



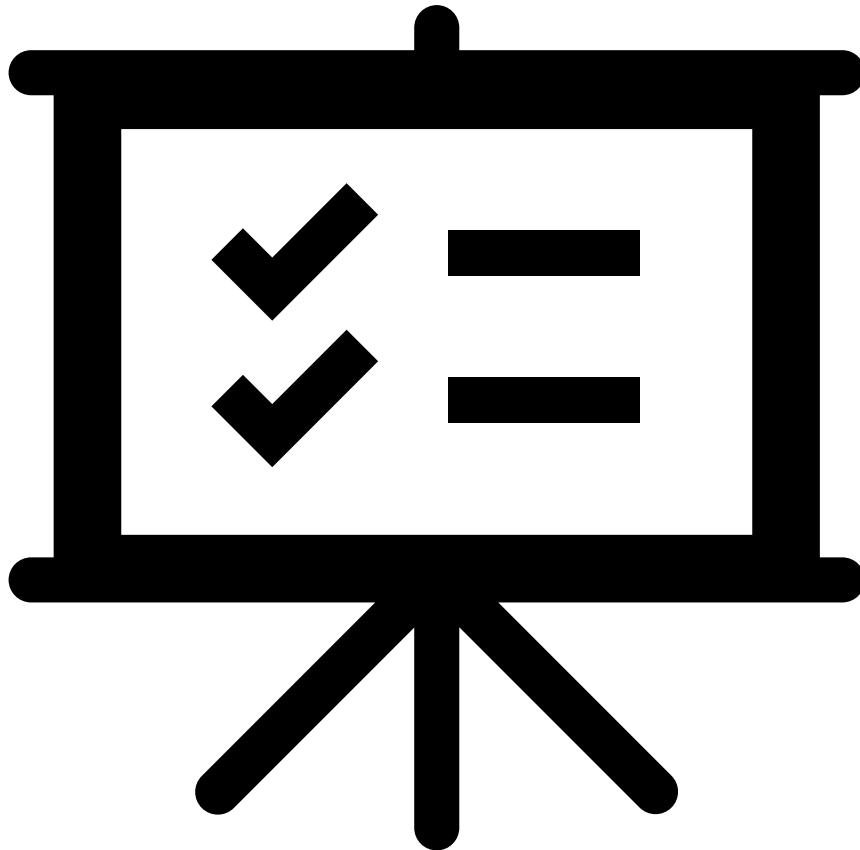
Space X Falcon 9 Launch Analysis

Guillermo Ivan Carrillo Zorrilla

March 2023

Table of Contents

- Executive Summary
- Introduction
- Methodology
- Results
 - Visualization – Charts
 - EDA with SQL
 - Geospatial Results (Folium)
 - Dashboard Results
 - Predictive Analysis
- Conclusions



EXECUTIVE SUMMARY

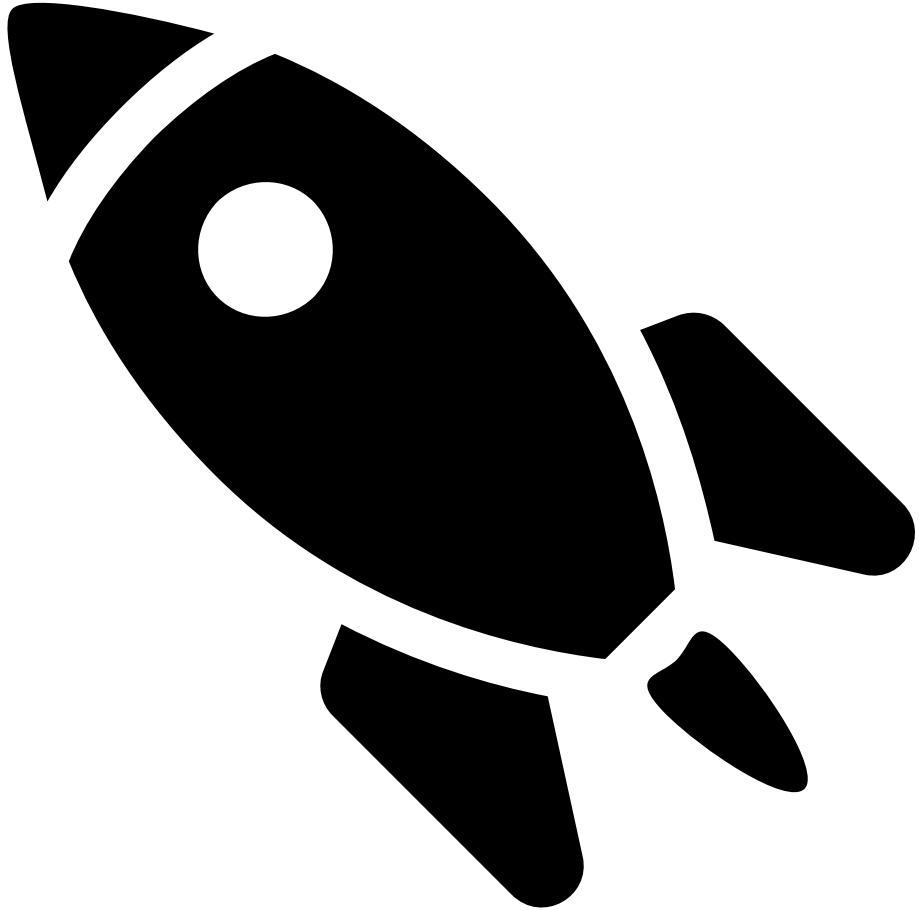


- We used data analysis methodologies to predict if a Space X Falcon 9 first-stage landing will be successful.
- Data collection and wrangling was necessary to conduct exploratory and predictive analyses.
- Exploratory Data Analysis provided insights on the relationship and weigh of variables.
- Geospatial analysis provided insight on common geographic features across launch sites.
- Several statistical models were tested:
 - Logistic Regression
 - SVM
 - Decision Tree
 - KNN

INTRODUCTION

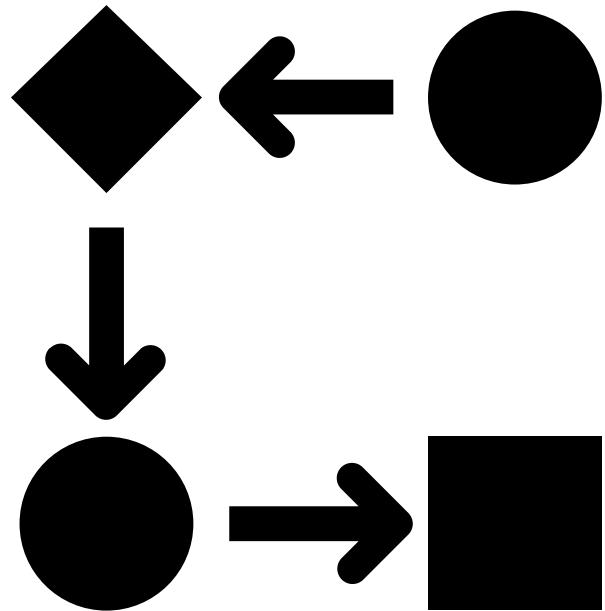
The project aims to predict if the first-stage Falcon 9 will land successfully. SpaceX reports that Falcon 9 rocket launches on its website with a cost of USD 62 million. Space X reports lower costs as it can reuse the first stage.

The advantage of predicting the landing outcome is that we could determine the cost of a launch. This information can be used if an alternate company wants to bid against SpaceX for a rocket launch.



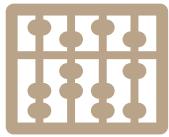
Methodology

METHODOLOGY OVERVIEW



- Data collection: API and web-scraping.
- Data wrangling:
 - Exploratory Data Analysis (EDA) to conduct initial calculations to find preliminary patterns and determine how variables influence the launching outcome.
 - Tools: SQL, Pandas, Matplotlib
- Create interactive visuals for further exploratory analysis and data manipulation:
 - Geospatial visualization: Folium
 - Interactive analytics dashboard: Plotly Dash
- Machine Learning prediction models.

Methodology: API Data Collection



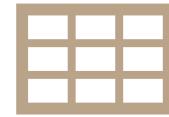
Preliminary steps

- Import libraries.
- Define functions to obtain data from API
- Convert JSON file into a pandas data frame.



Data cleaning

- Select subsets of data that are relevant for the analysis.
- Change the date_utc format to a datetime data type.



Create final data frame

- Create empty lists corresponding to the data we will use for the analysis.
- Use the defined functions to extract the desired values.
- Create a dictionary using the lists as values.
- Create a data frame using the dictionary.

Methodology: Data Collection and Wrangling

With the resulting data frame, the next step was to deal with the NaN values in the PayloadMass column.

[27]:

	FlightNumber	Date	BoosterVersion	PayloadMass	Orbit	LaunchSite	Outcome	Flights	GridFins	Reused	Legs	LandingPad	Block	ReusedCount	Serial	Longitude	Latitude
4	1	2010-06-04	Falcon 9	NaN	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0003	-80.577366	28.561857
5	2	2012-05-22	Falcon 9	525.0	LEO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0005	-80.577366	28.561857
6	3	2013-03-01	Falcon 9	677.0	ISS	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B0007	-80.577366	28.561857
7	4	2013-09-29	Falcon 9	500.0	PO	VAFB SLC 4E	False Ocean	1	False	False	False	None	1.0	0	B1003	-120.610829	34.632093
8	5	2013-12-03	Falcon 9	3170.0	GTO	CCSFS SLC 40	None None	1	False	False	False	None	1.0	0	B1004	-80.577366	28.561857
...
89	86	2020-09-03	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	2	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	12	B1060	-80.603956	28.608058
90	87	2020-10-06	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	3	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	13	B1058	-80.603956	28.608058
91	88	2020-10-18	Falcon 9	15600.0	VLEO	KSC LC 39A	True ASDS	6	True	True	True	5e9e3032383ecb6bb234e7ca	5.0	12	B1051	-80.603956	28.608058
92	89	2020-10-24	Falcon 9	15600.0	VLEO	CCSFS SLC 40	True ASDS	3	True	True	True	5e9e3033383ecbb9e534e7cc	5.0	12	B1060	-80.577366	28.561857
93	90	2020-11-05	Falcon 9	3681.0	MEO	CCSFS SLC 40	True ASDS	1	True	False	True	5e9e3032383ecb6bb234e7ca	5.0	8	B1062	-80.577366	28.561857

90 rows × 17 columns

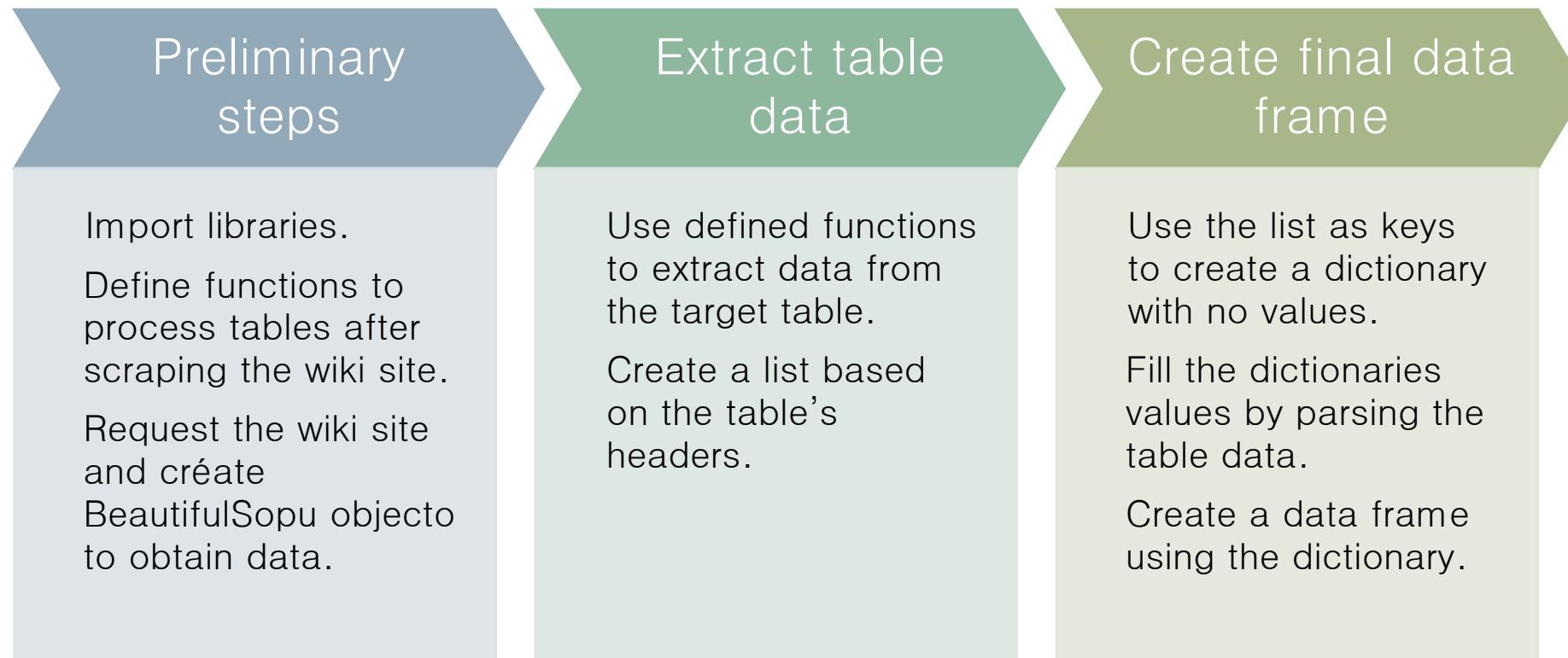
[60]:

```
# Calculate the mean value of PayloadMass column
PMmean = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].replace(np.nan, PMmean)
data_falcon9.isnull().sum()
```

[60]:

FlightNumber	0
Date	0
BoosterVersion	0
PayloadMass	0
Orbit	0
Launchsite	0
Outcome	0
Flights	0
GridFins	0
Reused	0
Legs	0
LandingPad	26
Block	0
ReusedCount	0
Serial	0
Longitude	0
Latitude	0
dtype: int64	

Methodology: Web Scrapping Data Collection



Methodology: EDA and Interactive Visual Analytics

SQL

I used SQL to further organize data and obtain useful information regarding launching events.

Libraries

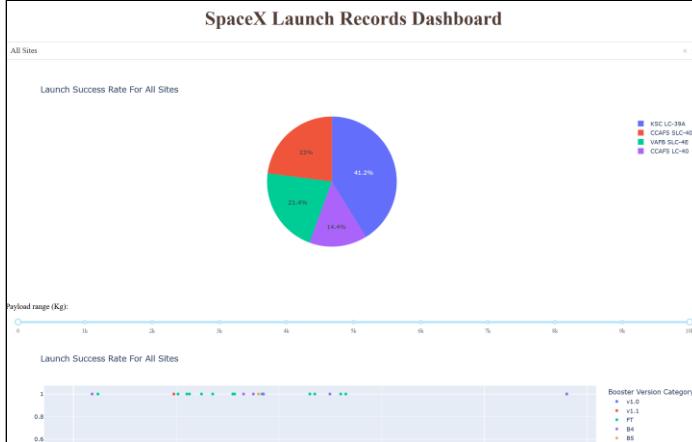
I used several libraries to visualize and analyze data:

- Pandas: Organize and manipulate data.
- Matplotlib: Allowed me to plot data.
- Seaborn: Render visuals and analyze variables' relationships.
- Folium: Visualize geospatial data.
- Plotly Dash: Create an interactive dashboard to analyze data.

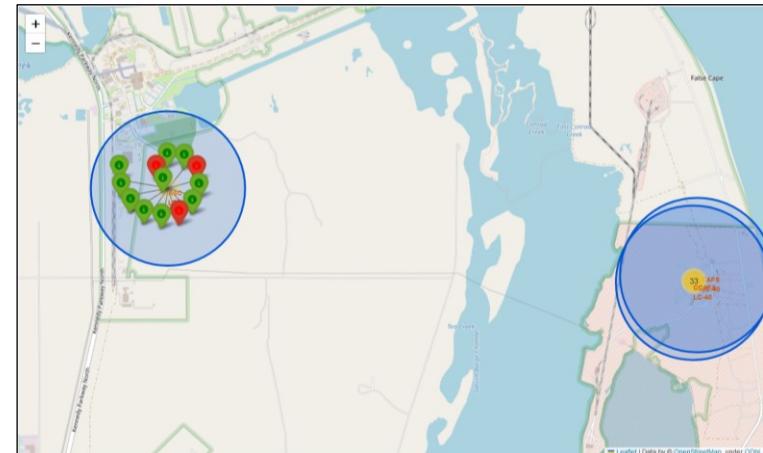
Methodology: EDA with Pandas/Matplotlib, SQL, and Visual Analytics

```
List the names of the booster_versions which have carried the maximum payload mass. Use a subquery
[17]: !sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS_KG_ = (SELECT MAX(PAYLOAD_MASS_KG_) FROM SPACEXTBL)
* sqlite:///my_data1.db
Done.
[17]: 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
```

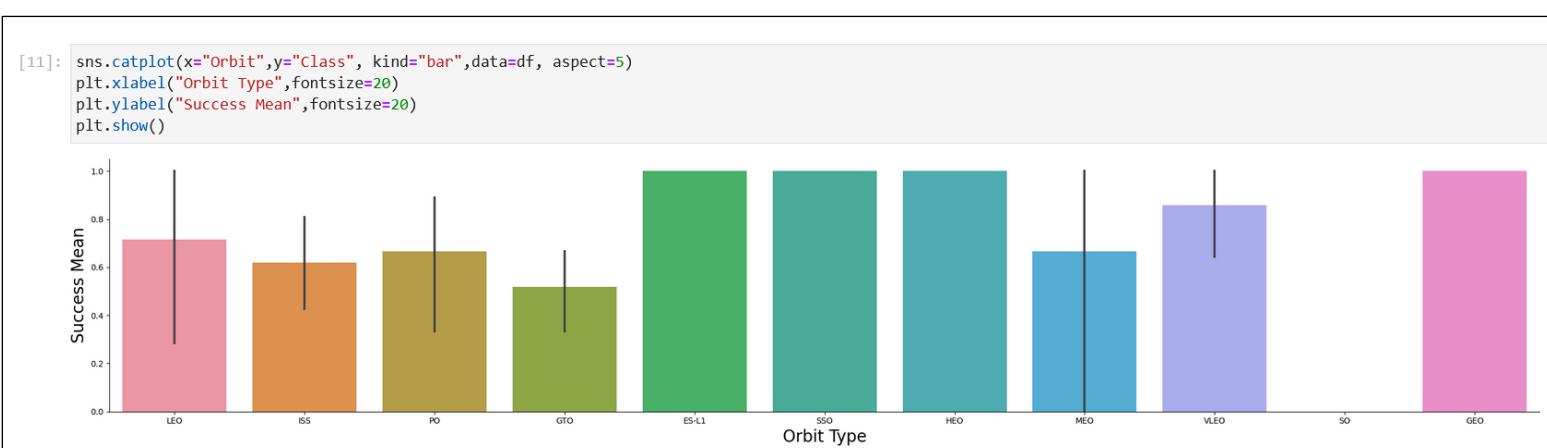
SQL sample



Dashboard overview



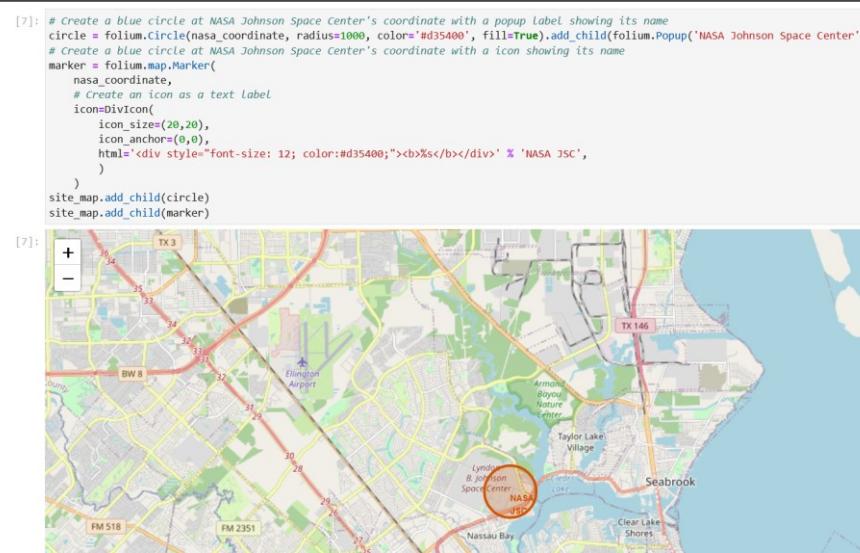
Folium sample



Matplotlib and seaborn sample

Folium Peculiarities

- To elaborate more interactive and useful charts, I had to create additional elements:
 - Icons
 - Colored markers
 - Clusters
 - Lines



Circle and name tags

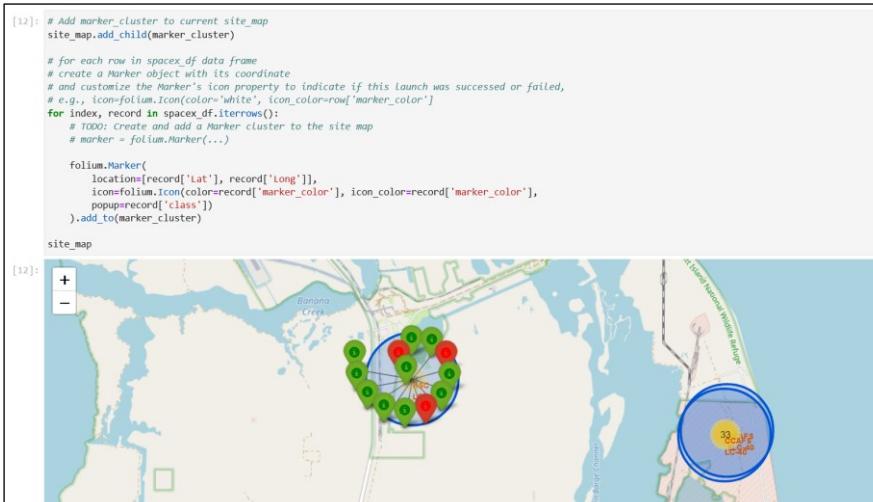
```
[11]: # Apply a function to check the value of `class` column
# If class=1, marker_color value will be green
# If class=0, marker_color value will be red

def assign_marker_color(launch_outcome):
    if launch_outcome == 1:
        return 'green'
    else:
        return 'red'

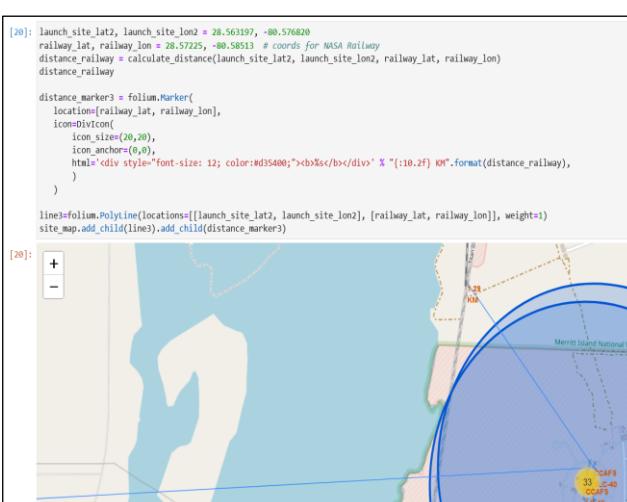
spacex_df['marker_color'] = spacex_df['class'].apply(assign_marker_color)
spacex_df.tail(10)
```

	Launch Site	Lat	Long	class	marker_color
46	KSC LC-39A	28.573255	-80.646895	1	green
47	KSC LC-39A	28.573255	-80.646895	1	green
48	KSC LC-39A	28.573255	-80.646895	1	green
49	CCAFS SLC-40	28.563197	-80.576820	1	green
50	CCAFS SLC-40	28.563197	-80.576820	1	green
51	CCAFS SLC-40	28.563197	-80.576820	0	red
52	CCAFS SLC-40	28.563197	-80.576820	0	red
53	CCAFS SLC-40	28.563197	-80.576820	0	red
54	CCAFS SLC-40	28.563197	-80.576820	1	green
55	CCAFS SLC-40	28.563197	-80.576820	0	red

Creating color markers



Clusters



Lines

Methodology: Predictive Analysis

1. Standardize and normalize data.
2. Split the data into test and training sets for the machine learning models.
3. After training the model, conduct Grid Search to establish the best hyperparameters for the algorithms.
4. Train and test the following methods:
 - Logistic Regression (LR)
 - Support Vector Machine (SVM)
 - Decision Tree Classifier
 - K–Nearest Neighbors (KNN)
5. Determine the accuracy of each model and render a confusion matrix to determine the best predictive model for our data.

Methodology: Predictive Analysis

Standardize the data in `X` then reassign it to the variable `X` using the transform provided below.

```
[28]: # students get this
x = preprocessing.StandardScaler().fit_transform(X)
X
```

```
[28]: array([[-1.71291154e+00, -5.29526321e-17, -6.53912840e-01, ...,
-8.35531692e-01, 1.93309133e+00, -1.93309133e+00],
[-1.67441914e+00, -1.19523159e+00, -6.53912840e-01, ...,
-8.35531692e-01, 1.93309133e+00, -1.93309133e+00],
[-1.63592675e+00, -1.16267307e+00, -6.53912840e-01, ...,
-8.35531692e-01, 1.93309133e+00, -1.93309133e+00],
...,
[ 1.63592675e+00, 1.99100483e+00, 3.49060516e+00, ...,
1.19684269e+00, -5.17306132e-01, 5.17306132e-01],
[ 1.67441914e+00, 1.99100483e+00, 1.00389436e+00, ...,
1.19684269e+00, -5.17306132e-01, 5.17306132e-01],
[ 1.71291154e+00, -5.19213966e-01, -6.53912840e-01, ...,
-8.35531692e-01, -5.17306132e-01, 5.17306132e-01]])
```

We split the data into training and testing data using the function `train_test_split`. The training data is divided into validation data, a second set used for training data; then the models are trained and hyperparameters are selected using the function `GridSearchCV`.

TASK 3

Use the function `train_test_split` to split the data `X` and `Y` into training and test data. Set the parameter `test_size` to 0.2 and `random_state` to 2. The training data and test data should be assigned to the following labels.

```
x_train, X_test, Y_train, Y_test
```

```
[31]: x_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

we can see we only have 18 test samples.

```
[32]: Y_test.shape
```

```
[32]: (18,)
```

Standardizing and splitting procedures

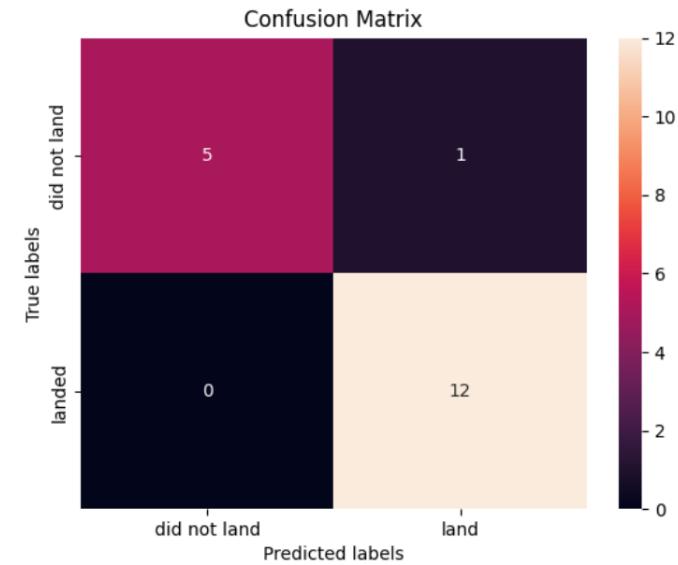
Calculate the accuracy on the test data using the method `score`:

```
[41]: print('score on train data: ', logreg_cv.score(X_train, Y_train)) # R2 score on train data
print('score on test data : ', logreg_cv.score(X_test, Y_test)) # R2 score on test data
```

score on train data: 0.875
score on test data : 0.9444444444444444

Lets look at the confusion matrix:

```
[42]: yhat=logreg_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



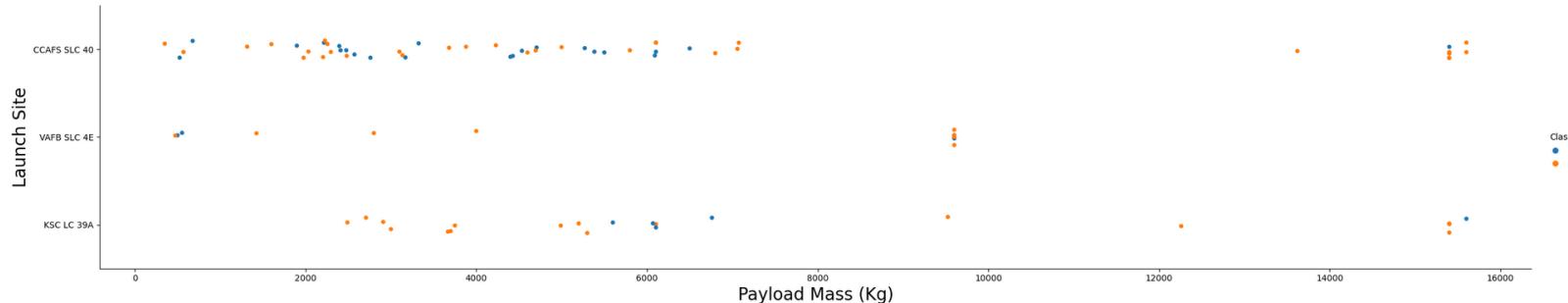
LR testing and confusion matrix example

RESULTS



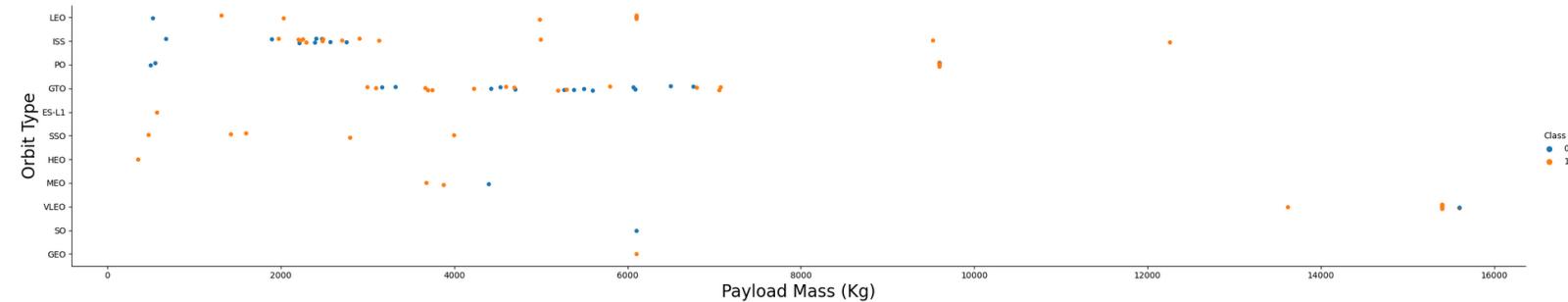
EDA with Visualizations

```
[9]: # Plot a scatter point chart with x axis to be Pay Load Mass (kg) and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite",x="PayloadMass",hue ="Class",data=df,aspect= 5)
plt.xlabel("Payload Mass (Kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



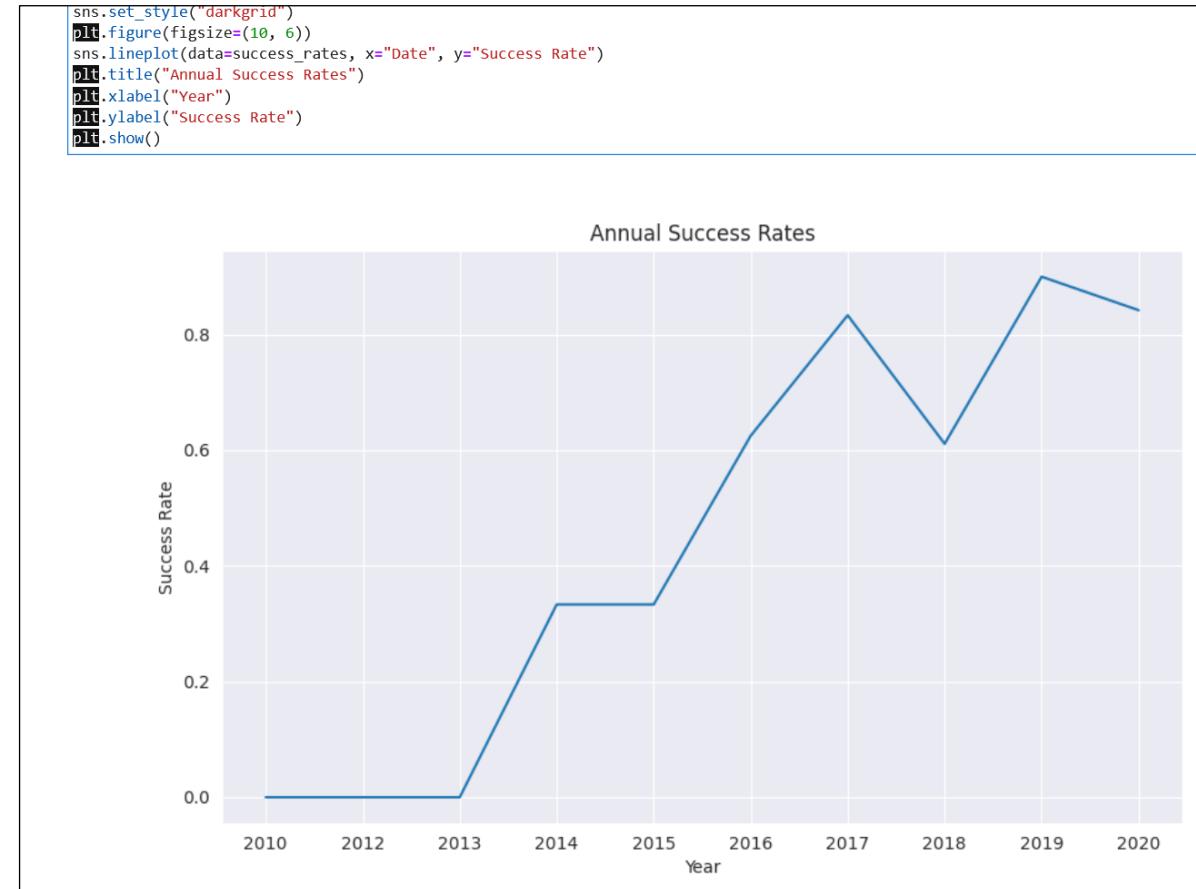
There are no rockets launched for heavy payload mass (greater than 10000Kg) for the VAFB-SLC launch site

```
[14]: # Plot a scatter point chart with x axis to be Payload and y axis to be the orbit, and hue to be the class value
sns.catplot(x="PayloadMass",y="Orbit",hue="Class",data = df, aspect=5)
plt.xlabel("payload Mass (Kg)",fontsize=20)
plt.ylabel("Orbit Type",fontsize=20)
plt.show()
```



With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS

EDA with Visualizations



Success rate since 2013 kept increasing, had a decrease in 2018, and a second fall in 2020.

EDA with SQL

The estimations from SQL queries mainly provided an overview of our data and general insights on some of the variables.

```
[10]: %sql SELECT AVG(PAYLOAD_MASS__KG_) FROM SPACEXTBL WHERE BOOSTER_VERSION='F9 v1.1'  
* sqlite:///my_data1.db  
Done.  
[10]: AVG(PAYLOAD_MASS__KG_)  
2928.4
```

Average payload mass carried by Falcon 9 booster

```
[47]: %sql SELECT BOOSTER_VERSION FROM SPACEXTBL WHERE PAYLOAD_MASS__KG_ between 4000 and 6000 AND "LANDING _OUTCOME"='Success (drone ship)'  
* sqlite:///my_data1.db  
Done.  
[47]: Booster_Version  
F9 FT B1022  
F9 FT B1026  
F9 FT B1021.2  
F9 FT B1031.2
```

Boosters with success outcome in drone ship and have payload mass greater than 4000 but less than 6000

EDA with SQL

```
[29]: %sql SELECT "LANDING _OUTCOME", COUNT(*) as COUNT FROM SPACEXTBL \
    WHERE "DATE" BETWEEN '2010-06-04' and '2017-03-20' \
    GROUP BY "LANDING _OUTCOME" ORDER BY 2 DESC;
```

```
* sqlite:///my_data1.db
```

```
Done.
```

```
[29]: Landing_Outcome COUNT
```

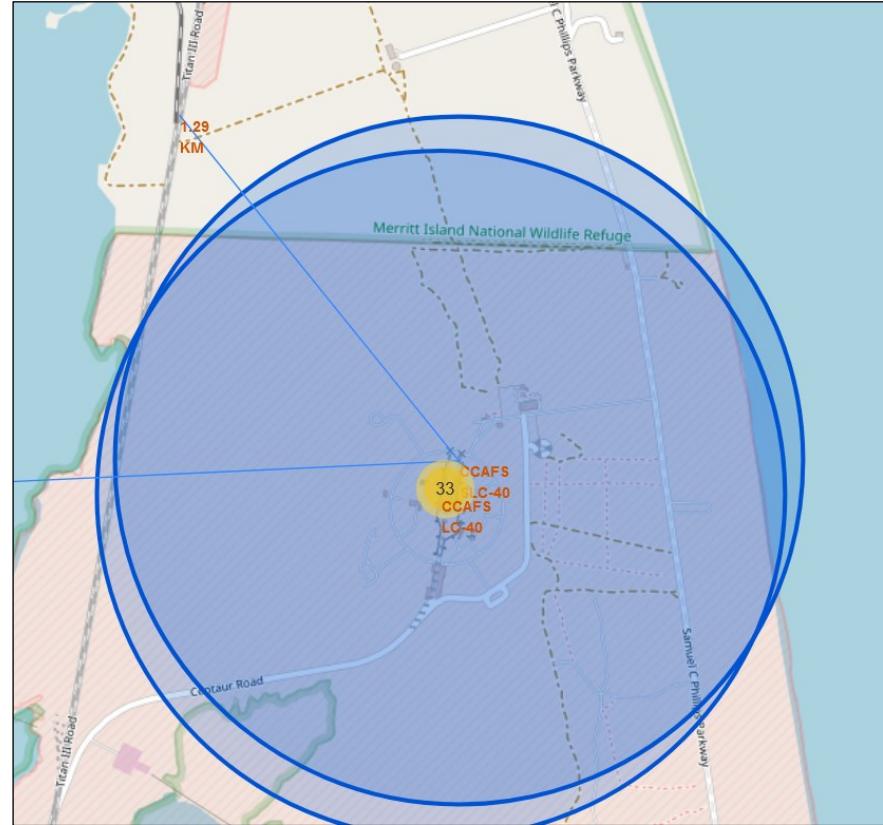
No attempt	10
Success (drone ship)	5
Failure (drone ship)	5
Success (ground pad)	3
Controlled (ocean)	3
Uncontrolled (ocean)	2
Failure (parachute)	2
Precluded (drone ship)	1

Count of successful landing outcomes between June 4, 2010,
and March 20, 2017, in descending order

Geospatial Results (Folium)

Based on geospatial data:

- All launch sites are near railways, likely to transport heavy machinery, pieces, and equipment necessary for the facilities' operations.
- Highways are relatively close to both locations in Florida, but farther away in California.
- All locations are near coastlines.
- And most launch sites are relatively far from cities. The California site is closer to Lompoc.
- Further, airports are close to all three facilities.

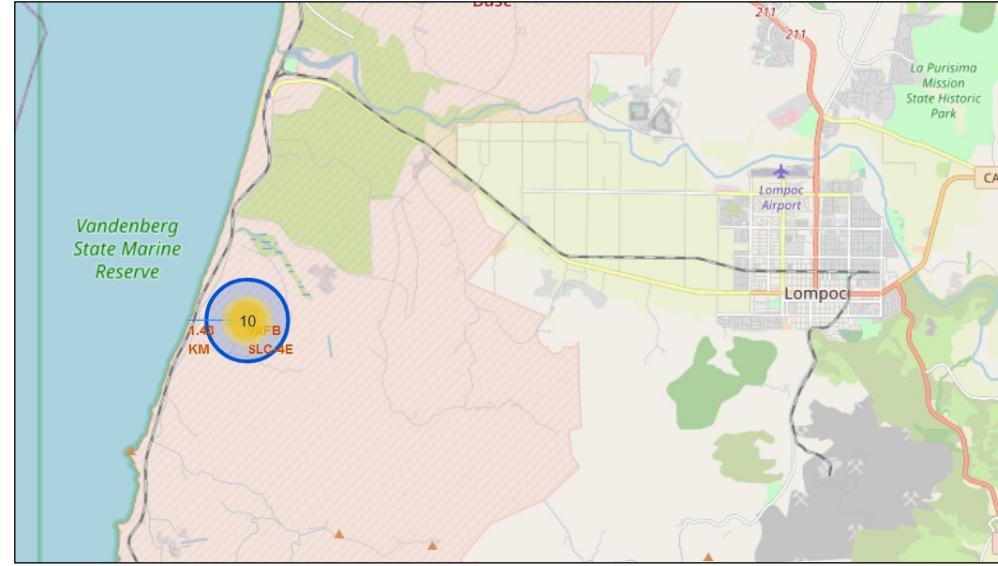


Railway distance from Florida sites.
The coastline is also close to the sites

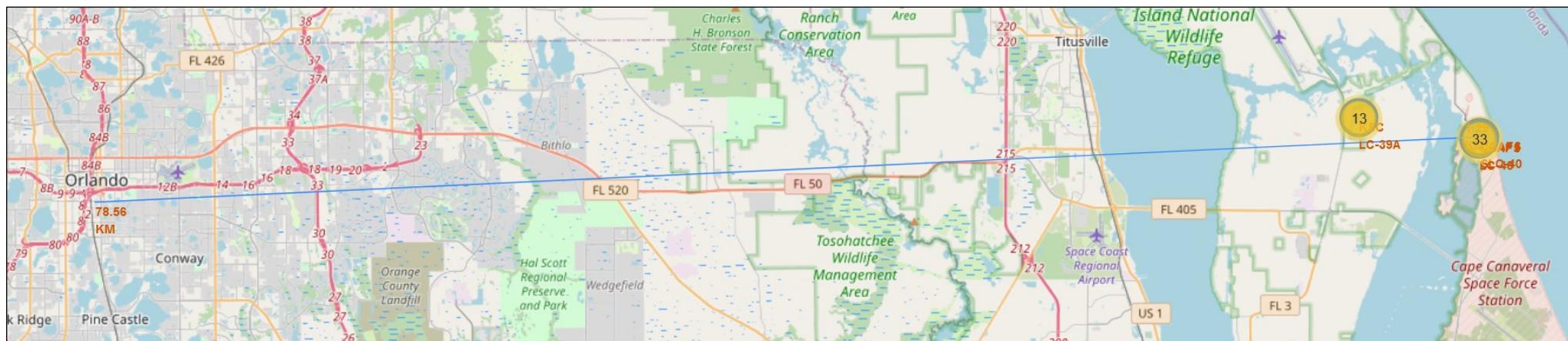
Geospatial Results (Folium)



Distance to coastline from the California launch site

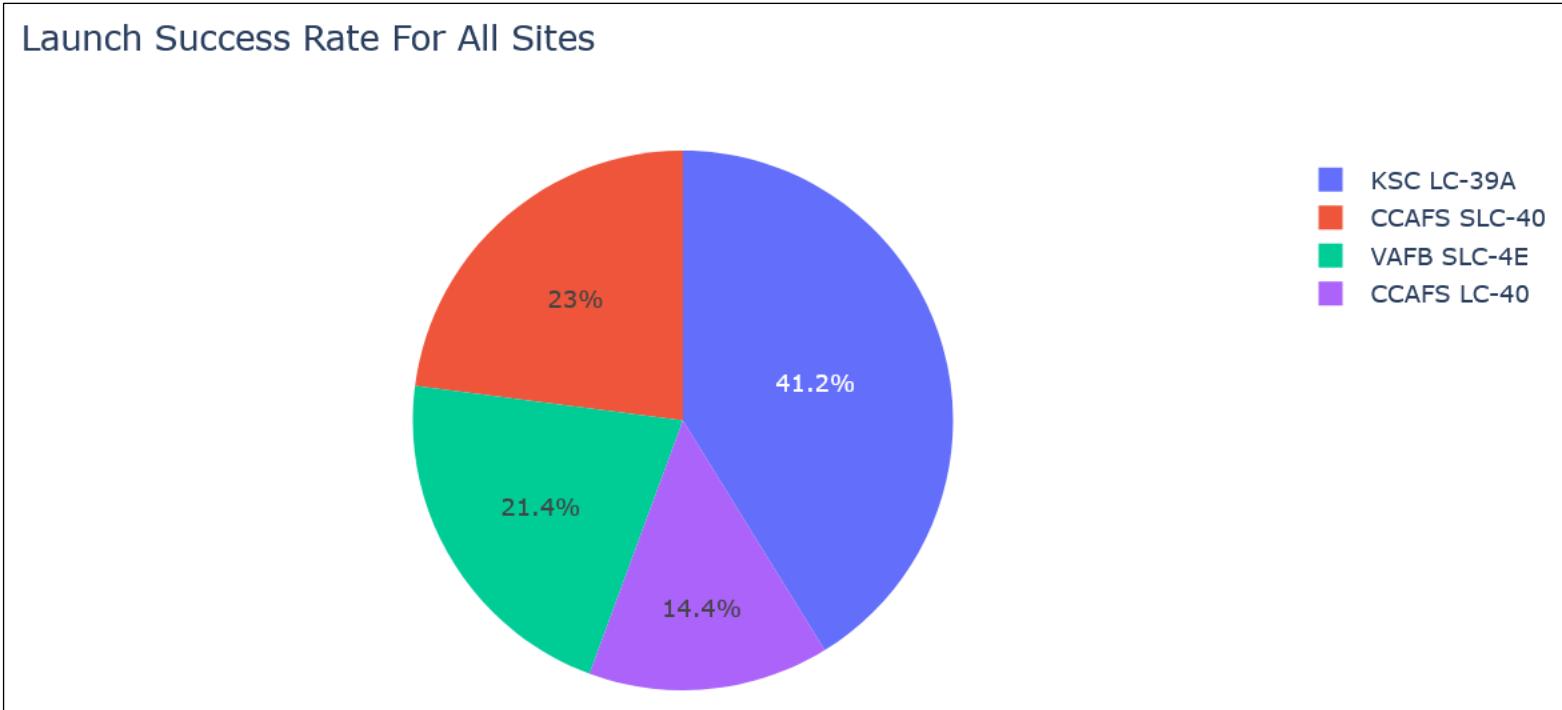


Airport and Lompoc city near VAFB SLC 4E



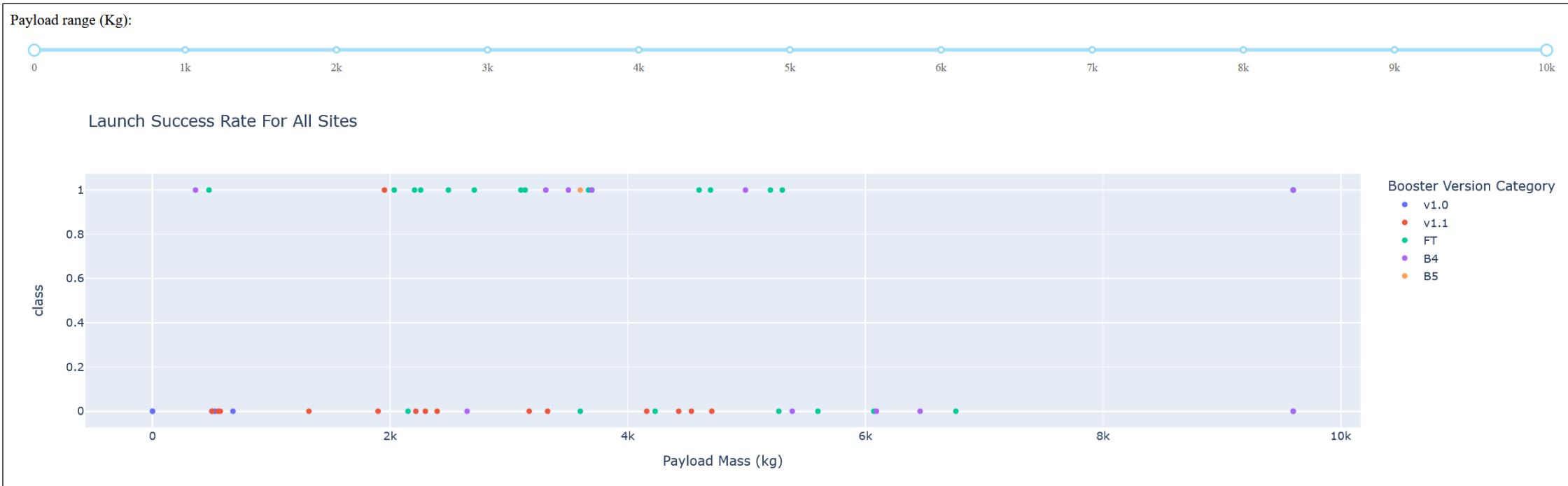
Distance to the nearest large city from a Florida launch site

Dashboard Results



KSC LC-39A had the highest success rate compared to all other launch sites.

Dashboard Results



The success rate of rockets with a payload mass higher
than 6K had minimal success rates



A dark blue background featuring a complex network of glowing blue and purple lines and dots, resembling a neural network or a complex web. The lines vary in brightness, creating a sense of depth and motion.

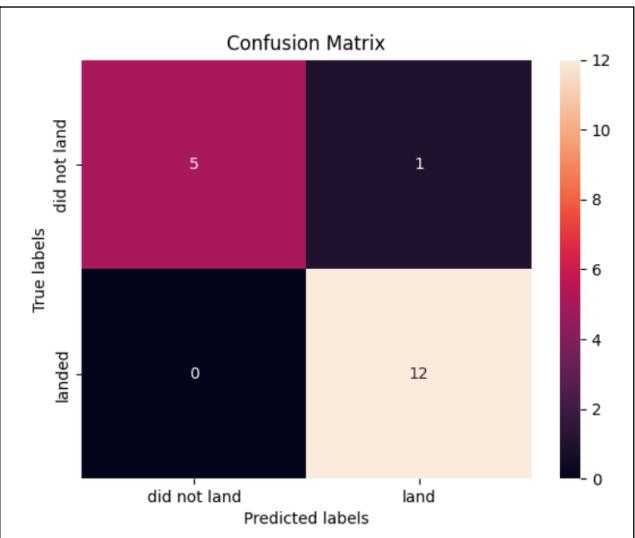
Predictive
Analysis

Predictive Analysis

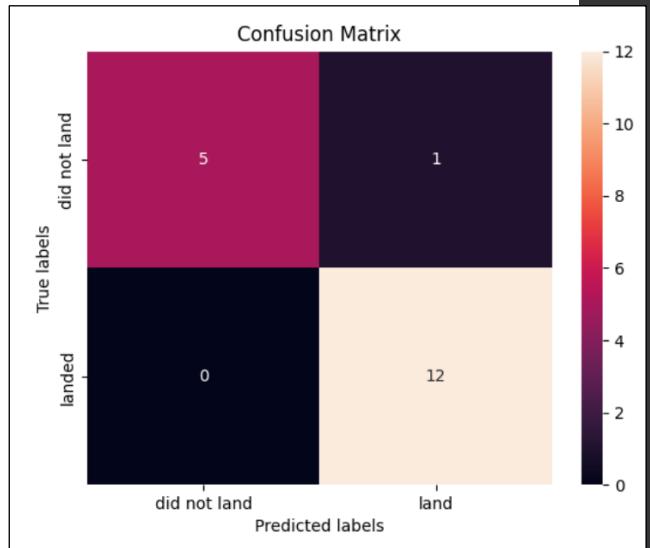
As stated in the methodology section, I prepared the data to run several predictive algorithms (LR, SVM, decision tree, and KNN).

The overall R scores of the predictive methods were as follow:

1. Logistic regression: 0.9444
2. Support vector machine: 0.9444
3. Decision tree: 0.888
4. K-Nearest Neighbor: 0.9444



LR



SVM

```
[52]: tree_cv = GridSearchCV(tree, parameters, cv=10)
tree_cv.fit(X, Y)
tree_cv.best_estimator_
print("tuned hyperparameters :(best parameters)",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)

tuned hyperparameters :(best parameters) {'criterion': 'gini', 'max_depth': 6, 'max_features': 2}
accuracy : 0.8888888888888889

[53]: print("tuned hyperparameters :(best parameters)",tree_cv.best_params_)
print("accuracy :",tree_cv.best_score_)

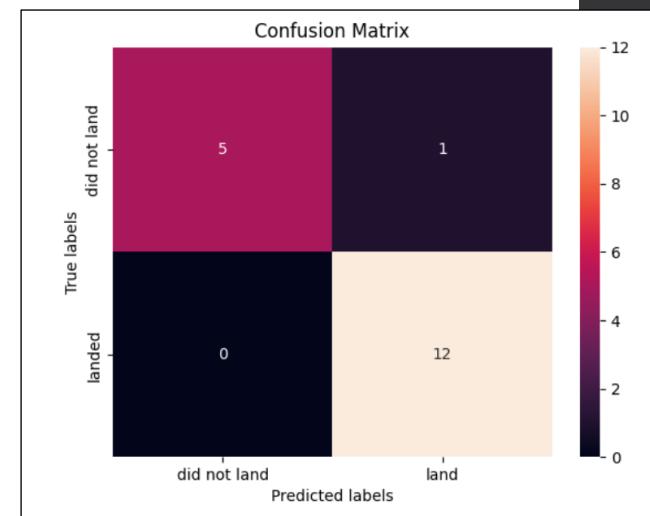
tuned hyperparameters :(best parameters) {'criterion': 'gini', 'max_depth': 6, 'max_features': 2}
accuracy : 0.8888888888888889
```

TASK 9

Calculate the accuracy of tree_cv on the test data using the method `score`:

```
[56]: print('score on train data: ', tree_cv.score(X_train, Y_train)) # R2 score on train data
print('score on test data: ', tree_cv.score(X_test, Y_test)) # R2 score on test data
score on train data:  0.8888888888888888
score on test data:  0.8888888888888888
```

Decision tree



KNN

Conclusions



Conclusions

Based on the EDA and the geospatial data analysis:

- The success rate at a launch site is likely to increase after several launches have taken place in that location.
- KSC LC-39A had the highest success rate compared to all other launch sites.
- There are no rockets launched for heavy payload mass (greater than 10000Kg) for the VAFB-SLC launch site.
- With heavy payloads the successful landing or positive landing rate are more for Polar, LEO and ISS.
- The geographic position of launch sites (near coastlines and railways, and further from urban centers) is highly likely due to safety and logistics.
- LR, SVM, and KNN had high R scores, eliciting high accuracy.
 - However, based on their confusion matrixes, false positives might remain an issue.
 - Thus, future changes might include collecting more data and review feature selection processes.



Thank you