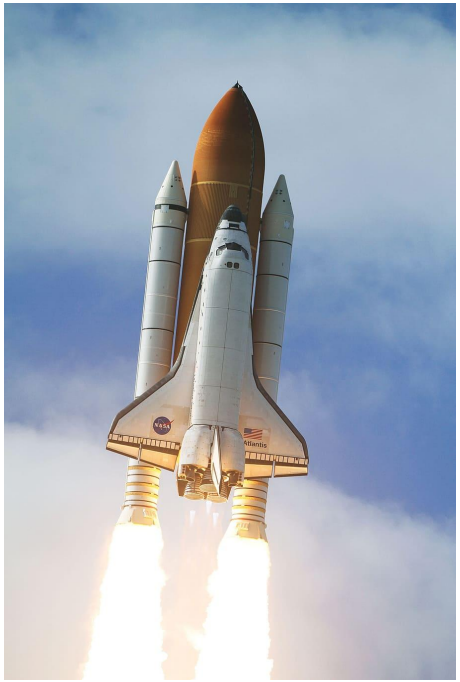
Adel Alshehri

SpaceX
IBM Data Science Project

The Outline

- Executive Summary
- Introduction
- Methodology
- Results
- Conclusion



Executive Summary

- **Summary of methodologies -**
  - Data Collection via API, SQL and Web Scraping
  - Data Wrangling and Analysis
  - Interactive Maps with Folium
  - Predictive Analysis for each classification model

Summary of all results

- Data Analysis with Interactive Visualizations.
- Best Mode for Predictive Analysis

Introduction

- **Project background and context:**

  We will build a model to predict the Falcon 9 first stage will land successfully. SpaceX advertised Falcon 9 rocket launches on its website, with a cost of 62 millions dollars; other providers cost around 165 million dollars. If we can determine if the first stage will land successfully. This analysis and predictions would be used as alternative company wants to fight against SpaceX for rocket launch.

- **Problems we want to find answers:**
  - With what factors, the rocket will land successfully?
  - The effect of each relationship of rocket variables on outcome.
  - Conditions which will aid SpaceX have to achieve the best results.

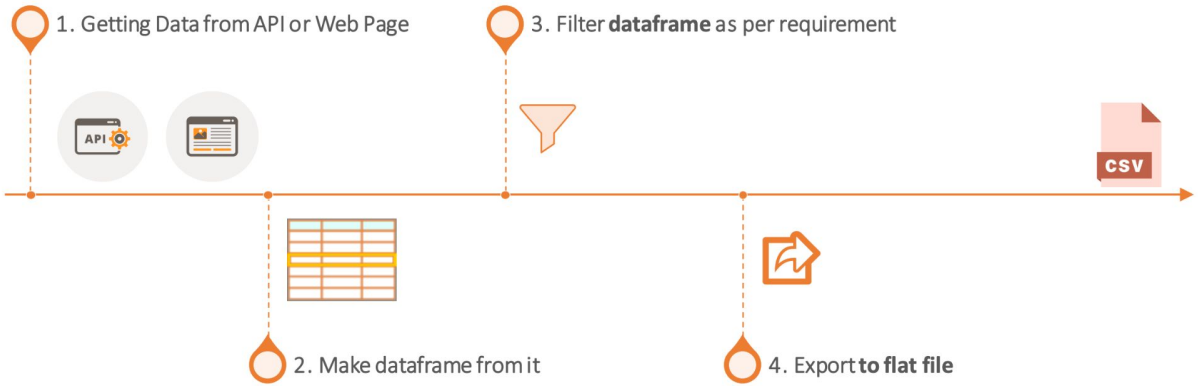
Methodology

- **Data collection methodology**:
  - Via SpaceX Rest API
  - Web Scrapping from Wikipedia
- **Perform data wrangling:**
  - One hot encoding data fields for machine learning and dropping irrelevant columns (Transforming data for Machine Learning)
- **Perform exploratory data analysis (EDA) using visualization and SQL:**
  - Scatter and bar graphs to show patterns between data
- **Perform interactive visual analytics:**
  - Using Folium and Plotly Dash Visualizations
- **Perform predictive analysis using classification models:**
  - Build and evaluate classification models

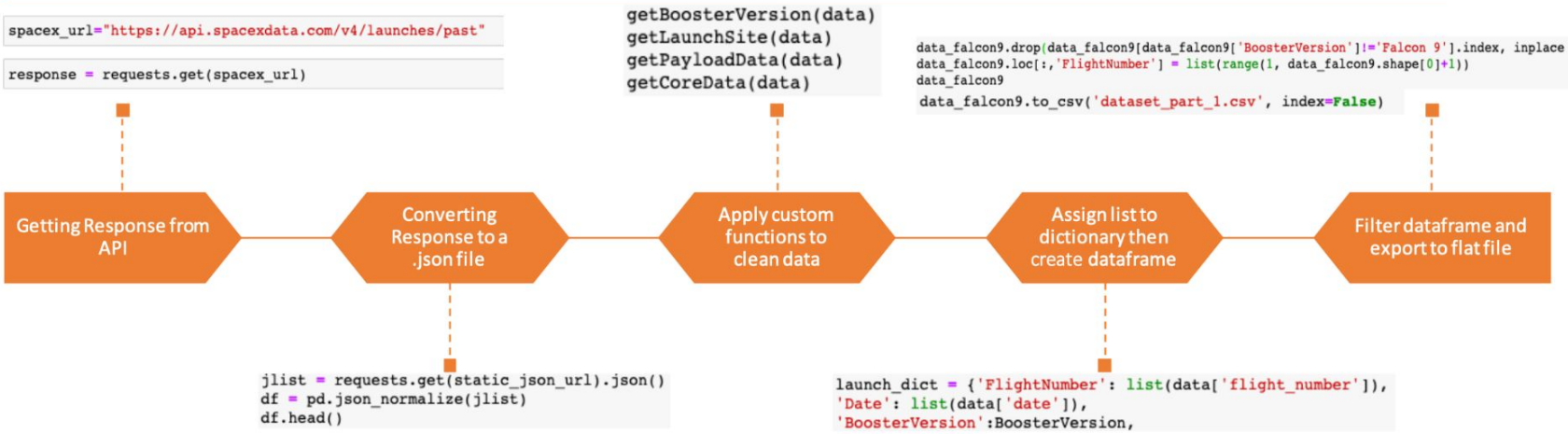
Methodology

Data Collection – Meaning & Basic Steps

Data collection is the process of gathering and measuring information on targeted variables in an established system, which then enables one to answer relevant questions and evaluate outcomes.



Data Collection – Via SpaceX API



	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial
4	1	2010-06-04	Falcon 9	6123.547647	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003
5	2	2012-05-22	Falcon 9	525.000000	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005
6	3	2013-03-01	Falcon 9	677.000000	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007
7	4	2013-09-29	Falcon 9	500.000000	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003
8	5	2013-12-03	Falcon 9	3170.000000	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004

Data Collection – Via Web Scraping

Getting Response from HTML

```
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
```

Creating BeautifulSoup Object

```
soup = BeautifulSoup(data, 'html5lib')
```

Finding tables

```
html_tables=soup.find_all("table")
first_launch_table = html_tables[2]
```

Getting column names

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

Creation of dictionary and appending data to keys

```
launch_dict= dict.fromkeys(column_names)
```

Converting dictionary to dataframe

	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	SpaceX	Success	F9 v1.0B0003.1	Failure	4 June 2010	18:45
1	2	CCAFS	Dragon	0	LEO	NASA	Success	F9 v1.0B0004.1	Failure	8 December 2010	15:43
2	3	CCAFS	Dragon	525 kg	LEO	NASA	Success	F9 v1.0B0005.1	No attempt	22 May 2012	07:44
3	4	CCAFS	SpaceX CRS-1	4,700 kg	LEO	NASA	Success	F9 v1.0B0006.1	No attempt	8 October 2012	00:35
4	5	CCAFS	SpaceX CRS-2	4,877 kg	LEO	NASA	Success	F9 v1.0B0007.1	No attempt	1 March 2013	15:10

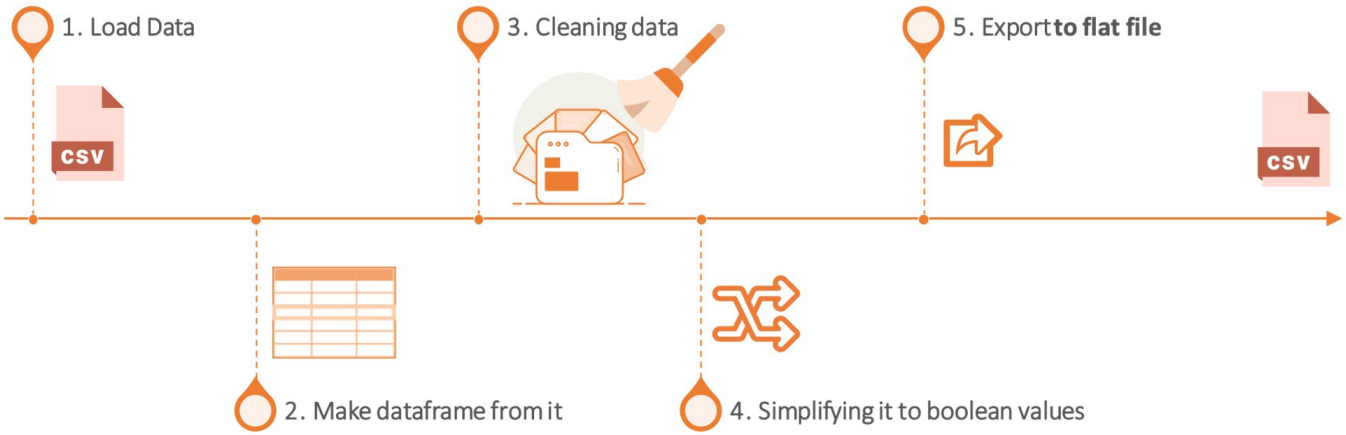
Dataframe to .CSV

Data Wrangling

Data wrangling is the process of cleaning and unifying messy and complex data sets for easy access and analysis.

Here we mainly convert those outcomes into Training Labels with 1 means the booster successfully landed 0 means it was unsuccessful.

```
df['Class'] = df['Outcome'].apply(lambda landing_class: 0 if landing_class in bad_outcomes else 1)
```

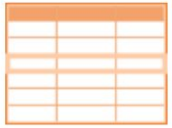


EDA – Meaning & Basic Steps

Exploratory data analysis is an approach of analyzing data sets to summarize their main characteristics, using statistical graphics and other data visualization methods.

1. Load Data

2. Make dataframe from it



3. Create Visualizations

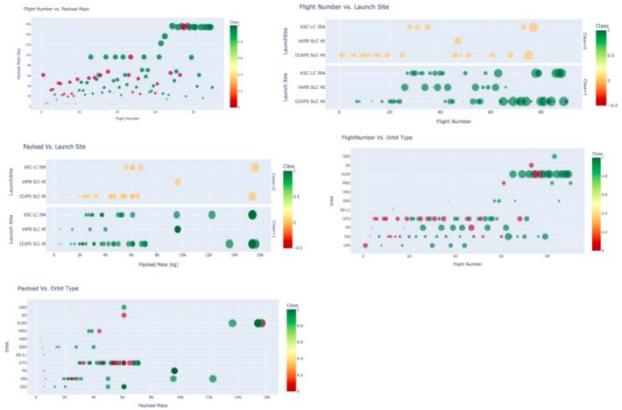
4. Collect Insights

EDA with Data Visualization

Scatter Graphs Drawn:

- Payload and Flight Number
- Flight Number and Launch Site
- Payload and Launch Site
- Flight Number and Orbit Type
- Payload and Orbit Type

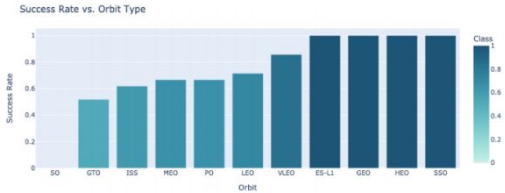
Scatter plots show dependency of attributes on each other. Once a pattern is determined from the graphs it's very easy to predict which factors will lead to maximum probability of success in both outcome and landing.



Bar Graph Drawn:

Success Rate VS. Orbit Type

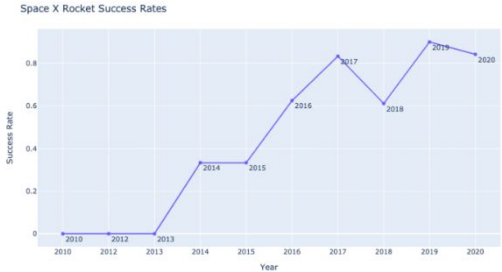
Bar graphs are easiest to interpret a relationship between attributes. Via this bar graph we can easily determine which orbits have the highest probability of success.



Line Graph Drawn:

Launch Success Yearly Trend

Line graphs are useful in that they show trends clearly and can aid in predictions for the future.



EDA with SQL

SQL is an indispensable tool for Data Scientists and analysts as most of the real-world data is stored in databases. It's not only the standard language for Relational Database operations, but also an incredibly powerful tool for analyzing data and drawing useful insights from it. Here we use IBM's Db2 for Cloud, which is a fully managed SQL Database provided as a service.

```
!pip install sqlalchemy==1.3.9
!pip install ibm_db_sa
!pip install ipython-sql
%load_ext sql

%sql ibm_db_sa://my-username:my-password@my-hostname:my-port/my-db-name
%sql <your query>
```

We performed SQL queries to gather information from given dataset :

- Displaying the names of the unique launch sites in the space mission
- Display 5 records where launch sites begin with the string 'CCA'
- Displaying the total payload mass carried by boosters launched by NASA (CRS)
- Displaying average payload mass carried by booster version F9 v1.1
- Listing the date where the successful landing outcome in drone ship was achieved
- Listing the names of the boosters which have success in ground pad and have payload mass greater than 4000 but less than 6000
- Listing the total number of successful and failure mission outcomes
- Listing the names of the booster_versions which have carried the maximum payload mass
- Listing the failed landing_outcomes in drone ship, their booster versions, and launch site names for the year 2015
- Ranking 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

Build an Interactive Map with Folium

Folium makes it easy to visualize data that's been manipulated in Python on an interactive leaflet map. We use the latitude and longitude coordinates for each launch site and added a Circle Marker around each launch site with a label of the name of the launch site. It is also easy to visualize the number of success and failure for each launch site with Green and Red markers on the map.

Map Objects	Code	Result
Map Marker	folium.Marker(	Map object to make a mark on map.
Icon Marker	folium.Icon(	Create an icon on map.
Circle Marker	folium.Circle(	Create a circle where Marker is being placed.
PolyLine	folium.PolyLine(	Create a line between points.
Marker Cluster Object	MarkerCluster()	This is a good way to simplify a map containing many markers having the same coordinate.
AntPath	folium.plugins.AntPath(	Create an animated line between points.

Build a Dashboard with Plotly Dash

Pie Chart showing the total success for all sites or by certain launch site

- *Percentage of success in relation to launch site*

Scatter Graph showing the correlation between Payload and Success for all sites or by certain launch site

- *It shows the relationship between Success rate and Booster Version Category.*

Map Objects	Code	Result
Dash and its components	*import* dash *import* dash_html_components *as* html *import* dash_core_components *as* dcc *from* dash.dependencies *import* Input, Output	Plotly stewards Python's leading data viz and UI libraries. With Dash Open Source, Dash apps run on your local laptop or server. The Dash Core Component library contains a set of higher-level components like sliders, graphs, dropdowns, tables, and more. Dash provides all of the available HTML tags as user-friendly Python classes.
Pandas	*import* pandas *as* pd	Fetching values from CSV and creating a dataframe
Plotly	*import* plotly.express *as* px	Plot the graphs with interactive plotly library
Dropdown	dcc.Dropdown(	Create a dropdown for launch sites
Rangeslider	dcc.RangeSlider(	Create a rangeslider for Payload Mass range selection
Pie Chart	px.pie(	Creating the Pie graph for Success percentage display
Scatter Chart	px.scatter(	Creating the Scatter graph for correlation display

Classification

Building Model

- Load our feature engineered data into dataframe
- Transform it into NumPy arrays
- Standardize and transform data
- Split data into training and test data sets
- Check how many test samples has been created
- List down machine learning algorithms we want to use
- Set our parameters and algorithms to GridSearchCV
- Fit our datasets into the GridSearchCV objects and train our model

Evaluating Model

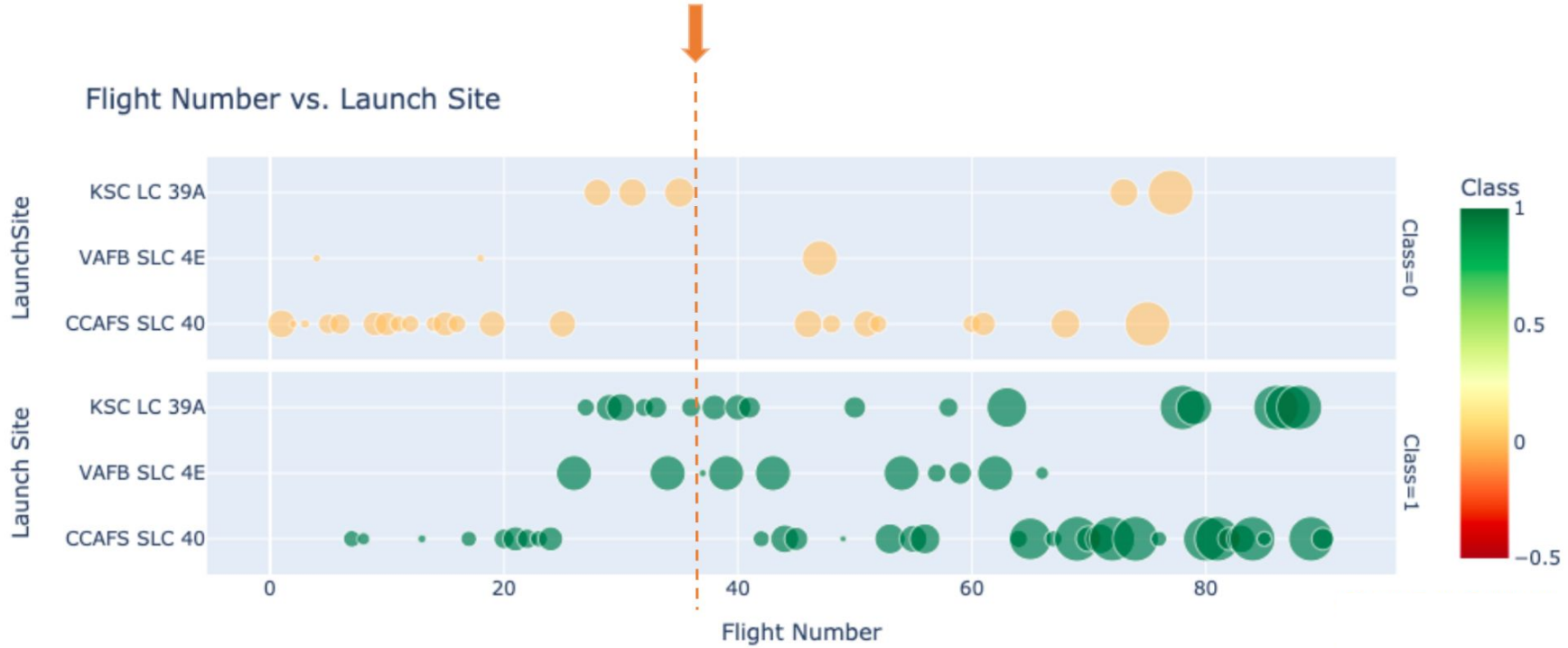
- Check accuracy for each model
- Get best hyperparameters for each type of algorithms
- Plot Confusion Matrix

Results

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

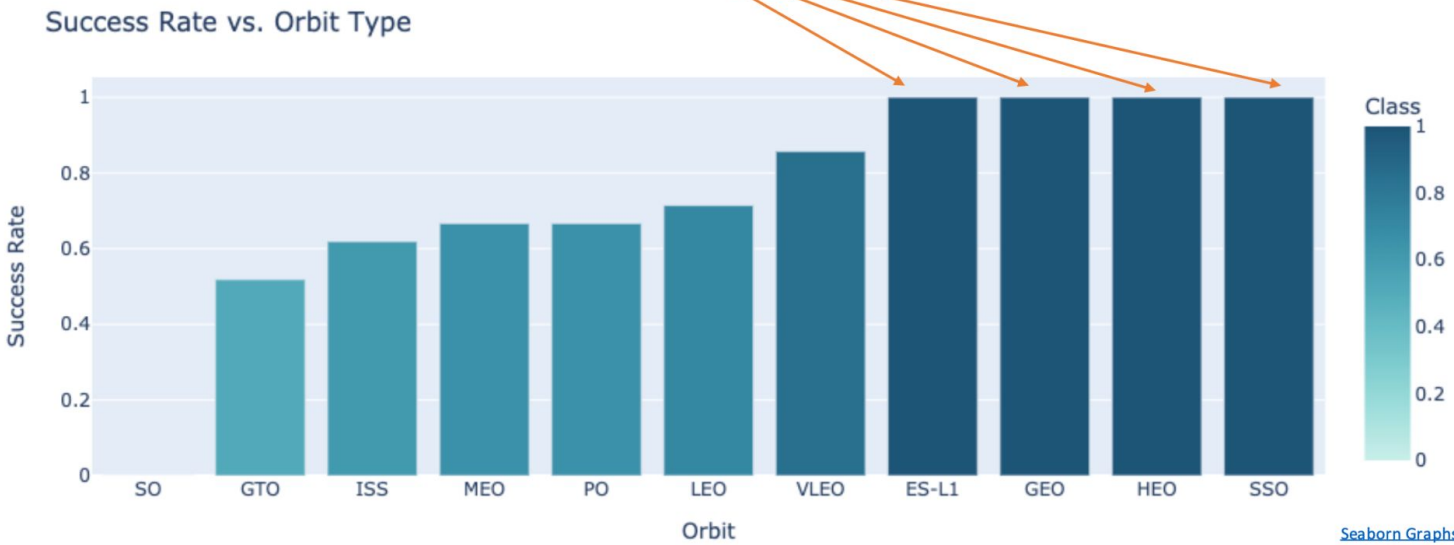
Flight Number vs. Launch Site

- With higher flight numbers (greater than 30) the success rate for the Rocket is increasing.



Success Rate vs. Orbit Type

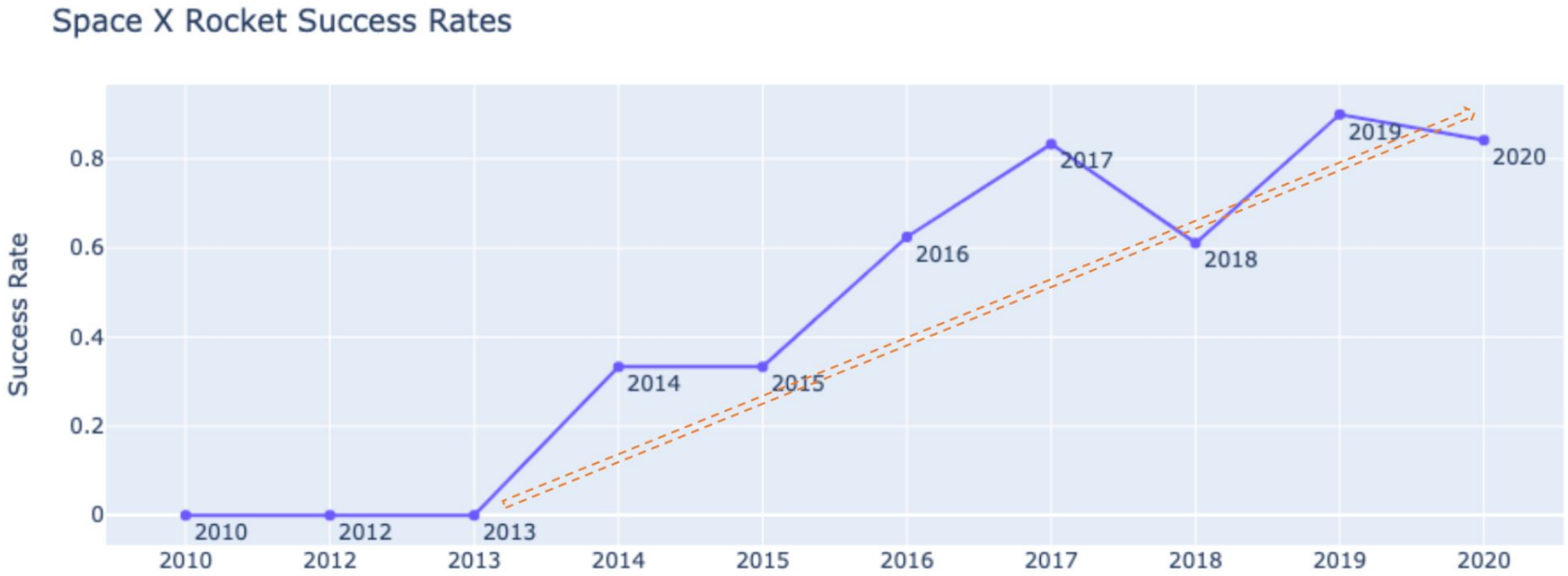
- ES-L1, GEO, HEO, SSO has highest Sucess rates.



Seaborn Graphs GitHub URL

Launch Success Yearly Trend

We can observe that the sucess rate since 2013 kept increasing relatively though there is slight dip after 2019.



EDA with SQL

All Launch Site Names

SQL Query

```
%sql SELECT DISTINCT LAUNCH_SITE as "Launch_Sites" FROM SPACEX;
```

Description

Using the word DISTINCT in the query we pull unique values for Launch_Site column from table SPACEX.

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

Launch Site Names begin with 'CCA'

SQL Query

```
%sql SELECT * FROM SPACEX WHERE LAUNCH_SITE LIKE 'CCA%' LIMIT 5;
```

Description

Using keyword 'LIMIT 5' in the query we fetch 5 records from table spacex and with condition LIKE keyword with wild card - 'CCA%' . The percentage in the end suggests that the Launch_Site name must start 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	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

Total Payload Mass

SQL Query

```
%sql SELECT SUM(PAYLOAD_MASS__KG_) AS "Total Payload Mass by NASA (CRS)" FROM SPACEX WHERE CUSTOMER = 'NASA (CRS)';
```

Description

Using the function SUM calculates the total in the column PAYLOAD_MASS_KG_ and WHERE clause filters the data to fetch Customer's by name "NASA (CRS)".

Total Payload Mass by NASA (CRS)
45596

Successful Drone Ship Landing with Payload

SQL Query

```
%sql SELECT BOOSTER_VERSION FROM SPACEX WHERE LANDING__OUTCOME = 'Success (drone ship)' \
AND PAYLOAD_MASS__KG_ > 4000 AND PAYLOAD_MASS__KG_ < 6000;
```

Description

Selecting only Booster_Version,
WHERE clause filters the dataset to Landing_Outcome = Success (drone ship)

AND clause specifies additional filter conditions
Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000

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

Interactive map with Folium

All Launch Sites on Folium Map

We can see that the SpaceX launch sites are near to the United States of America coasts i.e., Florida and California Regions.

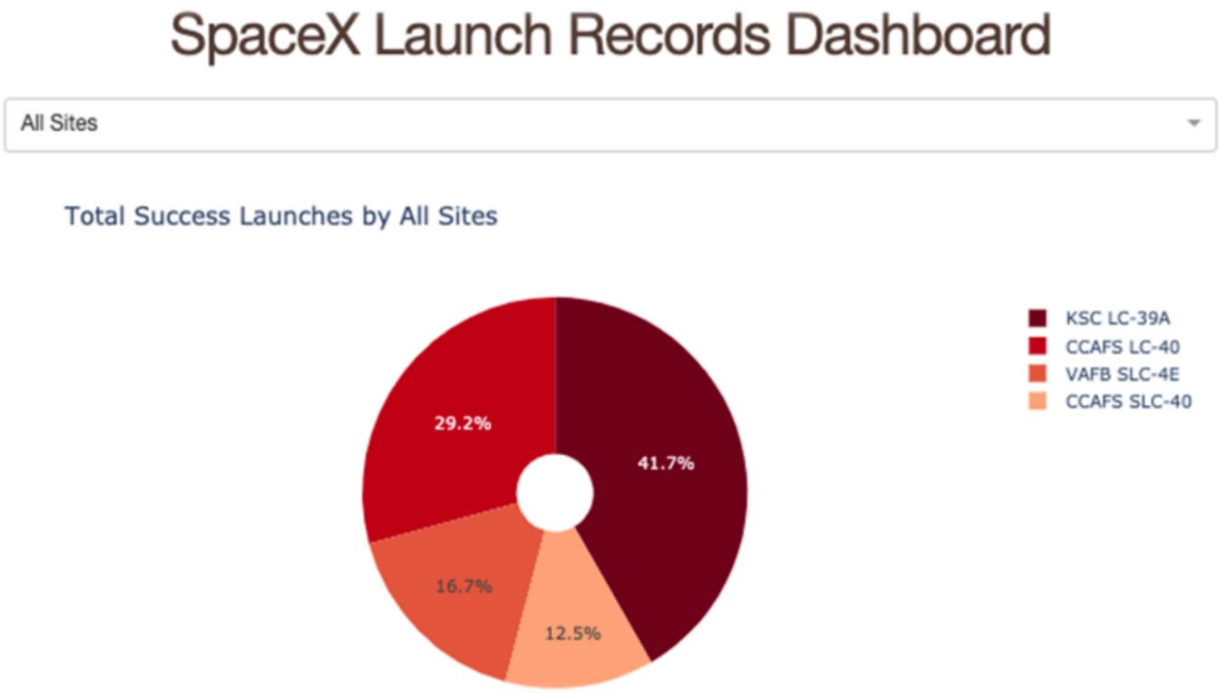
Launch Site Distances from Equator & Railways

Distance from Equator is greater than 3000 Km for all sites.

Distance for all launch sites from railway tracks are greater than .7 Km for all sites. So, launch sites are not so far away from railway tracks.

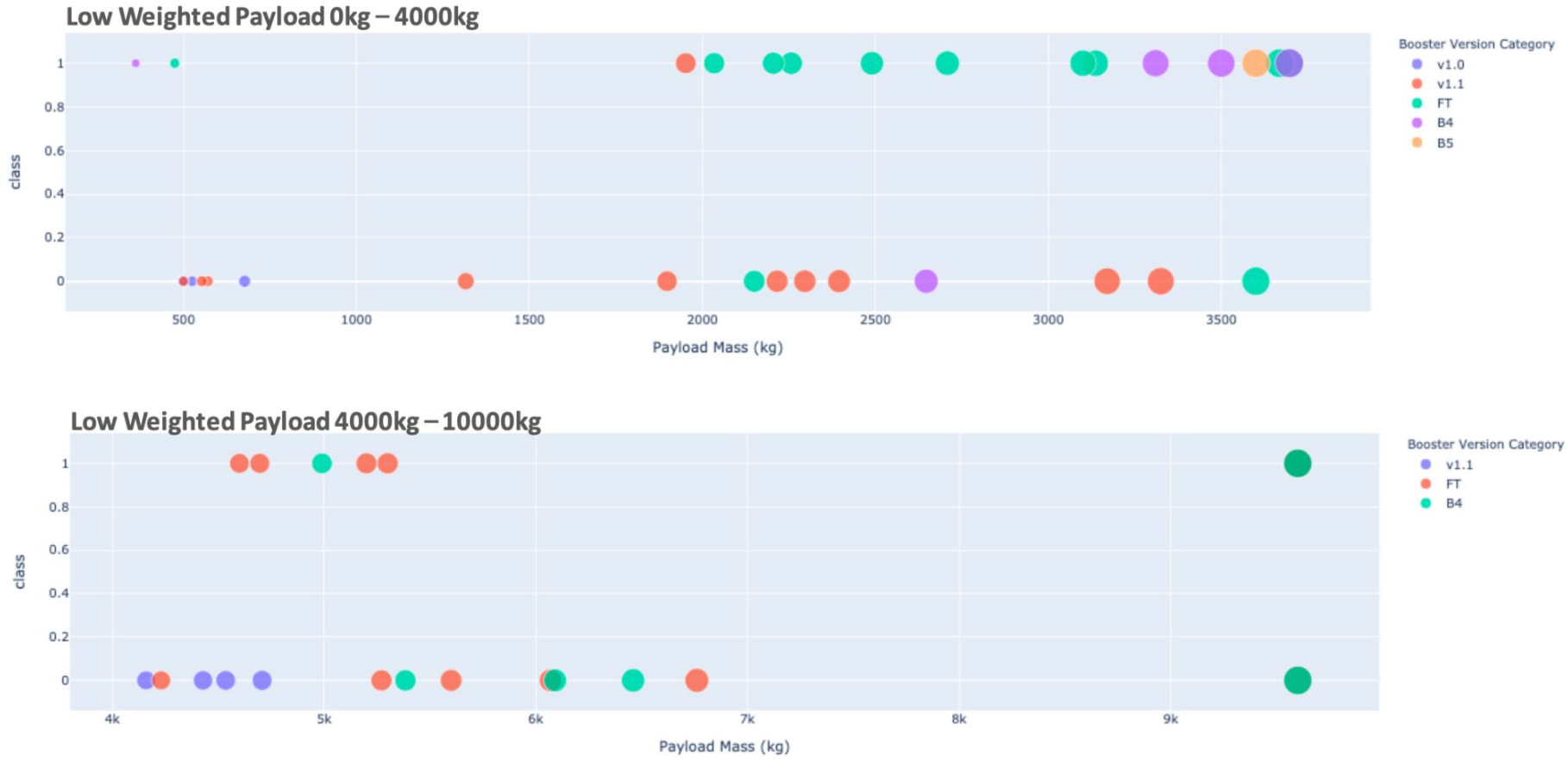

Build a Dashboard with
Plotly Dash

Launch Success Count for All Sites

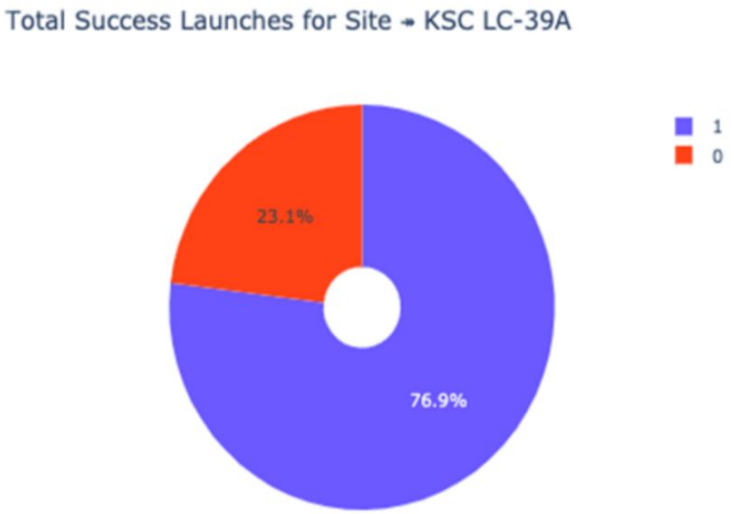


We can see that KSC LC-39A had the most successful launches from all the sites.

Payload vs. Launch Outcome Scatter Plot for All Sites



Launch Site with Highest Launch Success Ratio



KSC LC-39A achieved a 76.9% success rate while getting a 23.1% failure rate.

After visual analysis using the dashboard, we are able to obtain some insights to answer these questions:

- Which site has the highest launch success rate?
  KSC LC – 39A
- Which payload range(s) has the highest launch success rate?
  2000 Kg – 10000 Kg
- Which payload range(s) has the lowest launch success rate?
  0 Kg – 1000 Kg
- Which F9 Booster version (v1.0, v1.1, FT, B4, B5, etc.) has the highest launch success rate?
  FT


Classification

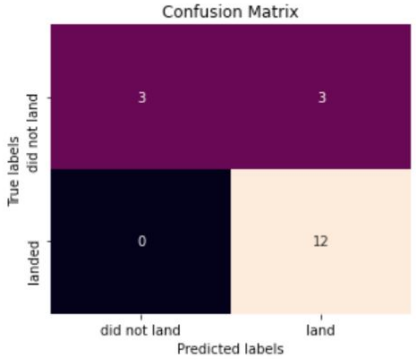
Confusion Matrix

Out here for all models unfortunately, we have same confusion matrix.

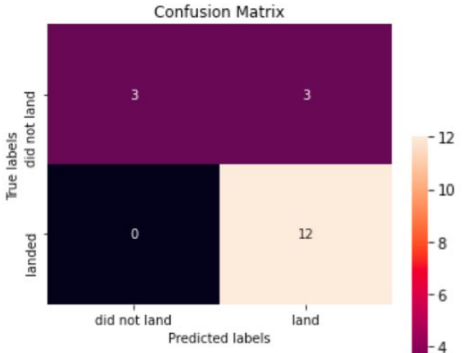
		Predicted Values: Predicted No	Predicted Values: Predicted Yes	
Actual Values	Actual No	True Negative TN = 3	False Positive FP = 3	6
	Actual Yes	False Negative FN = 0	True Positive TP = 12	12
		3	15	Total Cases = 18

Accuracy: (TP+TN)/Total = (12+3)/18 = 0.83333
Misclassification Rate: (FP+FN)/Total = (3+0)/18 = 0.1667
True Positive Rate: TP/Actual Yes = 12/12 = 1
False Positive Rate: FP/Actual No = 3/6 = 0.5
True Negative Rate: TN/Actual No = 3/6 = 0.5
Precision: TP/Predicted Yes = 12/15 = 0.8
Prevalence: Actual yes/Total = 12/18 = 0.6667

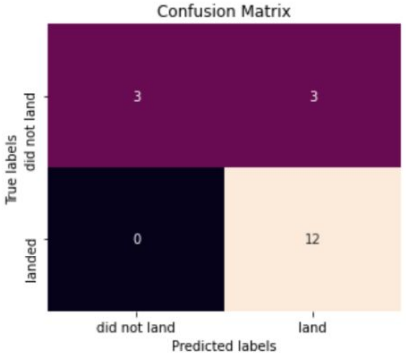
Logistic Regression



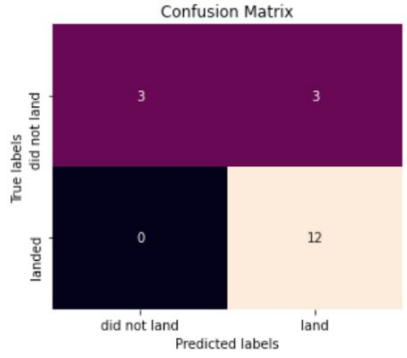
SVM



Decision Tree



KNN

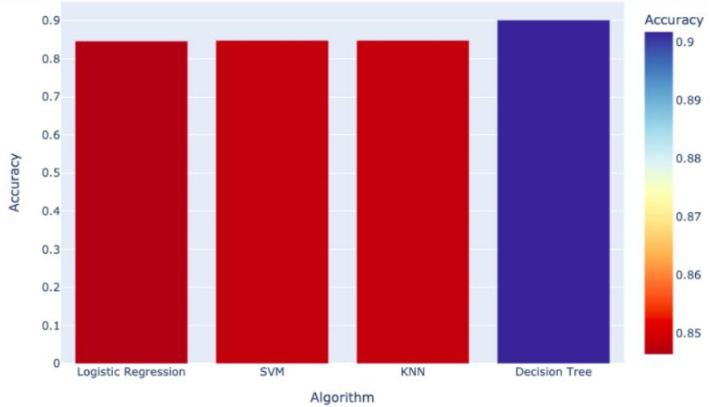


Classification Accuracy

As you can see our accuracy is extremely close, but we do have a clear winner which performs best - "**Decision Tree**" with a score of 0.90178.

Algorithm	Accuracy	Accuracy on Test Data	Tuned Hyperparameters
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'}
KNN	0.848214	0.833334	{'algorithm': 'auto', 'n_neighbors': 10, 'p': 1}
Decision Tree	0.901786	0.833334	{'criterion': 'gini', 'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 2, 'splitter': 'best'}

We trained four different models which each had an 83% accuracy rate.



Conclusion

CONCLUSION

- Orbits ES-L1, GEO, HEO, SSO has highest Sucess rates
- Success rates for SpaceX launches has been increasing relatively with time and it looks like soon they will reach the required target
- KSC LC-39A had the most successful launches but inceasing payload mass seems to have negative impact on success
- Decision Tree Classifier Algorithm is the best for Machine Learning Model for provided dataset

Thank You