IBM Developer
SKILLS NETWORK

# Winning Space Race with Data Science

Giovanni
Oct 17, 2021

# Outline

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

# Executive Summary

- Summary of methodologies:
  - We collect the data with the Spacex API to determine if Spacex will attempt to land the rocket or not.
  - We wrangle the data to make it useful for further analysis.
  - Explore the data and construct a dashboard to visualize the data.
  - Finally, train many models to predict whether Spacex will reuse the rocket.
- Summary of all results
  - We can predict with ~85% accuracy.

# Introduction

- Commercial space age is here! Spacex is perhaps the most successful because the rocket launches are relatively inexpensive since they can reuse their rockets.
- Problems you want to find answers:
  - Can we determine whether a Spacex launch will land?
  - Determine the price of each Spacex launch.
  - Will Spacex reuse the first stage?

Section 1

# Methodology

# Methodology

## Executive Summary

- Data collection methodology:
  - With the API
- Perform data wrangling
  - Using Pandas dataframes
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - Tried Logistic Regression, SVMs, Decision Trees, KNN

# Data Collection

- Describe how data sets were collected.
  - Using the Spacex REST API
- You need to present your data collection process use key phrases and flowcharts
  - Requested and parsed the Spacex launch data using the GET request
  - Filtered the dataframe to only include Falcon 9 launches
  - Dealt with Null values

# Data Collection – SpaceX API

- Requested and parsed the Spacex launch data using the GET request
- Filtered the dataframe to only include Falcon 9 launches
- Dealt with Null values

- Added notebook to Github

GET Request API

↓

Filtered Data

↓

Cleaned Data

# Data Collection - Scraping

- Requested the Falcon9 Launch wiki from its URL
- Extracted all columns from the html header
- Created a dataframe by parsing the launch html tables
- Added Notebook to Github

Request Launch Wiki

↓

Extract column header

↓

Created dataframe

# Data Wrangling

- Calculated the number of launches on each site
- Calculated the number of occurrences of each orbit
- Calculated the number of occurrences of mission outcomes per orbit type
- Created a land outcome label from the Outcome column

- Added Notebook to Github

# EDA with Data Visualization

- Visualized the relationship between Flight Number and Launch Site
- Visualized the relationship between Payload and Launch Site
- Visualized the relationship between success rate and each orbit type
- Visualized the relationship between Flight Number and Orbit Type
- Visualized the relationship between Payload and Orbit type
- Visualized the launch success yearly trend
- Created dummy variables of the categorical variables
- Casted all numeric variables to float64

- Added Notebook to Github

# EDA with SQL

- Connected to the database
- Queried the database
- Added Notebook to Github

# Build an Interactive Map with Folium

- Marked all launch sites on the map
- Marked success/failed launches for each site on the map
- Calculated the distance between a launch site to its proximities
- Added those objects to be able to visualize them better on the map.

- Added Notebook to Github

# Build a Dashboard with Plotly Dash

- Summarize what plots/graphs and interactions you have added to a dashboard
- Explain why you added those plots and interactions
- Add the GitHub URL of your completed Plotly Dash lab, as an external reference and peer-review purpose

# Predictive Analysis (Classification)

- Loaded the libraries and built the dataframe.
- Split into train/test sets.
- Trained a Logisitc Regression, Support Vector Machine, Decision Tree and KNN models.

- Added Notebook to Github

# Results

- Exploratory data analysis results
  - Different launch sites have different success rates.
  - VAFB-SLC launchsite there are no rockets launched for heavypayload mass
  - Success rate since 2013 keeps going up.

- After applying all the models, decision trees performed the best.

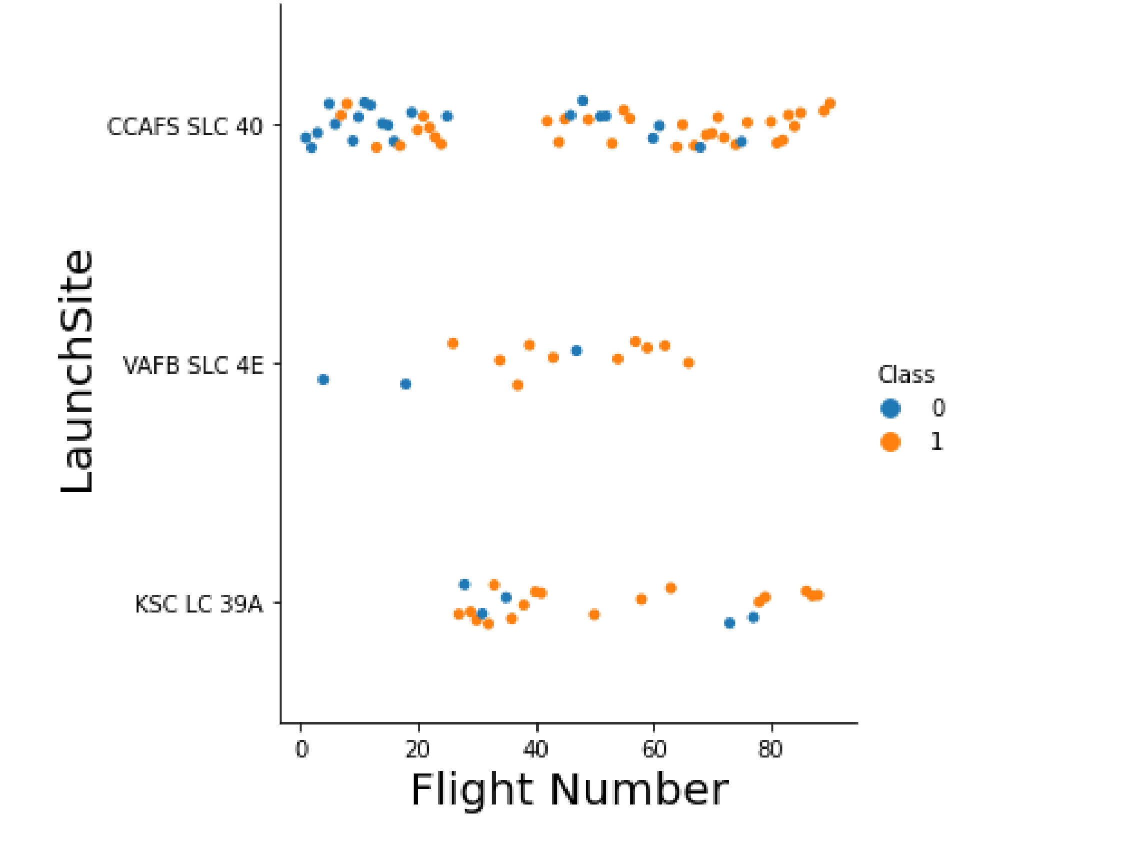
Section 2

# Insights drawn from EDA

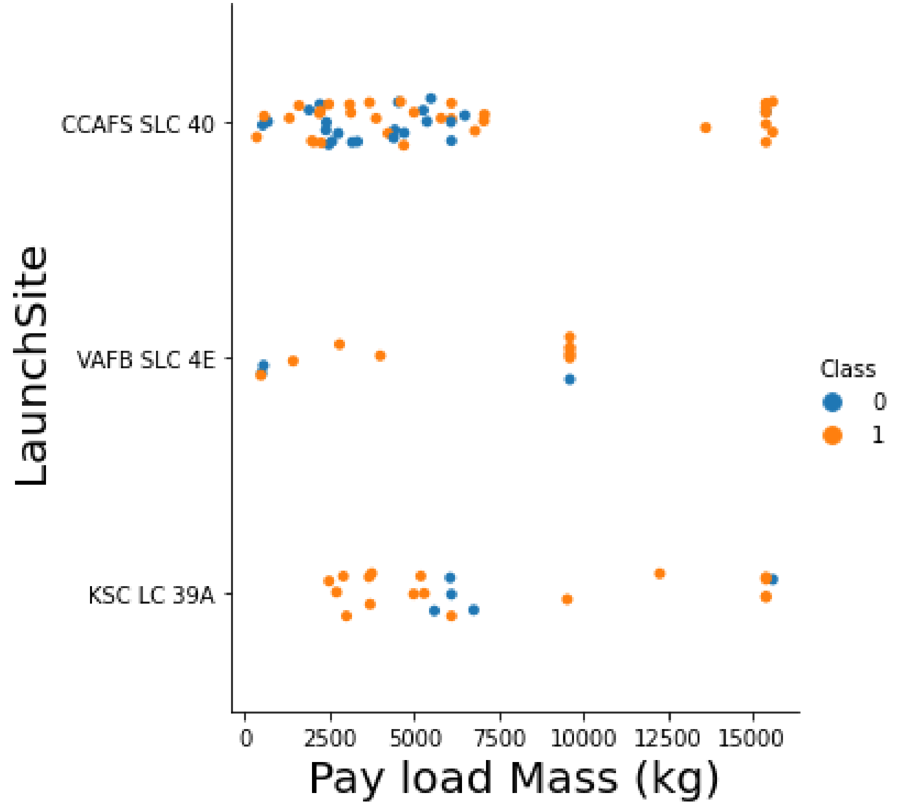
# Flight Number vs. Launch Site

- Show a scatter plot of Flight Number vs. Launch Site

- We see clustering depending on the launch site.



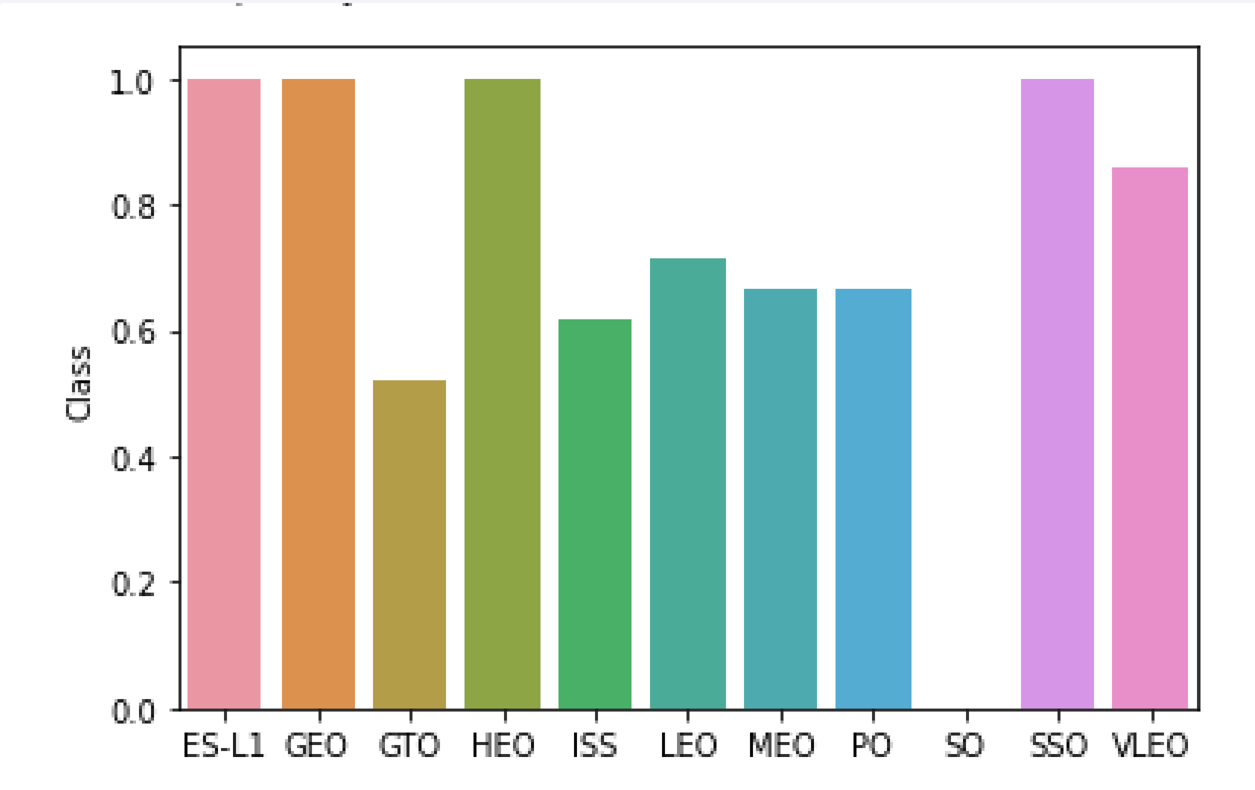
# Payload vs. Launch Site



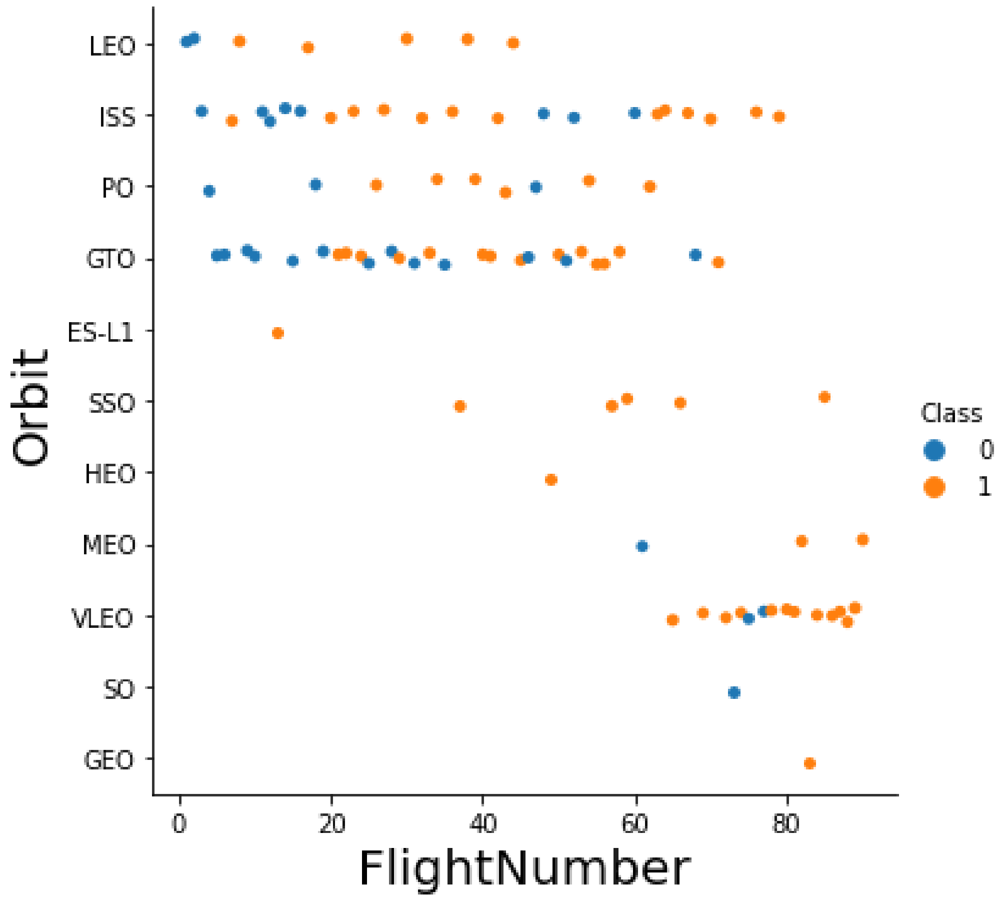

Now if you observe Payload Vs. Launch Site scatter point chart you will find for the VAFB-SLC launchsite there are no rockets launched for heavypayload mass(greater than 10000).

# Success Rate vs. Orbit Type

- ES-L1, GEO, HEO, SSO are most succesful launch sites.
- GTO not so much.

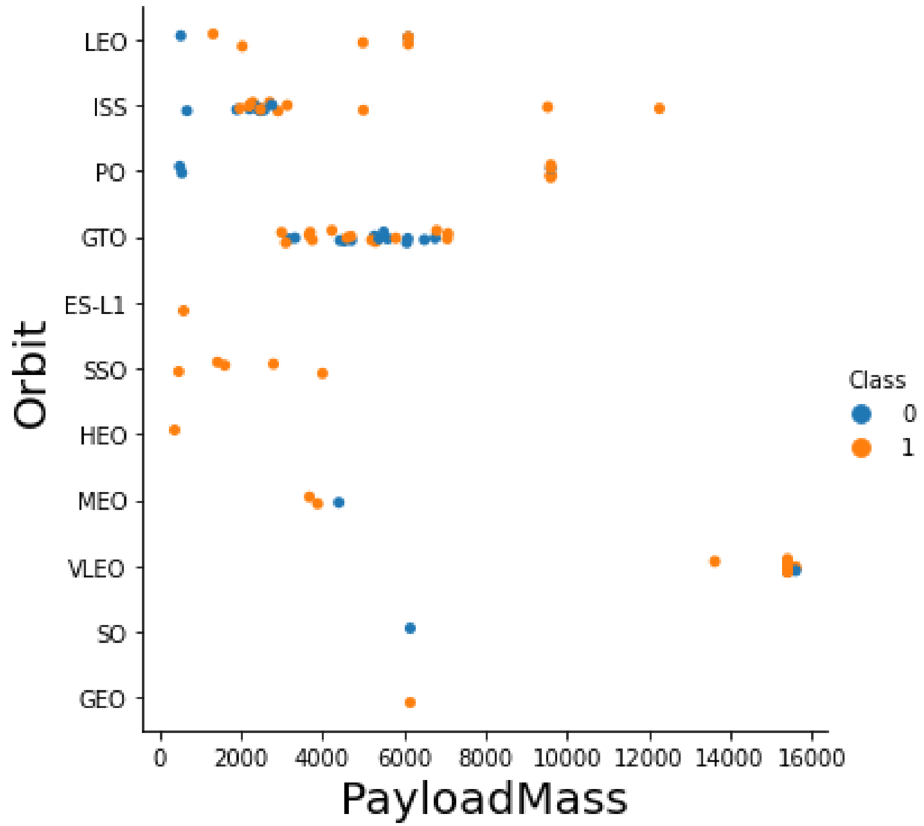


# Flight Number vs. Orbit Type




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.

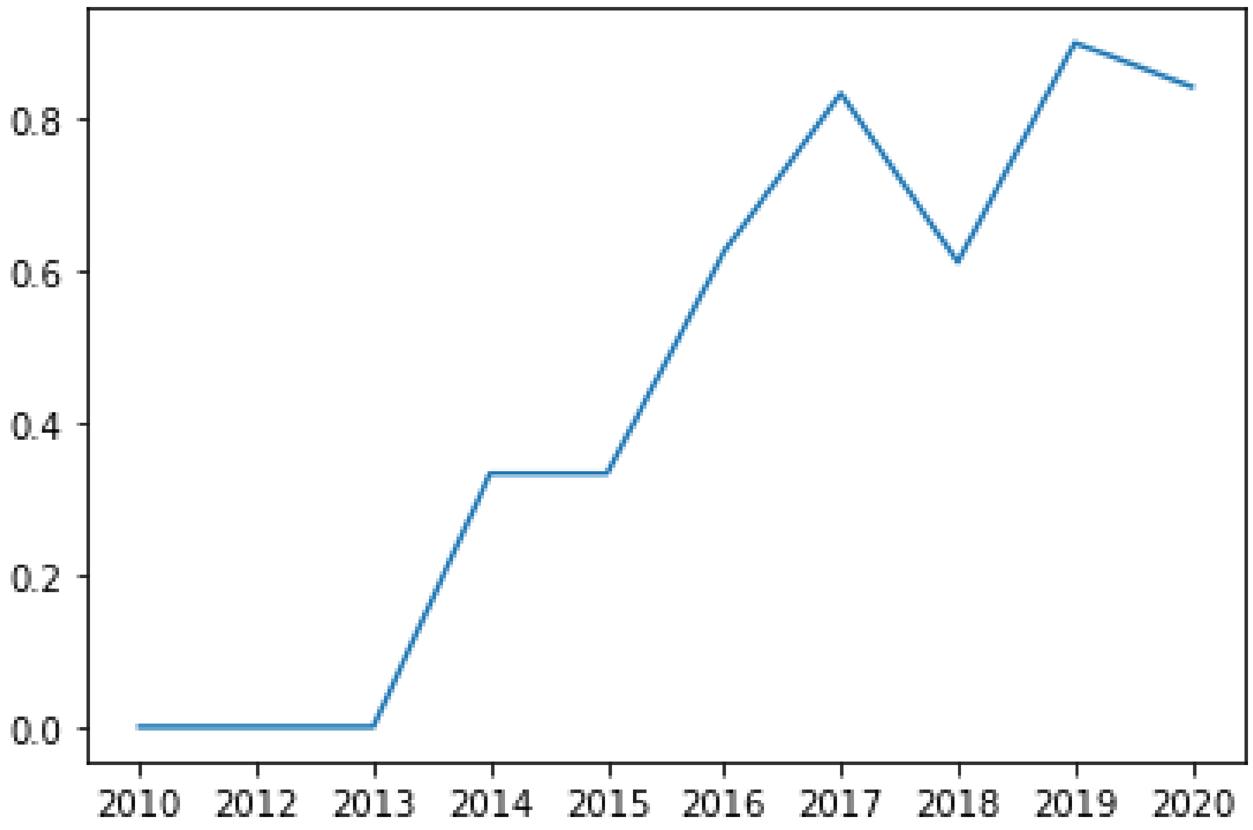
# Payload vs. Orbit Type




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.

# Launch Success Yearly Trend




you can observe that the sucess rate since 2013 kept increasing till 2020

# All Launch Site Names

- Find the names of the unique launch sites
  - Unique launch sites are CCAFS SLC 40, KSC LC 39A, VAFB SLC 4E

```
[11]: np.unique(features.LaunchSite)

[11]: array(['CCAFS SLC 40', 'KSC LC 39A', 'VAFB SLC 4E'], dtype=object)
```

# Launch Site Names Begin with 'CCA'

- Find 5 records where launch sites begin with `CCA`

```
features[features['LaunchSite'] == 'CCAFS SLC 40']
```

| | FlightNumber | PayloadMass | Orbit | LaunchSite | Flights | GridFins | Reused | Legs | LandingPad | Block | ReusedCount | Serial |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 6104.959412 | LEO | CCAFS SLC 40 | 1 | False | False | False | NaN | 1.0 | 0 | B0003 |
| 1 | 2 | 525.000000 | LEO | CCAFS SLC 40 | 1 | False | False | False | NaN | 1.0 | 0 | B0005 |
| 2 | 3 | 677.000000 | ISS | CCAFS SLC 40 | 1 | False | False | False | NaN | 1.0 | 0 | B0007 |
| 4 | 5 | 3170.000000 | GTO | CCAFS SLC 40 | 1 | False | False | False | NaN | 1.0 | 0 | B1004 |
| 5 | 6 | 3325.000000 | GTO | CCAFS SLC 40 | 1 | False | False | False | NaN | 1.0 | 0 | B1005 |

# Total Payload Mass

- Calculate the total payload carried by boosters from NASA

```
features.PayloadMass.sum()
```

```
549446.3470588236
```

# Average Payload Mass by F9 v1.1

- Calculate the average payload mass carried by booster version F9 v1.1
  - df was already filtered for Falcon9 rocket.

```
df.PayloadMass.mean()
```

```
6104.959411764707
```

# First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

```
np.min(df.Date[df.Class == 1])
```

```
'2014-04-18'
```

# Successful Drone Ship Landing with Payload between 4000 and 6000

- List the names of boosters which have successfully landed on drone ship and had payload mass greater than 4000 but less than 6000

- Present your query result with a short explanation here

# Total Number of Successful and Failure Mission Outcomes

- Calculate the total number of successful and failure mission outcomes
  - 60 successful missions and 30 failed missions.

```
np.sum(df.Class)
```

```
60
```

```
np.sum(1-df.Class)
```

```
30
```

# Boosters Carried Maximum Payload

- List the names of the booster which have carried the maximum payload mass
- Falcon9

# 2015 Launch Records

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

- Present your query result with a short explanation here

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

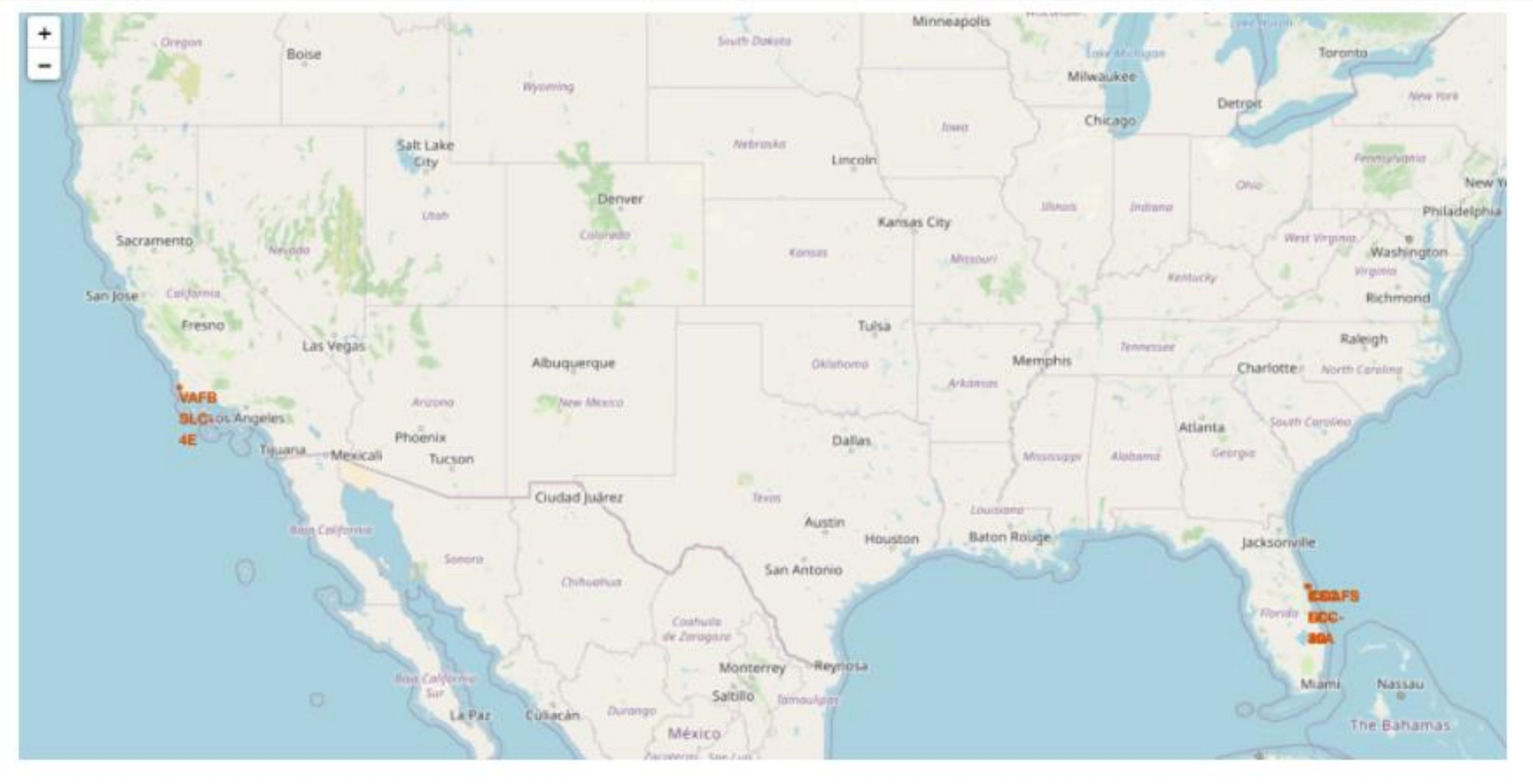
- Rank the count of landing outcomes (such as Failure (drone ship) or Success (ground pad)) between the date 2010-06-04 and 2017-03-20, in descending order

- Present your query result with a short explanation here

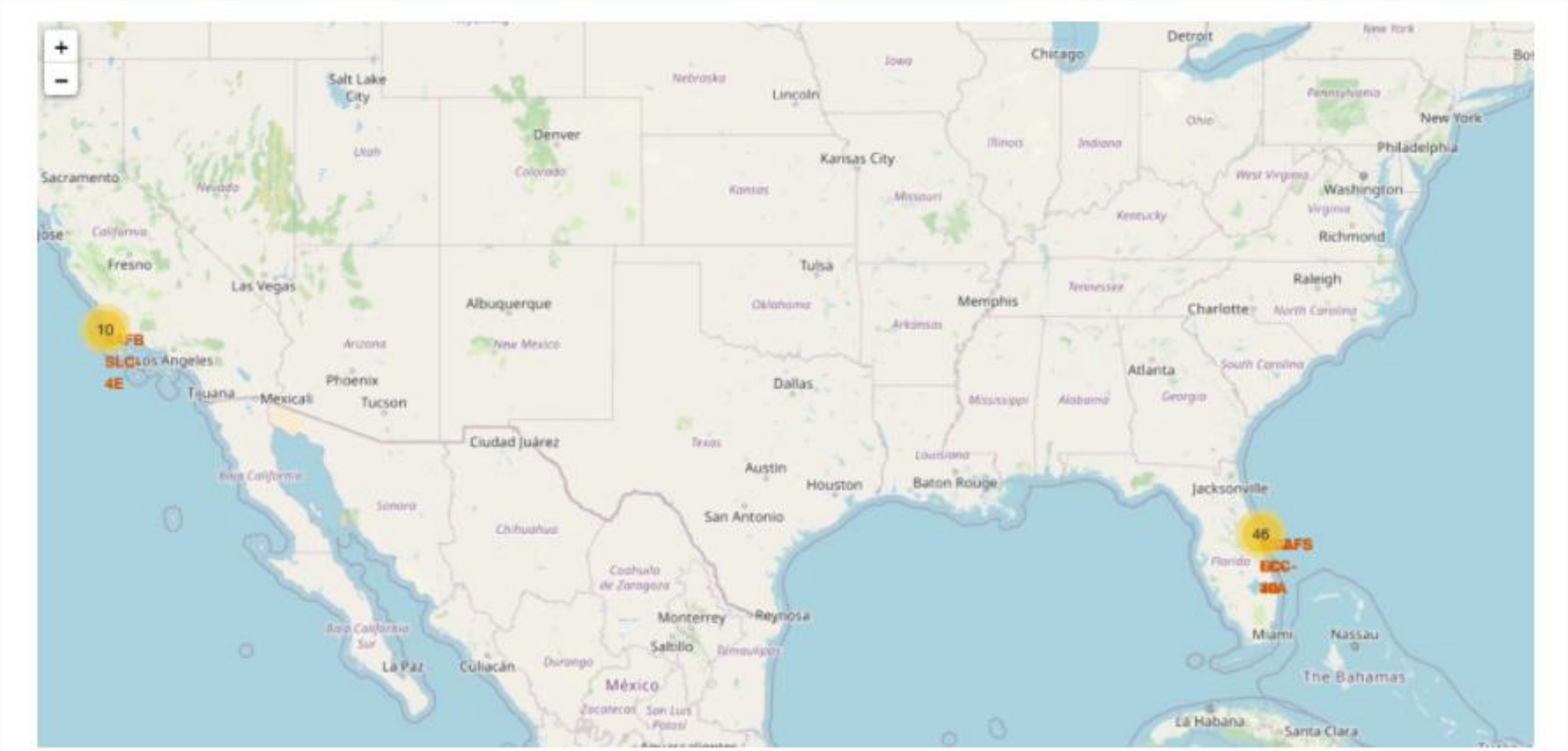
Section 4

# Launch Sites Proximities Analysis

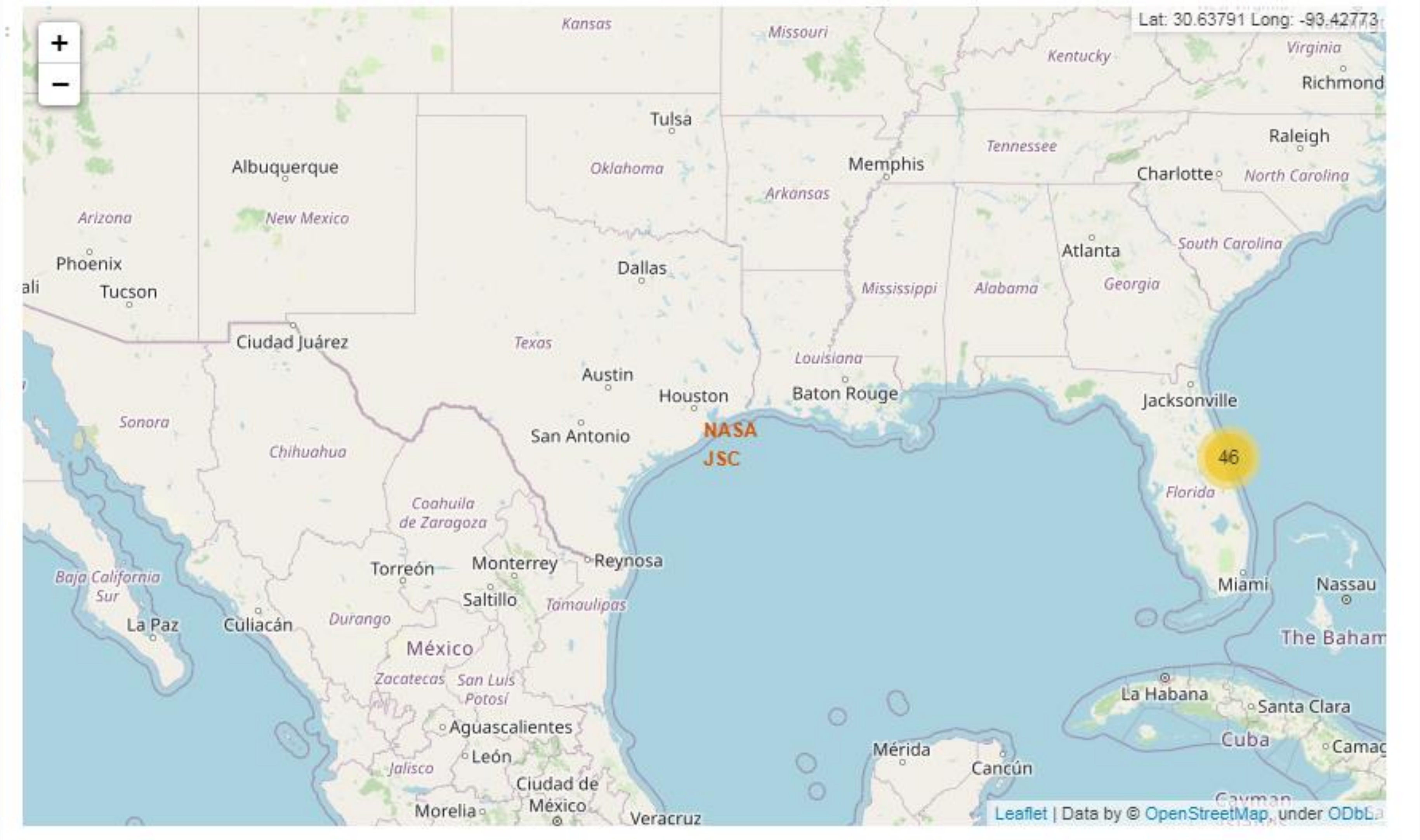
# Mark all Launch Sites



# Mark success/failed launches at each site



# Calculate distance between launch site and proximities



Section 5

# Build a Dashboard with Plotly Dash

# <Dashboard Screenshot 1>

- Replace <Dashboard screenshot 1> title with an appropriate title

- Show the screenshot of launch success count for all sites, in a piechart

- Explain the important elements and findings on the screenshot

# <Dashboard Screenshot 2>

- Replace <Dashboard screenshot 2> title with an appropriate title

- Show the screenshot of the piechart for the launch site with highest launch success ratio

- Explain the important elements and findings on the screenshot

# <Dashboard Screenshot 3>

- Replace <Dashboard screenshot 3> title with an appropriate title

- Show screenshots of Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

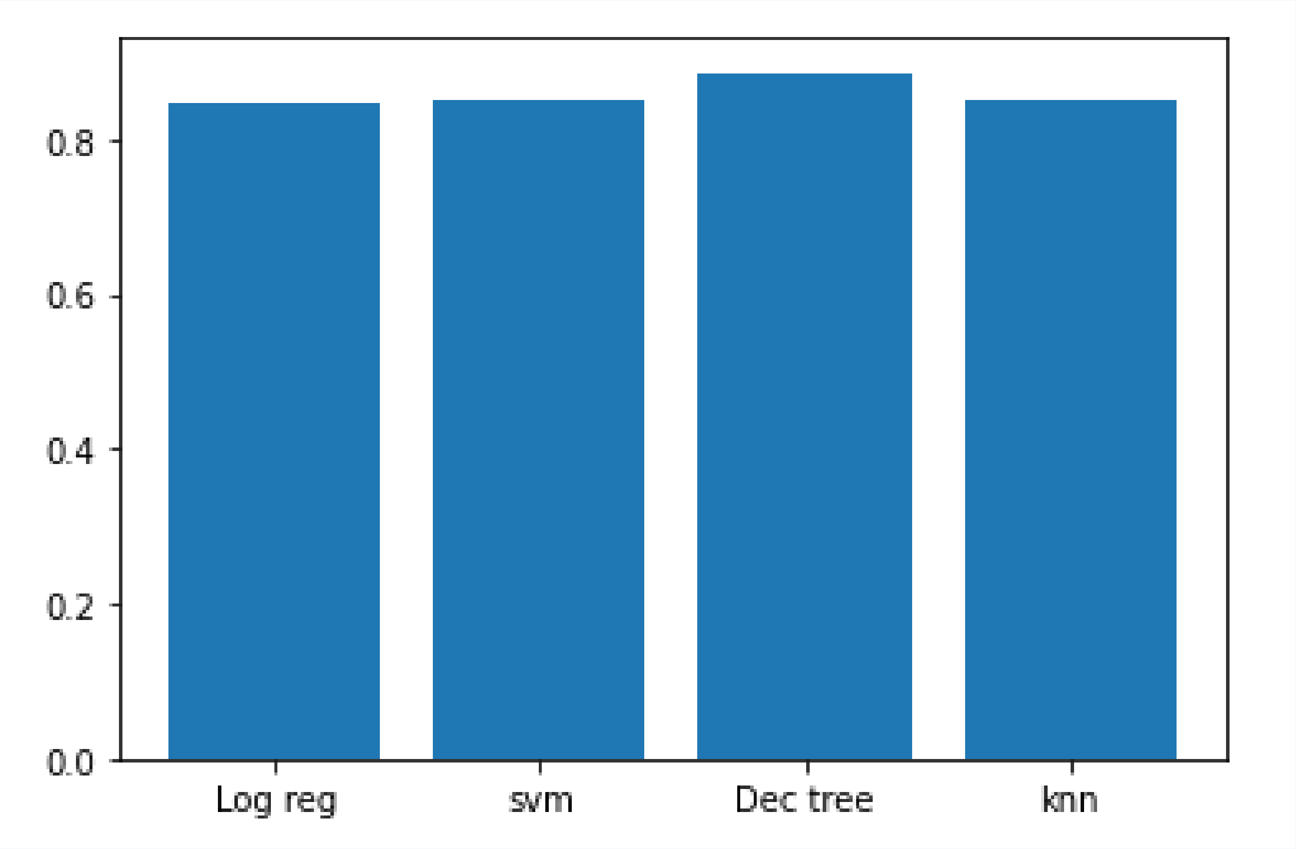
- Explain the important elements and findings on the screenshot, such as which payload range or booster version have the largest success rate, etc.

Section 6

# Predictive Analysis (Classification)

# Classification Accuracy

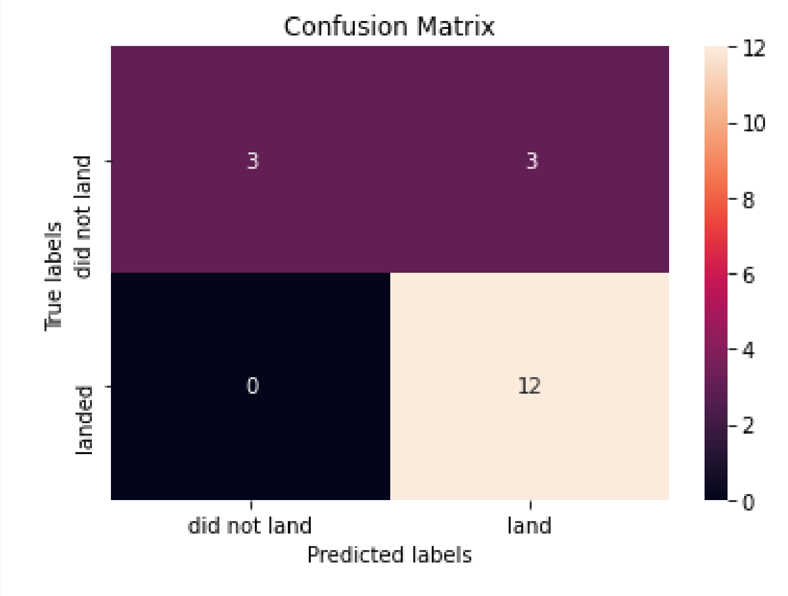
- Visualize the built model accuracy for a
  built classification models, in a bar char
- Find which model has the highest classification accuracy
  - Decision tree has the highest classification accuracy.



# Confusion Matrix

- Show the confusion matrix of the best performing model with an explanation

- Classifies most correctly, with some mistakes in the top right hand quadrant.



# Conclusions

- Decision tree is best on the train set
- They all perform the same on the test set

# Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project
    - All on github

# Thank you!