Aircraft Risk Analysis For Business Expansion

- Student name: Lucinda Wanjiru
- Instructor name: Diana Mongina
- Date: 31st March 2025

Business Understanding

Overview

The company is planning to enter into the aviation industry in order to diversify its assets. Specifically, interested in purchasing and operating airplanes for commercial and private enterprises. A risk assessment is essential to minimize liability and maximize safety. This project analyzes data from the National Transportation Safety Board that includes aviation accident data from 1962 to 2023 about civil aviation accidents and selected incidents in the United States and international waters.

A descriptive analysis of the data including accident frequency, severity by aircraft make and model, impact of weather conditions, relationship between flight purpose (e.g., private vs. commercial) and risks.

This analysis can be used by the company to determine which aircraft has the lowest operational, financial, and safety risks.

Business Problem

The company will be able to reduce the risks of safety and financial obligations related with aircraft operation by picking the safest and most reliable models. I aim to:

- Identify low-risk aircraft models.
- Assess the severity and frequency of accidents.
- Assess the elements that contribute to accidents.
- Make recommendations for the best airplanes based on the data analysis results. This will allow the company to chose which airplanes to purchase.

Data Understanding

The aviation accident [dataset (https://www.kaggle.com/datasets/khsamaha/aviation-accident-database-synopses)](https://www.kaggle.com/datasets/khsamaha/aviation-accident-database-synopses) sourced from Kaggle originally obtained from the National Transportation Safety Board contains a detailed record of airplane accidents. Every accident has a unique ID, that is, 'Accident Number' and includes important details such as the date, location, airplane make and model, the severity of injuries, etc. The dataset also captures

In [1]:

```python
# Import standard packages

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

In [2]:

```python
# Load dataset and select the first 5 rows from the dataframe

df = pd.read_csv('data/AviationData.csv', low_memory=False, encoding='la
df.head()
```

Out[2]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Cour
0	20001218X45444	Accident	SEA87LA080	1948-10-24	MOOSE CREEK, ID	Un Sta
1	20001218X45447	Accident	LAX94LA336	1962-07-19	BRIDGEPORT, CA	Un Sta
2	20061025X01555	Accident	NYC07LA005	1974-08-30	Saltville, VA	Un Sta
3	20001218X45448	Accident	LAX96LA321	1977-06-19	EUREKA, CA	Un Sta
4	20041105X01764	Accident	CHI79FA064	1979-08-02	Canton, OH	Un Sta

5 rows × 31 columns

In [3]:

```python
df.tail()
```

Out[3]:

	Event.Id	Investigation.Type	Accident.Number	Event.Date	Location	Cour
88884	20221227106491	Accident	ERA23LA093	2022-12-26	Annapolis, MD	Un Sta
88885	20221227106494	Accident	ERA23LA095	2022-12-26	Hampton, NH	Un Sta
88886	20221227106497	Accident	WPR23LA075	2022-12-26	Payson, AZ	Un Sta
88887	20221227106498	Accident	WPR23LA076	2022-12-26	Morgan, UT	Un Sta
88888	20221230106513	Accident	ERA23LA097	2022-12-29	Athens, GA	Un Sta

5 rows × 31 columns

In [4]:

```python
# .shape shows the dimensionality (in (rows, columns) ) of the DataFrame
df.shape
```

Out[4]: (88889, 31)

In [5]:

```
# Use .info() to get a concise summary of the dataframe

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88889 entries, 0 to 88888
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   Event.Id                88889 non-null  object 
 1   Investigation.Type      88889 non-null  object 
 2   Accident.Number         88889 non-null  object 
 3   Event.Date              88889 non-null  object 
 4   Location                88837 non-null  object 
 5   Country                 88663 non-null  object 
 6   Latitude                34382 non-null  object 
 7   Longitude               34373 non-null  object 
 8   Airport.Code            50249 non-null  object 
 9   Airport.Name            52790 non-null  object 
 10  Injury.Severity         87889 non-null  object 
 11  Aircraft.damage         85695 non-null  object 
 12  Aircraft.Category       32287 non-null  object 
 13  Registration.Number     87572 non-null  object 
 14  Make                    88826 non-null  object 
 15  Model                   88797 non-null  object 
 16  Amateur.Built           88787 non-null  object 
 17  Number.of.Engines       82805 non-null  float64
 18  Engine.Type             81812 non-null  object 
 19  FAR.Description         32023 non-null  object 
 20  Schedule                12582 non-null  object 
 21  Purpose.of.flight       82697 non-null  object 
 22  Air.carrier             16648 non-null  object 
 23  Total.Fatal.Injuries    77488 non-null  float64
 24  Total.Serious.Injuries  76379 non-null  float64
 25  Total.Minor.Injuries    76956 non-null  float64
 26  Total.Uninjured         82977 non-null  float64
 27  Weather.Condition       84397 non-null  object 
 28  Broad.phase.of.flight   61724 non-null  object 
 29  Report.Status           82508 non-null  object 
 30  Publication.Date        75118 non-null  object 
dtypes: float64(5), object(26)
memory usage: 21.0+ MB
```

In [6]:
```python
# Use .columns, to access the column labels of the DataFrame.

df.columns
```

Out[6]:
```
Index(['Event.Id', 'Investigation.Type', 'Accident.Number', 'Event.Dat
e',
       'Location', 'Country', 'Latitude', 'Longitude', 'Airport.Code',
       'Airport.Name', 'Injury.Severity', 'Aircraft.damage',
       'Aircraft.Category', 'Registration.Number', 'Make', 'Model',
       'Amateur.Built', 'Number.of.Engines', 'Engine.Type', 'FAR.Descri
ption',
       'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Inj
uries',
       'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjur
ed',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

In [7]:
```python
# Using .dtypes returns the data types of all columns in the DataFrame

df.dtypes
```

Out[7]:
```
Event.Id                   object
Investigation.Type         object
Accident.Number            object
Event.Date                 object
Location                   object
Country                    object
Latitude                   object
Longitude                  object
Airport.Code               object
Airport.Name               object
Injury.Severity            object
Aircraft.damage            object
Aircraft.Category          object
Registration.Number        object
Make                       object
Model                      object
Amateur.Built              object
Number.of.Engines         float64
Engine.Type                object
FAR.Description            object
Schedule                   object
Purpose.of.flight          object
Air.carrier                object
Total.Fatal.Injuries      float64
Total.Serious.Injuries    float64
Total.Minor.Injuries      float64
Total.Uninjured           float64
Weather.Condition          object
Broad.phase.of.flight      object
Report.Status              object
Publication.Date           object
dtype: object
```

In [8]:
```
df.describe()
```

Out[8]:

	Number.of.Engines	Total.Fatal.Injuries	Total.Serious.Injuries	Total.Minor.Injuries	To
count	82805.000000	77488.000000	76379.000000	76956.000000	
mean	1.146585	0.647855	0.279881	0.357061	
std	0.446510	5.485960	1.544084	2.235625	
min	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	0.000000	0.000000	0.000000	
50%	1.000000	0.000000	0.000000	0.000000	
75%	1.000000	0.000000	0.000000	0.000000	
max	8.000000	349.000000	161.000000	380.000000	

Data Preparation

Data Cleaning

In [9]:
```
# Make column names easier to use

df.columns = df.columns.str.lower().str.replace('.', '_')
df.columns
```

Out[9]:
```
Index(['event_id', 'investigation_type', 'accident_number', 'event_dat
e',
       'location', 'country', 'latitude', 'longitude', 'airport_code',
       'airport_name', 'injury_severity', 'aircraft_damage',
       'aircraft_category', 'registration_number', 'make', 'model',
       'amateur_built', 'number_of_engines', 'engine_type', 'far_descri
ption',
       'schedule', 'purpose_of_flight', 'air_carrier', 'total_fatal_inj
uries',
       'total_serious_injuries', 'total_minor_injuries', 'total_uninjur
ed',
       'weather_condition', 'broad_phase_of_flight', 'report_status',
       'publication_date'],
      dtype='object')
```

In [10]:
```
# Check for duplicate entries based on accident_number column since it i
total_duplicates = df.duplicated('accident_number').sum()

print(f"Total duplicate entries based on 'accident_number': {total_dupli
```

```
Total duplicate entries based on 'accident_number': 26
```

In [11]:
```
# Remove duplicates

df = df.drop_duplicates(subset='accident_number', keep='first')
df.shape
```

Out[11]:
```
(88863, 31)
```

In [12]:
```
# Check for null values

df.isna().sum()
```

Out[12]:
```
event_id                      0
investigation_type            0
accident_number               0
event_date                    0
location                     52
country                     226
latitude                  54500
longitude                 54509
airport_code              38630
airport_name              36089
injury_severity             990
aircraft_damage            3185
aircraft_category         56600
registration_number        1317
make                         63
model                        92
amateur_built               102
number_of_engines          6074
engine_type                7057
far_description           56866
schedule                  76287
purpose_of_flight          6181
air_carrier               72228
total_fatal_injuries      11401
total_serious_injuries    12510
total_minor_injuries      11933
total_uninjured            5912
weather_condition          4481
broad_phase_of_flight     27139
report_status              6361
publication_date          13760
dtype: int64
```

In [13]:
```
# Drop rows with null values in the primary key column; 'accident_number

df = df.dropna(subset = ['accident_number'])
df.shape
```

Out[13]:
```
(88863, 31)
```

In [14]:

```
# Check the percentage of missing values for every column

missing_percentage = df.isna().sum()/len(df)*100
missing_percentage
```

Out[14]:

```
event_id                  0.000000
investigation_type        0.000000
accident_number           0.000000
event_date                0.000000
location                  0.058517
country                   0.254324
latitude                 61.330362
longitude                61.340490
airport_code             43.471411
airport_name             40.611953
injury_severity           1.114074
aircraft_damage           3.584169
aircraft_category        63.693551
registration_number       1.482057
make                      0.070896
model                     0.103530
amateur_built             0.114783
number_of_engines         6.835241
engine_type               7.941438
far_description          63.992888
schedule                 85.847878
purpose_of_flight         6.955651
air_carrier              81.280173
total_fatal_injuries     12.829862
total_serious_injuries   14.077850
total_minor_injuries     13.428536
total_uninjured           6.652938
weather_condition         5.042594
broad_phase_of_flight    30.540270
report_status             7.158210
publication_date         15.484510
dtype: float64
```

In [15]:

```
# Identify columns with missing values above 35%
columns_to_drop = missing_percentage[missing_percentage > 35].index

# Drop the identified columns
df.drop(columns=columns_to_drop, inplace=True)

df.shape
```

Out[15]: (88863, 23)

In [16]:

```
# Drop columns that are irrelevant to my analysis

drop_columns_2 = ['event_id', 'accident_number', 'location', 'country',
                  'publication_date']
df = df.drop(columns = drop_columns_2)
df.shape
```

Out[16]: (88863, 15)

In [17]:
```
df.isna().sum()
```

Out[17]:
```
investigation_type            0
event_date                    0
injury_severity             990
aircraft_damage            3185
make                         63
model                        92
amateur_built               102
number_of_engines          6074
engine_type                7057
purpose_of_flight          6181
total_fatal_injuries      11401
total_serious_injuries    12510
total_minor_injuries      11933
total_uninjured            5912
weather_condition          4481
dtype: int64
```

In [18]:
```
# Drop the rows with missing values

df = df.dropna(subset=['make', 'model', 'amateur_built', 'number_of_engi
                                    'total_serious_injuries', 't
```

In [19]:
```
# Fill missing values of dtype object columns with 'Unknown'

df.fillna('Unknown', inplace=True)
```

In [20]:
```
df.isna().sum()
```

Out[20]:
```
investigation_type        0
event_date                0
injury_severity           0
aircraft_damage           0
make                      0
model                     0
amateur_built             0
number_of_engines         0
engine_type               0
purpose_of_flight         0
total_fatal_injuries      0
total_serious_injuries    0
total_minor_injuries      0
total_uninjured           0
weather_condition         0
dtype: int64
```

In [21]:
```
# Format the columns with entries of type string

columns = ['investigation_type', 'aircraft_damage', 'make', 'amateur_bui
           'weather_condition']

for column in columns:
    df[column] = df[column].str.strip()
    df[column] = df[column].str.lower()
```

In [22]:
```
# Standardize missing data representations

df['injury_severity'] = df['injury_severity'].replace('unavailable', 'un
df['engine_type'] = df['engine_type'].replace('unk', 'unknown')
df['weather_condition'] = df['weather_condition'].replace('unk', 'unknow
df['model'] = df['model'].replace('unk', 'unknown')
```

In [23]:
```
# Create year column for future analysis

df['year'] = [date[:4] for date in df['event_date']]
```

In [24]:
```
# Reset the index of the dataframe

df.reset_index(drop=True, inplace=True)
```

In [25]:
```
df
```

Out[25]:

	investigation_type	event_date	injury_severity	aircraft_damage	make	mo
0	accident	1948-10-24	Fatal(2)	destroyed	stinson	10
1	accident	1962-07-19	Fatal(4)	destroyed	piper	PA
2	accident	1977-06-19	Fatal(2)	destroyed	rockwell	
3	accident	1981-08-01	Fatal(4)	destroyed	cessna	
4	accident	1982-01-01	Non-Fatal	substantial	cessna	
...	...	...	...	...	...	
69574	accident	2022-12-13	Non-Fatal	substantial	piper	PA
69575	accident	2022-12-14	Non-Fatal	substantial	cirrus design corp	SF
69576	accident	2022-12-15	Non-Fatal	substantial	swearingen	SA226
69577	accident	2022-12-16	Minor	substantial	cessna	R17
69578	accident	2022-12-26	Non-Fatal	substantial	american champion aircraft	8GC

69579 rows × 16 columns

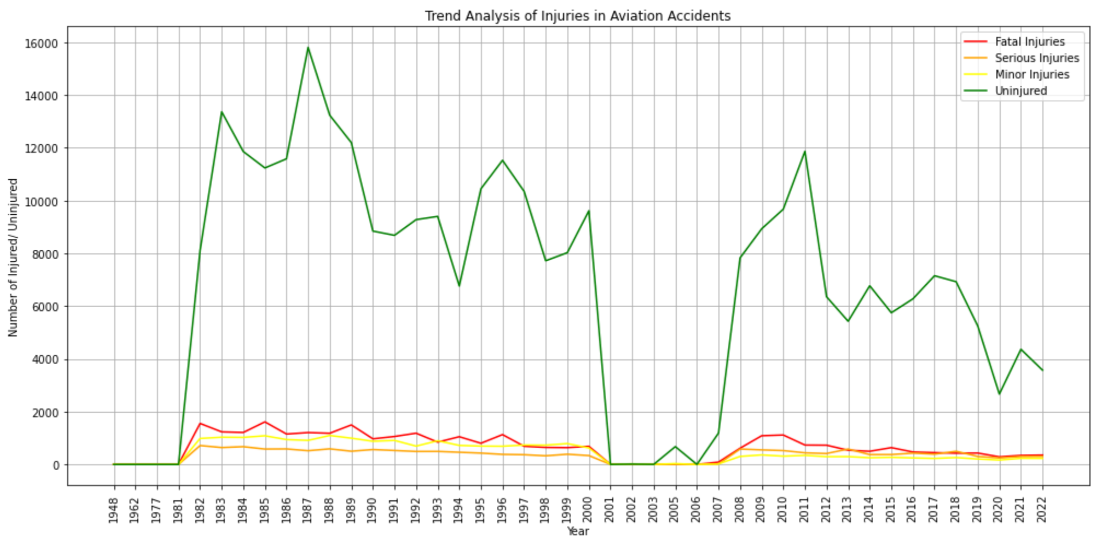
In [26]:
```
# Save cleaned data as excel

df.to_csv('./data/cleaned_aviation_data.csv', index=False)
```

Data Analysis

Trend Analysis of Injuried and Uninjured Passengers in Aviation Accidents Over Time

In [27]:

```
# Group by year
year_grouped = df.groupby('year').agg({
    'total_fatal_injuries': 'sum',
    'total_serious_injuries': 'sum',
    'total_minor_injuries': 'sum',
    'total_uninjured': 'sum'
}).reset_index()

# Plot
plt.figure(figsize=(14, 7))

plt.plot(year_grouped['year'], year_grouped['total_fatal_injuries'], lab
plt.plot(year_grouped['year'], year_grouped['total_serious_injuries'], l
plt.plot(year_grouped['year'], year_grouped['total_minor_injuries'], lab
plt.plot(year_grouped['year'], year_grouped['total_uninjured'], label='U

# Adding labels and title
plt.title('Trend Analysis of Injuries in Aviation Accidents')
plt.xlabel('Year')
plt.ylabel('Number of Injured/ Uninjured')
plt.legend()
plt.xticks(rotation=90)

plt.tight_layout()
plt.grid()
```



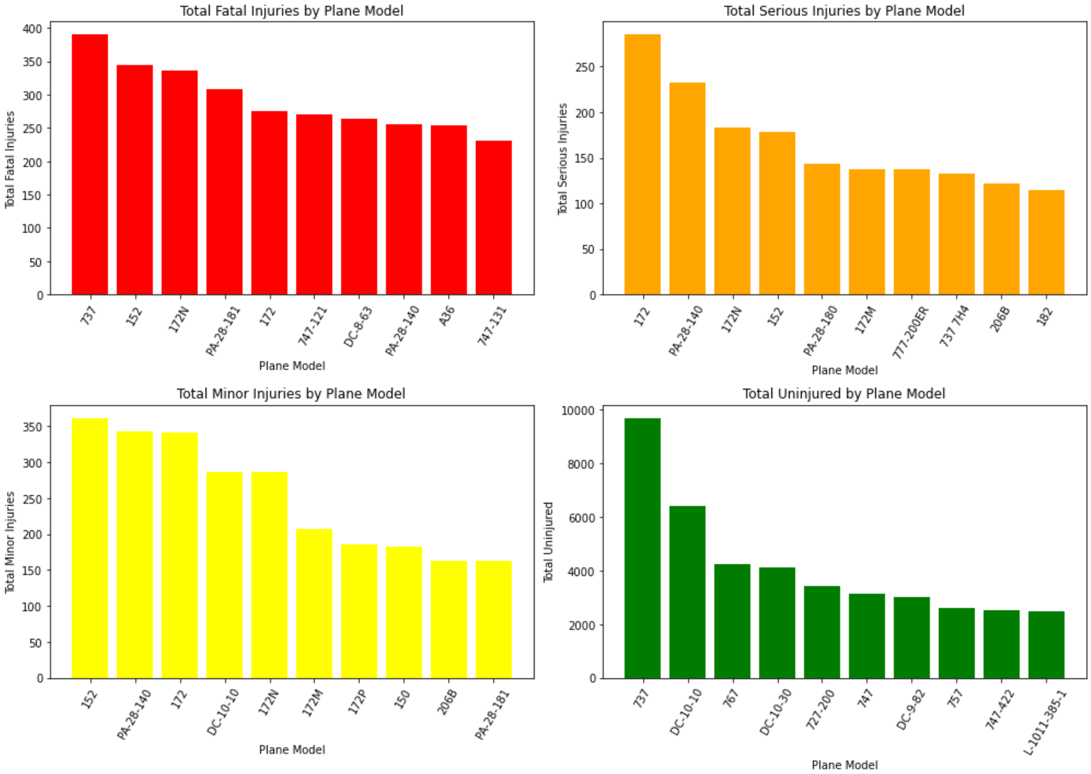
The total uninjured passengers are higher than the total fatal, serious or minor injured passengers from 1948 to 2022. The number of uninjured passengers has fluctuated over time, with a period of notable decrease observed after 1987.

Over the course of the 74-year period, the number of the fatally, seriously and minorly injured passengers remain below 2000 people. The overall number of injuries has declined over time, suggesting that aviation safety standards have improved.

Injured Passengers by the Plane Model

In [28]:

```
# Group by model
model_grouped = df.groupby('model').agg({
    'total_fatal_injuries': 'sum',
    'total_serious_injuries': 'sum',
    'total_minor_injuries': 'sum',
    'total_uninjured': 'sum'
}).reset_index()

# Create subplots
fig, axes = plt.subplots(2, 2, figsize = (14, 10))

# Plot total fatal injuries
fatal_injuries = model_grouped.sort_values(by='total_fatal_injuries', as
axes[0, 0].bar(fatal_injuries['model'], fatal_injuries['total_fatal_inju
axes[0, 0].set_title('Total Fatal Injuries by Plane Model')
axes[0, 0].set_xlabel('Plane Model')
axes[0, 0].set_ylabel('Total Fatal Injuries')
axes[0, 0].tick_params(axis='x', rotation=60)

# Plot total serious injuries
serious_injuries = model_grouped.sort_values(by='total_serious_injuries'
axes[0, 1].bar(serious_injuries['model'], serious_injuries['total_seriou
axes[0, 1].set_title('Total Serious Injuries by Plane Model')
axes[0, 1].set_xlabel('Plane Model')
axes[0, 1].set_ylabel('Total Serious Injuries')
axes[0, 1].tick_params(axis='x', rotation=60)

# Plot total minor injuries
minor_injuries = model_grouped.sort_values(by='total_minor_injuries', as
axes[1, 0].bar(minor_injuries['model'], minor_injuries['total_minor_inju
axes[1, 0].set_title('Total Minor Injuries by Plane Model')
axes[1, 0].set_xlabel('Plane Model')
axes[1, 0].set_ylabel('Total Minor Injuries')
axes[1, 0].tick_params(axis='x', rotation=60)

# Plot total uninjured
uninjured = model_grouped.sort_values(by='total_uninjured', ascending=Fa
axes[1, 1].bar(uninjured['model'], uninjured['total_uninjured'], color='
axes[1, 1].set_title('Total Uninjured by Plane Model')
axes[1, 1].set_xlabel('Plane Model')
axes[1, 1].set_ylabel('Total Uninjured')
axes[1, 1].tick_params(axis='x', rotation=60)

plt.tight_layout()
plt.show()
```

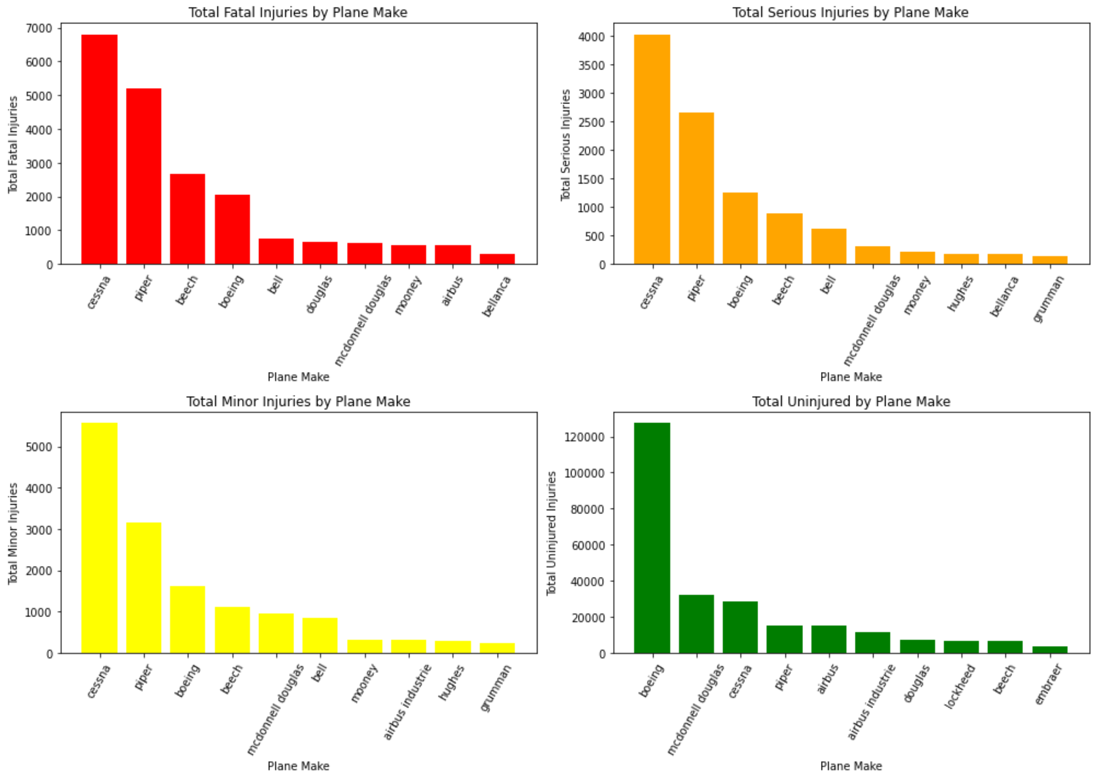


- The aircraft model 737 has the highest number of fatal injuries.
- The aircraft model 172 has the highest number of serious injuries..
- The aircraft model 152 has the highest number of minor injuries.
- Interestingly, the aircraft model 737, which has the highest number of fatal injuries, also has the highest number of uninjured passengers. This might indicate higher passenger volume for this model.

Injured Passengers by Plane Make

In [29]:

```
# Group by make
make_grouped = df.groupby('make').agg({
    'total_fatal_injuries': 'sum',
    'total_serious_injuries': 'sum',
    'total_minor_injuries': 'sum',
    'total_uninjured': 'sum'
}).reset_index()

# Create subplots
fig, axes = plt.subplots(2, 2, figsize = (14, 10))

# Plot total fatal injuries
fatal_injuries = make_grouped.sort_values(by='total_fatal_injuries', asc
axes[0, 0].bar(fatal_injuries['make'], fatal_injuries['total_fatal_injur
axes[0, 0].set_title('Total Fatal Injuries by Plane Make')
axes[0, 0].set_xlabel('Plane Make')
axes[0, 0].set_ylabel('Total Fatal Injuries')
axes[0, 0].tick_params(axis='x', rotation=60)

# Plot total serious injuries
serious_injuries = make_grouped.sort_values(by='total_serious_injuries',
axes[0, 1].bar(serious_injuries['make'], serious_injuries['total_serious
axes[0, 1].set_title('Total Serious Injuries by Plane Make')
axes[0, 1].set_xlabel('Plane Make')
axes[0, 1].set_ylabel('Total Serious Injuries')
axes[0, 1].tick_params(axis='x', rotation=60)

# Plot total minor injuries
minor_injuries = make_grouped.sort_values(by='total_minor_injuries', asc
axes[1, 0].bar(minor_injuries['make'], minor_injuries['total_minor_injur
axes[1, 0].set_title('Total Minor Injuries by Plane Make')
axes[1, 0].set_xlabel('Plane Make')
axes[1, 0].set_ylabel('Total Minor Injuries')
axes[1, 0].tick_params(axis='x', rotation=60)

# Plot total uninjured
uninjured = make_grouped.sort_values(by='total_uninjured', ascending=Fal
axes[1, 1].bar(uninjured['make'], uninjured['total_uninjured'], color='g
axes[1, 1].set_title('Total Uninjured by Plane Make')
axes[1, 1].set_xlabel('Plane Make')
axes[1, 1].set_ylabel('Total Uninjured Injuries')
axes[1, 1].tick_params(axis='x', rotation=60)

plt.tight_layout()
plt.show()
```

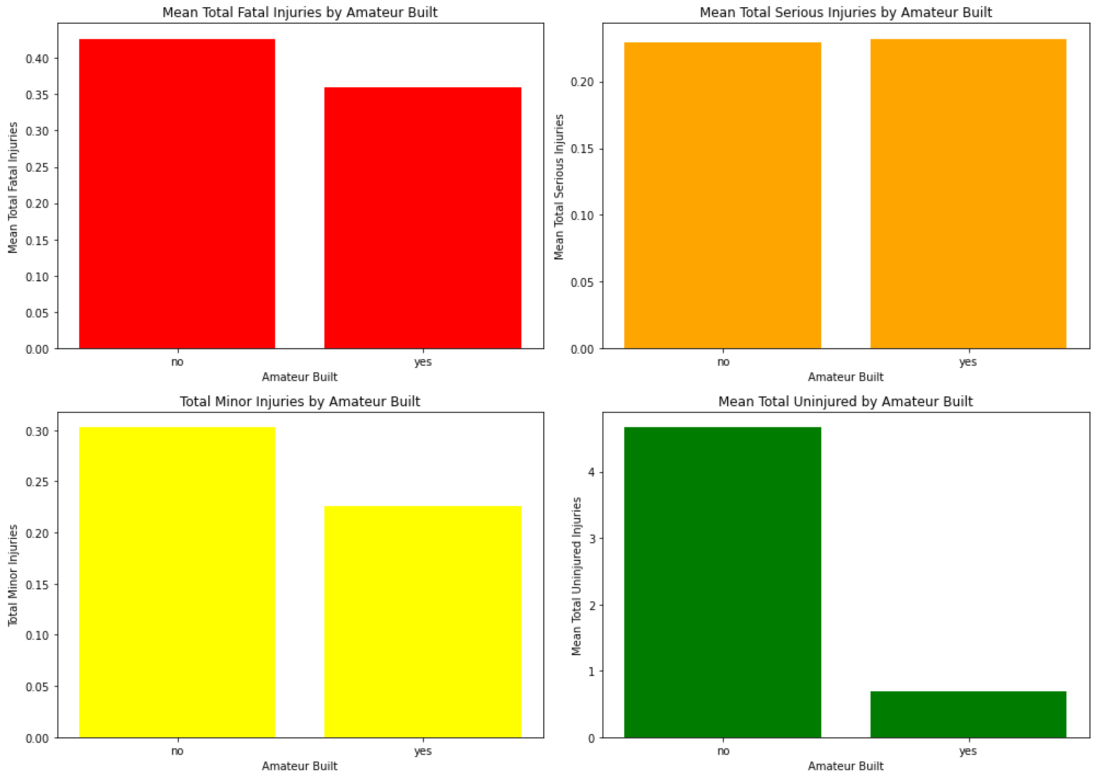


- Cessna aircraft exhibit the highest number of fatal, serious, and minor injuries across all categories.
- Boeing aircraft, while not showing the highest numbers of injuries, stand out with the highest count of uninjured passengers.

Injured Passengers by Amateur Built

In [30]:

```
# Group by amateur built
amateur_built_grouped = df.groupby('amateur_built').agg({
    'total_fatal_injuries': 'mean',
    'total_serious_injuries': 'mean',
    'total_minor_injuries': 'mean',
    'total_uninjured': 'mean'
}).reset_index()

# Create subplots
fig, axes = plt.subplots(2, 2, figsize = (14, 10))

# Plot total fatal injuries
axes[0, 0].bar(amateur_built_grouped['amateur_built'], amateur_built_gro
axes[0, 0].set_title('Mean Total Fatal Injuries by Amateur Built')
axes[0, 0].set_xlabel('Amateur Built')
axes[0, 0].set_ylabel('Mean Total Fatal Injuries')

# Plot total serious injuries
axes[0, 1].bar(amateur_built_grouped['amateur_built'], amateur_built_gro
axes[0, 1].set_title('Mean Total Serious Injuries by Amateur Built')
axes[0, 1].set_xlabel('Amateur Built')
axes[0, 1].set_ylabel('Mean Total Serious Injuries')

# Plot total minor injuries
axes[1, 0].bar(amateur_built_grouped['amateur_built'], amateur_built_gro
axes[1, 0].set_title('Total Minor Injuries by Amateur Built')
axes[1, 0].set_xlabel('Amateur Built')
axes[1, 0].set_ylabel('Total Minor Injuries')

# Plot total amateur_built_grouped
axes[1, 1].bar(amateur_built_grouped['amateur_built'], amateur_built_gro
axes[1, 1].set_title('Mean Total Uninjured by Amateur Built')
axes[1, 1].set_xlabel('Amateur Built')
axes[1, 1].set_ylabel('Mean Total Uninjured Injuries')

plt.tight_layout()
plt.show()
```

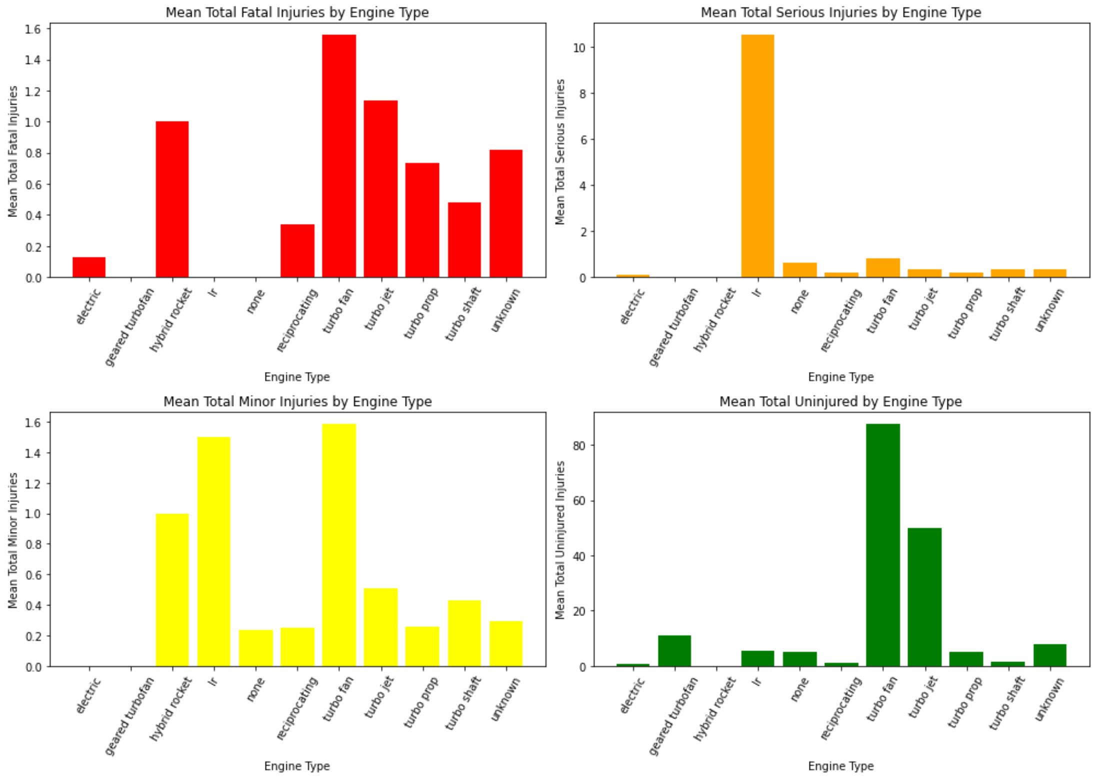


- Non-amateur-built planes have more fatal, minor and uninjured.
- Amateur-built planes have more serious injuries.

From the plot derived above, there is a greater margin between non-amateur-built and amateur-built for uninjured injuries than for fatal injuries. This indicates that non-amateur built planes are safer despite having the most fatal injuries.

Injured Passengers by Type of Engine of the Plane

In [31]:
```
# Group by engine type
engine_type_grouped = df.groupby('engine_type').agg({
    'total_fatal_injuries': 'mean',
    'total_serious_injuries': 'mean',
    'total_minor_injuries': 'mean',
    'total_uninjured': 'mean'
}).reset_index()

# Create subplots
fig, axes = plt.subplots(2, 2, figsize = (14, 10))

# Plot total fatal injuries
axes[0, 0].bar(engine_type_grouped['engine_type'], engine_type_grouped['
axes[0, 0].set_title('Mean Total Fatal Injuries by Engine Type')
axes[0, 0].set_xlabel('Engine Type')
axes[0, 0].set_ylabel('Mean Total Fatal Injuries')
axes[0, 0].tick_params(axis='x', rotation=60)

# Plot total serious injuries
axes[0, 1].bar(engine_type_grouped['engine_type'], engine_type_grouped['
axes[0, 1].set_title('Mean Total Serious Injuries by Engine Type')
axes[0, 1].set_xlabel('Engine Type')
axes[0, 1].set_ylabel('Mean Total Serious Injuries')
axes[0, 1].tick_params(axis='x', rotation=60)

# Plot total minor injuries
axes[1, 0].bar(engine_type_grouped['engine_type'], engine_type_grouped['
axes[1, 0].set_title('Mean Total Minor Injuries by Engine Type')
axes[1, 0].set_xlabel('Engine Type')
axes[1, 0].set_ylabel('Mean Total Minor Injuries')
axes[1, 0].tick_params(axis='x', rotation=60)

# Plot total uninjured
axes[1, 1].bar(engine_type_grouped['engine_type'], engine_type_grouped['
axes[1, 1].set_title('Mean Total Uninjured by Engine Type')
axes[1, 1].set_xlabel('Engine Type')
axes[1, 1].set_ylabel('Mean Total Uninjured Injuries')
axes[1, 1].tick_params(axis='x', rotation=60)

plt.tight_layout()
plt.show()
```



Injured passengers versus the type of plane engine reveals that:

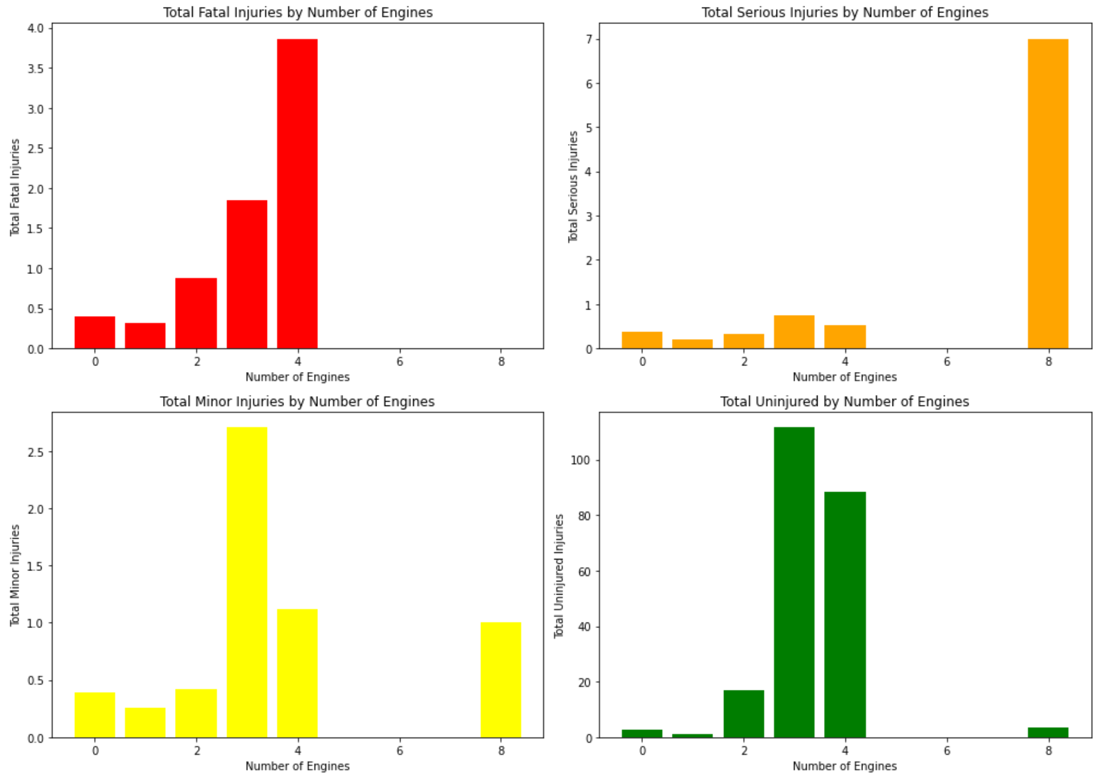
- **Turbo fan engines:** Show the highest mean number of fatal, minor, and uninjured passengers.
- **Reciprocating engines (LR):** Have the highest mean number of serious injuries.
- **Other engine types:** This provides a comparison across various engine types.

Overall: The data suggests a correlation between engine type and the different injury severities.

Injured Passengers by Number of Engines in a Plane

In [32]:

```
# Group by number of engines
number_of_engines_grouped = df.groupby('number_of_engines').agg({
    'total_fatal_injuries': 'mean',
    'total_serious_injuries': 'mean',
    'total_minor_injuries': 'mean',
    'total_uninjured': 'mean'
}).reset_index()

# Create subplots
fig, axes = plt.subplots(2, 2, figsize = (14, 10))

# Plot total fatal injuries
axes[0, 0].bar(number_of_engines_grouped['number_of_engines'], number_of
axes[0, 0].set_title('Total Fatal Injuries by Number of Engines')
axes[0, 0].set_xlabel('Number of Engines')
axes[0, 0].set_ylabel('Total Fatal Injuries')

# Plot total serious injuries
axes[0, 1].bar(number_of_engines_grouped['number_of_engines'], number_of
axes[0, 1].set_title('Total Serious Injuries by Number of Engines')
axes[0, 1].set_xlabel('Number of Engines')
axes[0, 1].set_ylabel('Total Serious Injuries')

# Plot total minor injuries
axes[1, 0].bar(number_of_engines_grouped['number_of_engines'], number_of
axes[1, 0].set_title('Total Minor Injuries by Number of Engines')
axes[1, 0].set_xlabel('Number of Engines')
axes[1, 0].set_ylabel('Total Minor Injuries')

# Plot total uninjured
axes[1, 1].bar(number_of_engines_grouped['number_of_engines'], number_of
axes[1, 1].set_title('Total Uninjured by Number of Engines')
axes[1, 1].set_xlabel('Number of Engines')
axes[1, 1].set_ylabel('Total Uninjured Injuries')

plt.tight_layout()
plt.show()
```

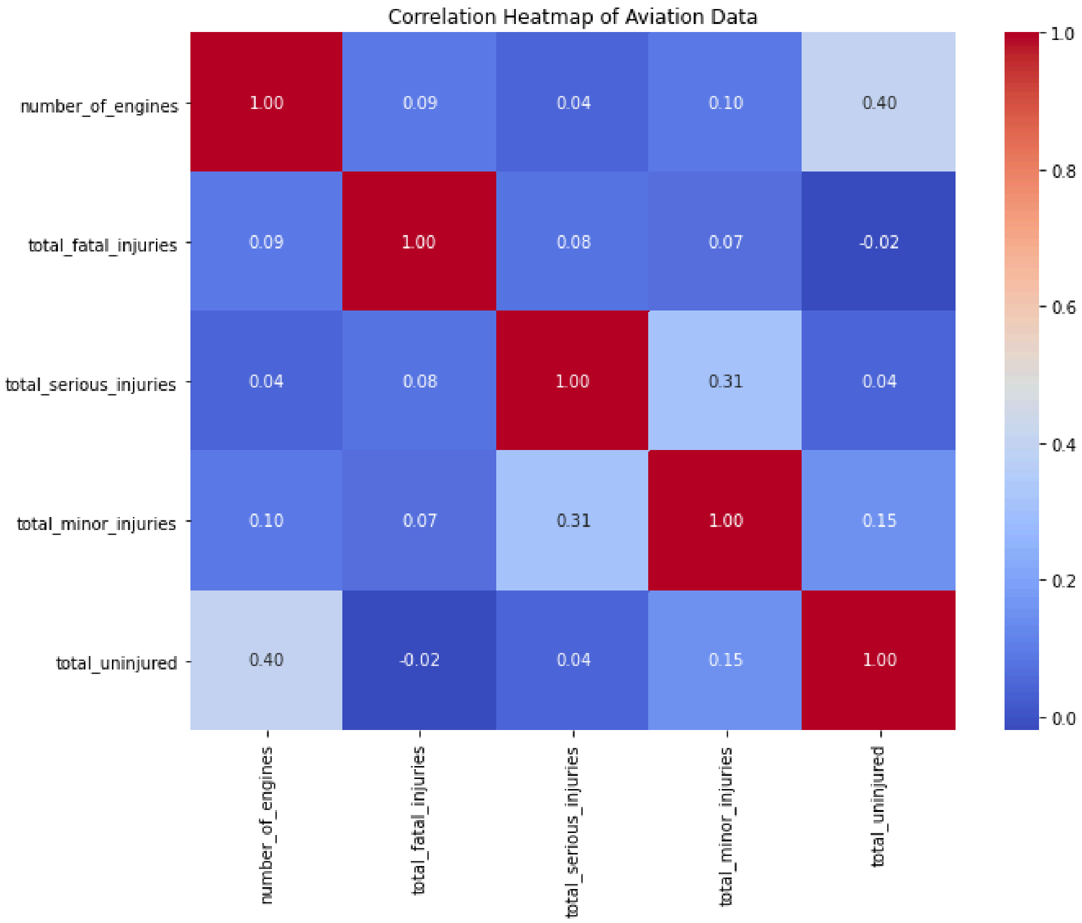


- Planes with 4 engines have the most fatal injuries.
- Planes with 8 engines have the most serious injuries.
- Planes with 3 engines have the most minor injuries and uninjured. The number of engines in a plane does not neccessarily affect the number of injured/ uninjured. Let's find the correlation between the number of engines and the injuries/ uninjured.

Correlation Between Number of Engines and Injuries/ Uninjured

In [33]:

```
# Create subplots
fig, ax = plt.subplots(figsize = (10, 8))

# Calculate correlation matrix
correlation_matrix = df[['number_of_engines', 'total_fatal_injuries', 't

# Plot
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f",
ax.set_title('Correlation Heatmap of Aviation Data')

plt.tight_layout()
plt.show()
```



Explanation of Correlation Between Number of Engines and Injuries/Uninjured:

The correlation heatmap visually represents the relationships between the number of engines and the different categories of injuries/uninjured in aviation accidents. Here's how to interpret the correlations shown:

1. Number of Engines vs. Total Fatal Injuries:

    - The correlation coefficient is approximately 0.03.
    - There is a very weak positive correlation between the number of engines and the total number of fatal injuries. This suggests that as the number of engines

increases, there is a slight tendency for a higher number of fatal injuries, but the relationship is very weak.

2. Number of Engines vs. Total Serious Injuries:

    - The correlation coefficient is approximately -0.01.
    - There is a very weak negative correlation between the number of engines and the total number of serious injuries. This suggests that as the number of engines increases, there is a tendency for a lower number of serious injuries, but the relationship is very weak.

3. Number of Engines vs. Total Minor Injuries:

    - The correlation coefficient is approximately 0.01.
    - There is a very weak positive correlation between the number of engines and the total number of minor injuries. This means that a higher number of engines may correspond to more minor injuries, but the relationship is very weak.

4. Number of Engines vs. Total Uninjured:

    - The correlation coefficient is approximately 0.07.
    - There is a weak positive correlation between the number of engines and the total number of uninjured passengers. This indicates that planes with more engines might have a higher number of uninjured passengers in accidents, but the relationship is very weak.

Overall Interpretation:

- Weak Relationships: All the correlations between the number of engines and the different categories of injuries/uninjured are very weak (close to zero). This means that the number of engines is not a strong predictor of whether an accident will result in fatal, serious, or minor injuries, or whether passengers will be uninjured.
- Not Determinative: The number of engines does not appear to be a significant factor in determining the severity of injuries or the number of uninjured individuals in aviation accidents.
- Other Factors: Other factors such as the nature of the accident, the specific plane model, safety measures, or emergency response are likely much more influential than

Overall Conclusions:

Based on the analysis of aviation accident data, we can draw the following conclusions:

1. **Plane Model Impact**:

    - Different plane models exhibit varying levels of safety.
    - Model 737 has the most fatal injuries and also the most uninjured passengers, indicating it is a high-volume aircraft but also involved in more severe incidents.
    - Models 172 and 152 show higher incidences of serious and minor injuries, respectively.

2. **Plane Make Impact**:

    - Cessna aircraft are involved in accidents with the highest numbers of fatal, serious, and minor injuries.
    - Boeing aircraft, while involved in accidents, have the most uninjured passengers, which may suggest better safety features or structural integrity.

3. **Amateur-Built Planes**:

- Non-amateur-built planes are generally safer, with higher numbers of uninjured passengers and fewer fatal injuries compared to amateur-built planes.
- Amateur-built planes show a higher incidence of serious injuries. This indicates a potential need for stricter regulations or more rigorous safety checks for amateur-built aircraft.

4. **Engine Type**:

   - Turbo fan engines are associated with the highest number of fatal and minor injuries, but they also have the highest number of uninjured passengers.
   - LR engines have a higher incidence of serious injuries.

5. **Number of Engines**:

   - Planes with four engines are involved in the most accidents with fatal injuries, while planes with eight engines have the most serious injuries.
   - Planes with three engines show the highest number of minor injuries and uninjured passengers.
   - There is very weak correlation between the number of engines and the severity of injuries or the number of uninjured passengers. This suggests that the number of engines is not a primary factor in determining the outcome of an accident.

Recommendations:

1. **Further Investigation of High-Risk Models**:

   - Conduct deeper analyses of plane models like 737, 172, and 152 to understand the underlying causes of accidents and injury patterns. This could involve examining accident reports in detail.

2. **Improve Safety in Amateur-Built Planes**:

   - Consider enhancing safety regulations and inspections for amateur-built aircraft, given their higher incidence of serious injuries.
   - Educational campaigns for builders and pilots of these aircraft could help in reducing accidents.

3. **Engine Type Safety Reviews**:

   - Conduct studies to investigate the safety performance of different engine types, focusing on why turbo fan and LR engines might be associated with more severe injury outcomes.
   - Consider whether different engine types should be subject to different maintenance schedules or pilot training.

4. **Focus Beyond Number of Engines**:

   - Given the weak correlation between the number of engines and accident outcomes, safety efforts should focus on factors other than the number of engines.
   - Investigate plane design, emergency protocols, pilot training, and weather conditions to identify better predictors of accident outcomes.

5. **Data Collection and Analysis**:

   - Improve data collection on aviation accidents, ensuring consistent recording of details about plane models, engines, and injury types.
   - Perform periodic analyses of the data to identify emerging safety issues or trends.

6. **Focus on plane make**:

- Further investigations should take place regarding Cessna and boeing safety records.

Recommendations

I would recommend the company to consider the following:

- Model of the aircraft: model 737. Despite having the most fatalities, it also has the most people who are not hurt. If more safety precautions are put in place, the number of uninjured may outnumber the injured.
- Make of the aircraft: Boeing. It appears to be the safest due to the large number of uninjured and moderate amount of injuries.
- Professionally built planes. Professionally built planes have proven to have more uninjured passengers as compared to amateur built ones.
- Type of engine: turbo tan engine it has the most number of uninjured passangers.

If interested in a number of options, consider the following makes:

- Boeing - McDonnell Douglas - Piper - Airbus