Winning Space Race with Data Science

José Eduardo Lezcano García

07/12/22

Outline

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

Summary

- Methodologies
  - Data Collection through API
  - Data Collection through web scraping
  - Data Wrangling
  - Exploratory Data Analysis with SQL
  - Exploratory Data Analysis with Data Visualization
  - Interactive Visual Analytics with Folium
  - Machine Learning Prediction
- Results
  - Exploratory Data Analysis results
  - Interactive Analytics results
  - Predictive Analytics

Introduction

- Project Background and Context
  - According to Space X, Falcon 9 rocket launches cost 62 million dollars whereas other providers charge upwards of 165 million dollars per launch; most of Space X's savings come from reusing 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.
- Questions we want to answer
  - What factor determines if a rocket will launch successfully?
  - What factor determines the probabilities of successful landing?

Methodology

Section 1

Methodology

Executive Summary

- Data collection methodology
  - Data was collected from Wikipedia and SpaceX public API
- Perform Data Wrangling
- Perform Exploratory Data Analysis with SQL
- Perform Interactive Visual Analytics with Folium
- Perform Predictive Machine Learning Models

Data Collection – SpaceX API

The API dataset can be obtained in five steps, as follows:

1) Get response from API.

```
spacex_url="https://api.spacexdata.com/v4/launches/past"
```

```
response = requests.get(spacex_url)
```

2) Convert response to a .json file.

```
# Use json_normalize meethod to convert the json result into a dataframe
data = pd.json_normalize(response.json())
```

Data Collection – SpaceX API

3) Clean Data.

```
# Lets take a subset of our dataframe keeping only the features we want and the flight number, and date_utc.
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]
```

4) Assign to a Dictionary, then to a DataFrame.

```
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}
```

```
# Create a data from launch_dict
data2 = pd.DataFrame(launch_dict)
```

5) Export to a .csv file.

```
data_falcon9.to_csv('dataset_part_1.csv', index=False)
```

Full notebook here

Data Collection - Scraping

We can get the dataset in 5 steps:

1) Request the Falcon9 Launch Wiki page from its URL.

```
# use requests.get() method with the provided static_url
# assign the response to a object
response = requests.get(static_url)
```

2) Create a BeautifulSoup object from the HTML response.

```
# Use BeautifulSoup() to create a BeautifulSoup object from a response text content
BeautifulSoup = BeautifulSoup(response.text, 'html')
```

3) Extract all column/variable names from the HTML table header.

```
# Use the find_all function in the BeautifulSoup object, with element type `table`
# Assign the result to a list called `html_tables`
html_tables = BeautifulSoup.find_all('table')
```

Data Collection - Scraping

4) Create a data frame by parsing the launch HTML tables.

```
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']=[]
```

) Export to a .csv file.

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Full notebook here

Data Wrangling

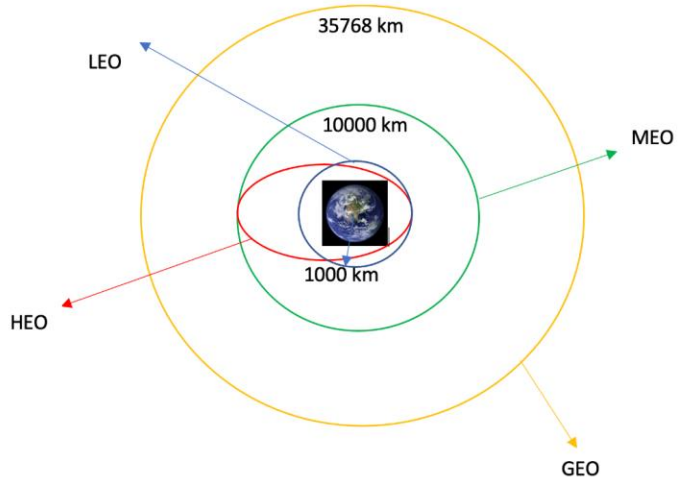
In this stage the first thing to do is calculate the number of launches on each site and Calculate the number and occurrence of each orbit.

```
# Apply value_counts() on column LaunchSite
df['LaunchSite'].value_counts()
```

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

```
# Apply value_counts on Orbit column
df['Orbit'].value_counts()
```

```
GTO      27
ISS      21
VLEO     14
PO        9
LEO       7
SSO       5
MEO       3
ES-L1     1
HEO       1
SO        1
GEO       1
Name: Orbit, dtype: int64
```



Data Wrangling

Landing outcome variable was created from Outcome column and worked out success rate for every landing in dataset.

```
# landing_outcomes = values on Outcome column
landing_outcomes = df['Outcome'].value_counts()

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

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

# landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = df['Outcome'].map(lambda x:0 if x in bad_outcomes else 1)
```

Full notebook here

Exploratory Data Analysis with SQL

- Using SQL, the following queries were performed:
  - Name of the unique launch sites in the space mission.
  - 5 records where launch sites begin with the string 'CCA'.
  - Total payload mass carried by boosters launched by NASA (CRS)
  - Average payload mass carried by booster version F9 v1.1.
  - Date when the first successful landing outcome in ground pad was achieved.
  - 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.
  - Names of the booster versions which have carried the maximum payload mass.
  - Failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015.
  - 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

Full notebook here

Exploratory Data Analysis with Data Visualization

We performed exploratory Data Analysis and Feature Engineering using Pandas and Matplotlib, using three different type of charts:

- Scatterplot: Shows how can a variable affect on another one.
- Bar Chart: Plots relationship between a categorical and a numerical variable.
- Line Chart: Ideal to see changes over periods.

Full notebook here

Data Visualization with Folium

In this stage we:

- Marked all launch sites on a map.
- Marked the success/failed launches for each site on the map.
- Calculate the distances between a launch site to its proximities.

Full notebook here

Insights drawn from EDA

Section 2

All unique Launch Sites

- We used the **DISTINCT** key word to show only the unique Launch Sites from SpaceX dataset.

```
sql SELECT DISTINCT LAUNCH_SITE FROM SPACEX;
```

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

5 records where Launch Sites begin with 'CCA'

- We use the wild card LIKE, which is used when we want to return the row if specific character string matches a specified pattern.
- The LIMIT keyword is used to LIMIT the number of rows of a result set returned.

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

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	07:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	00:35:00	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

Total payload mass carried by boosters launched by NASA (CRS)

```sql
SELECT SUM(PAYLOAD_MASS__KG_) AS TOTAL_PAYLOAD_MASS FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)'
```

- Using the function SUM aggregates the total in the column PAYLOAD_MASS_KG_
- The WHERE clause filters the dataset to only perform calculations on Customer NASA (CRS)

total_payload_mass
45596

Average payload mass carried by booster version F9 v1.1

- The AVG key word grabs all the values from the column AVERAGE_PAYLOAD_MASS and throws the average value.

```
sql SELECT AVG(PAYLOAD_MASS__KG_) AS AVERAGE_PAYLOAD_MASS FROM SPACEX WHERE BOOSTER_VERSION LIKE '%F9 v1.1%'
```

average_payload_mass
2534

First successful landing in ground pad

```
sql SELECT MIN(DATE) AS FIRST_SUCCESSFUL_LANDING FROM SPACEX WHERE LANDING__OUTCOME ='Success (ground pad)';
```

- Using the function MIN calculates the minimum date in the column DATE.
- The WHERE clause filters the dataset to only perform calculations on Landing_Outcome Success (ground pad)

first_successful_landing
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

```
sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' AND (PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000 );
```

```
 * ibm_db_sa://mwg60880:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb
Done.
```

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

- We use a **WHERE** clause to filter successful boosters land on the drone ship and apply the **AND** condition to determine success landing with payload mass greater than 4000 but less than 6000.

Total Number of Successful and Failure Mission Outcomes

- The **COUNT** function is used to count the number of records in the dataset.
- **GROUP BY** clause was added to present the results grouped by the mission outcome (failure and success)

```
sql SELECT MISSION_OUTCOME, COUNT(*) FROM SPACEX GROUP BY MISSION_OUTCOME
```

mission_outcome	2
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

```
sql SELECT BOOSTER_VERSION, PAYLOAD_MASS__KG_ FROM SPACEX WHERE PAYLOAD_MASS__KG_ = (SELECT MAX(PAYLOAD_MASS__KG_) FROM SPACEX)
```

- There is a subquery in the **WHERE** clause, which in this case gives the maximum value of PAYLOAD_MASS__KG_.

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

2015 Launch Records

```
sql SELECT LANDING__OUTCOME, BOOSTER_VERSION, LAUNCH_SITE, DATE FROM SPACEX WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND DATE LIKE '2015%
```

```
 * ibm_db_sa://mwg60880:***@3883e7e4-18f5-4afe-be8c-fa31c41761d2.bs2io90l08kqb1od8lcg.databases.appdomain.cloud:31498/bludb
Done.
```

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

- We use a combination of **WHERE** and **AND** clauses to filter the results for 2015 drone ship landing failures, their booster versions, and launch site names.

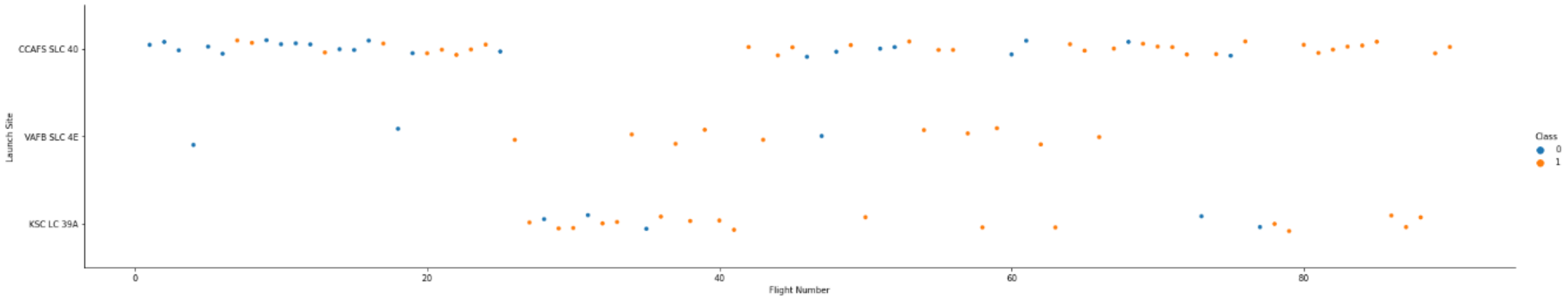
Rank Landing Outcomes Between 2010-06-04 and 2017-03-20

```
sql SELECT LANDING__OUTCOME, COUNT(*) AS COUNT_OUTCOME FROM SPACEX WHERE DATE BETWEEN '2010-06-04' AND '2017-03-20' GROUP BY LANDING__OUTCOME ORDER BY
```

- We selected Landing outcomes and the **COUNT** of landing outcomes from the data and used the **WHERE** clause to filter for landing outcomes **BETWEEN** 2010- 06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

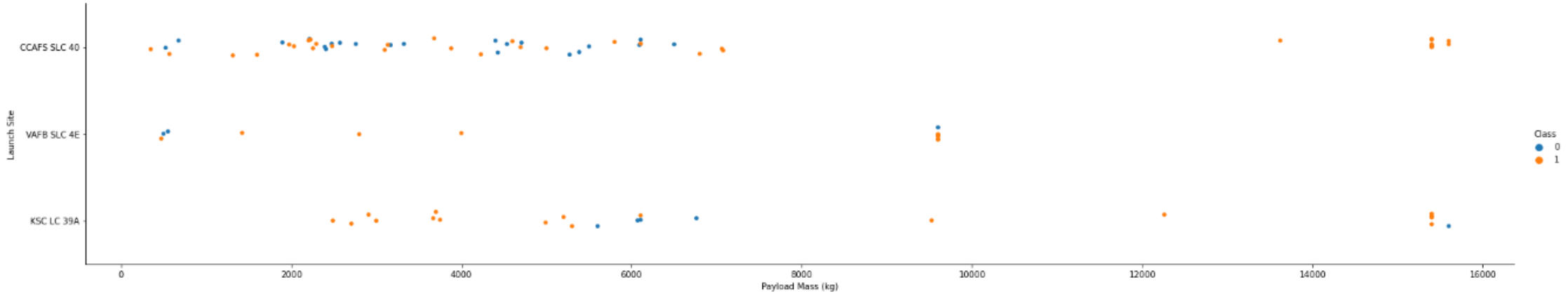
landing__outcome	count_outcome
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Flight Number vs. Launch Site



- First flights were launched mainly from CCAFS LC-40, with a low amount of success. After flight #40 is the predominant launch site and with high success rate.
- From flights 25 till 40, KSC LC-39A took CCAFS LC-40 place, with better overall performance. Onwards, it was a secondary launch site. Probably used when the former is under maintenance.

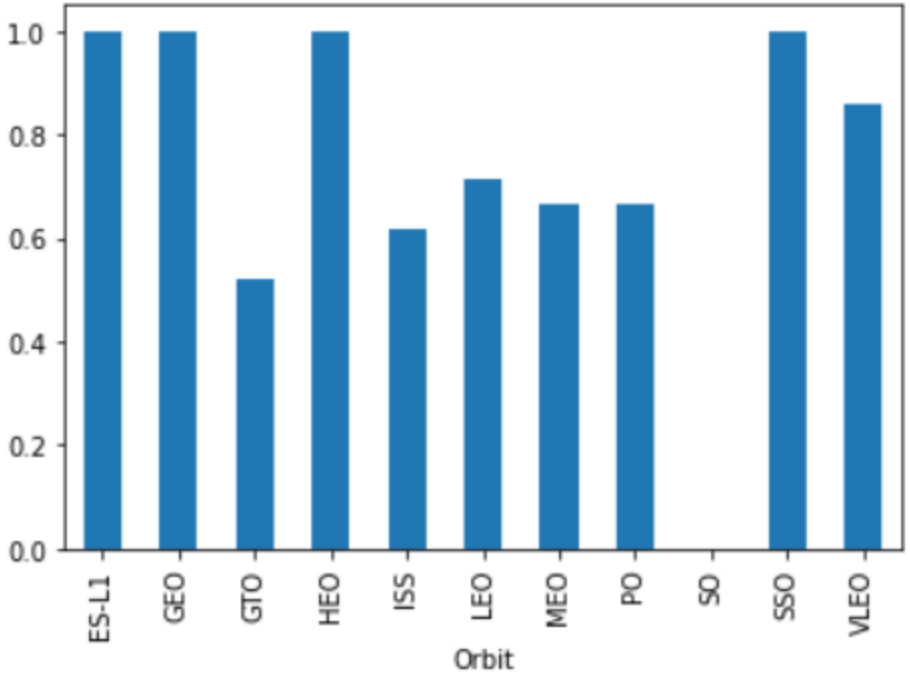
Payload vs. Launch Site



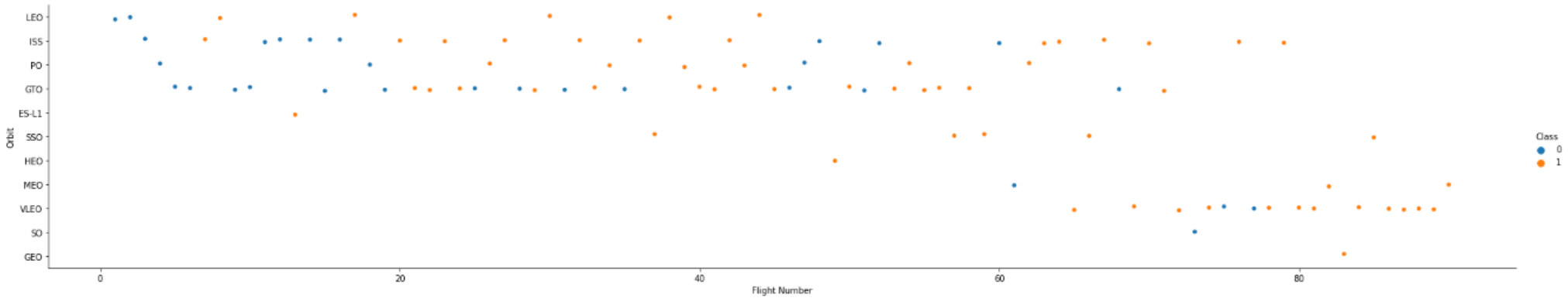
- For the VAFB-SLC Launch Site there are no rockets launched for heavy payload mass (greater than 10000)

Success Rate vs. Orbit Type

- ES-L1, GEO, HEO and SSO orbits have 100% success rate.
- VLEO +80% success rate.

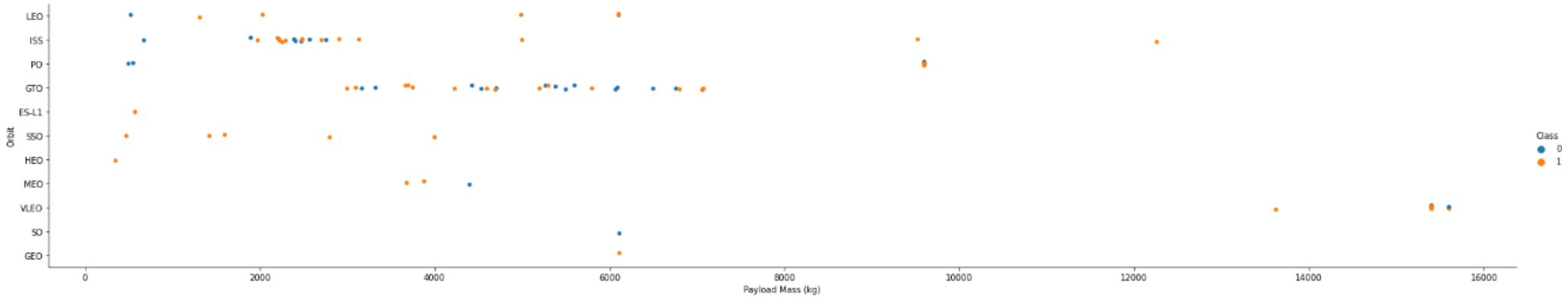


Flight Number vs. Orbit Type



- In the LEO Orbit, Success appears related to the Flight Number.
- There seems to be no relationship between Flight Number when in GTO orbit.

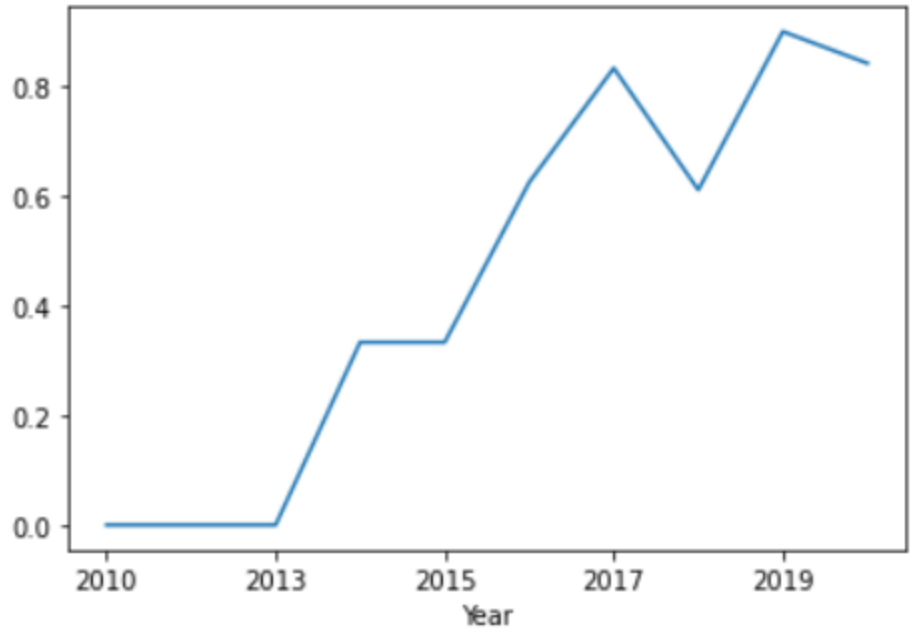
Payload vs. Orbit Type



- With heavy payloads the successful landing rate are higher for Polar, LEO and ISS orbits.
- However, for GTO we cannot distinguish this well as both positive landing rate and negative landing (unsuccessful mission) are both there here.

Launch Success Yearly Trend

- Success rate since 2013 kept increasing till 2020
- In 2017 established in the +80% level, with a dropdown in 2018



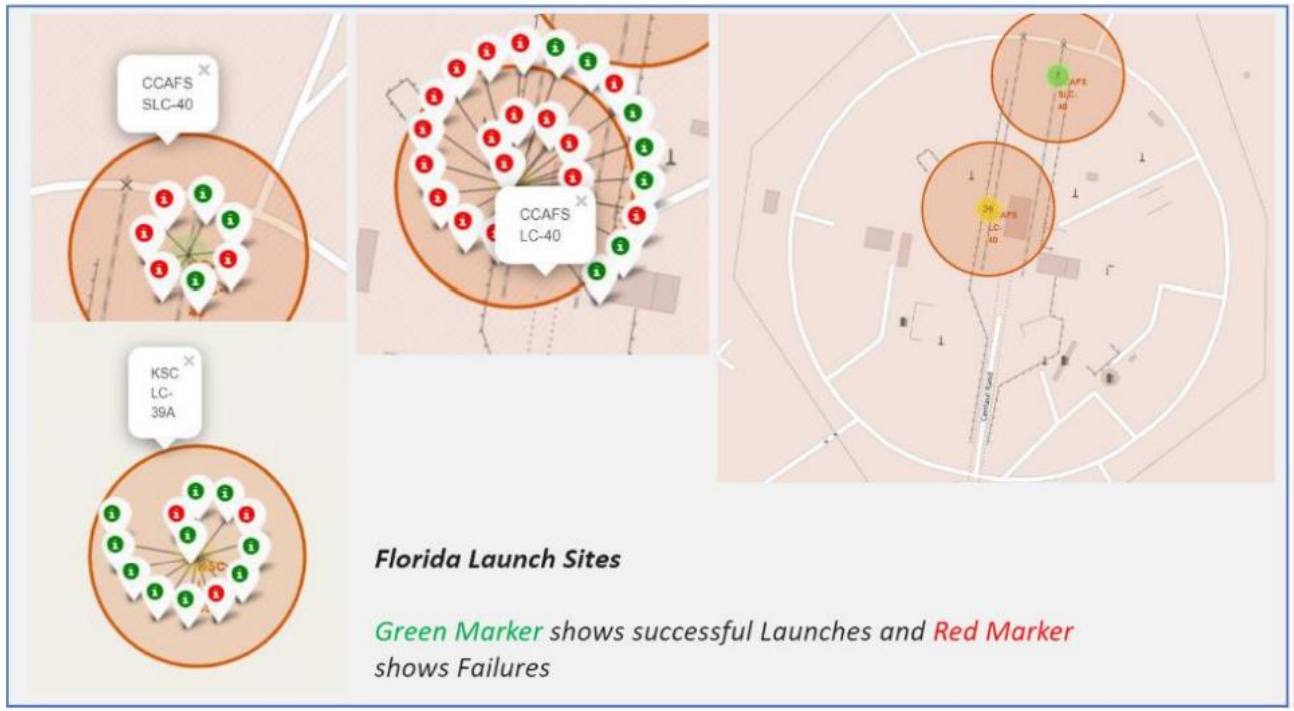
Launch Sites Proximities Analysis

Section 3

Launch Sites' Location Markers

- SpaceX Launch Sites are located in both US coasts, Florida and California.

Markers with color labels

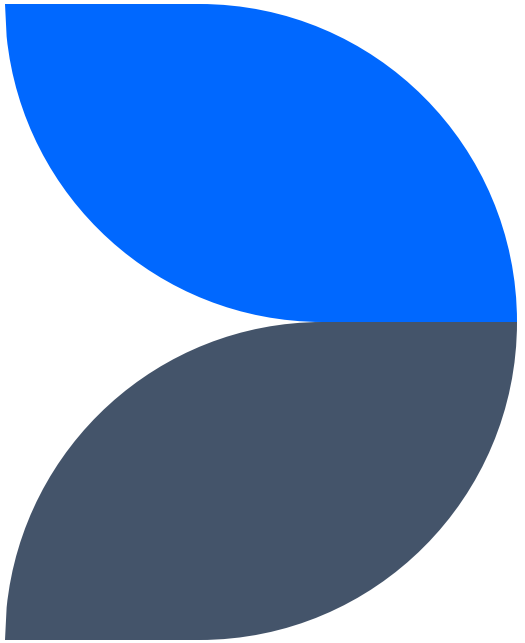


Launch Site’s distance to landmarks

- For safety reasons, SpaceX Launch Sites are located near the ocean.

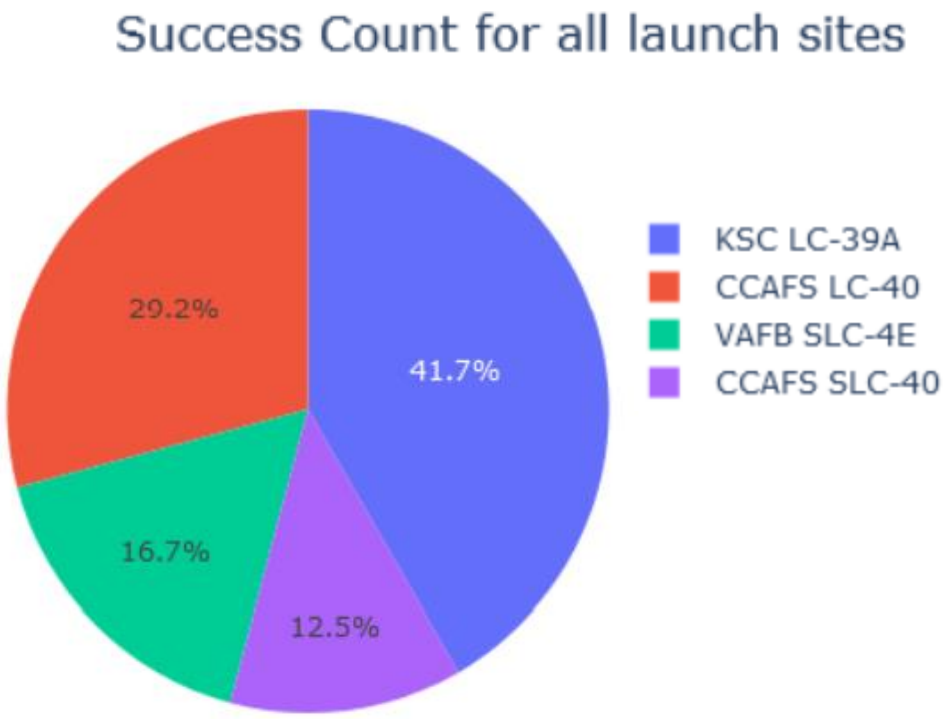
Build a Dashboard with Plotly Dash

Section 4



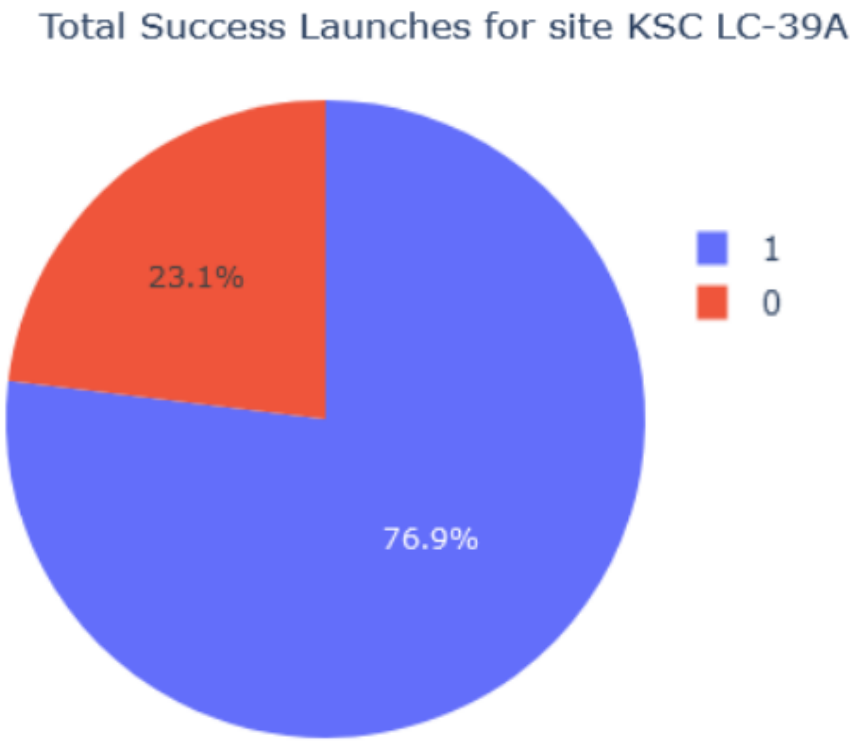
Success percentage achieved per Launch Site

- KSC LC-39A has the highest success percentage with 41.7% of successful launches.



Launch Site with the highest launch success

- KSC LC-39A launch site shows a 76.9% of successful launches and a 23.1% failure rate.

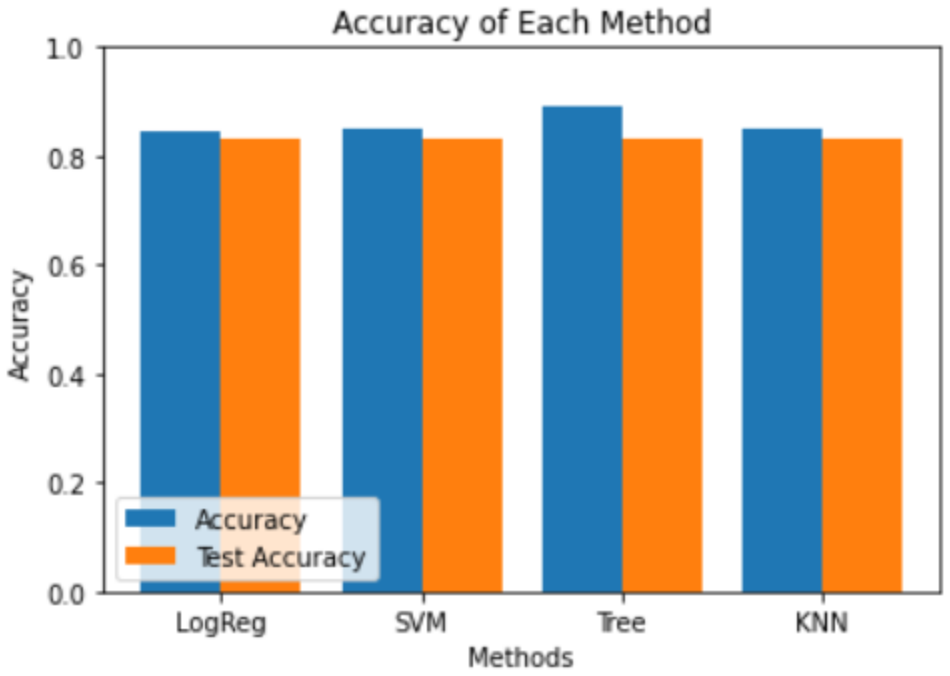


Predictive Analysis

Section 5

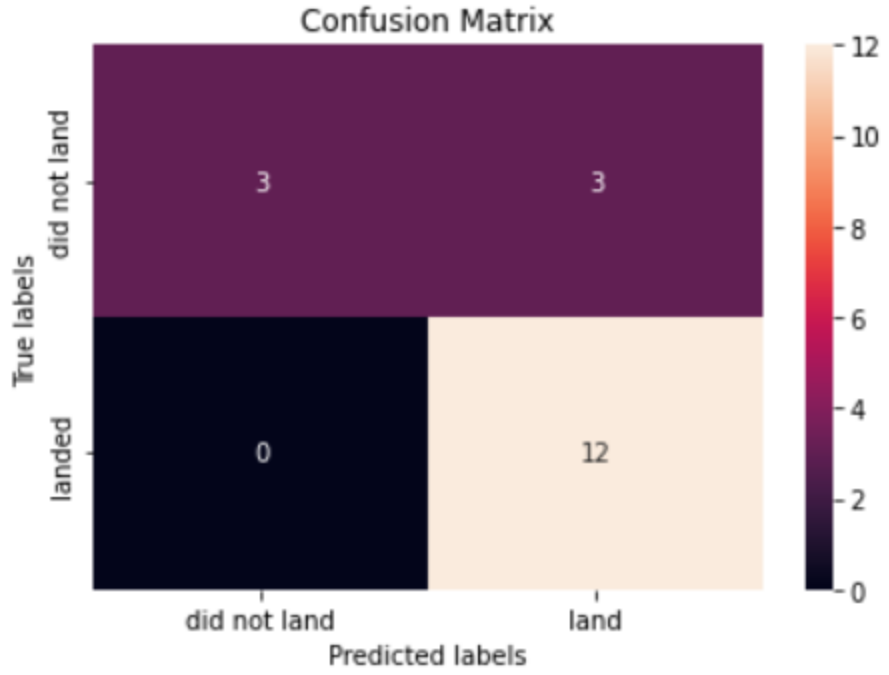
Classification Accuracy

- The final accuracy of the model in the test set, was 94% for 3 of the 4 models. Only Decision Tree showed a different performance of 83%.



Confusion Matrix

- All models show the same confusion matrix



Conclusions

- The larger the flight amount at a launch site, the greater the success rate at a launch site.
- Launch success rate has been increasing since 2013.
- Orbits ES-L1, GEO, HEO, SSO, VLEO had the highest success rate.
- KSC LC-39A had the most successful launches of any sites.
- All Machine Learning Algorithms performed the same, except for Decision Tree Classifier.

Thank you

José Eduardo Lezcano García

Joseleezcanog@gmail.com