Winning Space Race with Data Science

Arjun Talapatra
02/06/26

Outline

- Executive Summary
- Introduction
- Methodology (Section 1)
- Results (Sections 2 and 3)
- Conclusion
- Appendix

Executive Summary

Methodology Summary

- Collected SpaceX Falcon 9 launch data using SpaceX REST API and web scraping from Wikipedia
- Cleaned, merged, and engineered features through data wrangling
- Performed exploratory data analysis (EDA) using SQL and static visualizations
- Built and evaluated classification models (Logistic Regression, SVM, Decision Tree, KNN) to predict first-stage landing success

Results Summary

- Launch success rate improved significantly over time
- Certain launch sites, orbit types, and payload mass ranges showed higher landing success rates
- Classification models were able to predict landing success with strong accuracy, with SVM / Logistic Regression performing best on test data

Introduction

Project Background and Context

- SpaceX has significantly reduced launch costs by reusing Falcon 9 first-stage boosters
- Successful booster landings are a key factor enabling cost efficiency and competitive pricing
- Predicting first-stage landing success can help estimate launch costs and inform competitive bidding strategies for commercial space missions

Problem Statement

- What factors influence whether a Falcon 9 first stage lands successfully?
- How do launch site, payload mass, flight numbers, and orbit type affect landing outcomes?
- Can historical launch data be used to predict landing success using machine learning models?

Section 1

Methodology

Methodology

Data collection methodology:

- Collected launch data using the SpaceX REST API
- Supplemented data through web scraping of Falcon 9 launch records from Wikipedia

Perform data wrangling

- Cleaned, merged, and transformed datasets
- Handled missing values and engineered features
- Created a binary landing success label for supervised learning

Exploratory Data Analysis (EDA)

- Performed statistical and relational analysis using SQL
- Visualized trends using static charts (payload, launch site, booster version)

Predictive Modeling

- Built and evaluated classification models (Logistic Regression, SVM, Decision Tree, KNN)
- Tuned hyperparameters using cross-validation
- Compared model performance on test data

Data Collection – SpaceX API

Collected Falcon 9 launch data using the SpaceX REST API

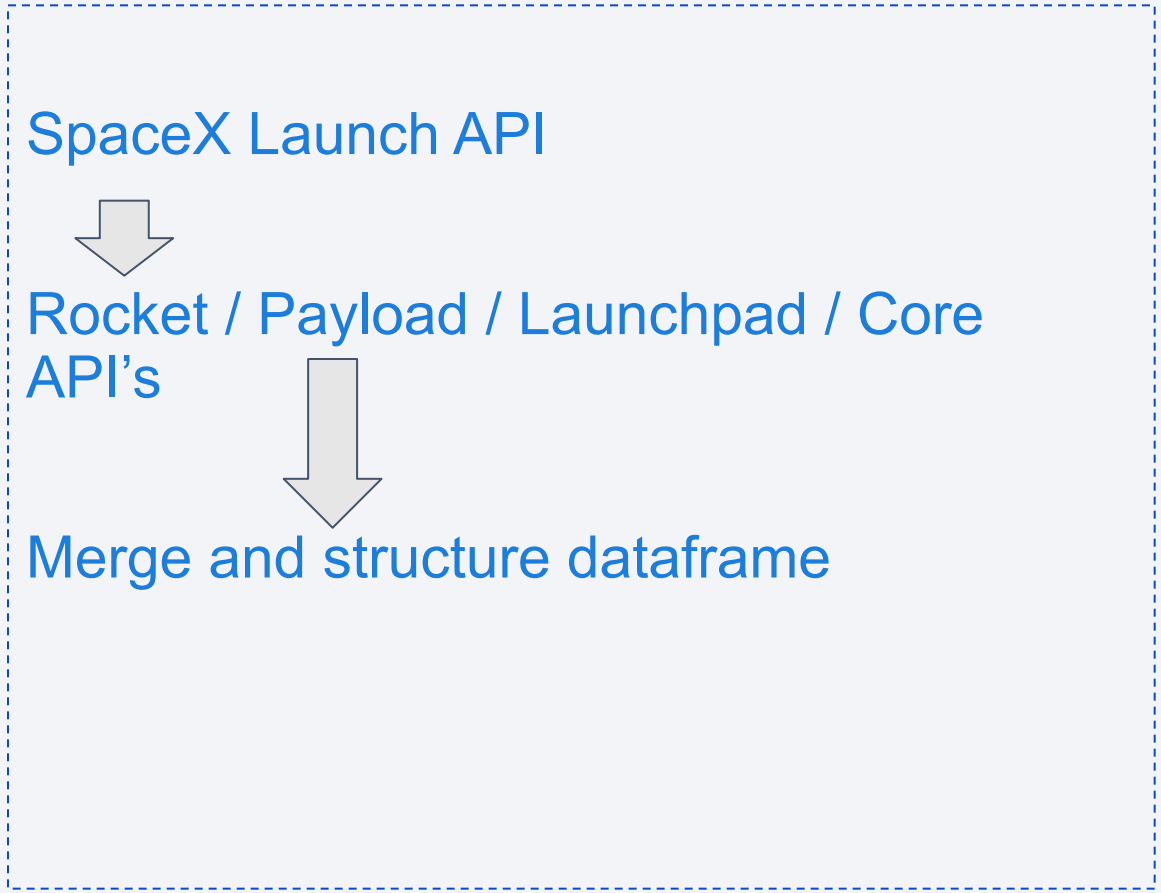
Retrieved historical launch records including:

- Rocket and booster information
- Launch site coordinates (latitude & longitude)
- Payload mass and orbit type
- Core reuse, landing outcome, and mission details

Performed multiple API calls to enrich launch data

Stored and structured the data into a unified tabular dataset for analysis

https://github.com/atalapatra72-bit/SpaceX-Data-Analysis/blob/main/Part%201%20-%20Data%20Collection%20API.pdf



Data Collection - Scraping

Extracted Falcon 9 and Falcon Heavy launch records from Wikipedia

Used BeautifulSoup to parse HTML tables containing historical launch data

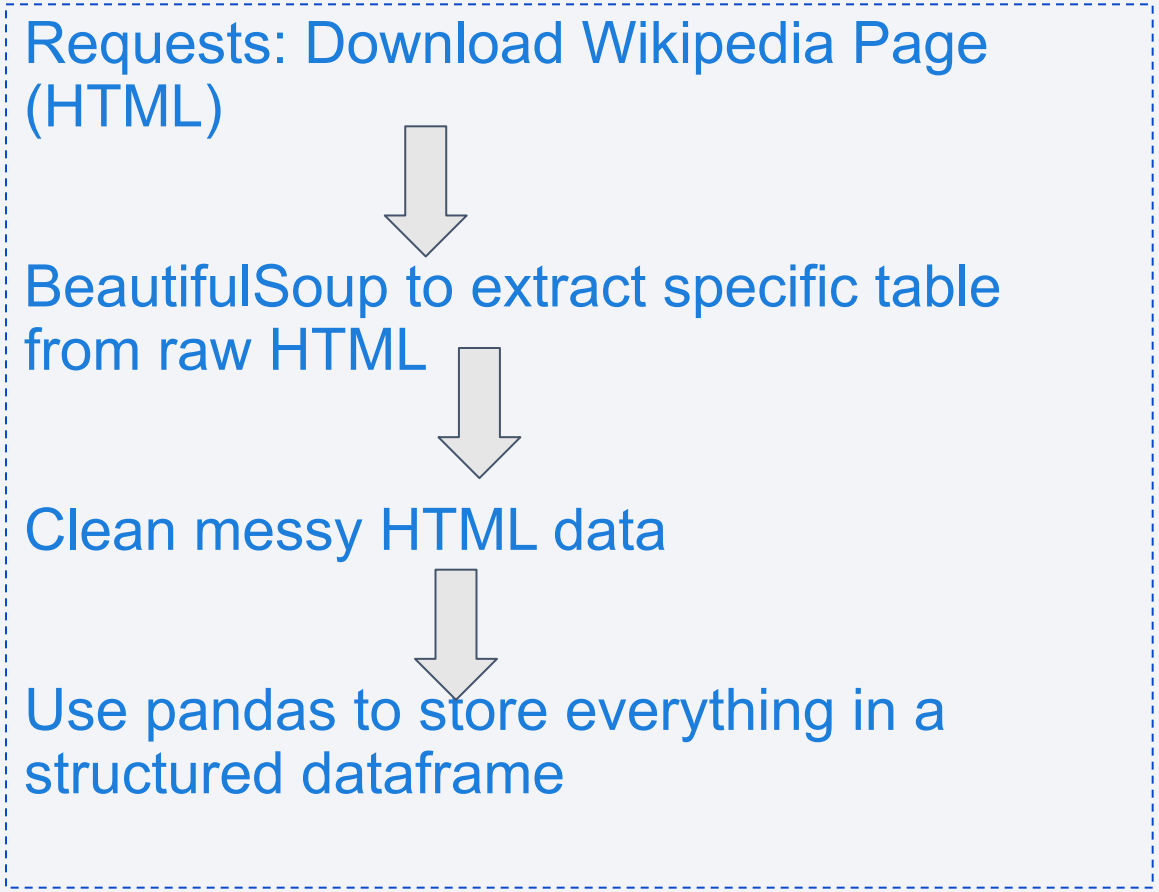
Cleaned and standardized scraped data fields:

- Date and time
- Booster version
- Launch site
- Payload mass and orbit
- Landing outcome

Converted scraped data into a structured Pandas DataFrame

Combined scraped data with API-collected data for downstream analysis

https://github.com/atalapatra72-bit/SpaceX-Data-Analysis/blob/main/Part%202%20-%20Webscraping.pdf



Data Wrangling

- Started with the raw SpaceX launch dataset
- Identified and handled missing values for “PayloadMass” column
- Create a new column “Class” to the dataframe
  - This was generated from the column “landing_outcomes”
  - If first stage landed successfully: Class=1
  - If first stage did NOT land successfully: Class=0

Raw Dataset ⟹ Handle Missing Values ⟹ Created a new column “Class”

https://github.com/atalapatra72-bit/SpaceX-Data-Analysis/blob/main/Part%203%20-%20Data%20Wrangling.pdf

EDA with Data Visualization

- Scatter point charts were used to visualize the following
  - Flight number vs launch site across “class=0” and “class=1”
  - Payload mass vs launch site across “class=0” and “class=1”
  - Flight number vs orbit across “class=0” and “class=1”
  - Payload mass vs orbit across “class=0” and “class=1”
- A bar chart was used to visualize the mean success rate across the different orbit types
- A line chart was used to visualize the mean success rate over time

https://github.com/atalapatra72-bit/SpaceX-Data-Analysis/blob/main/Part%205%20-%20EDA%20with%20Visualization.pdf

EDA with SQL

- Queried the dataset to identify distinct launch sites.
- Identified the date when the first successful landing outcome in ground pad was achieved.
- Determined the count of each landing outcome over a specific period of time.
- Used SQL aggregation, filtering, and grouping to efficiently explore various other aspects of the large dataset.

https://github.com/atalapatra72-bit/SpaceX-Data-Analysis/blob/main/Part%204%20-%20EDA%20with%20SQL.pdf

Predictive Analysis (Classification)

- Separated the target variable (the “Class” column) from the rest of the features, separating them into “X” and “Y”
- Standardized the columns in “X”
- Train/test split
- Used training data to train a logistic regression model (using GridSearchCV to find best hyperparameter values). Outputted accuracy on train data.
- Tested logistic regression model on test data and calculated accuracy + made Confusion Matrix
- Repeated above 2 steps for Support Vector Machine (SVM) model, decision tree model, and K-Nearest Neighbors (KNN) model
- Evaluated the best model

Feature Selection and Standardization

Train/test split

Model training (SVM, decision tree, KNN)

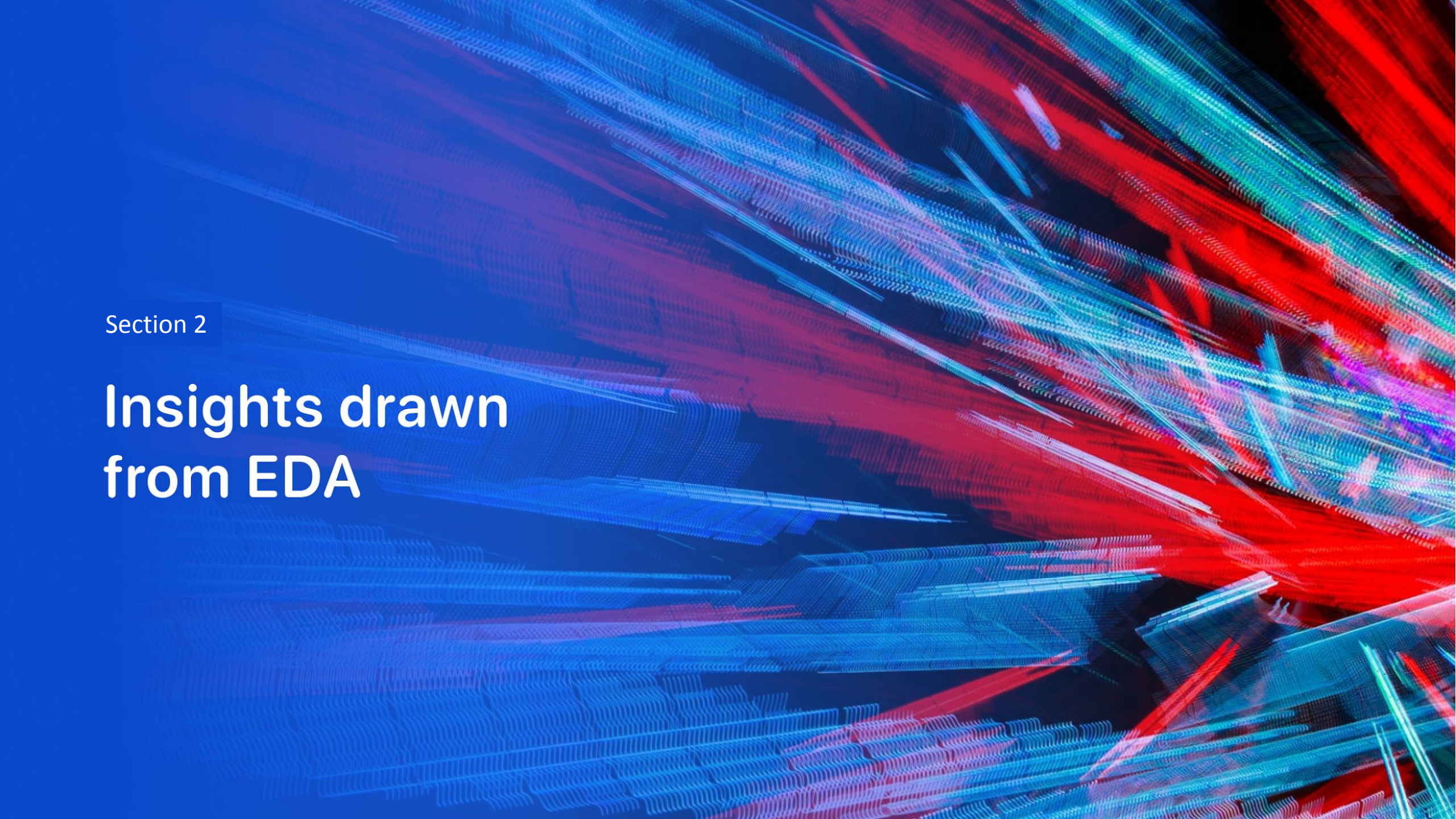
Model testing (SVM, decision tree, KNN)

Evaluate best model

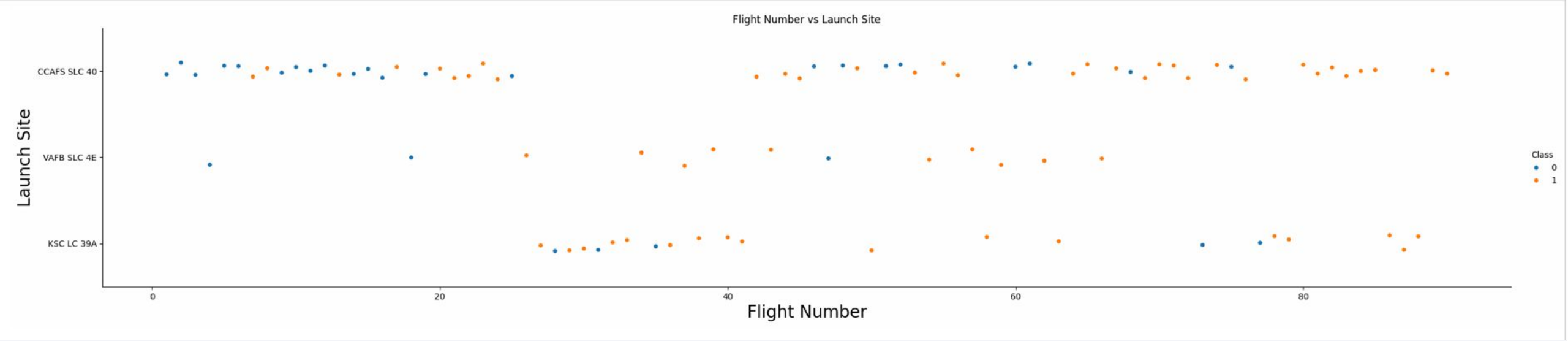
https://github.com/atalapatra72-bit/SpaceX-Data-Analysis/blob/main/Part%206%20-%20Machine%20Learning%20Prediction.pdf

Section 2

Insights drawn from EDA



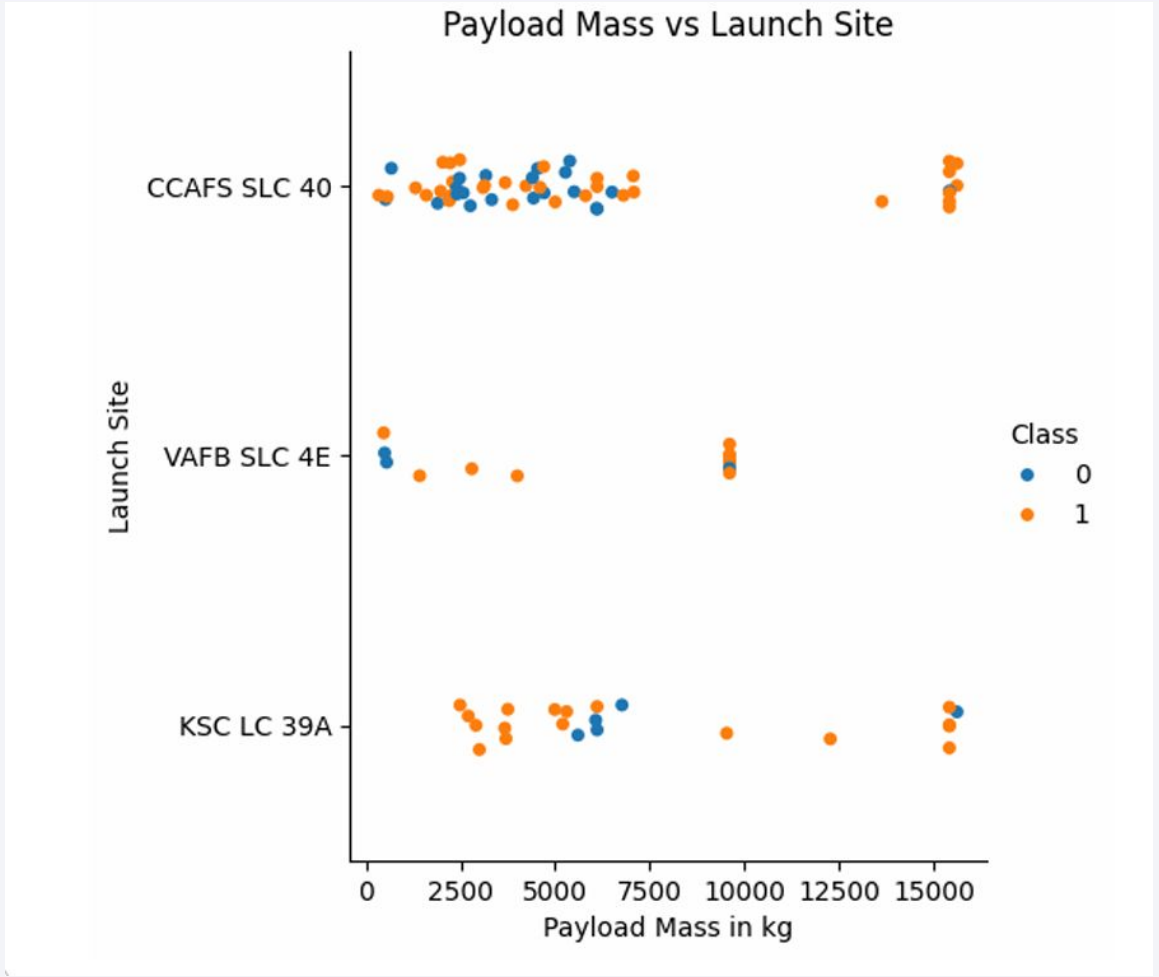
Flight Number vs. Launch Site



- Early flights (represented by low flight numbers) are dominated by failures
- As flight number increases, successes increase - this is the case for all launch sites
- At the beginning, most flights had their launch site at “CCAFS SLC 40” - KSC LC 39A came later
- VAFB SLC 4E has fewer total launches

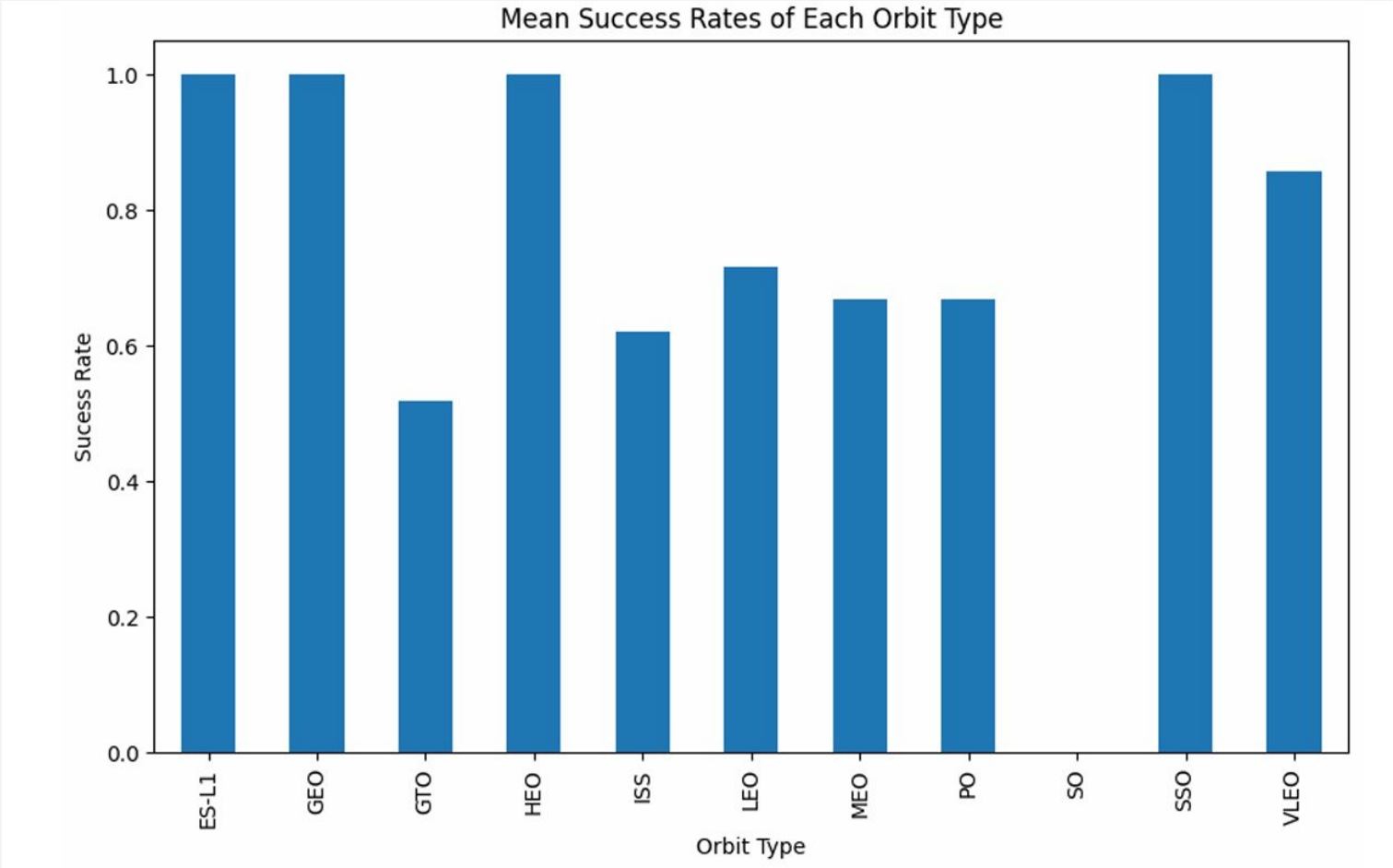
Payload vs. Launch Site

- At lower payload masses (<8000 kg), successes and failures are mixed
- At higher payload masses (>8000 kg), successes are much more common than failures across all launch sites
- Launch site CCAFS SLC 40 has the most number of launches with moderate success rate, especially at lower payload masses
- Launch sites VAFB SLC 4E and KSC LC 39A have fewer launches with higher success rates



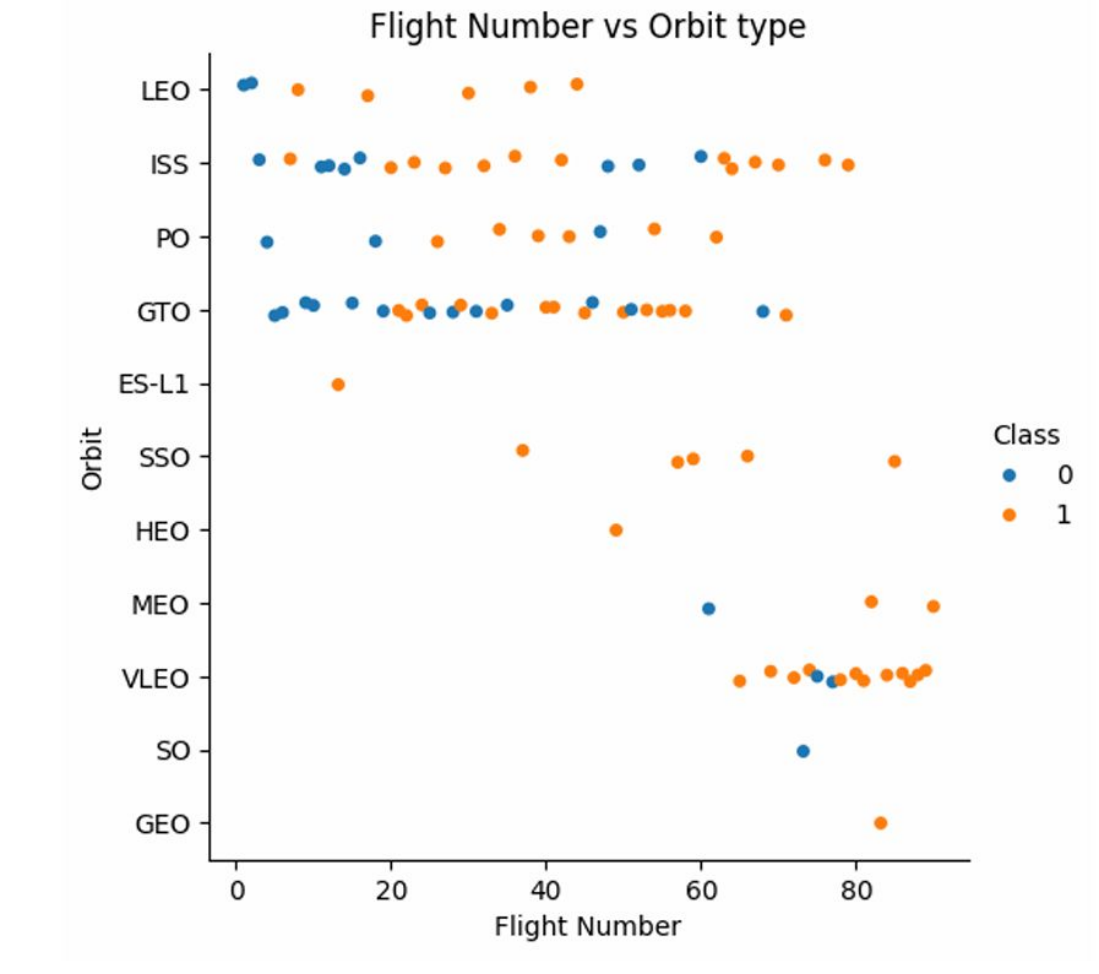
Success Rate vs. Orbit Type

- Orbits ES-L1, GEO, HEO, SSO, and VLEO have high success rates (>80%)
- GTO has noticeably lower success rate
- SO has no successes with a likely small sample size



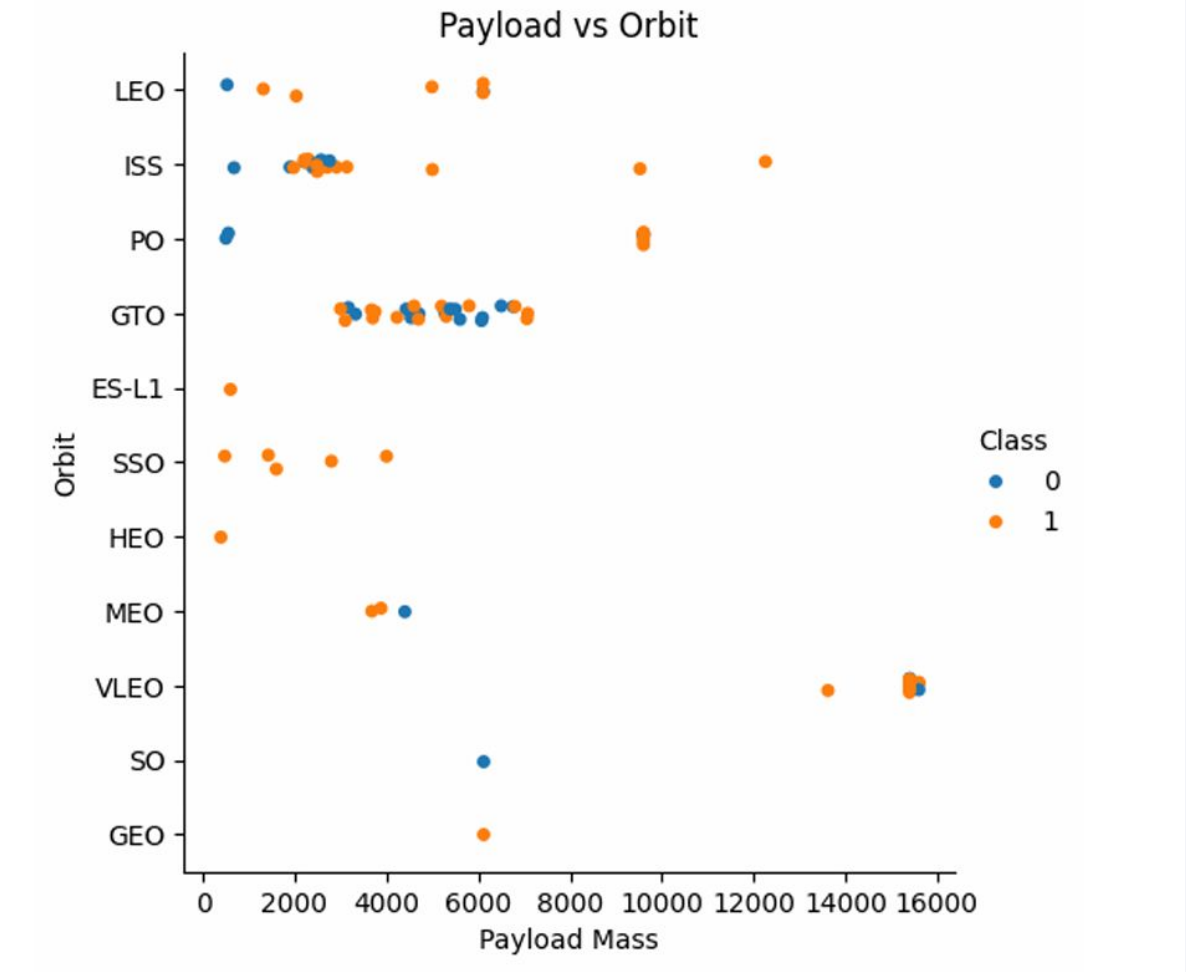
Flight Number vs. Orbit Type

- Across different orbit types, early flights showed more failures than later flights
- Certain orbit types (such as MEO, VLEO, GEO) only show up in later flights



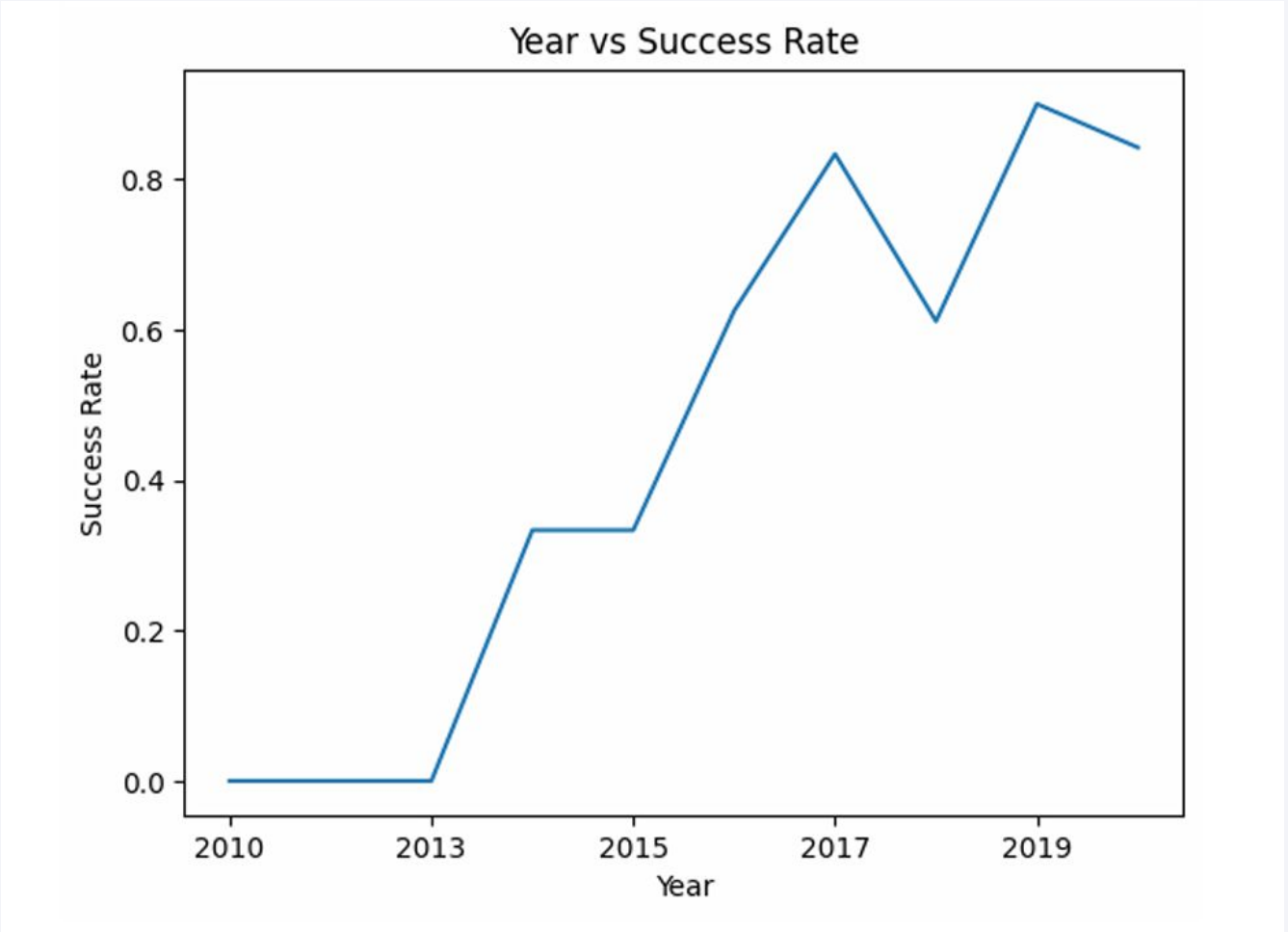
Payload vs. Orbit Type

- Different orbits operate in different payload mass ranges (ex: GTO clustered around payload mass of 2000 to 10000 kg)
- Failures tend to be more frequent at lower payload masses



Launch Success Yearly Trend

- From 2010-2013, very low success rate
- Steady improvement from 2013 to 2017
- Success rate remains over 50% since then
- Can confidently say success rate improved significantly over time



All Launch Site Names

- Used the following sql code to determine all the unique launch names

```
In [11]: %sql SELECT DISTINCT(Launch_Site) FROM SPACEXTBL
 * sqlite:///my_data1.db
Done.
Out[11]:
```

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

- There are 4 unique launch sites

Launch Site Names Begin with 'CCA'

```
%sql SELECT * from SPACEXTBL WHERE Launch_Site like 'CCA%' LIMIT 5
```

Out[12]:

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-06-04	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-12-08	15:43:00	F9 v1.0 B0004	CCAFS LC-40	Dragon demo flight C1, two CubeSats, barrel of Brouere cheese	0	LEO (ISS)	NASA (COTS) NRO	Success	Failure (parachute)
2012-05-22	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-03-01	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

- Used SQL to displays 5 records with a Launch Site name that is either CCAF LC-40 or CCAF SLC-40

Total Payload Mass

```
In [13]: %sql SELECT SUM(PAYLOAD_MASS__KG_) from SPACEXTBL WHERE Customer='NASA (CRS)'

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

Out[13]:

SUM(PAYLOAD_MASS__KG_)
45596

- The total payload mass carried by boosters launched by NASA is 45596

Average Payload Mass by F9 v1.1

```
In [14]: %sql SELECT AVG(PAYLOAD_MASS__KG_) from SPACEXTBL WHERE BOOSTER_VERSION="F9 v1.1"
 * sqlite:///my_data1.db
Done.
Out[14]: AVG(PAYLOAD_MASS__KG_)
         2928.4
```

- The average payload mass carried by booster version F9 v1.1 is 2928.4

First Successful Ground Landing Date

```
In [15]: %sql SELECT min(Date) FROM SPACEXTBL WHERE Landing_Outcome='Success (ground pad)'
 * sqlite:///my_data1.db
Done.
Out[15]: min(Date)
         2015-12-22
```

- On December 22 2015, the first successful landing outcome in ground pad was achieved

Successful Drone Ship Landing with Payload between 4000 and 6000

```
In [16]: %sql SELECT Booster_Version FROM SPACEXTBL WHERE Landing_Outcome='Success (drone ship)' AND (PAYLOAD_MASS__KG_>4000 AND PAYLOAD_MASS__KG_<6000)
 * sqlite:///my_data1.db
Done.
```

Out[16]:

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

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

Total Number of Successful and Failure Mission Outcomes

```
In [17]: %sql SELECT Mission_Outcome, COUNT(*) AS "# Missions" FROM SPACEXTBL GROUP BY Mission_Outcome
 * sqlite:///my_data1.db
Done.
```

Out[17]:

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

- The odds of a failure in flight is very low (approximately 1 percent based on historical data)

Boosters Carried Maximum Payload

```
In [18]: %sql SELECT Booster_Version FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEXTBL)
 * sqlite:///my_data1.db
Done.
```

Out[18]:

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

- Listed are all the booster_versions that have carried the maximum payload mass

2015 Launch Records

In [24]:
```
%sql SELECT substr(Date, 6,2) AS Month,Landing_Outcome,Booster_version,Launch_Site FROM SPACEXTBL WHERE substr(Date,1,4)='2015' AND Landing_Outcome='Failure (drone ship)'
```

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

Out[24]:

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

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

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

```
[32]: %sql SELECT Landing_Outcome, COUNT(Landing_Outcome) FROM SPACEXTBL WHERE (Date>'2010-06-04' AND Date<'2017-03-20') GROUP BY Landing_Outcome ORDER BY COUNT(Landing_Outcome) DESC
 * sqlite:///my_data1.db
Done.
```

[32]:

Landing_Outcome	COUNT(Landing_Outcome)
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Precluded (drone ship)	1
Failure (parachute)	1

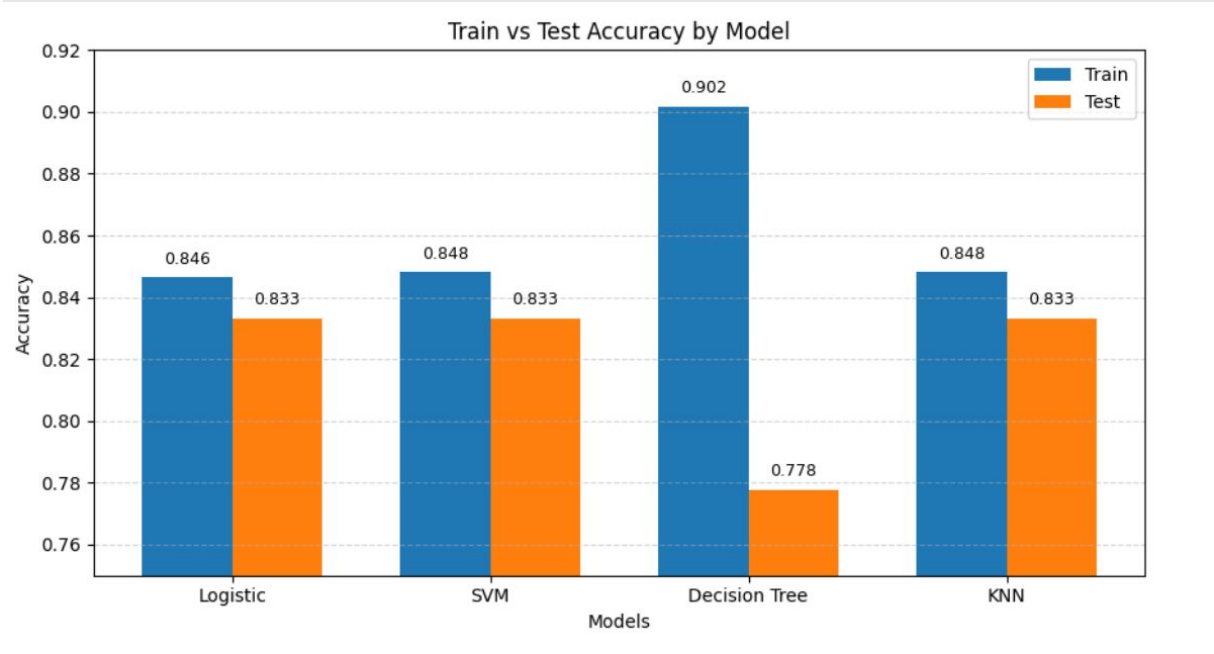
- I ranked the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

Section 3

Predictive Analysis (Classification)

Classification Accuracy

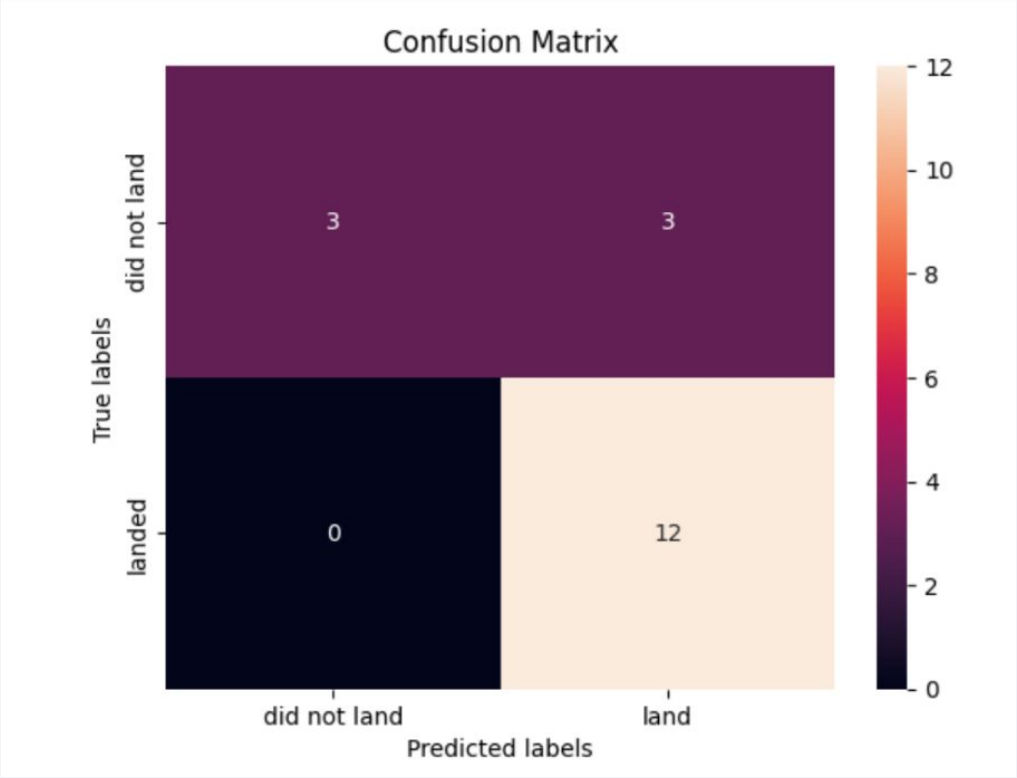
- Logistic Regression Model
  - Small gap between train and test - good. Strong accuracy and showing no signs of overfitting or underfitting
- Support Vector Machine (SVM)
  - Similar to Logistic Regression model - strong and stable
- Decision Tree
  - Large gap between train and test indicates overfitting
- K-Nearest Neighbors (KNN)
  - Similar to Logistic and SVM. Solid but not superior to those simpler models.



Logistic Regression and Support Vector Machine achieved the highest and most stable classification accuracy on the test dataset. The Decision Tree model showed signs of overfitting, while KNN performed comparably but did not outperform simpler models.

Confusion Matrix

- This is the confusion matrix for both the Logistic Regression model and the Support Vector Machine model
-
-



Conclusions

- Landing success of Falcon 9 first-stage boosters has improved significantly over time, reflecting increased operational maturity and technological advancement
- Launch site, orbit type, payload mass, and flight number were found to be associated with landing outcomes through exploratory data analysis (EDA)
- Supervised classification models were able to predict landing success with strong accuracy, with SVM / Logistic Regression performing best on the test dataset
- The results demonstrate that historical launch data can be effectively used to support predictive modeling and decision-making for reusable rocket missions

Appendix

Python code to output Train vs Test Accuracy by Model Chart:

```
[6]: import numpy as np
import matplotlib.pyplot as plt

models = ['Logistic', 'SVM', 'Decision Tree', 'KNN']

train_accuracy = [0.8464, 0.8482, 0.9018, 0.8482]
test_accuracy  = [0.8333, 0.8333, 0.7778, 0.8333]

x = np.arange(len(models))
width = 0.35

plt.figure(figsize=(9,5))

bars1 = plt.bar(x - width/2, train_accuracy, width, label='Train')
bars2 = plt.bar(x + width/2, test_accuracy, width, label='Test')

plt.xlabel('Models')
plt.ylabel('Accuracy')
plt.title('Train vs Test Accuracy by Model')

plt.xticks(x, models)

# 🔥 Zoom in on meaningful range
plt.ylim(0.75, 0.92)

# 🔥 Add value labels
for bars in [bars1, bars2]:
    for bar in bars:
        height = bar.get_height()
        plt.text(
            bar.get_x() + bar.get_width()/2,
            height + 0.003,
            f'{height:.3f}',
            ha='center',
            va='bottom',
            fontsize=9
        )

plt.legend()
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```

Thank you!