IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Leonardo Gomes Cardoso
4 March 2022

# Outline

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

# Executive Summary

The commercial space age is here, companies are making space travel affordable for everyone. Despite many companies trying to make this age happen, SpaceX is the most successful. Thanks to the reuse of the first stage of the rockets, SpaceX launch its rockets at US$ 62mm each, relatively inexpensive near your competitors.

To do that, we scrap the data from SpaceX from its API and from Wikipedia, wrangling the data through SQL, build some EDA and visualization tools and compare multiple Machine Learning algorithms, like logistic regression, SVM, decision tree and KNN. All models had this hyperparameters optimized by a GridSearch methodology.

Based in SpaceX information we can estimate with **89% of accuracy** the chances of the first stage be reusable, which reduce the flight costs, and set the main features that make this happen. We also find a strong relationship between the launch site and the successful rate, and the same for payload mass. Some orbits has best successful rate, and we can see that there is a learning curve for the launch, improving the success.

# Introduction

The commercial space age is here, companies are making space travel affordable for everyone. Perhaps the most successful is SpaceX which launch its rockets by very low cost compared to other companies. The secret behind its success is that they can recover and reuse the first stage most of the times.

Our goals:

- gathering information about Space X and creating dashboards
- determine the price of each launch.
- Predict if SpaceX will reuse the first stage.

Section 1

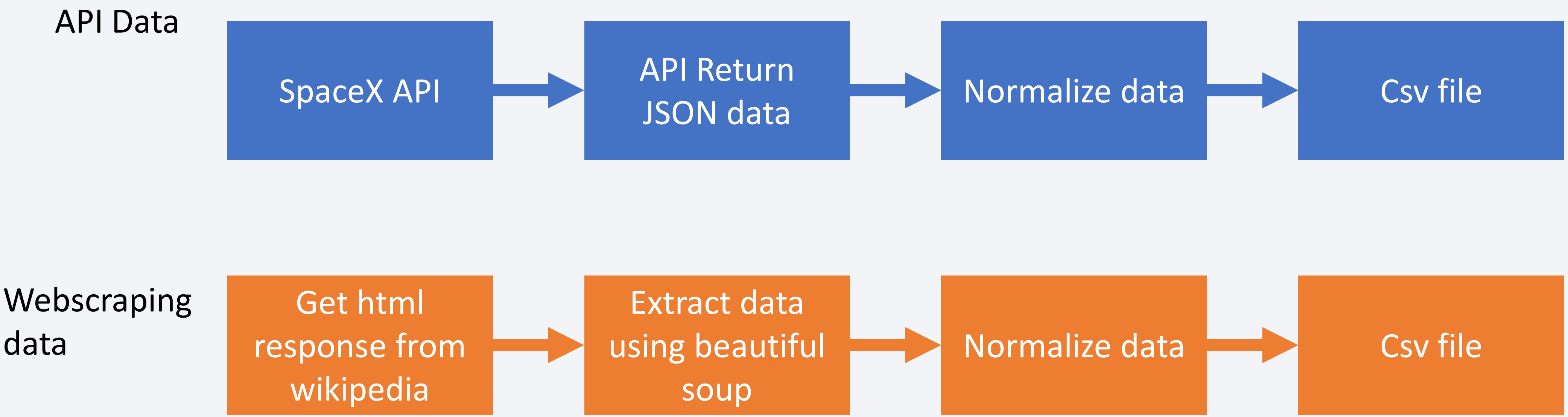
# Methodology

# Methodology

## Executive Summary

- Data collection methodology
  - SpaceX API
  - Web scraping from https://pt.wikipedia.org/wiki/SpaceX
- Perform data wrangling
  - One hot encoding data fields.
  - Drop irrelevant columns
- Exploratory data analysis (EDA) using visualization and SQL
  - Load the data on a SQL DB, prepare the data using SQL queries. Plot some visuals to show correlations between variables and patterns of data
- Interactive visual analytics using Folium and Plotly Dash
- Predictive analysis using classification models
  - Split data in training and test dataset, train the models (logistic regression, svm, decision tree and KNN). Compare the acuracy

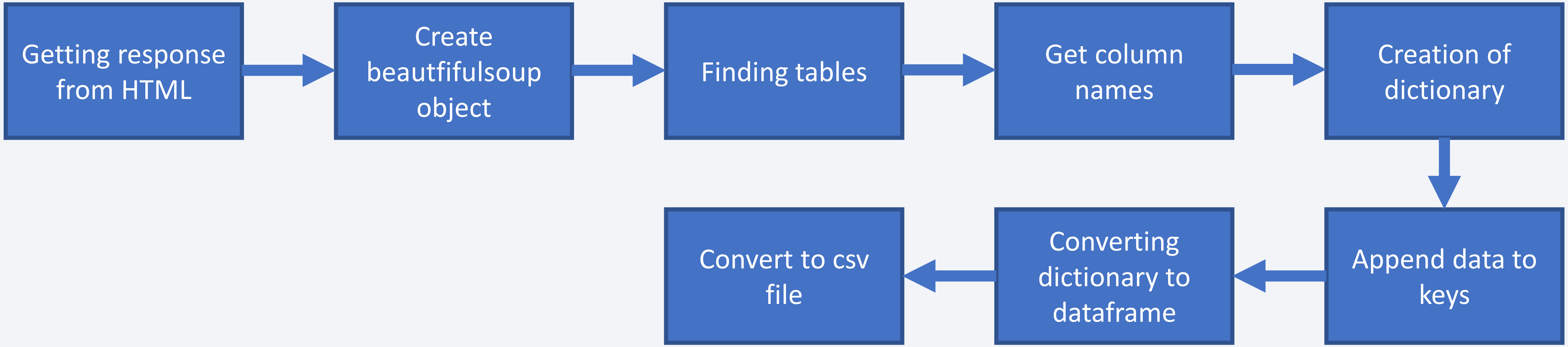
# Data Collection



# Data Collection – SpaceX API


[Link to Github]

# Data Collection - Scraping




Link to Github

# Data Wrangling

Process

Perform EDA on dataset

Number of launches by site

Number of launches by orbit

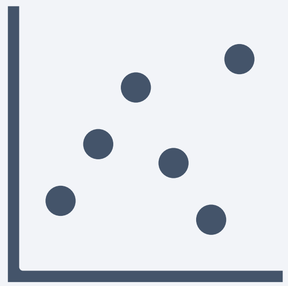
Outcome by orbit

Export csv

Landing outcome label

Sucess rate by landing

# EDA with Data Visualization



- Flight number x Payload
- Flight number x ite
- Payload x Launch Site
- Orbit x Flight number
- Payload x Orbit
- Orbit x Payload

**Scatterplot**

Used to try to identify correlation between its quantitative variables. Obs: Correlation does not mean casuality

- Mean x Orbit

**BarPlot**

Used to try to identify correlation between orbit type (categorical variable) and the rate of successfull (quantitative variables).

- Success Rate x Year

**TimeSeries LinePlot**

Try to find Evolution of a variable over time.

[Link to GitHub]

# EDA with SQL

Read csv files

↓

Load on SQL DB

↓

Run EDA throught SQL queries

- Unique launch sites in the space mission
- 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
- 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
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) bvy date



Link to GitHub

# Build an Interactive Map with Folium

OBJETIVE: Extract some info by maps, using geometrical forms and labels.

1. Mark all launch identify its site.
2. Add a label about the success of the landing
3. Find some POIs near each site, draw a straight line to them and calculate the distance.

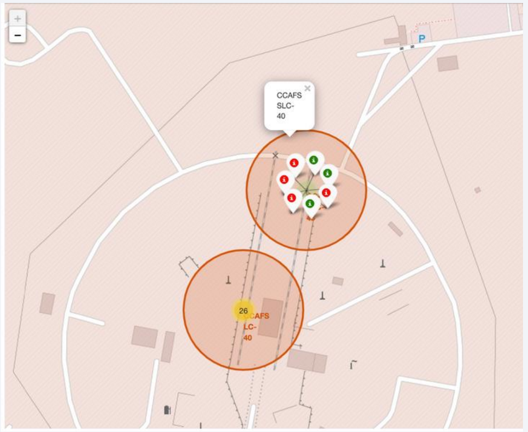
Example of some trends in which the Launch Site is situated in:

Are launch sites in close proximity to railways? No

Are launch sites in close proximity to highways? No

Are launch sites in close proximity to coastline? Yes

Do launch sites keep certain distance away from cities? Yes





[Link to GitHub]

# Build a Dashboard with Plotly Dash

- Objects:
  - dropdown list to enable Launch Site selection — For selection
  - pie chart to show the total successful launches count for all sites — It shows the relationship between two variables. It is the best method to show you a non-linear pattern. The range of data flow, i.e. maximum and minimum value, can be determined.
  - slider to select payload range — For selection
  - scatter chart to show the correlation between payload and launch success — Display relative proportions of multiple classes of data. Size of the circle can be made proportional to the total quantity it represents.
- Callbacks for interactions:
  - callback function for “site_dropdown” as input, “success-pie-chart” as output
  - callback function for “site_dropdown” and “payload-slider” as inputs, `success-payload-scatter-chart` as output

Link to GitHub

# Predictive Analysis (Classification)

Build Model → Evaluate Model → Improving Model → Compare Model

**Build Model**

- Load dataset
- Transform Data
- Training/test split
- Choose algorithms
- Set hyperparameters
- GridSearchCV
- Fit the GridSearchCV objects and train our dataset.

**Evaluate Model**

- Check accuracy for each model
- Get tuned hyperparameters
- Plot Confusion Matrix

**Improving Model**

- Feature Engineering
- Algorithm Tuning

**Compare Model**

- Compare accuracy scores
- Choose the algorithm with best result

Link to GitHub

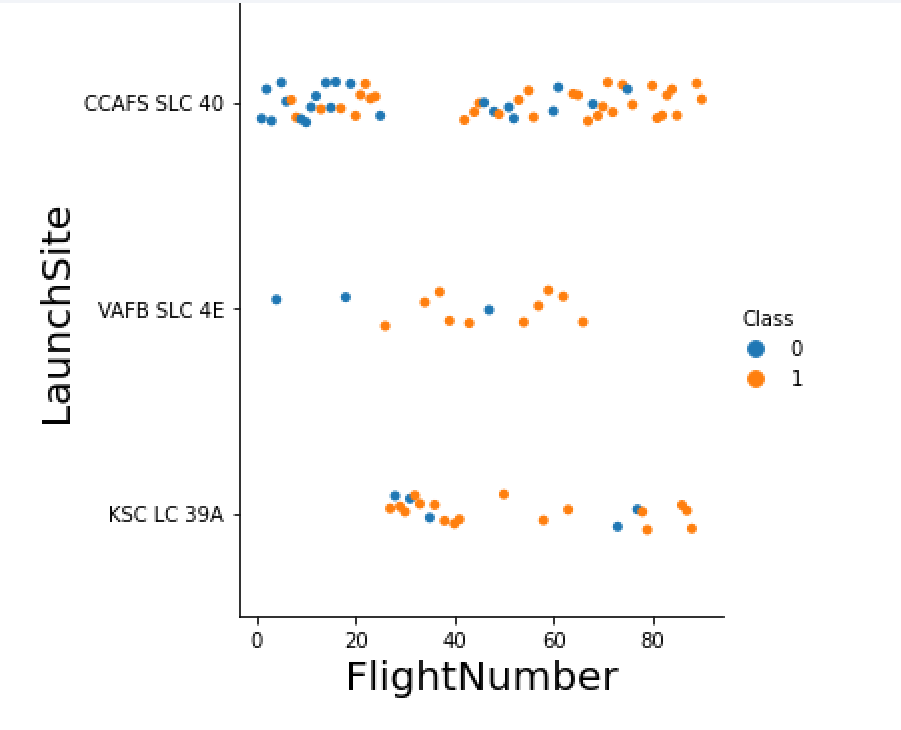
# Results

- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

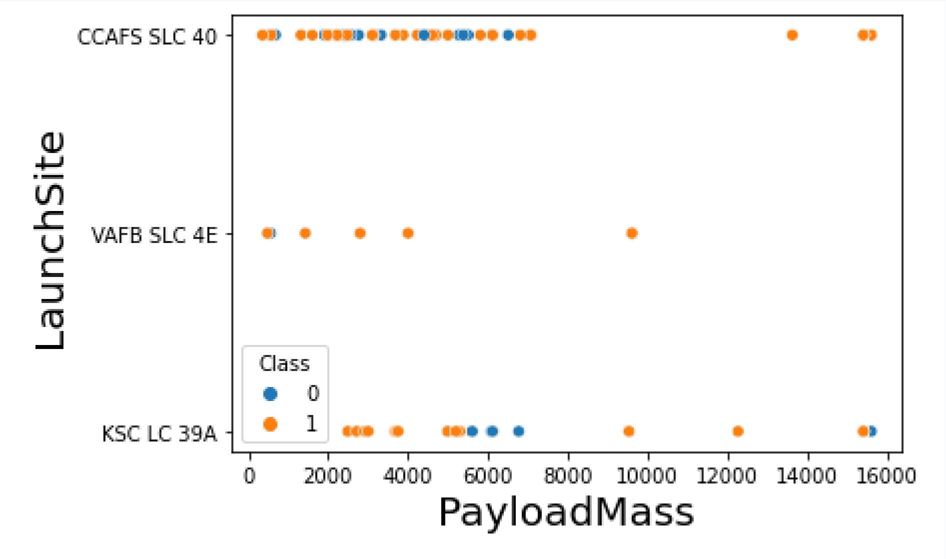
# Insights drawn from EDA

# Flight Number vs. Launch Site



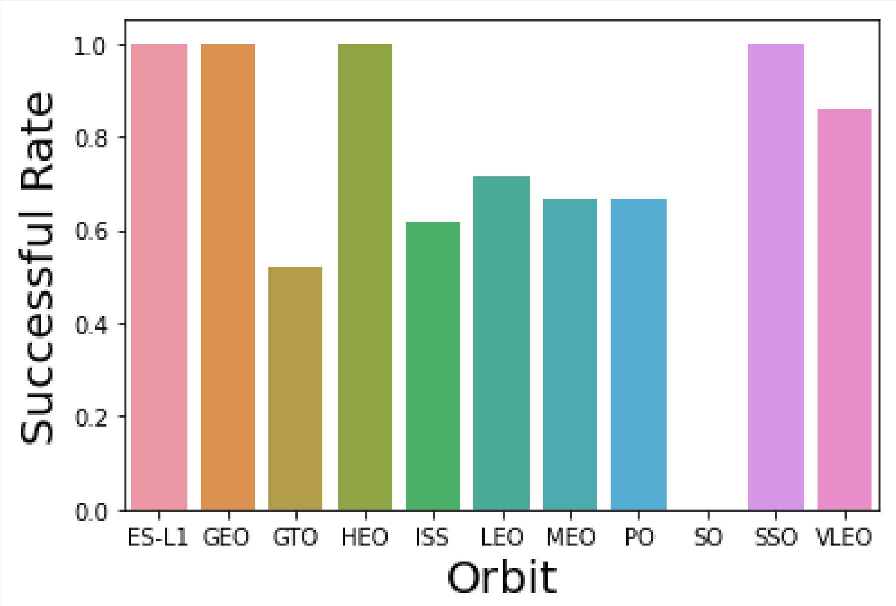

- The more amount of flights at a launch site the greater the success rate at a launch site

# Payload vs. Launch Site



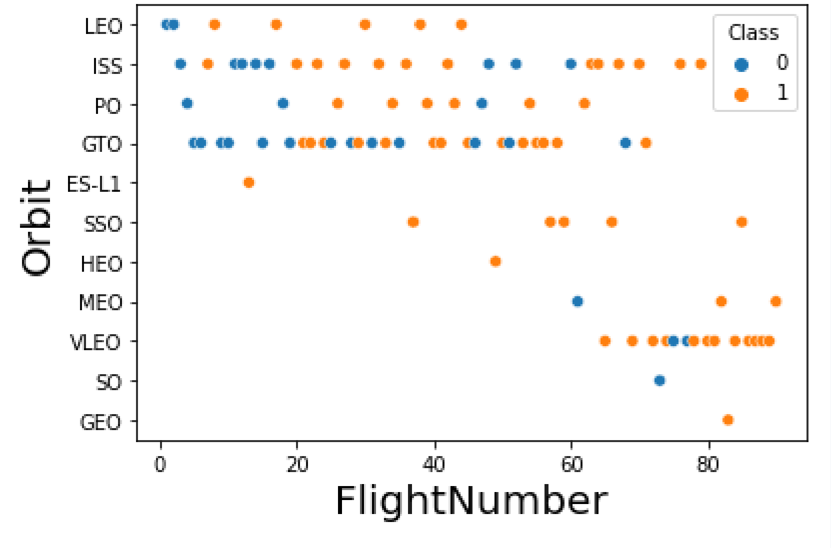

- The greater the payload mass for Launch Site CCAFS SLC 40 the higher the success rate for the Rocket.
- There is not quite a clear pattern to be found using this visualization to make a decision if the Launch Site is dependant on Pay Load Mass for a success launch.

# Success Rate vs. Orbit Type



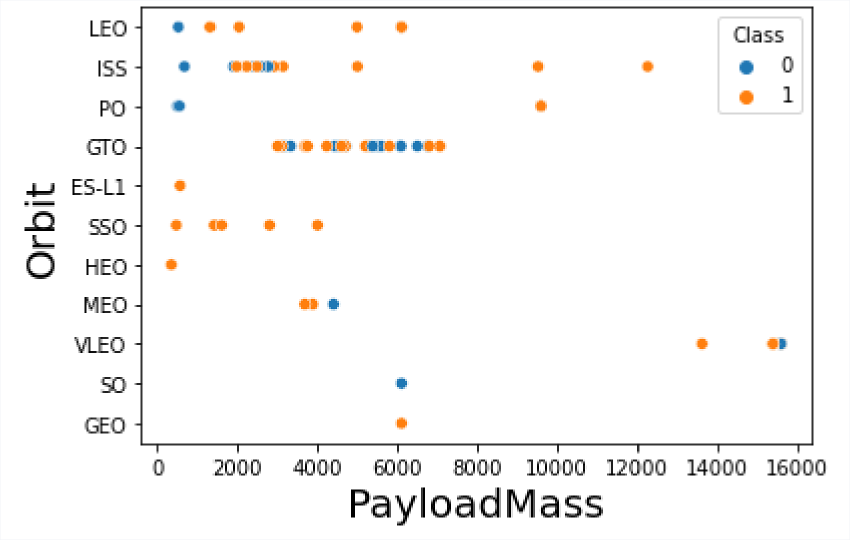

- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate

# Flight Number vs. Orbit Type



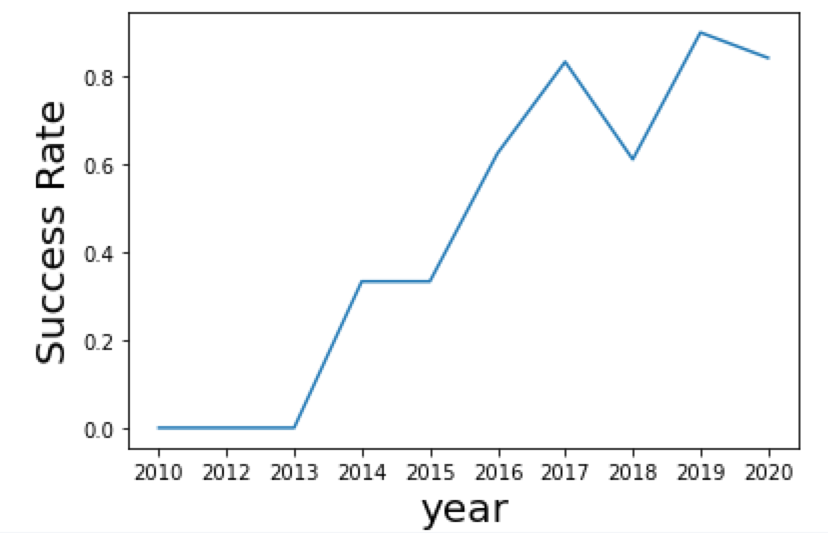

- You should see that in the LEO orbit the Success appears related to the number of flights; on the other hand, there seems to be no relationship between flight number when in GTO orbit.

# Payload vs. Orbit Type




- You should observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

# Launch Success Yearly Trend




- Success rate since 2013 kept increasing until 2020

# All Launch Site Names

```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACEXTBL
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

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

Using the word DISTINCT in the query means that it will only show Unique values in the **LAUNCH_SITE** column from **SPACEXTBL**

# Launch Site Names Begin with 'CCA'

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

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
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 | 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 |

Using the word TOP 5 in the query means that it will only show 5 records from tblSpaceX and LIKE keyword has a wild card with the words 'CCA%' the percentage in the end suggests that the Launch_Site name must start with CCA.

# Total Payload Mass

```
%sql SELECT SUM(payload_mass__kg_) FROM SPACEXTBL WHERE CUSTOMER = 'NASA (CRS)'

 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

| 1 |
|---|
| 45596 |

Using the function SUM summates the total in the column PAYLOAD_MASS_KG_

The WHERE clause filters the dataset to only perform calculations on Customer NASA (CRS)

# Average Payload Mass by F9 v1.1

```
%sql SELECT AVG(payload_mass__kg_) FROM SPACEXTBL WHERE booster_version = 'F9 v1.1'
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

| 1 |
|---|
| 2928 |

Using the function AVG works out the average in the column PAYLOAD_MASS_KG_
The WHERE clause filters the dataset to only perform calculations on Booster_version F9 v1.1

# First Successful Ground Landing Date

List the date when the first successful landing outcome in ground pad was acheived.

*Hint:Use min function*

```
%sql SELECT MIN(DATE) FROM SPACEXTBL WHERE landing__outcome = 'Success (ground pad)'
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

| 1 |
| --- |
| 2015-12-22 |

Using the function MIN works out the minimum date in the column Date
The WHERE clause filters the dataset to only perform calculations on landing_outcome Success (drone ship)

# Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql SELECT DISTINCT (booster_version) FROM SPACEXTBL WHERE landing__outcome = 'Success (drone ship)' AND payload_mass__kg_ > 4000 AN
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

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

Selecting only booster_version
The WHERE clause filters the dataset to landing_outcome = Success (drone ship)
The AND clause specifies additional filter conditions
payload_mass_kg_ > 4000 AND payload_mass_kg_ < 6000

# Total Number of Successful and Failure Mission Outcomes

***List the total number of successful and failure mission outcomes***

```
%%sql
SELECT Count(mission_outcome) as success from spacextbl where mission_outcome LIKE '%Success%'
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

6]:

| success |
|---|
| 100 |

```
%%sql
SELECT Count(mission_outcome)  as failure from spacextbl where mission_outcome LIKE '%Failure%'
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32716/bludb
Done.
```

8]:

| failure |
|---|
| 1 |

The mission outcome has labels with the words success and failure. To calculate the number of each one, I used two queries, looking for these words, associate with count function.

# Boosters Carried Maximum Payload

```
%%sql SELECT DISTINCT (booster_version)
    FROM SPACEXTBL
    WHERE payload_mass__kg_ =
        (SELECT MAX(payload_mass__kg_) FROM SPACEXTBL)
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c
Done.
```

]:

| booster_version |
|---|
| F9 B5 B1048.4 |
| F9 B5 B1048.5 |
| F9 B5 B1049.4 |
| F9 B5 B1049.5 |
| F9 B5 B1049.7 |
| F9 B5 B1051.3 |
| F9 B5 B1051.4 |
| F9 B5 B1051.6 |
| F9 B5 B1056.4 |
| F9 B5 B1058.3 |
| F9 B5 B1060.2 |
| F9 B5 B1060.3 |

- Using the word DISTINCT in the query means that it will only show Unique values in the BOOSTER_VERSION column from SPACEXTBL
- GROUP BY puts the list in order set to a certain condition.
- DESC means its arranging the dataset into descending orde

# 2015 Launch Records

```
%sql SELECT  landing__outcome, booster_version, launch_site FROM SPACEXTBL WHERE YEAR(DATE) = '2015'
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c6a74e.c1ogj3sd0tgtu0lqde00.databases.appd
Done.
```

]:

| landing__outcome | booster_version | launch_site |
|---|---|---|
| Failure (drone ship) | F9 v1.1 B1012 | CCAFS LC-40 |
| Controlled (ocean) | F9 v1.1 B1013 | CCAFS LC-40 |
| No attempt | F9 v1.1 B1014 | CCAFS LC-40 |
| Failure (drone ship) | F9 v1.1 B1015 | CCAFS LC-40 |
| No attempt | F9 v1.1 B1016 | CCAFS LC-40 |
| Precluded (drone ship) | F9 v1.1 B1018 | CCAFS LC-40 |
| Success (ground pad) | F9 FT B1019 | CCAFS LC-40 |

- Using the native function YEAR, I filter the data for the 2015 data.

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

```
%%sql
SELECT landing__outcome, COUNT(*) AS contagem FROM SPACEXTBL
    WHERE DATE >= '2010-06-04'
    AND DATE < '2017-03-21'
    GROUP BY landing__outcome
    ORDER BY contagem DESC
```

```
 * ibm_db_sa://vvb76970:***@b70af05b-76e4-4bca-a1f5-23dbb4c
Done.
```

]:

| landing__outcome | contagem |
|---|---|
| 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 |

Function COUNT counts records in column

- WHERE filters data
- LIKE (wildcard)
- AND (conditions)
- AND (conditions)

Section 3

# Launch Sites Proximities Analysis

# Where are the Launch Sites?


- Despite the multiple launch sites, they are all located at US Coasts, in Florida and California.

# Visualize the Success and Failure


- Red for failure
- Green for success

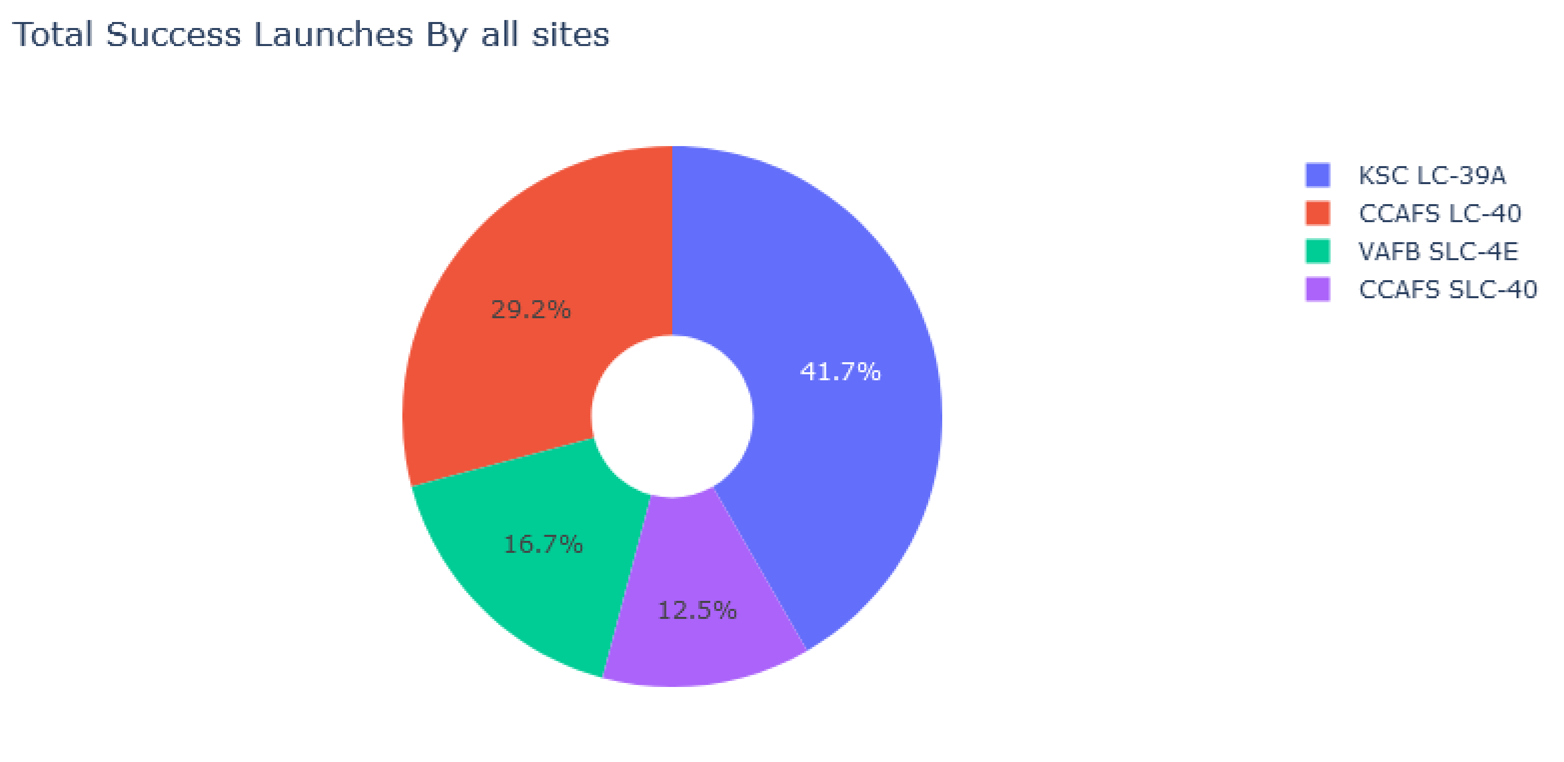
# Where are the POIs?


- The nearest coastline for CCAFS SLC-40 is less than 1km of distance.

Section 4

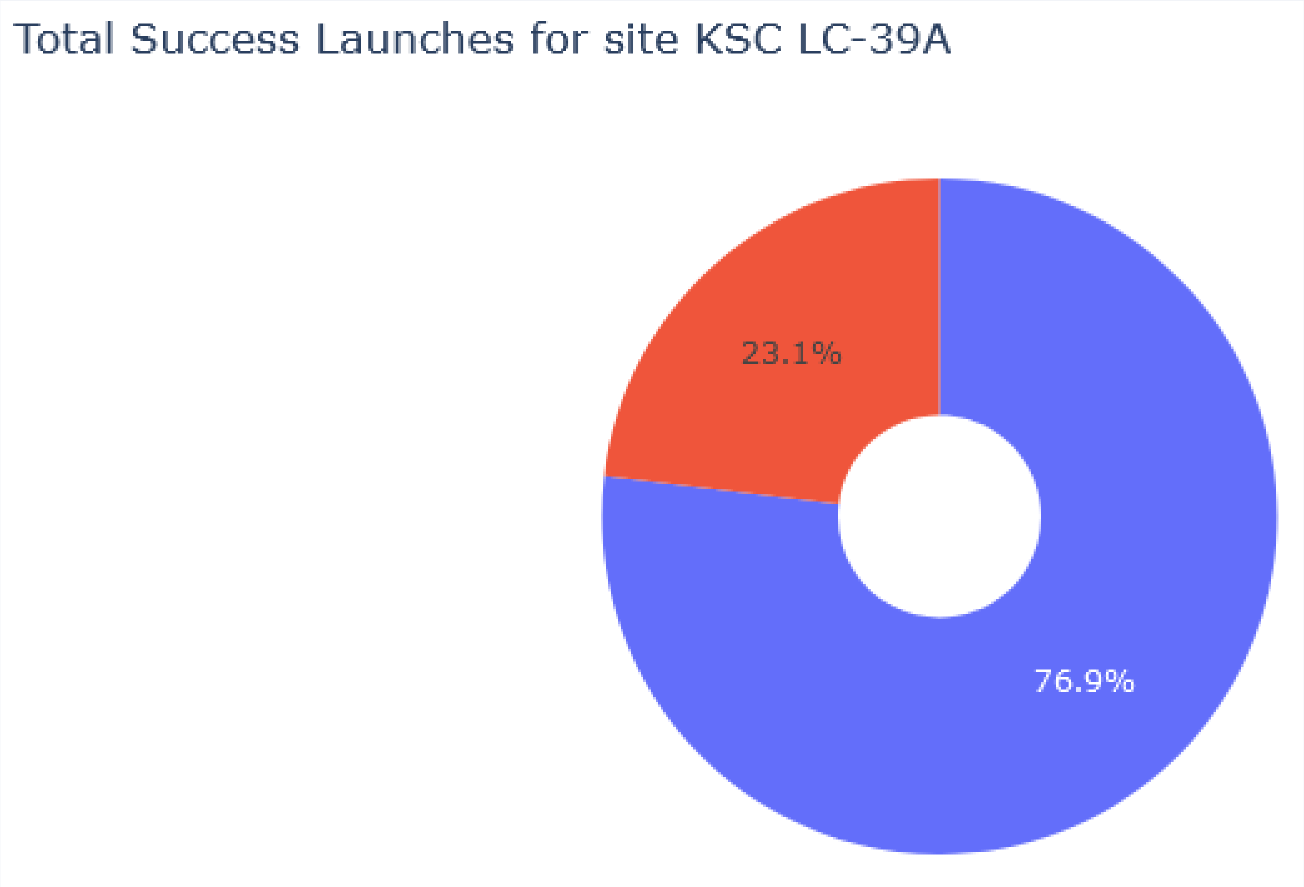
# Build a Dashboard with Plotly Dash

# Success Launches by site




- KSC LC-39A has the majority of success launches, follow by CCAFS LC-40. Together, they represent mores than 70% of success launches

# KSC LC-39A is the best performer launch site

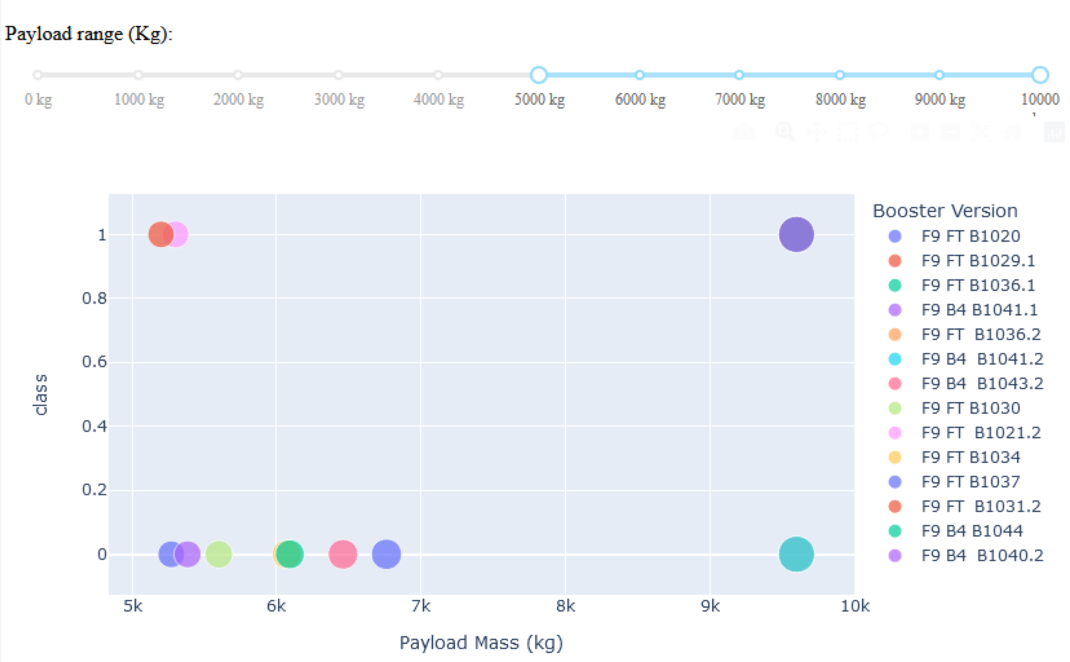



KSC LC-39A perform above 75% of success for its launchs, indicating that launch a rocket from this site could improve the chances of success.

# Payload mass reduce its chance of success

Payload mass 0kg – 5000kg


Payload mass 5000kg – 10000kg




Success rates for low weighted payloads are higher than the heavy weighted payloads

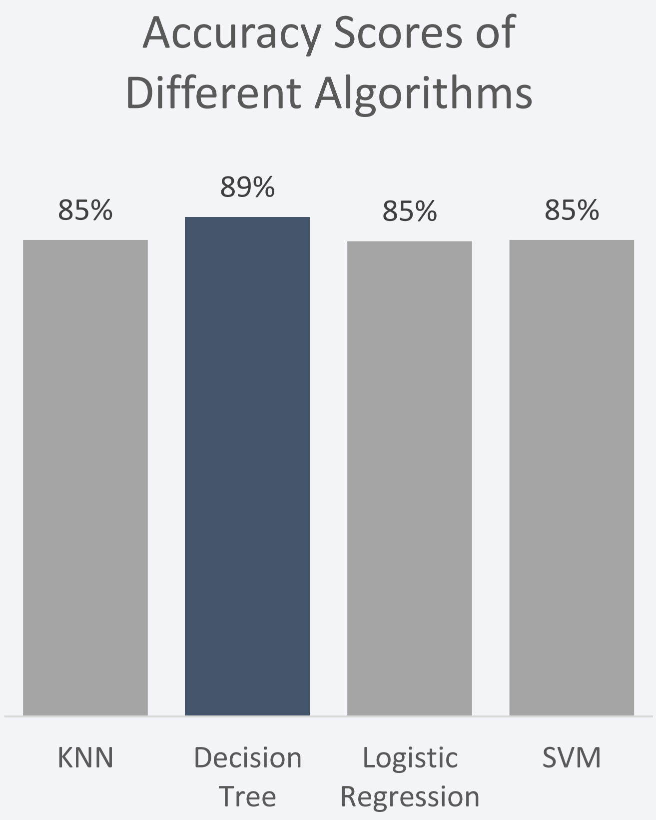
Section 5

# Predictive Analysis (Classification)

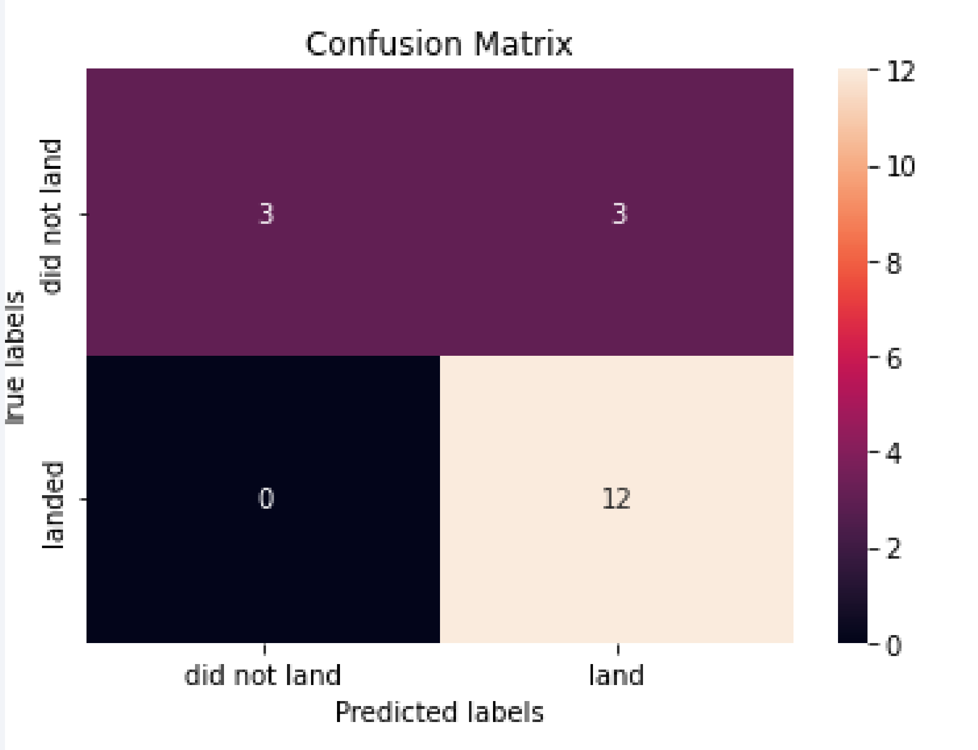
# Classification Accuracy

- As marked on a side plot, the Decision Tree was the best performer between the analyzed algorithms.

| | Accuracy |
|---|---|
| KNN | 0.848214 |
| DTree | 0.889286 |
| LR | 0.846429 |
| SVM | 0.848214 |



# Confusion Matrix




At the decision tree model we have:

- We have 0 false negatives
- And 3 false positives, which is the biggest problem of this model.

# Conclusions

- We can see that KSC LC-39A had the most successful launches from all the sites
- Orbit GEO,HEO,SSO,ES-L1 has the best Success Rate
- Low weighted payloads perform better than the heavier payloads
- The improving success rate over the years indicate a learning curve.
- The decision Tree Classifier Algorithm is the best for Machine Learning for this dataset after tunning the hyperparameters.

# Thank you!