Winning Space Race with Data Science

Frederico Azevedo
08/19/2022

Executive Summary

- **Our goal**
  - To beat SpaceX in rocket launching
- **How?**
  - By offering a better price after determining the probability of a successful landing of SpaceX Falcon 9 first stages
  - Successful landing => first stage reuse => reduced launching costs
- **Methodology**
  - Data collection
  - Data wrangling
  - Exploratory data analysis (EDA)
  - Predictions using different machine learning models
- **Conclusion**
  - Predictive power of our models = 83% (too risky to compete)
  - We need better models and/or more data

Introduction

- **Commercial space transportation industry is in rapid ascension**

- **Key player:** SPACEX
  - Falcon 9 rocket: reuse of its first landing stage reduced launching costs from U$165 million to U$ 62 million

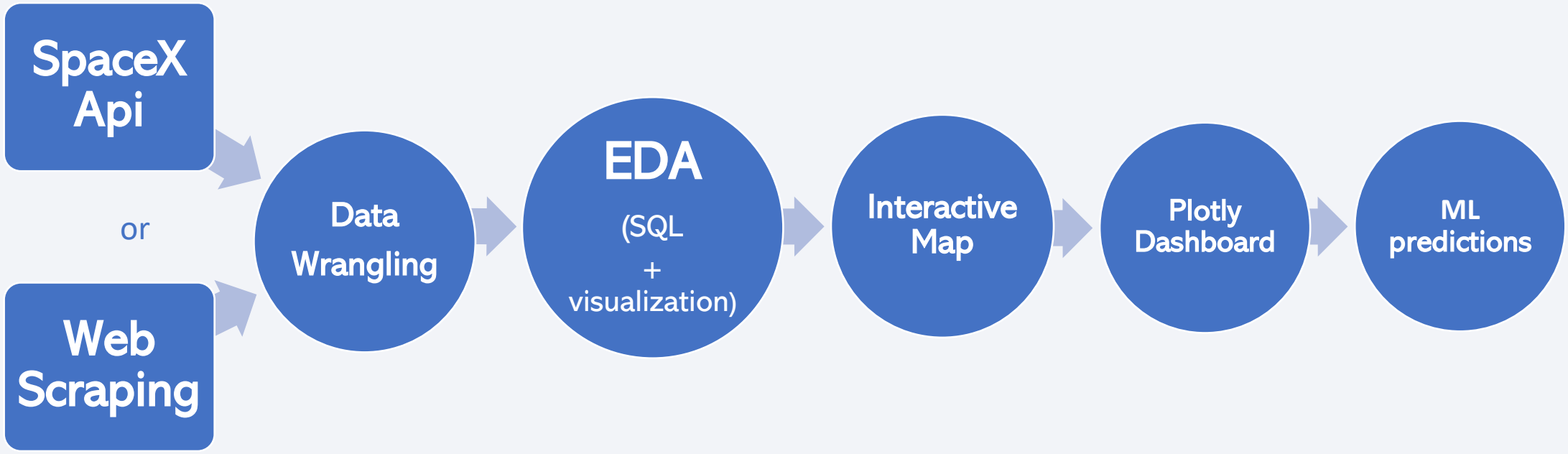
- **Not every landing is successful**
  - Launching site, payload mass, orbit type and booster versions are important factors for launch success
  - Costs of a launch are proportional to their success rate

- If we can **predict the probability of a successful landing**, we can strategically bid against SpaceX for a rocket launch

Section 1

Methodology

Data processing pipeline



Data Collection – SpaceX API

1. **Requested and parsed launch data from SpaceX API**
   - spacex_url = https://api.spacexdata.com/v4/launches/past
   - response = requests.get(spacex_url)
2. **Decoded the response content**
   - data = json.loads(response.text)
3. **Turned the response content into a Pandas dataframe**
   - data = pandas.json_normalize(data)
4. **Retrieved launch info from the API with custom functions**
   - getBoosterVersion(data); getPayloadData(data)
   - getLaunchSite(data); getCoreData(data)
5. **Constructed a launch dictionary with a subset of features from data**
   - launch_dict = {'FlightNumber': list(data['flight_number']), 'Date': list(data['date']), 'BoosterVersion': BoosterVersion, …}
6. **Created a Pandas dataframe containing only Falcon9 data**
   - dataframe = pandas.DataFrame(launch_dictionary)
7. **Filtered the dataframe to only include Falcon 9 launches**
   - data_falcon9 = data[data['BoosterVersion']!='Falcon 1'].copy()
8. **Replaced missing values with the mean of non-missing values**
   - mean_payloadmass = data_falcon9['PayloadMass'].mean(skipna=True)
   - data_falcon9['PayloadMass'].fillna(value=mean_payloadmass, inplace=True)

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	1	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
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
93	90	2020-11-05	Falcon 9	3681.0	MEO	CCSFS SLC 40	True ASDS	1	True	False	True	5e9e3032383ecb6bb234e7ca	5.0	7	B1062	-80.577366	28.561857

90 rows × 17 columns

Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook01_DataCollectionApi.ipynb

Data Collection – Web scraping

1. **Requested the Falcon9 Launch Wiki page from its URL**
   - spacex_url = https://api.spacexdata.com/v4/launches/past
   - response = requests.get(spacex_url)
2. **Parsed the html response with BeautifulSoup**
   - soup = BeautifulSoup(response.text, 'html.parser')
3. **Extracted the tables and columns from the soup object**
   - html_tables = soup.findAll('table')
   - th = html_tables[2].findAll('th')
4. **Extracted column_names and created a launch dictionary**
   - column_names = [name for row in th if (name:= extract_column_from_header(row)) is not None and len(name) > 0]
   - launch_dict = dict.fromkeys(column_names)
5. **Filled up the launch_dict using a custom function**
6. **Created a pandas dataframe from the dictionary**
   - df = pd.DataFrame(launch_dict)

	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	SpaceX	Success	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt	22 May 2012	07:44
...	...	...	...	...	...	...	...	...	...	...	...
120	121	CCSFS	SXM-8	7,000 kg	GTO	Sirius XM	Success	F9 B5	Success	6 June 2021	04:26

121 rows × 11 columns

Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook02_DataCollectionWebScraping.ipynb

Data Wrangling

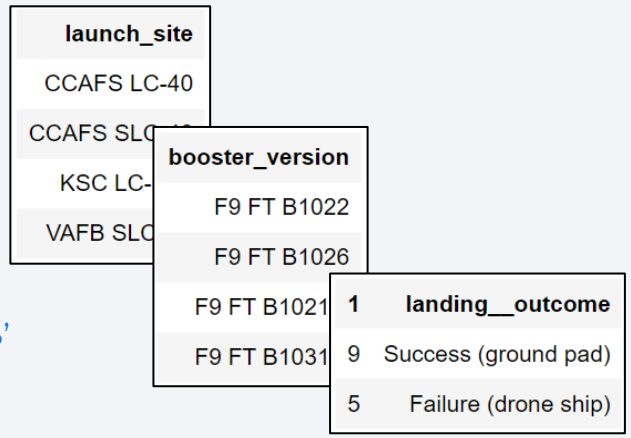
1. **Calculated the number of launches on each site**
   - df = dataframe_falcon9
   - df["LaunchSite"].value_counts()
2. **Calculated the number and occurrence of each orbit**
   - df["Orbit"].value_counts()
3. **Calculated number and occurrence of mission outcome per orbit type**
   - landing_outcomes = df["Outcome"].value_counts()
4. **Determined bad outcomes**
   - bad_outcomes = set(landing_outcomes.keys()[[1,3,5,6,7]])
5. **Created a landing outcome label from "outcome" column**
   - landing_class = [0 if item in bad_outcomes else 1 for item in df['Outcome']]
   - df['Class'] = landing_class.mean()
6. **Calculated landing success rate**
   - df["Class"].mean()
7. **Exported the dataframe to CSV format**
   - df.to_csv("dataset_part_2.csv", index=False)
8. **Stored it in a database**

	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

Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook03_DataWrangling.ipynb

EDA with SQL

1. **Loaded the SQL extension and established a connection with the database**
   - %load_ext sql
   - %sql ibm_db_sa://my-username:my-password@my-hostname:my-port/my-db-name?security=SSL
2. **Displayed the names of the unique launch sites**
   - %sql SELECT Launch_site FROM Spacexdataset GROUP BY launch_site
3. **Displayed 5 launch sites beginning with 'CCA'**
   - %sql SELECT Launch_site FROM Spacexdataset WHERE Launch_site LIKE 'CCA%' LIMIT 5
4. **Displayed the total payload mass of NASA boosters**
   - %sql SELECT SUM(Payload_mass__kg_) FROM Spacexdataset WHERE LOWER(Customer) LIKE 'nasa%'
5. **Displayed average payload mass of SpaceX F9 v1.1 boosters**
   - %sql SELECT AVG(Payload_mass__kg_) FROM Spacexdataset WHERE Booster_version LIKE 'F9 v1.1%'
6. **Listed the date of the first successful landing outcome in ground pad**
   - %sql SELECT MIN(DATE) FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%success%ground%pad%



launch_site
CCAFS LC-40
CCAFS SLC
KSC LC-
VAFB SLC

booster_version
F9 FT B1022
F9 FT B1026
F9 FT B1021
F9 FT B1031

1	landing__outcome
9	Success (ground pad)
5	Failure (drone ship)

Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook04_EdaSql.ipynb

EDA with SQL

7. **Listed the names of the boosters successful in drone ship landing that have payload mass greater than 4000 but less than 6000**
   - %sql SELECT Booster_version FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%success%drone%ship%' AND Payload_mass__kg_ > 4000 AND Payload_mass__kg_ < 6000

8. **Listed the total number of successful and failed missions**
   - %sql SELECT COUNT(*) FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%success%' OR LOWER(Landing__outcome) LIKE '%failure%'

9. **Listed the booster names that carried the maximum payload mass**
   - %sql SELECT Booster_version, Payload_Mass__Kg_ FROM Spacexdataset WHERE Payload_mass__kg_ = (Select MAX(Payload_mass__Kg_) FROM Spacexdataset)

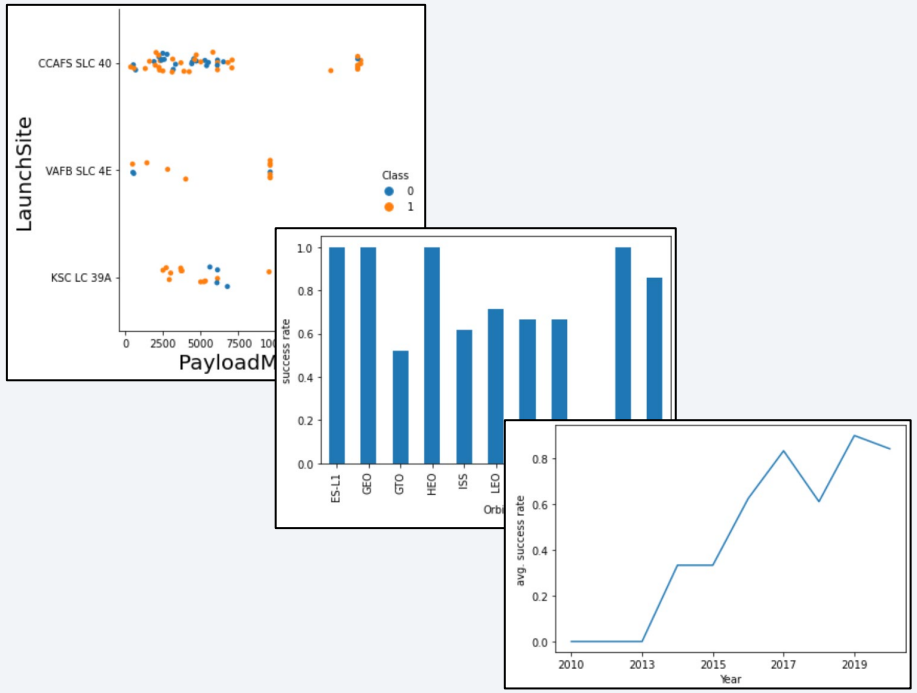
10. **Listed the failed drone ship landings, their booster versions, and launch site names in 2015**
    - %sql SELECT Date, Landing__outcome, Booster_version, Launch_site FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%failure%drone%ship%' AND YEAR(Date) = 2015

11. **Ranked the count of landing outcomes between 2010-06-04 and 2017-03-20 in descending order**
    - %sql SELECT COUNT(Landing__outcome), Landing__outcome FROM Spacexdataset WHERE (LOWER(Landing__outcome) LIKE '%success%ground%pad%' OR LOWER(Landing__outcome) LIKE '%failure%drone%ship%') AND Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing__outcome ORDER BY COUNT(Landing__outcome) DESC

Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook04_EdaSql.ipynb

EDA with Data Visualization

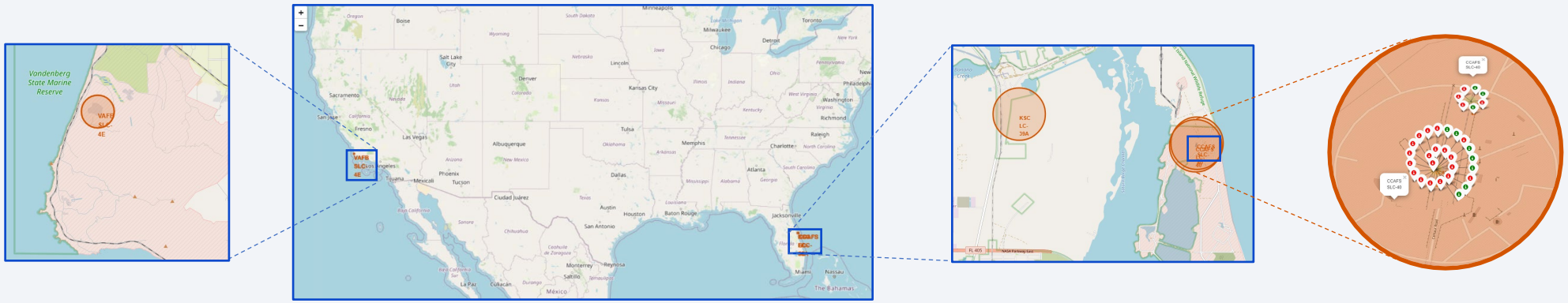
1. **Read the SpaceX dataset into a pandas dataframe**
   - df = pd.read_csv(csv_url)
2. **Displayed the plot relationships:**
   - Payload mass vs flight number
   - Launch site vs flight number
   - Launch site vs payload mass
   - Orbit vs flight number
   - Orbit vs payload mass
   - Success rate per orbit type
   - Average success rate per year
3. **One-hot encoded categorial variables with float casting**
   - pd.get_dummies(data=data, columns=['Orbit', …], dtype=float)



Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook05_EdaDataVizualization.ipynb

Interactive Map with Folium

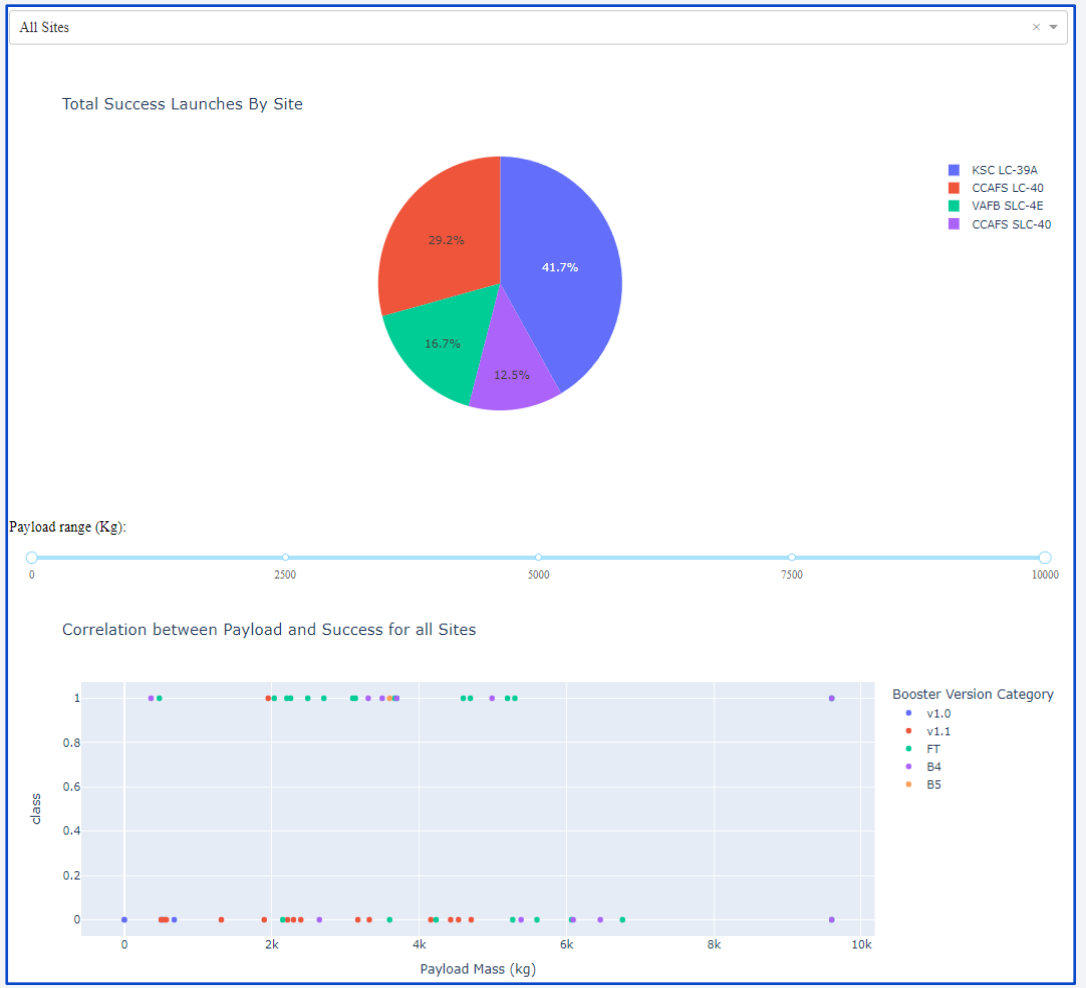
1. **Created a folium map object centered at NASA coordinates**
   - site_map = folium.Map(location=coords, zoom_start=10)
2. **Marked NASA with a blue circle and a marker**
   - circle = folium.Circle(coords, radius=1000, color='#d35400', fill=True).add_child(folium.Popup('NASA'))
   - marker = folium.map.Marker(coords, icon=icon_parameters))
   - site_map.add_child(circle); site_map.add_child(marker)
3. **Marked the success/failed launches on the map as clusters of markers**
   - # Similar code as above
4. **Calculated the Euclidean distance of arbitrary points to the launch sites**
   - # Custom function
5. **Traced a "folium.PolyLine" object between two of the points above**
   - lines=folium.PolyLine(locations=[[proximity_lat, proximity_lon], [launch_site_lat, launch_site_lon]], weight=1)
   - site_map.add_child(lines)



Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook06_FoliumLab.ipynb

Plotly Dashboard

1. **Added input components: dropdown & range slider**
   - dcc.Dropdown(…)
   - dcc. dcc.RangeSlider()
2. **Added output components: pie charts & scatter plot**
   - fig = px.pie(dataframe, values=…, names=…)
   - fig = px.scatter(dataframe, x=…, y=…)
3. **Linked the dropdown with pie charts through a callback**
   - @app.callback(Output(component_id='pie_chart', component_property='figure'), Input(component_id='dropdown', component_property='value'))
   - def get_pie_chart(entered_site): …
4. **Linked the range slider with the scatter through a callback**
   - @app.callback(Output(component_id='scatter', component_property='figure'), Input(component_id='dropdown', component_property='value'), Input(component_id='slider', component_property='value')])
   - def get_scatter(entered_site, payload_range): …

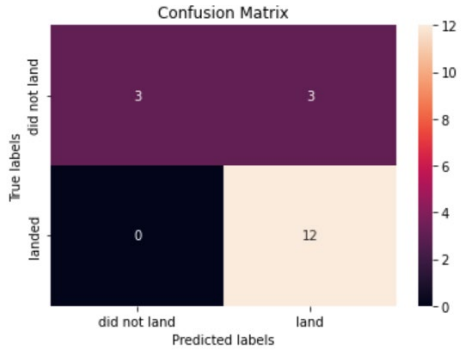


Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook07_Dash.ipynb

Predictive Analysis (Classification)

1. **One-hot encoded categorial variables with float casting**
   - X = pd.get_dummies(data=data, columns=['Orbit', …], dtype=float)
2. **Extracted target**
   - y = data['Class'].to_numpy()
3. **Standardized X**
   - transform = preprocessing.StandardScaler()
   - X = transform.fit_transform(X)
4. **Split dataset: 80% train, 20% test**
   - X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=2)

4. **Selected 4 models:** KNN, Decision Tree, SVM, Logistic Regression
5. **Tuned model hyperparameters with cross validation and grid search**
   - svm = SVC(); # Support Vector Machine object
   - svm_cv = GridSearchCV(svm, params, cv=10)
   - svm_cv.fit(X_train,y_train)
6. **Evaluated models on the test set**
   - Confusion matrices
   - Accuracies
   - Jaccard scores
   - F1-scores
   - Log loss for logistic regression



Algorithm	Accuracy	Jaccard	F1-score	Logloss
KNN	0.83	0.8	0.81	NA
Decision Tree	0.83	0.8	0.81	NA
SVM	0.83	0.8	0.81	NA
Logistic Regression	0.83	0.8	0.81	0.48

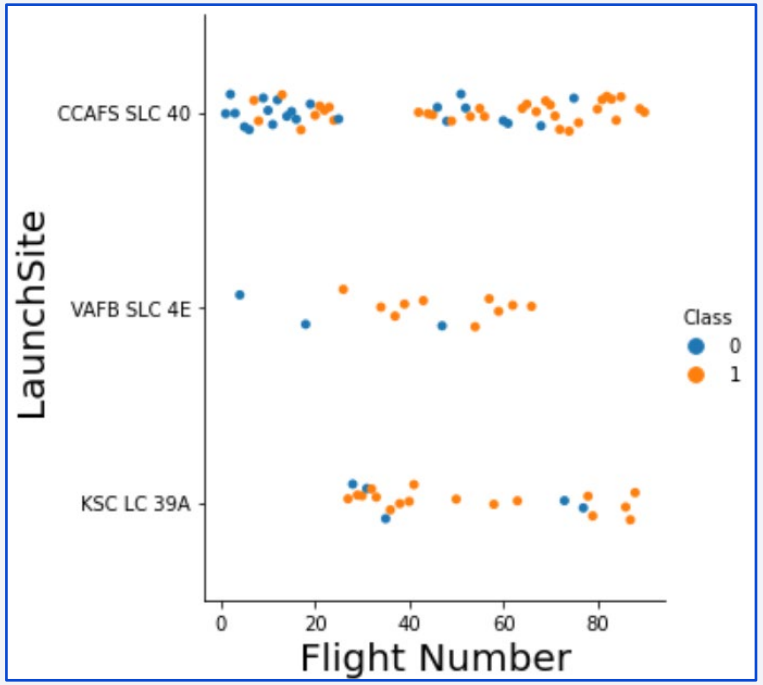
Code: https://github.com/facazevedo/IBM_data_analysis_course10_coursera/blob/main/Notebook08_MLPred.ipynb

Section 2

Insights drawn from EDA

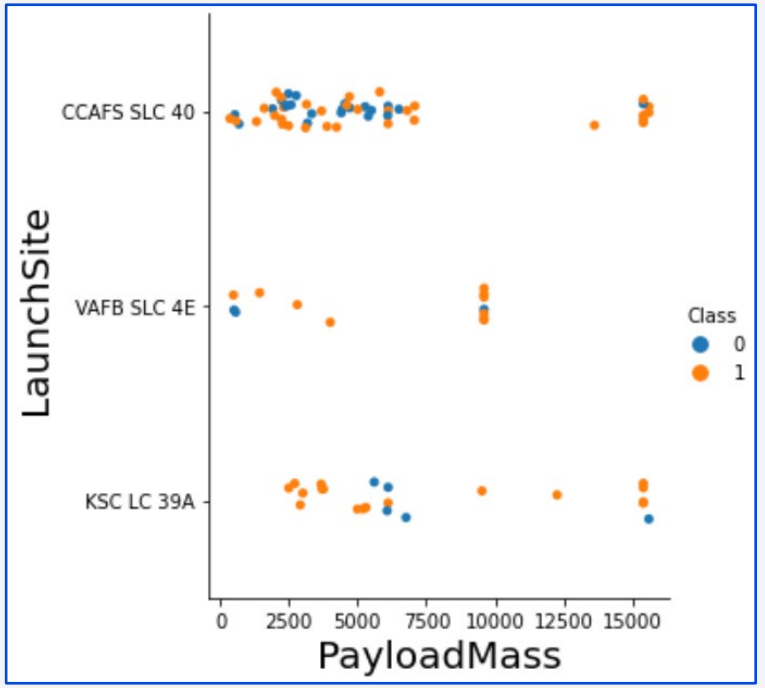
Flight Number vs. Launch Site

1. Most of the early launches used the site CCAFS SLC 40 but there were more failures than successes
2. SpaceX switched the launches to KSC LC 39A
3. After mostly successful launches, they returned to CCAFS SLC 40 with approximately the same success rate as KSC LC 39A
4. Most of launches were done at CCAFS SLC 40. Probably this site has better costs and/or infrastructure
5. Sporadic launches were tried in VAFB SLC 4E with most successes
6. In general, there seems to be positive relationship of between flight number and success for each launch site



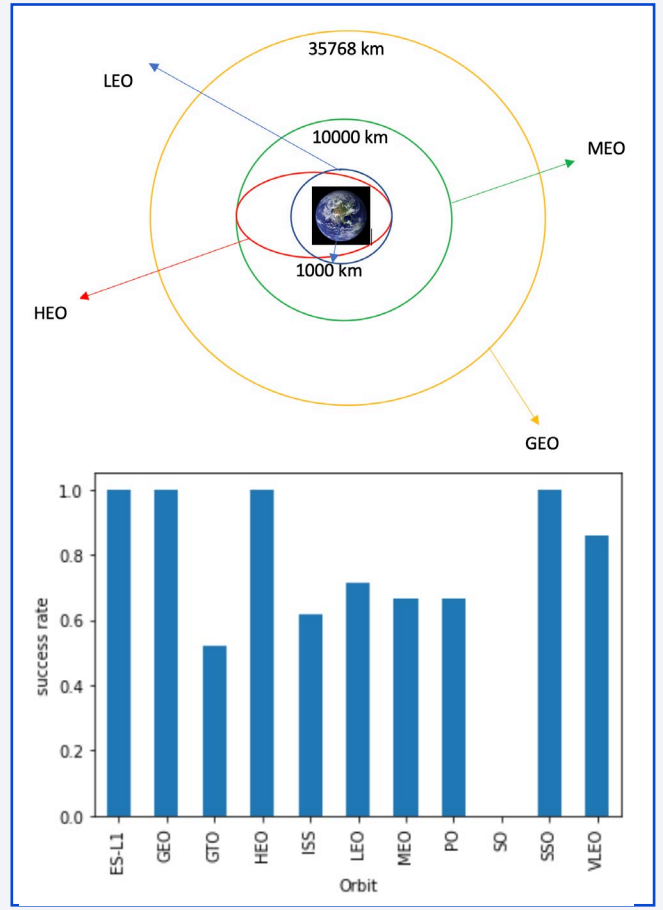
Payload vs. Launch Site

1. Launches with lighter payload mass (below 8k kg) were done mainly at CCAFS SLC 40
2. Sporadic launches with lighter payload were done at VAFB SLC 4E and KSC LC 39A
3. Most of the launches with10k kg payloads were done at VAFB SLC 4E
4. Heavy payloads (above 15k kg) were launched only at CCAFS SLC 40 and KSC LC 39A
5. In general, most of the failures happened in launches with lighter payload masses, probably because they were early trials



Success Rate vs. Orbit Type

1. Orbits ES-L1, GEO, HEO and SSO had 100% success rate but not many launches to those orbits were performed
2. VLEO is the next most successful orbit, with ~86% success rate for 14 launches
3. Orbits GTO, ISS, LFO, MEO and PO had between 50% to 70% success rate
4. Orbits GTO and ISS were the most tried ones with 27 and 21 launches respectively
5. Orbit SO has 0% success rate but it was only tried once

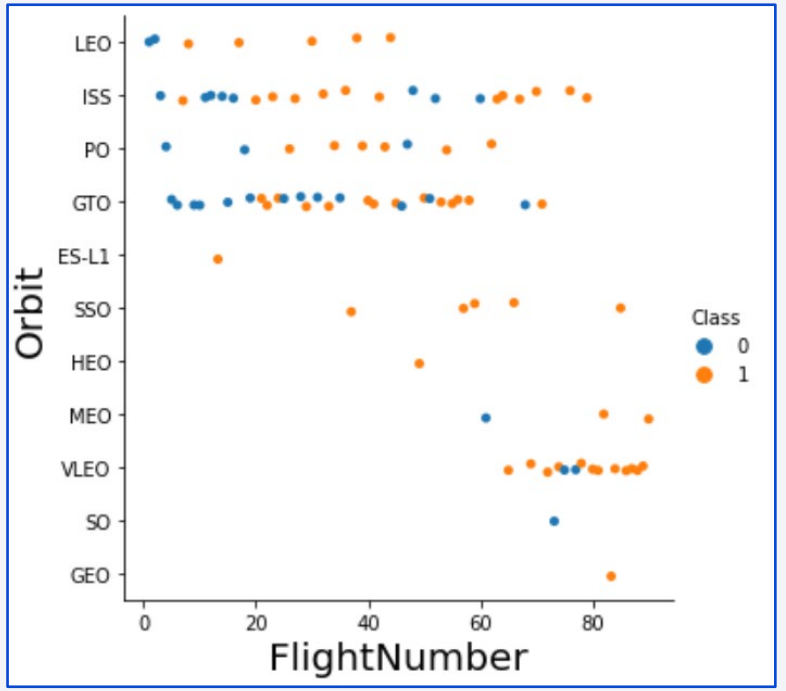


ES-L1 – European Space Agency Lagrange Point 1
GEO – Geostationary Orbit
GTO – Geostationary transfer orbit
HEO – Highly Elliptical Orbit
ISS – International Space Station Orbit
LEO – Low Earth Orbit
MEO – Medium Earth Orbit
PO – Polar Orbit
SO – Sun Orbit
SSO – Sun Synchronous Orbit
VLEO – Very Low Earth Orbit

Total launches:
ES-L1 = 1
GEO = 1
GTO = 27
HEO = 1
ISS = 21
LEO = 7
MEO = 3
PO = 9
SO = 1
SSO = 5
VLEO = 14

Flight Number vs. Orbit Type

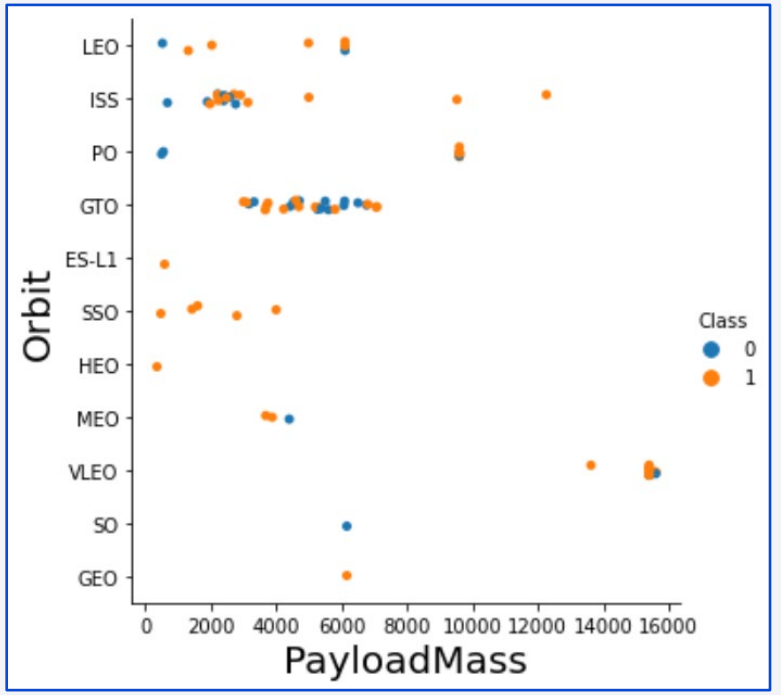
1. The first launches aimed at orbits LEO, ISS, PO and GTO respectively
2. After some initial failures, launches to those orbits started succeeding
3. SpaceX decided then to try new orbits as well, such as SSO, HEO, MEO, VLEO, SO and GEO
4. ESL1 was successfully tried at the initial stages but only once
5. SO and GEO were also tried only once but at later stages. Only the latter succeeded.
6. Most of the late launches aimed at VLEO
7. Success seems to be related to flight number in LEO and ISS but not in GTO



ES-L1 – European Space Agency Lagrange Point 1
GEO – Geostationary Orbit
GTO – Geostationary transfer orbit
HEO – Highly Elliptical Orbit
ISS – International Space Station Orbit
LEO – Low Earth Orbit
MEO – Medium Earth Orbit
PO – Polar Orbit
SO – Sun Orbit
SSO – Sun Synchronous Orbit
VLEO – Very Low Earth Orbit

Payload vs. Orbit Type

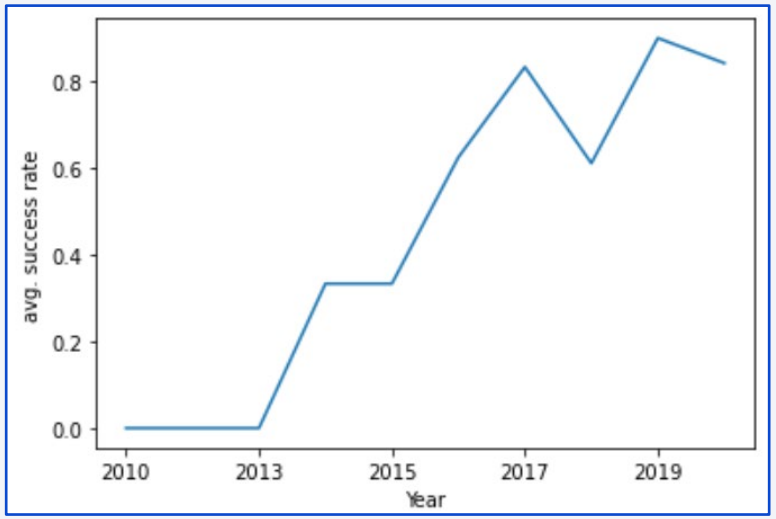
1. Most of the launches of medium-light payload masses (around 3k kg) aimed orbit ISS
2. Most of the launches containing medium payloads (from 4k to 8k kgs) launched to orbit GTO
3. Heavy payload masses (above 13k kg) were launched only to VLEO
4. Even though there seems to be a positive trend for payload mass and success for orbits LEO, ISS and PO, the relationship isn't clear
5. In general, there is no obvious relationship between payload mass and success for any orbit



Launch Success Yearly Trend

1. In general, there is a clear positive relationship between success rate and years

2. The strong dip in 2018 and the light dip in 2020 could be related to early tests of innovations



All Launch Site Names

- %sql SELECT Launch_site FROM Spacexdataset GROUP BY launch_site

For this query, we just need to group all the rows of launch_site and display the result

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

Launch Site Names Begin with 'CCA'

- %sql SELECT Launch_site FROM Spacexdataset WHERE Launch_site LIKE 'CCA%' LIMIT 5

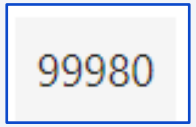
For this query, we just need the display the rows of launch_site that start with the string CCA and limit this display to 5 rows

CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40
CCAFS LC-40

Total Payload Mass

- %sql SELECT SUM(Payload_mass__kg_) FROM Spacexdataset WHERE LOWER(Customer) LIKE 'nasa%'

For this query, we just need to select all the rows that start with the string “nasa” and display the sum of their payload mass. The function lower() is used to avoid mismatches due to uppercase of characters



Average Payload Mass by F9 v1.1

➢ %sql SELECT AVG(Payload_mass__kg_) FROM Spacexdataset WHERE Booster_version LIKE 'F9 v1.1%'

Here we just need to select the rows of Booster_version that start with “F9 v1.1” and display the average of their payload mass

2534

First Successful Ground Landing Date

- %sql SELECT MIN(DATE) FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%success%ground%pad%

For this query, we just need to select the rows of landing__outcome that contain the words “success”, “ground” and “pad” and display the minimum date of those rows

2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

➢ %sql SELECT Booster_version FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%success%drone%ship%' AND Payload_mass__kg_ > 4000 AND Payload_mass__kg_ < 6000

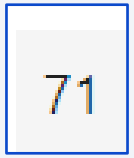
One way to solve this task is to select the rows of Landing__outcome that contain the words “success”, “drone” and “ship”, and that contain a payload mass greater than 4k and lower than 6k

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

Total Number of Successful and Failure Mission Outcomes

- %sql SELECT COUNT(*) FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%success%' OR LOWER(Landing__outcome) LIKE '%failure%'

Here we just need to select the rows of Landing__outcome that contain the word “success” or the word “failure” and display the row count



Boosters Carried Maximum Payload

- %sql SELECT Booster_version, Payload_Mass__Kg_ FROM Spacexdataset WHERE Payload_mass__kg_ = (Select MAX(Payload_mass__Kg_) FROM Spacexdataset)

One way to solve this task is to use a subquery to select the rows with max payload mass and then display their booster version and the values of their payload mass

booster_version	payload_mass__kg_
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

Failed drone ship landings in 2015

- %sql SELECT Date, Landing__outcome, Booster_version, Launch_site FROM Spacexdataset WHERE LOWER(Landing__outcome) LIKE '%failure%drone%ship%' AND YEAR(Date) = 2015

Here we just need to select and display the rows of Landing__outcome that contain the words “failure”, “drone” and “ship” and the year 2015

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

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

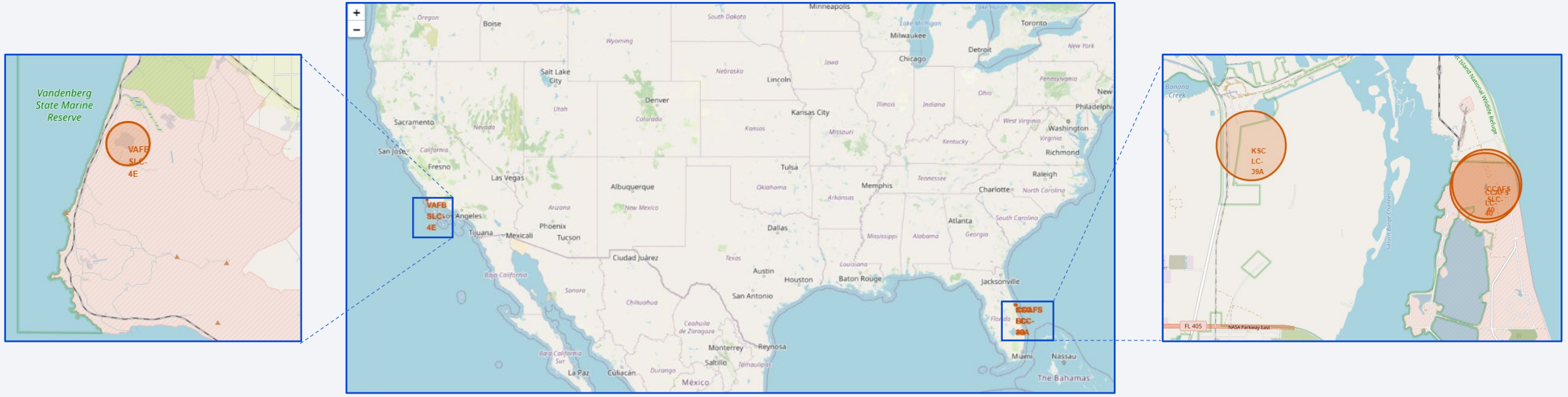
➢ %sql SELECT COUNT(Landing__outcome), Landing__outcome FROM Spacexdataset WHERE (LOWER(Landing__outcome) LIKE '%success%ground%pad%' OR LOWER(Landing__outcome) LIKE '%failure%drone%ship%') AND Date BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY Landing__outcome ORDER BY COUNT(Landing__outcome) DESC

One way to solve this task is to select the rows of Landing_outcome that contains the words “success”, “ground” and “pad” in a sentence or the words “failure”, “drone” and “ship”. Of those rows, we select the ones that contain dates between '2010-06-04' AND '2017-03-20’. Finally, we group them by landing outcome and we display them by the count of outcomes in a descending order

5	Failure (drone ship)
3	Success (ground pad)

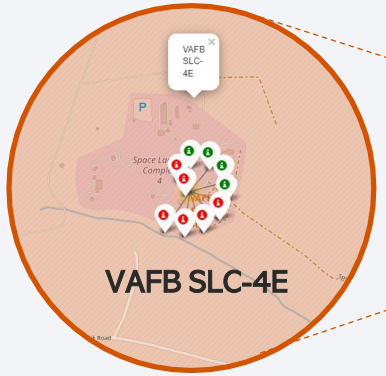

Section 3
Launch Sites
Proximities Analysis

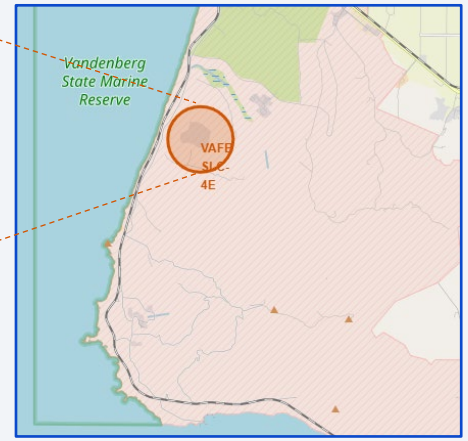
Launch site locations

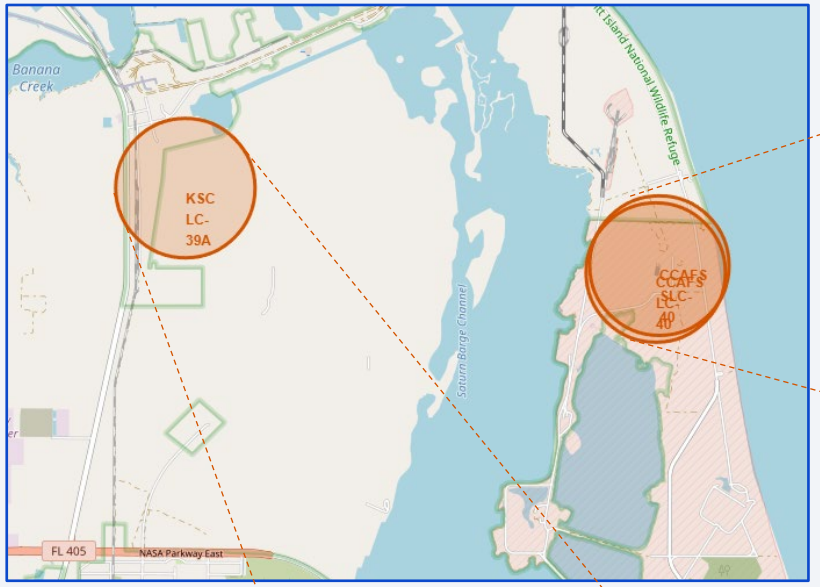


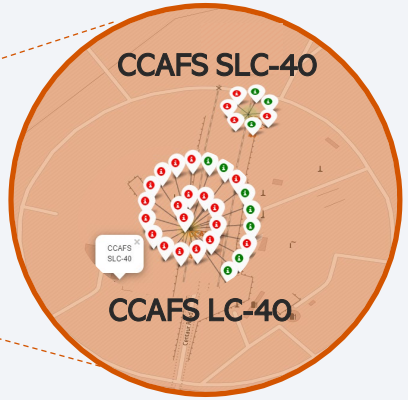
- 3 launch sites are in Florida: the Kennedy Space Center Launch Complex 39 (KSC LC-39A), the Cape Canaveral Air Force Station Launch Complex 40 (CCAFS LC-40) and the Cape Canaveral Air Force Station Space Launch Complex 40 (CCAFS SLC-40)
- 1 launch site is in California: the Vandenberg Air Force Base Space Launch Complex 4E (VAFB SLC-4E)

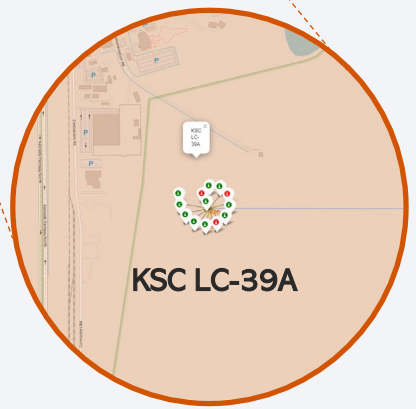
Launch outcomes on each site







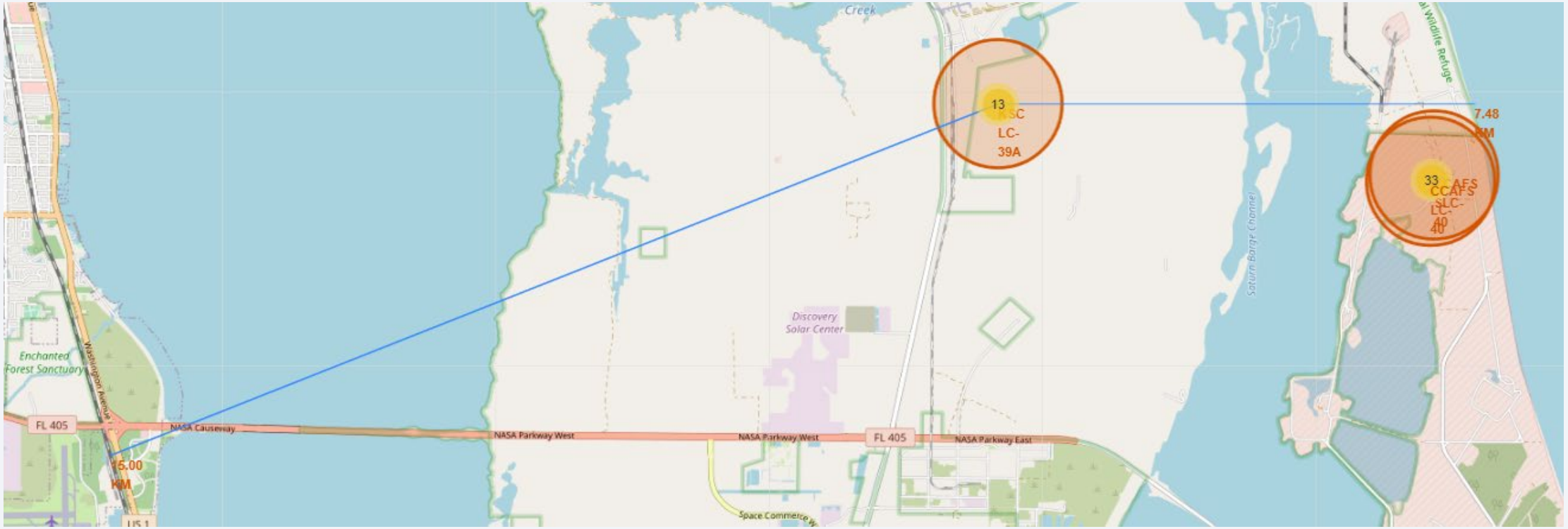




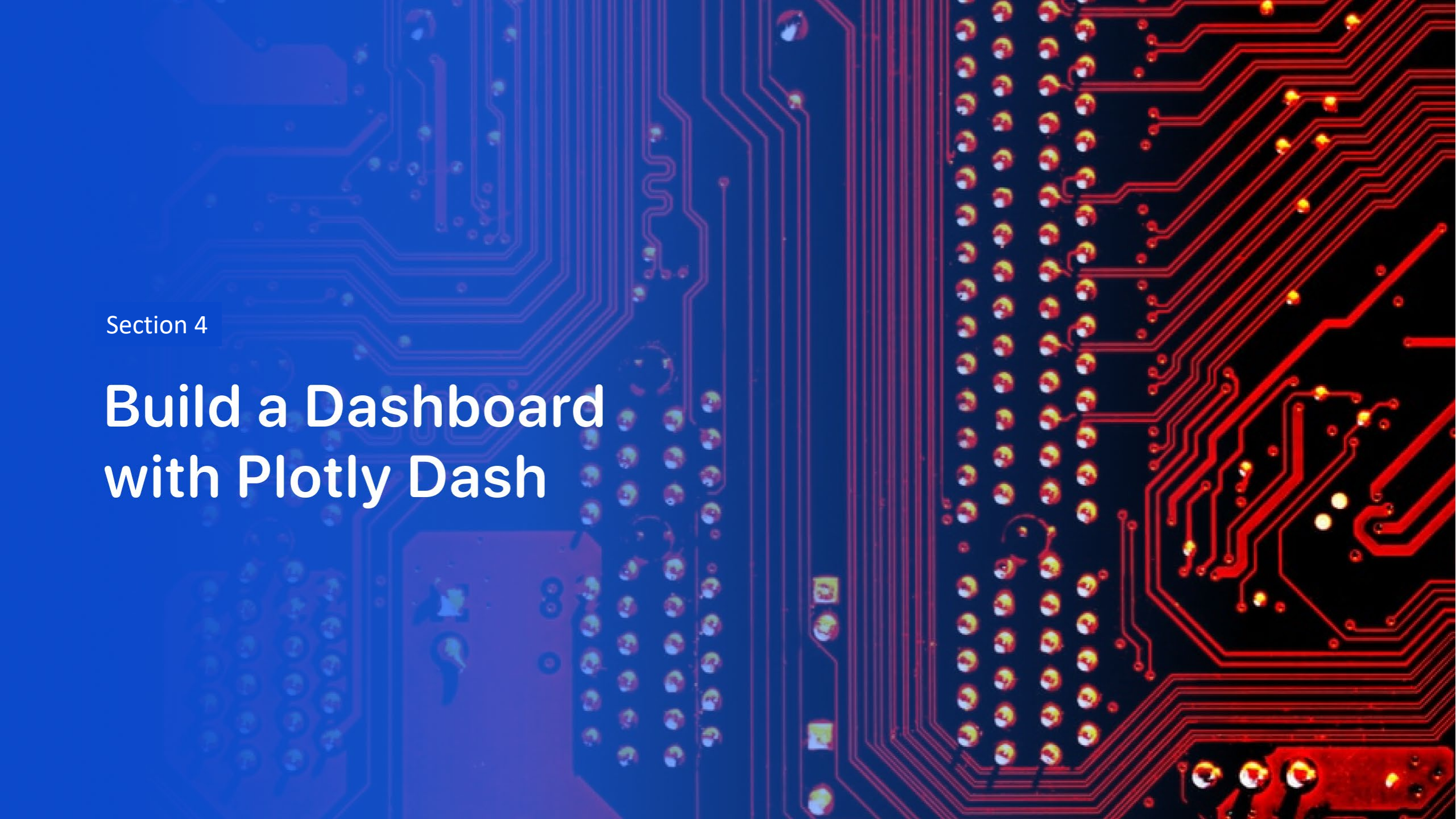
Success to failure ratio:

- VAFB SLC-4E = 4 / 6
- KSC LC-39A = 10 /3
- CCAFS SLC-40 = 7 / 19
- CCAFS SLC-40 = 3 / 4

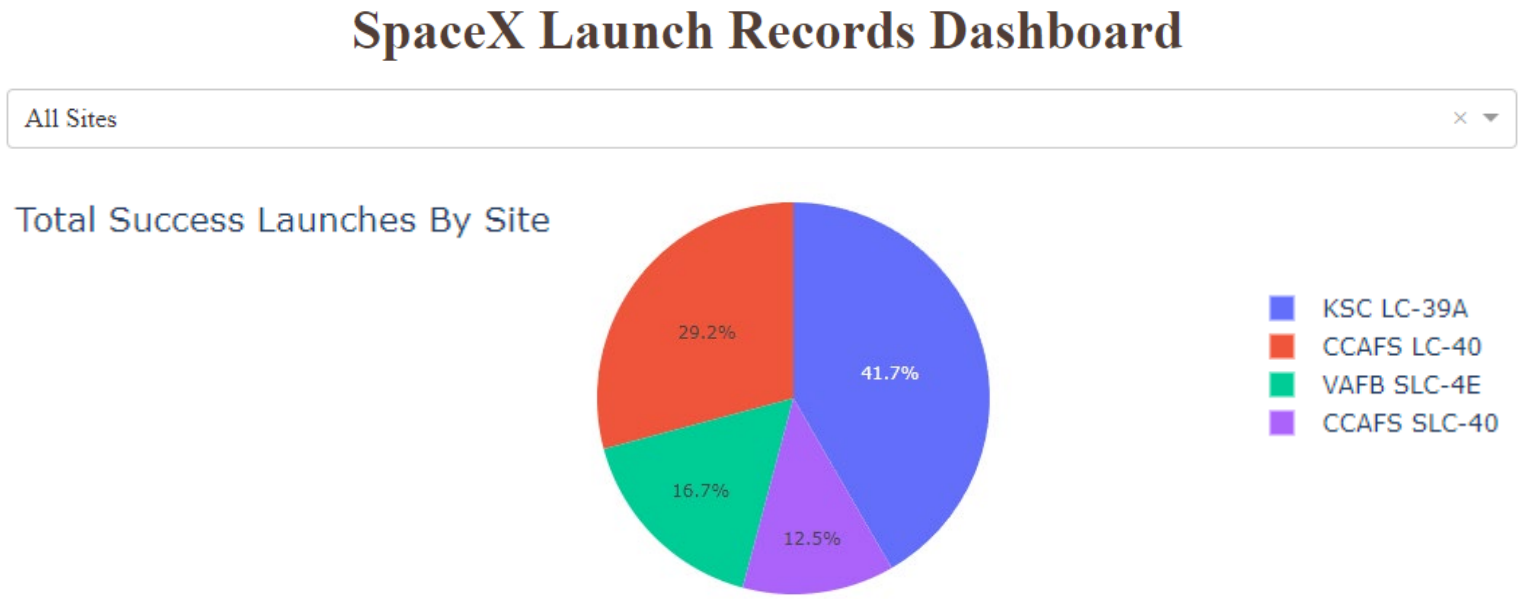
Distance of KCS LC-39A to coast and railway



- KSC LC-39A is at approximately 7.48 km of the east coast and at approximately15 km of the intersection of the Florida East Coast Railway with the Washington Avenue

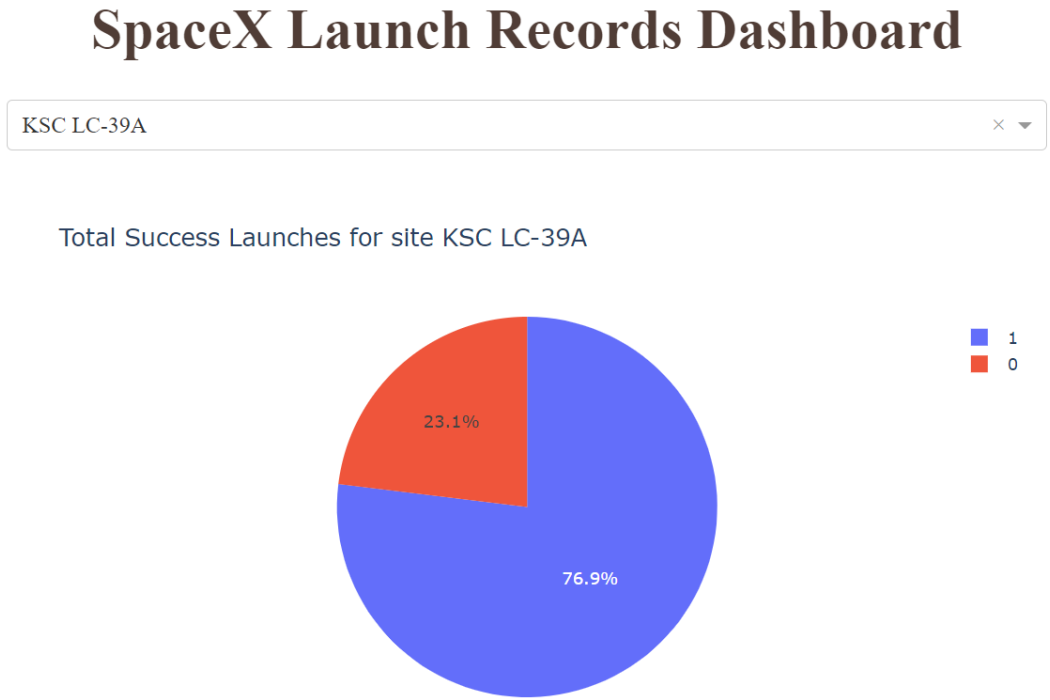

Section 4
Build a Dashboard with Plotly Dash

Proportions of successful launches by site



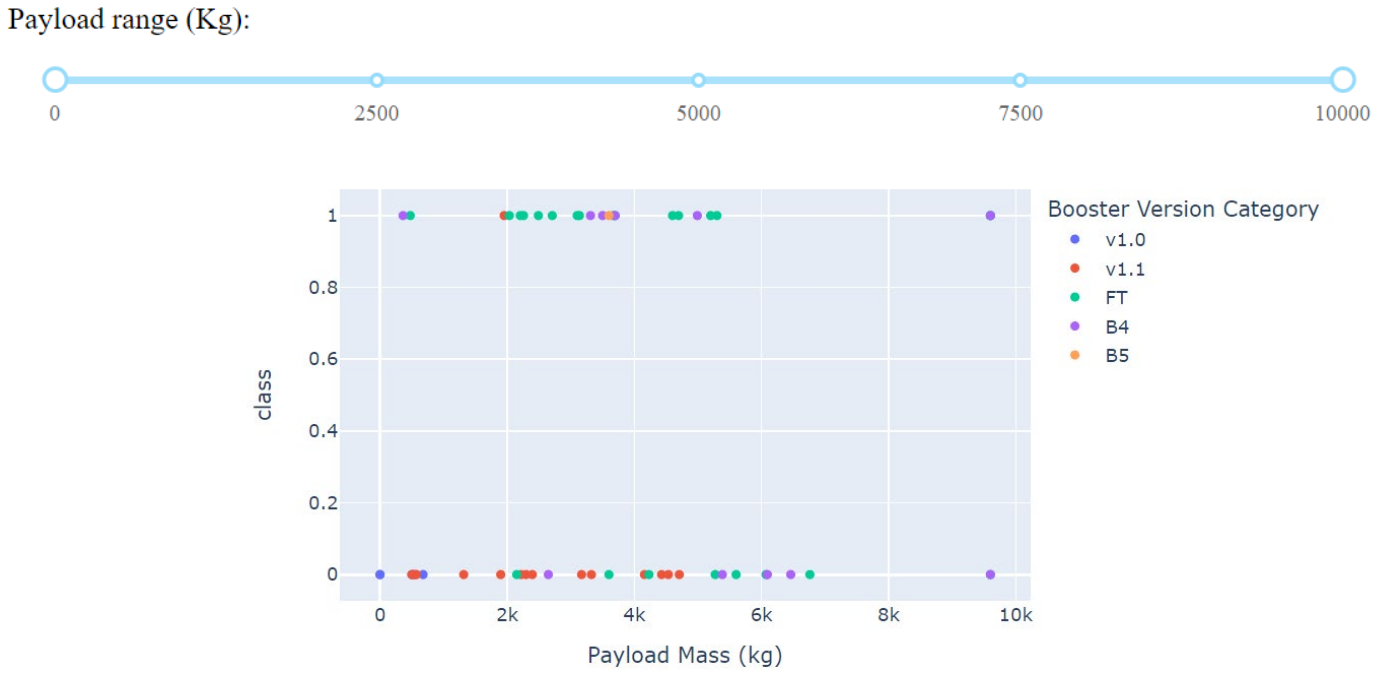
- The KSC LC-39A is the most successful launch site, with ~42% of all successful launches (10 out of all 24 successful launches)
- The CCAFS LC-40 is the runner up, with ~29% of all successful launches (7 of 24)
- The VAFB SLC-4E and the CCAFS SLC-40 come next with ~17% (4 of 24) and ~12% (3 of 24) of all successes respectively

Launch success rate at the KSC LC-39A



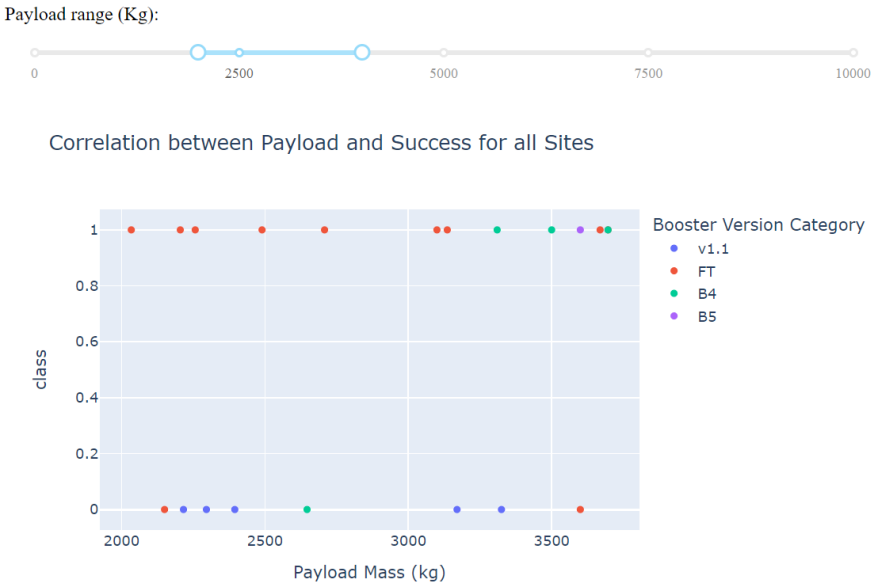
- 13 launches were performed at the KSC LC-39A.
- ~77% (10 out of 13) were successful launches and ~23% (3 of 13) were failures

Correlation between Payload and success for all sites



- We can separate the plot above in 4 ranges of payload mass: below 2K, 2k to 4k, 4k to 6k, above 6k
- The 2k-4k range seems to be have the highest success to failure ratio. In this range, the FT Booster seems to work best
- The 6k-8k range is the least successful, with 0 successes

Selected ranges



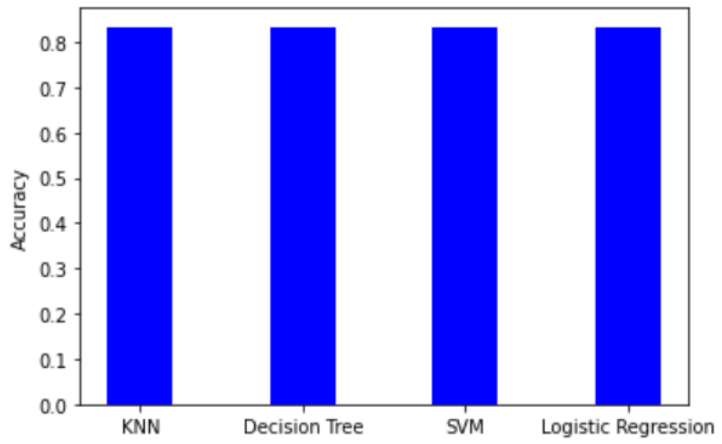
- The 2k-4k range has the highest success rate: 12 out of 20 (60%) launches were successful
- In this range, the FT Booster worked best, with 8 of 10 successes (80%), followed by the B4 Booster, with 3 of 4 successes (75%)
- 4 launches were tried in the 6k-8k range but none of them succeeded

Section 5

Predictive Analysis (Classification)

Model evaluation

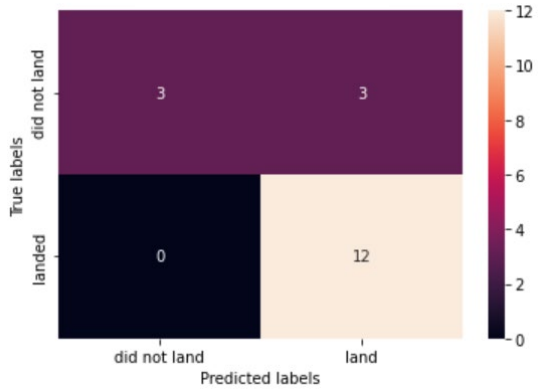
- All models performed similarly
  - Accuracy = 0.83
  - Jaccard index = 0.8
  - F1-score = 0.81



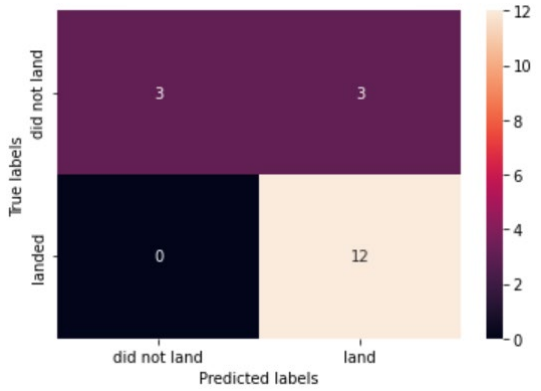
Algorithm	Accuracy	Jaccard	F1-score	Logloss
KNN	0.83	0.8	0.81	NA
Decision Tree	0.83	0.8	0.81	NA
SVM	0.83	0.8	0.81	NA
Logistic Regression	0.83	0.8	0.81	0.48

Confusion Matrices

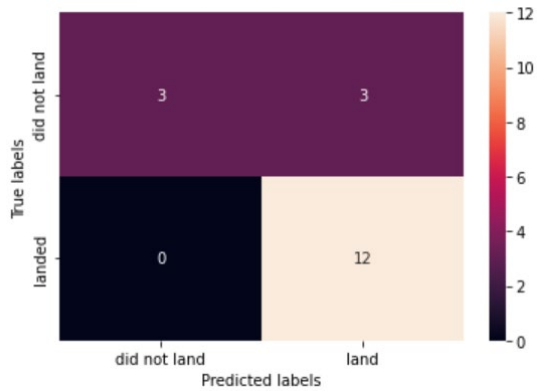
KNN



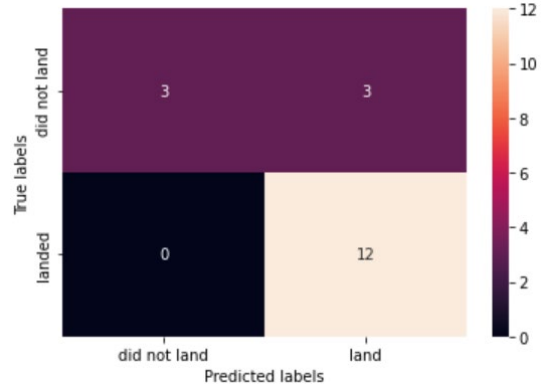
Decision Tree



SVM



Logistic Regression



- The confusion matrices of all models were similar
- Out of 18 cases in the test set, the models correctly predicted that 3 of them did not land and that 12 of them landed
- There were no false negatives; however, there were 3 false positives
- This bias to positive predictions might be related to our small (n = 90) and unbalanced dataset (2x more positive than negative outcomes)

Conclusions

- Most of launches were done in CCAFS SLC-40, with light or heavy payloads
- Early launches were to orbits LEO, ISS, PI and GTO but the focus of late launches aimed mainly orbit VLEO
- VLEO had a success rate of ~86% for 14 launches
- Success rate trend increased through the years
- There was no clear relationship between success rate and payload mass
- Our models were able to predict landing outcome in 83% of the cases

Should we compete against SpaceX?

- A prediction power of 83% is not good enough
- We need better models and/or more data to increase our predictive power
- 95% would be a strategic cut off

Thank you!