IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

V. Kogan
1st of May 2024

Outline

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

Executive Summary

This investigation delves into the drivers behind successful SpaceX rocket landings. The various data science techniques employed are listed below.

Summary of Methodologies

- Web scraping: Using REST API
- Data wrangling to transform it into a format fit for further analysis
- Data exploration: Visualization and trend analysis
- Machine learning: Dividing data into a train and test sets, employing logistic regression, support vector machines, and k-nearest neighbors methods

Results

- Positive trend can be observed in terms of success ratio over time
- Most launch sites are situated near the coast and are around the equator
- Different machine learning techniques yielded similar results

Introduction

Project background and context

Space X advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because Space X can reuse the first stage. Therefore, if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against space X for a rocket launch. This goal of the project is to create a machine learning pipeline to predict if the first stage will land successfully.

Research questions

- What factors determine if the rocket will land successfully?
- What operating conditions needs to be in place to ensure a successful landing program.

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
  - Data was collected using SpaceX API and web scraping from Wikipedia.
- Perform data wrangling
  - ‘One-hot encoding’ was applied to create binary categorical features
- 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 set was split into train and test sets
  - Grid search was used to identify best sets of (hyper)parameters with predefined number of folds (CV)

Data Collection

Data was collected with the following methods:

- REST request to the SpaceX API
- Decode the response content as Json and turn it into a pandas dataframe
- Cleaned the data and check for missing values filling with averages where appropriate
- We also performed web scraping from Wikipedia using BeautifulSoup. We extracted the launch records as HTML table, and converted it to a pandas dataframe

Data Collection – SpaceX API

- We used the get request to the SpaceX API to collect data, cleaned the requested data and performed data wrangling

- The GitHub URL: https://github.com/VKGN89/IBMCourse/blob/main/1.Capstone_Data_Collection.ipynb

1. Get 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)
```

2. Use json_normalize method to convert json result to dataframe

In [12]:
```
# Use json_normalize method to convert the json result into a dataframe

# decode response content as json
static_json_df = res.json()
```

In [13]:
```
# apply json_normalize
data = pd.json_normalize(static_json_df)
```

3. We then performed data cleaning and filling in the missing values

In [30]:
```
rows = data_falcon9['PayloadMass'].values.tolist()[0]

df_rows = pd.DataFrame(rows)
df_rows = df_rows.replace(np.nan, PayloadMass)

data_falcon9['PayloadMass'][0] = df_rows.values
data_falcon9
```

Data Collection - Scraping

We performed web scrapping with BeautifulSoup where we parsed the table converting it into a pandas data frame

The GitHub URL:

https://github.com/VKGN89/IBMCourse/blob/main/2.Capstone_Web_Scraping.ipynb

1. Apply HTTP Get method to request the Falcon 9 rocket launch page

In [4]:
```
static_url = "https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922"
```

In [5]:
```
# use requests.get() method with the provided static_url
# assign the response to a object
html_data = requests.get(static_url)
html_data.status_code
```

Out[5]:
```
200
```

2. Create a BeautifulSoup object from the HTML response

In [6]:
```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
soup = BeautifulSoup(html_data.text, 'html.parser')
```

Print the page title to verify if the `BeautifulSoup` object was created properly

In [7]:
```
# Use soup.title attribute
soup.title
```

Out[7]:
```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

3. Extract all column names from the HTML table header

In [10]:
```
column_names = []

# Apply find_all() function with `th` element on first_launch_table
# Iterate each th element and apply the provided extract_column_from_header() to get a column name
# Append the Non-empty column name (`if name is not None and len(name) > 0`) into a list called column_names

element = soup.find_all('th')
for row in range(len(element)):
    try:
        name = extract_column_from_header(element[row])
        if (name is not None and len(name) > 0):
            column_names.append(name)
    except:
        pass
```

4. Create a dataframe by parsing the launch HTML tables
5. Export data to csv

Data Wrangling

- We performed EDA calculated the number of launches at each site, and the frequency of occurrences for each orbit
- We created landing outcome label from outcome column and exported the results to csv.

The GitHub URL:

https://github.com/VKGN89/IBMCourse/blob/main/3.Capstone_Data_Wrangling.ipynb

EDA with Data Visualization

- We explored the data by visualizing the relationships between flight numbers, pay load, launch sites, and orbit type as well as shown trends over time.
- The GitHub URL:

https://github.com/VKGN89/IBMCourse/blob/main/5.Capstone_EDA_Visualizations.ipynb

EDA with SQL

- Using bullet point format, summarize the SQL queries you performed
  - Names of unique launch sites
  - 5 records where launch site begins with ‘CCA’
  - Total payload mass carried by NASA boosters
  - Average payload for booster F9 v1.1

The GitHub URL:

https://github.com/VKGN89/IBMCourse/blob/main/4.Capstone_EDA_SQL.ipynb

Build an Interactive Map with Folium

- We marked launch sites and added objects such as markers, circles and lines for each site on the map.
- Using the colored marker clusters, we highlighted which launch sites have relatively high success rate.
- We have calculated the distances between a launch site and infrastructure objects in vicinity as well as a nearby city

The GitHub URL:

https://github.com/VKGN89/IBMCourse/blob/main/6.Capstone_Folium.ipynb

Build a Dashboard with Plotly Dash

- We created an interactive dashboard with Plotly Dash
- We generated pie charts depicting the launches by specific launch sites
- We created a scatter graph showing the relationship between Outcome and Payload Mass (Kg) for the different booster version

The GitHub URL:

https://github.com/VKGN89/IBMCourse/blob/main/7.Capstone_Plotly.py

Predictive Analysis (Classification)

- Data was loaded with numpy and pandas then split it into training and testing sets
- We built different machine learning models and tuned the optimal hyperparameters using GridSearchCV.
- We have employed various classification methods such as K-NN, logistic regression and a decision tree

The GitHub URL:

https://github.com/VKGN89/IBMCourse/blob/main/8.Predictive_Analytics.ipynb

Results

Exploratory data analysis results

- Success rate trend was positive over time
- Orbits ES-L1, GEO, HEO and SSO have the highest success rate

Visual exploration

- All launch sites are near the coast near the equator

Predictive analysis

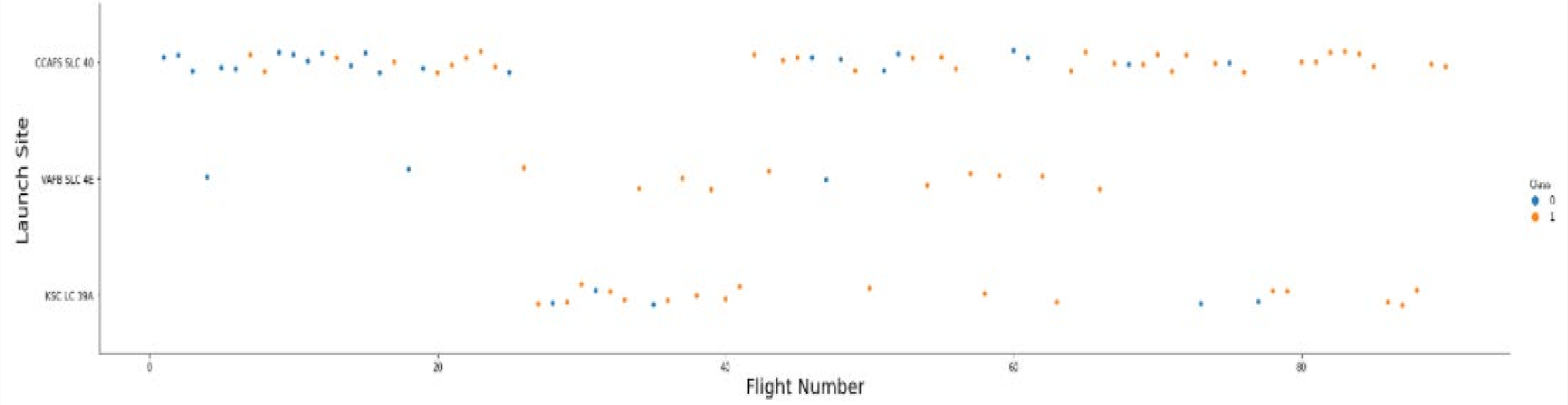
- The employed machine learning techniques performance was similar

Section 2

Insights drawn from EDA

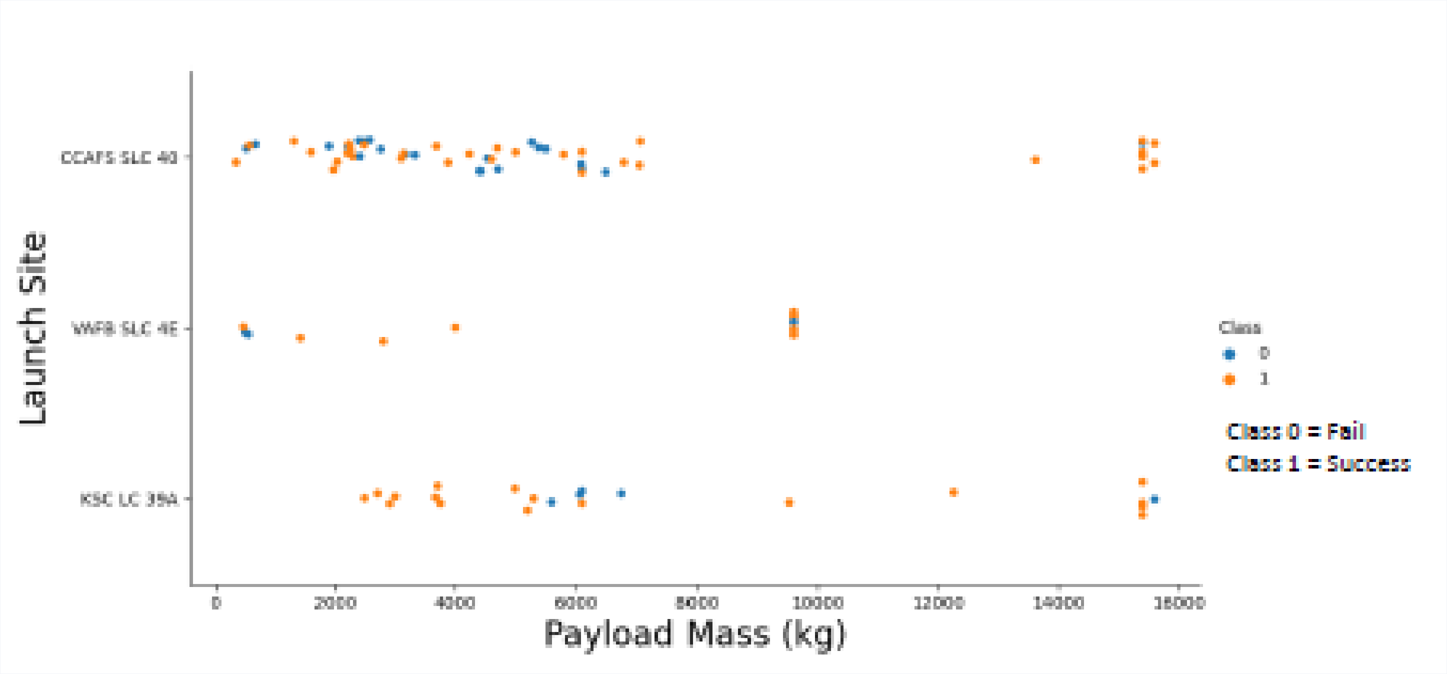
Flight Number vs. Launch Site

- There is a positive trend over time (higher success chance)



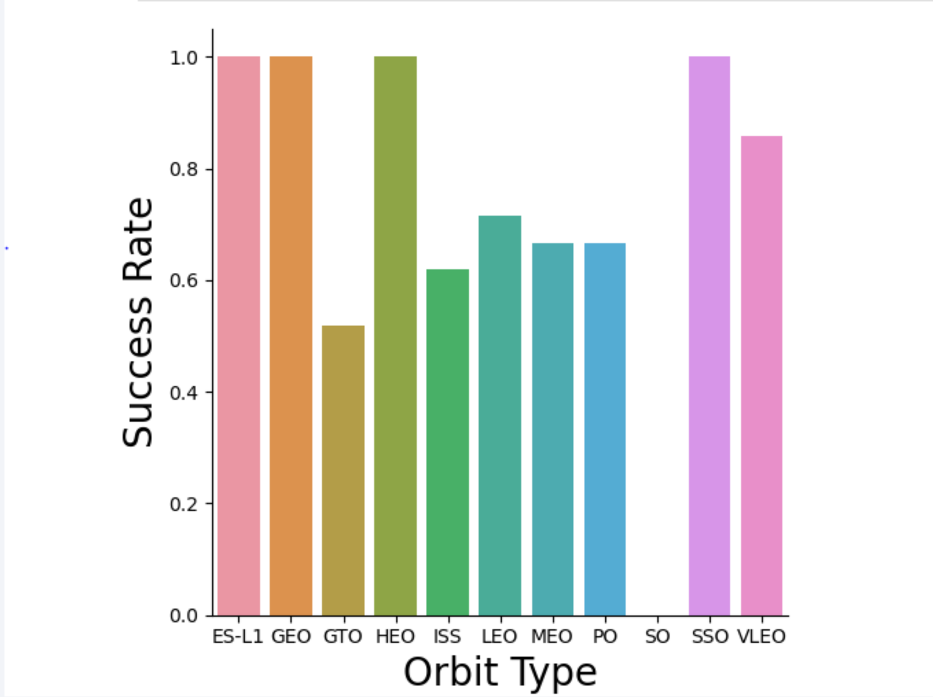
Payload vs. Launch Site

- The higher the payload the higher the success rate



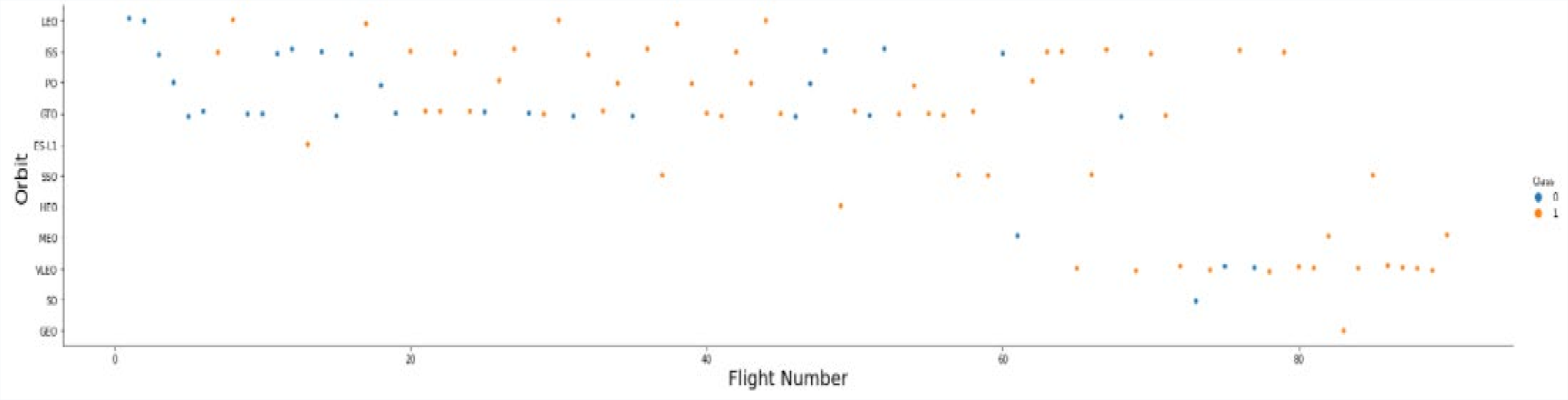
Success Rate vs. Orbit Type

- SO has a zero percent success rate
- 4 orbit types have a 100% success rate



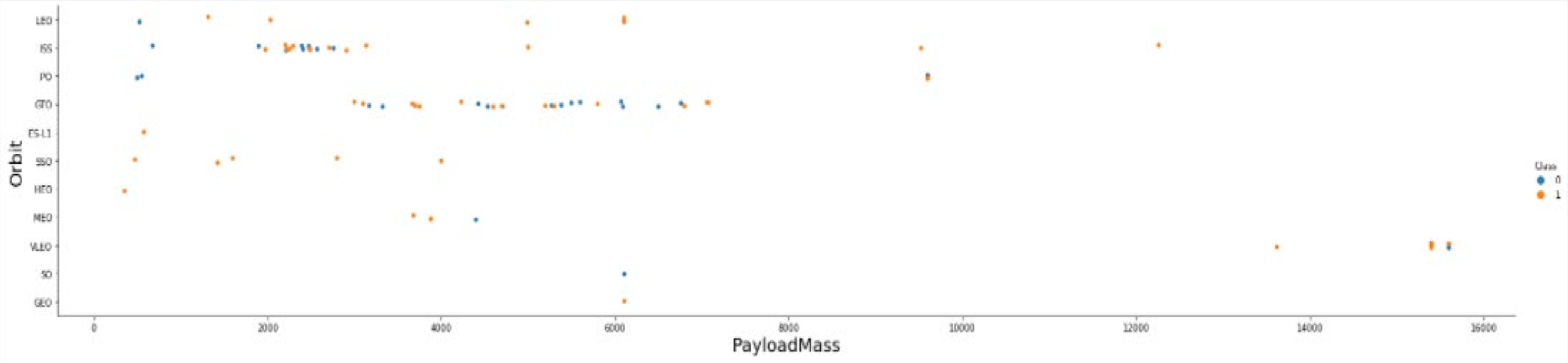
Flight Number vs. Orbit Type

- Success rate increases with the number of flights for each orbit (except GTO)



Payload vs. Orbit Type

- In general heavy payloads are more successful with LEO, PO and ISS



Launch Success Yearly Trend

- The success rates follows a positive trend

All Launch Site Names

- We used SQL DISTINCT to create a list of unique launch sites

```
SELECT DISTINCT LaunchSite
FROM SpaceX
```

	launchsite
0	KSC LC-39A
1	CCAFS LC-40
2	CCAFS SLC-40
3	VAFB SLC-4E

Launch Site Names Begin with 'KSC'

- We used SQL LIMIT to retrieve top 5 sites beginning with KSC

Task 2

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

In [9]:

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

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

Out[9]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing _Outcome
04-06-2010	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
08-12-2010	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
22-05-2012	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
08-10-2012	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
01-03-2013	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- We used SQL SUM to obtain the total payload for NASA

```
In [10]:  %sql SELECT SUM(PAYLOAD_MASS__KG_) \
              FROM SPACEXTBL \
              WHERE CUSTOMER = 'NASA (CRS)';

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

Out[10]:

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

- We used SQL AVG to obtain the average payload for F9 v1.1

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

In [11]:

```
%sql SELECT AVG(PAYLOAD_MASS__KG_) \
    FROM SPACEXTBL \
    WHERE BOOSTER_VERSION = 'F9 v1.1';
```

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

Out[11]:

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

- We used SQL MIN to obtain the first successful landing outcome on the ground pad

```
SELECT MIN(Date) AS FirstSuccessfull_landing_date
FROM SpaceX
WHERE LandingOutcome LIKE 'Success (ground pad)'
```

	firstsuccessfull_landing_date
0	2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

- We used SQL WHERE/AND conditions to obtain the list of boosters that landed successfully on a drone ship with a payload between 4000 and 6000

```
SELECT BoosterVersion
FROM SpaceX
WHERE LandingOutcome = 'Success (drone ship)'
    AND PayloadMassKG > 4000
    AND PayloadMassKG < 6000
```

	boosterversion
0	F9 FT B1022
1	F9 FT B1026
2	F9 FT B1021.2
3	F9 FT B1031.2

Total Number of Successful and Failure Mission Outcomes

We have summarized the missing outcomes using Group By and Count functions

```
%sql SELECT MISSION_OUTCOME, COUNT(*) as total_number
FROM SPACEXTBL \
GROUP BY MISSION_OUTCOME;
```

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

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

Boosters Carried Maximum Payload

- We used a subquery to find the maximum payload and used that as an argument

```
%sql SELECT BOOSTER_VERSION \
FROM SPACEXTBL \
WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL);
```

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

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

2015 Launch Records

- We used substr and date functions to perform date-specific filtering

```
%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.
```

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

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

- We used group by and order by commands to present a ranking

```
%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.
```

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

Section 3

Launch Sites Proximities Analysis

Launch sites

- All launch sites are located in the US

Launch outcomes

- Green icons show success and red failure

Distance to landmarks

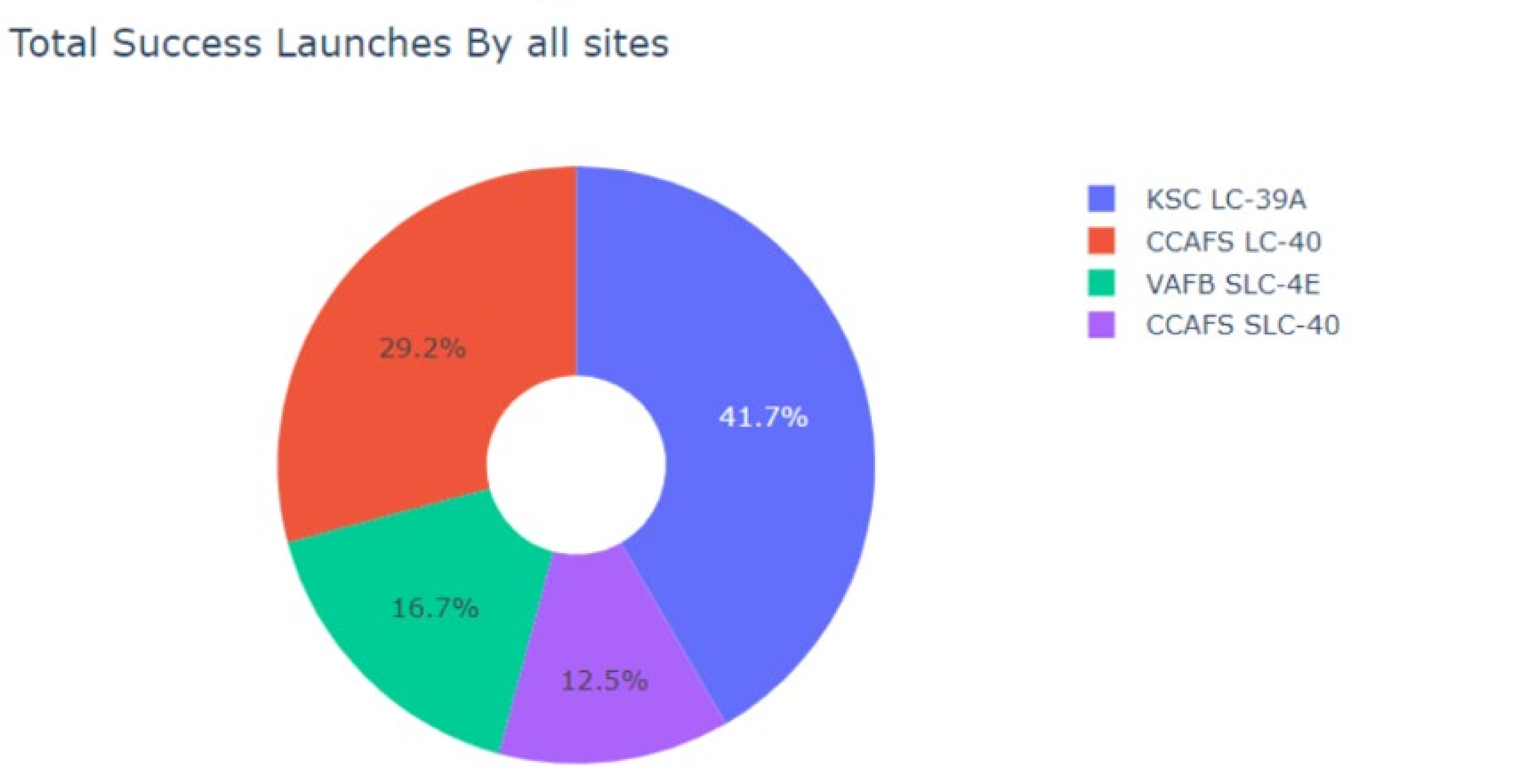
- Showing distances of **CCAFS SLC-40** to nearby landmarks

Section 4

Build a Dashboard with Plotly Dash

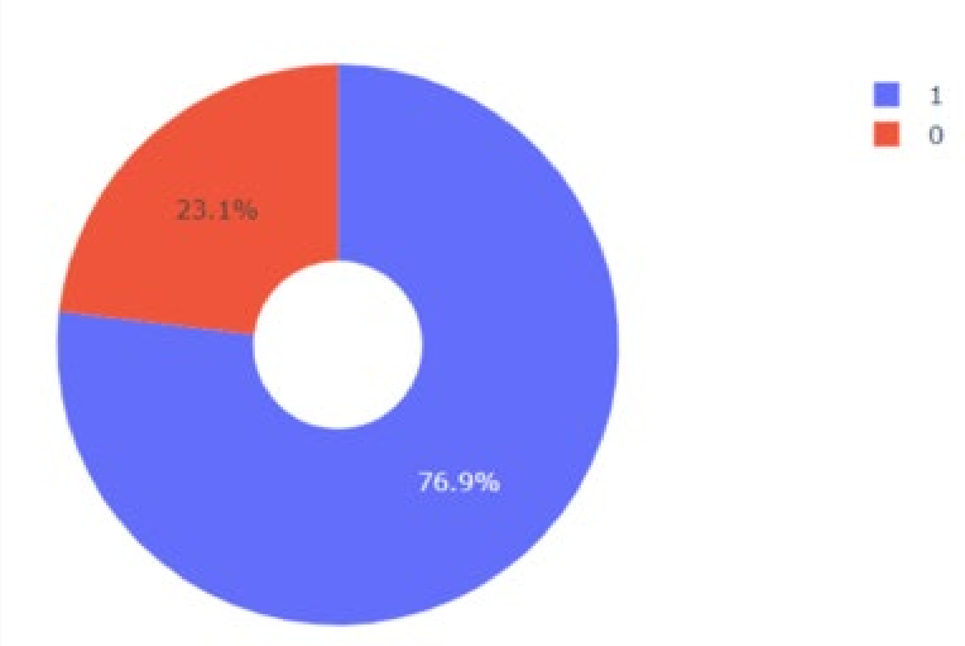
Success per launch site

Highest number of successful launches were from CCAFS



Site with highest chance of successful launch

- KSC has the highest success rate on average



Payload and success

- Between 2 and 5k is the highest success rate

Section 5

Predictive Analysis (Classification)

Classification Accuracy

- All models performed similarly

	LogReg	SVM	Tree	KNN
Jaccard_Score	0.800000	0.800000	0.800000	0.800000
F1_Score	0.888889	0.888889	0.888889	0.888889
Accuracy	0.833333	0.833333	0.833333	0.833333

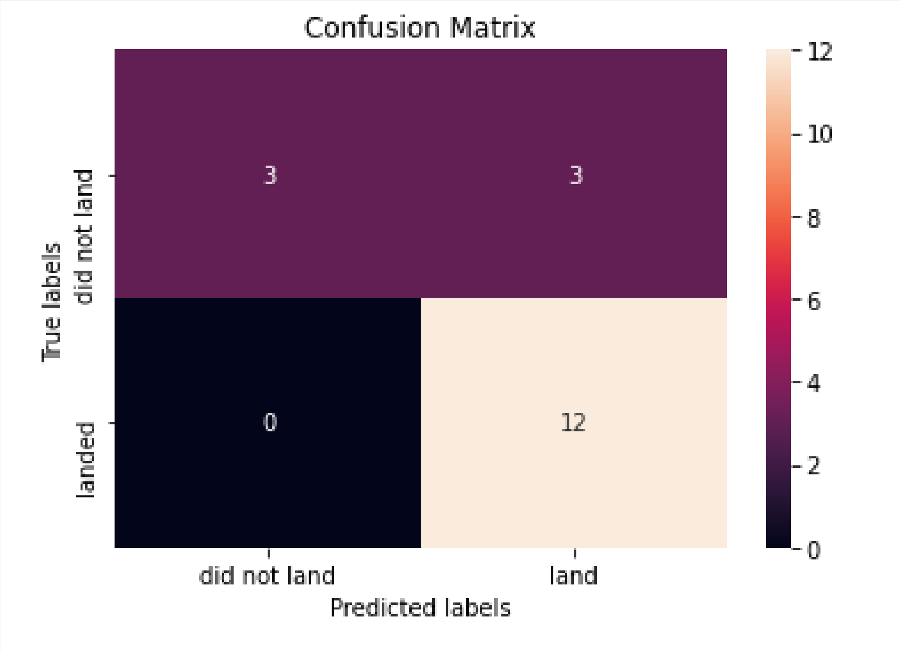
```
models = {'KNeighbors':knn_cv.best_score_,
              'DecisionTree':tree_cv.best_score_,
              'LogisticRegression':logreg_cv.best_score_,
              'SupportVector': svm_cv.best_score_}

bestalgorithm = max(models, key=models.get)
print('Best model is', bestalgorithm,'with a score of', models[bestalgorithm])
if bestalgorithm == 'DecisionTree':
    print('Best params is :', tree_cv.best_params_)
if bestalgorithm == 'KNeighbors':
    print('Best params is :', knn_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best params is :', logreg_cv.best_params_)
if bestalgorithm == 'SupportVector':
    print('Best params is :', svm_cv.best_params_)
```

```
Best model is DecisionTree with a score of 0.9017857142857142
Best params is : {'criterion': 'gini', 'max_depth': 16, 'max_features': 'auto', 'min_samples_leaf': 4, 'min_samples_split': 10, 'splitter': 'random'}
```

Confusion Matrix

- False positive remained a persistent problem in each model



Conclusions

- There is a positive trend in success over time
- All launch site are near the coast around the equator
- KSC had the most successful launches of all
- The higher the payload, the higher the success rate
- Models performed similarly to predict the landing outcome

Appendix

- The GitLab repo can be found here:

https://github.com/VKGN89/IBMCourse/tree/main

Thank you!