   # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
        else:
            flag=False
        #get table element
        row=rows.find_all('td')
        #if it is number save cells in a dictonary
        if flag:
```

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-webscraping.ipynb

Data Wrangling

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
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	0
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	0
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	0
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	0
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	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
85	86	2020-09-03	Falcon 9	15400.000000	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	2	B1060	-80.603956	28.608058	1
86	87	2020-10-06	Falcon 9	15400.000000	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	2	B1058	-80.603956	28.608058	1
87	88	2020-10-18	Falcon 9	15400.000000	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	5	B1051	-80.603956	28.608058	1
88	89	2020-10-24	Falcon 9	15400.000000	VLEO	CCAFS SLC 40	True ASDS	3	True	True	True	5e9e3033383ecbb9e534e7cc	5.0	2	B1060	-80.577366	28.561857	1
89	90	2020-11-05	Falcon 9	3681.000000	MEO	CCAFS SLC 40	True ASDS	1	True	False	True	5e9e3032383ecb6bb234e7ca	5.0	0	B1062	-80.577366	28.561857	1

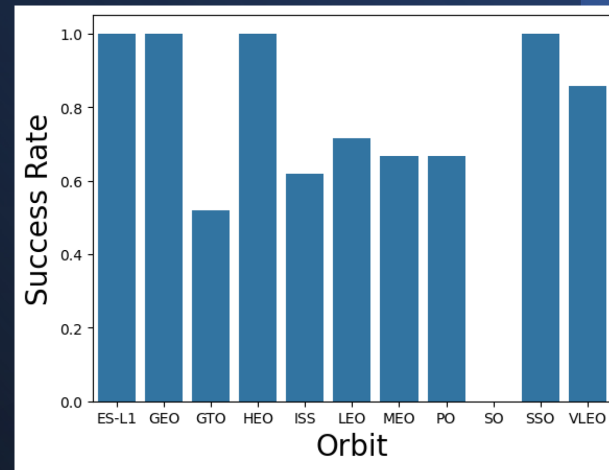
Filtered data for only Falcon 9 booster version.
Calculated a variety of parameters including launches at each site and success rates by several categories.
Results exported to csv data files for later use.

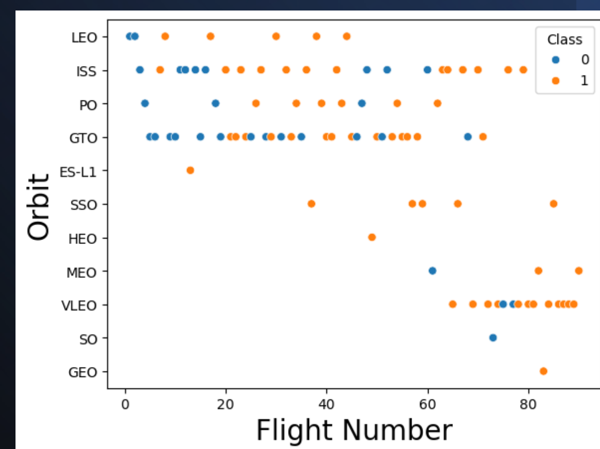
https://github.com/heath-barnett/ibm_ds_cert/blob/main/labs-jupyter-spacex-Data%20wrangling.ipynb

Exploratory Data Analysis with Data Visualization

Figure	Plot Type	Data
Figure 1	Categorical	Flight Number vs Launch Site
Figure 2	Categorical	Payload vs Launch Site
Figure 3	Bar Chart	Success Rate vs Orbit
Figure 4	Scatter	Flight Number vs Orbit
Figure 5	Categorical	Payload vs Orbit
Figure 6	Line	Annual Success Rate

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb





Exploratory Data Analysis with SQL

SQL Commands/Queries Used

- SELECT
- FROM
- DISTINCT
- WHERE
- BETWEEN
- LIKE
- LIMIT
- MIN
- MAX
- SUM
- COUNT

```
%%sql
SELECT Booster_Version as 'Booster', PAYLOAD_MASS__KG_ AS 'Payload Mass (kg)', Landing_Outcome AS 'Outcome'
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)'
AND PAYLOAD_MASS__KG_ BETWEEN '4000' AND '6000'
```

```
%%sql
SELECT *
FROM SPACEXTABLE
WHERE Launch_Site
LIKE 'CCA%'
LIMIT 5
```

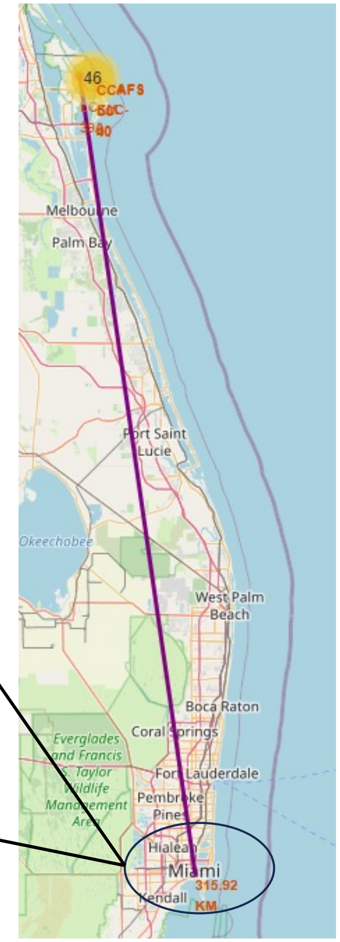
```
%%sql
SELECT Customer, SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTABLE
WHERE Customer = 'NASA (CRS)'
```

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Marked all launch sites
- Assigned all sites with success or failure makers.
- Calculated distances between several objects and displayed the results with line features.

Folium

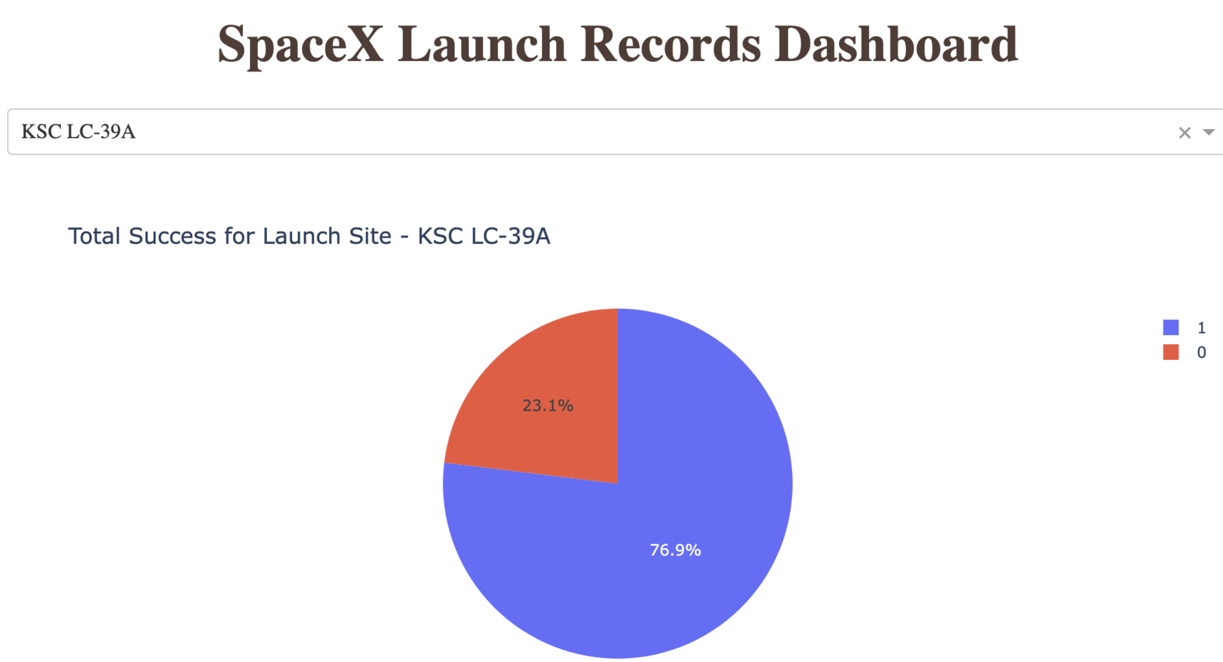


Distance of 315.92 km.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Build a Dashboard with Plotly Dash

- Pie Charts for Each Launch that show success and failure rates.
- Scatter plots showing the relationship between success/failure and payload with an adjustable slider for payload selection.



Predictive Analysis (Classification)

DATA IMPORTED WITH PANDAS

TRANSFORMED WITH SCI-KIT LEARN

SPLIT DATA INTO TEST/TRAINING SETS

MACHINE LEARNING MODELS PARAMETERS WERE OPTIMIZED WITH GRID-CV SEARCH

MODELS INVESTIGATED:

DECISION TREE, KNN, SVM, LOG REGRESSION

ACCURACY ON TEST SETS WERE CALCULATED AND CONFUSION MATRICES WERE PRODUCED

https://github.com/heath-barnett/ibm_ds_cert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

Exploratory data analysis results

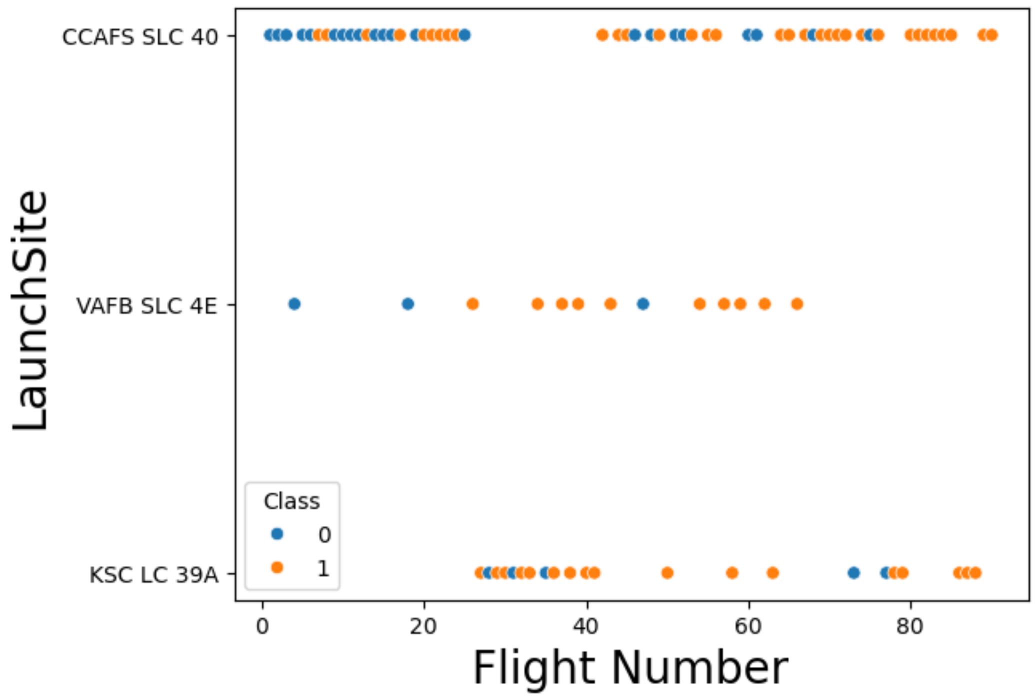
Interactive analytics demo in screenshots

Predictive analysis results

Section 2

Insights from Exploratory Data Analysis

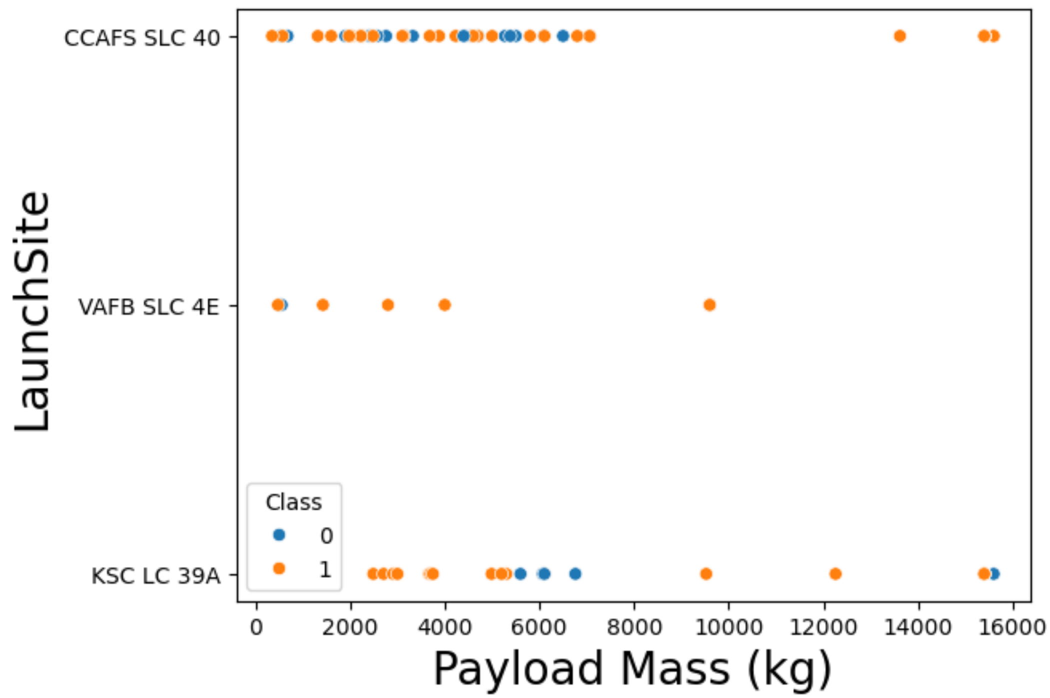
Flight Number vs. Launch Site



- Data indicates that as flight numbers increase, the chance of success also increases.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

Payload vs. Launch Site

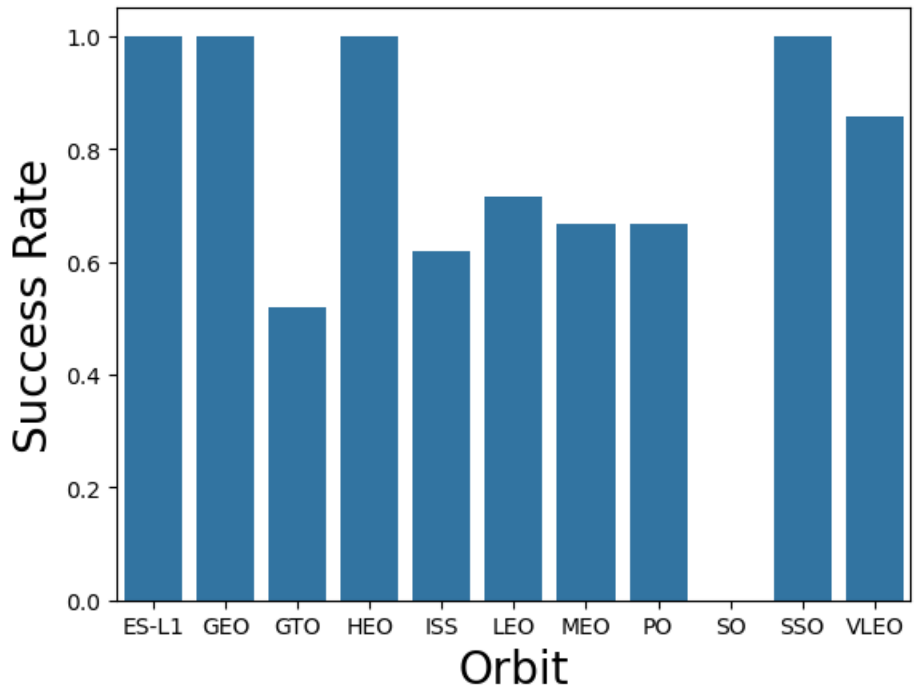


There is no clear relationship between payload mass and the launch site for CCAFS SLS 40 and KSC LC 39A.

Launch site VAFB SLC 4E does show consistent success rates regardless of payload.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

Success Rate vs. Orbit Type



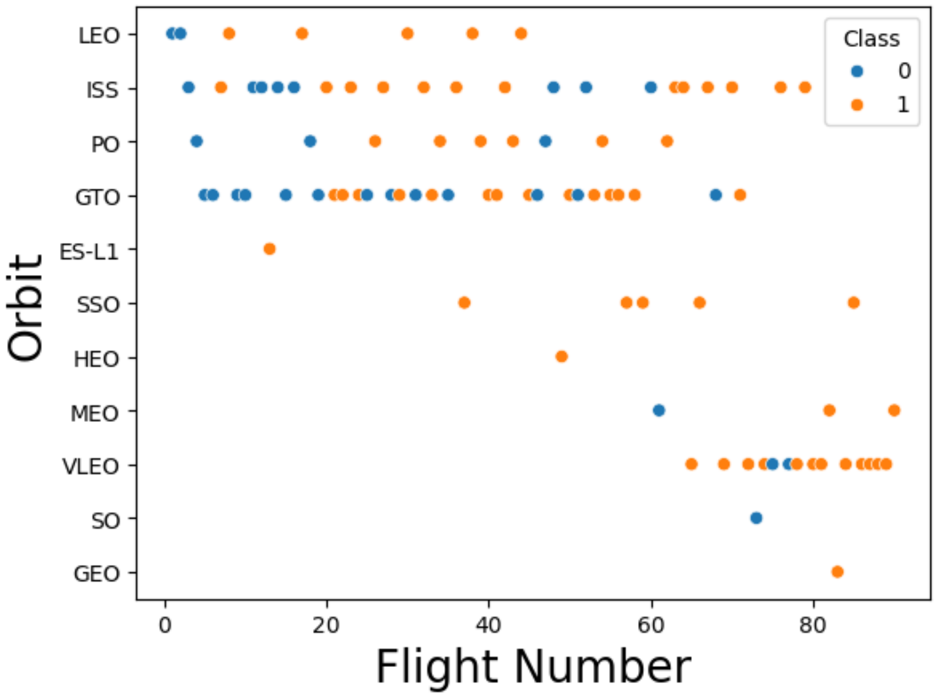
- Several orbit types have high success rates at or near 100%.
  - ES-L1, GEO, HEO, SSO and VLEO
- SO orbit always fails.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

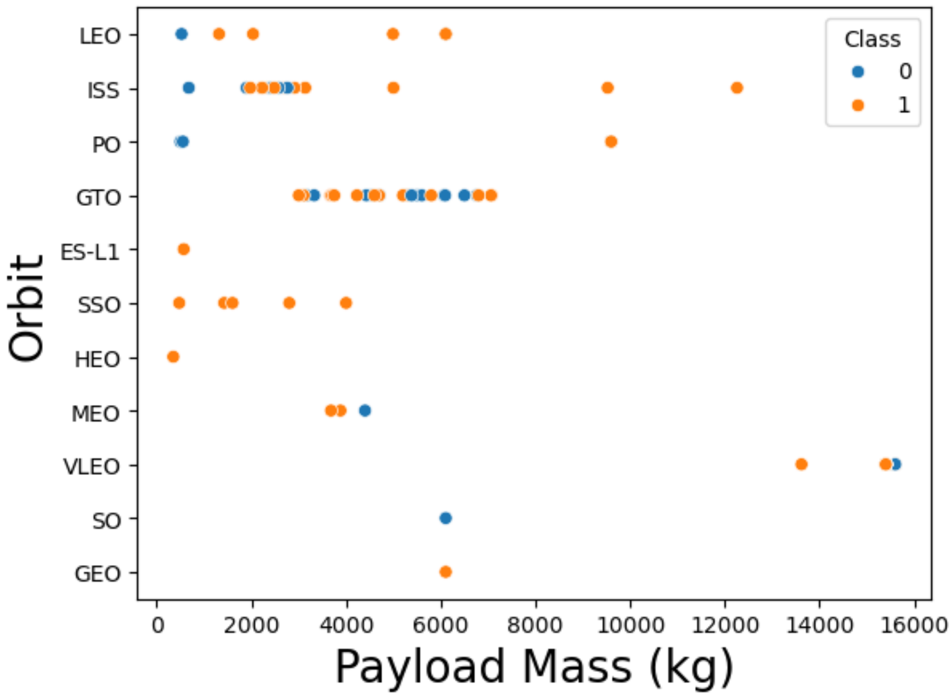
Flight Number vs. Orbit Type

- Early flight numbers are more likely to fail regardless of orbit. Consistent with previous findings regarding flight numbers and launch sites.
- Some orbit types are not attempted until much later in development cycle.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb



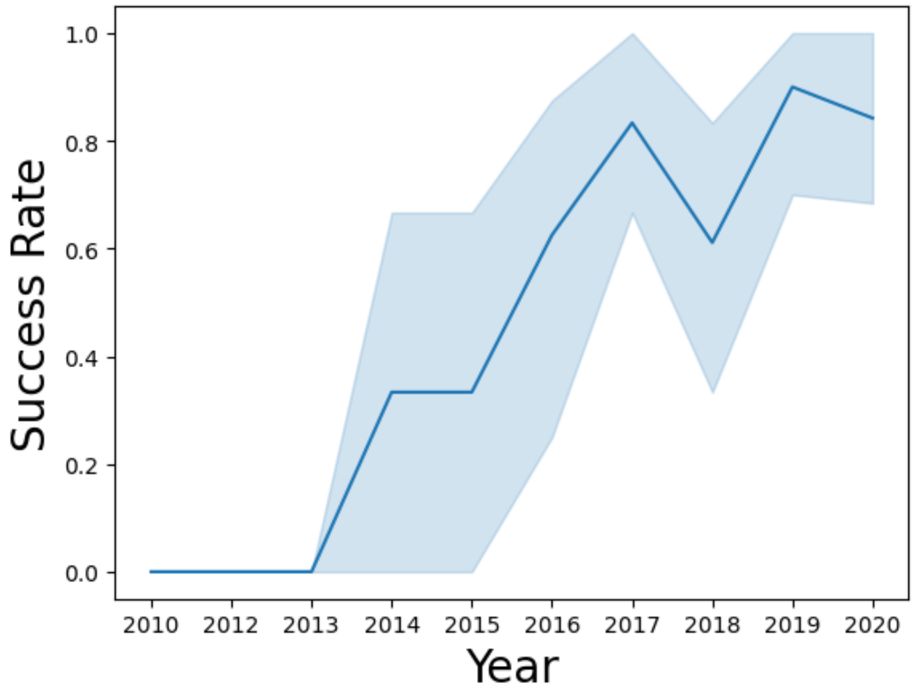
Payload vs. Orbit Type



- With heavy payloads the successful landing or positive landing rate are more for Polar,LEO and ISS.
- GTO is inconclusive, positive landing rate and negative landing(unsuccessful mission) are both relatively equal in the payload range of 3500 to 7000.

  https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

Launch Success Yearly Trend



- Launch success on a steady increase from 2013 and 2017.
- 2018 show a downturn but the following year success rates rebounded.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-dataviz.ipynb.jupyterlite.ipynb

All Launch Site Names

```
%%sql
SELECT DISTINCT Launch_Site
FROM SPACEXTABLE
```

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

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Launch Site Names Begin with 'CCA'

```
%%sql
SELECT *
FROM SPACEXTABLE
WHERE Launch_Site
LIKE 'CCA%'
LIMIT 5
```

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

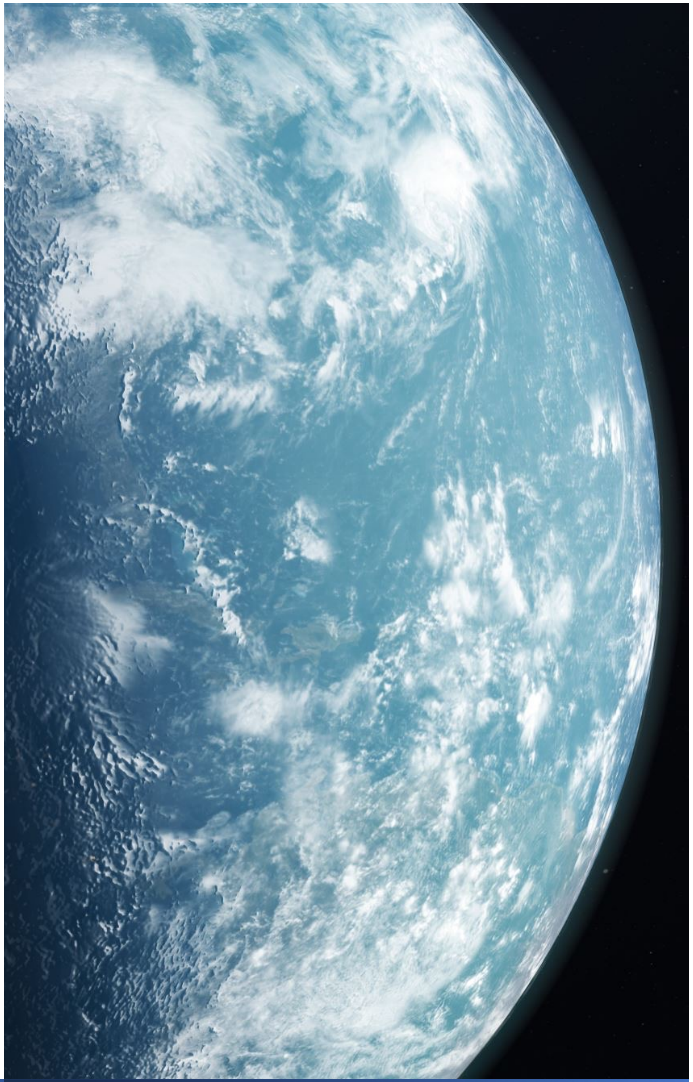
Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight 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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

```
%%sql
SELECT Customer, SUM(PAYLOAD_MASS__KG_)
FROM SPACEXTABLE
WHERE Customer = 'NASA (CRS)'
```

Customer	SUM(PAYLOAD_MASS__KG_)
NASA (CRS)	45596

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb



Average Payload Mass by F9 v1.1

```
%%sql
SELECT Booster_Version, AVG(PAYLOAD_MASS__KG_) AS 'Average Payload'
FROM SPACEXTABLE
WHERE Booster_Version = 'F9 v1.1'
```

Booster_Version	Average Payload
F9 v1.1	2928.4

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

First Successful Ground Landing Date

```
%%sql
SELECT min(Date) as 'Date of First Ground Pad Success'
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (ground pad)'
```

Date of First Ground Pad Success
2015-12-22

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql
SELECT Booster_Version as 'Booster', PAYLOAD_MASS__KG_ AS 'Payload Mass (kg)', Landing_Outcome AS 'Outcome'
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Success (drone ship)'
AND PAYLOAD_MASS__KG_ BETWEEN '4000' AND '6000'
```

Booster	Payload Mass (kg)	Outcome
F9 FT B1022	4696	Success (drone ship)
F9 FT B1026	4600	Success (drone ship)
F9 FT B1021.2	5300	Success (drone ship)
F9 FT B1031.2	5200	Success (drone ship)

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Total Number of Successful and Failure Mission Outcomes

```
%%sql
SELECT Mission_Outcome, count(*) as 'Count'
FROM SPACEXTABLE
GROUP BY Mission_Outcome
```

Mission_Outcome	**Count**
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Boosters Carried Maximum Payload

```
%%sql
SELECT Booster_Version as 'Booster', PAYLOAD_MASS__KG_
AS 'Max Payload'
FROM SPACEXTABLE
WHERE PAYLOAD_MASS__KG_ = (SELECT
MAX(PAYLOAD_MASS__KG_) FROM SPACEXTABLE)
```

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Booster	Max Payload
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

2015 Launch Records

```
%%sql
SELECT
CASE substr(Date,6,2)
WHEN '01' THEN 'January'
WHEN '02' THEN 'Febuary'
WHEN '03' THEN 'March'
WHEN '04' THEN 'April'
WHEN '05' THEN 'May'
WHEN '06' THEN 'June'
WHEN '07' THEN 'July'
WHEN '08' THEN 'August'
WHEN '09' THEN 'September'
WHEN '10' THEN 'October'
WHEN '11' THEN 'November'
WHEN '12' THEN 'December'
ELSE ''
END AS Month, Landing_Outcome, Booster_Version, Launch_Site
FROM SPACEXTABLE
WHERE Landing_Outcome = 'Failure (drone ship)'
AND SUBSTR(Date,0,5)='2015'
```

Month	Landing_Outcome	Booster_Version	Launch_Site
October	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

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

```
%%sql
SELECT Landing_Outcome, count(Landing_Outcome) AS 'Total', Date
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY Landing_Outcome
ORDER BY count(Landing_Outcome) DESC
```

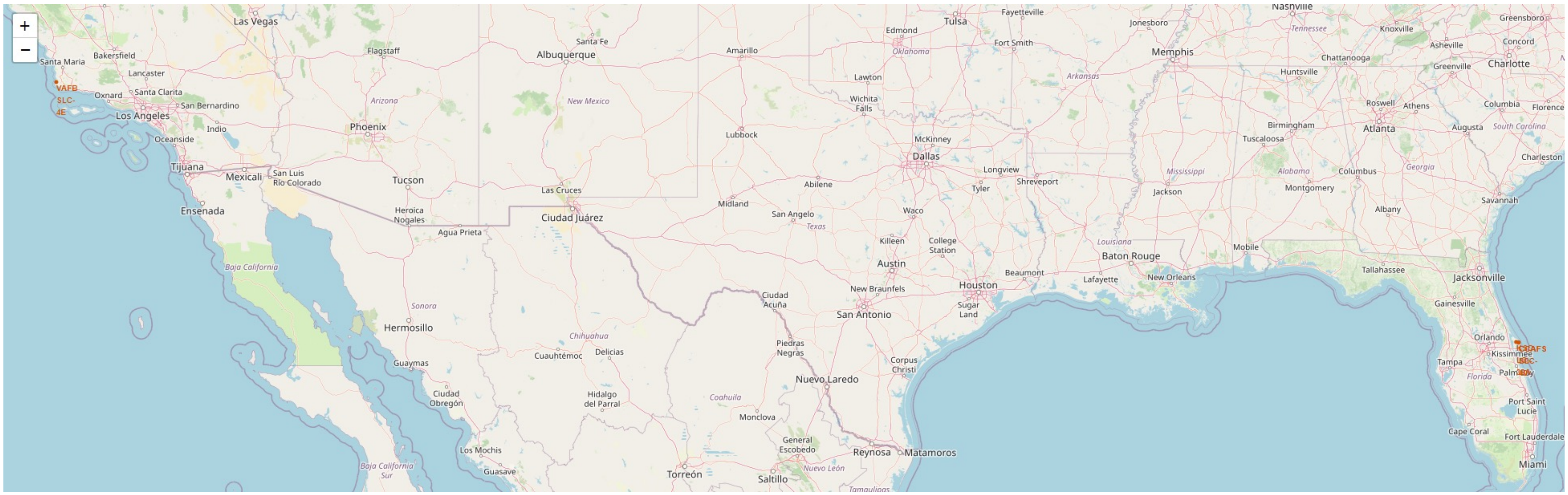
Landing_Outcome	Total	Date
No attempt	10	2012-05-22
Success (ground pad)	5	2015-12-22
Success (drone ship)	5	2016-08-04
Failure (drone ship)	5	2015-10-01
Controlled (ocean)	3	2014-04-18
Uncontrolled (ocean)	2	2013-09-29
Precluded (drone ship)	1	2015-06-28
Failure (parachute)	1	2010-08-12

https://github.com/heath-barnett/ibm_ds_cert/blob/main/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Section 3

Launch Site Proximities Analysis

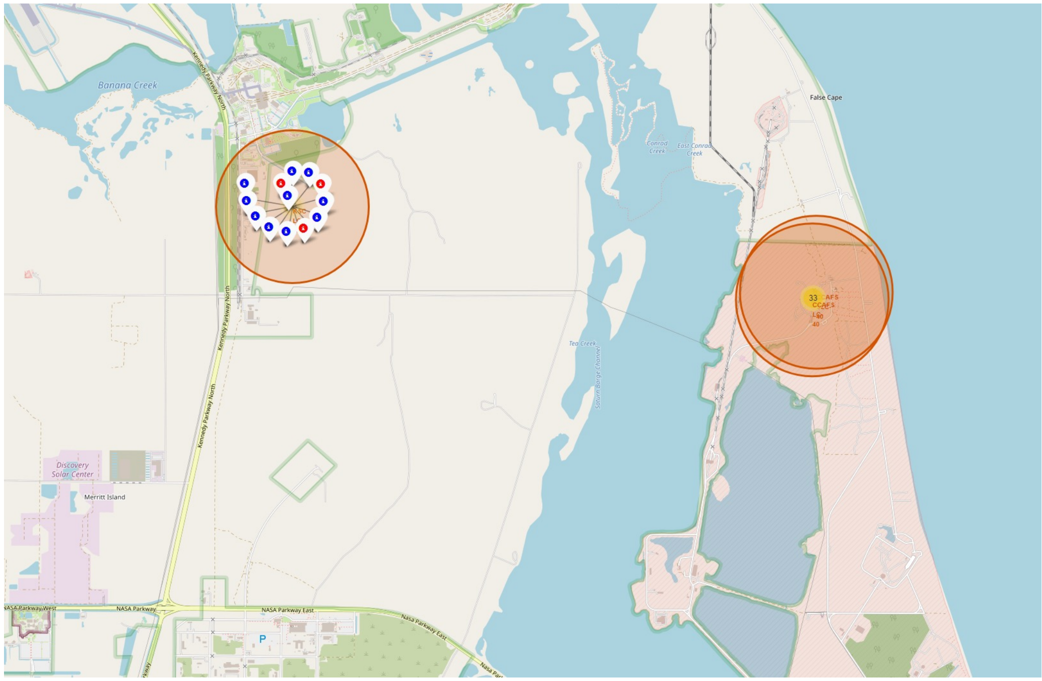
All Launch Sites



- Located on coast areas, both southwest and southeast United States.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

Launch Site Outcomes at KSC LC-39A



- Blue indicates a successful launch, red indicates a failure.
- Located in Florida, southeast coast.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

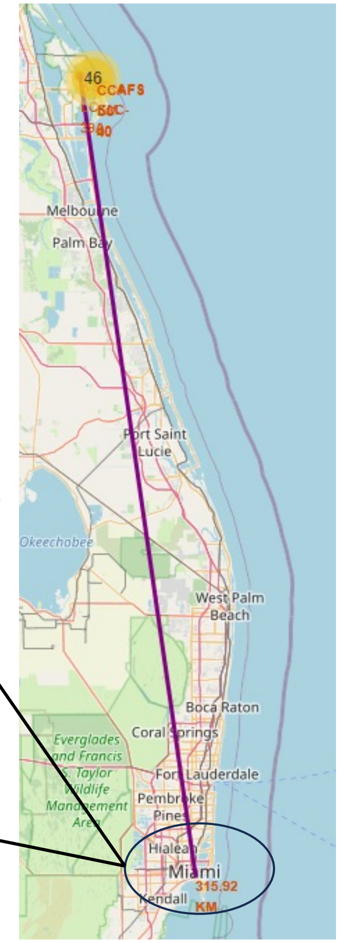
Launch Site KSC LC-39A Distance to Miami Florida

```
def calculate_distance(lat1, lon1, lat2, lon2):
    # approximate radius of earth in km
    R = 6373.0

    lat1 = radians(lat1)
    lon1 = radians(lon1)
    lat2 = radians(lat2)
    lon2 = radians(lon2)

    dlon = lon2 - lon1
    dlat = lat2 - lat1

    a = sin(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
    c = 2 * atan2(sqrt(a), sqrt(1 - a))

    distance = R * c
    return distance
```

```
launch_site_lat, launch_site_lon, city_lat, city_lon = 28.57299, -80.64621,25.761681,-80.191788
distance_city = calculate_distance(launch_site_lat, launch_site_lon, city_lat, city_lon)
distance_city
```

https://github.com/heath-barnett/ibm_ds_cert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_3_lab_jupyter_launch_site_location.jupyterlite.ipynb

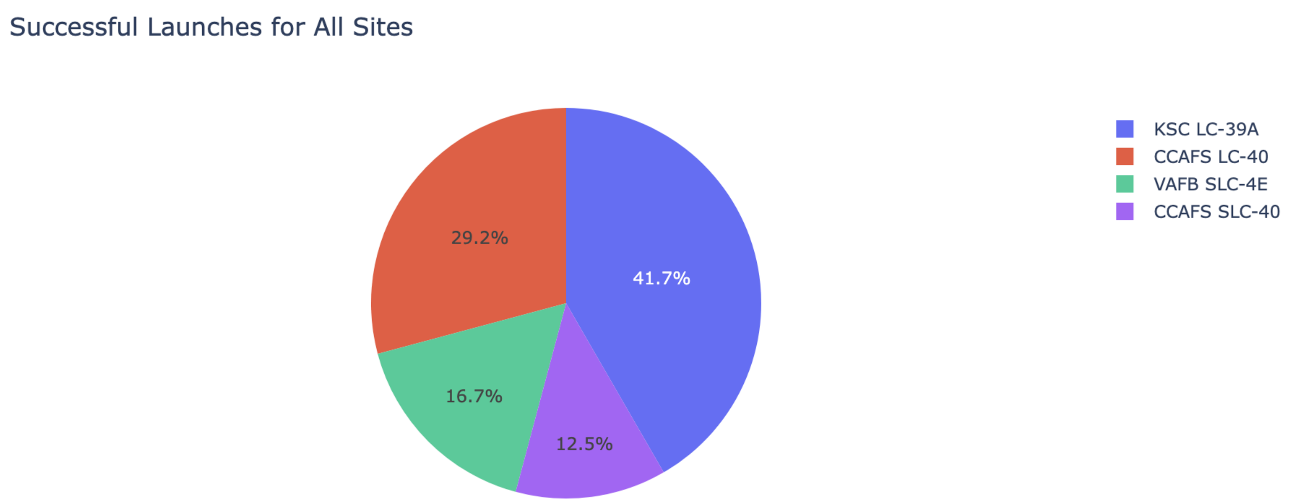
Distance of 315.92 km.



Section 4

Build a Dashboard with Plotly Dash

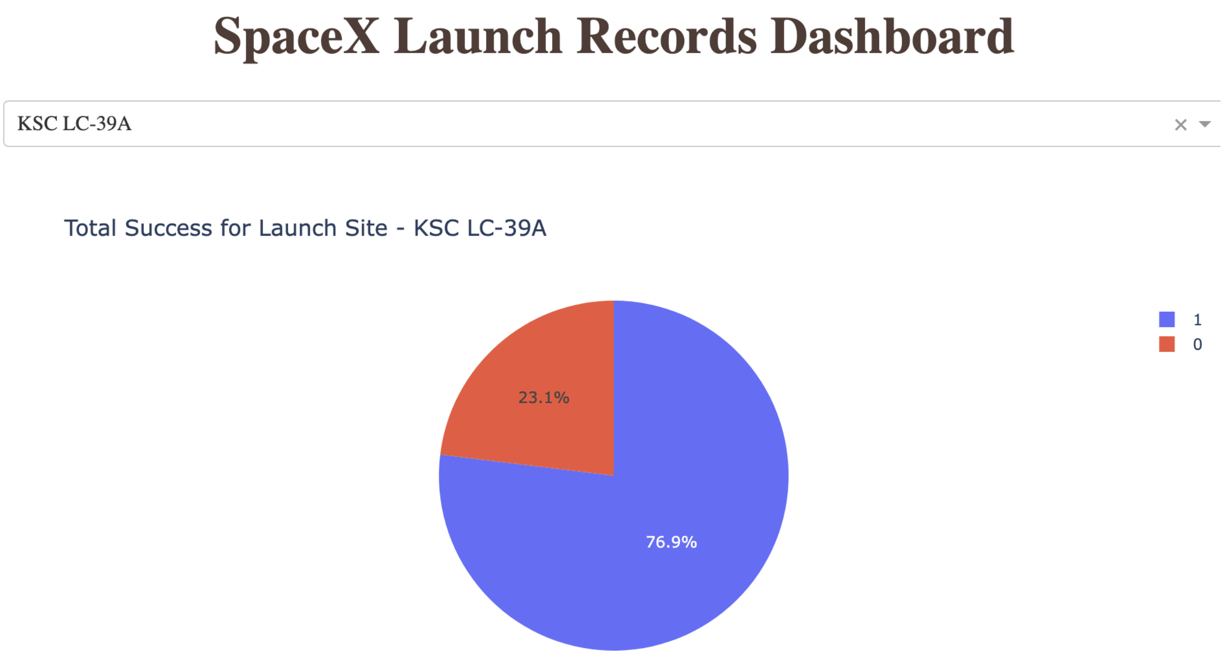
Launch Outcome for all Sites



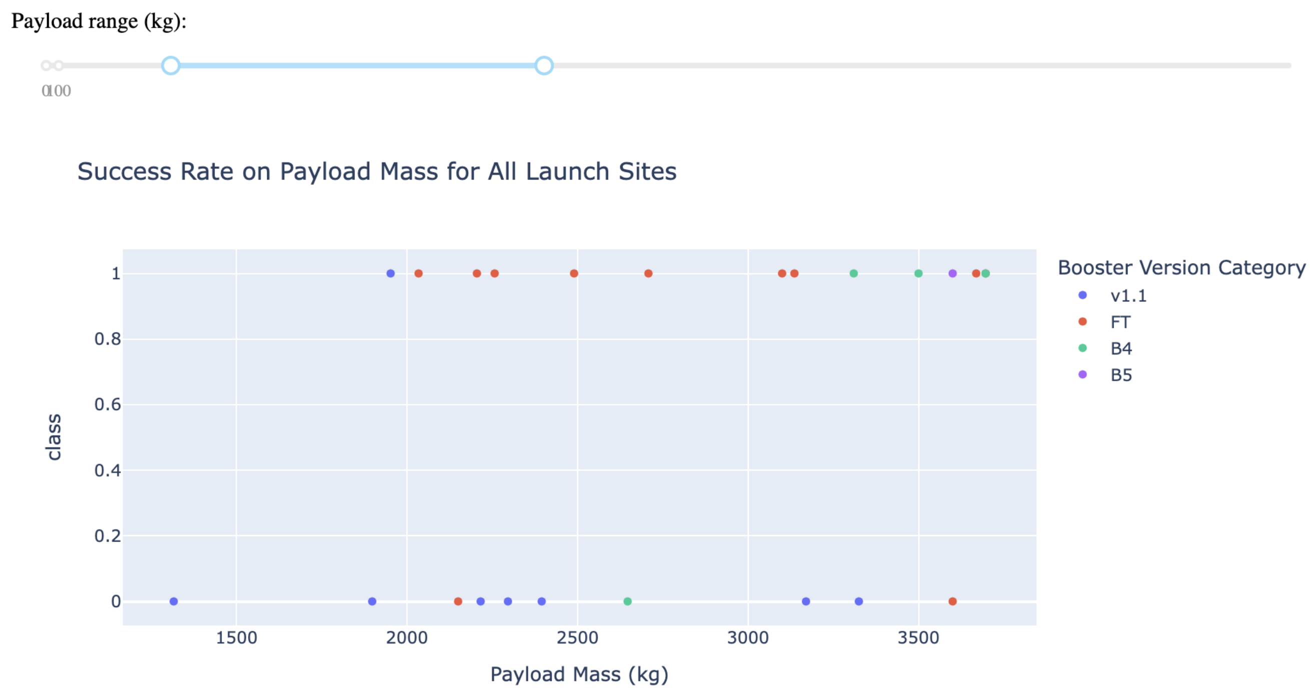
- Launch Outcome for All Sites
- KSC LC-39A has the highest success rate compared to all other launch sites.
- CCAFS SLC-40 has the worst success of all launch sites.

Launch Site with Highest Success Rate

- Launch site KSC LC-39A had the highest success rate out of all landing sites with a 76.9% success rate.
- The success/failure rate at this site is represented with a pie chart.



Payload vs. Launch Outcome

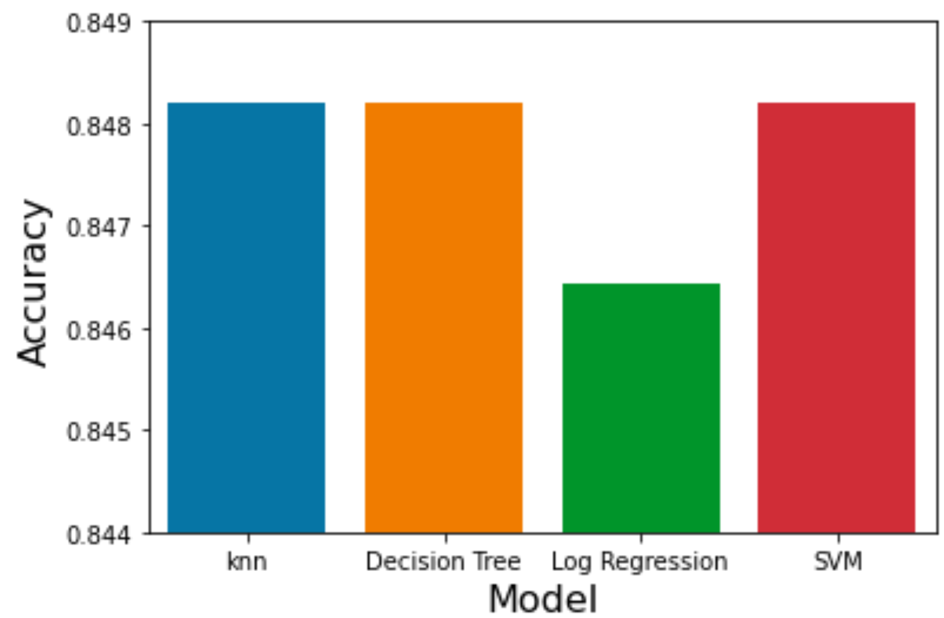


- Payload vs Launch Outcome for All Sites
- The FT booster version has the highest success rate compared to all other boosters in the payload range of 1000-4000 kg.

Section 5

Predictive Analysis (Classification)

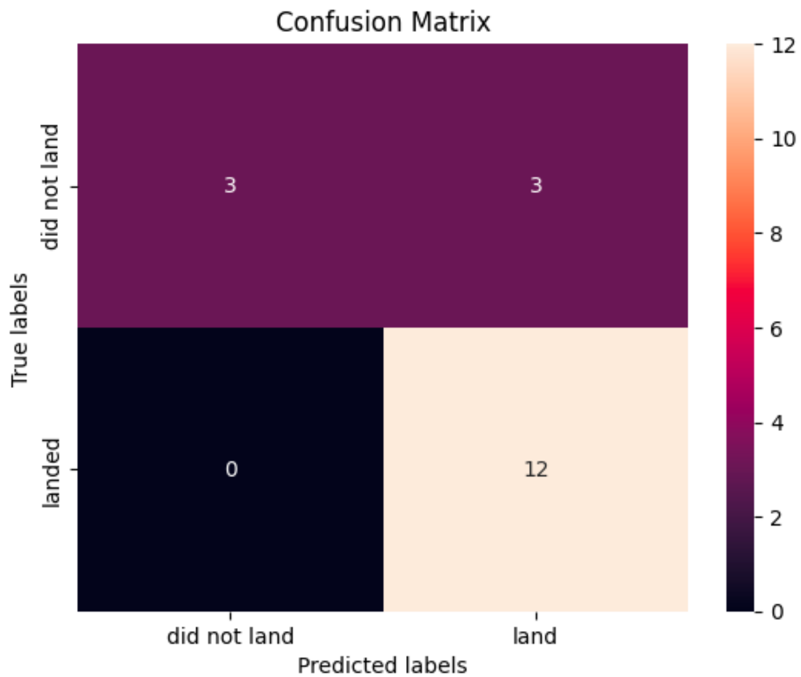
Classification Accuracy



Decision Tree Classifier was the most accurate on test data. However, it was not much better than other models.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Confusion Matrix
Decision Tree Classifier



- Decision Tree Classifier performed the best on training data with an accuracy of 87.7%.
- The model did not predict outcomes any better than other models investigated.
- False negatives are an issue with this model.

https://github.com/heath-barnett/ibm_ds_cert/blob/main/IBM-DS0321EN-SkillsNetwork_labs_module_4_SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Conclusions

Higher Flight Numbers is Correlated with Success

Success Rates have an upward trend since 2013.

Certain orbit launch targets are highly successful:

GEO, HEO, SSO, VLEO and ES-L1

Optimal launch site for success is KSC LC-39A

Decision Tree model was most accurate (marginally) but false negatives are an issue.

Thank You!