IBM Developer SKILLS NETWORK

Python SQL Machine Learning

SPACEX LANDING PREDICTION

Applied Data Science Capstone

Kristina Cinova

GitHub link

Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion

Executive Summary

Project objective

Predict the success of SpaceX Falcon 9 first-stage landings to help competitors bid against SpaceX for launches.

Key methodologies

- Data collection via APIs and web scraping.
- Data wrangling and cleaning.
- Exploratory data analysis (EDA) using visualizations and SQL.
- Interactive visual analytics with Folium and Plotly Dash.
- Predictive analysis with various classification models.

Results

Logistic Regression, SVM, and KNN models achieved similar high accuracy in predictions.

KRISTINA CINOVA

Introduction

Background

SpaceX's competitive advantage in reducing launch costs through the reuse of rocket stages.

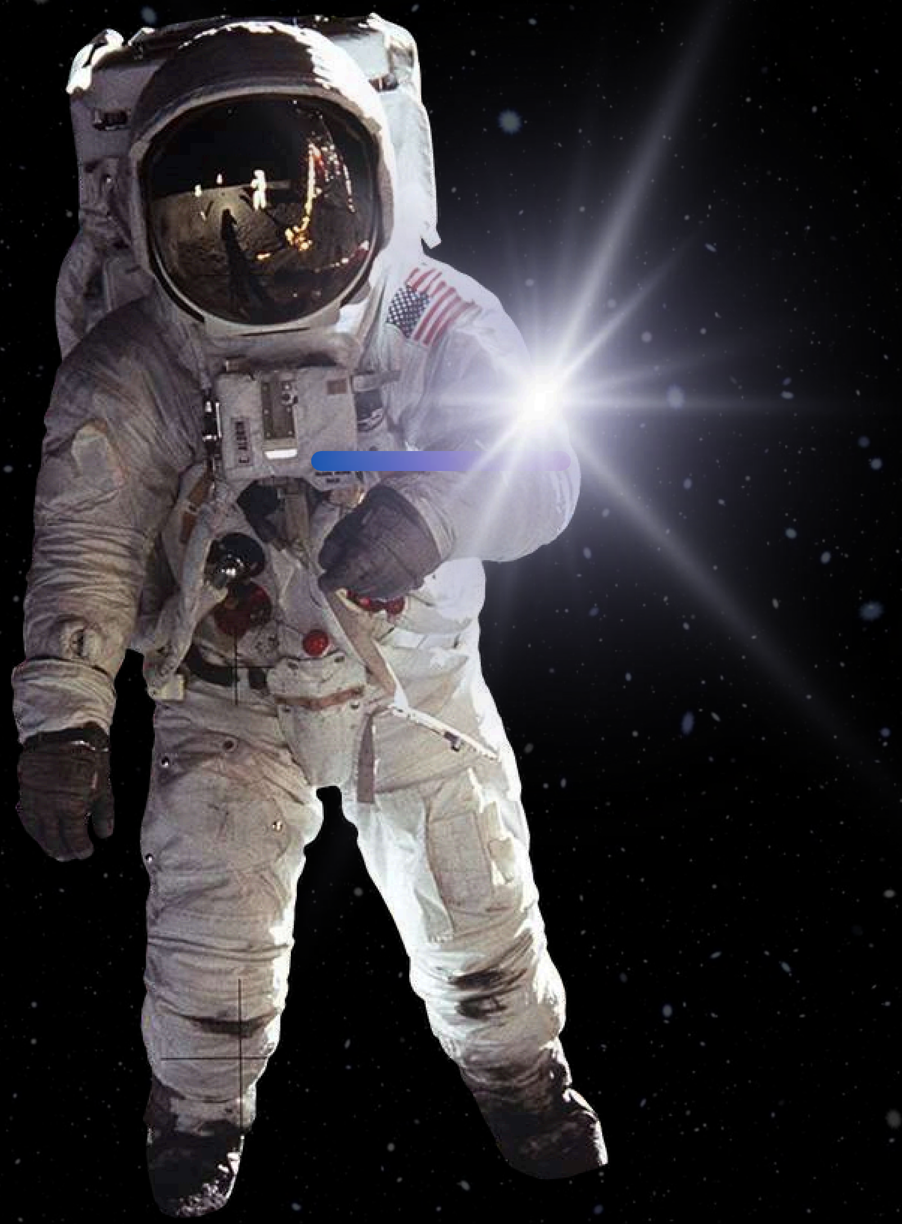
Problem statement

Determining the likelihood of a successful landing of the Falcon 9 first stage.

KRISTINA CINOVA

METHODOLOGY

Section 1



KRISTINA CINOVA

Methodology - Data Collection

Data Sources

- APIs for SpaceX data, web scraping for additional details.

Techniques

- RESTful API calls, BeautifulSoup for scraping

GitHub Links

- API Data Collection, Web Scraping

```python
BoosterVersion = []

def getBoosterVersion(data):
    global BoosterVersion
    for rocket_id in data['rocket']:
        if rocket_id:
            response = requests.get(f"https://api.spacexdata.com/v4/rockets/{rocket_id}")
            if response.status_code == 200:
                rocket_info = response.json()
                BoosterVersion.append(rocket_info['name'])
            else:
                BoosterVersion.append(None)
        else:
            BoosterVersion.append(None)
```

```python
LaunchSite = []
Longitude = []
Latitude = []

def getLaunchSite(data):
    global LaunchSite, Longitude, Latitude
    for site_id in data['launchpad']:
        if site_id:
            response = requests.get(f"https://api.spacexdata.com/v4/launchpads/{site_id}")
            if response.status_code == 200:
                site_info = response.json()
                LaunchSite.append(site_info.get('name', None))
                Longitude.append(site_info.get('longitude', None))
                Latitude.append(site_info.get('latitude', None))
            else:
                LaunchSite.append(None)
                Longitude.append(None)
                Latitude.append(None)
```

KRISTINA CINOVA

Methodology - Webscraping

Step 1

IMPORTING LIBRARIES AND SETTING UP THE REQUEST

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

Request the Falcon9 Launch Wiki page from its URL

Creating a BeautifulSoup object To parse the HTML content, I create a BeautifulSoup object. This step is crucial as it allows us to navigate and search through the HTML structure easily.

```
response = requests.get(static_url)

soup = BeautifulSoup(response.content, 'html.parser')

print(soup.title)
```

```
<title>List of Falcon 9 and Falcon Heavy launches - Wikipedia</title>
```

KRISTINA CINOVA

Methodology - Webscraping

Step 2

EXTRACTING COLUMN NAMES

```
html_tables = soup.find_all('table', class_='wikitable plainrowheaders collapsible')
```

Starting from the third table is our target table contains the actual launch records.

```
first_launch_table = html_tables[2]
print(first_launch_table)
```

```
<table class="wikitable plainrowheaders collapsible" style="width: 100%;">
<tbody><tr>
<th scope="col">Flight No.
</th>
<th scope="col">Date and<br/>time (<a href="/wiki/Coordinated_Universal_Time" title="Coordinated Universal Time">UTC</a>)
</th>
<th scope="col"><a href="/wiki/List_of_Falcon_9_first-stage_boosters" title="List of Falcon 9 first-stage boosters">Version,<br/>Booster</a><
sup class="reference" id="cite_ref-booster_11-2"><a href="#cite_note-booster-11">[b]</a></sup>
</th>
<th scope="col">Launch site
</th>
<th scope="col">Payload<sup class="reference" id="cite_ref-Dragon_12-2"><a href="#cite_note-Dragon-12">[c]</a></sup>
</th>
<th scope="col">Payload mass
</th>
<th scope="col">Orbit
</th>
<th scope="col">Customer
```

Next, we just need to iterate through the `<th>` elements and apply the provided `extract_column_from_header()` to extract column name one by one

```
column_names = []

for th in first_launch_table.find_all('th'):
    column_names.append(th.text.strip())

print(column_names)
```

```
['Flight No.', 'Date andtime (UTC)', 'Version,Booster[b]', 'Launch site', 'Payload[c]', 'Payload mass', 'Orbit', 'Customer', 'Launchoutcome',
'Boosterlanding', '14', '15', '16', '17', '18', '19', '20']
```

Checking the extracted column names.

```
print(column_names)
```

```
['Flight No.', 'Date andtime (UTC)', 'Version,Booster[b]', 'Launch site', 'Payload[c]', 'Payload mass', 'Orbit', 'Customer', 'Launchoutcome',
'Boosterlanding', '14', '15', '16', '17', '18', '19', '20']
```

KRISTINA CINOVA

Methodology - Webscraping

Step 3

EXTRACTING DATA ROWS

```python
launch_dict = {
    'Flight No.': [],
    'Date': [],
    'Time': [],
    'Version Booster': [],
    'Launch site': [],
    'Payload': [],
    'Payload mass': [],
    'Orbit': [],
    'Customer': [],
    'Launch outcome': [],
    'Booster landing': []
}
```

```python
for table_number, table in enumerate(soup.find_all('table', class_="wikitable plainrowheaders collapsible")):
    for rows in table.find_all("tr"):
        # Check if the row is a header row with a flight number
        if rows.th:
            if rows.th.string:
                flight_number = rows.th.string.strip()
                flag = flight_number.isdigit()
        else:
            flag = False

        row = rows.find_all('td')

        # If it's a valid flight number row, parse the data
        if flag:
            # Flight Number
            launch_dict['Flight No.'].append(flight_number)

            datatimelist = [i.strip() for i in row[0].text.split()]
            date = datatimelist[0].strip(',')
            time = datatimelist[1] if len(datatimelist) > 1 else None
            launch_dict['Date'].append(date)
            launch_dict['Time'].append(time)

            # Version Booster
            bv = row[1].text.strip()
            launch_dict['Version Booster'].append(bv)

            # Launch Site
            launch_site = row[2].text.strip()
            launch_dict['Launch site'].append(launch_site)

            # Payload
            payload = row[3].text.strip()
            launch_dict['Payload'].append(payload)

            # Payload Mass
            payload_mass = row[4].text.strip().split()[0] if row[4].text.strip() else None
            launch_dict['Payload mass'].append(payload_mass)

            # Orbit
            orbit = row[5].text.strip()
            launch_dict['Orbit'].append(orbit)

            # Customer
            customer = row[6].text.strip() if row[6].text.strip() else None
            launch_dict['Customer'].append(customer)

            # Launch outcome
            launch_outcome = row[7].text.strip() if row[7].text.strip() else None
            launch_dict['Launch outcome'].append(launch_outcome)

            # Booster landing
            booster_landing = row[8].text.strip() if len(row) > 8 else None
            launch_dict['Booster landing'].append(booster_landing)

# Converting the dictionary to a Pandas DataFrame
df_launches = pd.DataFrame(launch_dict)
```

KRISTINA CINOVA

Methodology - Data Wrengling

Process

- Data cleaning, handling missing values, feature engineering.

Tools

- Pandas, SQL for database operations.

GitHub Link

- Data Wrangling

```
df.isnull().sum()/len(df)*100
```

```
FlightNumber       0.000000
Date               0.000000
BoosterVersion     0.000000
PayloadMass        0.000000
Orbit              0.000000
LaunchSite         0.000000
Outcome            0.000000
Flights            0.000000
GridFins           0.000000
Reused             0.000000
Legs               0.000000
LandingPad        28.888889
Block              0.000000
ReusedCount        0.000000
Serial             0.000000
Longitude          0.000000
```

```
launch_site_counts = df['LaunchSite'].value_counts()
print(launch_site_counts)
```

```
LaunchSite
CCSFS SLC 40    55
KSC LC 39A      22
VAFB SLC 4E     13
```

```
for i,outcome in enumerate(landing_outcomes.keys()):
    print(i,outcome)
```

```
0 True ASDS
1 None None
2 True RTLS
3 False ASDS
4 True Ocean
5 False Ocean
6 None ASDS
7 False RTLS
```

We create a set of outcomes where the second stage did not land successfully:

```
bad_outcomes=set(landing_outcomes.keys()[[1,3,5,6,7]])
bad_outcomes
```

```
{'False ASDS', 'False Ocean', 'False RTLS', 'None ASDS', 'None None'}
```

KRISTINA CINOVA

EDA and Visual Analytics

Techniques

- Visualization with Matplotlib, Seaborn, SQL queries for data exploration.

GitHub Links

- EDA with SQL, EDA with Visualizations

Bar charts

- Usage: Compare the number of launches per site.
- Purpose: Clear comparison of categorical data.

Heatmaps

- Usage: Show success rates by launch site and rocket type.
- Purpose: Highlight patterns in data.

Pie Charts

- Usage: Show the proportion of successful vs. unsuccessful landings.
- Purpose: Visualize category proportions.

Histogram

- Usage: Display the distribution of payload mass.
- Purpose: Identify data distribution and outliers.

Scatter plots

- Usage: Explore relationships between payload mass and orbit type.
- Purpose: Detect trends and correlations.

EDA with SQL

Some of the queries:

- **Displayed** launches from sites starting with 'CCA'.
- **Found** the average payload mass carried by the Falcon 9 v1.1 booster.
- **Counted** the total number of each mission outcome (successes and failures).
- **Highlighted** failures on drone ships during 2015.
- **Ranked** landing outcomes from 2010 to 2017 based on frequency.

Ranking 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

[19]:

```
%%sql
SELECT
    "Landing_Outcome",
    COUNT(*) AS Outcome_Count
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome"
ORDER BY Outcome_Count DESC;
```

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

[19]:

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

KRISTINA CINOVA

Methodology - Predictive Analysis

Models
- Logistic Regression, SVM, Decision Tree, KNN.

Process
- Model building, hyperparameter tuning, evaluation.

GitHub Link
- Predictive Analysis

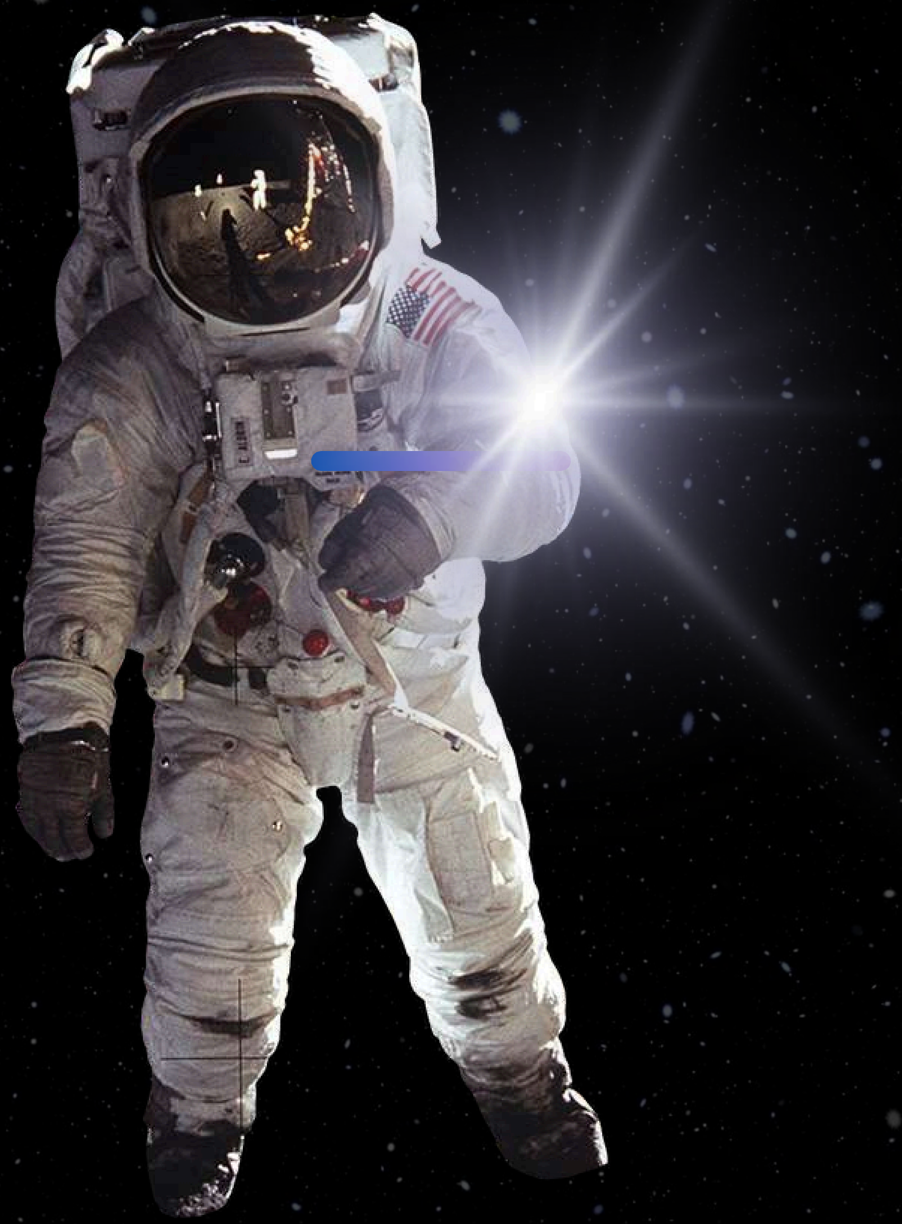
KEY PHASES

- Data preprocessing
- Training and Evaluation
- Feature Selection
- Hyperparameter Tuning
- Model Selection
- Model Deployment

KRISTINA CINOVA

Key Takeaways from SpaceX Exploratory Data Analysis

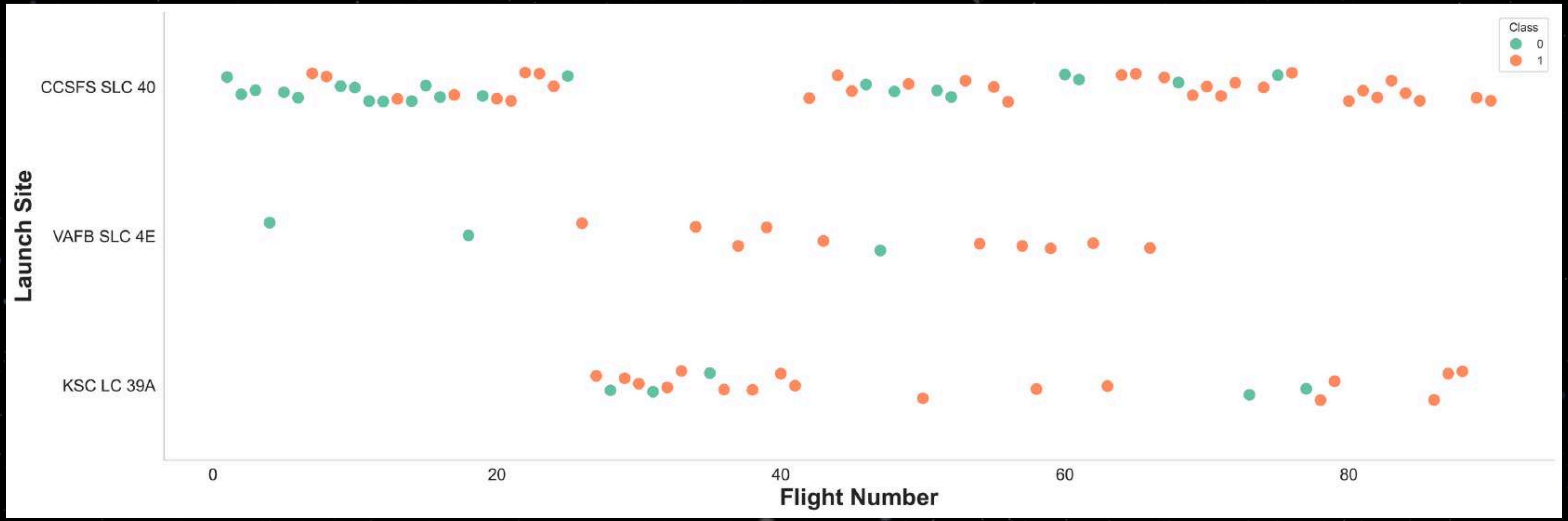
Section 2



EDA and SQL Exploration

- The analysis highlights SpaceX's cost advantage due to Falcon 9's reusable first stage, significantly reducing launch costs compared to competitors.
- The dataset includes SpaceX mission records, enabling an in-depth analysis. SQL queries extracted information on launch sites, payload mass, and mission outcomes.
- Queries revealed trends such as frequent landing outcomes and the first successful ground pad landing.
- Analysis of mission outcomes, including the success rates of different landing types (e.g., ground pad, drone ship), offers insights into predicting launch success and evaluating SpaceX's reliability and reusability strategy

Flight Number vs. Launch Site

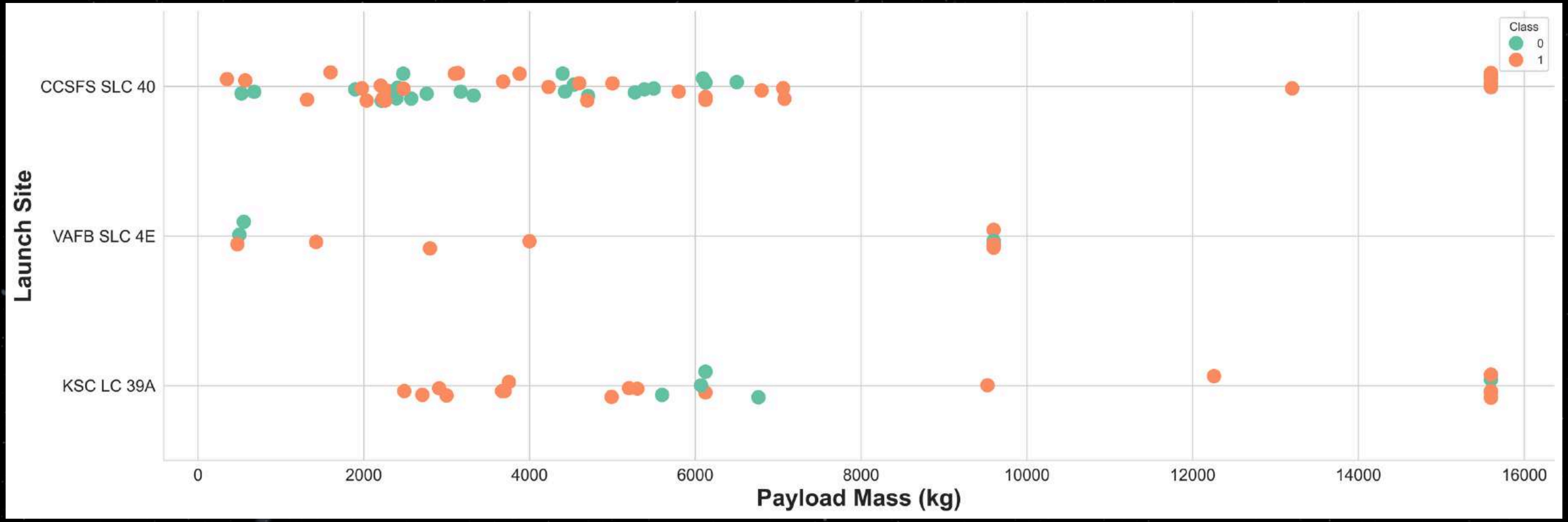


Improvement over time - across all launch sites, there is a clear trend towards higher success rates (more orange points) as flight numbers increase, indicating advancements in technology and procedures.

Site-specific success rate - CCSFS SLC 40 has shown a steady improvement in success rates, while VAFB SLC 4E has experienced more variability, particularly with higher instances of unsuccessful landings.

KRISTINA CINOVA

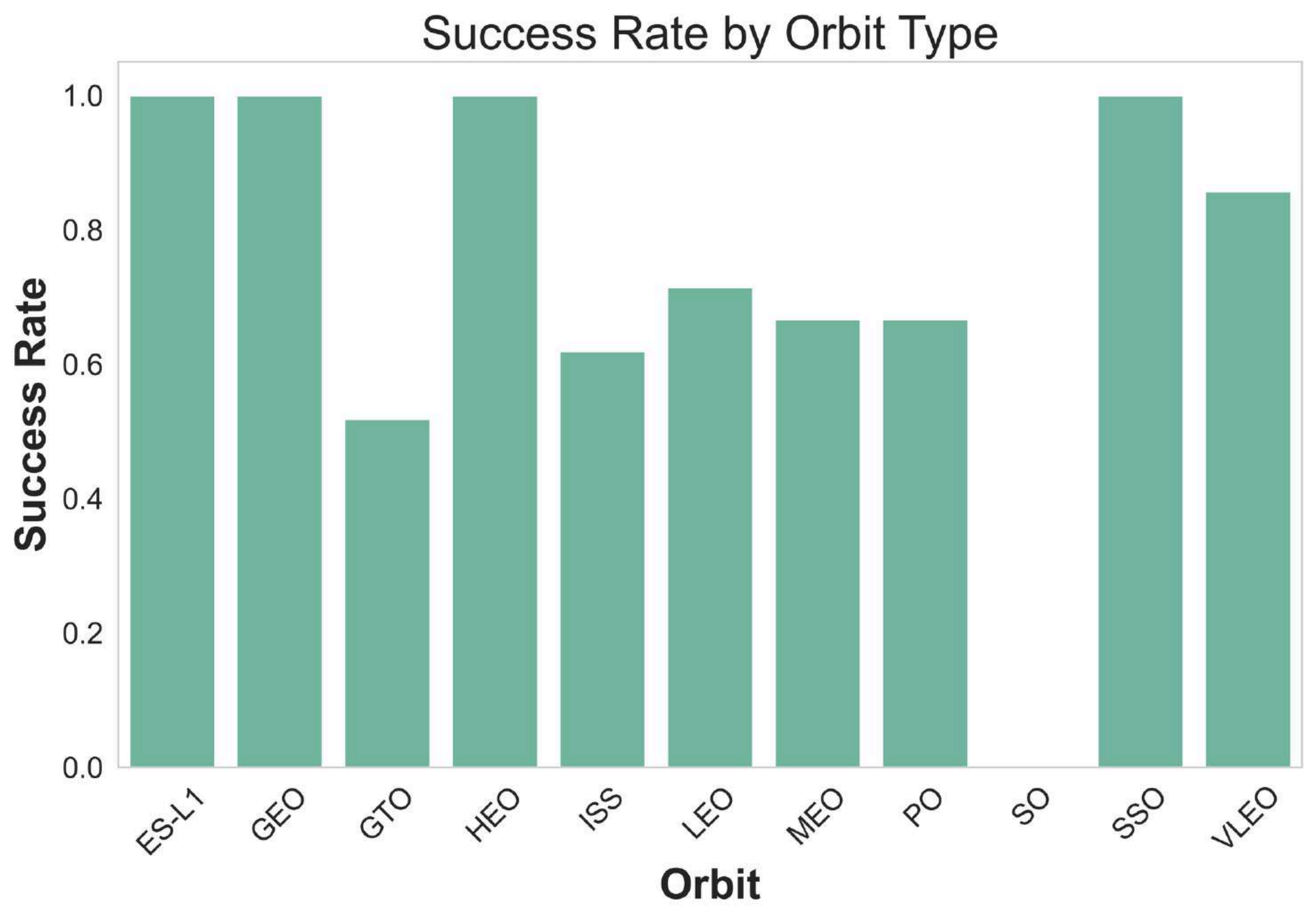
Payload vs. Launch Site



There are no rockets launched in VAFB-SLC launchsite for heavypayload mass(greater than 10000).

KRISTINA CINOVA

Success Rate vs. Orbit Type



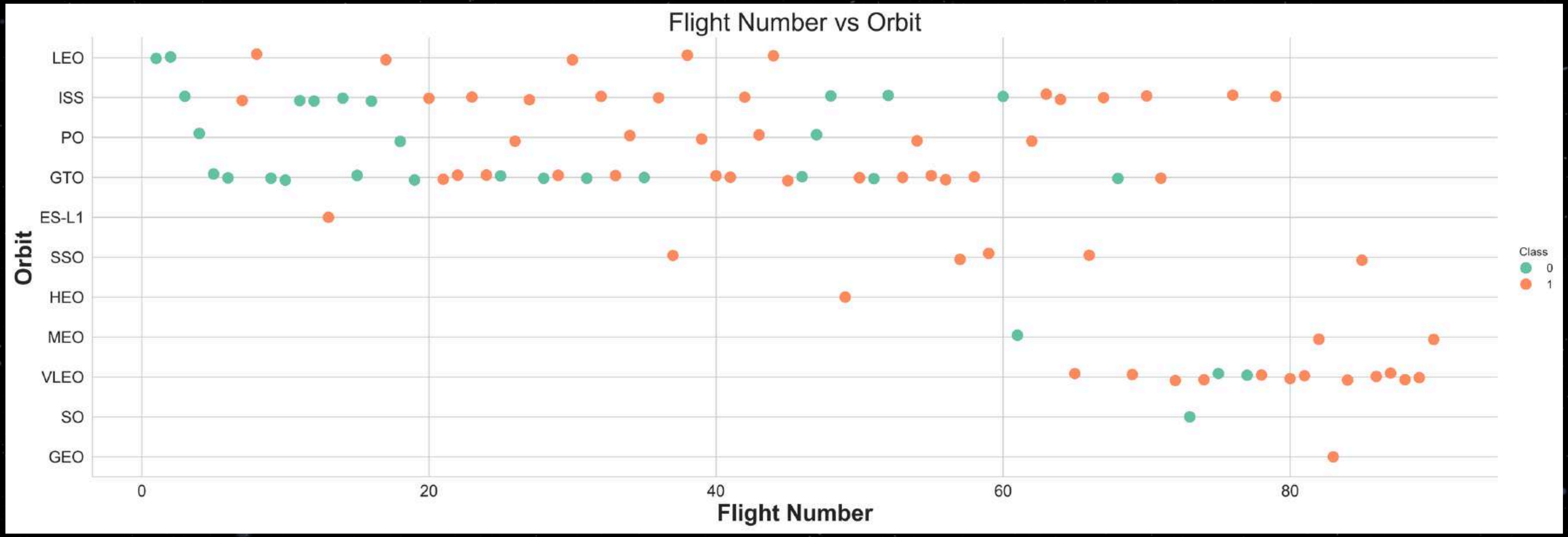
- **ES-L1, GEO, HEO, SSO** exhibit a 100% success rate, indicating that all launches to these orbits have been successful. This high success rate could be due to well-established procedures, favorable conditions, or less complex mission requirements for these orbits.

- **GTO (Geostationary Transfer Orbit)** has a notably lower success rate, around 50%. GTO missions typically involve higher payload masses and more complex maneuvers, which may contribute to a higher failure rate.

- **SO** has 100% failure rate.

KRISTINA CINOVA

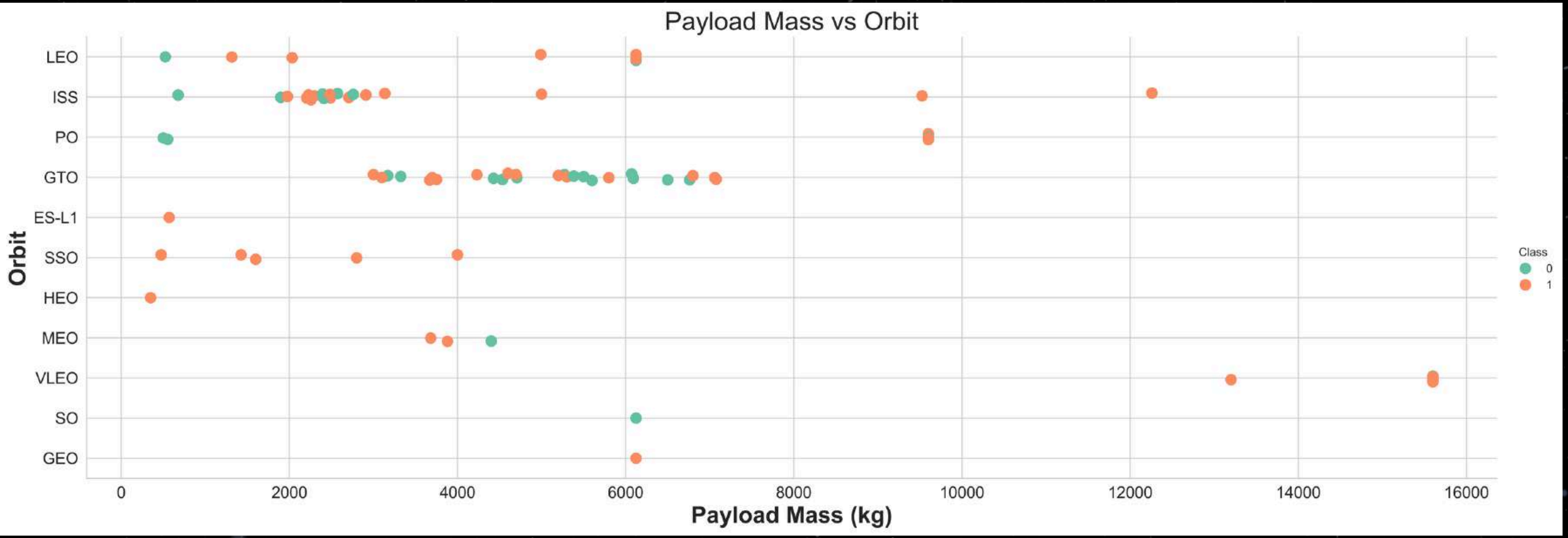
Flight Number vs. Orbit Type



In the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

KRISTINA CINOVA

Payload Mass vs. Orbit Type

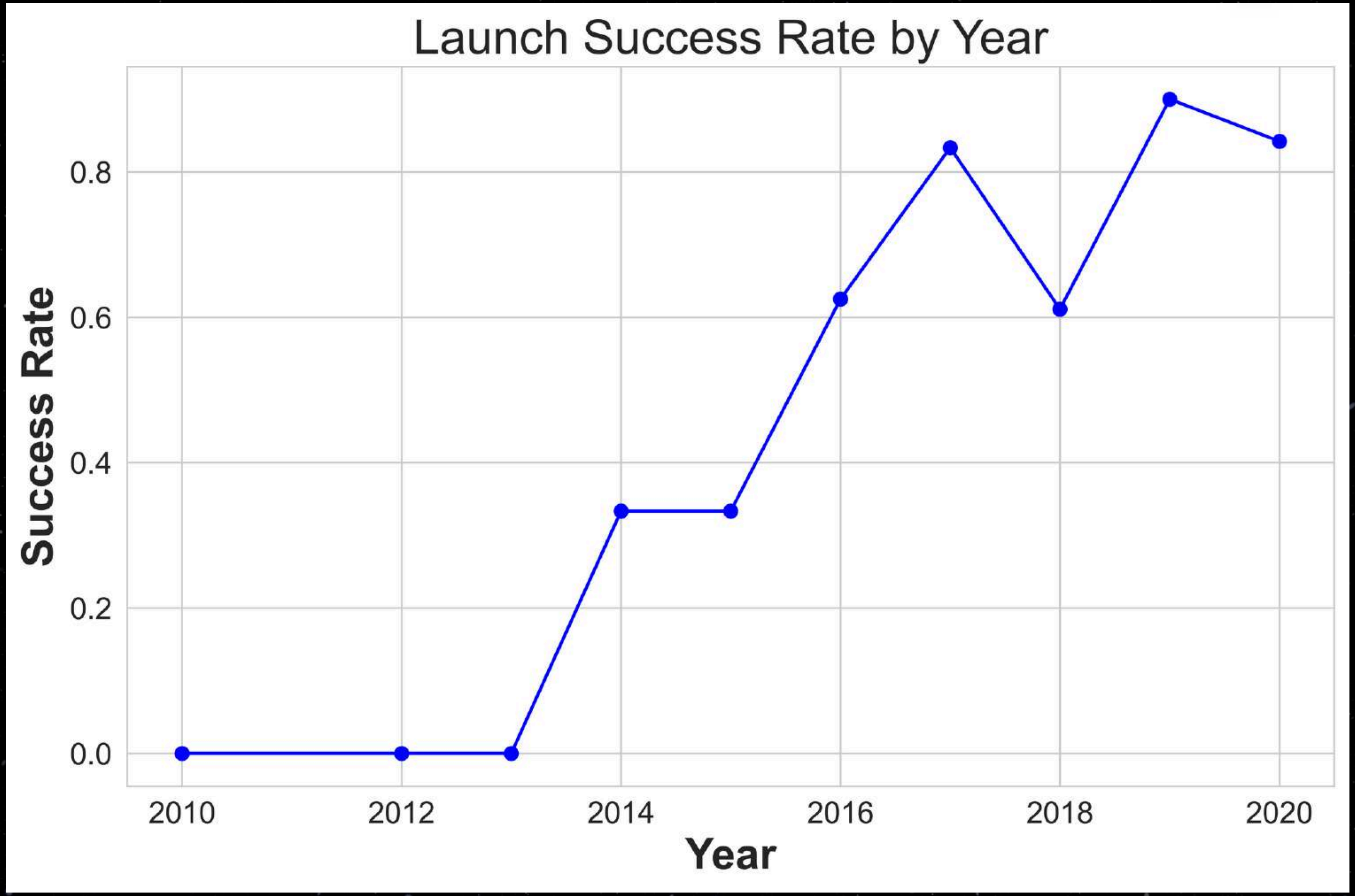


With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.

However for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here.

KRISTINA CINOVA

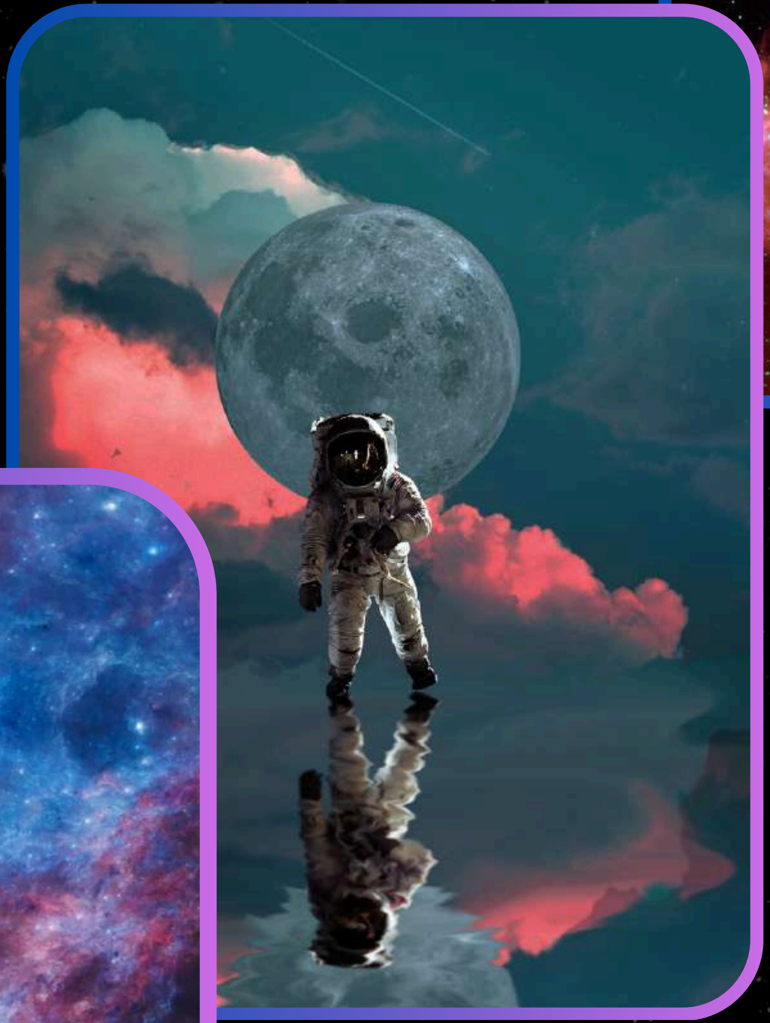
Launch Success Yearly Trend



The success rate increased steadily from 2013 to 2017, showing stability in 2014, with a marked improvement after 2015.

KRISTINA CINOVA

All Launch Site Names



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

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

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

Scientific Project

Launch Site Names Begin with 'CCA'

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

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

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

KRISTINA CINOVA

Total Payload Mass

48,213

The total payload mass carried by boosters launched by NASA (CRS)

```
%%sql
SELECT SUM("PAYLOAD_MASS__KG_") AS Total_Payload_Mass
FROM SPACEXTABLE
WHERE "Customer" LIKE '%NASA (CRS)%';
```

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

Total_Payload_Mass
48213

KRISTINA CINOVA

Average Payload Mass by F9 v1.1

2,928.4

Average payload mass carried by booster version F9 v1.1

```
%%sql
SELECT AVG("PAYLOAD_MASS__KG_") AS Average_Payload_Mass
FROM SPACEXTABLE
WHERE "Booster_Version" = 'F9 v1.1';
```

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

Average_Payload_Mass
2928.4

KRISTINA CINOVA

First Successful Ground Landing Date

Dec 22 2015

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

```
%%sql
SELECT MIN(Date) AS First_Successful_Landing
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (ground pad)';
```

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

First_Successful_Landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%%sql
SELECT "Booster_Version"
FROM SPACEXTABLE
WHERE "Landing_Outcome" = 'Success (drone ship)'
AND "PAYLOAD_MASS__KG_" > 4000
AND "PAYLOAD_MASS__KG_" < 6000;
```

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

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

KRISTINA CINOVA

Total Number of Successful and Failure Mission Outcomes

TOTAL SUCCESSES

61

TOTAL FAILURES

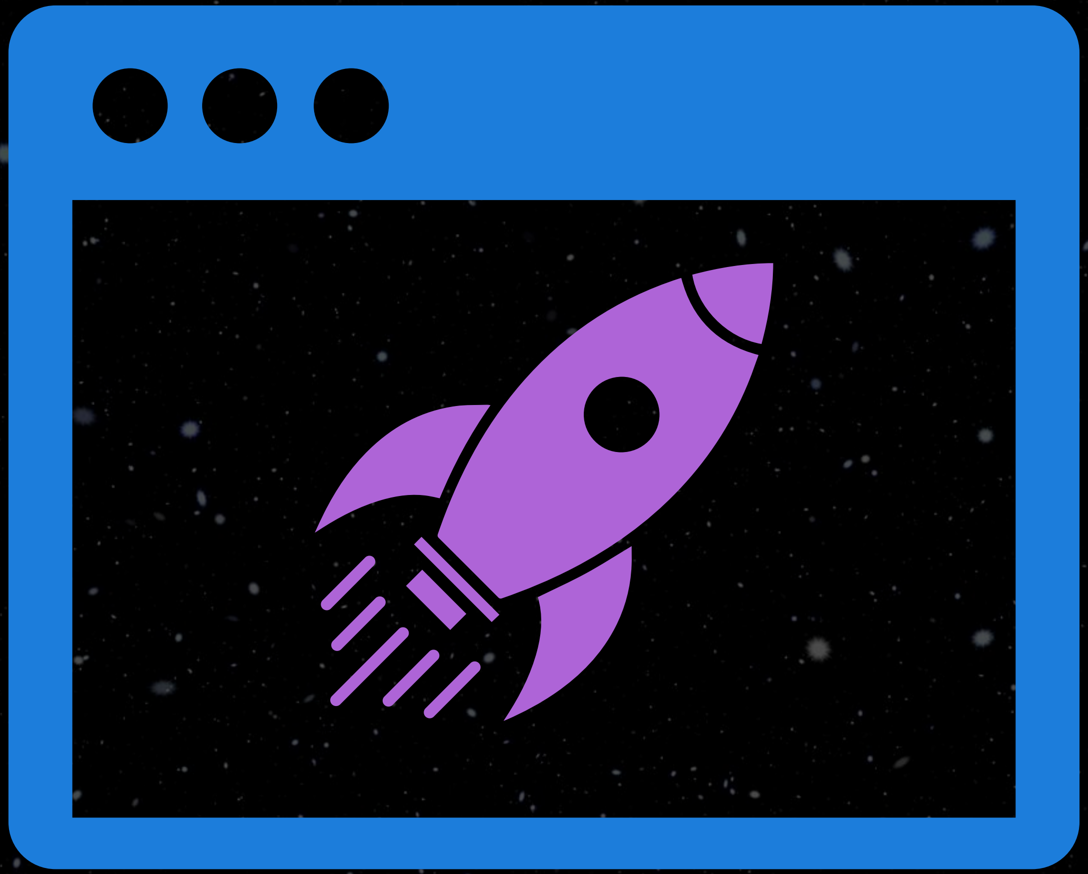
18

```
%%sql
SELECT
    "Landing_Outcome",
    COUNT(*) AS Total
FROM SPACEXTABLE
GROUP BY "Landing_Outcome";
```

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

Landing_Outcome	Total
Controlled (ocean)	5
Failure	3
Failure (drone ship)	5
Failure (parachute)	2
No attempt	21
No attempt	1
Precluded (drone ship)	1
Success	38
Success (drone ship)	14
Success (ground pad)	9
Uncontrolled (ocean)	2

Boosters Carried Maximum Payload



```
%%sql
SELECT "Booster_Version"
FROM SPACEXTABLE
WHERE "PAYLOAD_MASS__KG_" = (SELECT MAX("PAYLOAD_MASS__KG_") FROM SPACEXTABLE);
```

```
 * 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

KRISTINA CINOVA

2015 Launch Records

```
%%sql
SELECT
    substr(Date, 6, 2) AS Month,
    "Landing_Outcome",
    "Booster_Version",
    "Launch_Site"
FROM SPACEXTABLE
WHERE substr(Date, 0, 5) = '2015'
AND "Landing_Outcome" = 'Failure (drone ship)';
```

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

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

KRISTINA CINOVA

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

```
%%sql
SELECT
    "Landing_Outcome",
    COUNT(*) AS Outcome_Count
FROM SPACEXTABLE
WHERE Date BETWEEN '2010-06-04' AND '2017-03-20'
GROUP BY "Landing_Outcome"
ORDER BY Outcome_Count DESC;
```

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

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

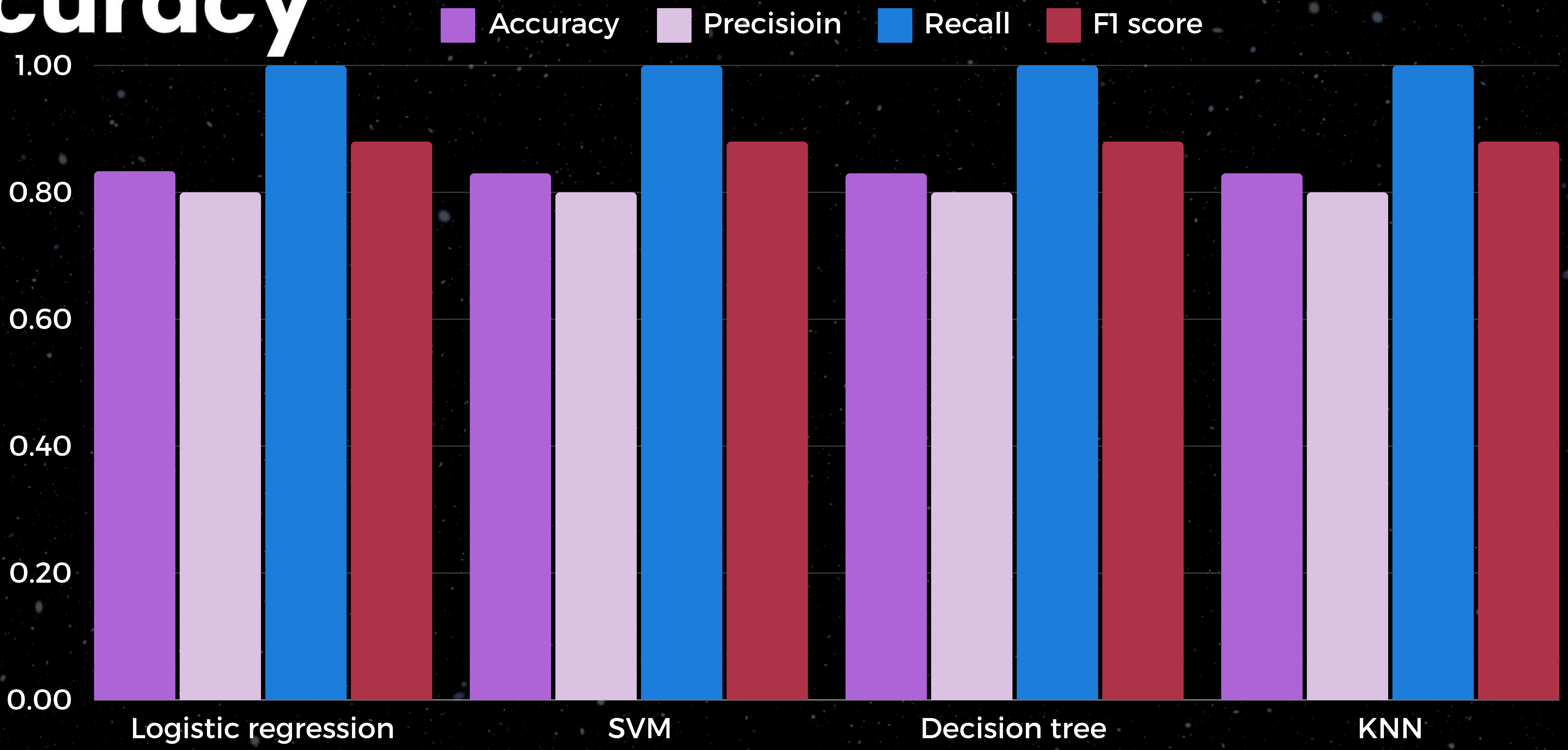
KRISTINA CINOVA

PREDICTIVE ANALYSIS (CLASSIFICATION)

Section 3

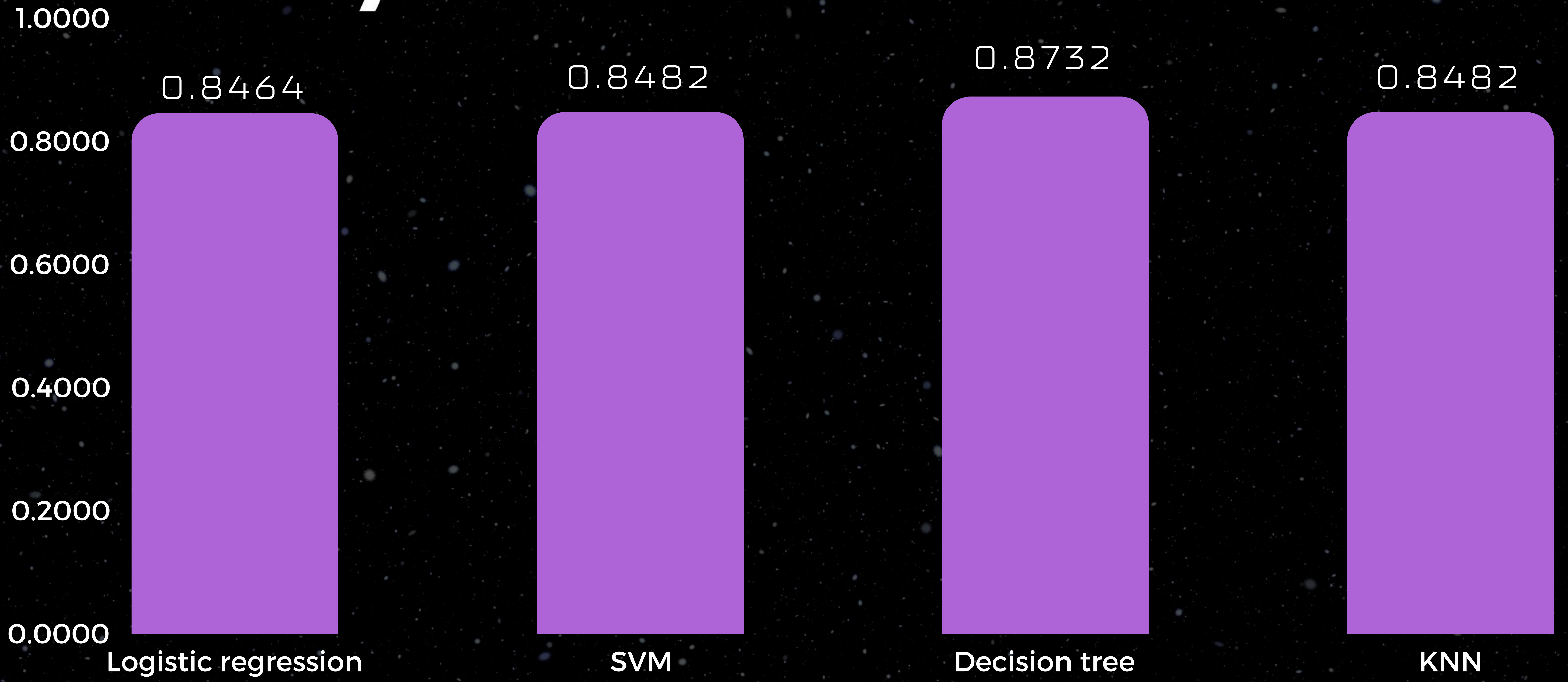
KRISTINA CINOVA

Classification Accuracy



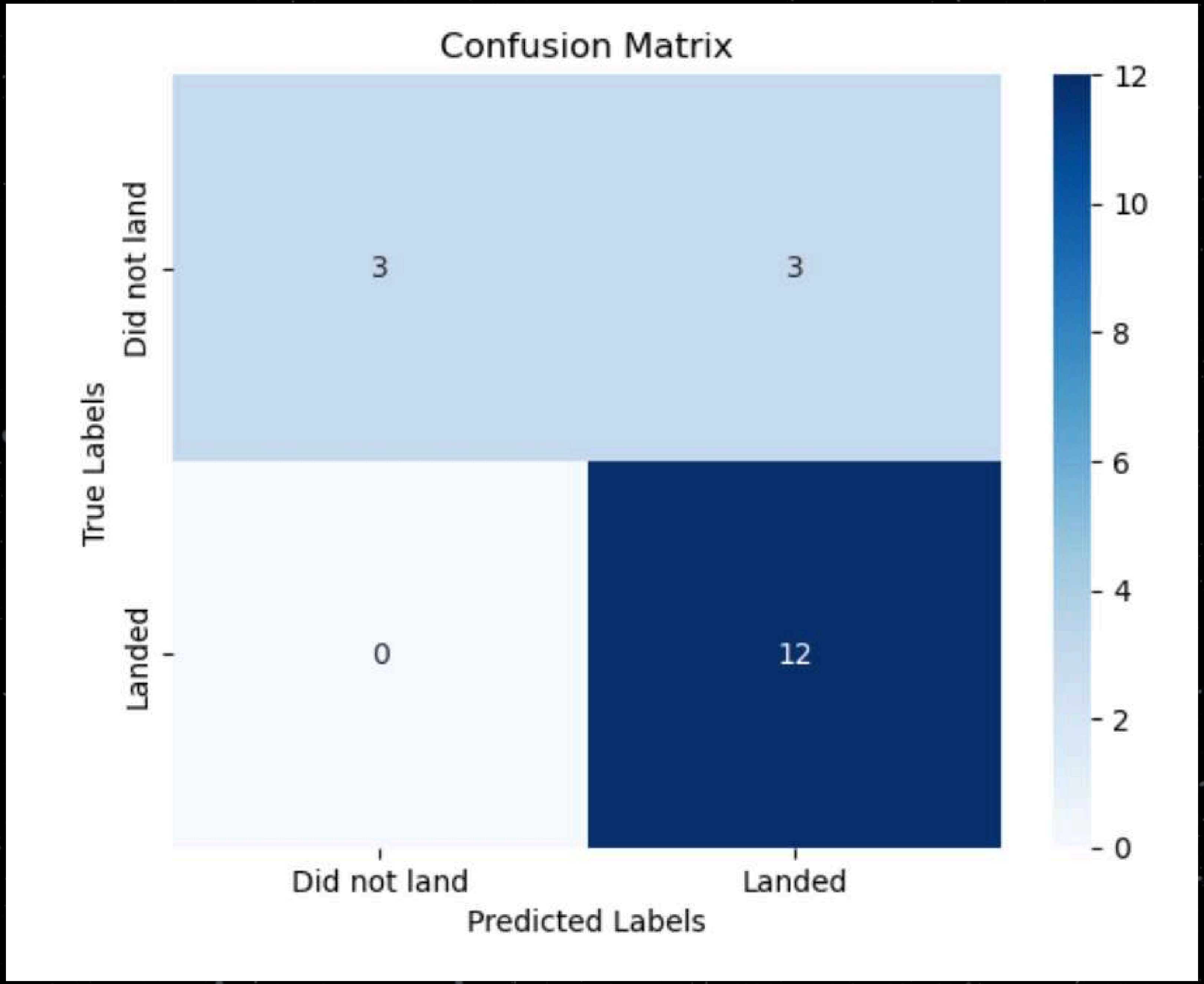
KRISTINA CINOVA

Best Cross-Validation Scores by Model



KRISTINA CINOVA

Confusion matrix



KRISTINA CINOVA

Conclusions

Key Findings

- Falcon 9 launches have shown improving landing success rates.
- Heavier payloads generally correlate with lower landing success.
- Decision Tree model performed best with 87% validation accuracy.
- Logistic Regression, SVM, and KNN models achieved 83% test accuracy.

THANK YOU