



IBM Developer  
SKILLS NETWORK

# Winning Space Race with Data Science


<René Heinitz>

<29.10.2021>



# Outline

---

	Executive Summary	3
	Introduction	4
	Methodology	5
	Results	16
	Conclusion	29
	Appendix	30

# Executive Summary



At a glance - What insights will be shown in this presentation?

- ? Main question: What factors can be identified that have a positive influence on predicting whether the first launch module of the Space X rocket can be returned safely to Earth?
- ↻ The **more flight tests** have been carried out at a Space X launch location, the **more likely** it is that the **landing of the first launch module will be successful**
- 🎯 Flight tests that target an **orbit range of 1000km from earth (LEO) are more successful** when it comes to bringing the first launch module safely back to earth
- 🎯 In addition to the LEO orbit range, **above-average success can be achieved in the ES-L1, SSO, HEO and GEO orbit ranges** when it comes to bringing the first launch module safely back to Earth
- 📈 The **longer the Space X program runs**, the **more successful** the organisation will be in bringing the first test module safely back to Earth
- 📍 **Space X has 4 Launch locations:** VAFB SLC-4E, KSC LC-39A, CCAFS SLC-40, CCAFS
- 🌳 If you want to predict the success of a safe landing of the first launch module, you should **use a data model based on a decision tree**



# Introduction



## Project background and context

- Space X advertises the cost of launching a Falcon 9 rocket on its website as \$62 million; other providers cost up to \$165 million each, with much of the savings coming from the fact that Space X can reuse the first stage. Being able to determine whether the first stage will land can also determine the cost of a launch.
- This is the capstone project of the IBM Data Science Certificate Program. Participants of the program should show the skills they have learned in the past months by demonstrating them in a real project.

## Several milestones are to be achieved in the project



**Data collection** (e.g., collecting data from a homepage)



**Data wrangling** (e.g., deleting missing values in the data set)



**Exploratory data analysis (EDA) of the data** (e.g., the identification of the correlation between a successful landing of a launch modul and possible impact factors)



**Interactive visual analytics of the data** (e.g., the identification of the launch locations from Space X in the U.S.)



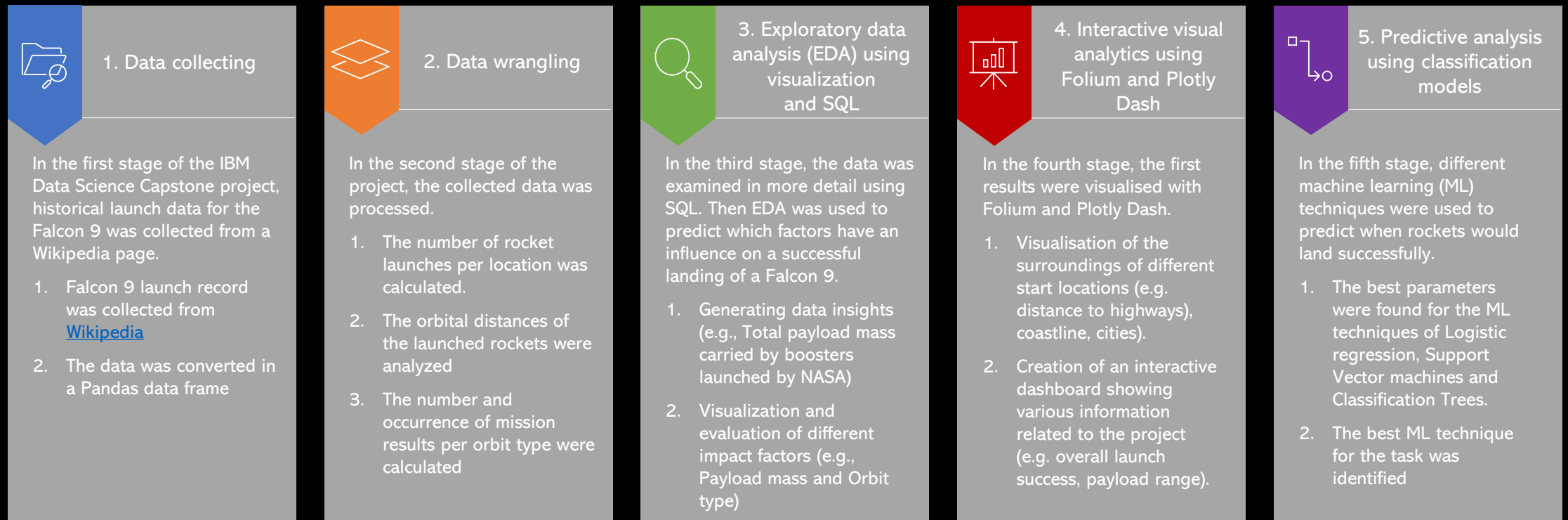
**Predictive analysis of the data** (e.g., the identification of the best data model to predict a successful landing of a launch modul)

# Methodology

# Methodology



Five steps were taken to prepare, analyse and evaluate the data of the IBM Data Science Capstone project.





# Data collection – SpaceX API

- In the first stage of the Data collection stage two objectives were to achieve:
  1. Request to the SpaceX API
  2. Clean the requested data
- Those objectives were achieved through 3 tasks
  1. The SpaceX launch data had to be requested and parsed
  2. The dataframe had to be filtered to only include Falcon 9 launches
  3. The first missing values had to be removed

[GitHub URL](#)

## Task 1: The SpaceX launch data had to be requested and parsed

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

```
In [6]: spacex_url="https://api.spacexdata.com/v4/launches/past"

In [7]: response = requests.get(spacex_url)
```

Now we decode the response content as a Json using `.json()` and turn it into a Pandas dataframe using `.json_normalize()`

```
In [11]: # Use json_normalize method to convert the json result into a dataframe
response = requests.get(static_json_url).json()
data = pd.json_normalize(response)
```

## Task 2: The dataframe had to be filtered to only include Falcon 9 launches

Finally we will remove the Falcon 1 launches keeping only the Falcon 9 launches. Filter the data dataframe using

```
In [23]: # Hint data['BoosterVersion']!= 'Falcon 1'
data_falcon9 = df.loc[df['BoosterVersion']!= 'Falcon 1']
```

Now that we have removed some values we should reset the FlightNumber column

```
In [27]: data_falcon9.loc[:, 'FlightNumber'] = list(range(1, data_falcon9.shape[0]+1))
data_falcon9
print(data_falcon9)
```

## Task 3: The first missing values had to be removed

### Task 3: Dealing with Missing Values

Calculate below the mean for the `PayloadMass` using the `.mean()`. Then use the mean and the

```
In [1]: # Calculate the mean value of PayloadMass column
mean = data_falcon9['PayloadMass'].mean()
# Replace the np.nan values with its mean value
data_falcon9['PayloadMass'] = data_falcon9['PayloadMass'].fillna(mean)
```



# Data collection - Scrapping

- In the second stage of the Data collection stage two objectives were to achieve:
  1. Extract a Falcon 9 launch records HTML table from Wikipedia
  2. Parse the table and convert it into a Pandas data frame
- Those objectives were achieved through 2 tasks
  1. The Falcon 9 launch record had to be extract from Wikipedia
  2. A data frame had to be created by parsing the launch HTML tables

[GitHub URL](#)

## Task 1: The Falcon 9 launch record had to be extract from Wikipedia

First, let's perform an HTTP GET method to request the Falcon9 Launch HTML page, as an HTTP response.

```
In [71]: # use requests.get() method with the provided static_url
# assign the response to a object

Wpage = requests.get(static_url)
print(Wpage.content)
Wpage.status_code
```

## Task 2: A data frame had to be created by parsing the launch HTML tables

```
In [23]: extracted_row = 0
#Extract each table
for table_number,table in enumerate(soup.find_all('table',"wikitable plainrowheaders collapsible")):
    # get table row
    for rows in table.find_all("tr"):
        #check to see if first table heading is as number corresponding to launch a number
        if rows.th:
            if rows.th.string:
                flight_number=rows.th.string.strip()
                flag=flight_number.isdigit()
            else:
                flag=False
            #get table element
            row=rows.find_all('td')
            #if it is number save cells in a dictionary
            if flag:
                extracted_row += 1
                # Flight Number value
                launch_dict["Flight No."].append(flight_number)

                datatimelist=date_time(row[0])
                # Date value
                date = datatimelist[0].strip(',')
                launch_dict["Date"].append(date)

                # Time value
                time = datatimelist[1]
                launch_dict["Time"].append(time)
```

Etc.



# Data wrangling



**Task 1:** For each location the number of launches were calculated

```
In [5]: # Apply value_counts() on column LaunchSite
df["LaunchSite"].value_counts()
```

**Task 2:** The different orbit types were calculated in number and occurrence

```
In [7]: # Apply value_counts on Orbit column
df["Orbit"].value_counts("Orbit")
```

**Task 3:** The number and occurrence of mission results per orbit type were calculated

```
In [8]: # landing_outcomes = values on Outcome column
landing_outcomes = df["Outcome"].value_counts()
```

**Task 4:** A landing outcome label from the outcome column was created

```
In [12]: # landing_class = 0 if bad_outcome
# landing_class = 1 otherwise
landing_class = []
for key,value in df["Outcome"].items():
    if value in bad_outcomes:
        landing_class.append(0)
    else:
        landing_class.append(1)
```

- In the Data wrangling stage two objectives were to achieve:
  1. A first Exploratory Data Analysis of the data
  2. Determine Training Labels
- Those objectives were achieved through 4 tasks
  1. For each location where Space X launches rockets the number of launches were calculated
  2. The different orbit types were calculated in number and occurrence
  3. The number and occurrence of mission results per orbit type were calculated
  4. A landing outcome label from the outcome column was created

[GitHub URL](#)

# EDA with Data Visualization



- In the EDA with Data Visualization stage two objectives were to achieve:

1. Expanded Exploratory Data Analysis of the data
2. Preparing Data Feature Engineering

- Those objectives were achieved through 3 Tasks

1. Visualization and evaluation of different impact factors that can have an impact on the successfully landing of Space X Falcon9 rocket:
  - Flight number, Payload each with Launch location
  - Success rate, Flight number, Payload each with Orbit type
2. Visualization of the launch success yearly trend
3. Creation of the dummy variables to categorical columns

[GitHub URL](#)

**Task 1:** Visualization and evaluation of different impact factors that can have an impact on the successfully landing of Space X Falcon9 rocket

```
In [11]: # Plot a scatter point chart with x axis to be Flight Number and y axis to be the
lass 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()
```

Example

**Task 2:** Visualization of the launch success yearly trend

```
In [24]: # Plot a line chart with x axis to be the extracted year
year = pd.DatetimeIndex(df['Date']).year
year = np.array(list(year))
successratelist = []
successrate = 0.00
records = 1
data = 0
for x in df['Class']:
    data = x + data
    successrate = data/records
    successratelist.append(successrate)
    records= records +1

successratelist = np.array(successratelist)
d = {'successrate':successratelist,'year':year}
sns.set(rc={'figure.figsize':(11.7,8.27)})
sns.lineplot(data=d, x="year", y="successrate" )

plt.xlabel("Year",fontsize=20)
plt.title('Space X Rocket Success Rates')
plt.ylabel("Success Rate",fontsize=20)
plt.show()
```

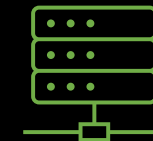
**Task 3:** Creation of the dummy variables to categorical columns

```
In [26]: # HINT: Use get_dummies() function on the categorical columns
features_one_hot = pd.get_dummies(features, columns=['Orbit', 'LaunchSite', 'LandingPad', 'Serial'])
features_one_hot.head()
```

# EDA with SQL



- In the EDA with SQL stage two objectives were to achieve:
  1. Execute SQL queries to answer assignment questions
  2. Understand the SpaceX DataSet
- Those objectives were achieved through 10 Tasks
  1. Identification of the unique launch site names of Space X
  2. Identification of 5 records where the launch sites begin with the string "CCA"
  3. Identification of the total payload mass carried by boosters launched by NASA
  4. Identification of the average payload mass carried by booster version F9
  5. Listing of the data when the first successful landing outcome was achieved
  6. Listing the names of the boosters which have success in drone ship
  7. Listing the total number of successful and failure mission outcomes
  8. Listing the names of the booster\_versions which have carried the maximum payload mass
  9. Listing the failed landing\_outcomes in drone ship, their booster versions, and launch site names for in year 2015
  10. Ranking the count of landing outcomes between 10/2010 and 03/2017
- Code of the tasks on the next slide



[GitHub URL](#)

# EDA with SQL



## Task 1: Identification of the unique launch site names of Space X

```
In [8]: db.GetRecordsOfColumn('select DISTINCT Launch_Site from tblSpaceX','Launch_Site')
```

## Task 2: Identification of 5 records where the launch sites begin with "CCA"

```
In [1]: import pyodbc
import pandas as pd
import numpy as np
conn = pyodbc.connect('Driver={SQL Server};'
                      'Server=localhost;'
                      'Database=SpaceX;'
                      'User ID=admin;Password=admin;')

cursor = conn.cursor()

cursor.execute("select TOP 5 * from tblSpaceX WHERE Launch_Site LIKE 'CCA%')
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

## Task 3: Identification of the total payload mass (PM) carried by boosters launched by NASA

```
In [57]: TFM = db.GetRecordsOfColumn("select SUM(PAYLOAD_MASS_KG_) TotalPayloadMass from tblSpaceX where Customer = 'NASA (CRS)'", 'TotalPayloadMass')
ndf = pd.DataFrame(TFM)
ndf.columns = ['Total Payload Mass']
ndf
```

## Task 4: Identification of the average payload mass carried by booster version F9

```
In [62]: AFM = db.GetRecordsOfColumn("select AVG(PAYLOAD_MASS_KG_) AveragePayloadMass from tblSpaceX where Booster_Version = 'F9 v1.1'", 'AveragePayloadMass')
ndf = pd.DataFrame(AFM)
ndf.columns = ['Average Payload Mass']
ndf
```

## Task 5: Listing of the data when the first successful landing outcome was achieved

```
In [64]: SLO = db.GetRecordsOfColumn("select MIN(Date) SLO from tblSpaceX where Landing_Outcome = 'Success (drone ship)'", 'SLO')
ndf = pd.DataFrame(SLO)
ndf.columns = ['Date which first Successful landing outcome in drone ship was achiaved.']
ndf
```

## Task 6: Listing the names of the boosters which have success in drone ship

```
In [69]: SLO = db.GetRecordsOfColumn("select Booster_Version from tblSpaceX where Landing_Outcome = 'Success (ground pad)' AND Payload_MASS_KG_ > 4000 AND Payload_MASS_KG_ < 6000", 'Booster_Version')
ndf = pd.DataFrame(SLO)
ndf.columns = ['Date which first Successful landing outcome in drone ship was achiaved.']
ndf
```

## Task 7: Listing the total number of successful and failure mission outcomes

```
In [84]: conn = pyodbc.connect('Driver={SQL Server};'
                              'Server=localhost;'
                              'Database=SpaceX;'
                              'User ID=admin;Password=admin;')

cursor = conn.cursor()

cursor.execute("SELECT COUNT(Mission_Outcome) from tblSpaceX where Mission_Outcome LIKE '%Success%' as Successful_Mission_Outcome, (SELECT COUNT(Mission_Outcome) from tblSpaceX where Mission_Outcome LIKE '%Failure%') as Failure_Mission_Outcome")
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

## Task 8: Listing the names of the booster which have carried the max. PM

```
In [94]: conn = pyodbc.connect('Driver={SQL Server};'
                              'Server=localhost;'
                              'Database=SpaceX;'
                              'User ID=admin;Password=admin;')

cursor = conn.cursor()

cursor.execute("SELECT DISTINCT Booster_Version, MAX(PAYLOAD_MASS_KG_) AS [Maximum Payload Mass] FROM tblSpaceX GROUP BY Booster_Version ORDER BY [Maximum Payload Mass] DESC")
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

## Task 9: Listing the failed landing\_outcomes in drone ship, their booster versions, and launch site names for in year 2015

```
In [86]: conn = pyodbc.connect('Driver={SQL Server};'
                              'Server=localhost;'
                              'Database=SpaceX;'
                              'User ID=admin;Password=admin;')

cursor = conn.cursor()

cursor.execute("SELECT DatePart(month, DateAdd(month, MONTHS(CONVERT(date, Date, 101) - 9) - 1) as Month, Booster_Version, Launch_Site, Landing_Outcome FROM tblSpaceX WHERE (Landing_Outcome LIKE '%Success%') AND YEAR(CONVERT(date, Date, 101)) = '2015'")
columns = [column[0] for column in cursor.description]
results = []
for row in cursor.fetchall():
    results.append(dict(zip(columns, row)))

df = pd.DataFrame.from_dict(results)
df
```

## Task 10: Ranking the landing outcomes between 10/2010 and 03/2017

```
In [90]: sl = db.GetRecordsOfColumn("SELECT COUNT(Landing_Outcome) AS sl FROM dbo.tblSpaceX WHERE (Landing_Outcome LIKE '%Success%') AND (Date > '04-04-2010') AND (Date < '20-03-2017')", 'sl')
ndf = pd.DataFrame(sl)
ndf.columns = ['Successful Landing Outcomes Between 2010-04-04 and 2017-03-20']
ndf
```





# Interactive visual analytics – Map with Folium

## Task 1: The launch location were marked with `folium.circle()`

```
In [24]: # Initial the map
site_map = folium.Map(location=nasa_coordinate, zoom_start=6)
# For each launch site, add a Circle object based on its coordinate (lat, long) values. In addition, add launch site name as a popup label
for index, site in launch_sites_df.iterrows():
    circle = folium.Circle([site['lat'], site['long']], color='#d35400', radius=50, fill=True).add_child(folium.Popup(site['Launch Site']))
    marker = folium.Marker(
        [site['lat'], site['long']],
        # Create an icon as a text label
        icon=DivIcon(
            icon_size=(20,20),
            icon_anchor=(0,0),
            html=f"<div style='font-size: 12px; color: #d35400;'><b>{site['Launch Site']}</b></div>"
        )
    )
    site_map.add_child(circle)
    site_map.add_child(marker)

site_map
```

## Task 2: The success/failed launches were marked with `MarkerCluster()`

```
In [30]: # Add marker cluster to current site_map
site_map.add_child(marker_cluster)

for index, record in spaceX_df.iterrows():
    marker = folium.Marker([record['lat'], record['long']],
        icon=folium.Icon(color='white', icon_color=record['marker_color']))
    marker_cluster.add_child(marker)

site_map
```

## Task 3: The distances between the launch locations and some proximities were calculated with `folium.PolyLine()`

```
In [33]: # find coordinate of the closet coastline
# e.g.: Lat: 28.56367 Lon: -80.57163
# distance_coastline = calculate_distance(launch_site_lat, launch_site_lon, coastline_lat, coastline_lon)

coordinates = [
    [28.56342, -80.57674],
    [28.56342, -80.56756]]

lines=folium.PolyLine(locations=coordinates, weight=1)
site_map.add_child(lines)
distance = calculate_distance(coordinates[0][0], coordinates[0][1], coordinates[1][0], coordinates[1][1])
distance_circle = folium.Marker(
    [28.56342, -80.56794],
    icon=DivIcon(
        icon_size=(20,20),
        icon_anchor=(0,0),
        html=f"<div style='font-size: 12px; color: #d35400;'><b>{distance:.2f} KM</b></div>"
    )
)
site_map.add_child(distance_circle)

site_map
```

- In the fourth stage of the project the first visualizations were created. For that a interactive map was created with Folium. The creation had 3 main objects.

1. Marking all launch locations of Space X on the map
2. Marking all success/failed launches for each location on the map
3. Calculating distances between the launch locations and some proximities e.g., Highways, cities or the nearest coastline.

- Those objectives were achieved through 3 tasks

1. The launch location were marked with `folium.circle()`
2. The success/failed launches were marked with `MarkerCluster()`
3. The distances between the launch locations and some proximities were calculated with `folium.PolyLine()`

[GitHub URL](#)



# Interactive visual analytics – Dashboard (Plotly)

## Task 1: The creation of a launch location drop-down

```
dcc Dropdown(id='site_dropdown',options=lsites,placeholder='Select a Launch Site here', searchable = True , value = 'All Sites'),
html.Br(),
```

## Task 2: The creation of a callback function to render a success-pie-chart

```
@app.callback(
    Output(component_id='success-pie-chart',component_property='figure'),
    [Input(component_id='site_dropdown',component_property='value')]
)
def update_graph(site_dropdown):
    if (site_dropdown == 'All Sites'):
        df = spacex_df[spacex_df['class'] == 1]
        fig = px.pie(df, names = 'Launch Site',hole=.3,title = 'Total Success Launches By all sites')
    else:
        df = spacex_df.loc[spacex_df['Launch Site'] == site_dropdown]
        fig = px.pie(df, names = 'class',hole=.3,title = 'Total Success Launches for site '+site_dropdown)
    return fig
```

## Task 3: The creation of a ranger slider to select different payloads

```
dcc.RangeSlider(
    id='payload_slider',
    min=0,
    max=10000,
    step=1000,
    marks = {
        0: '0 kg',
        1000: '1000 kg',
        2000: '2000 kg',
        3000: '3000 kg',
        4000: '4000 kg',
        5000: '5000 kg',
        6000: '6000 kg',
        7000: '7000 kg',
        8000: '8000 kg',
        9000: '9000 kg',
        10000: '10000 kg'
    },
    value=[min_payload,max_payload]
```

## Task 4: The creation of a callback function to render a success-payload-scatter-chart

```
@app.callback(
    Output(component_id='success-payload-scatter-chart',component_property='figure'),
    [Input(component_id='site_dropdown',component_property='value'),Input(component_id='payload_slider', component_property='value')]
)
def update_scattergraph(site_dropdown,payload_slider):
    if site_dropdown == 'All Sites':
        low, high = payload_slider
        df = spacex_df
        mask = (df['Payload Mass (kg)'] > low) & (df['Payload Mass (kg)'] < high)
        fig = px.scatter(
            df[mask], x='Payload Mass (kg)', y='class'
```

- In the fourth stage of the project the first visualizations were created. After the interactive map additionally, a Dashboard was created with Plotly Dash. The creation had 1 objective

1. Creation of an interactive Dashboard that showed the total success launches by location and the correlation between different payloads and the success for all locations.

- Those objectives were achieved through 4 tasks

1. The creation of a launch location drop-down
2. The creation of a callback function to render a success-pie-chart
3. The creation of a ranger slider to select different payloads
4. The creation of a callback function to render a success-payload-scatter-chart

[GitHub URL](#)

# Predictive analysis using classification models

- In the final stage of the project, a data model was created. The data model is used to determine whether the launch capsule of a Space-X rocket can successfully land back on Earth. The development had 1 main objective
  1. The development of a data model that best predicts a successful landing.
- Those objectives were achieved through 4 Tasks
  1. Preparation of the dataset (standardization, Splitting in train and test set)
  2. Development of different data models (Logistic regression, support vector machines and Decision tree)
  3. Identification of the best data model

[GitHub URL](#)

## Task 1: Preparation of the dataset (standardization, Splitting in train and test set)

```
In [9]: X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2, random_state=2)
print('Train set:', X_train.shape, Y_train.shape)
print('Test set:', X_test.shape, Y_test.shape)
```

## Task 2: Development of different data models (Logistic regression, support vector machines and Decision tree)

```
In [11]: parameters = {"C": [0.01, 0.1, 1], 'penalty': ['l2'], 'solver': ['lbfgs']} # 11 lasso 12 ridge
lr=LogisticRegression()
gscv = GridSearchCV(lr, parameters, scoring='accuracy', cv=10)
logreg_cv = gscv.fit(X_train, Y_train)
```

example

## Task 3: Identification of the best data model

```
In [61]: algorithms = {'KNN':KNN_cv.best_score_, 'Tree':tree_cv.best_score_, 'LogisticRegression':logreg_cv.best_score_}
bestalgorithm = max(algorithms, key=algorithms.get)
print('Best Algorithm is',bestalgorithm,'with a score of',algorithms[bestalgorithm])
if bestalgorithm == 'Tree':
    print('Best Params is :',tree_cv.best_params_)
if bestalgorithm == 'KNN':
    print('Best Params is :',KNN_cv.best_params_)
if bestalgorithm == 'LogisticRegression':
    print('Best Params is :',logreg_cv.best_params_)
```

# Results





The background is a complex, abstract composition. It features a dark blue base color. Overlaid on this are numerous diagonal streaks and bands of lighter blue and vibrant red. These streaks vary in thickness and intensity, creating a sense of motion and depth. A faint, semi-transparent grid pattern is visible across the entire image, adding a technical or digital feel. The overall effect is one of high-tech, data-driven aesthetics.

Insights drawn  
from EDA



# Correlation between potential impact factors



Figure 1. shows that the more flights are carried out at a location, the higher the probability of a successful landing of the launch module.

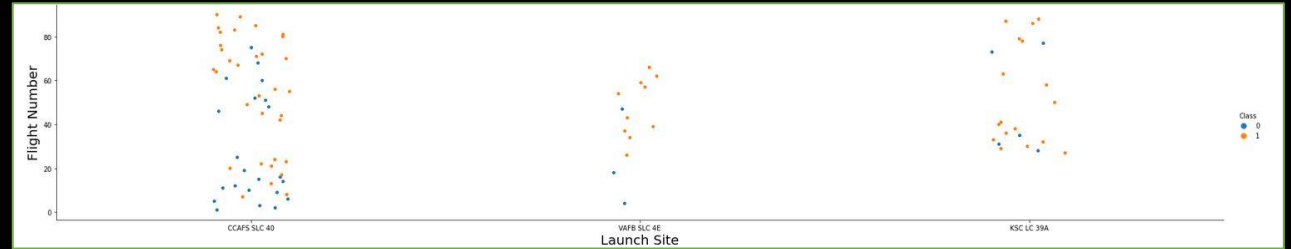


Fig. 1: Flight Number VS. Launch location

From Figure 2. it can be deduced that at the CCAFS SLC 40 and VAFB SLC 4E locations there is a correlation between the severity of the missiles and a successful landing of the launch module. For KSC LC 39A this can not clearly be said.

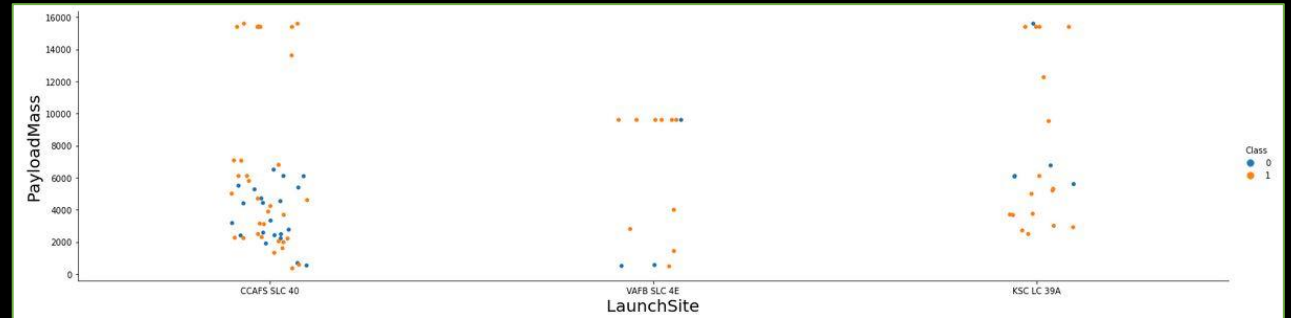


Fig. 2: Payload Mass VS. Launch location

# Correlation between potential impact factors



Figure 3. shows that when a rocket is brought into the orbit range LEO (1000 km from earth), the landing has become very reliable after the first two unsuccessful attempts. For the orbit range GTO (35768 km from earth) no such pattern can be detected.

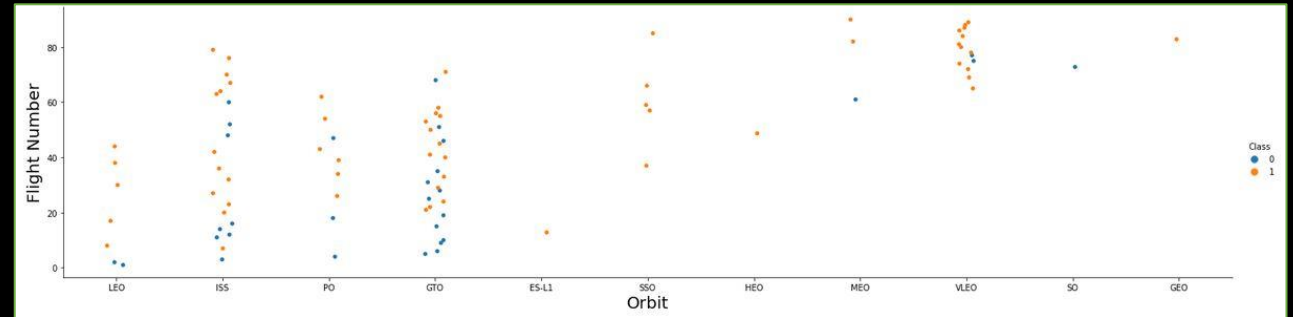


Fig. 3: Flight Number VS. Orbit

From figure 4. it can be deduced that an increased weight can be positively related to a successful landing when rockets are sent into the orbit range LEO or ISS. Again, such a correlation cannot be established when it comes to the orbit range GTO.

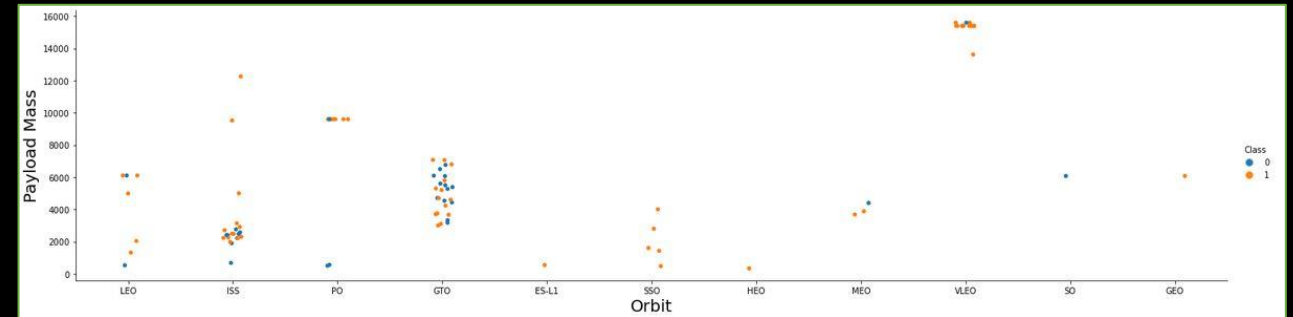


Fig. 4: Payload Mass VS. Orbit

# Correlation between potential impact factors

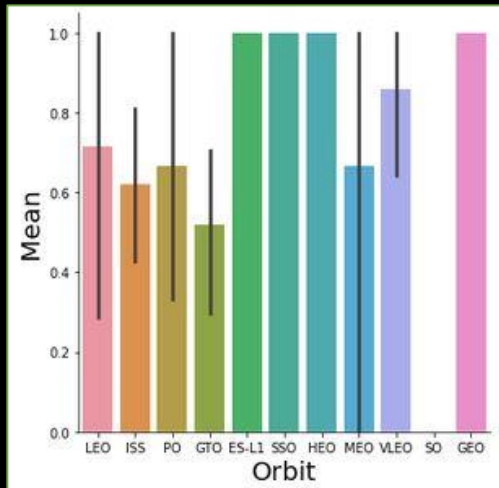


Fig. 5: Success rate VS. Orbit type

Figure 5. shows that the rate of successful landing of the launch module of a Falcon 9 rocket is highest when it is in one of the orbital ranges: ES-L1, SSO, HEO or GEO.

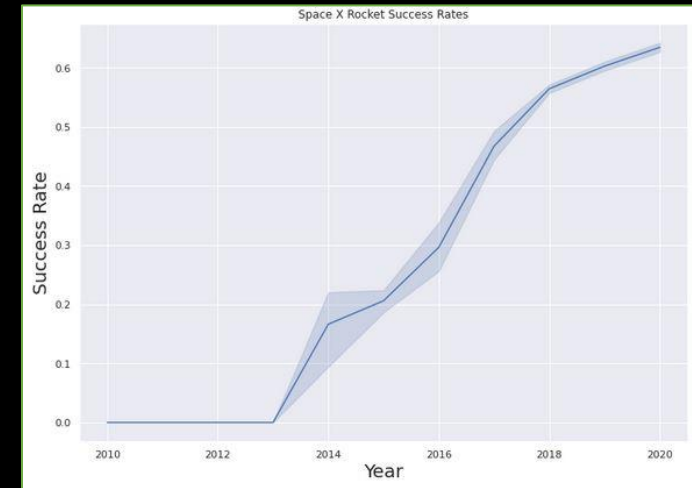


Fig. 6: Launch success yearly trend

Figure 6. shows that there is a correlation between the duration of the Space X project (in years) and the successful landing of the launch modules.



# SQL query results



Task	Result	Explanation
Display the <b>names of the unique launch location</b> in the space mission	CCAFS LC-40; CCAFS SLC-40; CCAFSSLC-40; KSC LC-39A; VAFB SLC-4E	5 unique launch locations were found
Display the <b>total payload mass</b> carried by boosters launched by NASA (CRS)	45596 (Kg.)	The total payload mass was identified
Display <b>average payload mass</b> carried by booster version F9 v1.1	2928 (Kg.)	The average payload was identified
List the <b>date</b> where the succesful landing outcome in drone ship was acheived	06-05-2016	The date was identified
List the <b>names of the boosters</b> which have success in ground pad and have payload mass greater than 4000 but less than 6000	F9 FT B1032.1; F9 B4 B1040.1; F9 B4 B1043.1	The names were identified
List the <b>total number of successful and failure mission outcomes</b>	100 (Successful_Mission_Outcomes)/ 1 (Failure_Mission_Outcomes)	The outcomes were identified
<b>Rank the count of successful landing_outcomes</b> between the date 2010-06-04 and 2017-03-20 in descending order.	34	The rank was identified

# SQL query results



Task	Result (Booster_Version/ Maximum Payload Mass)	Explanation
List the names of the booster_versions which have carried the maximum payload mass.	<ul style="list-style-type: none"><li>• F9 B5 B1048.4 (15600 kg)</li><li>• F9 B5 B1048.5 (15600 kg)</li><li>• F9 B5 B1049.4 (15600 kg)</li><li>• F9 B5 B1049.5 (15600 kg)</li><li>• F9 B5 B1049.7 (15600 kg)</li></ul>	5 unique launch locations were found

A satellite view of Earth from space, showing the curvature of the planet and the glowing city lights of the Eastern United States and parts of Canada at night. The background is the deep blue of the atmosphere and the black of space.

# Launch Sites Proximities Analysis

# Launch locations of Space X (Succ./Unsucc.)

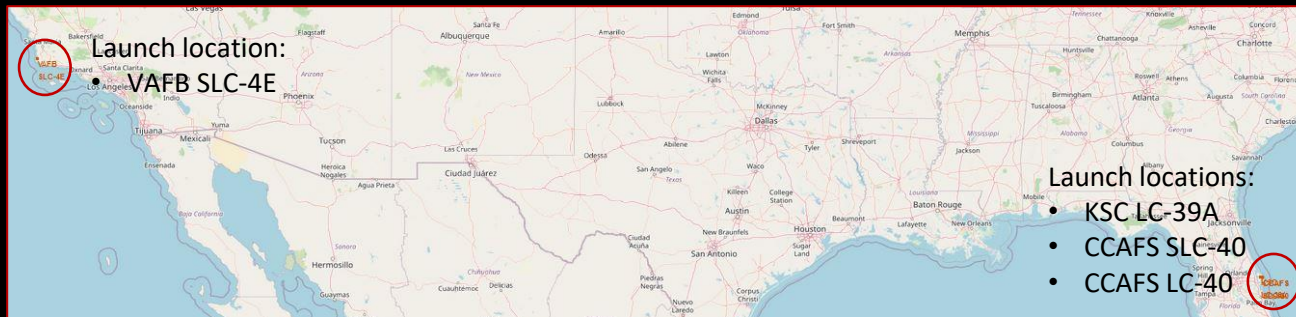


Fig. 7: All launch locations on a global map, generated with Folium

Figure 7. shows all Space X launch locations with a marker in orange

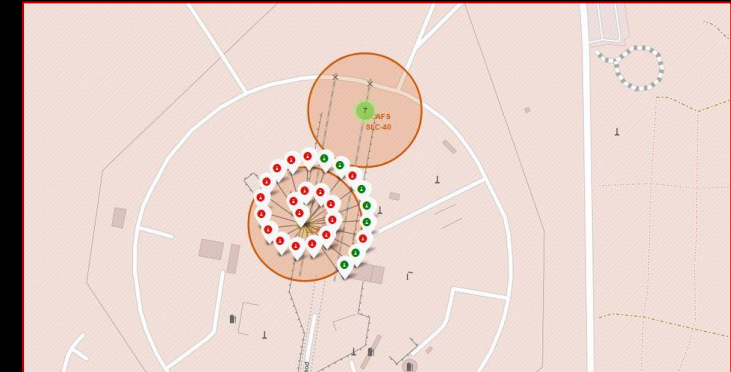


Fig. 8: Two launch locations with color-labelled launch outcomes

Figure 8. shows two launch locations (CCAFS SLC-40 u. CCAFS LC-40). The performed rocket starts of CCAFS LC-40 are shown in green (successful landing of the launch capsule) and red (failed landings of the launch capsule).



# Dist. between CCAFS SLC-40 and sel. locations



Figure 9 shows the distance between the Space X launch location CCAFS SLC-40 and the nearest coastline, highway and railway.

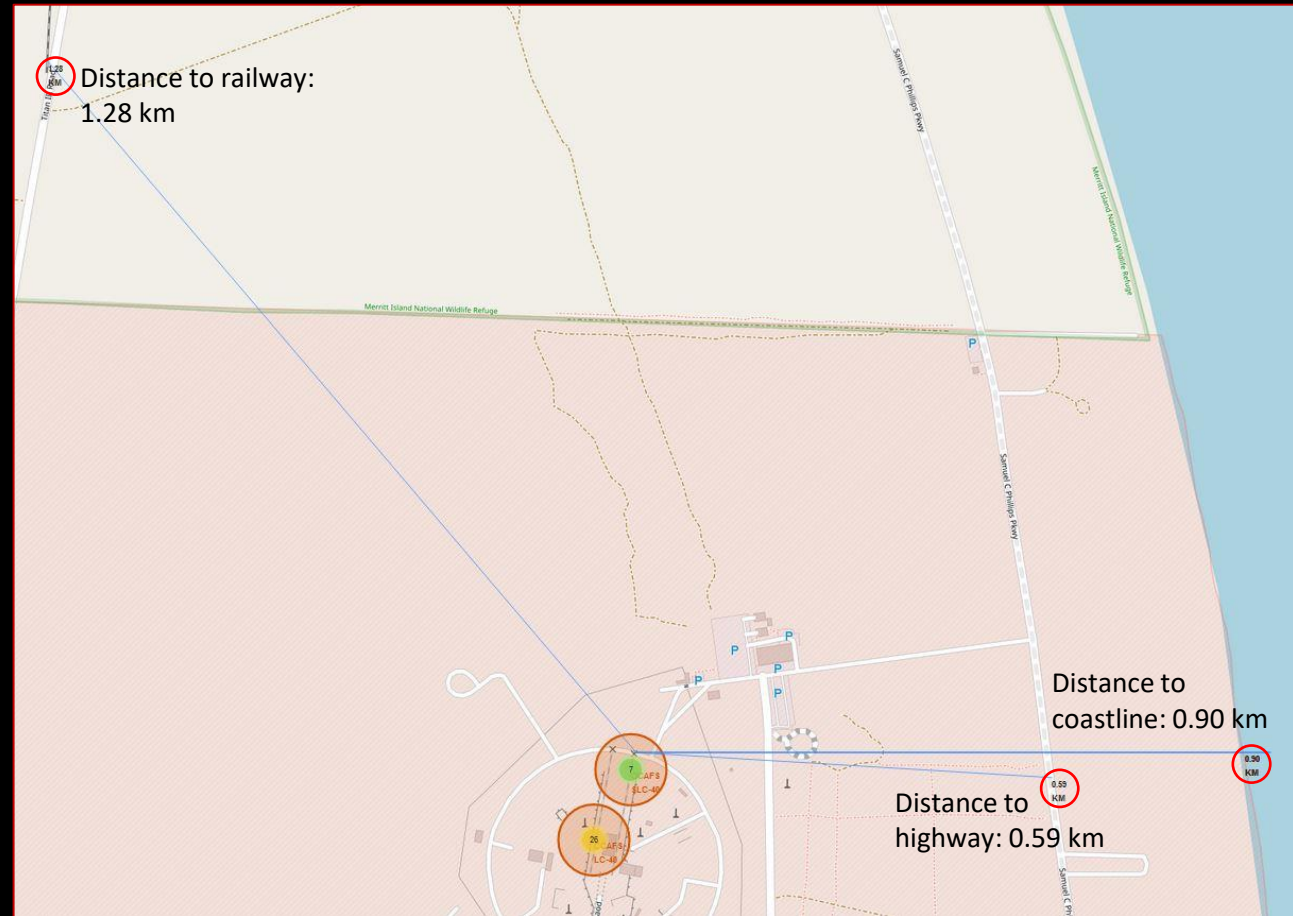


Fig. 9: Distance between CCAFS SLC-40 and selected locations (Coastline, highway, railway)

# Predictive Analysis (Classification)

The background of the slide is an abstract composition. The left half is a solid, vibrant blue. The right half features a series of concentric, curved lines in shades of blue and white, creating a sense of depth and movement, reminiscent of a tunnel or a futuristic architectural design. The lines curve from the bottom towards the top right, where they meet a bright, white, curved surface that looks like the interior of a tunnel with lights in the distance.

# Classification Accuracy

Figure 10 shows the different accuracies of the data models investigated.

The data model based on the decision tree method achieved the best accuracy (0.88) of all models.

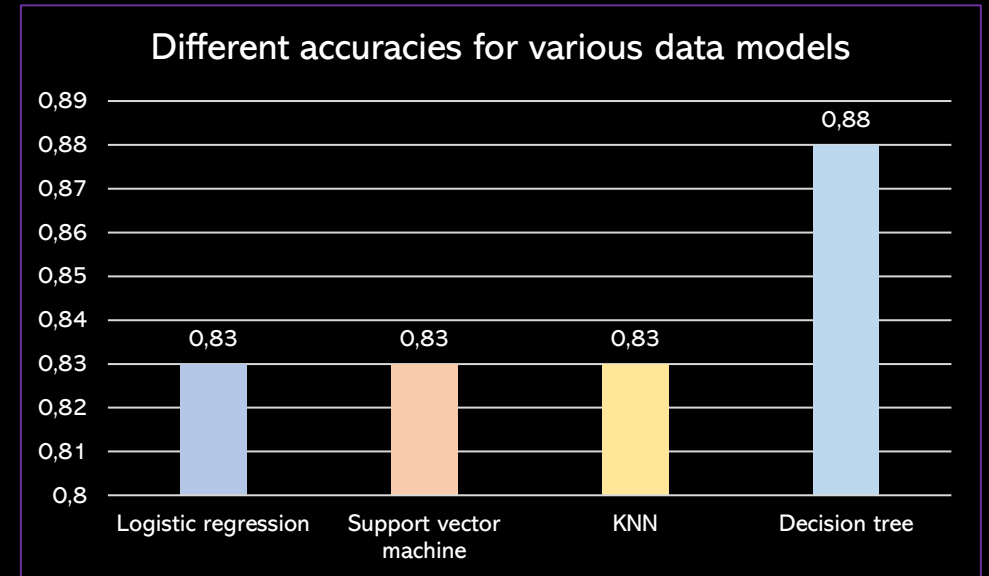


Fig. 10: Different accuracies for various data models

# Confusion Matrix

Figure 10 shows the confusion matrix of the decision tree data model.

A confusion matrix is read along the line. Based on the first line, we can say that the algorithm was able to correctly assign 5 of the 6 start modules that did not land successfully (did not land/did not land). Only one start module was calculated incorrectly.

On the basis of the second line, we can say that the algorithm was able to correctly assign 11 of 12 start modules that landed successfully (land/landed). Only one launch module was calculated incorrectly.

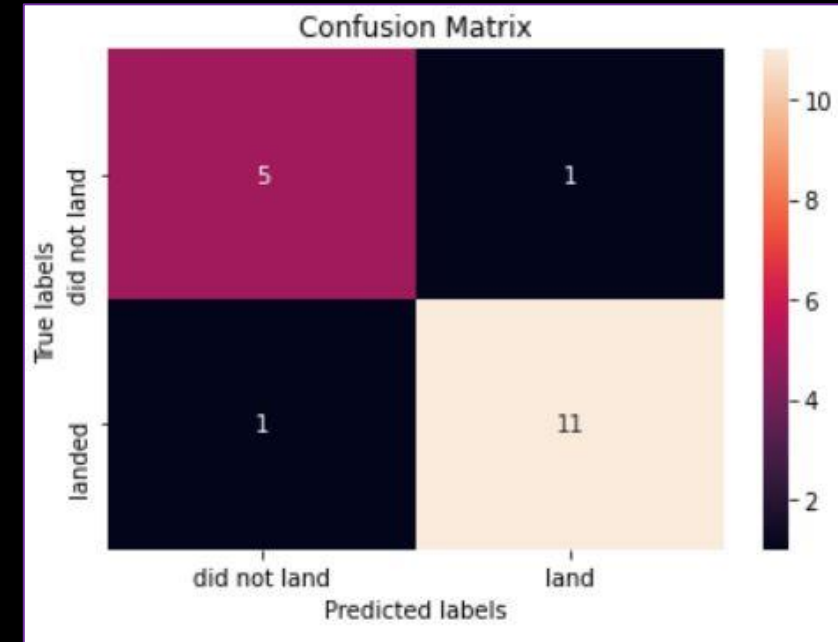


Fig. 10: Different accuracies for various data models



# Conclusions



In summary - What insights were provided in this presentation?

- ? Main question: What factors can be identified that have a positive influence on predicting whether the first launch module of the Space X rocket can be returned safely to Earth?
- ↻ The **more flight tests** have been carried out at a Space X launch location, the **more** likely it is that the **landing of the first launch module will be successful**
- 🎯 Flight tests that target an **orbit range of 1000km from earth (LEO)** are **more successful** when it comes to bringing the first launch module safely back to earth
- 🎯 In addition to the LEO orbit range, **above-average success can be achieved in the ES-L1, SSO, HEO and GEO orbit ranges** when it comes to bringing the first launch module safely back to Earth
- 📈 The **longer the Space X program runs**, the **more successful** the team will be in bringing the first test module safely back to Earth
- 📍 **Space X has 4 Launch locations:** VAFB SLC-4E, KSC LC-39A, CCAFS SLC-40, CCAFS
- 🌳 If you want to predict the success of a safe landing of the first launch module, you should **use a data model based on a decision tree**

# Appendix

---

nothing to add.

Thank you!

