IBM Developer
SKILLS NETWORK

Winning Space Race with Data Science

Sai Pyae Sone Thu
10/10/2024

Outline

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

Executive Summary

- This project utilized various data collection methods, including web scraping and API calls, to analyze SpaceX launch data. Through Exploratory Data Analysis (EDA) and machine learning, the goal was to develop a classification model that predicts whether the Falcon 9 first stage will successfully land.
- The results showed that decision trees were the best model for predicting the success of the landings.

Introduction

- This project utilized various data collection methods, including web scraping and API calls, to analyze SpaceX launch data. Through Exploratory Data Analysis (EDA) and machine learning, the goal was to develop a classification model that predicts whether the Falcon 9 first stage will successfully land.
- The results showed that decision trees were the best model for predicting the success of the landings.

Section 1

Methodology

Methodology

Executive Summary

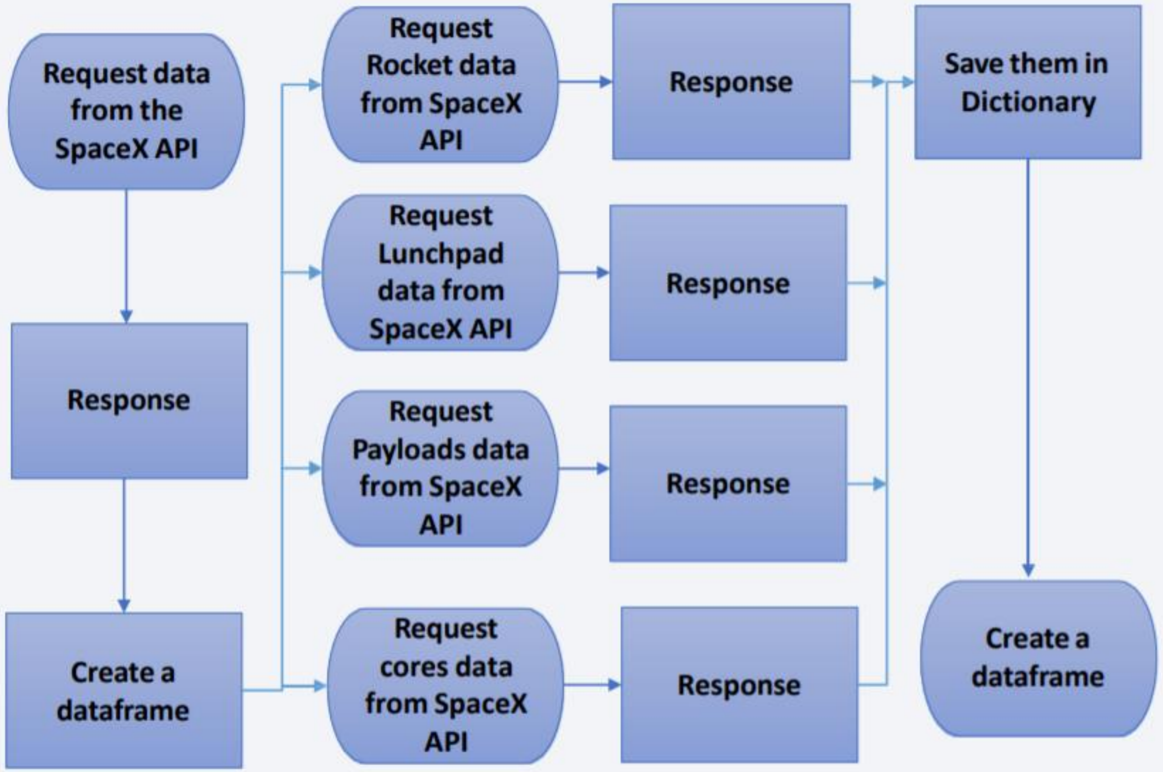
- Data collection methodology:
  - Describe how data was collected
- Perform data wrangling
  - Describe how data was processed
- Perform exploratory data analysis (EDA) using visualization and SQL
- Perform interactive visual analytics using Folium and Plotly Dash
- Perform predictive analysis using classification models
  - How to build, tune, evaluate classification models

Data Collection

- This project utilized various data collection methods, including web scraping and API calls, to analyze SpaceX launch data. Through Exploratory Data Analysis (EDA) and machine learning, the goal was to develop a classification model that predicts whether the Falcon 9 first stage will successfully land.
- The results showed that decision trees were the best model for predicting the success of the landings.

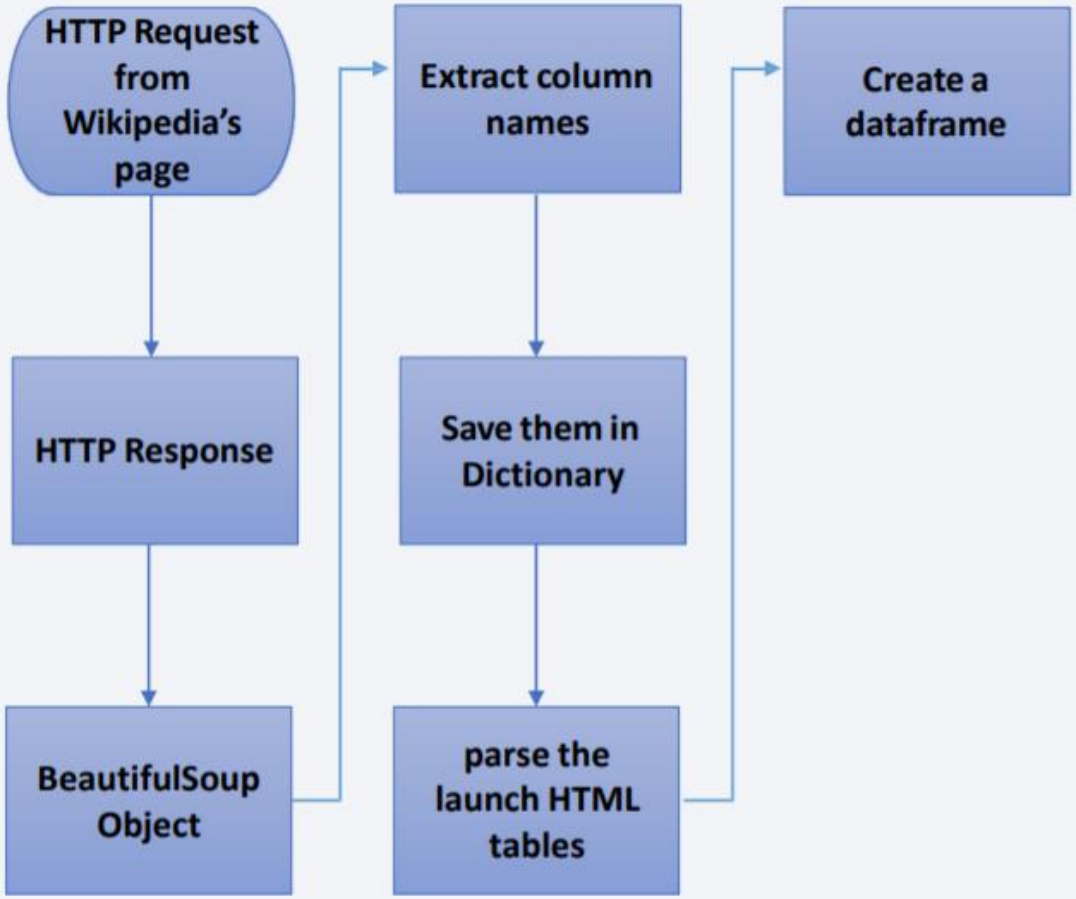
Data Collection – SpaceX API

- We'll be collecting launch data from SpaceX API, First we request launch data from SpaceX API using the GET command (requests.get), then we create a pandas dataframe from the response, After that we make several sub requests to get more detailed and consistent information about the IDs stored in the dataframe.
- With the help of some helper functions, we save the responses into a dictionary, and then we transform it into a dataframe, which is our data set.



Data Collection - Scraping

- We will be performing web scraping to collect Falcon 9 historical launch records from a Wikipedia page. First we perform an HTTP GET(using requests.get command)method to request the Falcon9 Launch HTMLpage, as an HTTP response. Then we create a BeautifulSoup object from the HTML response, We extract the column names from the object and use it as dictionary keys.
- We parse the HTML tables and fill the dictionary keys with launch records from table rows, and finally we transform it into a dataframe.



Data Wrangling

- Exploratory data analysis is an important step while preprocessing data, it is useful to find some patterns in the data and determine what would be the label for training supervised models.
- This process was done in the following order:

1. First thing to do is to identify the data types of the columns.
2. Determine the number of values for each attribute.
3. Calculate the percentage of the missing values.
4. To determine the label, weapply zero/one hot encoding to the “Outcome” column to classify landing to either 1(Success) of 0 (Failure)

EDA with SQL

- In order to better understand the datasets, we ran the following SQL queries:

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 achieved.
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

EDA with Data Visualization

- In order to understand the relations between different features, we visualize the data by plotting scatter plots, bar charts and line charts, it helps finding hidden patterns in data and gain insights about the dataset.

1. Pay load mass against the Flight number.
2. Lunch site against the Flight number.
3. Lunch site against the Pay load mass.
4. Orbit type against Class success rate.
5. Flight number against Orbit type.
6. Orbit type against the Pay load mass.
7. launch success yearly trend.

Build an Interactive Map with Folium

- Here, we complete the interactive visual analytics using Folium.
- First we create Folium map object, with an initial center location around Nasa Johnson space center, Houston-Texas.
- We add a circle on the map for each launch site from the dataset by creating a folium circle and folium marker, now the launch sites are marked on the map which means we can see which one is proximate to the equator line or close to a coastline.
- In order to mark the success/failure launches, we create a marker on the map for each launch record from the dataset, a green marker indicates a successful lunching and a red one indicates failure,
- we need to explore and analyze the proximities of launch sites, we calculate the distance between the launch site and its proximities and then we draw a polyline between them.

Build a Dashboard with Plotly Dash

- Here, we complete the interactive visual analytics using Folium.
- First we create Folium map object, with an initial center location around Nasa Johnson space center, Houston-Texas.
- We add a circle on the map for each launch site from the dataset by creating a folium circle and folium marker, now the launch sites are marked on the map which means we can see which one is proximate to the equator line or close to a coastline.
- In order to mark the success/failure launches, we create a marker on the map for each launch record from the dataset, a green marker indicates a successful lunching and a red one indicates failure,
- we need to explore and analyze the proximities of launch sites, we calculate the distance between the launch site and its proximities and then we draw a polyline between them.

Predictive Analysis (Classification)

- Now that we finished the exploratory analysis, the next step is to determine the training labels and build a predictor using machine learning algorithms. After using the 'Class' column as the label, first thing to do is normalizing the data. We split the normalized data into test/train sets, The training data is divided into validation data, a second set used for training data.
- For the model development phase, we use the following algorithms:

1. Logistic regression
2. Support vector machine
3. Decision trees
4. K nearest neighbor

- We build a grid search object for each of the algorithms and f i t it to find the best parameters of the model(hyper parameters tuning), then we choose the most accurate model.

Results

- Success rate increased noticeably from 2013 and on.
- Launch site and the orbit type are the features with the largest effect on the outcome.
- KNN and SVM models have a validation set accuracy of 83% and an out of sample accuracy of 77%.

Section 2

Insights drawn from EDA

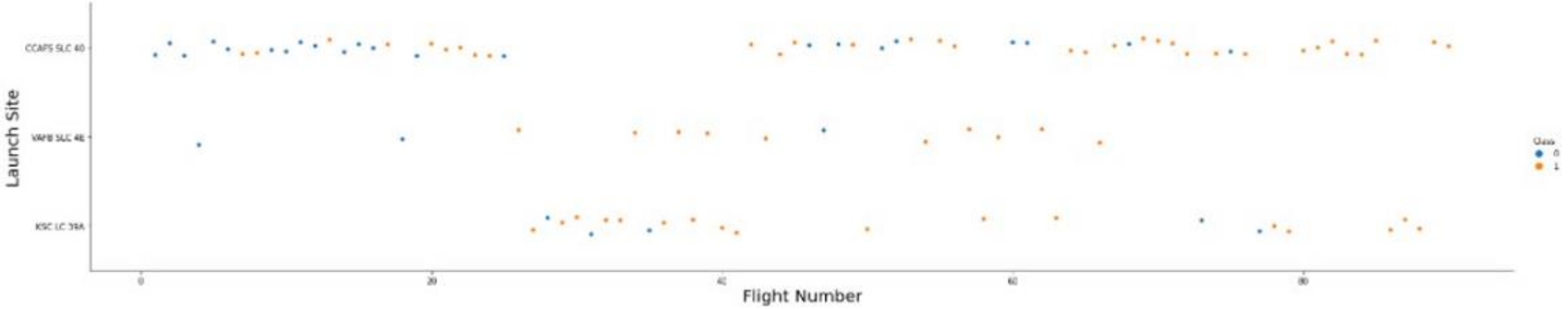
Flight Number vs. Launch Site

scatter plot of Flight Number vs. Launch Site

TASK 1: Visualize the relationship between Flight Number and Launch Site

Use the function `catplot` to plot `FlightNumber` vs `LaunchSite`, set the parameter `x` parameter to `FlightNumber`,set the `y` to `Launch Site` and set the parameter `hue` to `'class'`

In [4]:
```
# Plot a scatter point chart with x axis to be Flight Number and y axis to be the launch site, and hue to be the class value
sns.catplot(y="LaunchSite", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Now try to explain the patterns you found in the Flight Number vs. Launch Site scatter point plots.

In [5]:
```
#CCAFS SLC-40 flights have higher success rate as the flight number increases
```

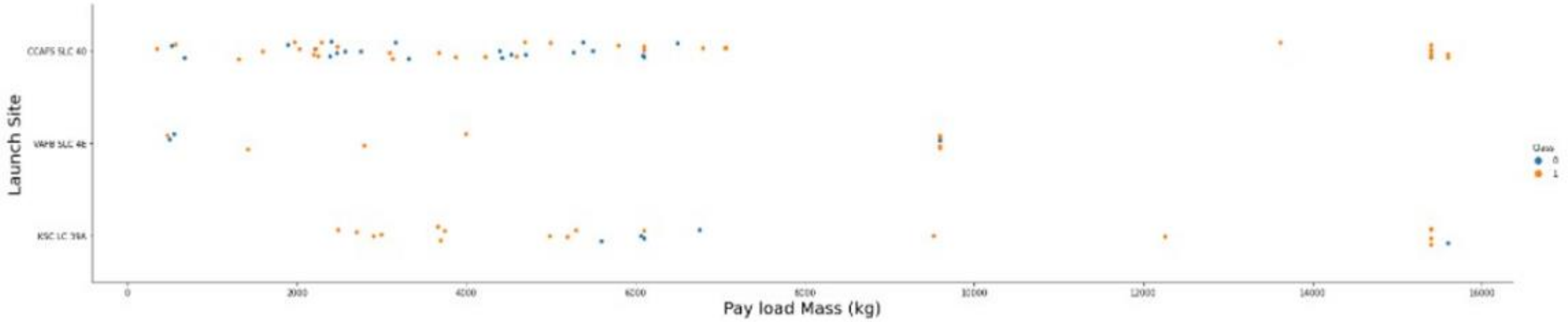
Payload vs. Launch Site

scatter plot of Payload vs. Launch Site

TASK 2: Visualize the relationship between Payload and Launch Site

We also want to observe if there is any relationship between launch sites and their payload mass.

In [6]:
```
# 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("Pay load Mass (kg)",fontsize=20)
plt.ylabel("Launch Site",fontsize=20)
plt.show()
```



Now try to explain any patterns you found in the Payload Vs. Launch Site scatter point chart.

In [7]:
```
#small pay load mass values have bad influence on CCAFS SLC-40 flights
```

Success Rate vs. Orbit Type

bar chart for the success rate of each orbit type

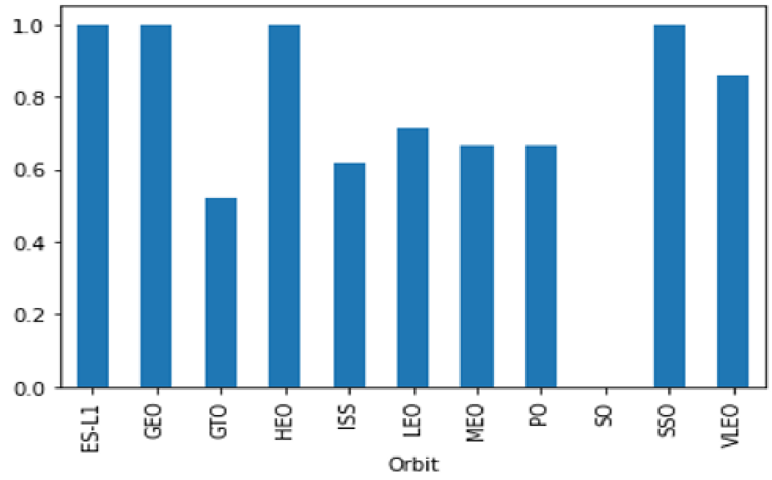
TASK 3: Visualize the relationship between success rate of each orbit type

Next, we want to visually check if there are any relationship between success rate and orbit type.

Let's create a `bar chart` for the sucess rate of each orbit

In [8]:
```
# HINT use groupby method on Orbit column and get the mean of Class column
mean = df.groupby(['Orbit'])['Class'].mean()
mean.plot(kind = 'bar')
```

Out[8]: <AxesSubplot:xlabel='Orbit'>



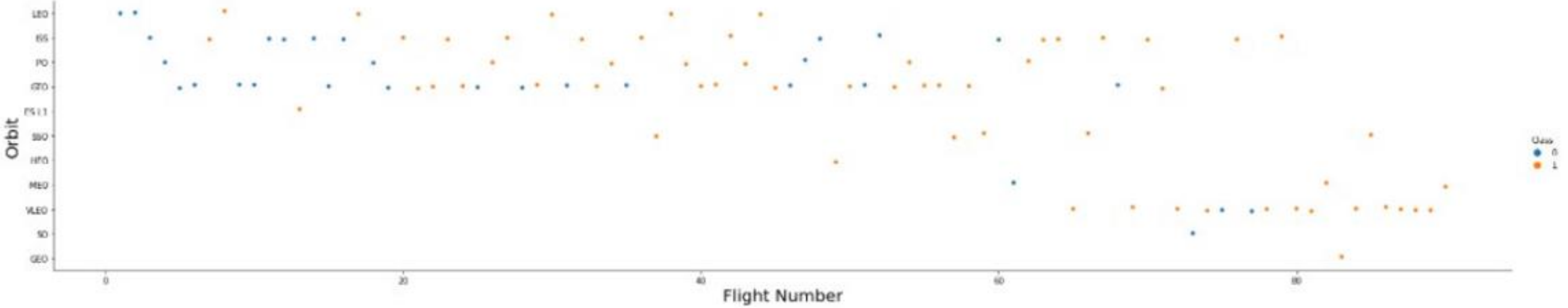
Flight Number vs. Orbit Type

scatter point of Flight number vs. Orbit type

TASK 4: Visualize the relationship between FlightNumber and Orbit type

For each orbit, we want to see if there is any relationship between FlightNumber and Orbit type.

In [10]:
```
# Plot a scatter point chart with x axis to be FlightNumber and y axis to be the Orbit, and hue to be the class value
sns.catplot(y="Orbit", x="FlightNumber", hue="Class", data=df, aspect = 5)
plt.xlabel("Flight Number",fontsize=20)
plt.ylabel("Orbit",fontsize=20)
plt.show()
```



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.

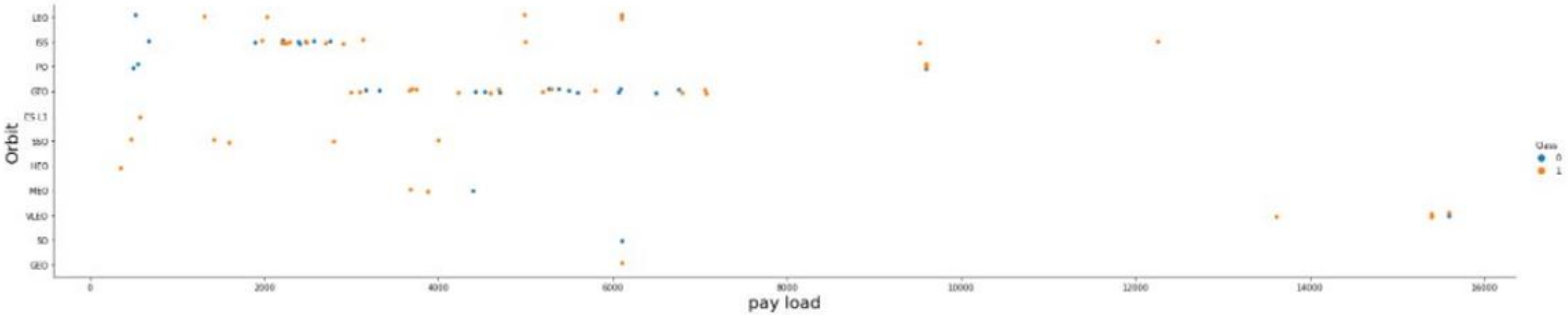
Payload vs. Orbit Type

scatter point of payload vs. orbit type

TASK 5: Visualize the relationship between Payload and Orbit type

Similarly, we can plot the Payload vs. Orbit scatter point charts to reveal the relationship between Payload and Orbit type

In [21]:
```
# 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(y="Orbit", x="PayloadMass", hue="Class", data=df, aspect = 5)
plt.ylabel("Orbit",fontsize=20)
plt.xlabel("pay load",fontsize=20)
plt.show()
```



You should observe that Heavy payloads have a negative influence on GTO orbits and positive on GTO and Polar LEO (ISS) orbits.

Launch Success Yearly Trend

line chart of yearly average success rate

You can plot a line chart with x axis to be `Year` and y axis to be average success rate, to get the average launch success trend.

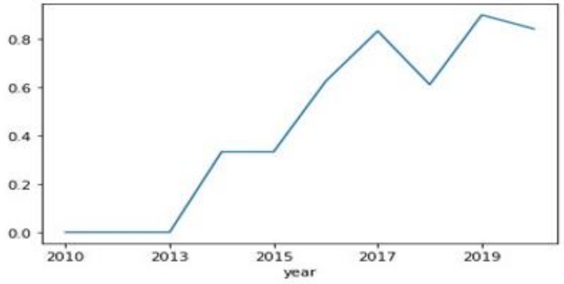
The function will help you get the year from the date:

In [12]:
```python
# A function to Extract years from the date
year=[]
def Extract_year(date):
    for i in df["Date"]:
        year.append(i.split("-")[0])
    return year
```

In [13]:
```python
Extract_year(df)
```

In [14]:
```python
Class = df['Class'].tolist()
new_df = pd.DataFrame(list(zip(year,Class)),columns = ['year','class'])
mean = new_df.groupby(['year'])['class'].mean()
mean
```

In [15]:
```python
# Plot a line chart with x axis to be the extracted year and y axis to be the success rate
line_plt = mean.plot(kind='line')
```



you can observe that the sucess rate since 2013 kept increasing till 2020

All Launch Site Names

names of the unique launch sites

Task 1

Display the names of the unique launch sites in the space mission

In [5]:
```
%sql SELECT DISTINCT(LAUNCH_SITE) FROM SPACX;
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[5]:

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

Launch Site Names Begin with 'CCA'

5 records where launch sites begin with `CCA`

Task 2

Display 5 records where launch sites begin with the string 'CCA'

```
In [20]: %sql SELECT * from SPACX where LAUNCH_SITE like 'CCA%' LIMIT 5;
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[20]:

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

- Calculate the total payload carried by boosters from NASA

Task 3

Display the total payload mass carried by boosters launched by NASA (CRS)

In [28]:
```
%sql SELECT SUM(PAYLOAD_MASS__KG_) as total_payload_mass  from SPACX where CUSTOMER = 'NASA (CRS)';
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[28]:

total_payload_mass
45596

Average Payload Mass by F9 v1.1

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

Task 4

Display average payload mass carried by booster version F9 v1.1

In [27]:
```
%sql SELECT avg(PAYLOAD_MASS__KG_) as average_payload_mass from SPACX where BOOSTER_VERSION ='F9 v1.1';
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[27]:

average_payload_mass
2928

First Successful Ground Landing Date

- Find the dates of the first successful landing outcome on ground pad

Task 5

List the date when the first successful landing outcome in ground pad was acheived.

Hint:Use min function

In [30]:
```
%sql select min(DATE) from SPACX where LANDING__OUTCOME = 'Success (ground pad)';
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[30]:

1
2015-12-22

Successful Drone Ship Landing with Payload between 4000 and 6000

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

Task 6

List the names of the boosters which have success in drone ship and have payload mass greater than 4000 but less than 6000

```
31]: %%sql
SELECT BOOSTER_VERSION FROM SPACX
WHERE LANDING__OUTCOME = 'Success (drone ship)' AND PAYLOAD_MASS__KG_ BETWEEN 4000 AND 6000;
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

31]:

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

Total Number of Successful and Failure Mission Outcomes

Calculate the total number of successful and failure mission outcomes

Task 7

List the total number of successful and failure mission outcomes

In [36]:
```
%sql SELECT MISSION_OUTCOME,COUNT(MISSION_OUTCOME) AS Count FROM SPACX GROUP BY MISSION_OUTCOME;
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[36]:

mission_outcome	COUNT
Failure (in flight)	1
Success	99
Success (payload status unclear)	1

Boosters Carried Maximum Payload

List the names of the booster which have carried the maximum payload mass

Task 8

List the names of the booster_versions which have carried the maximum payload mass. Use a subquery

In [37]:
```
%%sql
SELECT BOOSTER_VERSION FROM SPACX WHERE PAYLOAD_MASS__KG_ =(select max(PAYLOAD_MASS__KG_) from SPACX);
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[37]:

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

- List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

Task 9

List the failed landing_outcomes in drone ship, their booster versions, and launch site names for in year 2015

In [44]:
```
%%sql
SELECT LANDING__OUTCOME,BOOSTER_VERSION,LAUNCH_SITE FROM SPACX
WHERE LANDING__OUTCOME = 'Failure (drone ship)' AND DATE LIKE '%2015%';
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[44]:

landing__outcome	booster_version	launch_site
Failure (drone ship)	F9 v1.1 B1012	CCAFS LC-40
Failure (drone ship)	F9 v1.1 B1015	CCAFS LC-40

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

- 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

Task 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

In [51]:

```
%%sql
SELECT LANDING__OUTCOME,COUNT(LANDING__OUTCOME) AS COUNT FROM SPACX
WHERE DATE > '2010-06-04' and DATE <'2017-03-20'
GROUP BY LANDING__OUTCOME
ORDER BY COUNT DESC;
```

```
 * ibm_db_sa://sdw99696:***@54a2f15b-5c0f-46df-8954-7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud:32733/bludb
Done.
```

Out[51]:

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

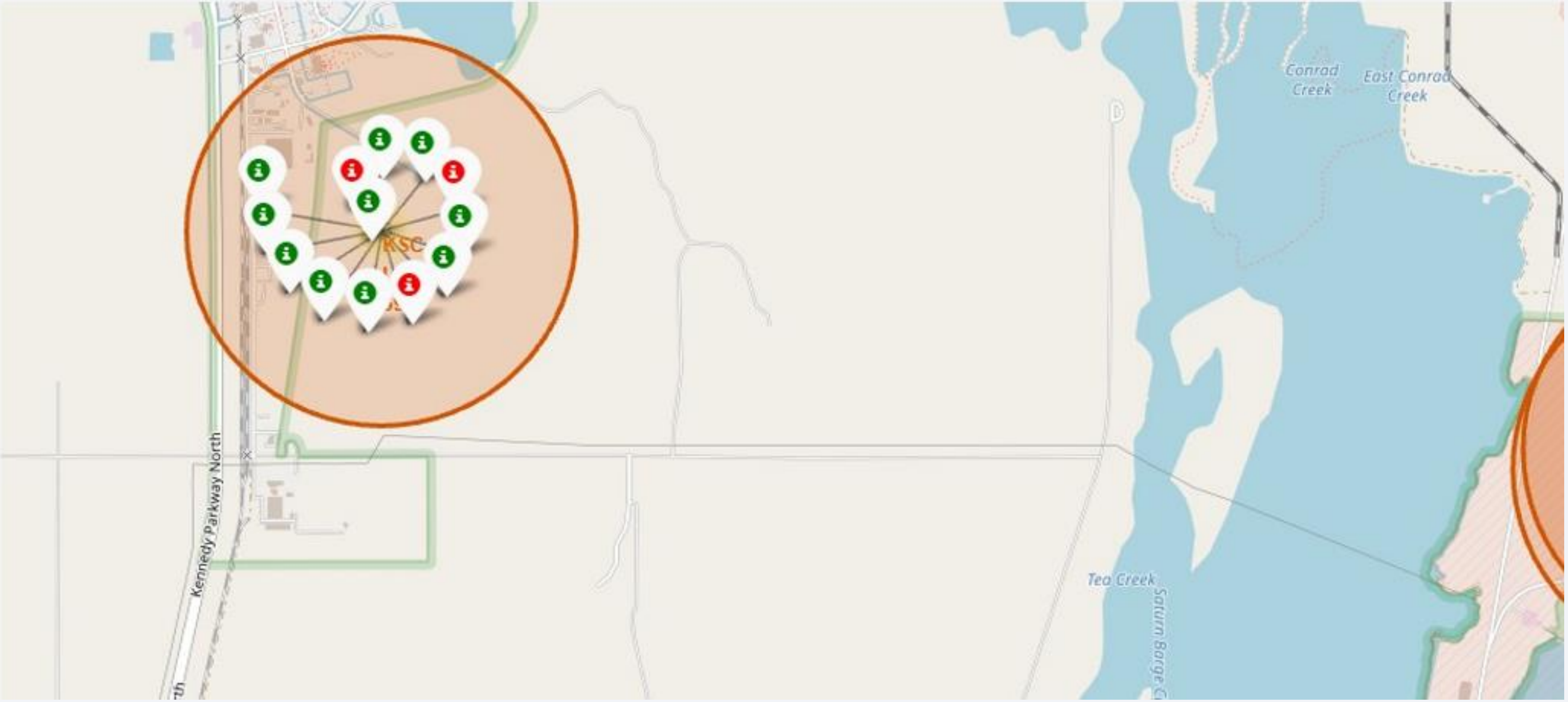

Section 3
Launch Sites
Proximities Analysis

Launch Sites Marking

The markers on this maps show the launch site locations on the map.

Mark the Success/Failed Launches for Each Site on the Map

A green marker represents a successful landing outcome, while a red one represents failure.



Distances between a launch site to its proxomities

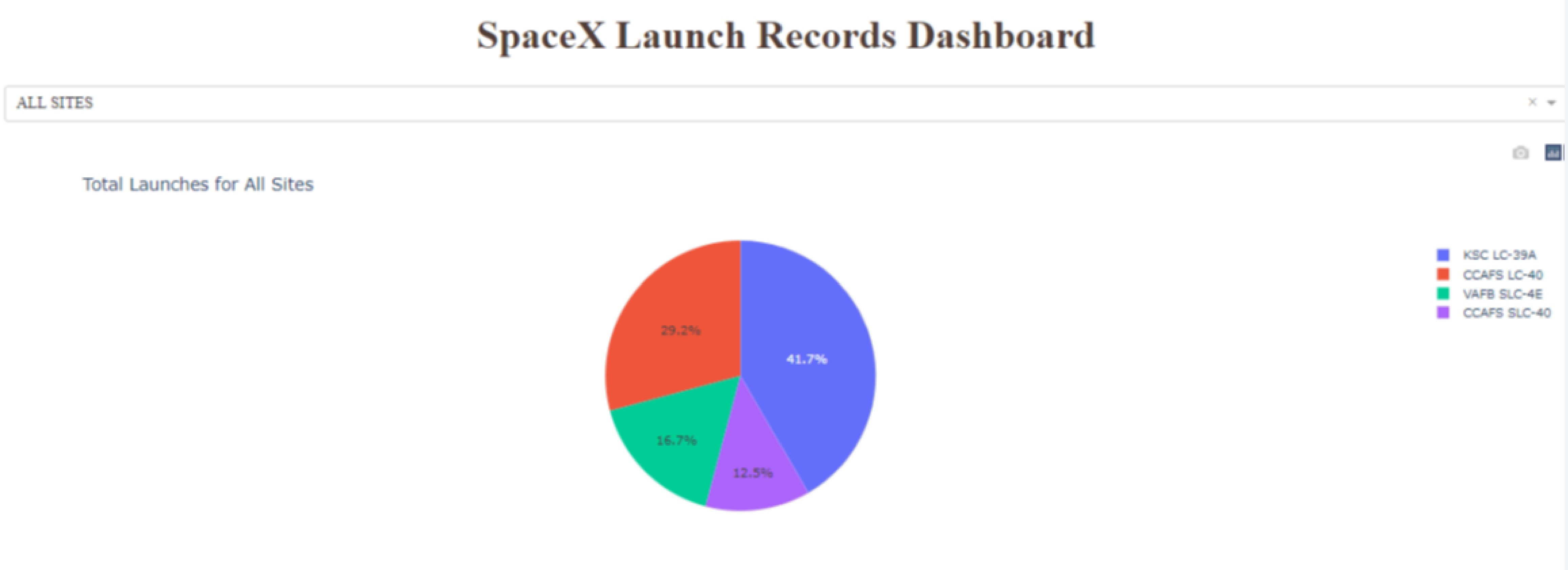
- The blue line represents the distance between the lunch site and the closest coastline.

Section 4

Build a Dashboard with Plotly Dash

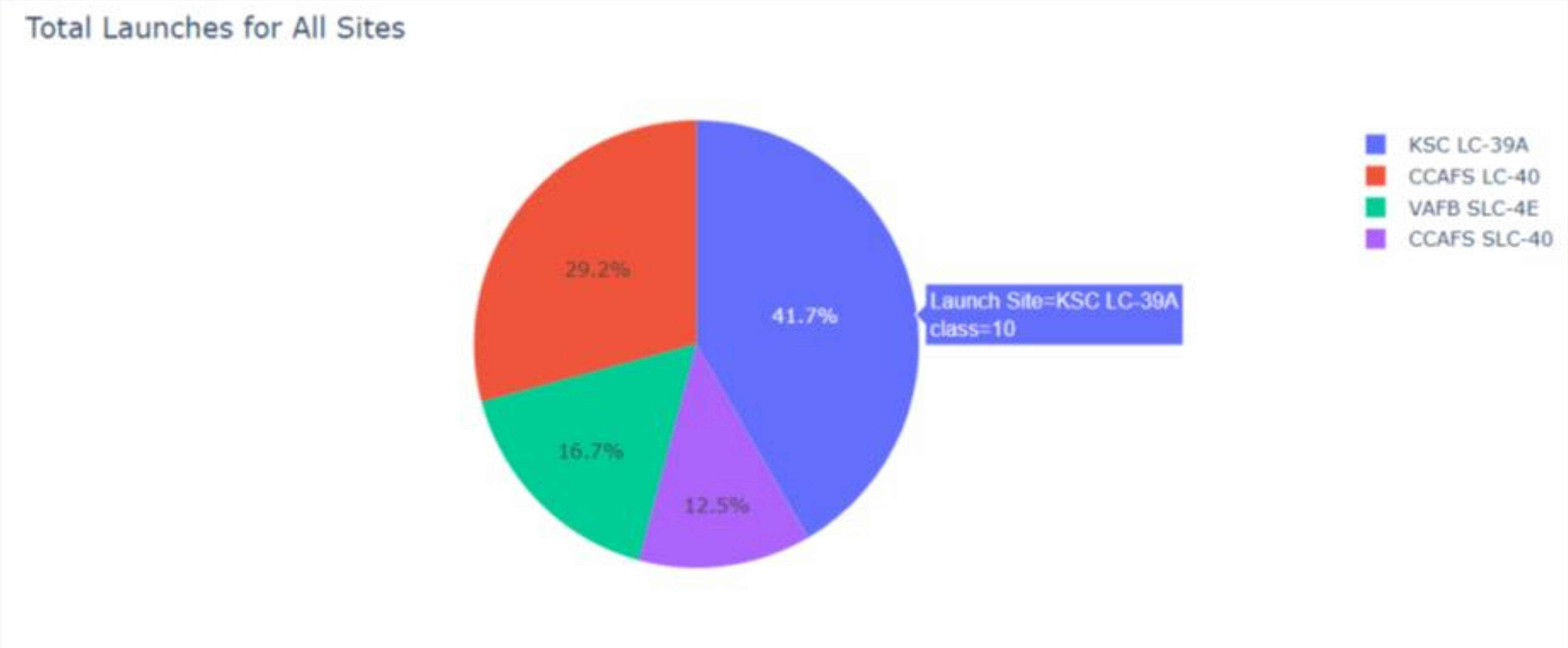
Launch Success Dashboard for All Sites

Showing the screenshot of launch success count for all sites, in a Piechart.



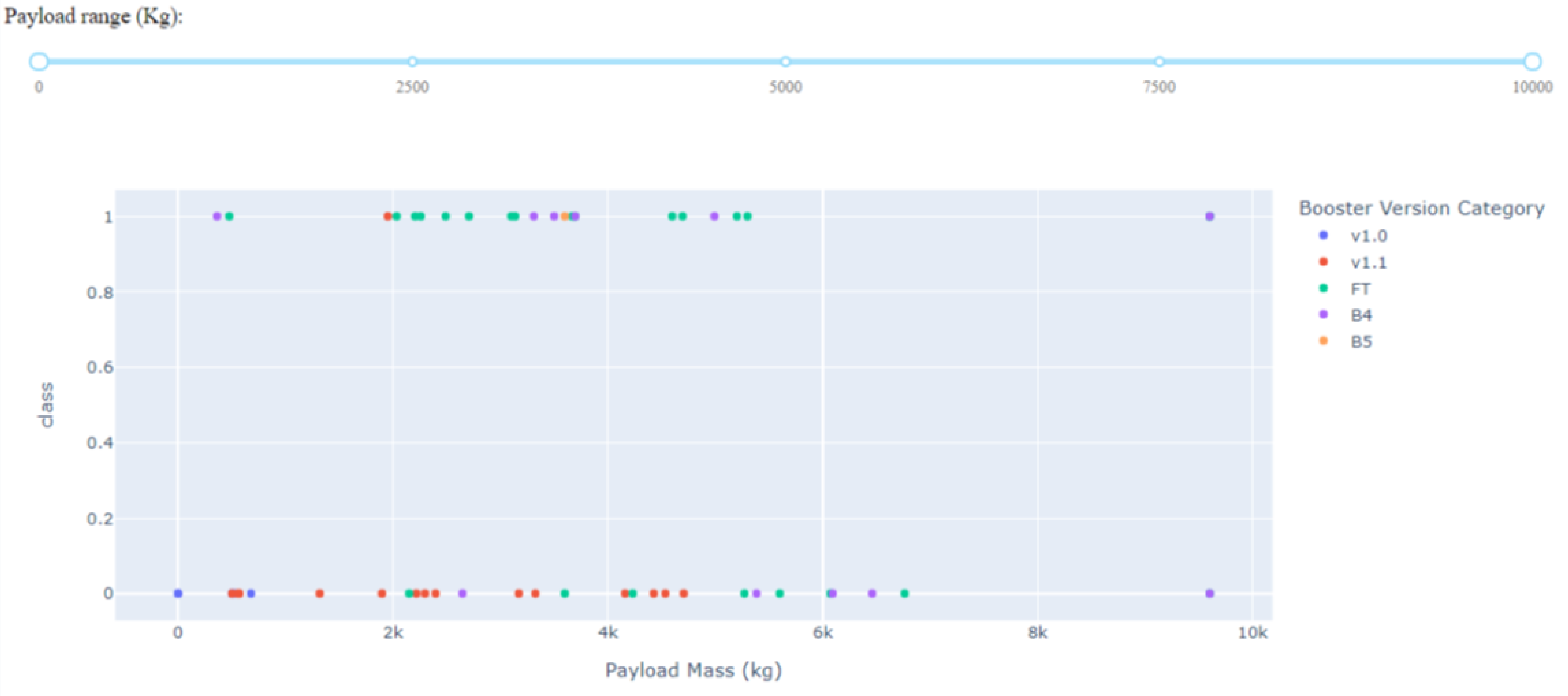
Pie Chart for Highest Launch Success Site

Show the screenshot of the piechart for the launch site with highest launch success ratio



Payload Vs Launch Outcome

Payload vs. Launch Outcome scatter plot for all sites, with different payload selected in the range slider

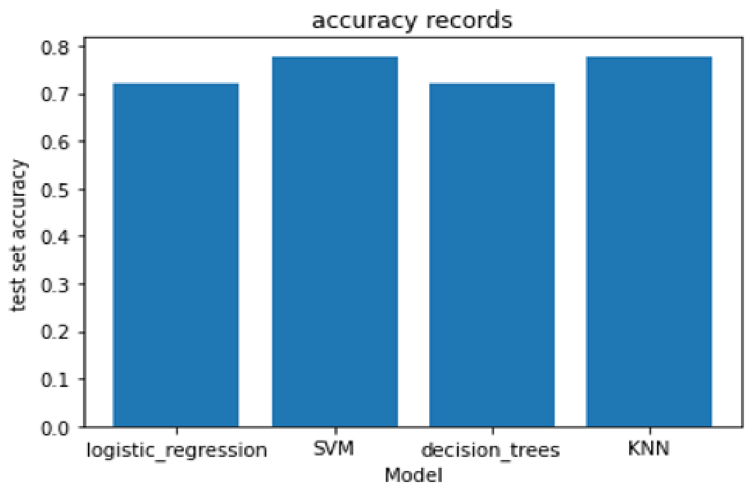


Section 5

Predictive Analysis (Classification)

Classification Accuracy

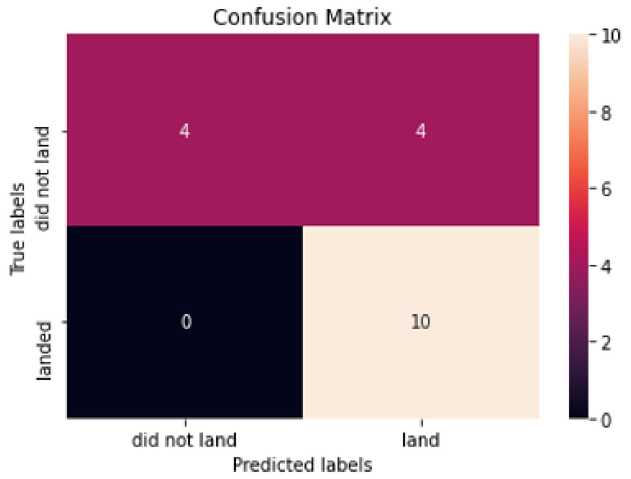
```
In [44]: x = [lr.score(X_test,Y_test),svm.score(X_test,Y_test),tree.score(X_test,Y_test),KNN.score(X_test,Y_test)]
         y = ['logistic_regression','SVM','decision_trees','KNN']
         plt.bar(y, x)
         plt.title('accuracy records')
         plt.xlabel('Model')
         plt.ylabel('test set accuracy')
         plt.show

Out[44]: <function matplotlib.pyplot.show(*args, **kw)>
```



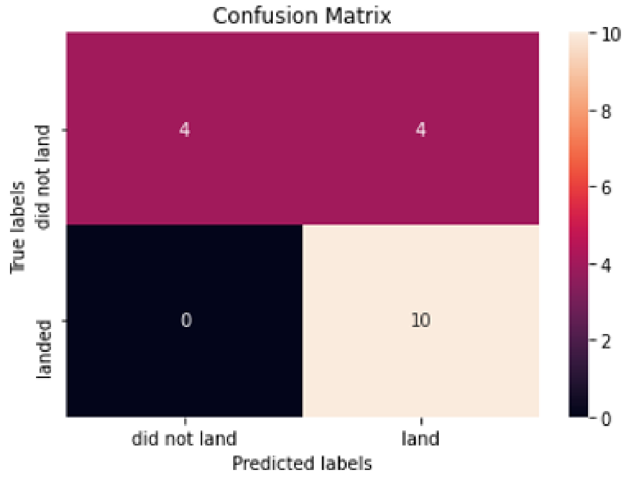

```
In [ ]: #SVM and KNN have the highest out of sample accuracy, thus they are the most accurate models.
```

Confusion Matrix

In [36]:
```
yhat = knn_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



In [22]:
```
yhat=svm_cv.predict(X_test)
plot_confusion_matrix(Y_test,yhat)
```



Conclusions

Not all the data is important, the collected data may contain irrelevant columns and it is normal to drop them. Visualizing data is a good way of determining what features have the strongest effect. SQL queries provide wider scope to explore datasets in comparison with traditional EDA. SVM and KNN models are the most reliable since they have the highest out of sample accuracy and f1-score.

Appendix

- Include any relevant assets like Python code snippets, SQL queries, charts, Notebook outputs, or data sets that you may have created during this project

Thank you!