IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

<Tracey Centorbi>
<September 6, 2021>

Outline

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

Executive Summary

- Summary of Methodologies
  - Data collection and Data wrangling
  - EDA with data visualization
  - EDA with SQL
  - Interactive map with Folium
  - Plotly Dash dashboard
  - Predictive analysis with logistic regression, SVM, decision tree, and KNN
- Summary of all results
  - Exploratory data analysis results
  - Interactive analytics demonstration in screenshots
  - Predictive analysis results

Introduction

- Project background and context
  - We predicted whether the Falcon 9 first stage will land successfully based on a number of factors. This is important because if SpaceX can reuse the first stage, they can realize significant savings. On the SpaceX website, the cost per Falcon 9 launch can be as low as $62 million compared to a cost of $165 for their competitors. Therefore, if we can determine if the first stage will land successfully, we can more accurately predict the cost of a launch.
- Problems to address with this project
  - Which factors are most correlated with a successful rocket landing?
    - These factors might include launch site, payload, intended orbit, landing platform, etc.
  - Are there relationships between different variables that will affect the success rate of landings?
  - Are there factors that are more useful in predicting a successful rocket landing?

Section 1

Methodology

Methodology

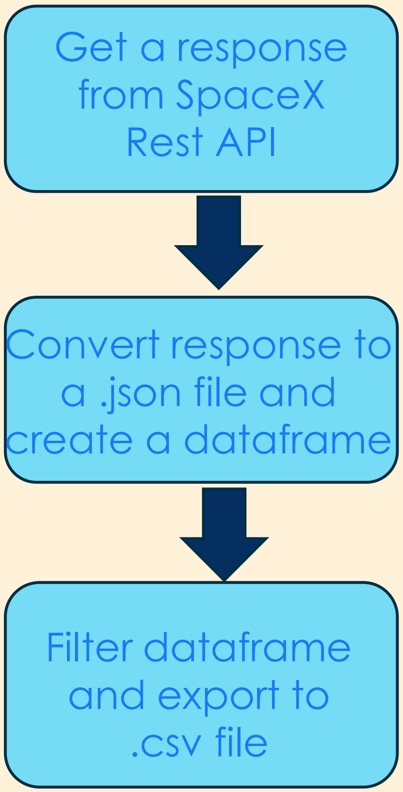
Executive Summary

- Data collection methodology:
  - SpaceX Rest API
- Perform data wrangling
  - One Hot Encoding for machine learning
- Perform exploratory data analysis (EDA) using visualization and SQL
  - Scatter and Bar graphs to show data relationships
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection – SpaceX API

- The dataset was collected by:
  - Define a series of helper functions
  - Request rocket launch data from SpaceX API
  - Parse and decode the response content as a Json
  - Turn the response content into a Pandas dataframe
  - Requests more details of the data to be stored as a list and used to create a new dataframe
  - Combine the data into a dictionary and use it to create a Pandas dataframe
  - Filter the dataframe to include only some rocket data
  - Assess and manage missing data

Data Collection – SpaceX API




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

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

```
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

```
df = pd.DataFrame.from_dict(launch_dict)
```

```
data_falcon9 = df.loc[df['BoosterVersion']!="Falcon 1"]
```

```
data_falcon9.loc[:,'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
```


https://github.com/tracento/IBM-DataScience-Capstone-SpaceX/blob/master/SpaceX%20Data%20Collection%20API%20project.ipynb

Data Wrangling

- The dataset was cleaned by:
  - Identifying missing values
  - Identifying data types
  - Counting launch types, orbits, and outcomes
  - Representing the outcome with a new class

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

```
df.dtypes
```

```
df["Orbit"].value_counts()
```

```
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```


https://github.com/tracento/IBM-DataScience-Capstone-SpaceX/blob/master/SpaceX%20Data%20Wrangling%20Project.ipynb

EDA with Data Visualization

- Scatter Plots show how much 1 variable is affected by another variable shown as a correlation:
  - Flight Number vs. Payload Mass
  - Flight Number vs. Launch Site
  - Payload Mass vs. Launch Site
  - Flight Number vs. Orbit
  - Payload Mass vs. Orbit
- Bar Plots to compare sets of data between different groups at a glance:
  - Orbit vs. Success Rate
- Line Plot to show variables and trends over time:
  - Year vs. Success Rate

https://github.com/tracento/IBM-DataScience-Capstone-SpaceX/blob/master/EDA%20with%20Visualization.ipynb

EDA with SQL

- Performed SQL queries to gather the following information:
  - Distinct launch sites
  - 5 records for launch sites with the string 'CCA”
  - Total payload mass for boosters launched by NASA
  - Average payload mass carried by Falcon 9 v1.1 boosters
  - Date of first successful ground pad landing
  - Boosters which have successfully landed on drone ships in a specific payload range
  - Total number of successful and failed mission outcomes
  - Boosters which carried the maximum payload
  - Details of records for 2015
  - Successful outcomes from 2010-06-04 and 2017-03-20

```
%sql select * from SPACEXTBL where Launch_Site like 'CCA%' \
fetch first 5 rows only
```


https://github.com/tracento/IBM-DataScience-Capstone-SpaceX/blob/master/EDA%20with%20SQL%20project.ipynb

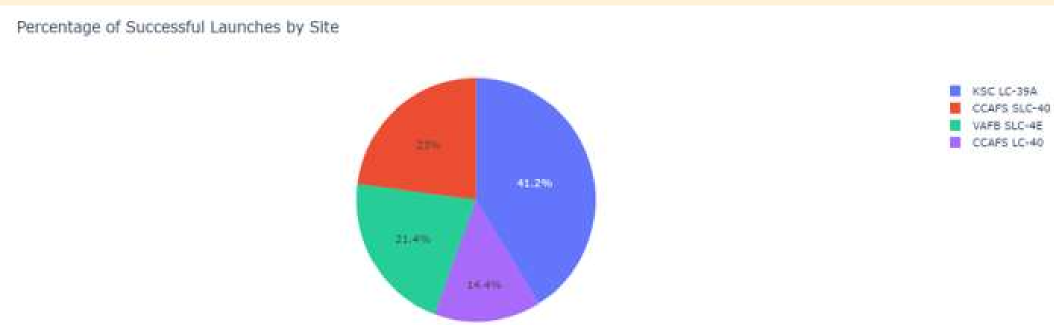
Build an Interactive Map with Folium

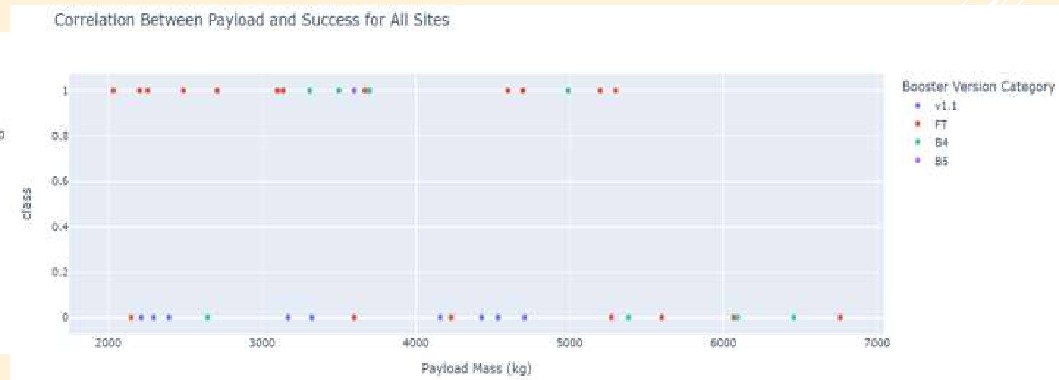
- Built an interactive map with Folium to visualize the following information:
  - Launch site locations and coordinates
    - Added a circle marker around each site and name label
  - Assign launch outcomes in a Marker Cluster
    - Failures and successes marked with red and green markers
  - Calculated the distance from launch site to nearest railway and other features
    - Lines and distances marked for each instance
- Folium features allowed us to answer the following questions:
  - 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

https://github.com/tracento/IBM-DataScience-Capstone-SpaceX/blob/master/Interactive%20Visual%20Analytics%20project.ipynb

Build a Dashboard with Plotly Dash

- Built an interactive Dashboard with Plotly Dash to visualize the following information:
  - A dropdown menu to select different launch sites
  - A pie chart to visualize launch success counts for different launch sites
  - A range slider to select payload mass to assess the success of different size payloads
  - A scatter plot with payload mass vs. launch outcome and booster version to see if there is a pattern





https://labs.cognitiveclass.ai/tools/theiadocker/?md_instructions_url=https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN SkillsNetwork/labs/module_3/lab_theia_plotly_dash.md&lti=true

Predictive Analysis (Classification)

- Built a predictive analysis model to evaluate launch data:

```
y = data['Class'].to_numpy()
```

  - Create a NumPy array and standardize the data

```
transform = preprocessing.StandardScaler()
X = transform.fit_transform(X)
```

  - Split the data into training and testing data
  - Created a *logistic regression* object

```
gscv = GridSearchCV(lr, parameters, scoring='accuracy', cv=10)
logreg_cv = gscv.fit(X_train, Y_train)
```

  - Created a Grid Search object and tuned parameters

```
print("tuned hpyerparameters :(best parameters) ",logreg_cv.best_params_)
print("accuracy :",logreg_cv.best_score_)
```

  - Calculated accuracy on test data

```
print('Accuracy= ', logreg_cv.score(X_test, Y_test))
```

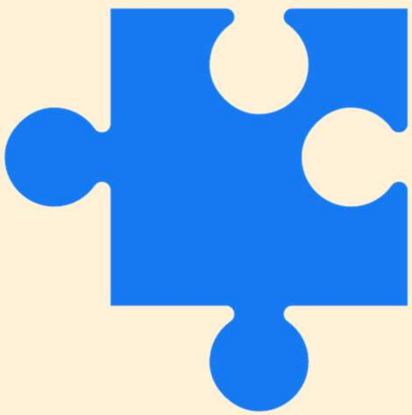
  - Created a confusion matrix to distinguish between classes

```
yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```

  - Repeated these steps for *support vector machine, decision tree*, and *k nearest neighbors*
  - Created an algorithm to find which method performed best for this data

https://github.com/tracento/IBM-DataScience-Capstone-SpaceX/blob/master/Machine%20Learning%20Prediction%20project.ipynb

Results

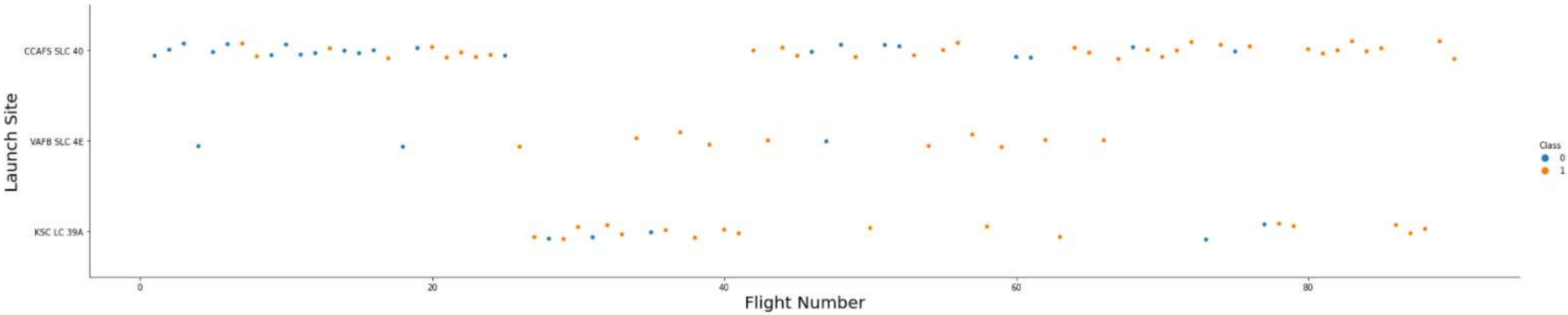


- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

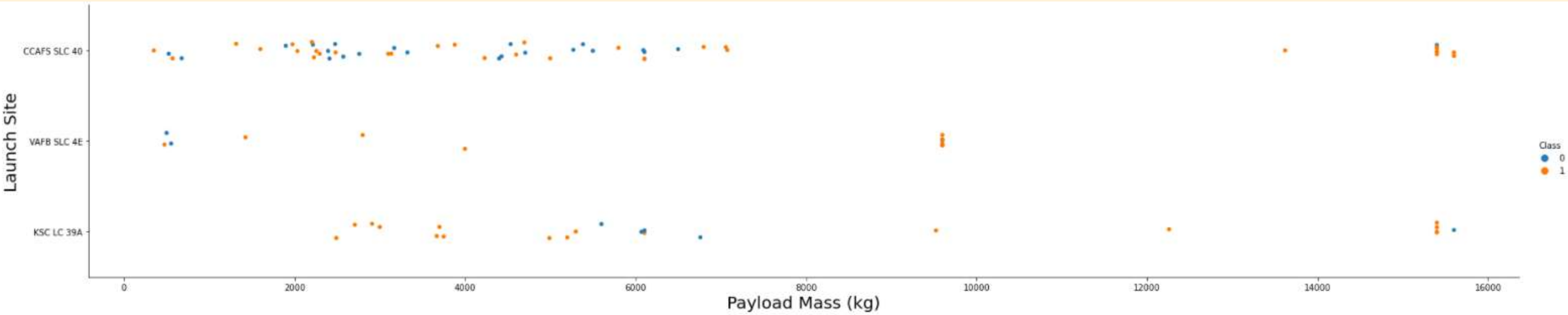
Insights drawn from EDA

Flight Number vs. Launch Site



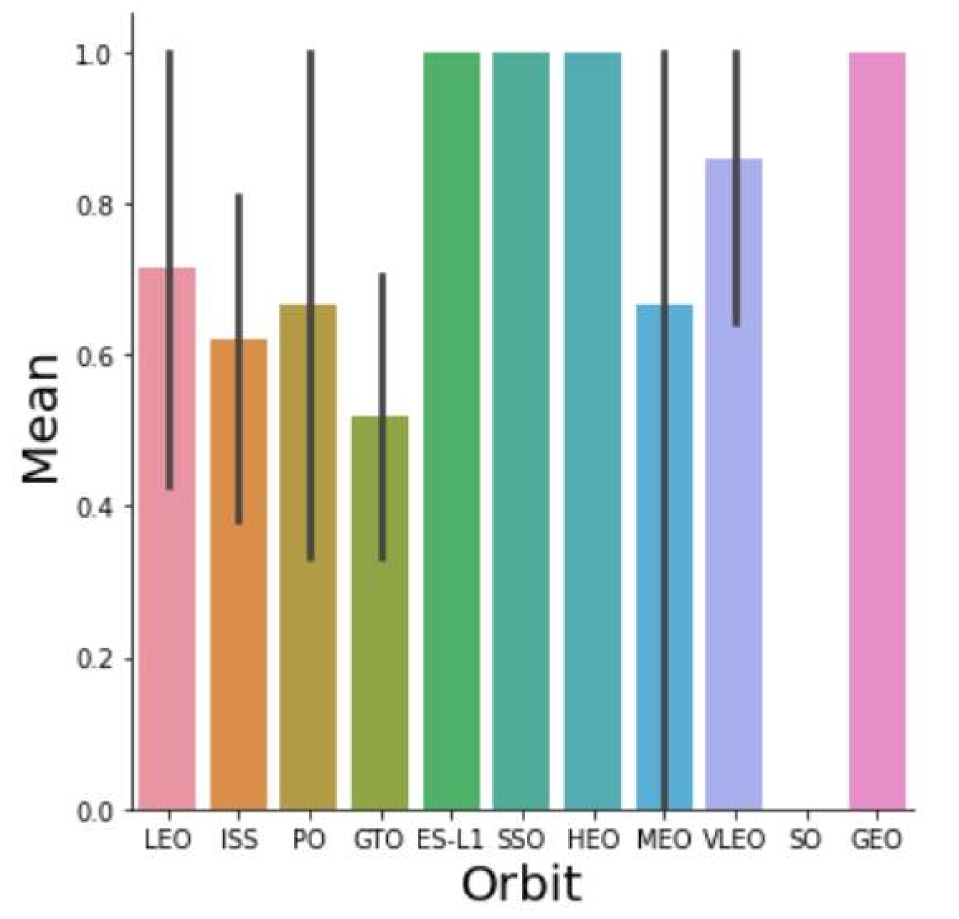
- Sites with a larger number of flights have a higher success rate

Payload vs. Launch Site



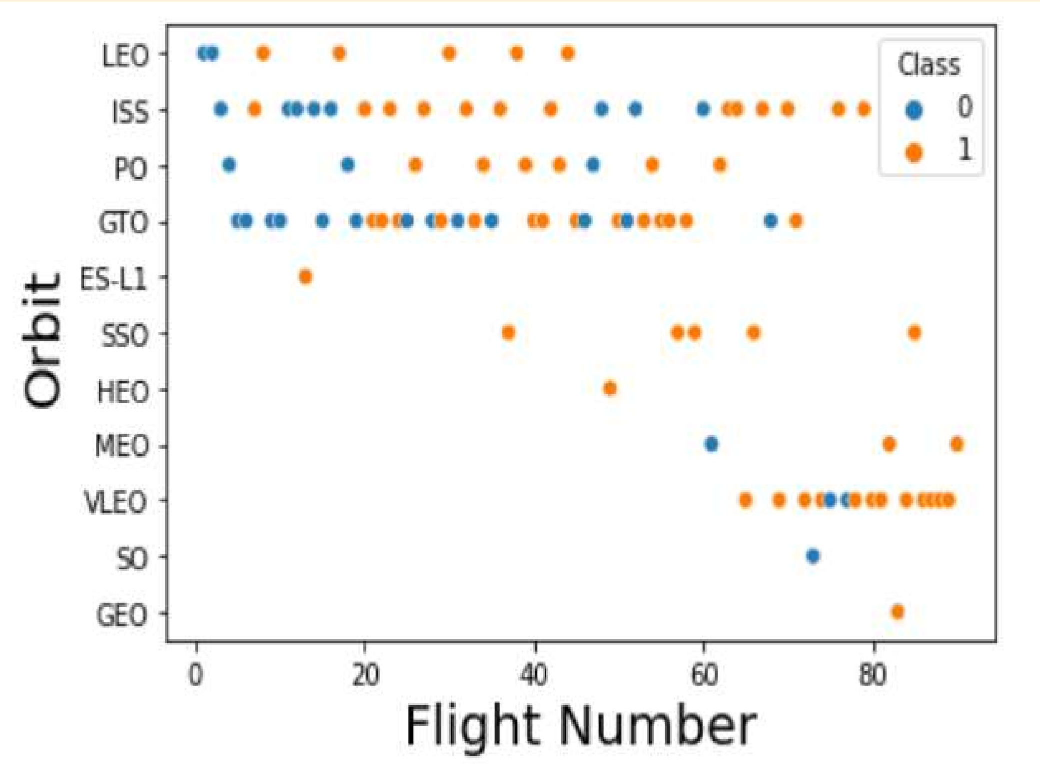
- At launch site CCAFS SLC 40, a greater payload mass correlates to a higher success.
- At other sites, there is not a clear pattern to indicate whether success is dependent on payload mass at a launch site.

Success Rate vs. Orbit Type



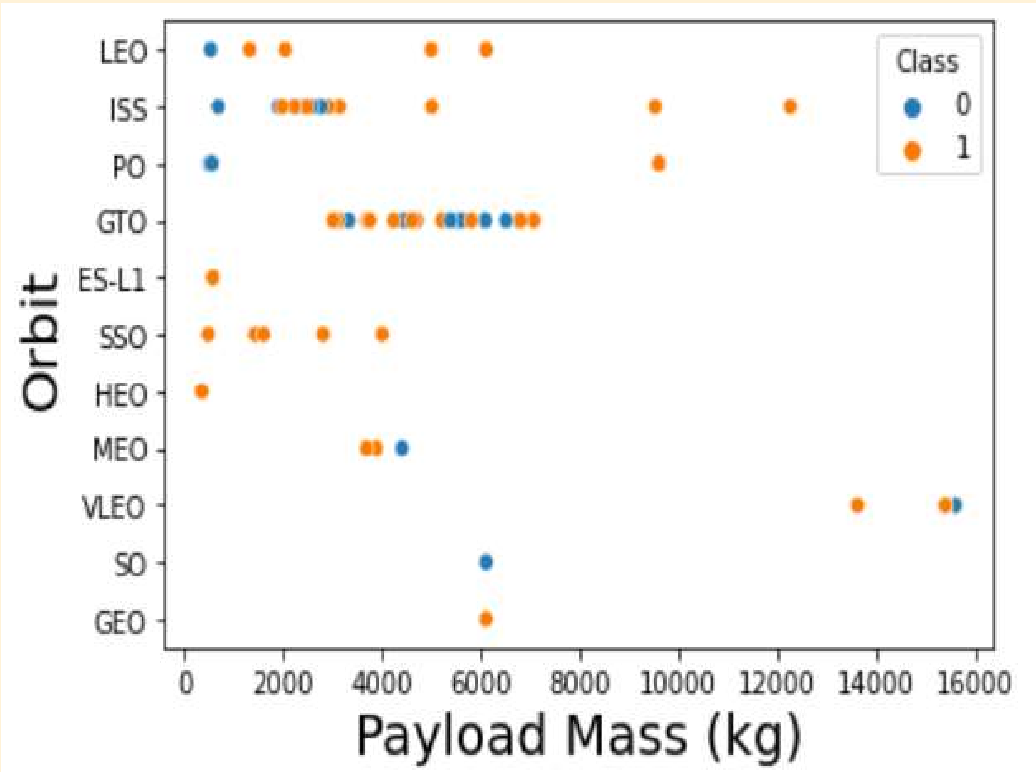
- ES-L1, SSO, HEO, and GEO are the most successful orbits.

Flight Number vs. Orbit Type



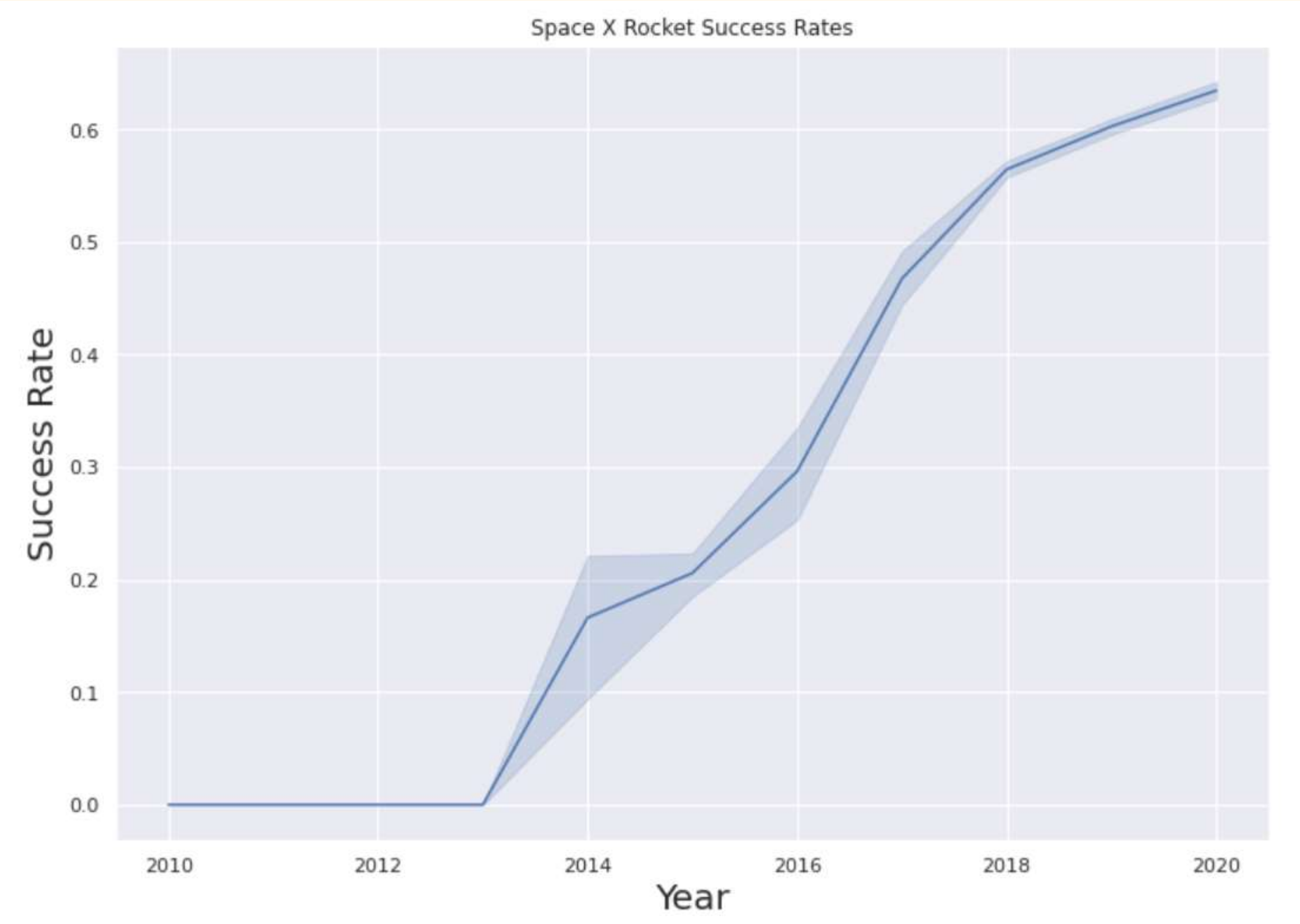
- Flight number appears to be highly correlated with success in LEO orbit.
- Flight number appears to be uncorrelated with success in GTO orbit

Payload vs. Orbit Type



- Payload mass appears to be highly correlated with success in LEO and ISS orbit.
- Payload mass appears to be uncorrelated with success in GTO, MEO, and VLEO orbits.

Launch Success Yearly Trend



- The success rate continued to increase from 2013 to 2020.

All Launch Site Names

```
%sql select DISTINCT Launch_Site from SPACEXTBL
```

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

- Using DISTINCT in the query yields only unique values

Launch Site Names Begin with 'CCA'

```
%sql select * from SPACEXTBL where Launch_Site like 'CCA%' \
fetch first 5 rows only
```

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 like keyword and the 'CCA%' means that the launch site name must start with CCA.
- Fetch first 5 rows only limits the display number of records in the output

Total Payload Mass

```
%sql select SUM(payload_mass__kg_) TotalPayloadMass from SPACEXTBL where Customer = 'NASA (CRS)'
```

totalpayloadmass
45596

- Using the SUM function calculates the total in the payload mass column
- The WHERE clause filters the dataset to only the customer NASA (CRS)

Average Payload Mass by F9 v1.1

```
%sql select AVG(payload_mass__kg_) AveragePayloadMass from SPACEXTBL where Booster_Version = 'F9 v1.1'
```

averagepayloadmass
2928.400000

- The function AVG calculates the average of the payload mass column
- The WHERE clause filters the results to only the F9 v1.1 booster

First Successful Ground Landing Date

```
%sql select MIN(Date) SuccessfulLandingOutcome from SPACEXTBL where landing__outcome ='Success (ground pad)''
```

successfullandingoutcome
2016-04-08

- Using the MIN function yields the earliest date in the Date column
- The WHERE clause filters to only return a successful landing outcome on a ground pad

Successful Drone Ship Landing with Payload between 4000 and 6000

```
%sql select booster_version from SPACEXTBL where landing__outcome = 'Success (ground pad)' AND payload_mass__kg_ > 4000 AND payload_mas
s__kg_ < 6000
```

booster_version
F9 FT B1032.1
F9 B4 B1040.1
F9 B4 B1043.1

- Selecting only booster version
- The WHERE clause filters to only landings on the ground pad
- The AND clause returns only payloads between 4000 and 6000 kg

Total Number of Successful and Failure Mission Outcomes

```
%sql select count(*) from SPACEXTBL where mission_outcome = 'Success'
```

1
99

```
%sql select count(*) from SPACEXTBL where mission_outcome = 'Failure'
```

1
0

- The COUNT function totals the number of mission outcomes in each class
- The WHERE clause filters to only the desired mission outcome

Boosters Carried Maximum Payload

```
%sql SELECT DISTINCT booster_version, MAX(payload_mass__kg_) MaximumPayloadMass FROM SPACEXTBL GROUP BY booster_version ORDER BY Maximu
mPayloadMass DESC
```

booster_version	maximumpayloadmass
F9 B5 B1048.4	15600
F9 B5 B1048.5	15600
F9 B5 B1049.4	15600
F9 B5 B1049.5	15600
F9 B5 B1049.7	15600

Table truncated for brevity

- Using DISTINCT means it will only show unique booster versions
- GROUP BY with DESC puts the list on order based on maximum payload mass

2015 Launch Records

```
%sql SELECT MONTHNAME(DATE), landing__outcome, booster_version, launch_site from SPACEXTBL where landing__outcome = 'Failure (drone ship)' AND YEAR(DATE) = '2015'
```

1	landing__outcome	booster_version	launch_site
January	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
April	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

- The MONTHNAME returns the name of the month
- The WHERE and AND clauses narrow the return to only 2015 failures on the drone ship

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

```
%sql select landing__outcome, count(landing__outcome) from SPACEXTBL where landing__outcome = 'Success' group by landing__outcome order
by count(landing__outcome) desc
```

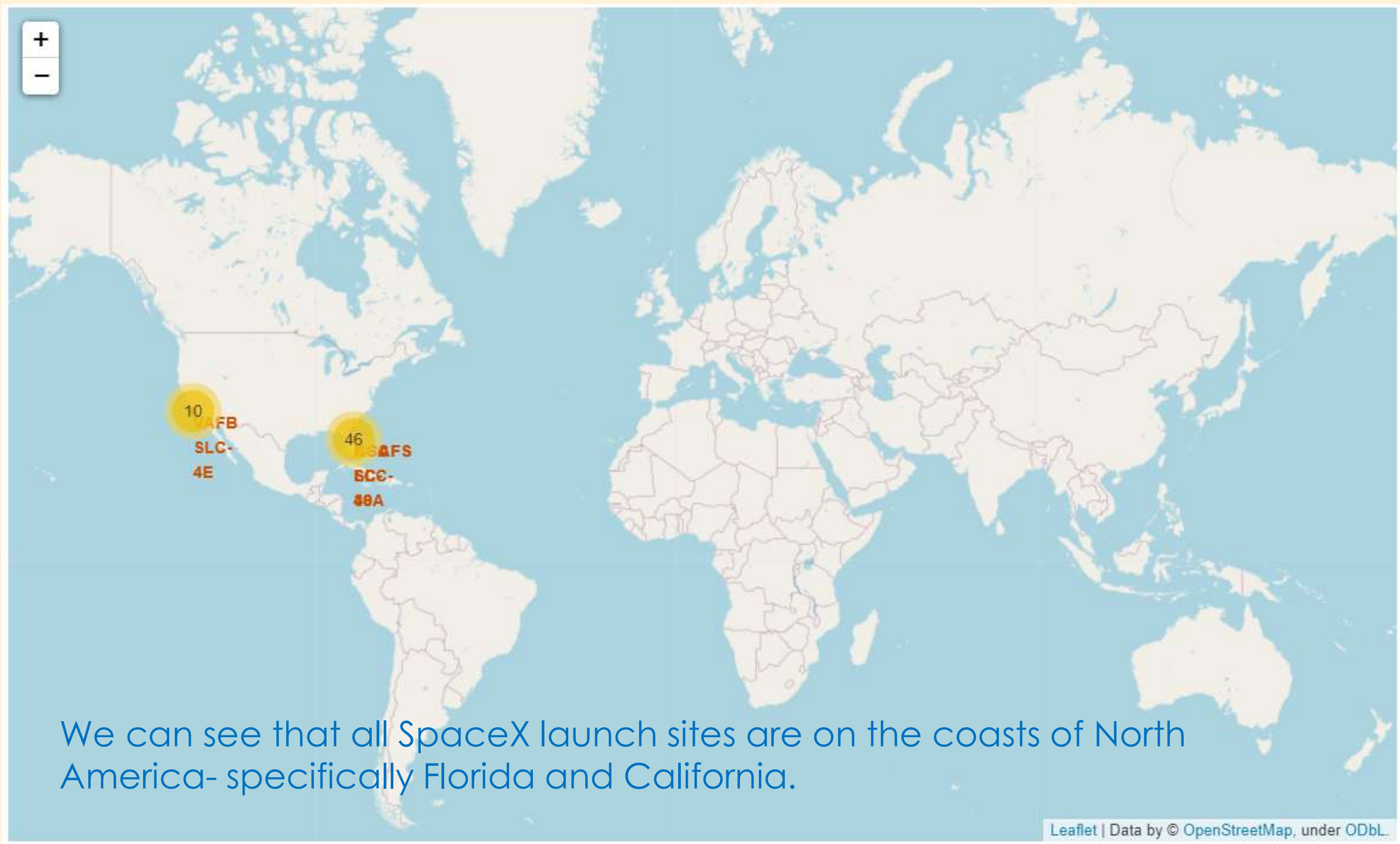
landing__outcome	2
Success	38

- The instruction in the notebook for this query was "***Rank the count of successful landing_outcomes between the date 2010-06-04 and 2017-03-20 in descending order.***"
- This yielded only 'Success' results numbering 38 total

Section 4

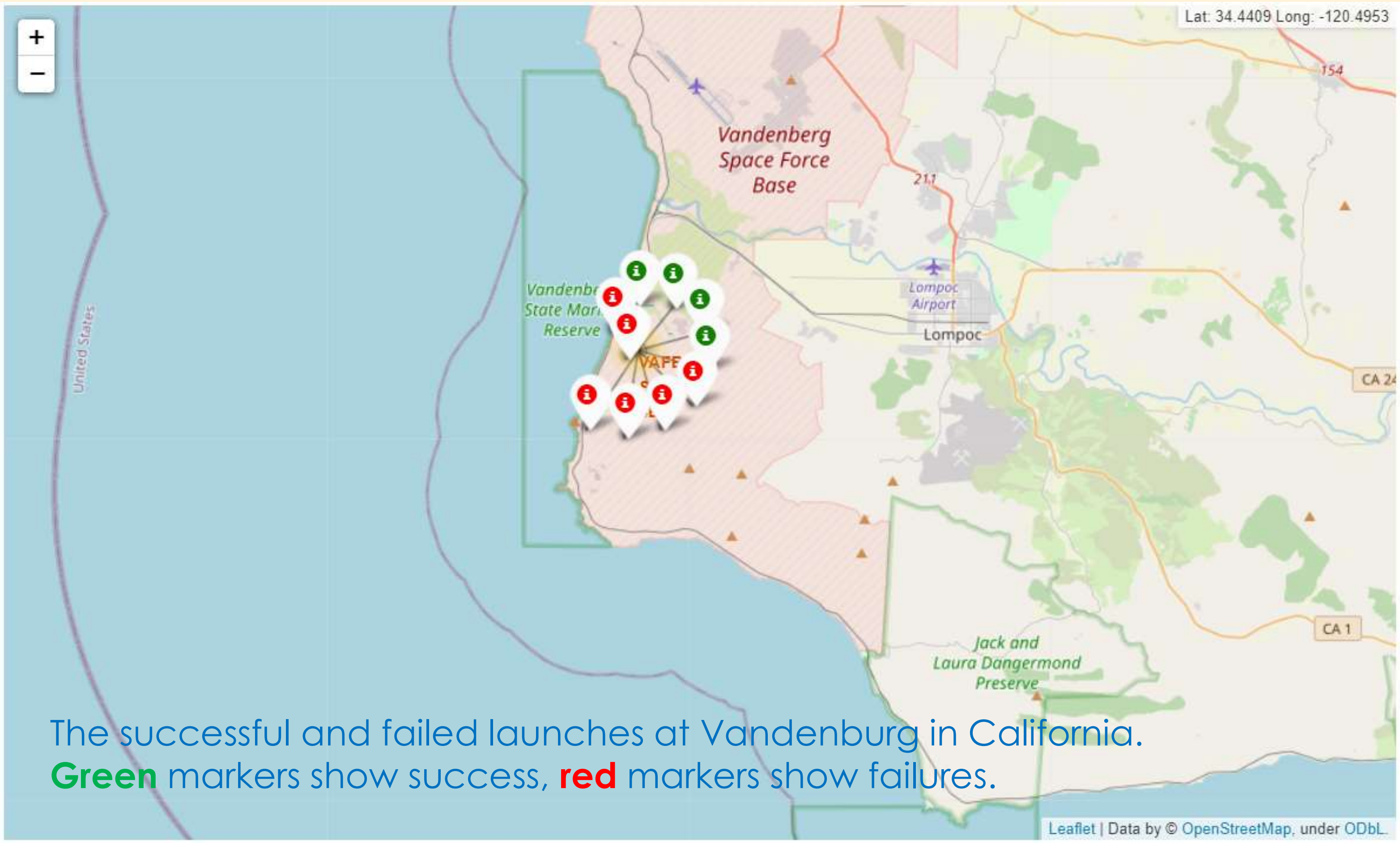
Launch Sites Proximities Analysis

Launch Site Global Map Markers



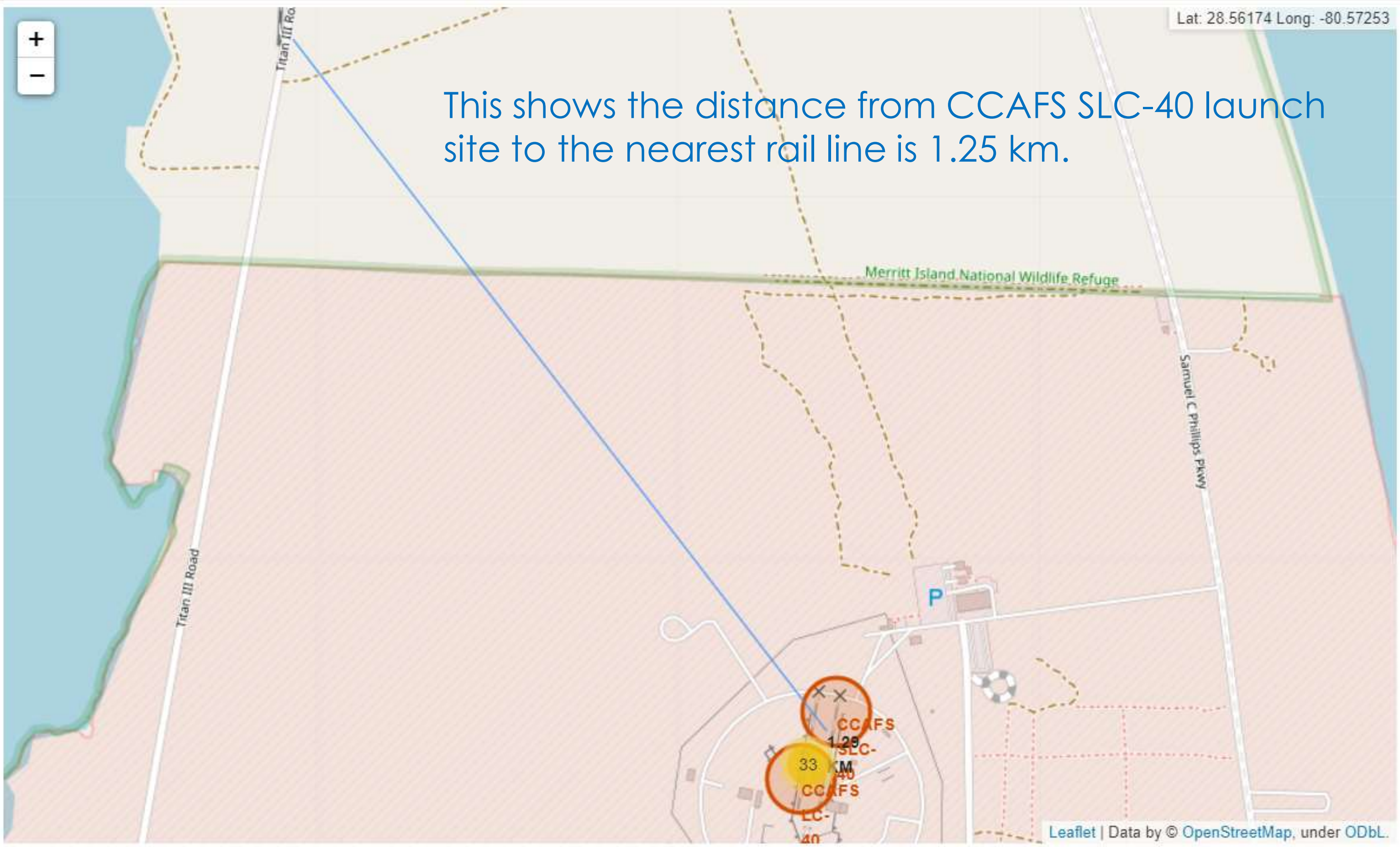
We can see that all SpaceX launch sites are on the coasts of North America- specifically Florida and California.

Launch Site Success Visualization



The successful and failed launches at Vandenburg in California.
Green markers show success, **red** markers show failures.

Launch Site Distance from Landmarks

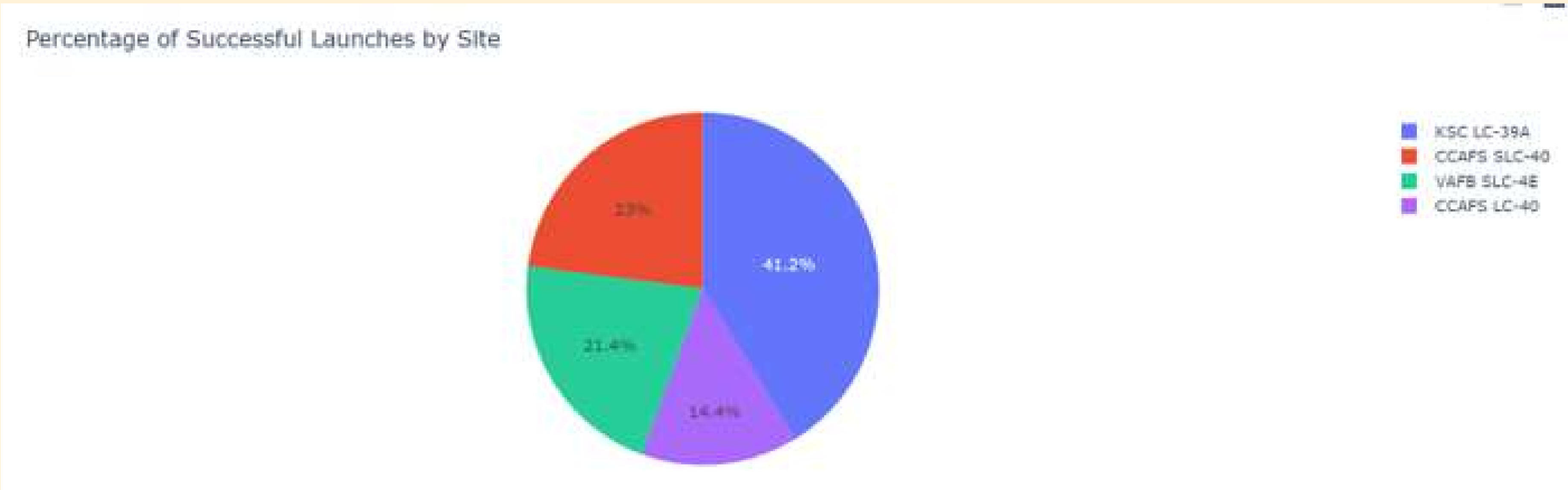


This shows the distance from CCAFS SLC-40 launch site to the nearest rail line is 1.25 km.

Section 5

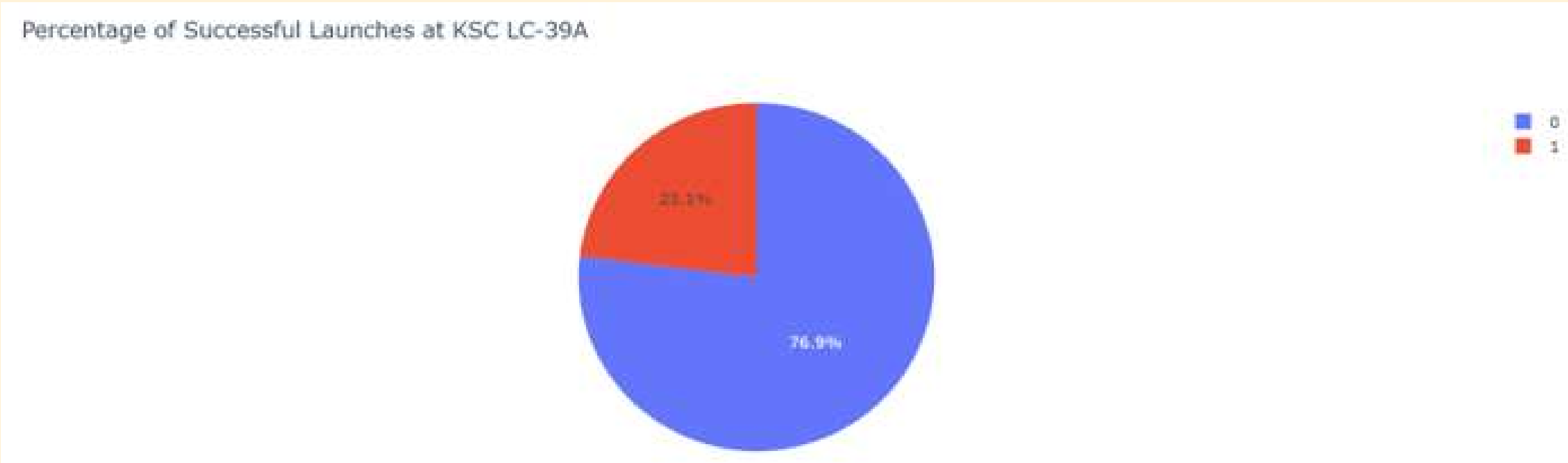
Build a Dashboard with Plotly Dash

Success Percentage by Launch Site



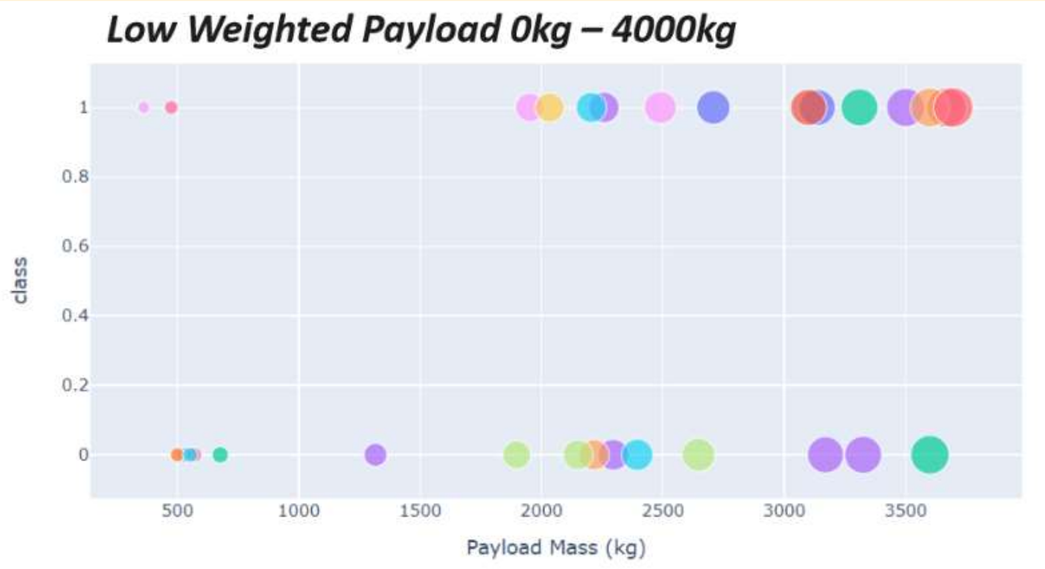
- We can see that the KSC LC-39A site has the most successful launches

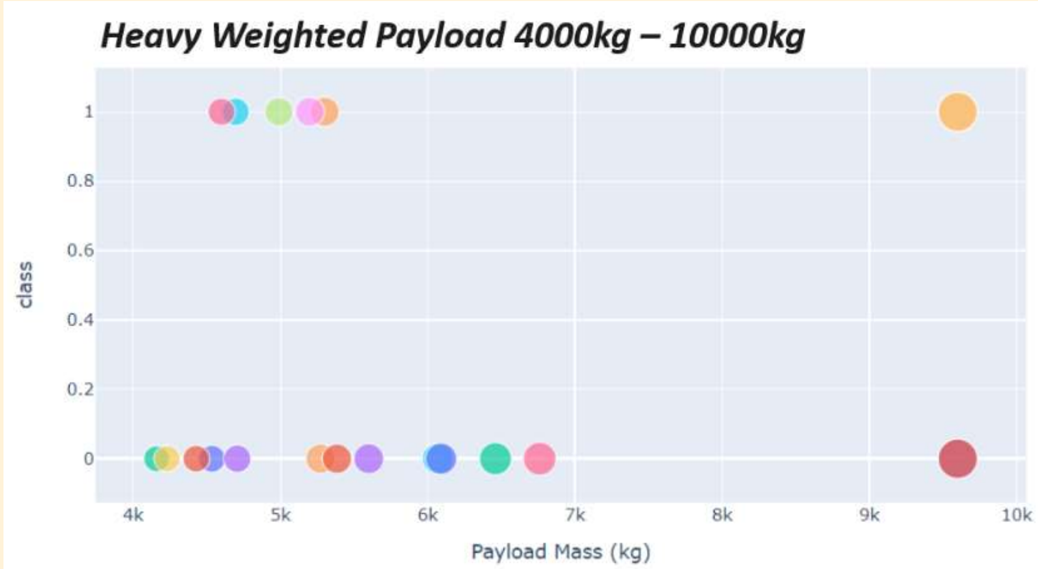
Highest Success Ratio Launch Site



- We can see that the KSC LC-39A site has the highest success ratio of all launch sites.

Payload vs. Launch Outcome



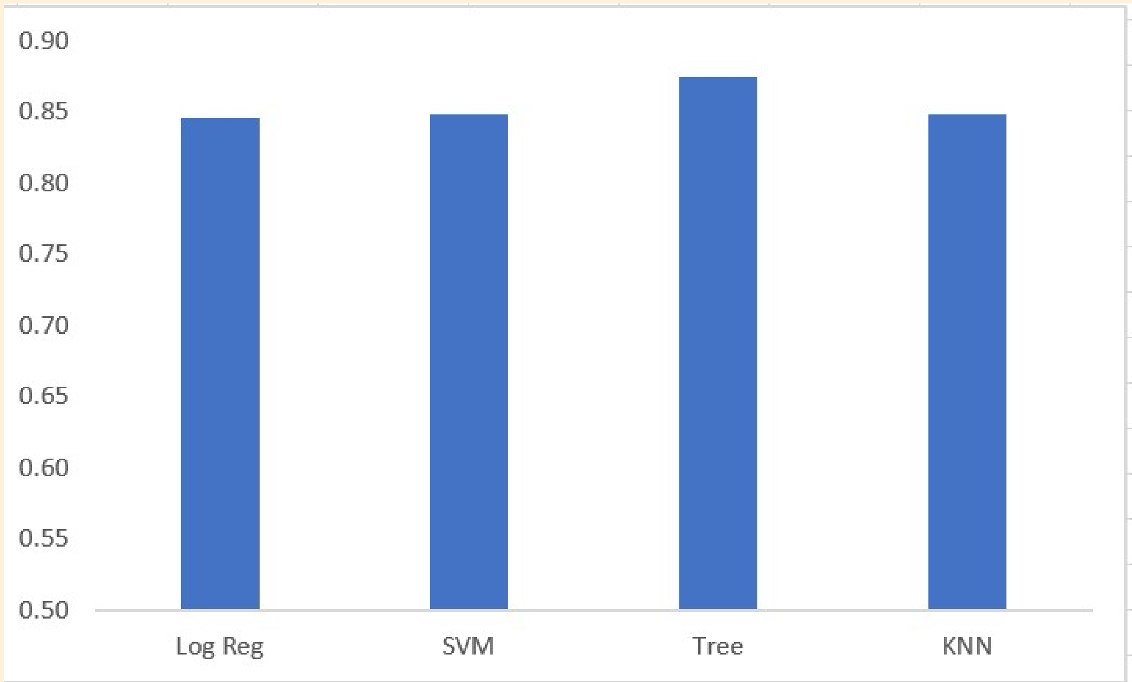


- We can see that the success rate for lower payload masses is greater than the success rate for higher payload masses.

Section 6

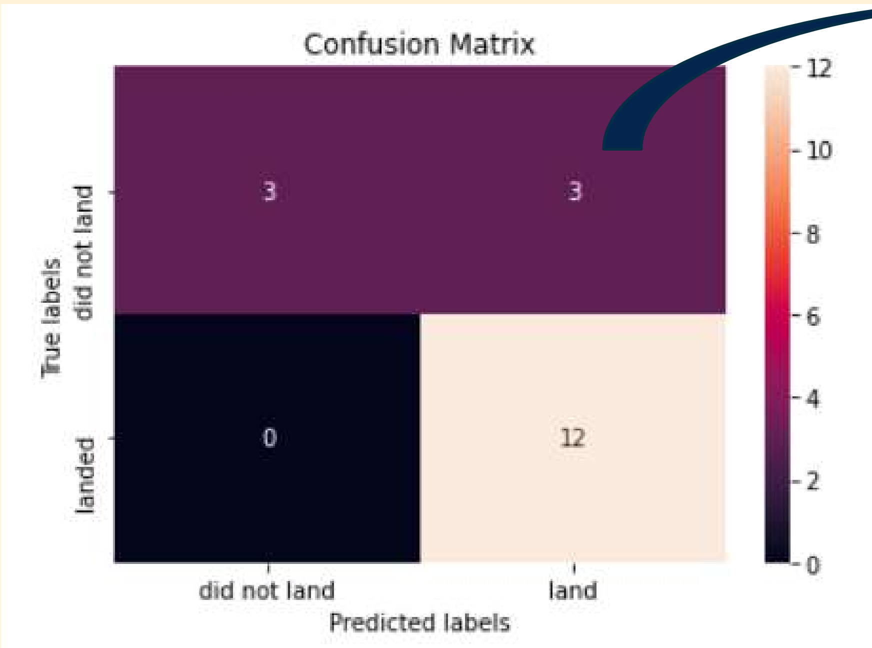
Predictive Analysis (Classification)

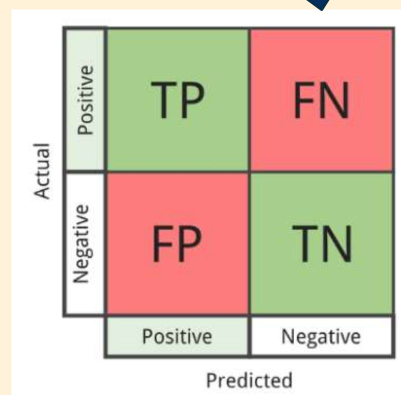
Classification Accuracy



- While all the models are relatively close, the Decision Tree model is the best with an accuracy of 0.875 or 87.5%.

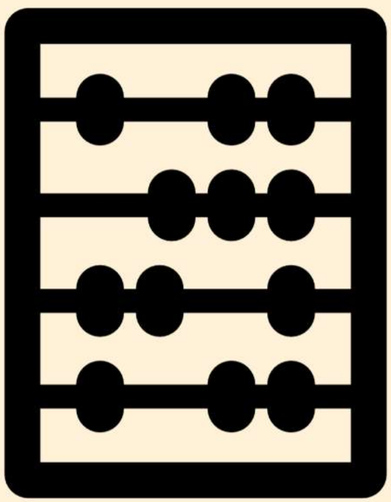
Confusion Matrix for the Decision Tree





- We can see that the Decision Tree can distinguish between different classes.
- The biggest problem is the incident of false negatives.
- Confusion matrix example from: https://towardsdatascience.com/visual-guide-to-the-confusion-matrix-bb63730c8eba

Conclusions



- Launch site appears to be an important factor in predicting successful launches
  - KSC LC-39A had had the most successful launches and highest successful launch ratio of all the sites.
- Lower mass payloads have performed more successfully than higher mass payloads.
- From 2013 to 2020 the rate of successful launches increased significantly
- The Decision Tree classifier algorithm has the highest accuracy on the test data.

Appendix

```
# Pandas is a software library written for the Python programming language for data manipulation and analysis.
import pandas as pd
# NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along
with a large collection of high-level mathematical functions to operate on these arrays
import numpy as np
# Matplotlib is a plotting library for python and pyplot gives us a MatLab like plotting framework. We will use this in our plot
ter function to plot data.
import matplotlib.pyplot as plt
#Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive a
nd informative statistical graphics
import seaborn as sns
# Preprocessing allows us to standarsize our data
from sklearn import preprocessing
# Allows us to split our data into training and testing data
from sklearn.model_selection import train_test_split
# Allows us to test parameters of classification algorithms and find the best one
from sklearn.model_selection import GridSearchCV
# Logistic Regression classification algorithm
from sklearn.linear_model import LogisticRegression
# Support Vector Machine classification algorithm
from sklearn.svm import SVC
# Decision Tree classification algorithm
from sklearn.tree import DecisionTreeClassifier
# K Nearest Neighbors classification algorithm
from sklearn.neighbors import KNeighborsClassifier
```

- A selection of some of the libraries used during the course of this project. Image from the IBM Developer Skills Network Capstone project.

Thank you!