IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Blu LeBlanc
11/26/25

Outline

EXECUTIVE SUMMARY

INTRODUCTION

METHODOLOGY

RESULTS

CONCLUSION

APPENDIX

Executive Summary

Methodologies:

- Data Collection -> web scraping SpaceX API
- Exploratory Data Analysis -> data wrangling, data manipulation, data visualization
- Machine Learning Classification Prediction Models

Summary of all results:

- Collected data on SpaceX rocket launches
- EDA utilized to identify valuable features to predict outcome of launch
- ML models yielded best model to predict success or failure of SpaceX launch, given features of the launch

Introduction

SpaceX claims much lower cost (62 million) for Falcon 9 launches because they can reuse the first stage. By determining if a first stage will land successfully, we can predict the cost of a SpaceX launch

Project Goal: Predict if Falcon 9 first stage will land successfully using features from SpaceX launch data

Section 1

Methodology

Methodology

Executive Summary

Data collection methodology:

Scraped SpaceX API for launch data

Perform data wrangling

Utilized Pandas to track landing outcomes and perform mean imputation

Perform exploratory data analysis (EDA) using visualization and SQL

Perform interactive visual analytics using Folium and Plotly Dash

Perform predictive analysis using classification models

Compared various classification models from sci-kit learn with GridSearchCV optimization

Data Collection

Libraries: requests, pandas → Helper functions to extract information using identification numbers in launch data → Pulled data from SpaceX API → Converted json result into pandas dataframe → Called helper functions to replace IDs with launch information → Constructed final dataframe

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block
0	1	2006-03-24	Falcon 1	20.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
1	2	2007-03-21	Falcon 1	NaN	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
2	4	2008-09-28	Falcon 1	165.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
3	5	2009-07-13	Falcon 1	200.0	LEO	Kwajalein Atoll	None None	1	False	False	False	None	NaN
4	6	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0

Data Collection – SpaceX API

Data Collection helper functions

Notebook: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/blob/main/SpaceX_Data%20Collection%20API.ipynb

```python
# Takes the dataset and uses the rocket column to call the API and append the data to the list
def getBoosterVersion(data):
    for x in data['rocket']:
       if x:
        response = requests.get("https://api.spacexdata.com/v4/rockets/"+str(x)).json()
        BoosterVersion.append(response['name'])
```

From the `launchpad` we would like to know the name of the launch site being used, the logitude, and the latitude.

```python
# Takes the dataset and uses the launchpad column to call the API and append the data to the list
def getLaunchSite(data):
    for x in data['launchpad']:
       if x:
         response = requests.get("https://api.spacexdata.com/v4/launchpads/"+str(x)).json()
         Longitude.append(response['longitude'])
         Latitude.append(response['latitude'])
         LaunchSite.append(response['name'])
```

From the `payload` we would like to learn the mass of the payload and the orbit that it is going to.

```python
# Takes the dataset and uses the payloads column to call the API and append the data to the lists
def getPayloadData(data):
    for load in data['payloads']:
       if load:
        response = requests.get("https://api.spacexdata.com/v4/payloads/"+load).json()
        PayloadMass.append(response['mass_kg'])
        Orbit.append(response['orbit'])
```

From `cores` we would like to learn the outcome of the landing, the type of the landing, number of flights with that core, whether gridfins were used, wheter the core is reused, wheter legs were used, the landing pad used, the block of the core which is a number used to seperate version of cores, the number of times this specific core has been reused, and the serial of the core.

```python
# Takes the dataset and uses the cores column to call the API and append the data to the lists
def getCoreData(data):
    for core in data['cores']:
            if core['core'] != None:
                response = requests.get("https://api.spacexdata.com/v4/cores/"+core['core']).json()
                Block.append(response['block'])
                ReusedCount.append(response['reuse_count'])
                Serial.append(response['serial'])
            else:
                Block.append(None)
                ReusedCount.append(None)
                Serial.append(None)
            Outcome.append(str(core['landing_success'])+' '+str(core['landing_type']))
            Flights.append(core['flight'])
            GridFins.append(core['gridfins'])
            Reused.append(core['reused'])
            Legs.append(core['legs'])
            LandingPad.append(core['landpad'])
```

Now let's start requesting rocket launch data from SpaceX API with the following URL:

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

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

Data Wrangling

- Libraries: Pandas, Numpy
- Mean imputation on PayloadMass column
- Filtered for Falcon 9
- Created count of landing outcomes
- Created Landing Class label column

Data Wrangling – Landing Outcomes

Data Wrangling samples

Notebook: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/blob/main/SpaceX_Data%20Wrangling.ipynb

In [12]:

```
df['Class']=landing_class
df[['Class']].head(8)
```

Out[12]:

	Class
0	0
1	0
2	0
3	0
4	0
5	0
6	1
7	1

In [13]:

```
df.head(5)
```

Out[13]:

s	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
2	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
0	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
0	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
0	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0

We can use the following line of code to determine the success rate:

In [14]:

```
df["Class"].mean()
```

Out[14]:

```
np.float64(0.6666666666666666)
```

EDA with Data Visualization

Libraries: pyplot, seaborn, pandas

Charts:

Scatter: FlightNumber vs PayloadMass, FlightNumber vs LaunchSite, PayloadMass vs LaunchSite, FlightNumber vs Orbit, PayloadMass vs Orbit

Bar: Success Rate of Orbit Type

Line: Launch Success Yearly Trend

Data visualizations used to determine which variables affect success rate. Performed feature selection and encoding

Notebook: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/blob/main/SpaceX_EDA%20Data%20Viz.ipynb

EDA with SQL

- Libraries: sqlite3
- EDA Queries:
  - Display the name of the unique launch sites in the space mission
  - Display 5 records where launch sites begin with the string ‘CCA’
  - Display the total payload mass carried by boosters launched by NASA (CRS)
  - Display average payload mass carried by booster version F9 v1.1
  - List the date when the first succesful landing outcome in ground pad was acheived.
  - List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000
  - List the total number of successful and failure mission outcomes
  - List all the booster_versions that have carried the maximum payload mass, using a subquery with a suitable aggregate function.
  - List the records which will display the month names, failure landing_outcomes in drone ship ,booster versions, launch_site for the months in year 2015.
  - 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.
    - Notebook: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/blob/main/SpaceX_EDA%20SQL.ipynb

Build an Interactive Map with Folium

- Interactive Map tasks:
  - Marked all launch sites on map using circles and coordinates
  - Created green and red markers to denote success/failed launches for each site on the map
  - Created distance markers to calculate the distance between launch sites and objects
- Summarize what map objects such as markers, circles, lines, etc. you created and added to a folium map
- Interactive map can be used to view launch site locations, launch sites with highest success rates, distance between proximities and launch sites
- Notebook: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/blob/main/SpaceX_Interactive_Visual_Analytics_with_Folium_lab.ipynb

Build a Dashboard with Plotly Dash

- Dashboard tasks:
  - Add Launch Site drop-down selector
  - Add callback function to render success pie chart for launch site
  - Add range slider for choosing Payload
  - Add callback function to render success scatter plot for payload
- Dashboard can answer questions about comparing success rate of launch sites, comparing success rates of payload ranges, comparing success rate of F9 Booster versions
- Notebook: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/blob/main/spacex_dash_app.py

Predictive Analysis (Classification)

- Found best hyperparameters for SVM, Classification Trees, and Logistic Regression using GridSearchCV and Sci-kit Learn functions

Libraries: sklearn, train_test_split, GridSearchCV, LogisticRegression, SVC, DecisionTreeClassifier, KNeighborsClassifier

Converted Class to array format

Standardized and split data into train test split

Created different classification models. Models: LR, SVM, Tree, KNN

Evaluated each classification model with accuracy and confusion matrix

- Notebook: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/blob/main/SpaceX_Machine%20Learning%20Prediction.ipynb

Results

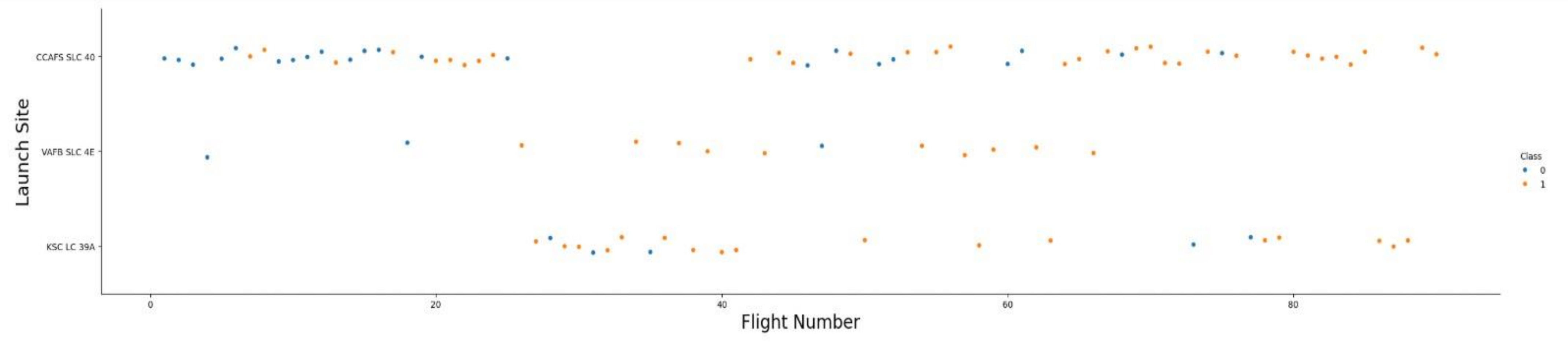
- Exploratory Data Analysis ->
    - Created SpaceX launch dataframe to predict outcome of rocket launch
    - Determined best features for analysis
- Interactive analytics demo in screenshots ->
    - Created interactive Folium maps to view rocket launch sites and successes
    - Created Dashboard to view statistics on launch sites
- Predictive analysis results ->
    - Compared multiple binary classification models
    - Optimized models with GridSearchCV
    - Found best model: Decision Tree Classifier

Section 2

Insights drawn from EDA

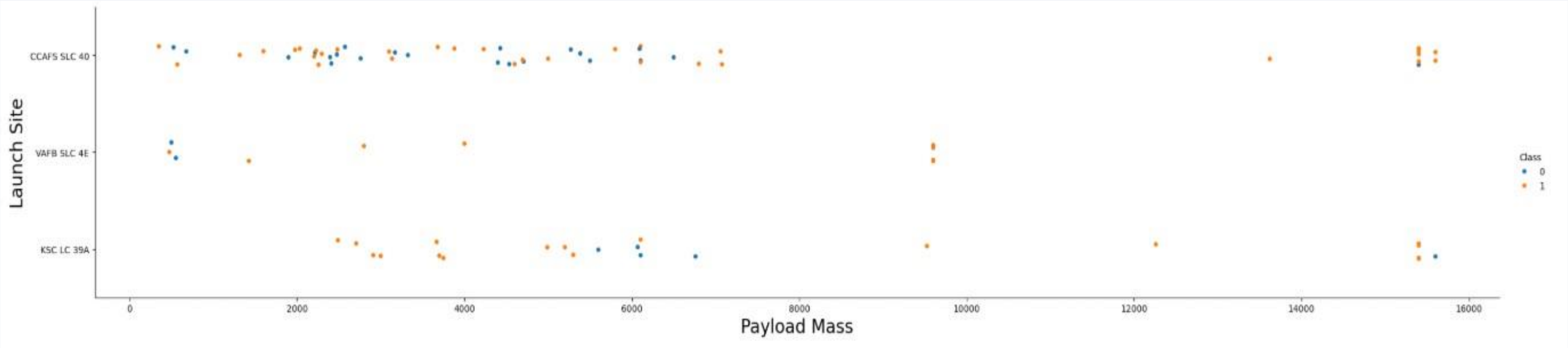
Flight Number vs. Launch Site

- Certain launch sites were out-of-use/not used for flights. ;
- CCAFS SLC 40 has the highest usage and more successful flights more recently
- General success rate improved over flight numbers



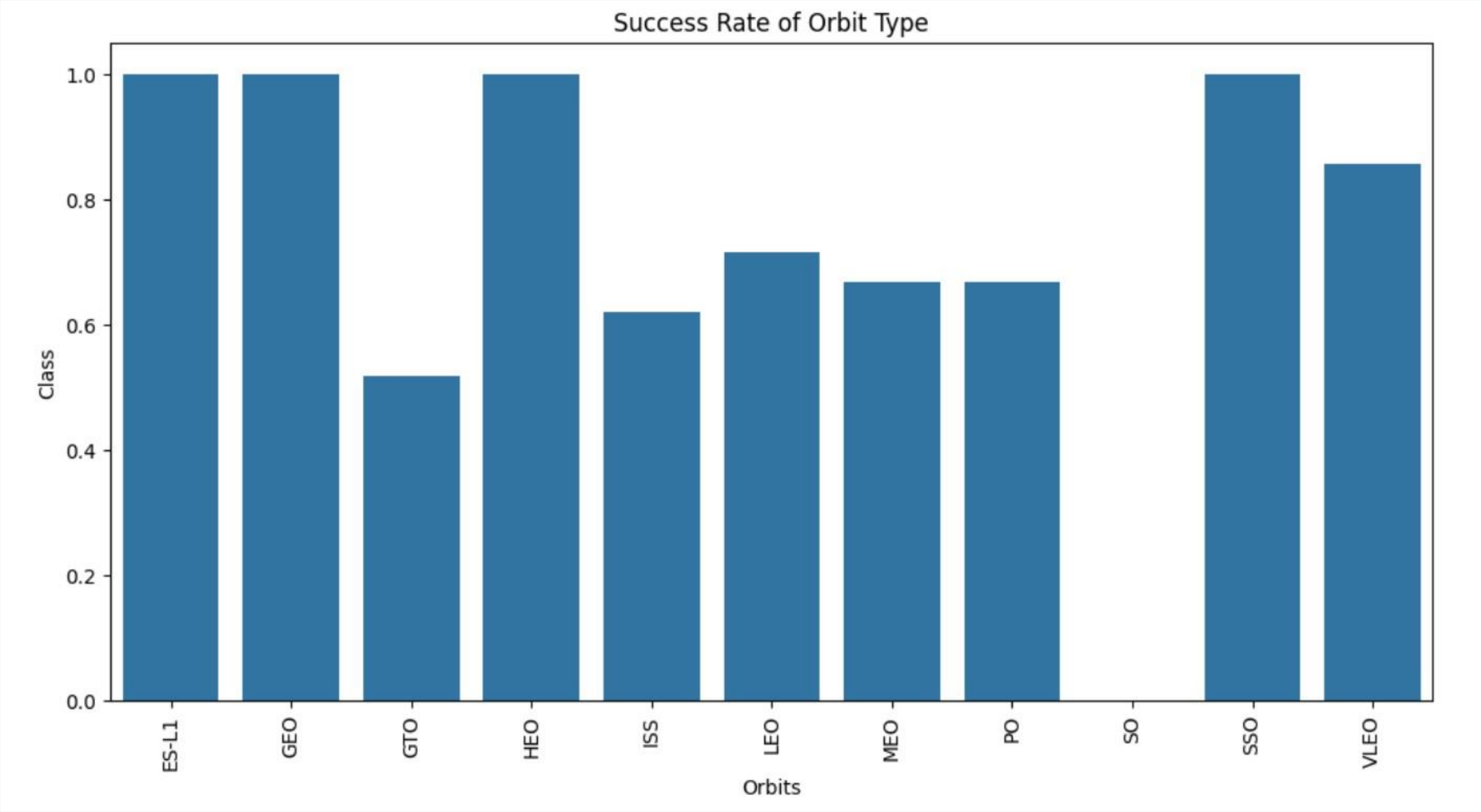
Payload vs. Launch Site

- For VAFB-SLC launchsite, there are no rockets launched for heavypayload mass (greater than 10000)
- Overall, very few rockets launched with heavypayload mass



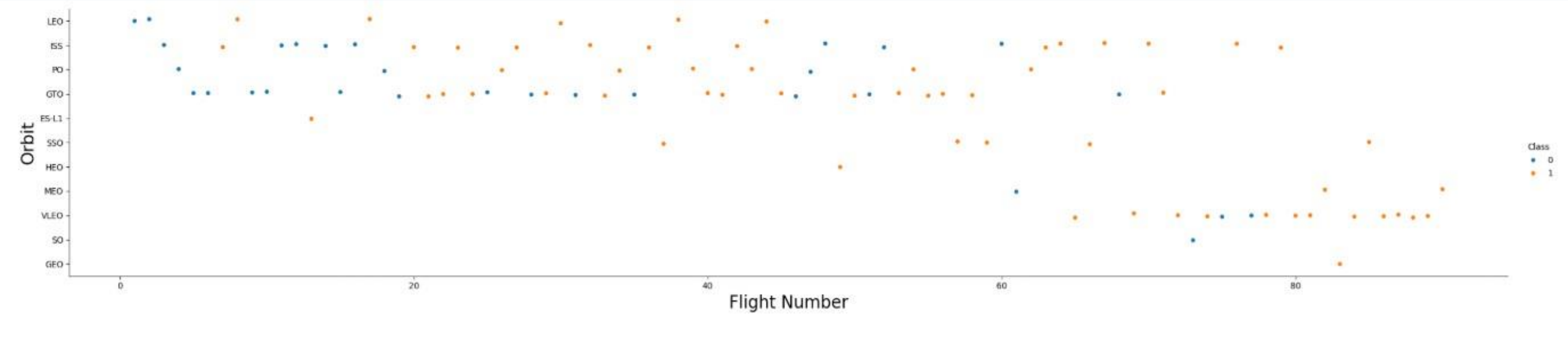
Success Rate vs. Orbit Type

- Orbits with highest success rate: ES-L1, GEO, HEO, SSO



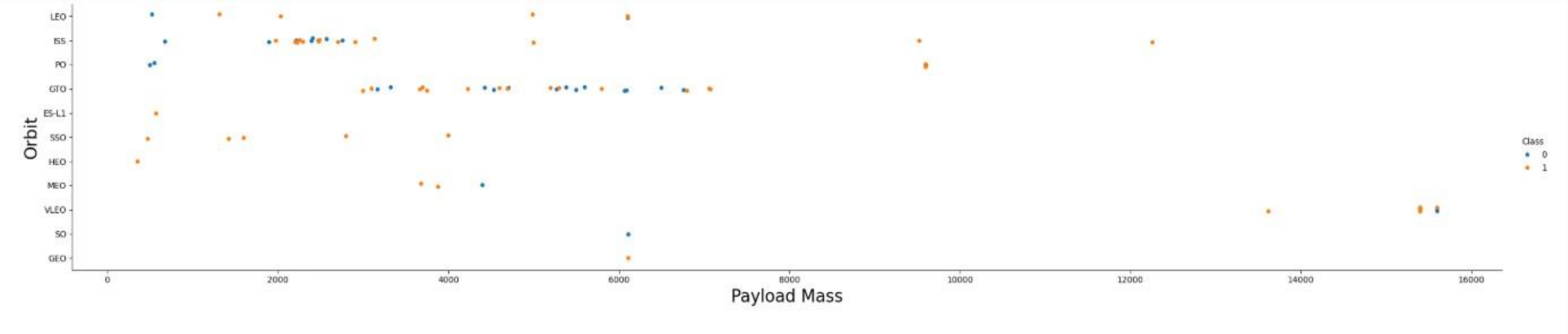
Flight Number vs. Orbit Type

- LEO orbit has a positive relation between success and number of flights
- GTO orbit has no relation between flight number and success
- Certain orbits, like GEO, were not used until recently



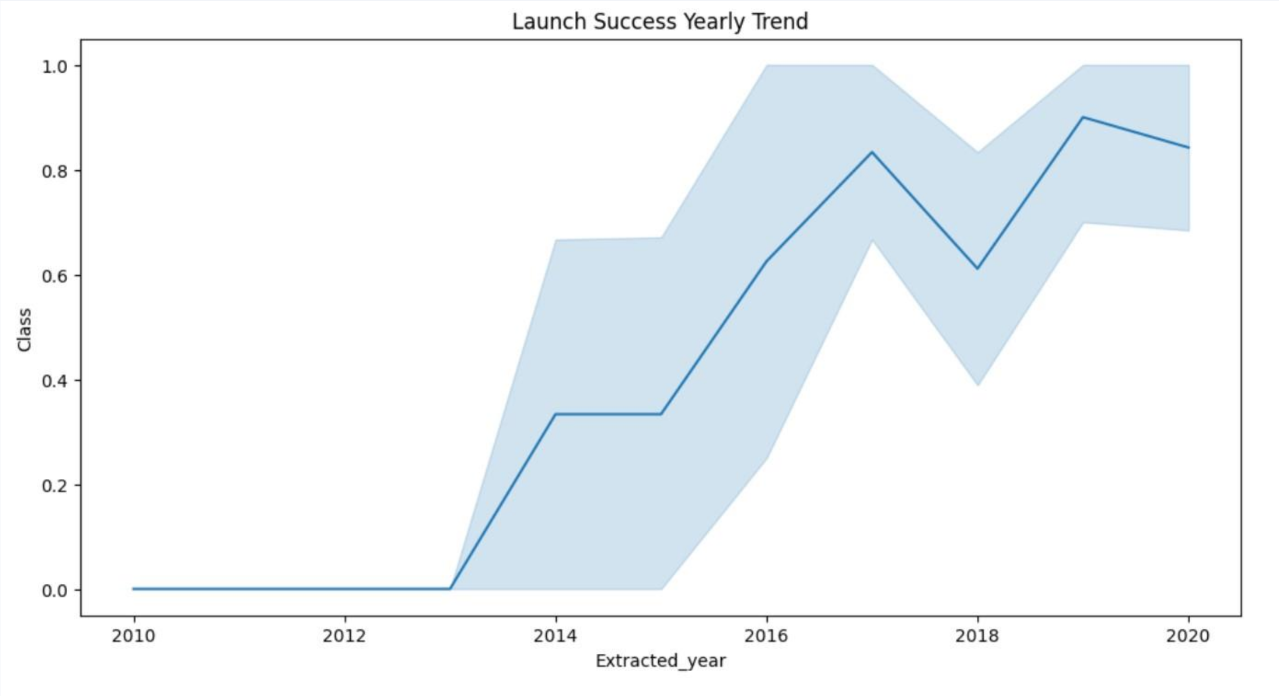
Payload vs. Orbit Type

- With heavy payloads, the successful landing or positive landing rate are more for Polar, LEO, ISS
- For GTO, it is difficult to distinguish between successful and unsuccessful landings as both outcomes are present



Launch Success Yearly Trend

- Success rate kept increasing until a slight dip from 2017 to 2018
- Success rate dropped again from 2019 to 2020, less steep decline this time



All Launch Site Names

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

Launch Site Names Begin with 'CCA'

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	7:44:00	F9 v1.0 B0005	CCAFS LC-40	Dragon demo flight C2	525	LEO (ISS)	NASA (COTS)	Success	No attempt
2012-10-08	0:35:00	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

SUM(PAYLOAD_MASS__KG_)
45596

Average Payload Mass by F9 v1.1

AVG(PAYLOAD_MASS__KG_)
2928.4

First Successful Ground Landing Date

MIN(DATE)
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

Total Number of Successful and Failure Mission Outcomes

COUNT("Mission_Outcome")
101

Boosters Carried Maximum Payload

Booster_Version
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

Month	Year	"Landing_Outcomes"	Booster_Version	Launch_Site
01	2015	Landing_Outcomes	F9 v1.1 B1012	CCAFS LC-40
04	2015	Landing_Outcomes	F9 v1.1 B1015	CCAFS LC-40

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

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

Section 3

Launch Sites Proximities Analysis

Folium Launch Site Locations

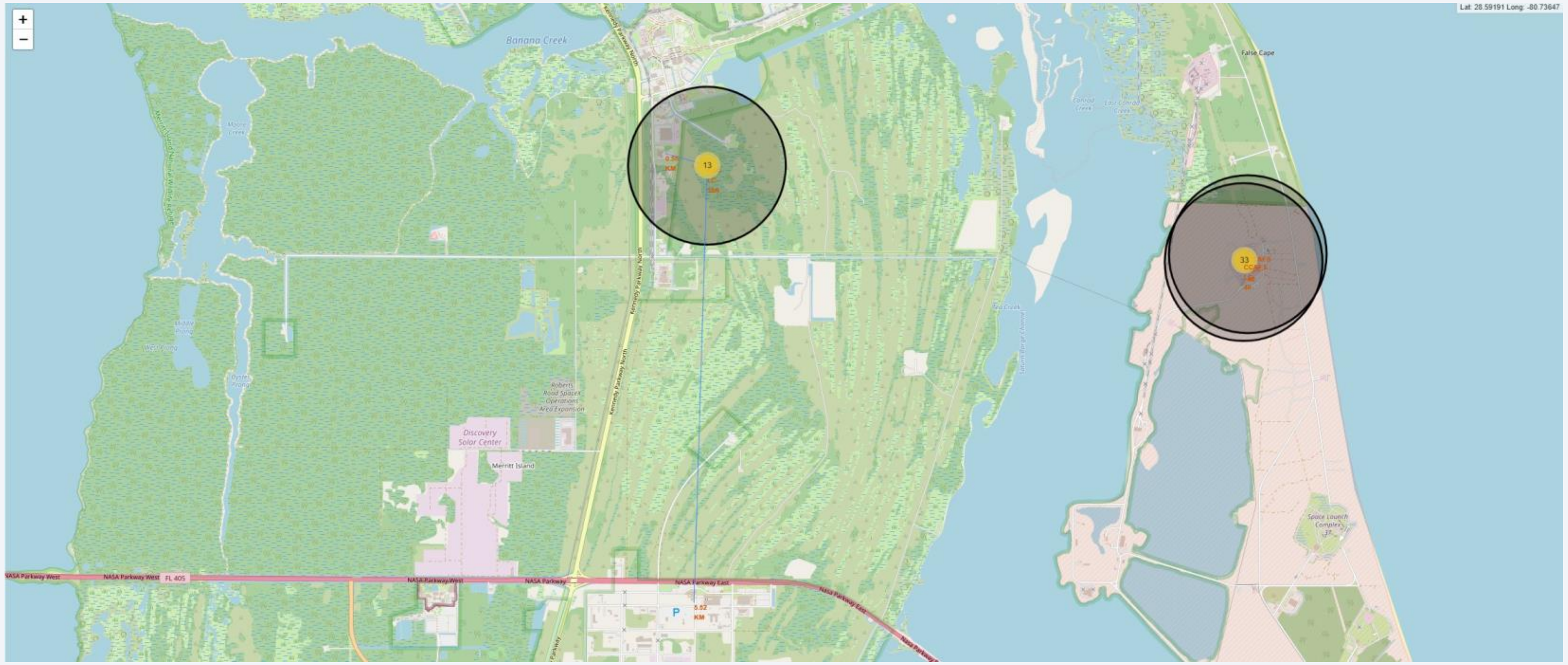
- Launch Sites located on West Coast (California) and East Coast (Florida)

Folium Launch Outcomes

- Showcases successful launches (green) and failed launches (red) for each launch site location

Folium Proximity Map

- Utilized Distance calculations on interactive map for the user to determine the distance between a launch site and various proximities
- Proximities include body of water, major road, building, etc…

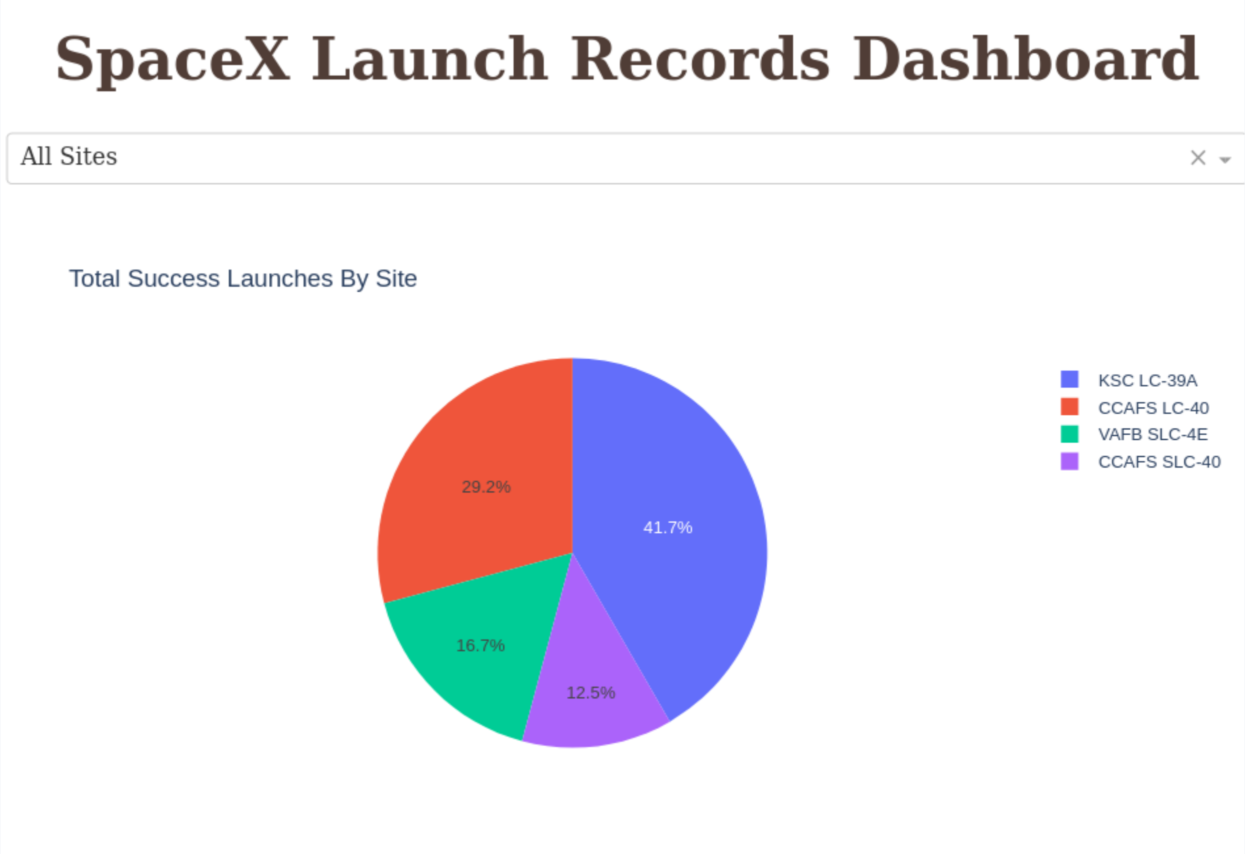


Section 4

Build a Dashboard with Plotly Dash

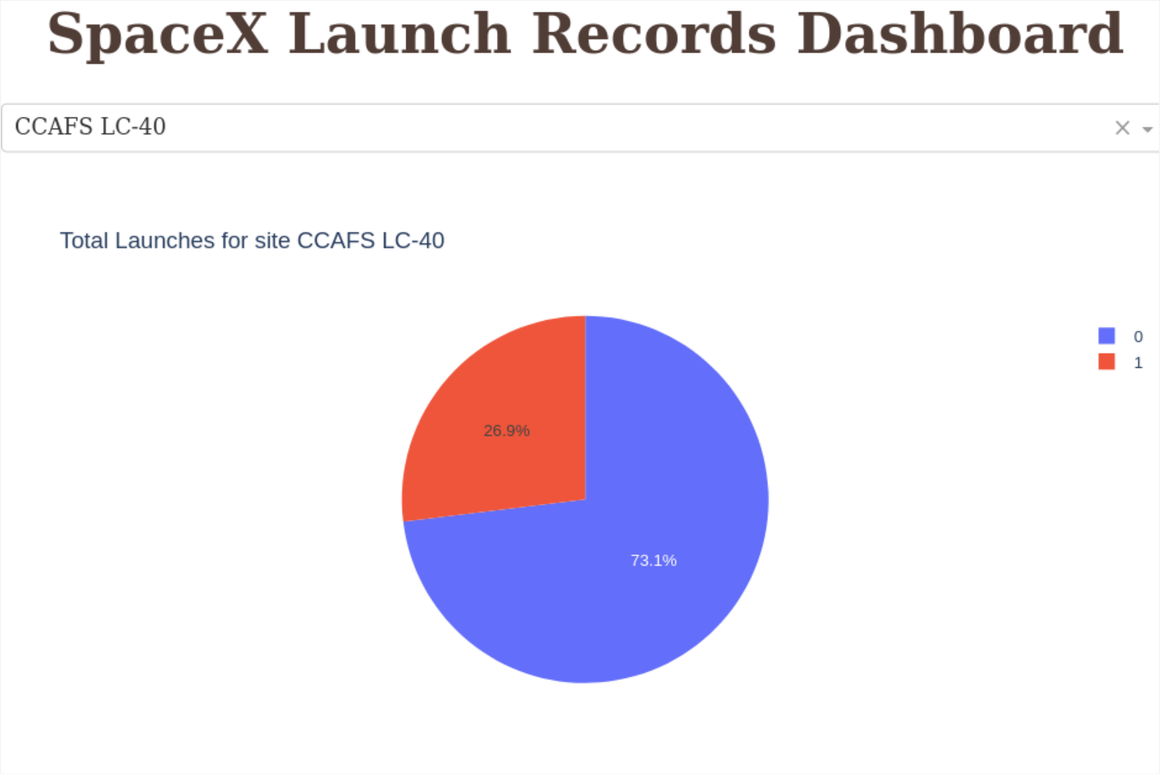
Total Success Launches by Site

- Visualizes the success likelihood of launch sites to determine the “best” launch site



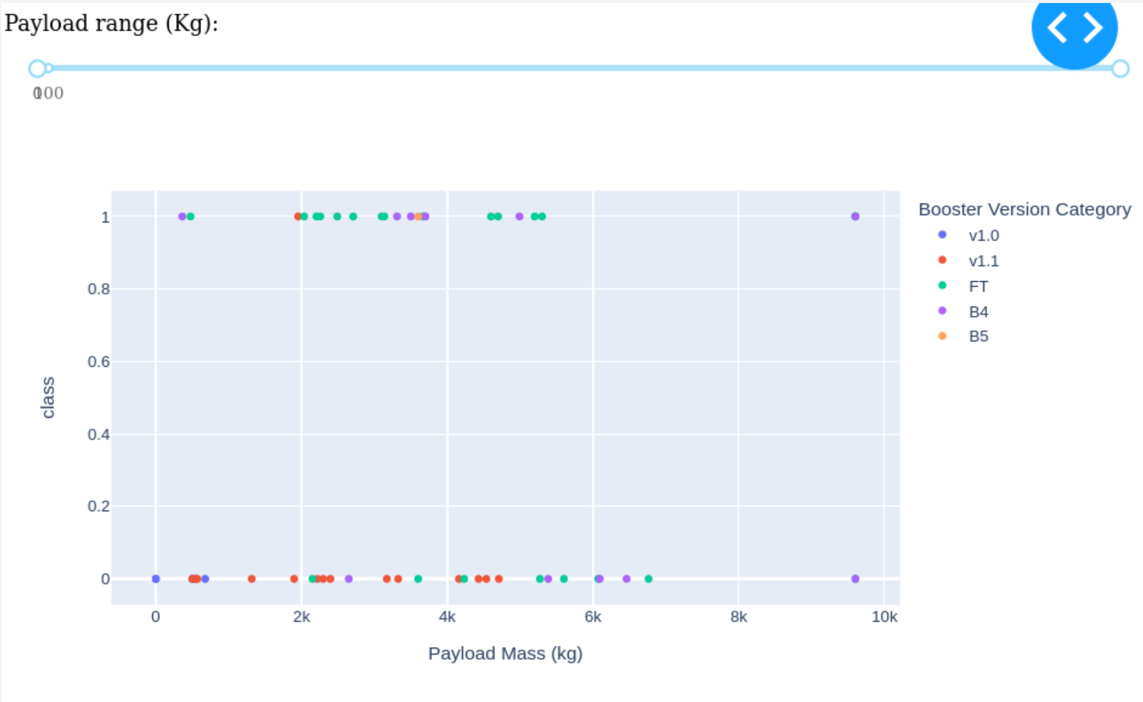
Total Launches for Site CCAFS LC-40

- Visualizes successful launch ratio for the launch site with the best success rate
- 0 -> success
- 1 -> fail



Payload vs Launch Outcome

- Most successful Booster Version: FT
- Most successful Payload Mass: over 7000 kg

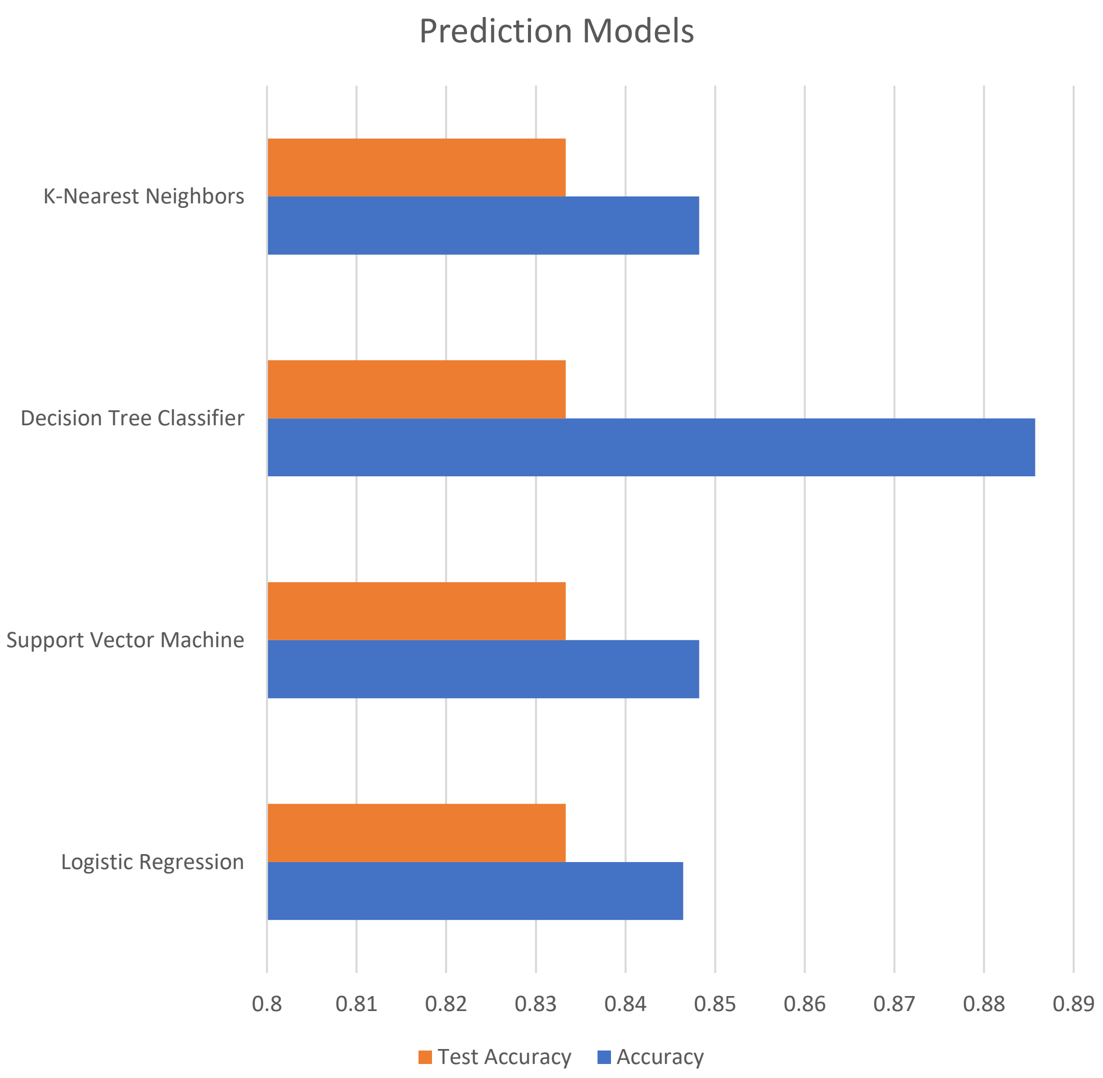


Section 5

Predictive Analysis (Classification)

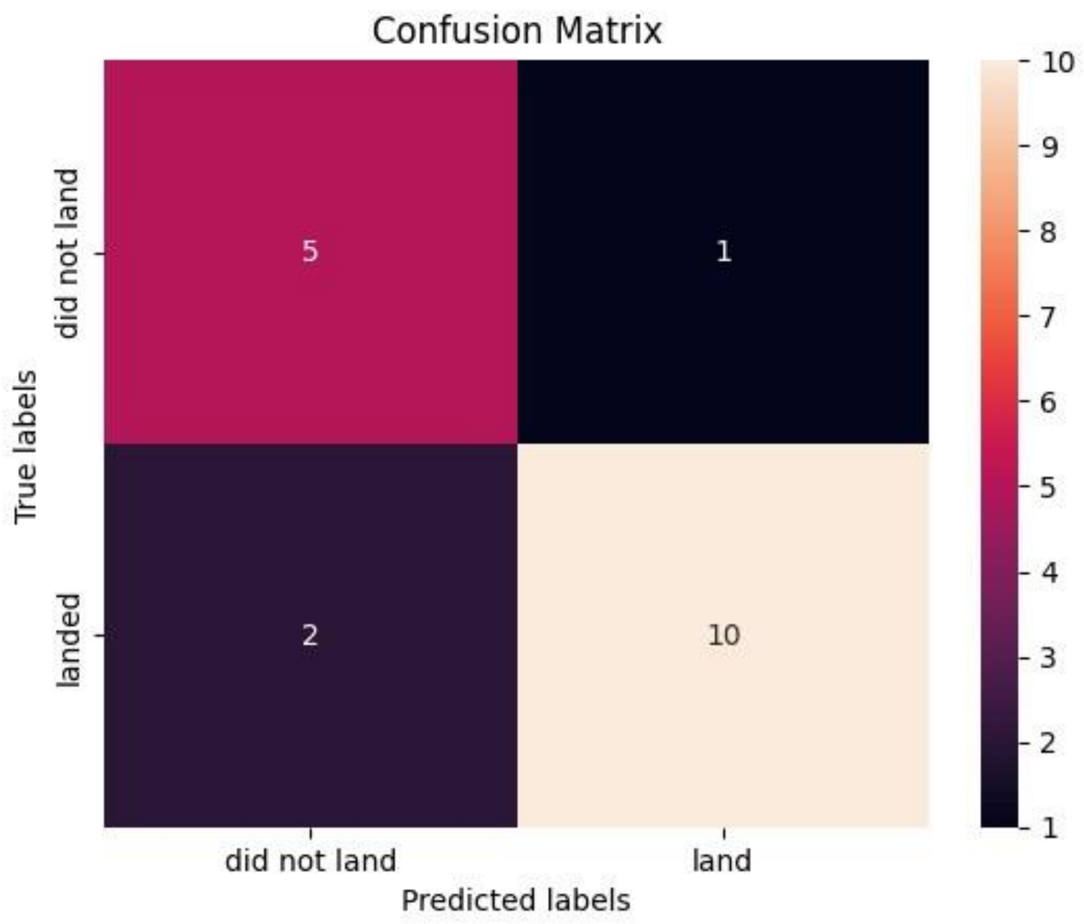
Classification Accuracy

- Decision Tree Classifier had the best model accuracy on the train data
- Test accuracy remained the same across all models
- Issue with amount of data



Confusion Matrix

- Decision Tree Classifier Confusion Matrix
- 3 incorrectly labeled launches
- 15 correctly labeled launches



Conclusions

- Decision Tree Classifier is our best prediction model
    - Refining model necessary to predict successful launches and increase profits
- Launch Site with most success: KSC LC-39A
- Successful landing outcomes increased over time, showcasing improvements of rocket technology and techniques
- Launch Site location details provided with Folium maps

Appendix

- Repository: https://github.com/Blu-LeBlanc/IBM-Data-Science-Labs/tree/main
- *Note: Folium Maps do not render on github

Thank you!