
IBM Developer
SKILLS NETWORK
IBM
Applied
Data Science
Capstone Project
Gustavo Zamboni
April 12th, 2022
SPACEX

Outline

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

Executive Summary

Summary of methodologies

- Data Collection with an API
- Data Collection with Web Scraping
- Data Wrangling
- Exploratory Data Analysis using SQL
- Exploratory Data Analysis using Data Visualization Techniques
- Interactive Visual Analytics using Folium and Dashboards
- Predictive Analysis using classification methods

Summary of all results

- Exploratory Data Analysis results
- Data analysis using Interactive visualization results
- Identify the Best Model to be used for Spacex Predictive Analysis

Introduction

Project background and context

SpaceX advertises Falcon 9 rocket launches on its website with a cost of 62 million dollars; other providers cost upward of 165 million dollars each, much of the savings is because SpaceX can reuse the first stage. if we can determine if the first stage will land, we can determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.

Problems you want to find answers

We were asked to predict if Spacex Falcon 9 first stage will land successfully

To Analyse this problem we need to:

1. Check which are the main features that influence the landing result
2. their relationship the diferente features
3.Which are the best operational conditions to be used

Section 1

Methodology

Methodology

Executive Summary

- **Data collection methodology:**
  - We collected Data from the SpaceX REST API which URL, starts with api.spacexdata.com/v4/
  - We complete the Collection of Falcon 9 Launch data with Web Scraping from wikipedia pages using the Python BeautifulSoup package.
- **Perform data wrangling**

  Data wrangling tasks are the pre-processing phase of data analysis. These tasks included handling coded IDs, handling missing/null values in data, remove Falcon 1 launches formatting data to standardize it and make it consistent, normalizing data, removing not relevant data columns and converting categorical variables into numerical quantitative variables using One hot encoding.

Methodology

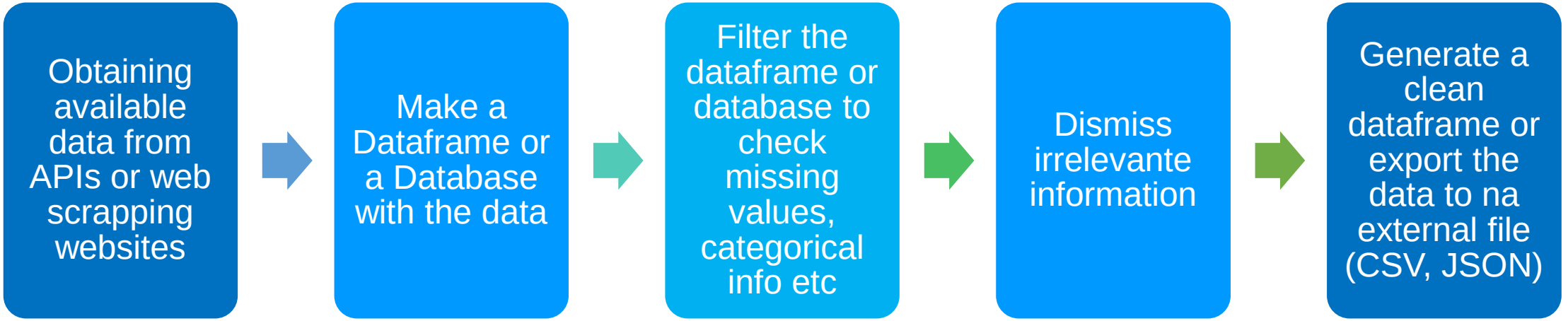
Executive Summary (cont…)

- **Perform exploratory data analysis (EDA) using visualization and SQL**
  - Use of scatter, bar and line graphs to visualy analyse data.
- **Perform interactive visual analytics using Folium and Plotly Dash**
- **Perform predictive analysis using classification models**
  - We first split the data into training data and test data
  - Then we use the training data to find the best Hyperparameters for:
    - Logistic Regression, Support Vector Machine (SVM), Decision Trees and K Nearest Neighbors (KNN) classification models.
  - Then find the classification model that performed best using the test data.

Data Collection - Concepts

Data collection is the process of collecting, measuring and analyzing different types of information using a set of standard validated techniques with the objective of providing rich and reliable data to make critical business decisions

Basic Data Collection Process



Data Collection – with SPACEX Restful API

Request rocket launch data from SpaceX API

- Identify SPACEX API URL - spacex_url="https://api.spacexdata.com/v4/launches/past"
- Request and Parse data using request.get() - response = requests.get(spacex_url)
- Check request success - response.status_code

Recover response data as JSON file and Normalize

- Recovering response data as JSON file - response_json = response.json()
- **Normalize data** - data = pd.json_normalize(response_json)

Remove unnecessary rows

Get information about the launches using the IDs

From column	We'll get
rocket	booster name
payload	Mass, orbit
payload	launch site, longitude, latitude
cores	Outcome, type, number of flights, etc.

Create Dataframe, filter data end export to CSV file

- Create dataframe with data obtained
- Filter Only Falcon 9 launches

Data Collection – with SPACEX Restful API

SPACEX Restful API final dataframe

```
1 falcon9_data.head()
```

BoosterVersion	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
Falcon 9	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
Falcon 9	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
Falcon 9	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
Falcon 9	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.632093
Falcon 9	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004	-80.577366	28.561857

[Data collection with SPACEX Restful API GITHUB file]

Data Collection – with web scraping

Request the Falcon9 Launch Wiki page from its URL

- static_url = https://en.wikipedia.org/w/index.php?title=List_of_Falcon_9_and_Falcon_Heavy_launches&oldid=1027686922
- data = requests.get(static_url).text

Create a BeautifulSoup object from the HTML response

- soup = BeautifulSoup(data, 'html5lib')

Finding the right table

- html_tables=soup.find_all('table')
- first_launch_table = html_tables

Collect all relevant column names from the HTML table header

```
table_columns=first_launch_table.find_all('th')
for i,col in enumerate(table_columns):
    name = extract_column_from_header(col)
    if name is not None and len(name) > 0:
        column_names.append(name)
```

Create a data frame by parsing the launch HTML tables

- create an empty dictionary with keys from the extracted column names => launch_dict = dict.fromkeys(column_names)
- Fill up the launch_dict with launch records extracted from the table rows

Create a data frame from the dictionary

```
df= pd.DataFrame({ key:pd.Series(value) for key, value in launch_dict.items() })
```

Export data set to csv file

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

SPACEX Wikipedia webscraping GITHUB file

Data Collection – with web scraping

SPACEX Wikipedia WEBSCRAPPING final dataframe

```
1 df.head()
```

	Flight No.	Launch site	Payload	Payload mass	Orbit	Customer	Launch outcome	Version Booster	Booster landing	Date	Time
0	1	CCAFS	Dragon Spacecraft Qualification Unit	0	LEO	NaN	Success\n	F9 v1.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NaN	Success	F9 v1.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NaN	Success	F9 v1.1	No attempt\n	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NaN	Success\n	F9 v1.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NaN	Success\n	F9 v1.1	No attempt\n	1 March 2013	15:10

SPACEX Wikipedia webscraping GITHUB file

Data Wrangling - Concept

Data preprocessing is a necessary step in data analysis. It is the process of converting or mapping data from one raw form into another format to make it ready for further analysis. Data preprocessing is often called data cleaning or data wrangling.

Five important items in data Wrangling are:

1. Identify and handle missing values: This is when a data entry is left empty.
2. Data formats. Data from different sources usually have different formats.
3. Data normalization: Is a way to bring all data into a similar *range.*
4. Data binning: It is particularly useful for comparison between groups of data.
5. Categorical variables: convert categorical values into numeric variables to make statistical modeling easier

Data wrangling GITHUB file

Data Wrangling Process

Load dataset saved in our Lab Data Collection API and create a pandas dataframe

- url = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_1.csv"
- df=pd.read_csv(url)

Calculate the number of launches on each site

```
df['LaunchSite'].value_counts()
```

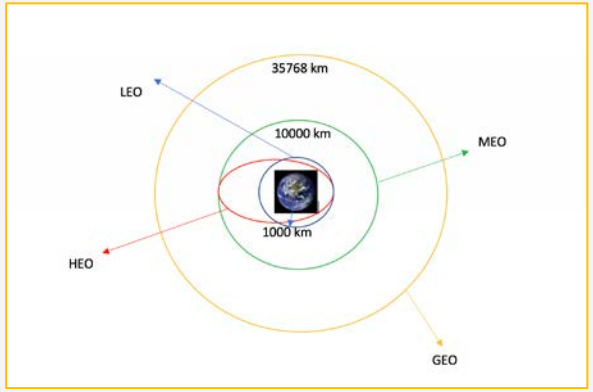
Launch site	Number of launches
CCAFS SLC 40	55
KSC LC 39A	22
VAFB SLC 4E	13

- Cape Canaveral Space Launch Complex 40 **VAFB SLC 4E**
- Vandenberg Air Force Base Space 4E **(SLC-4E)**
- Kennedy Space Center Launch Complex 39A **KSC LC 39A**

Calculate the number and occurrence of each orbit

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

Orbit	Occurences
GTO	27
ISS	21
VLEO	14
PO	9
LEO	7
SSO	5
MEO	3
ES-L1	1
HEO	1
SO	1
GEO	1



- **LEO**: Low Earth orbit (LEO)is an Earth-centred orbit with an altitude of 2,000 km (1,200 mi) .
- **VLEO**: Very Low Earth Orbits can be defined as the orbits with a mean altitude below 450 km.
- **GTO** A geosynchronous orbit is a high Earth orbit that allows selatlites to match Earth's rotation.
- **SSO (or SO)**: It is a Sun-synchronous orbit also called a heliosynchronous orbit is a nearly polar orbit around a planet..
- **ES-L1** :At the Lagrange points the gravitational forces of the two large bodies cancel .
- **HEO** A highly elliptical orbit, is an elliptic orbit with high eccentricity.
- **ISS** A modular space station (habitable artificial satellite) in low Earth orbit.
- **MEO** Geocentric orbits ranging in altitude from 2,000 km to just below 35,786 kilometers.
- **HEO** Geocentric orbits above the altitude of geosynchronous.
- **GEO** It is a circular geosynchronous orbit 35,786 kilometres above Earth's.
- **PO** It is one type of satellites in which a satellite passes above or nearly above both .

Continue

Data wrangling GITHUB file

Data Wrangling Process

Calculate the number and occurence of mission outcome per orbit type

```
landing_outcomes = df['Outcome'].value_counts()
```

Outcome	Occurences
True ASDS	41
None None	19
True RTLS	14
False ASDS	6
True Ocean	5
False Ocean	2
None ASDS	2
False RTLS	1

- True Ocean means the mission outcome was successfully landed to a specific region of the ocean while
- False Ocean means the mission outcome was unsuccessfully landed to a specific region of the ocean.
- True RTLS means the mission outcome was successfully landed to a ground pad
- False RTLS means the mission outcome was unsuccessfully landed to a ground pad.
- True ASDS means the mission outcome was successfully landed to a drone ship
- False ASDS means the mission outcome was unsuccessfully landed to a drone ship.
- None ASDS and None None these represent a failure to land.

Create a landing outcome label from Outcome column

```
for outcome in df['Outcome']:
    if outcome in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
df['Class']=landing_class
```

Export data set to csv file

```
df.to_csv('spacex_web_scraped.csv', index=False)
```

Data wrangling GITHUB file

Data Wrangling Process

Data wrangling final dataframe

```
: 1 df.head(10)
```

:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude	Class
0	1	2010-06-04	Falcon 9	6104.959412	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0003	-80.577366	28.561857	0
1	2	2012-05-22	Falcon 9	525.000000	LEO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0005	-80.577366	28.561857	0
2	3	2013-03-01	Falcon 9	677.000000	ISS	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B0007	-80.577366	28.561857	0
3	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	NaN	1.0	0	B1003	-120.610829	34.632093	0
4	5	2013-12-03	Falcon 9	3170.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1004	-80.577366	28.561857	0
5	6	2014-01-06	Falcon 9	3325.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1005	-80.577366	28.561857	0
6	7	2014-04-18	Falcon 9	2296.000000	ISS	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1006	-80.577366	28.561857	1
7	8	2014-07-14	Falcon 9	1316.000000	LEO	CCAFS SLC 40	True Ocean	1	False	False	True	NaN	1.0	0	B1007	-80.577366	28.561857	1
8	9	2014-08-05	Falcon 9	4535.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1008	-80.577366	28.561857	0
9	10	2014-09-07	Falcon 9	4428.000000	GTO	CCAFS SLC 40	None None	1	False	False	False	NaN	1.0	0	B1011	-80.577366	28.561857	0

Data wrangling GITHUB file

EDA with Data Visualization – Considerations

- We need to see if the data can be used to automatically to determine if the Falcon 9's second stage will land successfully. So we need to know which are the right attributes to be consider and the correlation between them.
- For example:
  - We observe that the success rate increased since 2013.
  - We see that different launch sites have different success rates. While CCAFS LC-40 has a success rate of 60%, KSC LC-39A and VAFB SLC 4E have a success rate of around 77%.
  - Combining attributes also gives us interesting information:
    1. CCAFS LC-40, has a success rate of 60%, but with masses above 10,000 kg the success rate is 100%.
    2. In the lab we combined multiple features and determined which attributes were correlated with successful landings.
    3. Finaly using One Hot Encoding we converted categorical variables to numbers to facilitate the use for machine learning model that will predict if the first stage will successfully land.

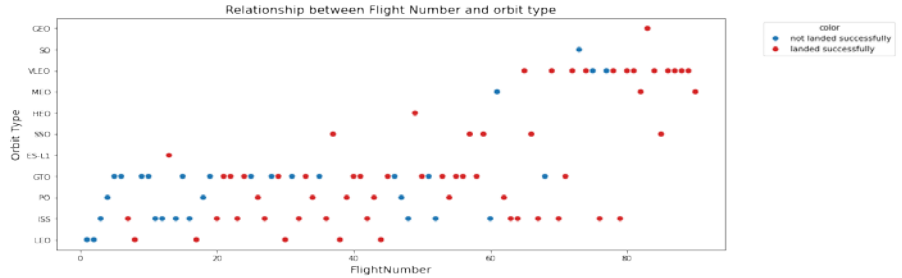
EDA with Data Visualization GITHUB file

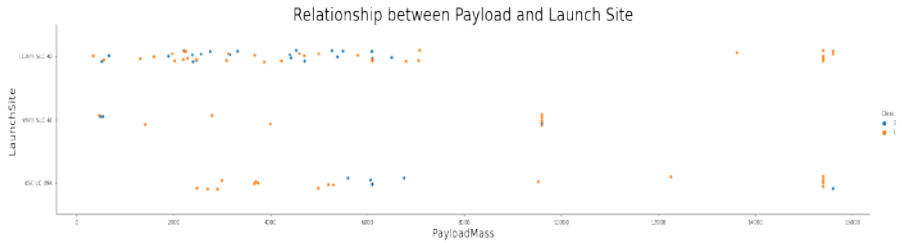
EDA with Data Visualization

Scatter Graph analysis

1. Relationship between Flight Number and Payload
2. Relationship between Flight Number and Launch Site.
3. Relationship between Payload and Launch Site.
4. Relationship between FlightNumber and Orbit type
5. Relationship between Payload and Orbit type

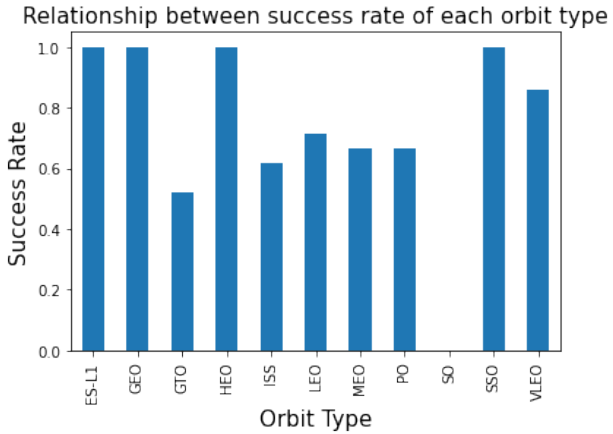
Scatter plots are important in statistics because they can show the correlation between variables, If no correlation exists between the variables, the points appear randomly scattered on the coordinate plane. If a large correlation exists, the points concentrate near a straight line

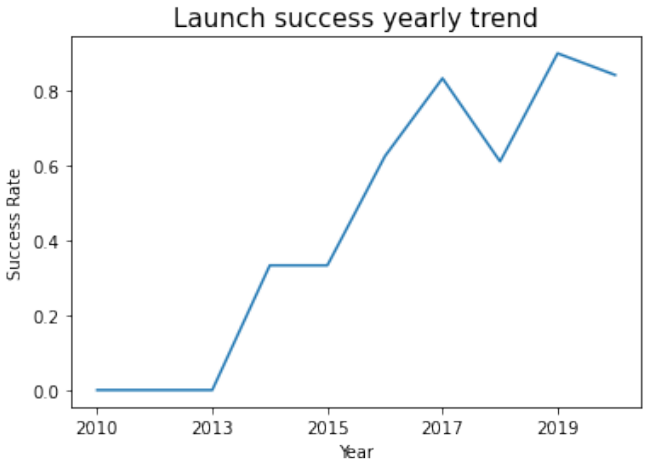




Bar graph

Bar graphs are used to compare and contrast numbers, frequencies or other measures of distinct categories of data.
In the lab using this graph we could easily see which orbits have the highest success rates.





Line graph

A line graph is a type of chart used to show information that changes over time
In the lab using this graph we could easily verify the positive trend of the Launch success rate through the years

EDA with Data Visualization GITHUB file

EDA with SQL

- Data science is all about data, collecting it, cleaning it, analyzing it, visualizing it, and using it but handling large amounts of data can be a challenging task for data scientists. Here where databases come to play. A database is defined as a structured set of data held in a computer's memory or on the cloud that is accessible in various ways.
- In this lab we performed SQL queries to extract information to answer Launch questions:
  1. ***Display the names of the unique launch sites in the space mission***
  2. ***Display 5 records where launch sites begin with the string 'CCA'***
  3. ***Display the total payload mass carried by boosters launched by NASA (CRS)***
  4. ***Display average payload mass carried by booster version F9 v1.1***
  5. ***List the date when the first successful landing outcome in ground pad was acheived***
  6. ***List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000***
  7. ***List the total number of successful and failure mission outcomes***
  8. ***List the names of the booster_versions which have carried the maximum payload mass. Use a subquery***
  9. ***List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015***
  10. ***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***

EDA with SQL GITHUB file

Build an Interactive Map with Folium

After the exploratory data analysis labs, we have discovered some preliminary correlations between the launch site and success rates. With FOLIUM we performed more interactive visual analytics of the data. And here we:

- Mark all launch sites on a map
- Mark the success/failed launches for each site on the map
- Calculate the distances between a launch site to its proximities

Object	Code	why we added these objects
Map	folium.Map()	create a base map
Circle	folium.Circle()	Create a circle arround a specific location
popup	folium.popup()	Creates a popup and tooltip
Marker	folium.map.Marker()	Shows a location marker
Marker Cluster	MarkerCluster()	Creates a cluster of markers
Mouse Position	MousePosition()	Indicates the position (x,y) of the mouse on the map
PolyLine	folium.PolyLine()	Creates a polyline in a map
Icon	folium.icon()	Creates na Icon on a map

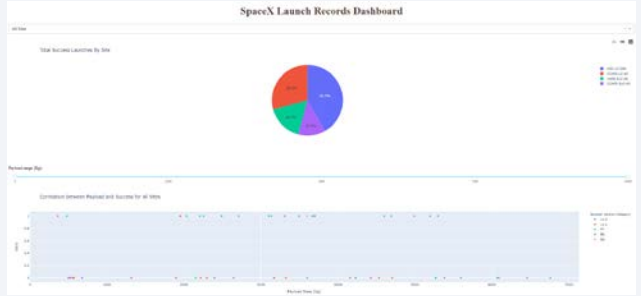
Interactive Map with Folium GITHUB file

Build a Dashboard with Plotly Dash

Developed a dashboard application containing a dropdown list and a range slider to interact with a pie chart and a scatter plot chart. In this way we added:

1. We added a dropdown list to enable Launch Site selection
2. We draw a pie chart showing the total successful launches count for all sites and selecting a specific launch site it shows the Success vs. Failed counts for the site
3. We added a slider to select payload range

Object	Code	why we added these objects
Dropdown list	dcc.Dropdown()	Dropdown List to enable Launch Site selection
Pie Chart	px.pie()	Create a pie chart to shoe launch sites success information
range slider	dcc.RangeSlider()	Slider to select payload range
Scatter Chart	px.scatter()	Shows a location marker



Dashboard with plotly Dash GITHUB file

Predictive Analysis (Classification)

Load data from CSV file

url= https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_2.csv

data = pd.read_csv(url)

Load features data

url= https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DS0321EN-SkillsNetwork/datasets/dataset_part_3.csv

X = pd.read_csv(url)

Create a NumPy array from the column Class in data and assign it to the variable Y

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

Standardize the data in X then reassign it to the variable X using the transform

transform = preprocessing.StandardScaler()

X = transform.fit_transform(X)

Split the data into training and testing data using `train_test_split`

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

Evaluate four classification models

Logistic regression | SVM | Decision tree | KNN

For each one of the models we will do:

- create a GridSearchCV
- Fit the object to find the best parameters
- Calculate the ACCURACY

Calculate the accuracy on the test data using the method score

Lets look at the confusion matrix

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

Classification Model	Accuracy
KNN	0.848214
Decision Tree	0.901786
Logistic Regression	0.846429
SVM	0.848214

BEST MODEL

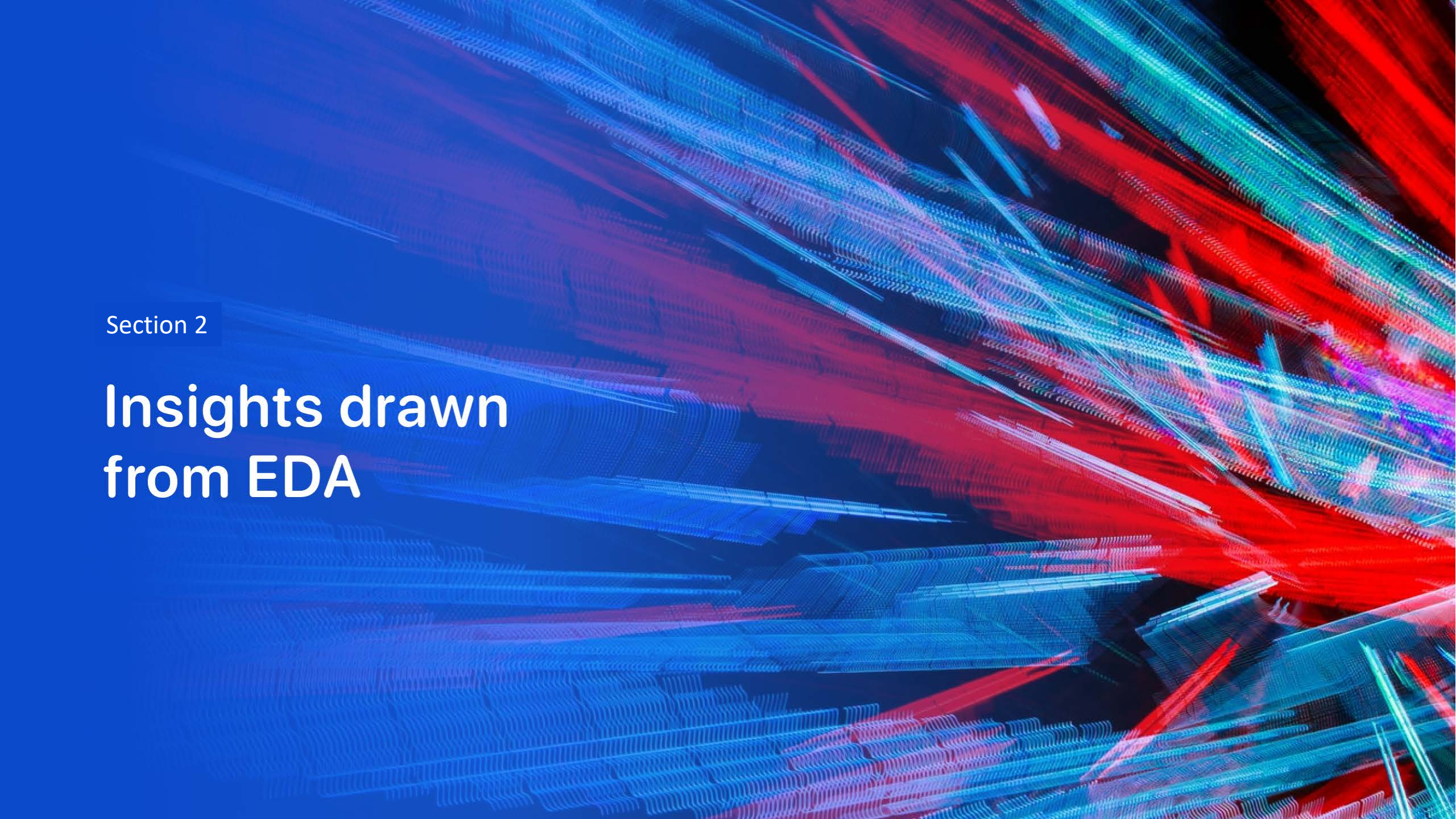
Predictive analysis GITHUB file

Results

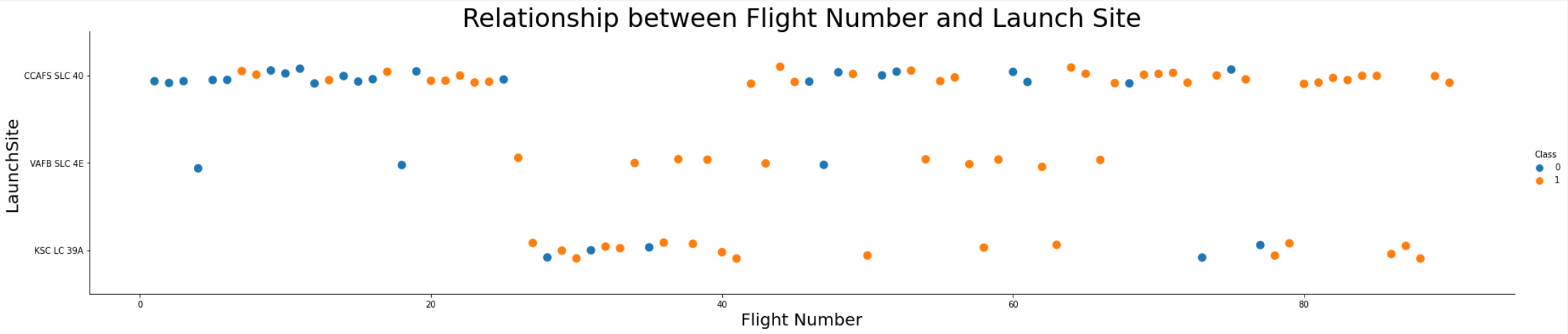
- Exploratory data analysis results
- Interactive analytics demo in screenshots
- Predictive analysis results

Section 2

Insights drawn from EDA

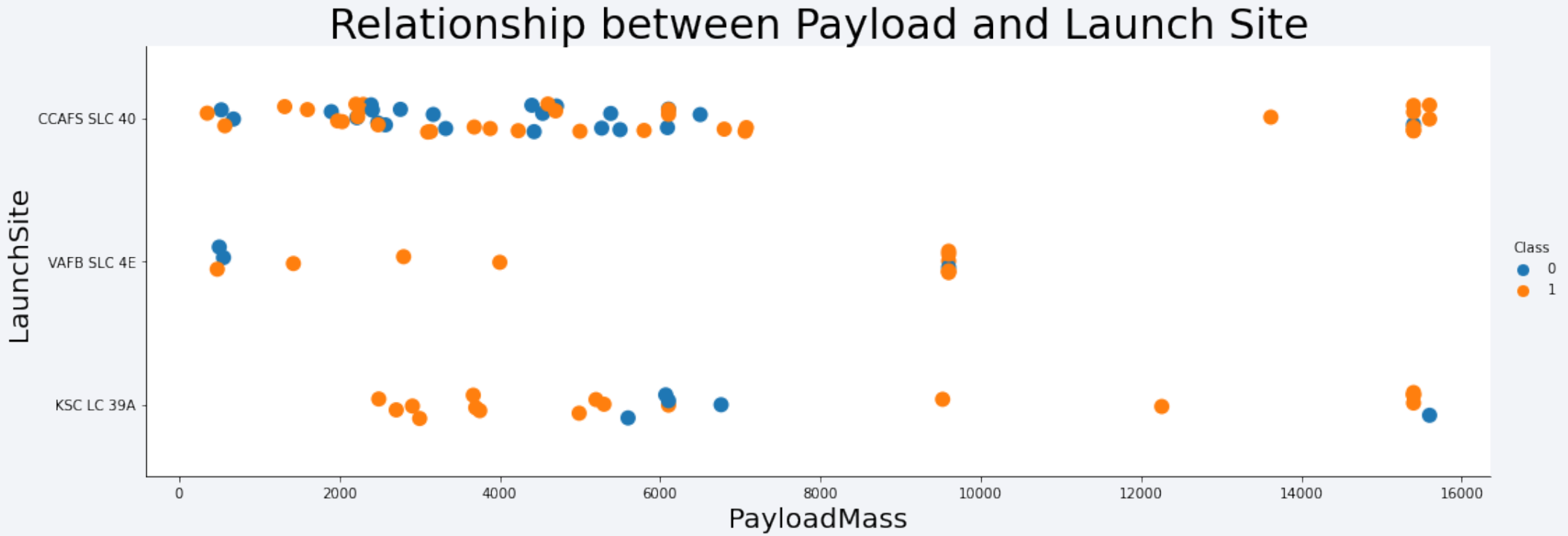


Flight Number vs. Launch Site



From the scatter graph we verify that as long as it increases the number of flights we get a higher success rate at the launch site.

Payload vs. Launch Site

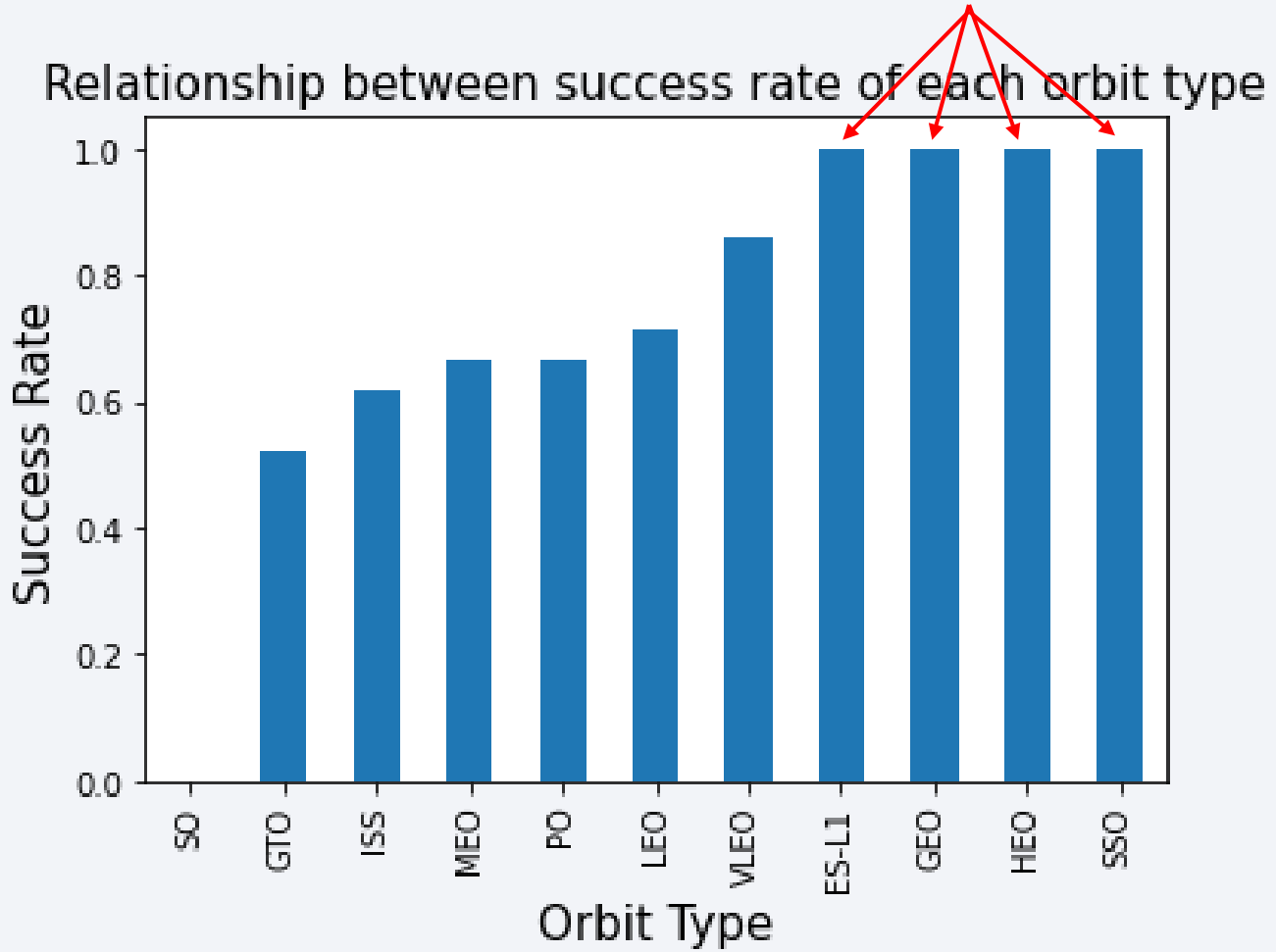


As far as the Payload mass increases we get a higher success rate at the launch site.

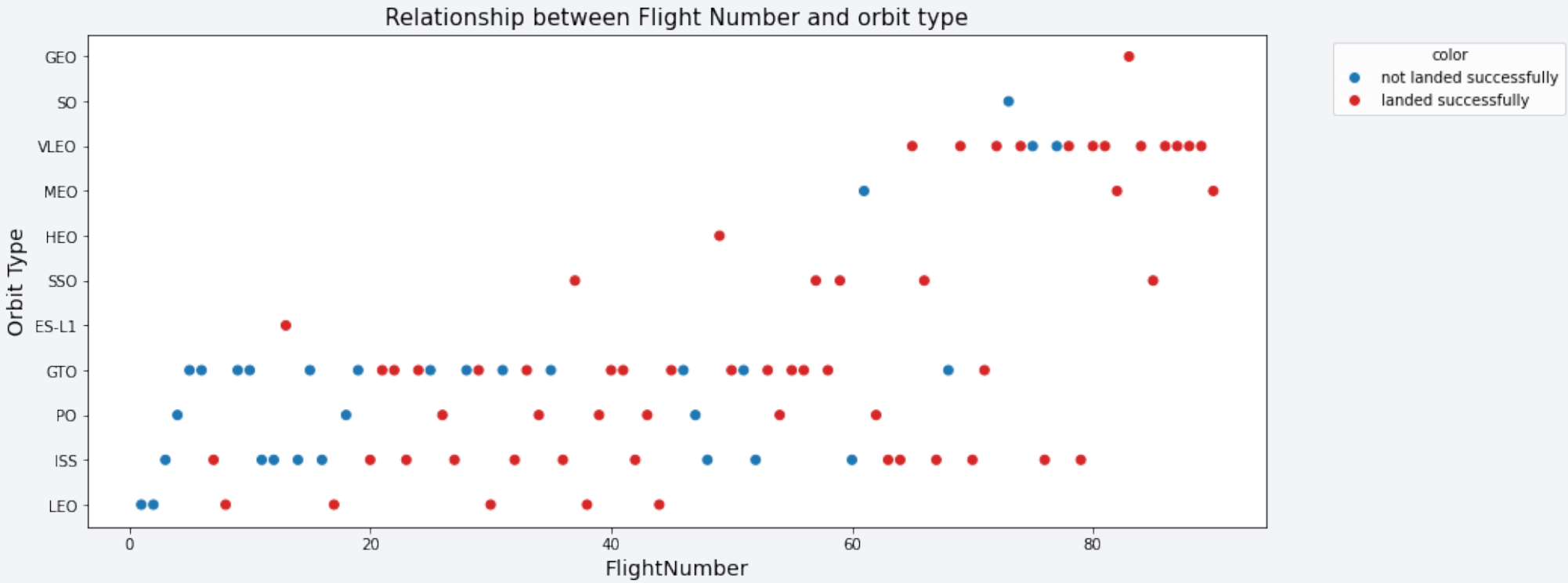
We can see that VAFB SLC 4E does not uses payload mass higher than 10,000 Kg while Kennedy and Cape Canaveral goes up to 16,000kg.

Success Rate vs. Orbit Type

ES-L1, GEO,HEO and SSO are the ones with higher success rates

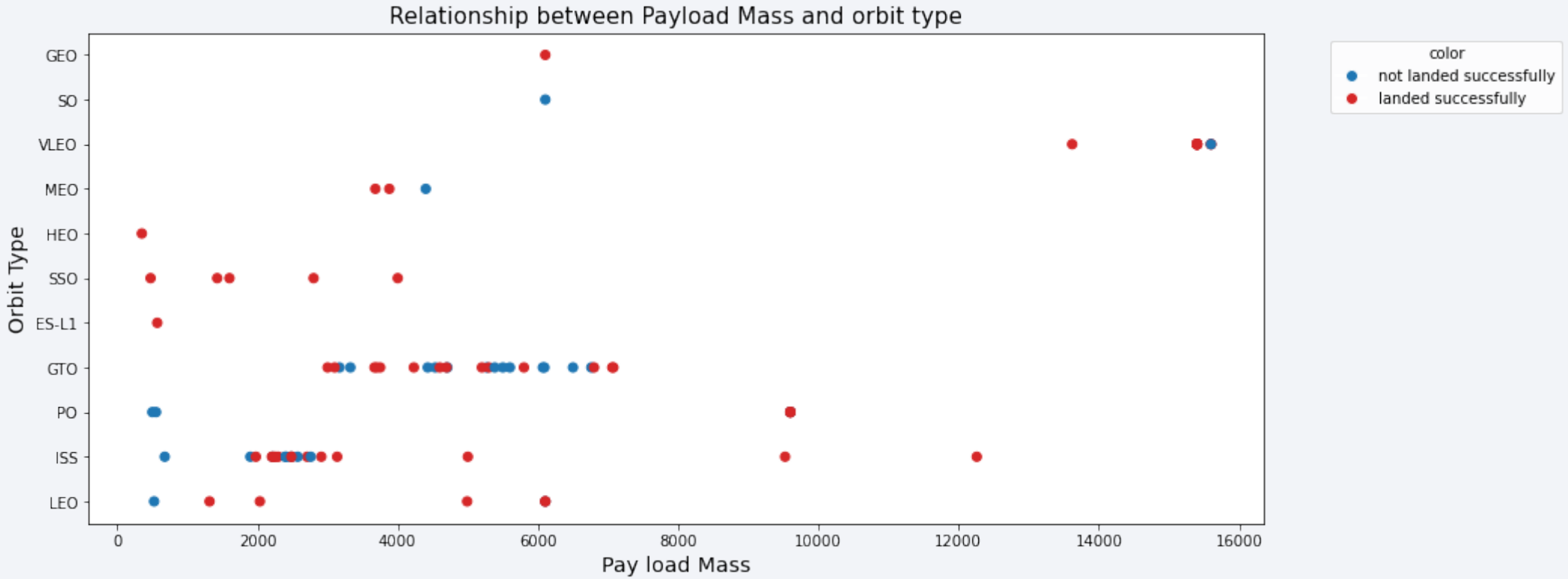


Flight Number vs. Orbit Type



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

We can observe that the success rate since 2013 kept increasing till 2020

All Launch Site Names

```
1 %sql select distinct(LAUNCH_SITE) from SPACEXTBL
```

Result

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

Description

Using the word **DISTINCT** in our query we retrieve, from the SpaceX table “SPACEXTBL”, only the unique launch site names from the column “LAUNCH_SITE”.

Launch Site Names Begin with 'CCA'

```
1 %sql select * from SPACEXTBL WHERE LAUNCH_SITE like 'CCA%' limit 5
```

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

Description

Using the keyword **LIMIT 5** we are only retrieving 5 records from the table SPACEXTBL where the condition is defined by the keyword **LIKE** and wildcard “CCA%”. The % symbol at the end mean that the result must include all the information beginning with CCA and having anything after it.

Total Payload Mass

```
1  %sql select SUM (PAYLOAD_MASS__KG_) from spacextbl where customer = 'NASA (CRS)'
```

Result

1
45596

Description

The function SUM will retrieve the total sum of the column “PAYLOAD_MASS_KG_” from the table SPACEXTBL where the customers are ”NASA (CRS).

Average Payload Mass by F9 v1.1

```
1  %sql select avg(PAYLOAD_MASS__KG_) from spacextbl where BOOSTER_VERSION = 'F9 v1.1'
```

Result

1
2928

Description

The function **AVG** calculates the average of all the values in column PAYLOAD_MASS_KG_ that satisfies the where expression filter that only include the values of BOOSTER_VERSIONS = **F9 v1.1**

First Successful Ground Landing Date

```
1 %sql select min(DATE) from spacextbl where LANDING__OUTCOME = 'Success (ground pad)'
```

Result

1
2015-12-22

Description

Using the function **MIN** the execution of the sql query will retrieve the first date of the table where the column LANDING__OUTCOME = ‘Success (ground pad)’

Successful Drone Ship Landing with Payload between 4000 and 6000

```
1 %sql select BOOSTER_VERSION from spacextbl where LANDING__OUTCOME = 'Success (drone ship)' and PAYLOAD_MASS__KG_ between
  4000 and 6000
```

Result

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

Description

We selected the BOOSTER_VERSION from table SPACEX with two **where** clauses:

The first clause specifies that the column LANDING_OUTCOME should be “Success (drone ship)”

and the second clause specifies that the column PAYLOAD_MASS__KG_ must be between 4000 and 6000 kg

Total Number of Successful and Failure Mission Outcomes

```
1 %sql select MISSION_OUTCOME as Results, count(MISSION_OUTCOME) as Quantity from spacextbl group by MISSION_OUTCOME
```

Result

results	quantity
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Description

In this query we select the MISSION_OUTCOME and count them with the clause of grouping them by the MISSION_OUTCOME

It shows that we have 100 successful outcomes but one of them has an unclear status of the payload.

Boosters Carried Maximum Payload

```
%sql select BOOSTER_VERSION, PAYLOAD_MASS__KG_ from spacextbl where PAYLOAD_MASS__KG_ = (select Max(PAYLOAD_MASS__KG_) from spacextbl)
```

booster_version	payload_mass__kg_
F9 B5 B1048.4	15600
F9 B5 B1049.4	15600
F9 B5 B1051.3	15600
F9 B5 B1056.4	15600
F9 B5 B1048.5	15600
F9 B5 B1051.4	15600
F9 B5 B1049.5	15600
F9 B5 B1060.2	15600
F9 B5 B1058.3	15600
F9 B5 B1051.6	15600
F9 B5 B1060.3	15600
F9 B5 B1049.7	15600

Description

We are retrieving the Booster_version and the Payload_mass__kg_ with a where clause using a subquery and specifying that the PAYLOAD_MASS_KG__ should be selected only for those that have the maximum PAYLOAD_MASS__KG_ (function MAX) from the SpaceX table.

2015 Launch Records

```
1 %sql select BOOSTER_VERSION, LANDING__OUTCOME, LAUNCH_SITE, Date from spacextbl where LANDING__OUTCOME = 'Failure (drone
  ship)' and year(Date) = '2015'
```

booster_version	landing__outcome	launch_site	DATE
F9 v1.1 B1012	Failure (drone ship)	CCAFS LC-40	2015-01-10
F9 v1.1 B1015	Failure (drone ship)	CCAFS LC-40	2015-04-14

Description

We selected the Booster_Version, Landing_outcome and date from the SpaceXtbl with a where clause defining two conditions

First one: Landing_outcome should be ‘Failure (drone ship)’

Second condition: the year should be ‘ 2015’. This was done using the function year over the column DATE

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

```
%sql select LANDING__OUTCOME, COUNT(LANDING__OUTCOME) as total from spacextbl where date >='2010-06-04' and date<='2017-03-20' group by LANDING__OUTCOME order by COUNT(LANDING__OUTCOME) desc
```

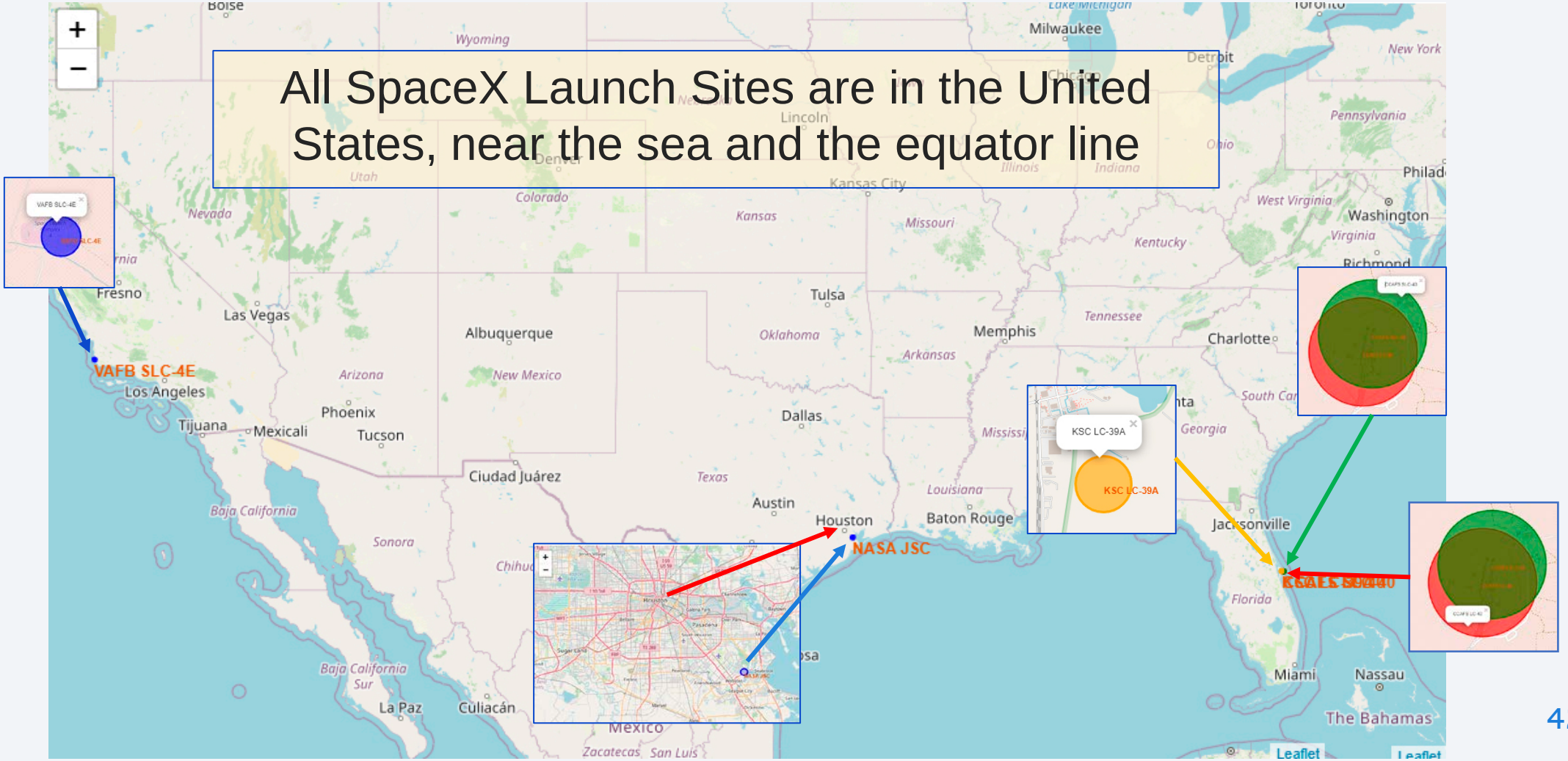
landing__outcome	total
No attempt	10
Failure (drone ship)	5
Success (drone ship)	5
Controlled (ocean)	3
Success (ground pad)	3
Failure (parachute)	2
Uncontrolled (ocean)	2
Precluded (drone ship)	1

Description

- We selected Landing outcomes and the **COUNT** of them from the spacextbl and used a **WHERE** clause to filter for landing outcomes **BETWEEN** 2010-06-04 to 2010-03-20.
- We applied the **GROUP BY** clause to group the landing outcomes and the **ORDER BY** clause to order the grouped landing outcome in descending order.

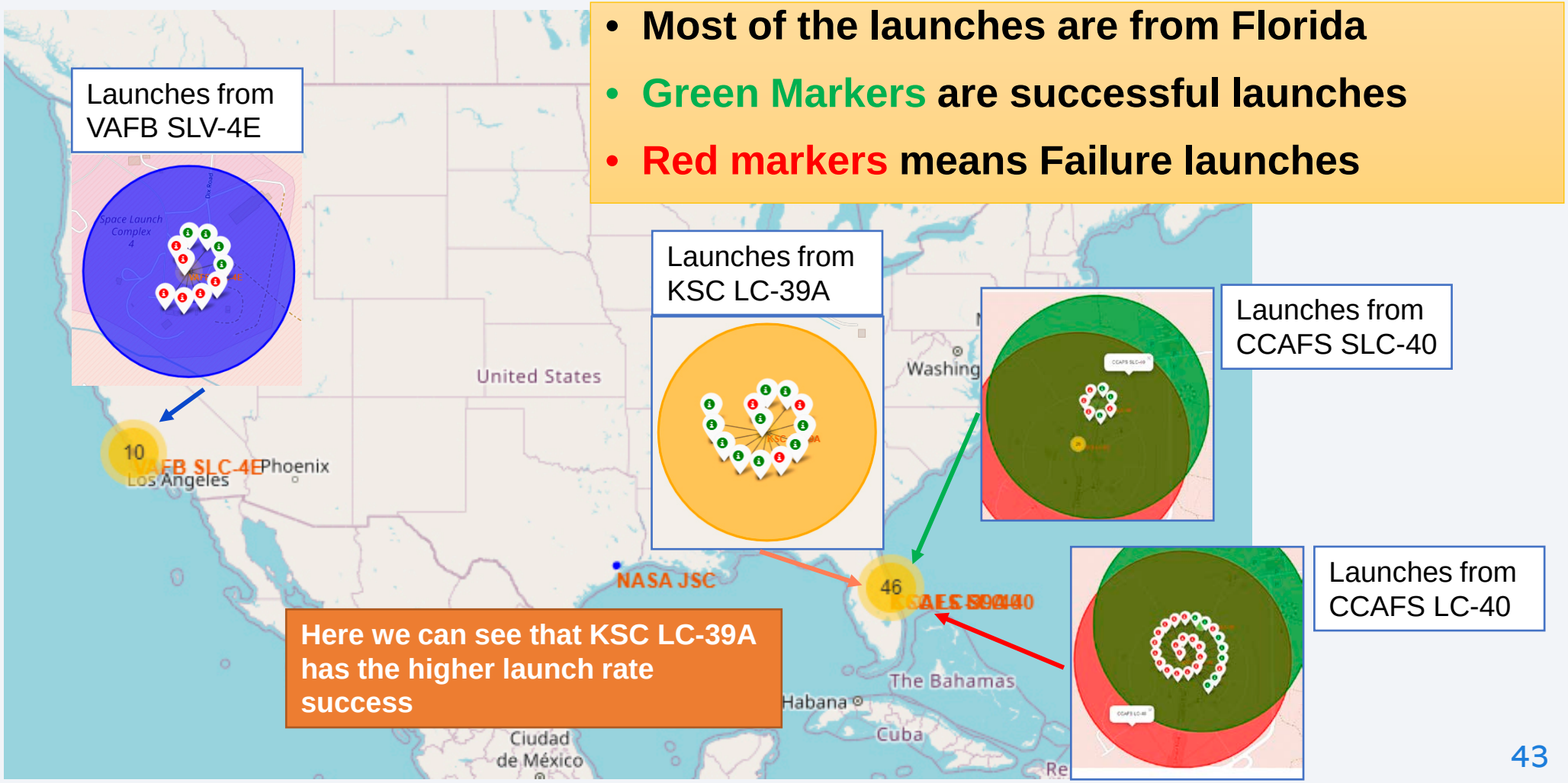

Section 3
Launch Sites
Proximities Analysis

SpaceX launch sites on a Folium Map

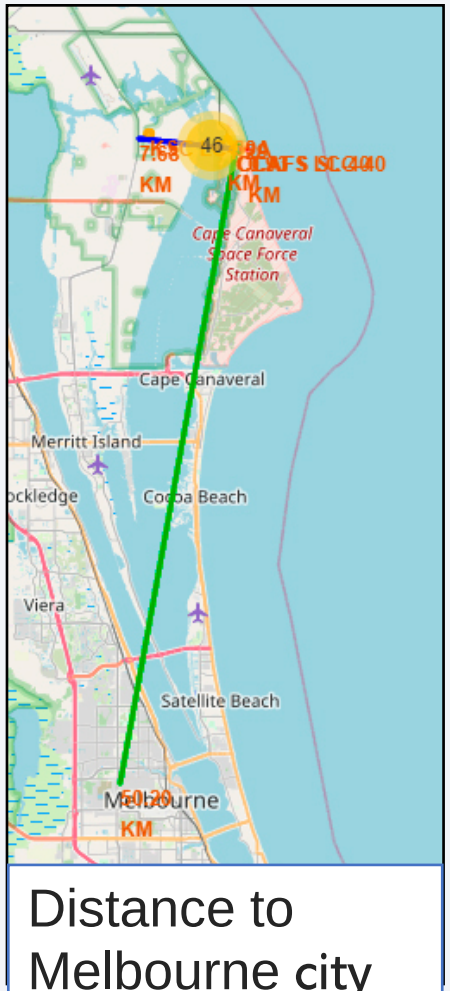


Color-labeled launch outcomes

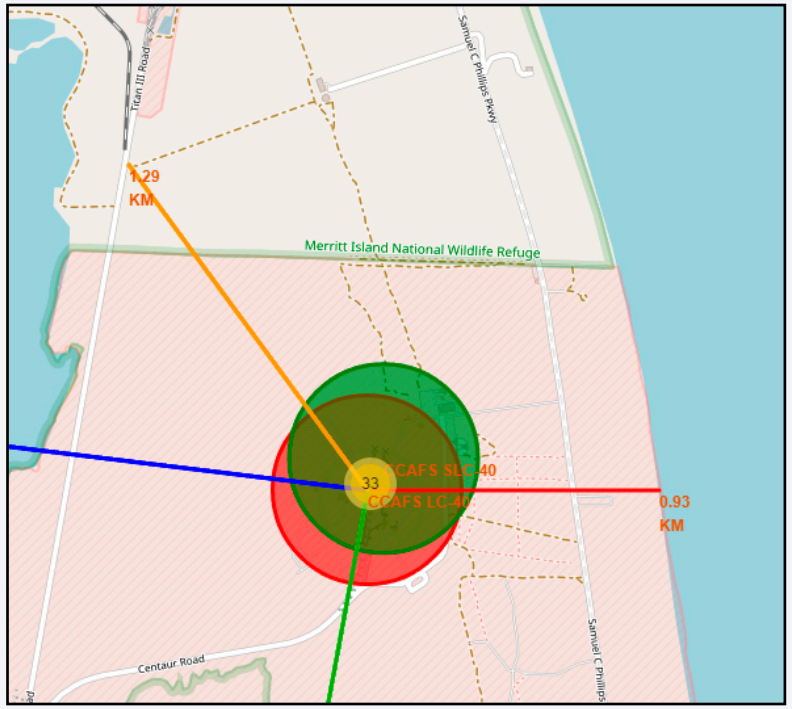
- **Most of the launches are from Florida**
- **Green Markers are successful launches**
- **Red markers means Failure launches**



CCAFS_LC40 Launch site distances

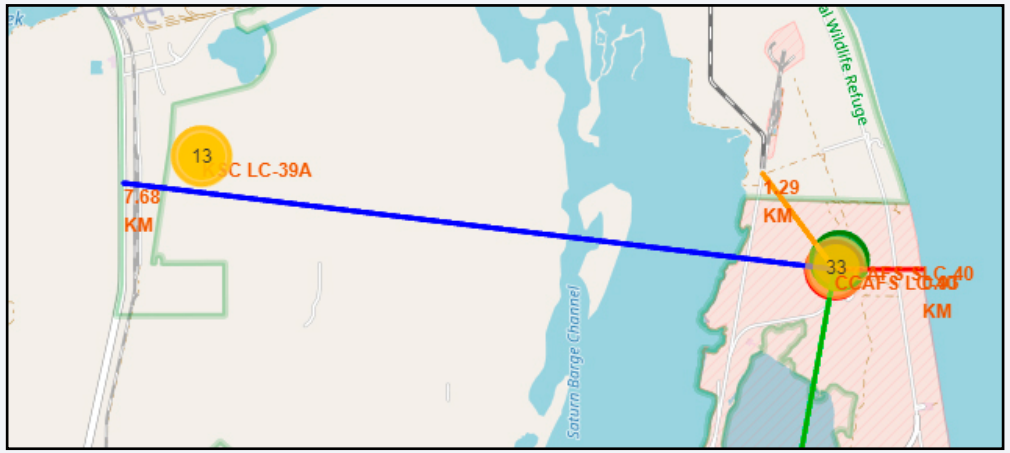


Distance to Melbourne city **Green line**

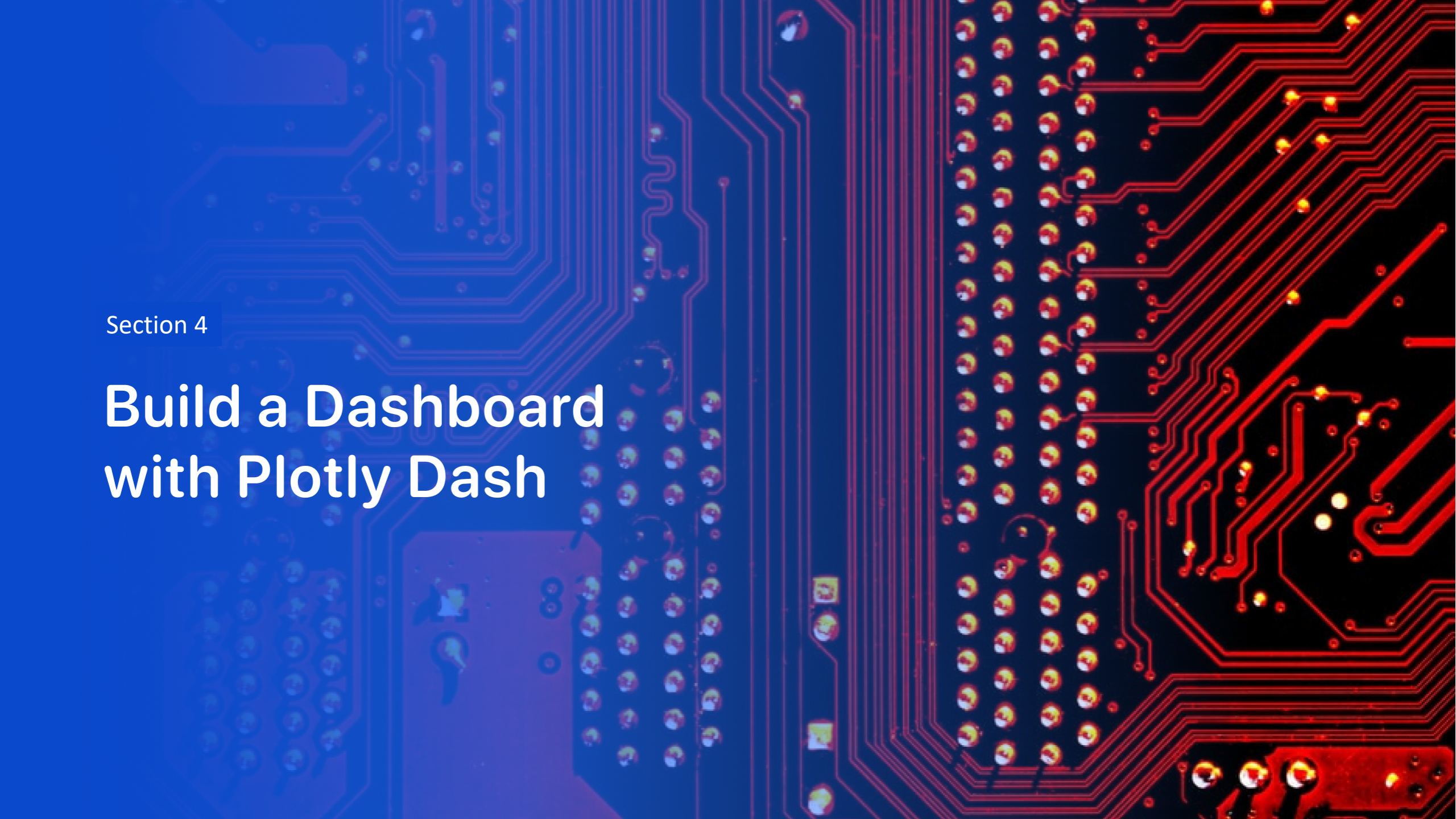


Distance to Railway **Orange line**

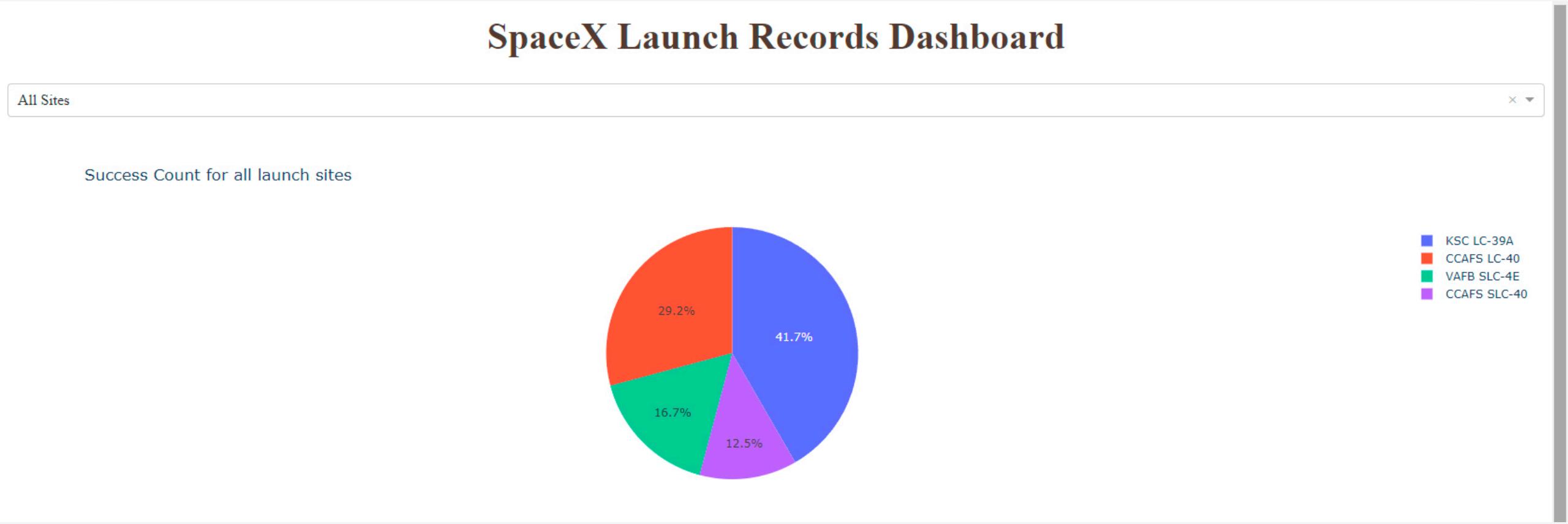
Distance to coastline **Red line**



Distance to Kennedy highway **Blue line**

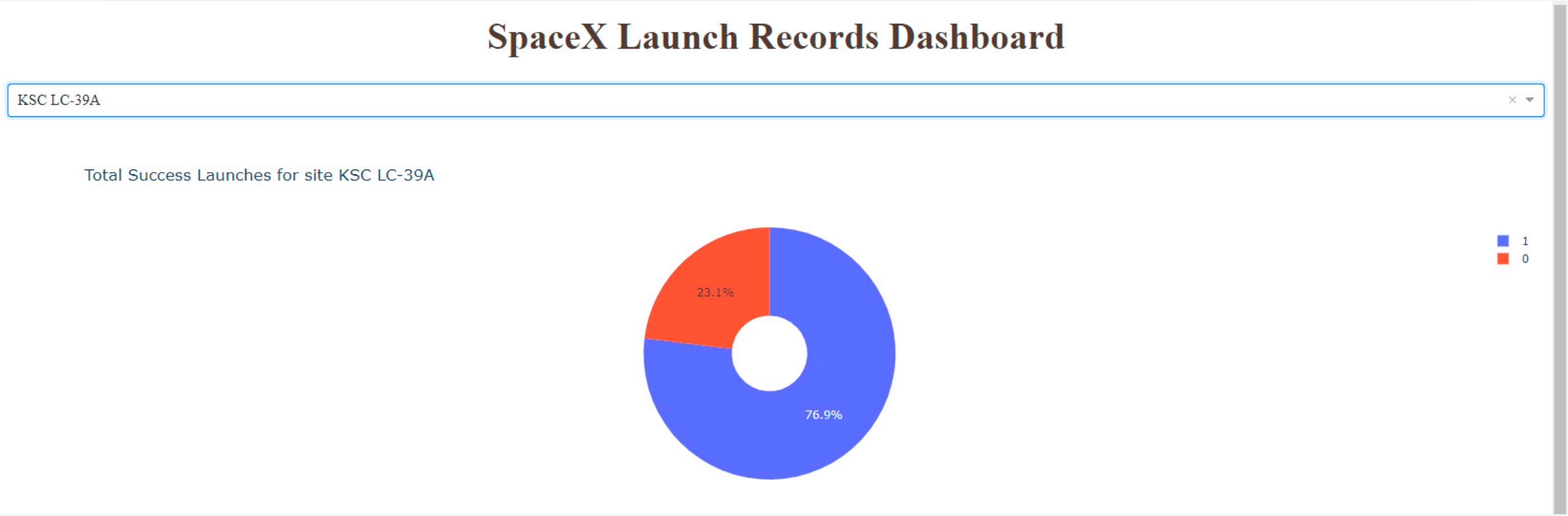

Section 4
Build a Dashboard with Plotly Dash

Launch success count for all sites



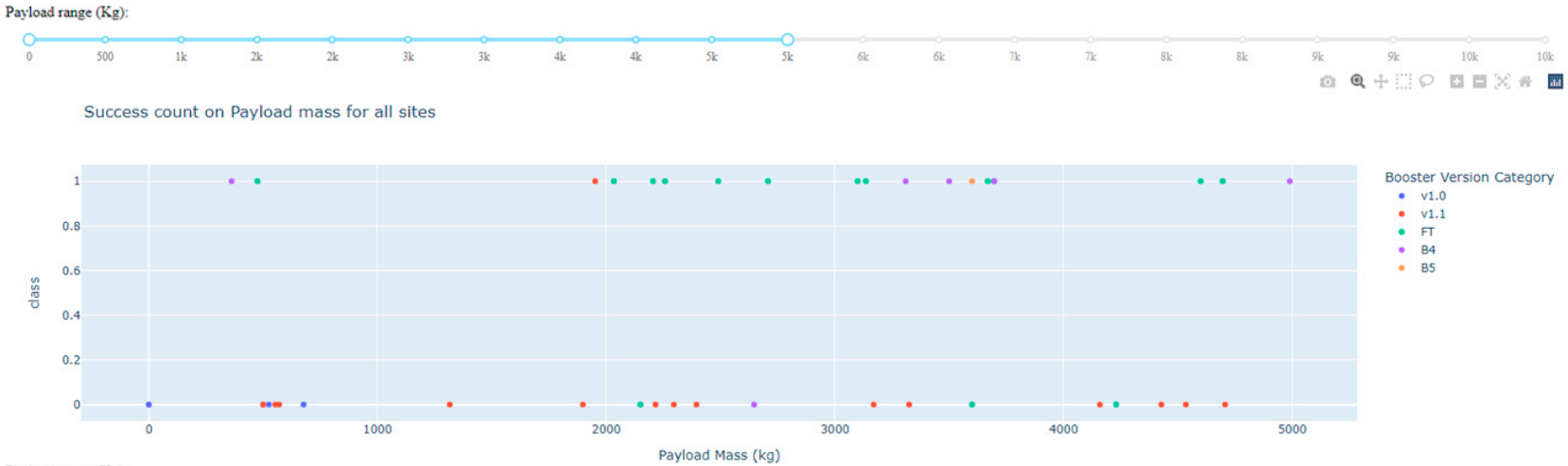
We can see that launch site KSC LC-39A has the higher success rate among all sites (41,7%)

Piechart for the launch site with highest launch success ratio



KSC LC-39A achieved 76,9% of success rate and only 23,1% of failure.

Payload vs. Launch Outcome scatter plot for all sites



We can see that the success rate for low weighted payloads is higher than for heavy weighted payloads

Section 5

Predictive Analysis (Classification)

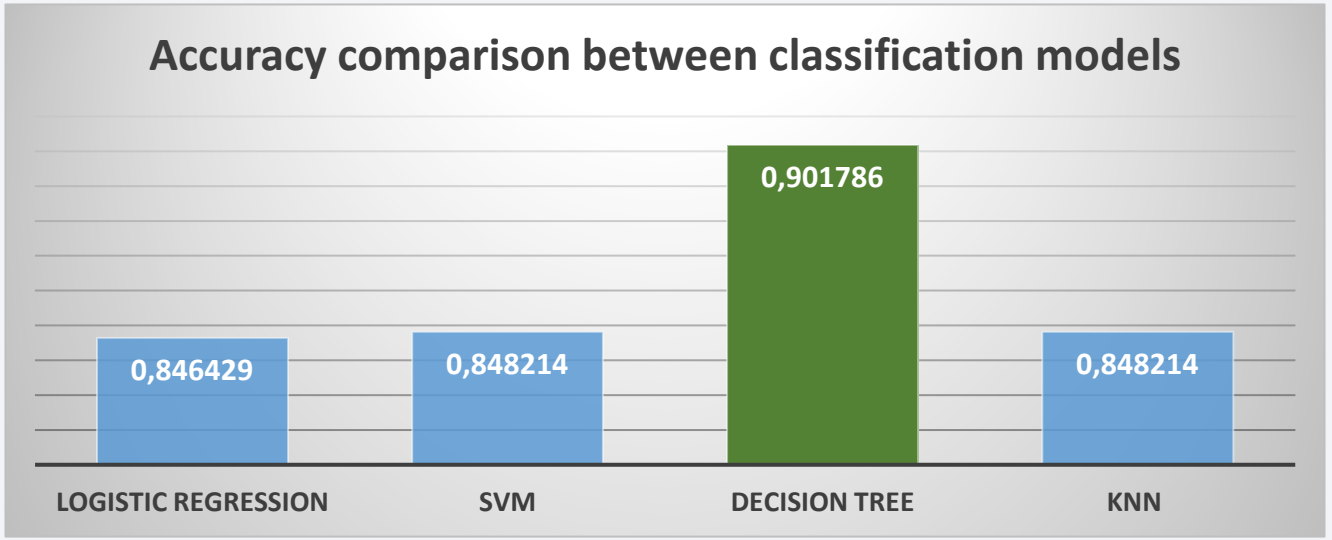
Classification Accuracy

Classification Model	Accuracy	Accuracy on test data	Best Parameters
Logistic Regression	0.846429	0.833334	{'C': 0.01, 'penalty': 'l2', 'solver': 'lbfgs'}
SVM	0.848214	0.833334	{'C': 1.0, 'gamma': 0.03162277660168379, 'kernel': 'sigmoid'}
Decision Tree	**0.901786**	0.833334	{'criterion': 'gini', 'max_depth': 4, 'max_features': 'sqrt', 'min_samples_leaf': 4, 'min_samples_split': 2, 'splitter': 'best'}
KNN	0.848214	0.833334	{'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}

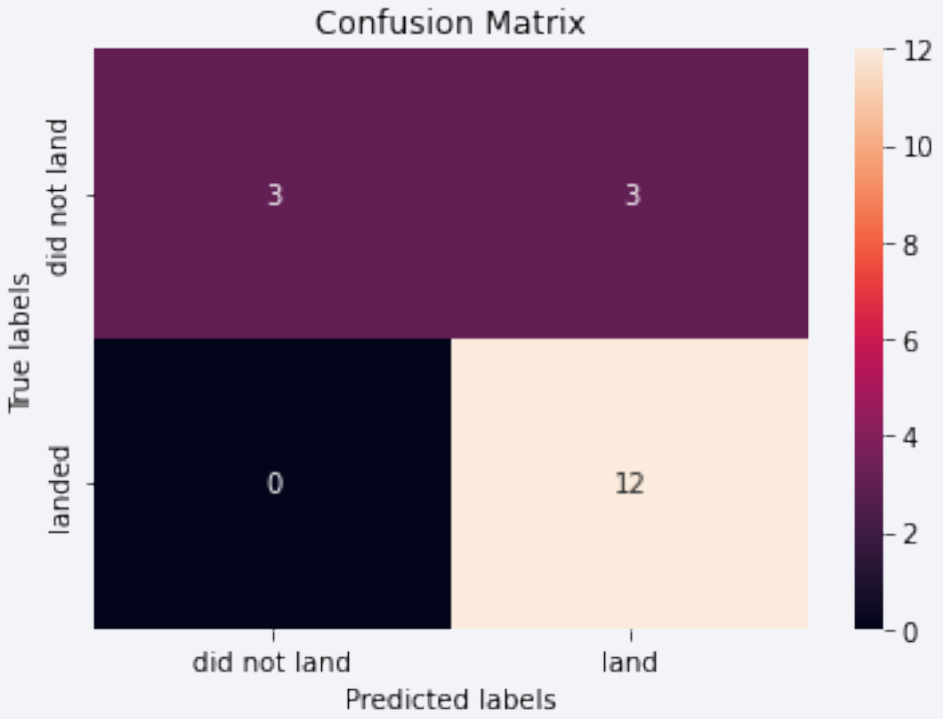
We can see the accuracy is very close but, anyway, there is one winner.

DECISION TREE is the selected one with:

accuracy of 0.901786



Confusion Matrix



The confusion matrix for the decision tree classifier shows that the classifier can distinguish between the different classes. The major problem is the false positives .i.e., unsuccessful landing marked as successful landing by the classifier.

Conclusions

- The launch sites are proximate to the Equator line to improve the scape velocity
- The launch sites are near the coast to avoid crashes over the continental area.
- Success rate increases with time and with the number of flights done.
- Launch success rate started to increase in 2013 and has a positive trend.
- Orbits ES-L1, GEO, HEO, SSO, VLEO have the higher success rates.
- KSC LC-39A is the launch site with better success launch performance
- The Decision tree classifier is the best machine learning algorithm for this task.


Thank you!
SPACEX