IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sheng Khang

Outline

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

Executive Summary

- Space launches are expensive
- SpaceX provides a cost effective rocket that can cut cost
- Using machine learning, we can predict which launches will be more successful
  - Saving more money not having to cover failed launches

Introduction

- Project background and context
  - The Falcon 9 rocket launches are a cost saving way to deploy assets into orbit
    - Quoted from the SpaceX website as costing $62 million per launch while others can cost up to $165 million per launch
      - A 2.65 times less
- Problems we are facing
  - Only cost effective if the first stage successfully lands safely
  - Find a way to predict if there will be successful landing of the Falcon 9 rocket's first stage

Section 1

Methodology

Methodology

Executive Summary

- Data collection methodology:
    - SpaceX Rest API
    - Web Scrap Falcon 9 Wiki page
- Perform data wrangling
    - One Hot Encoding and pruning null entries
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
    - LR, SVC, Decision Tree, KNN

Data Collection

- First, we requested rocket launch data using the SpaceX API
  - Using `https://api.spacexdata.com/v4/launches/past`
    - Provides massive amount of information about SpaceX launches
- Needs to be normalized for easier viewing
  - Performed with the Pandas library's json_normalize() function in Python
- Extract relevant data
  - Primarily information about rocket type, payload weight, which launchpad was used, the core used, flight number, and the date of launch
- Filter Falcon 9 rocket launches only

Data Collection – SpaceX API

- Using the SpaceX API, we used the GET request to collect the data
- Performed basic data wrangling and formatted our data into a data frame
- https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/f80d7ab2aae278df011ecc013d59e75351334bad/data_collection/jupyter-labs-spacex-data-collection-api.ipynb

1. Get rocket launch data from SpaceX API:

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

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

2. Normalize data and convert to data frame with json

```
response_json = response.json()
data = pd.json_normalize(response_json)
```

3. Prune and clean data
   a. Select only features we want

```
data = data[['rocket', 'payloads', 'launchpad', 'cores', 'flight_number', 'date_utc']]
```

   b. Remove multiple cores

```
data = data[data['cores'].map(len)==1]
data = data[data['payloads'].map(len)==1]
```

   c. Extract payload as single value

```
data['cores'] = data['cores'].map(lambda x : x[0])
data['payloads'] = data['payloads'].map(lambda x : x[0])
```

   d. Convert date_utc to datetime datatype and restricting to date of launches

```
data['date'] = pd.to_datetime(data['date_utc']).dt.date
```

```
data = data[data['date'] <= datetime.date(2020, 11, 13)]
```

Data Collection –Scrapping

- Using the Falcon 9 wiki page, we parsed the tables into a Pandas data frame using BeautifulSoup
- https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/f80d7ab2aae278df011ecc013d59e75351334bad/data_collection/jupyter-labs-webscraping.ipynb

1. Obtain Falcon 9 Wiki page URL and parse the page using Beautiful Soup

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

```
response = requests.get(static_url)
print(response.status_code)
```

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

2. Extract table and column names

```
html_tables = soup.find_all("table")
```

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

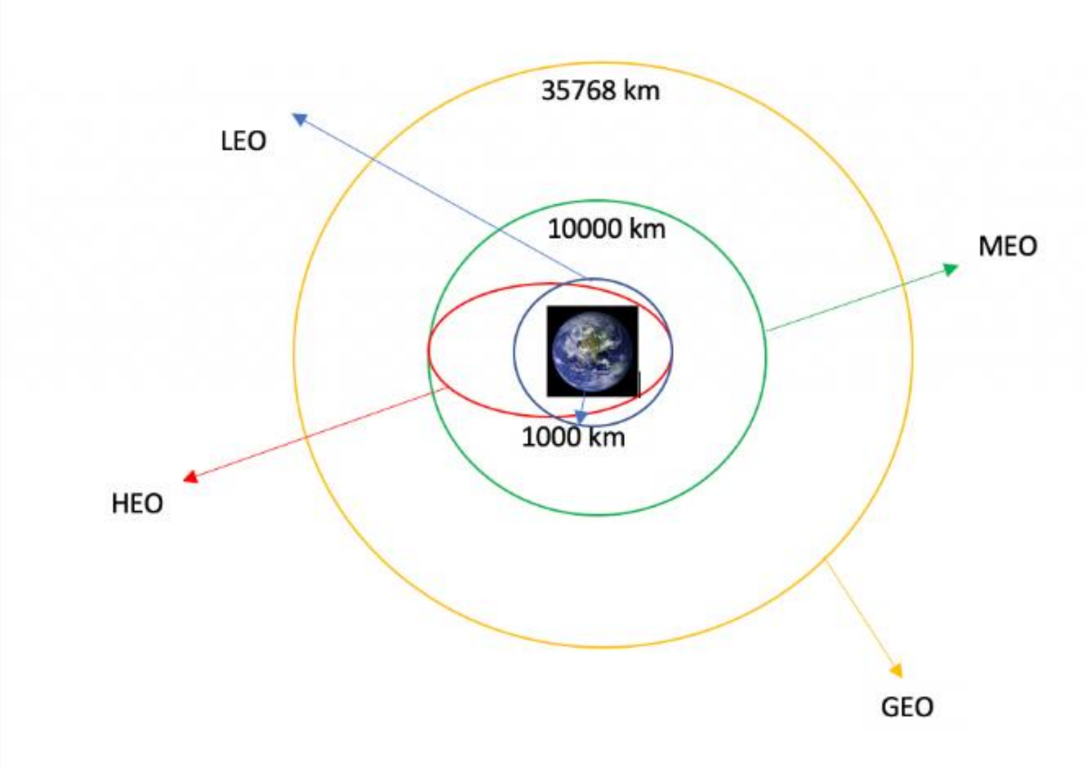
```
first_launch_table.find_all('th')
```

```
column_names = []
```

```
for row in first_launch_table.find_all('th'):
    name = extract_column_from_header(row)
    if (name != None and len(name) > 0):
        column_names.append(name)
```

Data Wrangling

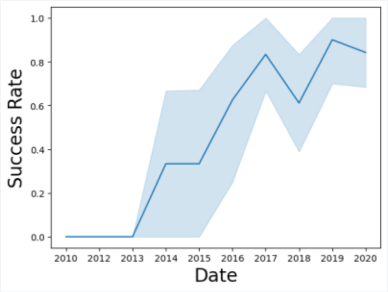
- Performed basic exploratory data analysis
  - Finding which variables has null values
- Determined the number of launches from each launch sights
- Created an "Outcome" column to depict the whether or not the mission outcome was successful

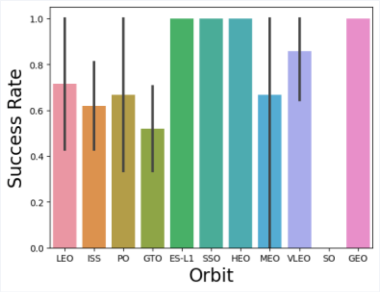


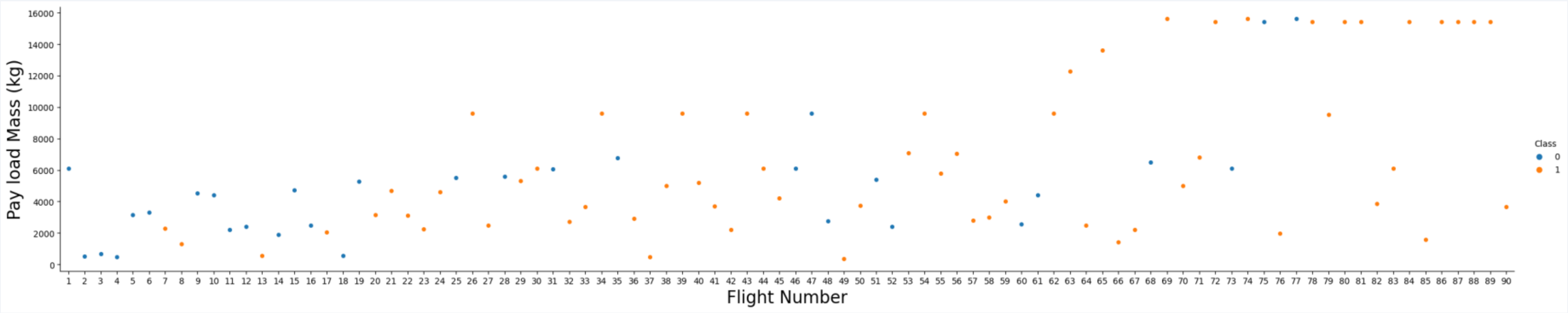
https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/49a4c8e4e082e763dd2d5c178d52c1835bdbd494/data_wrangling/labs-jupyter-spacex-Data%20wrangling.ipynb

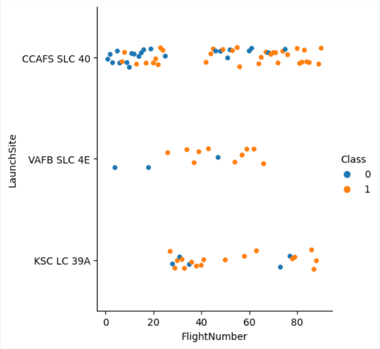
EDA with Data Visualization

- The relationship between variables were plotted
  - Found that the success of the landings increased with time
  - Certain launch sites had better outcomes than others









https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/7fbdf1abce2a4cba4e1a6ffc227a5a51b65e36df/exploratory-data-analysis/exploratory-analysis-using-pandas-and-matplotlib/jupyter-labs-eda-dataviz.ipynb

EDA with SQL

- SQL Queries Performed
  - Displaying unique names of launch sites
  - Displaying certain launch sites
  - Displaying the largest mass carried at a specific launch sites
  - Displaying average load carried by all F9 v1.1 rockets
  - Displaying the first date a successful landing outcome on a ground pad was achieved
  - Displaying the names of booster that had successful landing on drone ship with payload mass within a certain range

https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/7fbdf1abce2a4cba4e1a6ffc227a5a51b65e36df/exploratory-data-analysis/exploratory-analysis-using-sql/jupyter-labs-eda-sql-coursera_sqllite.ipynb

Build an Interactive Map with Folium

- Marked all launch sites on a map
  - For easy visualization of where the launch sites are
- Marked all the successful and unsuccessful launches from each site on the map
  - For easy visualization of which sites had more successful outcomes
- Calculated the distance between the launch sights to its proximities
  - Closest coastline, railroad, highway
  - See if any of these may be attributing factors to a successful outcome

https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/24e925bf2ad00a2e4bdde2f775142a52f2f492e9/interactive-visual-analytics-and-dashboard/lab_jupyter_launch_site_location.ipynb

Build a Dashboard with Plotly Dash

- Built a dropdown menu to select different launch sites
  - To change between the launch sites and the different initial data analysis that we found
- Created a callback function to make a pie chart
  - Depicts percentage of successful launches
- Built a slider that the user can change the payload size to see the statistics of their launch outcome
- Made a callback function that showed a scatter plot showing the relationship between the outcome and the payload mass of different booster versions

https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/24e925bf2ad00a2e4bdde2f775142a52f2f492e9/interactive-visual-analytics-and-dashboard/spacex_dash_app.py

Predictive Analysis (Classification)

1. Created a NumbPy array with the outcome column to use for training assigning it as the Y variable

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

2. Standardized the remaining features with scikit-learn's preprocessing into input for training, assigning it as the X variable

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

3. Split all the data into training and testing sets using scikit-learn's train_test_split

```
X_train, X_test, Y_train, Y_test = train_test_split(X_transform, Y, test_size=0.2, random_state=2)
```

4. Created a Logistic Regression object and trained it with the training data and tested to see the accuracy of the analysis

```
parameters ={"C":[0.01,0.1,1],'penalty':['l2'], 'solver':['lbfgs']}# l1 lasso l2 ridge
lr=LogisticRegression()
```

```
logreg_cv = GridSearchCV(lr,parameters,cv=10)
```

```
logreg_cv.fit(X_train,Y_train)
```

5. Created a Support Vector Machine object and trained it with the training data and tested to see the accuracy of the analysis

```
parameters = {'kernel':('linear', 'rbf','poly','rbf', 'sigmoid'),
              'C': np.logspace(-3, 3, 5),
              'gamma':np.logspace(-3, 3, 5)}
svm = SVC()
```

```
svm_cv = GridSearchCV(svm,parameters,cv=10)
```

```
svm_cv.fit(X_train,Y_train)
```

6. Created a Decision Tree Classifier object and trained it with the training data and tested to see the accuracy of the analysis

```
parameters = {'criterion': ['gini', 'entropy'],
     'splitter': ['best', 'random'],
     'max_depth': [2*n for n in range(1,10)],
     'max_features': ['auto', 'sqrt'],
     'min_samples_leaf': [1, 2, 4],
     'min_samples_split': [2, 5, 10]}

tree = DecisionTreeClassifier()
```

```
tree_cv = GridSearchCV(tree,parameters,cv=10)
```

```
tree_cv.fit(X_train,Y_train)
```

7. Created a K Nearest Neighbors object and trained it with the training data and tested to see the accuracy of the analysis

```
parameters = {'n_neighbors': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
              'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
              'p': [1,2]}

KNN = KNeighborsClassifier()
```

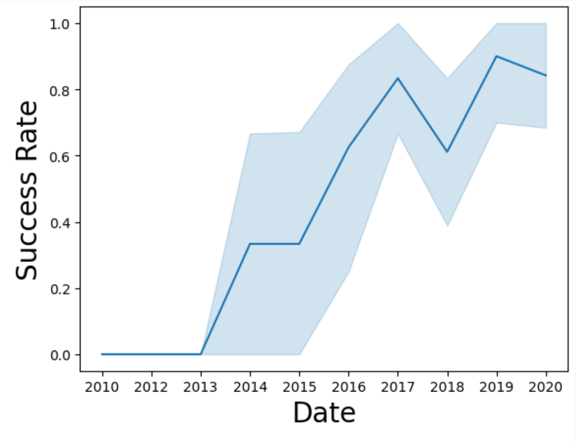
```
knn_cv = GridSearchCV(KNN, parameters, cv = 10)
```

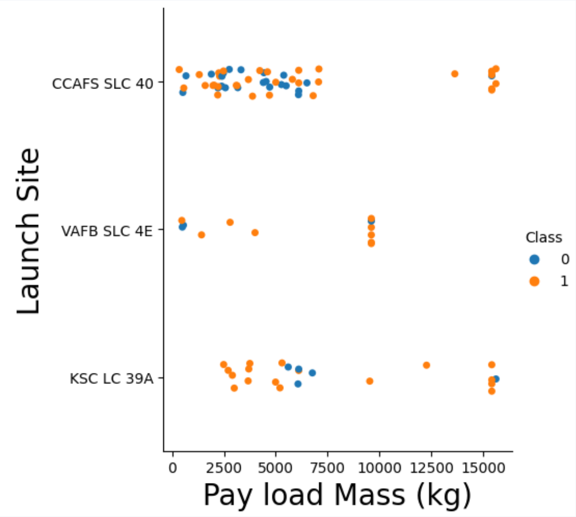
```
knn_cv.fit(X_train,Y_train)
```


https://github.com/khangsheng1/Coursera_Data_Science_Capstone/blob/35d94b34c8fc63af8f782965f04290e237b955a8/predictive-analysis/SpaceX_Machine_Learning_Prediction_Part_5.jupyterlite.ipynb

Results

- As time went on, the better the success rate for the outcome became
- Payload size and launch sites played a role in the outcome of launches
  - Heavier rockets seemed to be more successful
  - VAFB SLC 4E and KSC LSC 39A had a better percentage of successful launches





Results

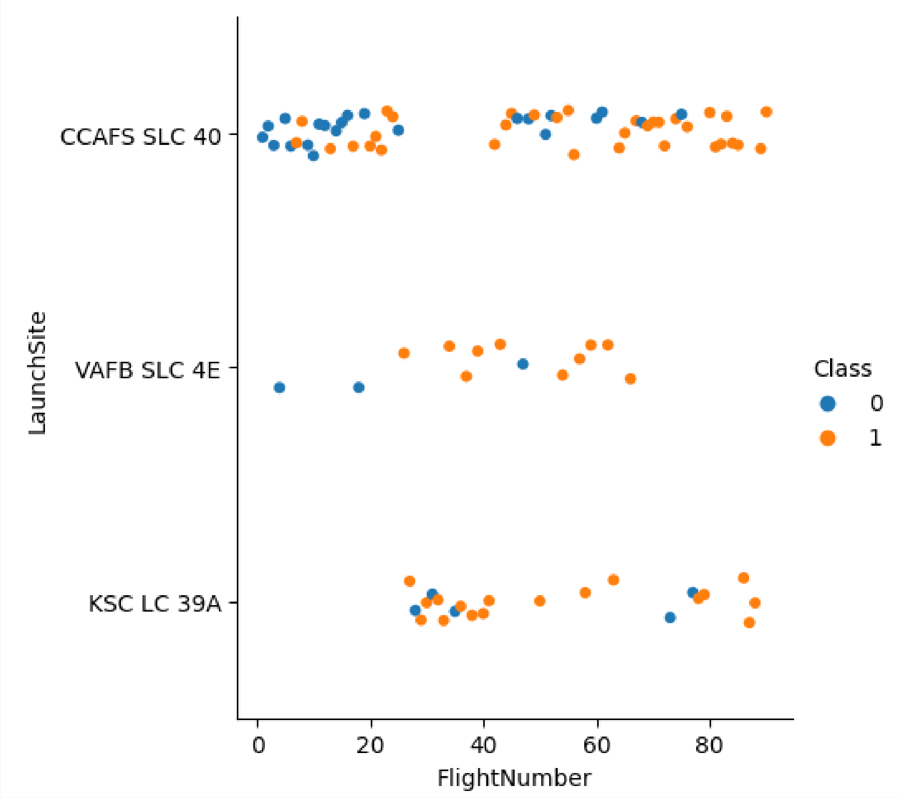
- Predictive analysis results
- Out of all the learning objects created, Decision Tree Classifier performed the best
    - Logistic Regression Classifier
        - Accuracy = 84.6%
    - Support Vector Machine Classifier
        - Accuracy = 84.8%
    - Decision Tree Classifier
        - Accuracy = 87.5%
    - K Nearest Neighbors Classifier
        - Accuracy = 84.8%

Section 2

Insights drawn from EDA

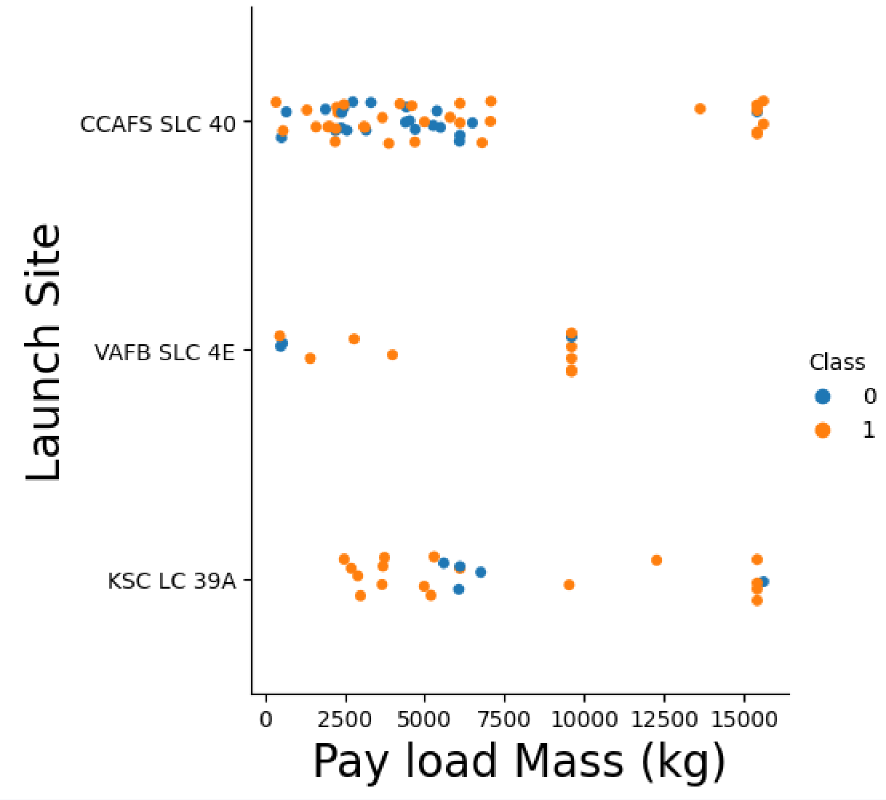
Flight Number vs. Launch Site

- As the flight numbers increased, the more successful the outcomes became
  - This shows the learning of the SpaceX team as they continued to launch more rockets



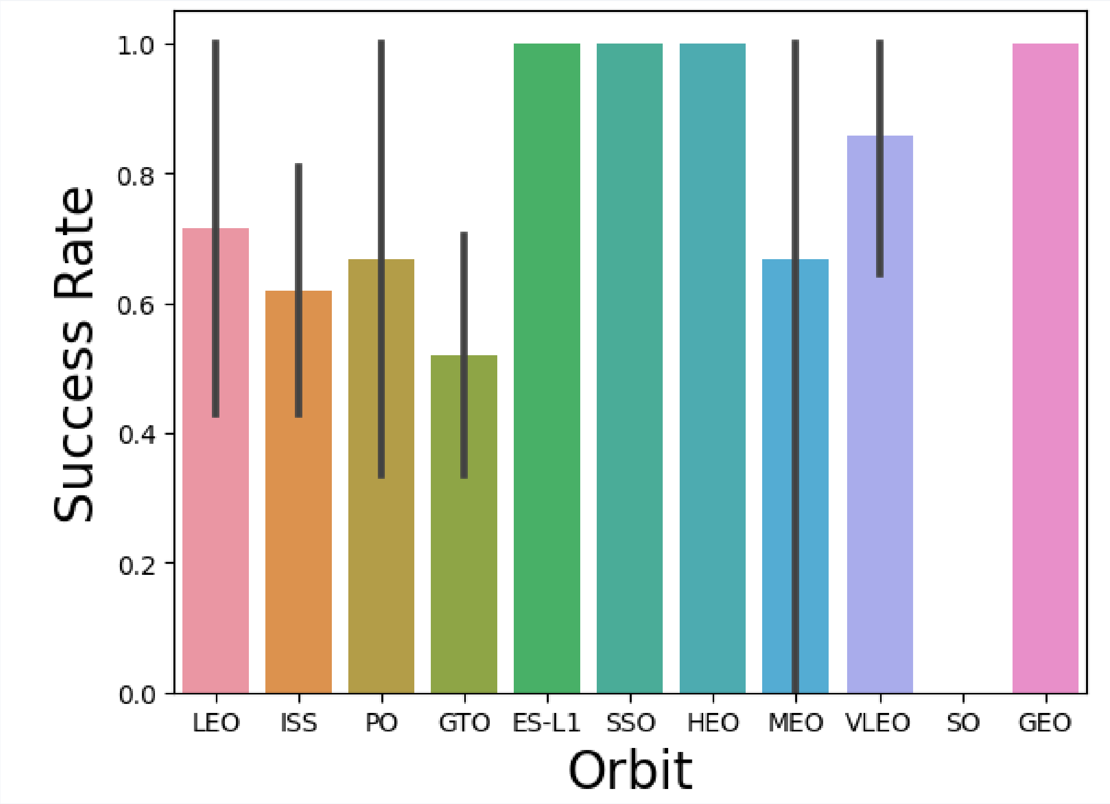
Payload vs. Launch Site

- Different launch sites had different success rates
- Payload mass also influenced successful outcomes



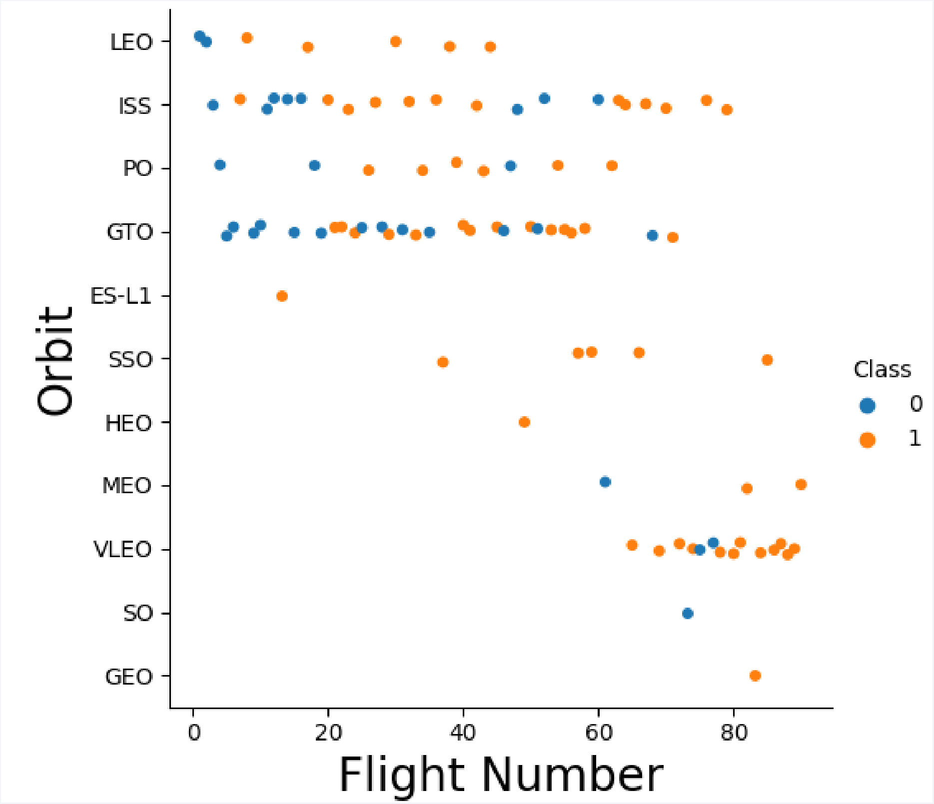
Success Rate vs. Orbit Type

- ES-L1, SSO, HEO, and GEO orbit types had perfect success rates



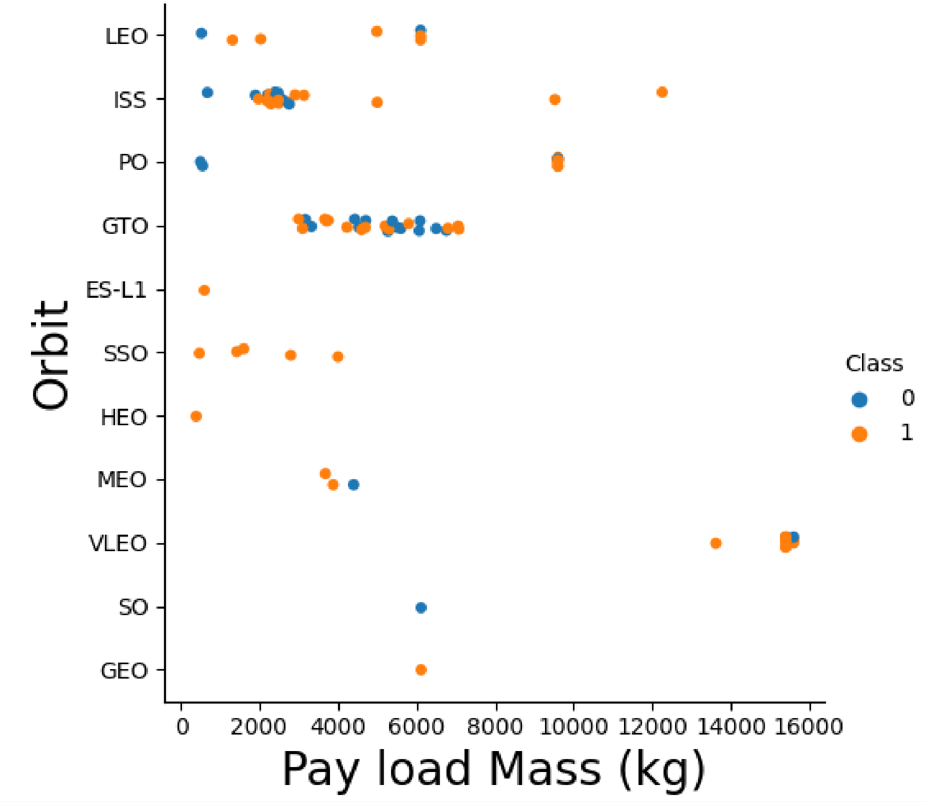
Flight Number vs. Orbit Type

- As the flight numbers increased the more successful the launches became
- Indicates that the SpaceX team learned as they kept launching rockets



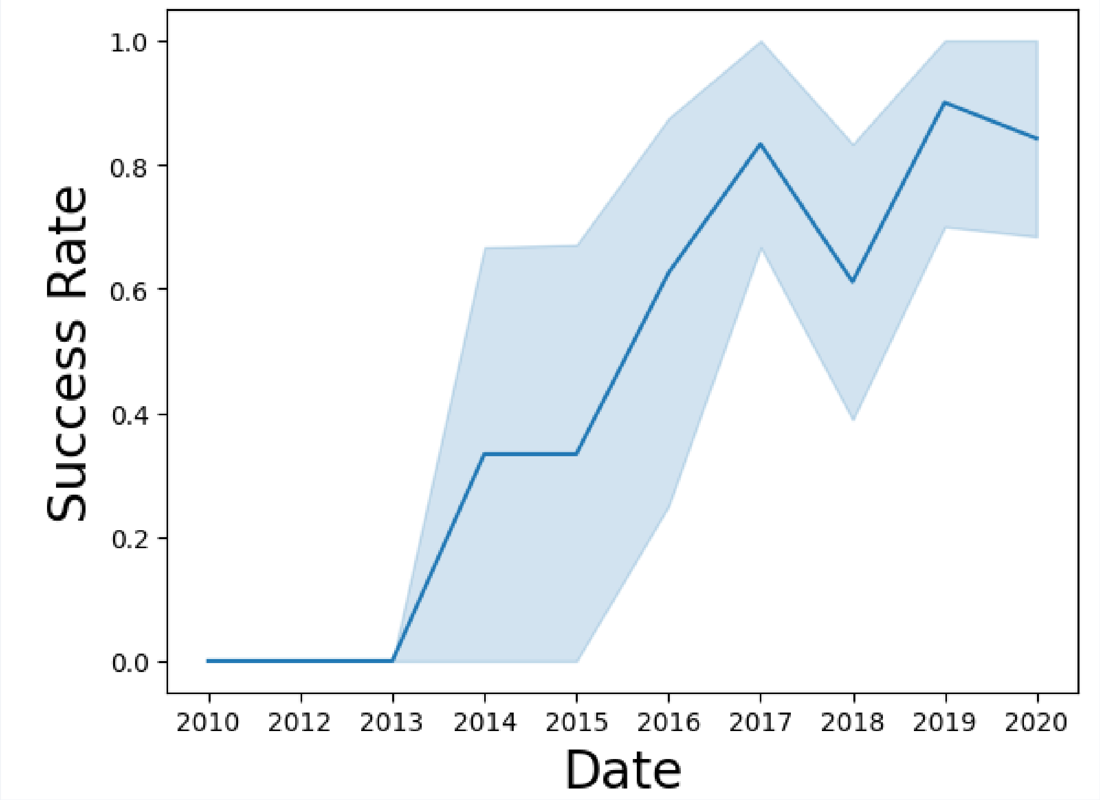
Payload vs. Orbit Type

- Payloads between 4000 and 6000 kg had a mix bag of successful and unsuccessful launches
- There are a few orbits that had perfect launches
  - SSO
  - HEO
  - ES-L1



Launch Success Yearly Trend

- As time went on, the SpaceX team learned how to make launches more successful



All Launch Site Names

- The launches are in code
- These are areas in California and Florida

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

Launch Site Names Begin with 'CCA'

- Using SQL, we were able to find 5 random launches from the launch sight beginning with 'CCA'

Date	Time (UTC)	Booster_Version	Launch_Site	Payload	PAYLOAD_MASS__KG_	Orbit	Customer	Mission_Outcome	Landing_Outcome
2010-04-06	18:45:00	F9 v1.0 B0003	CCAFS LC-40	Dragon Spacecraft Qualification Unit	0	LEO	SpaceX	Success	Failure (parachute)
2010-08-12	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
2012-08-10	00:35:00	F9 v1.0 B0006	CCAFS LC-40	SpaceX CRS-1	500	LEO (ISS)	NASA (CRS)	Success	No attempt
2013-01-03	15:10:00	F9 v1.0 B0007	CCAFS LC-40	SpaceX CRS-2	677	LEO (ISS)	NASA (CRS)	Success	No attempt

Total Payload Mass

- The total mass carried by NASA

Total Payload Mass (kg) carried by NASA
107010

Average Payload Mass by F9 v1.1

- The average payload mass carried by booster version F9 v1.1

avg(PAYLOAD_MASS__KG_)
2534.6666666666665

First Successful Ground Landing Date

- The date of the first successful landing outcome on ground pad

min(Date)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

Booster_Version
F9 FT B1021.1
F9 FT B1022
F9 FT B1023.1
F9 FT B1026
F9 FT B1029.1
F9 FT B1021.2
F9 FT B1029.2
F9 FT B1036.1
F9 FT B1038.1
F9 B4 B1041.1
F9 FT B1031.2
F9 B4 B1042.1
F9 B4 B1045.1
F9 B5 B1046.1

Total Number of Successful and Failure Mission Outcomes

- The total number of successful and failure mission outcomes

Mission Outcome	Count
Failure (in flight)	1
Success	98
Success	1
Success (payload status unclear)	1

Boosters Carried Maximum Payload

- The names of the booster which have carried the maximum payload mass

boosterversion
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

2015 Launch Records

- The failed landing outcomes in drone ship, their booster versions, and launch site names for in year 2015

month	Landing Outcome	Booster Version	Launch Site
10	Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
04	Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

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

Landing Outcome	Date
No attempt	2017-03-16
Success (ground pad)	2017-03-06
Success (ground pad)	2017-02-19
Success (drone ship)	2017-01-14
Success (ground pad)	2017-01-05
Success (drone ship)	2016-08-14
Success (drone ship)	2016-08-04
Success (ground pad)	2016-07-18
Failure (drone ship)	2016-06-15
Success (drone ship)	2016-06-05
Success (drone ship)	2016-05-27
Failure (drone ship)	2016-04-03
Failure (drone ship)	2016-01-17
Success (ground pad)	2015-12-22
Controlled (ocean)	2015-11-02
Failure (drone ship)	2015-10-01
Precluded (drone ship)	2015-06-28
No attempt	2015-04-27
Failure (drone ship)	2015-04-14
No attempt	2015-02-03
Uncontrolled (ocean)	2014-09-21
Controlled (ocean)	2014-07-14
No attempt	2014-07-09
No attempt	2014-06-01
No attempt	2014-05-08
Controlled (ocean)	2014-04-18
Uncontrolled (ocean)	2013-09-29
No attempt	2013-03-12
No attempt	2013-01-03
No attempt	2012-08-10
No attempt	2012-05-22
Failure (parachute)	2010-08-12

Section 3

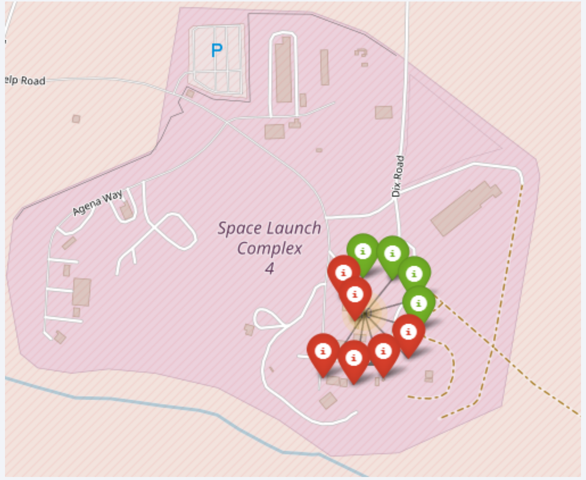
Launch Sites Proximities Analysis

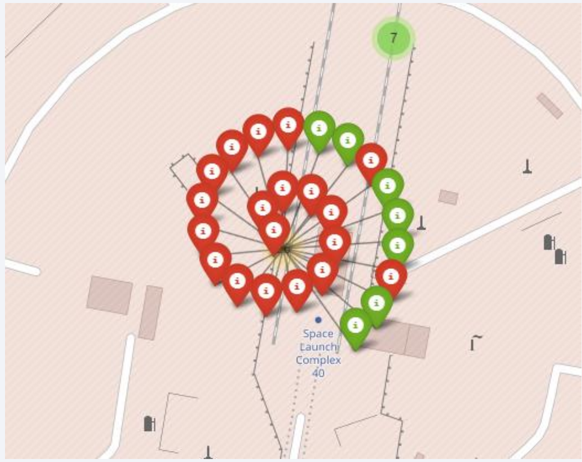
Launch Sites of Falcon 9 Rockets

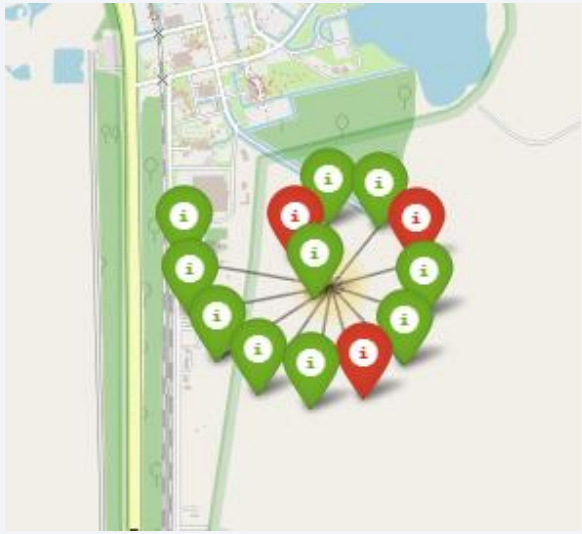
- Launch sites are in California and Florida

Landing Outcomes at Each Lauch Site

- Green means successful outcome
- Red means unsuccessful outcome

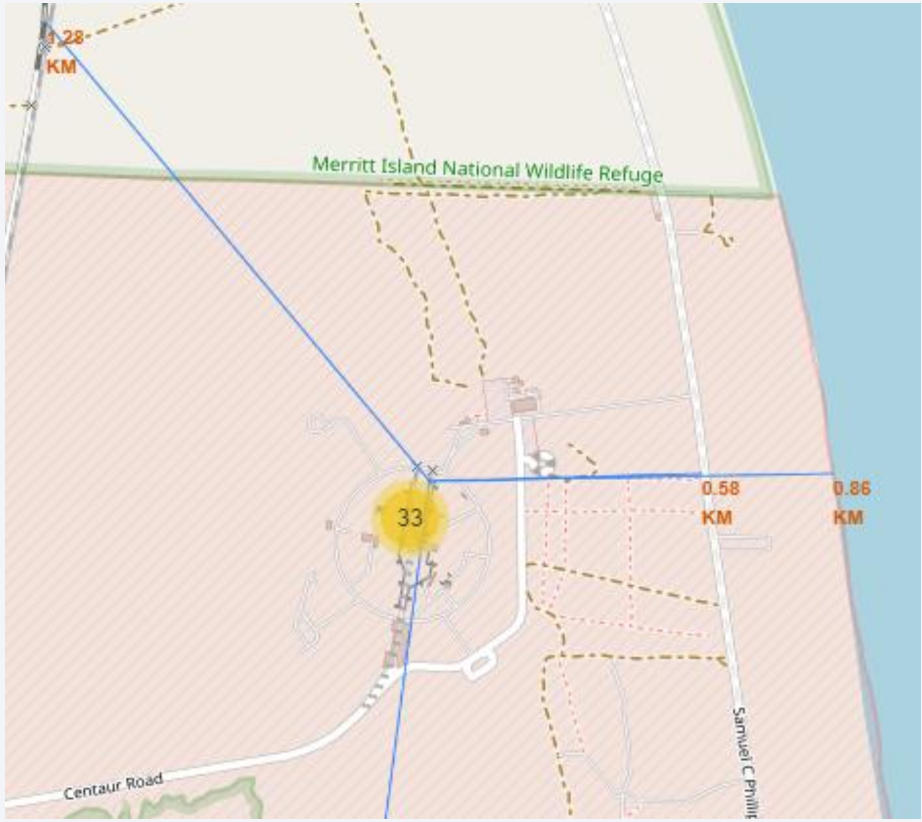






Proximity of Launch Site to Point of Interests

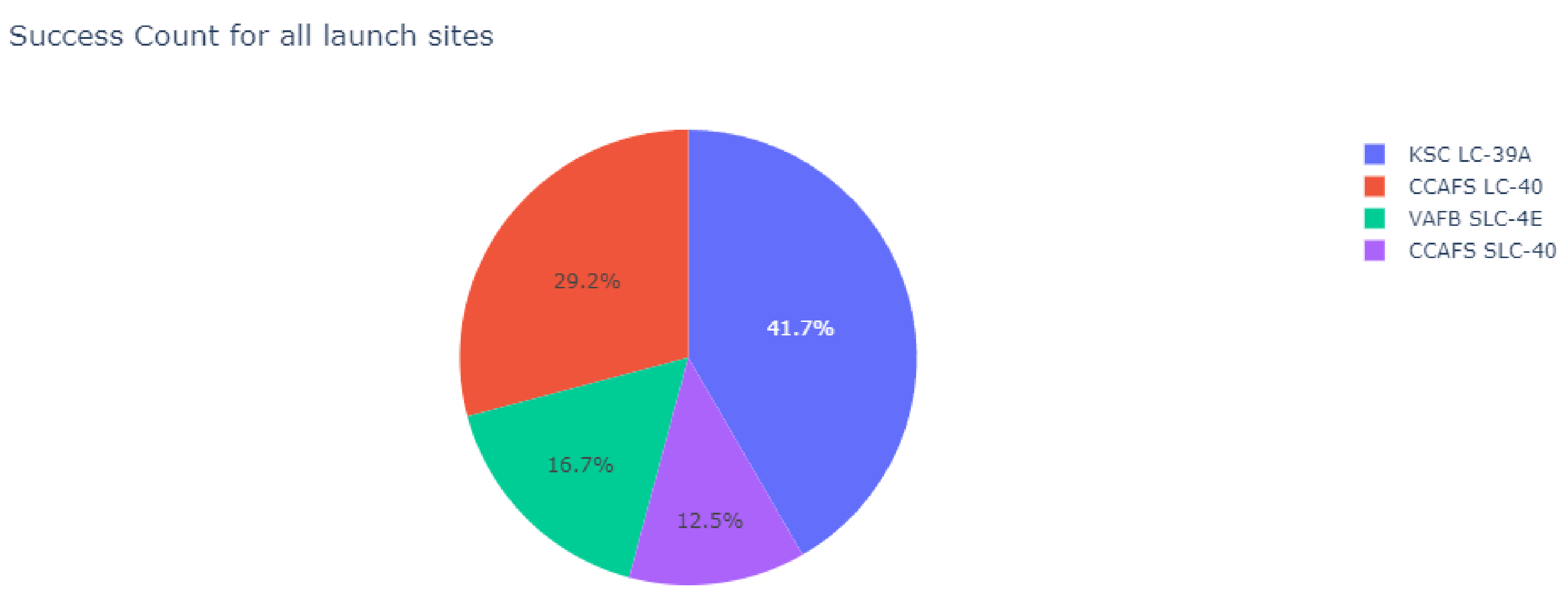
- Each line represents the distance from the launch site to a different point of interest
  - To the nearest
    - Railway
    - Highway
    - Coast line



Section 4

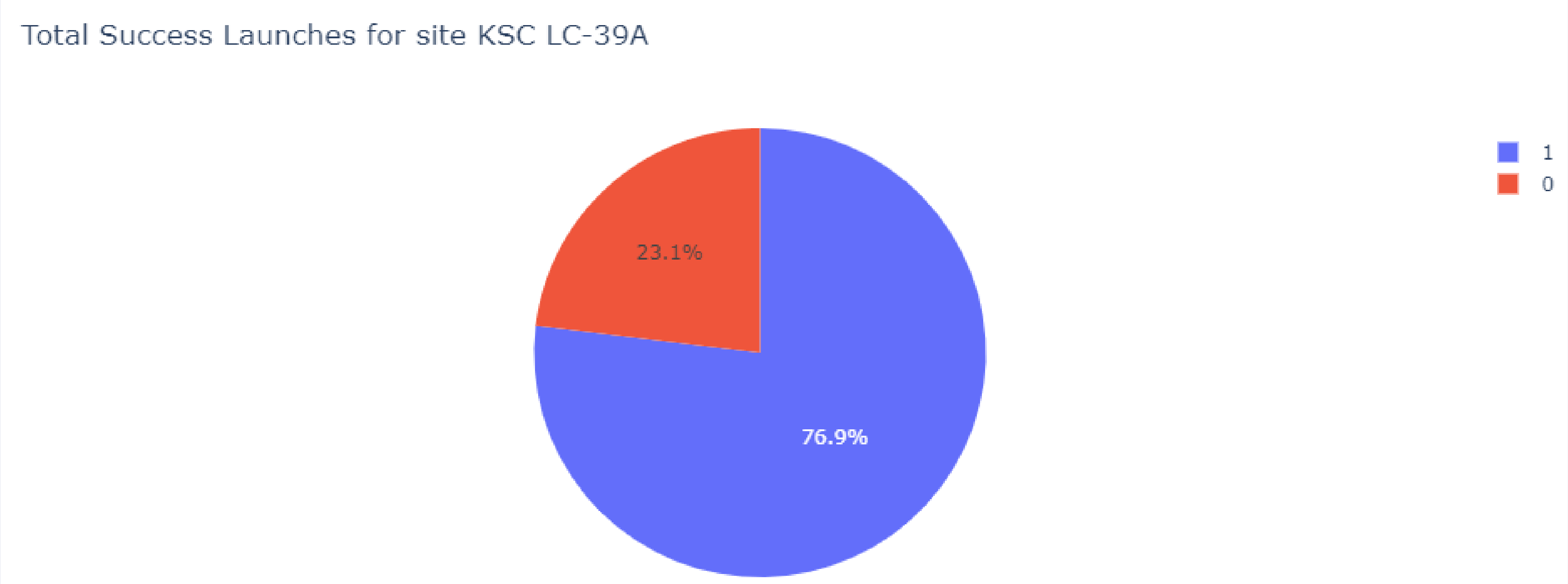
Build a Dashboard with Plotly Dash

Percent of Successful Launches at Each Site



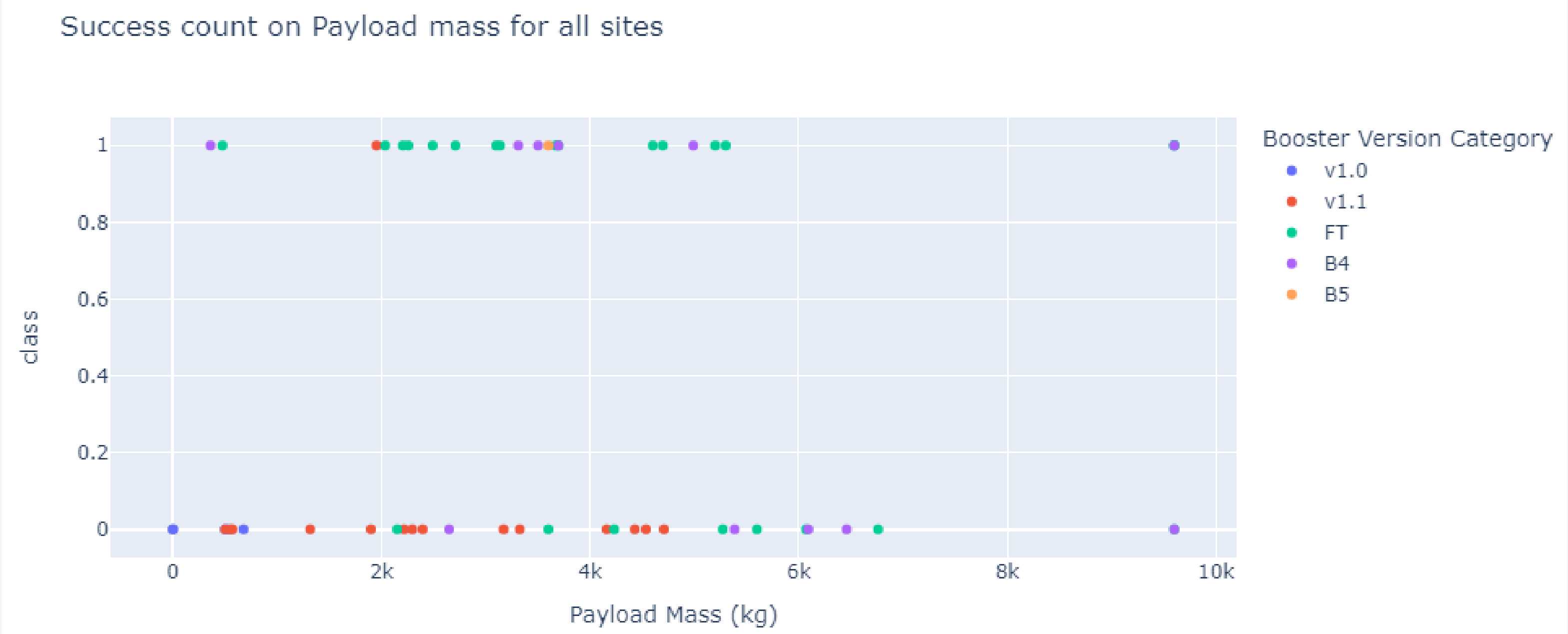
- KSC LC-39A has the highest percentage of successful launches

KSC LC-39A Ratio of Launch Outcomes



- More than ¾'s of the launches are successful at this location

Payload vs. Launch Outcome



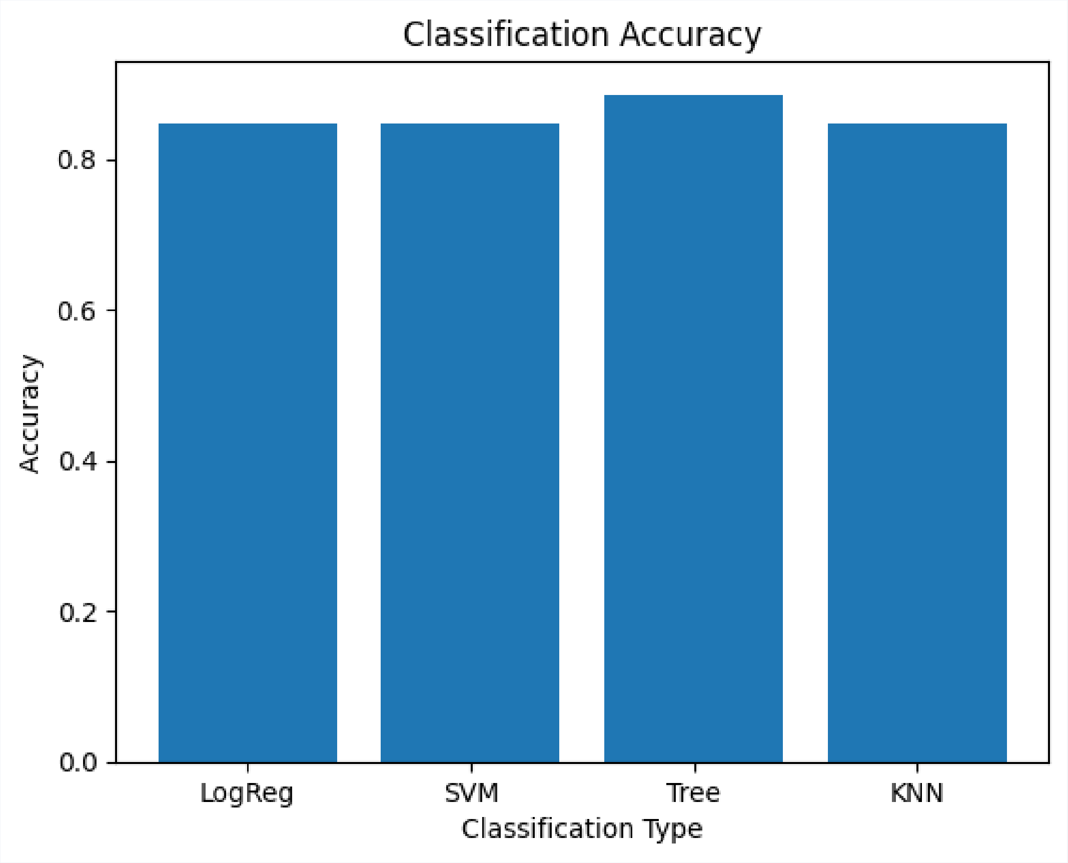
- The largest payload (close to 10K kg) had both a successful and unsuccessful launch
- FT Booster Version has the most successful launches
- v1.1 had the least successful launches

Section 5

Predictive Analysis (Classification)

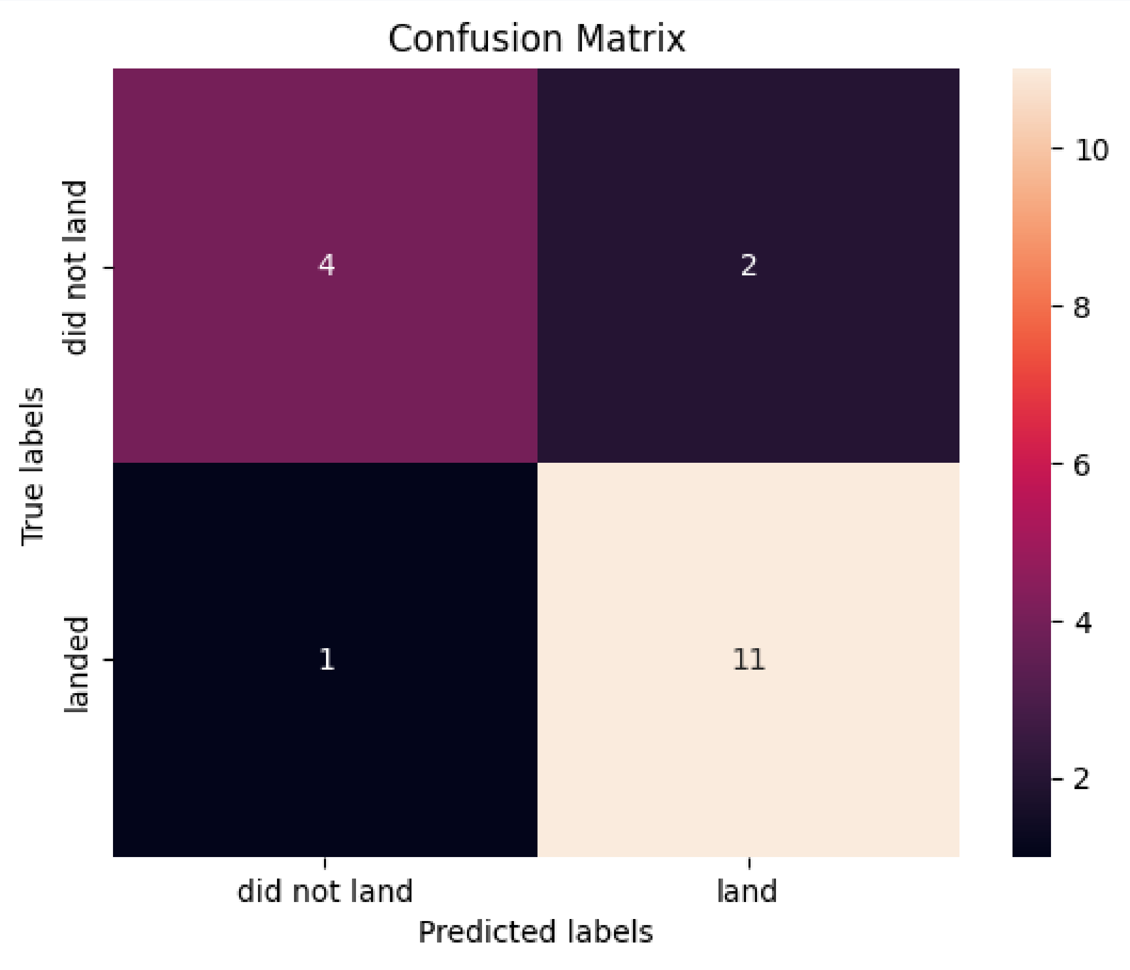
Classification Accuracy

- Decision Tree had the highest accuracy rate
    - 0.87
- All others performed relatively the same



Confusion Matrix

- Decision Tree had only 3 mislabeled predictions
  - 2 false positives
  - 1 false negative



Conclusions

- Orbits SSO, HEO, ES-L1 had perfect launch outcomes
- KSC LC-39A had the highest percentage of launch outcomes
- Decision Tree classifier is the best at predicting whether or not a launch will be successful

Thank you!