Final Project Submission

Please fill out:

- Student name: HILDA JEROTICH
- Student pace: Self paced / part time / full time
- Scheduled project review date/time: 27/04/2025
- Instructor name: GEORGE KAMUNDIA
- Blog post URL:

ANALYSIS OF AIRCRAFTS

Introduction

SkyNova is a company in the tourism and hospitality industry that is yet to expand into the airline industry so as to diversify its portfolio. It is interested in purchasing and operating airplanes for commercial and private enterprises. There are several potential risks facing aircrafts. The aim of the project is to determine which aircraft has the lowest risk for the company to commence its business. The data set from the National Transportation and Safety Board will be annalysed and valuable insights would be gained that would assit in making decisions on which aircraft to purchase.

Import Library

I will be using pandas library to perform data analysis and data cleaning. Pandas is imported under the alias pd.

```
import pandas as pd
```

Loading data set

```
#Import file
df = pd.read_csv('./data/Aviation_Data.csv')
```

```
c:\Users\Dell\anaconda3\envs\learn-env\lib\site-packages\IPython\core\
interactiveshell.py:3145: DtypeWarning: Columns (6,7,28) have mixed
types.Specify dtype option on import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

Exploration of Data

I will use methods and functions such as .head(), .tail(), .columns , .info(), .describe(), .shape to get more understanding about the data structure of the data set.

```
#Print the first 5 rows
df. head()

          Event.Id Investigation.Type Accident.Number  Event.Date  \
0  20001218X45444           Accident      SEA87LA080  1948-10-24
1  20001218X45447           Accident      LAX94LA336  1962-07-19
2  20061025X01555           Accident      NYC07LA005  1974-08-30
3  20001218X45448           Accident      LAX96LA321  1977-06-19
4  20041105X01764           Accident      CHI79FA064  1979-08-02

          Location        Country Latitude Longitude Airport.Code  \
0  MOOSE CREEK, ID  United States      NaN       NaN          NaN
1   BRIDGEPORT, CA  United States      NaN       NaN          NaN
2    Saltville, VA  United States  36.9222  -81.8781          NaN
3       EUREKA, CA  United States      NaN       NaN          NaN
4       Canton, OH  United States      NaN       NaN          NaN

  Airport.Name  ... Purpose.of.flight Air.carrier Total.Fatal.Injuries
\
0          NaN  ...          Personal         NaN                  2.0

1          NaN  ...          Personal         NaN                  4.0

2          NaN  ...          Personal         NaN                  3.0

3          NaN  ...          Personal         NaN                  2.0

4          NaN  ...          Personal         NaN                  1.0


  Total.Serious.Injuries Total.Minor.Injuries Total.Uninjured  \
0                    0.0                  0.0             0.0
1                    0.0                  0.0             0.0
2                    NaN                  NaN             NaN
3                    0.0                  0.0             0.0
4                    2.0                  NaN             0.0

  Weather.Condition  Broad.phase.of.flight   Report.Status
Publication.Date
0               UNK                 Cruise  Probable Cause
NaN
1               UNK                Unknown  Probable Cause       19-
09-1996
2               IMC                 Cruise  Probable Cause       26-
02-2007
3               IMC                 Cruise  Probable Cause       12-
09-2000
4               VMC               Approach  Probable Cause       16-
04-1980
```

```
[5 rows x 31 columns]

#Print the last five rows
df.tail()

             Event.Id Investigation.Type Accident.Number
Event.Date  \
90343  20221227106491           Accident      ERA23LA093  2022-12-26

90344  20221227106494           Accident      ERA23LA095  2022-12-26

90345  20221227106497           Accident      WPR23LA075  2022-12-26

90346  20221227106498           Accident      WPR23LA076  2022-12-26

90347  20221230106513           Accident      ERA23LA097  2022-12-29


            Location        Country Latitude Longitude Airport.Code  \
90343  Annapolis, MD  United States      NaN       NaN          NaN
90344    Hampton, NH  United States      NaN       NaN          NaN
90345     Payson, AZ  United States  341525N  1112021W          PAN
90346     Morgan, UT  United States      NaN       NaN          NaN
90347     Athens, GA  United States      NaN       NaN          NaN

      Airport.Name  ... Purpose.of.flight         Air.carrier  \
90343          NaN  ...          Personal                 NaN
90344          NaN  ...               NaN                 NaN
90345       PAYSON  ...          Personal                 NaN
90346          NaN  ...          Personal  MC CESSNA 210N LLC
90347          NaN  ...          Personal                 NaN

      Total.Fatal.Injuries Total.Serious.Injuries Total.Minor.Injuries
\
90343                  0.0                    1.0                  0.0

90344                  0.0                    0.0                  0.0

90345                  0.0                    0.0                  0.0

90346                  0.0                    0.0                  0.0

90347                  0.0                    1.0                  0.0


      Total.Uninjured Weather.Condition  Broad.phase.of.flight
Report.Status  \
90343             0.0               NaN                    NaN
NaN
90344             0.0               NaN                    NaN
```

```
NaN
90345               1.0                 VMC                       NaN
NaN
90346               0.0                 NaN                       NaN
NaN
90347               1.0                 NaN                       NaN
NaN

      Publication.Date
90343       29-12-2022
90344              NaN
90345       27-12-2022
90346              NaN
90347       30-12-2022

[5 rows x 31 columns]
```

```
#Print the column names
df.columns
```

```
Index(['Event.Id', 'Investigation.Type', 'Accident.Number',
'Event.Date',
       '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.Description',
       'Schedule', 'Purpose.of.flight', 'Air.carrier',
'Total.Fatal.Injuries',
       'Total.Serious.Injuries', 'Total.Minor.Injuries',
'Total.Uninjured',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

```
#Print the number of rows and columns
df.shape
```

```
(90348, 31)
```

```
#Print summary information on data types and non null counts
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 90348 entries, 0 to 90347
Data columns (total 31 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Event.Id                88889 non-null  object
 1   Investigation.Type      90348 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        73659 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.4+ MB
```

```
#Print statistical summary of the data
df.describe()
```

```
       Number.of.Engines  Total.Fatal.Injuries  Total.Serious.Injuries  \
count       82805.000000          77488.000000            76379.000000

mean            1.146585              0.647855                0.279881

std             0.446510              5.485960                1.544084

min             0.000000              0.000000                0.000000

25%             1.000000              0.000000                0.000000

50%             1.000000              0.000000                0.000000

75%             1.000000              0.000000                0.000000
```

```
max                  8.000000             349.000000              161.000000


       Total.Minor.Injuries  Total.Uninjured
count          76956.000000     82977.000000
mean               0.357061         5.325440
std                2.235625        27.913634
min                0.000000         0.000000
25%                0.000000         0.000000
50%                0.000000         1.000000
75%                0.000000         2.000000
max              380.000000       699.000000
```

Data Cleaning

After analysis of the data, I found out that the column names have (.) that needed to be replaced with (_). Some of the characters in the column names are in upper case and needed to be replaced with lower case. There are also missing values that need to be removed or substituted with data such as mean, median or any predicted value based on other variables.

```
#Print column names
df.columns

Index(['Event.Id', 'Investigation.Type', 'Accident.Number',
'Event.Date',
       '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.Description',
       'Schedule', 'Purpose.of.flight', 'Air.carrier',
'Total.Fatal.Injuries',
       'Total.Serious.Injuries', 'Total.Minor.Injuries',
'Total.Uninjured',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')

#clean by replacing (.) to (_)
df.columns=[col.replace('.','_') for col in df.columns]

#clean column names to lower case
df.columns=[col.lower() for col in df.columns]

#print the cleaned column names
df.columns

Index(['event_id', 'investigation_type', 'accident_number',
'event_date',
       '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_description',
       'schedule', 'purpose_of_flight', 'air_carrier',
'total_fatal_injuries',
       'total_serious_injuries', 'total_minor_injuries',
'total_uninjured',
       'weather_condition', 'broad_phase_of_flight', 'report_status',
       'publication_date'],
      dtype='object')

#creating new variable df_filtered
columns_to_keep= ['make', 'model', 'total_fatal_injuries',
'total_serious_injuries', 'total_minor_injuries', 'total_uninjured',
'injury_severity', 'aircraft_damage',
'broad_phase_of_flight','weather_condition', 'number_of_engines',
'engine_type', 'aircraft_category']

df_filtered=df[columns_to_keep]

print('\nFiltered dataset:')
print(df_filtered.head())

#save to a new file
df_filtered.to_csv('filtered.csv', index=False)


Filtered dataset:
       make     model  total_fatal_injuries  total_serious_injuries  \
0   Stinson     108-3                   2.0                     0.0
1     Piper  PA24-180                   4.0                     0.0
2    Cessna      172M                   3.0                     NaN
3  Rockwell       112                   2.0                     0.0
4    Cessna       501                   1.0                     2.0

   total_minor_injuries  total_uninjured injury_severity
aircraft_damage  \
0                   0.0              0.0        Fatal(2)
Destroyed
1                   0.0              0.0        Fatal(4)
Destroyed
2                   NaN              NaN        Fatal(3)
Destroyed
3                   0.0              0.0        Fatal(2)
Destroyed
4                   NaN              0.0        Fatal(1)
Destroyed

  broad_phase_of_flight weather_condition  number_of_engines
```

```
engine_type  \
0                 Cruise                    UNK                  1.0
Reciprocating
1                Unknown                    UNK                  1.0
Reciprocating
2                 Cruise                    IMC                  1.0
Reciprocating
3                 Cruise                    IMC                  1.0
Reciprocating
4               Approach                    VMC                  NaN
NaN

  aircraft_category
0               NaN
1               NaN
2               NaN
3               NaN
4               NaN

#remove rows with null data
df_filtered_cleaned= df_filtered.dropna()

print('.\ncleaned data:')
print(df_filtered_cleaned.head())

#save to file
df_filtered_cleaned.to_csv('cleaned data.csv', index=False)

.
cleaned data:
        make   model  total_fatal_injuries  total_serious_injuries  \
7     Cessna     140                   0.0                     0.0
8     Cessna    401B                   0.0                     0.0
12  Bellanca  17-30A                   0.0                     0.0
13    Cessna   R172K                   1.0                     0.0
14    Navion       A                   1.0                     0.0

    total_minor_injuries  total_uninjured injury_severity
aircraft_damage  \
7                    0.0              2.0       Non-Fatal
Substantial
8                    0.0              2.0       Non-Fatal
Substantial
12                   1.0              0.0       Non-Fatal
Destroyed
13                   0.0              0.0        Fatal(1)
Destroyed
14                   0.0              0.0        Fatal(1)
Destroyed
```

```
   broad_phase_of_flight weather_condition  number_of_engines
engine_type  \
7                 Takeoff               VMC                1.0
Reciprocating
8                 Landing               IMC                2.0
Reciprocating
12                 Cruise               IMC                1.0
Reciprocating
13                Takeoff               IMC                1.0
Reciprocating
14                 Cruise               IMC                1.0
Reciprocating

   aircraft_category
7           Airplane
8           Airplane
12          Airplane
13          Airplane
14          Airplane
```

```
df_filtered_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3585 entries, 7 to 63908
Data columns (total 13 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   make                    3585 non-null   object
 1   model                   3585 non-null   object
 2   total_fatal_injuries    3585 non-null   float64
 3   total_serious_injuries  3585 non-null   float64
 4   total_minor_injuries    3585 non-null   float64
 5   total_uninjured         3585 non-null   float64
 6   injury_severity         3585 non-null   object
 7   aircraft_damage         3585 non-null   object
 8   broad_phase_of_flight   3585 non-null   object
 9   weather_condition       3585 non-null   object
 10  number_of_engines       3585 non-null   float64
 11  engine_type             3585 non-null   object
 12  aircraft_category       3585 non-null   object
dtypes: float64(5), object(8)
memory usage: 392.1+ KB
```

```
#set 'model' as index
df_filtered_cleaned.set_index('model', inplace=True)

#checking my index
df_filtered_cleaned.index
```

```
Index(['140', '401B', '17-30A', 'R172K', 'A', '19', '280C', '180',
'172',
       'WCS-222 (BELL 47G)',
       ...
       'A36TC', 'Aero Canard', 'PA32-260', '777-222', '182R', 'M20E',
       'PA-46-310P', 'Sonex', 'RAF 2000 GTX', '206L-3'],
      dtype='object', name='model', length=3585)
```

Data Visualization

After annalyzing and cleaning my data, I will have to visualize my data by plotting bar charts. My goal is to determine which aircraft model has the lowest risk. Risk will depend on injury severity(total fatal injuries, total serious injuries and total minor injuries), aircraft damage, broad phase of flight, weather conditions, engine type, number of engines and artifact category.

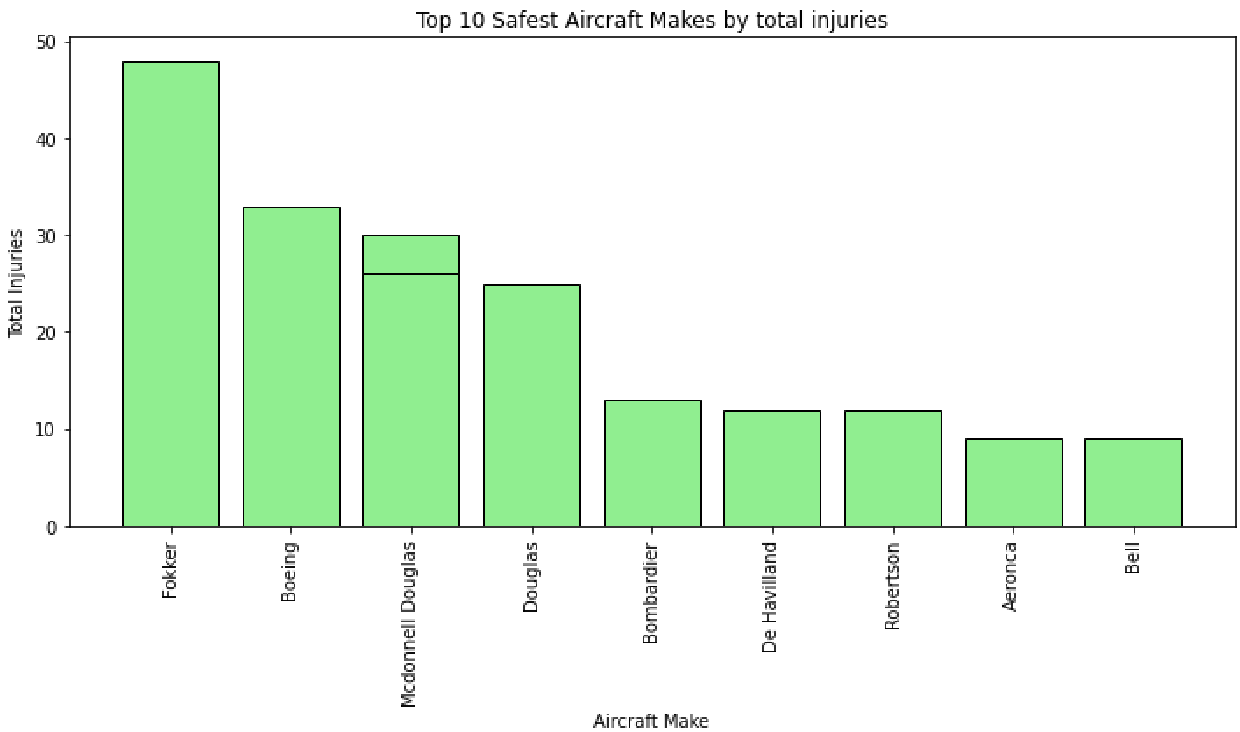
Importing Library

```
#importing the necessary library
import matplotlib.pyplot as plt
%matplotlib inline
```

Comparing Injuries by Make

```
#Calculate total risk per make
df_filtered_cleaned['total_injuries']
=(df_filtered_cleaned['total_fatal_injuries'] +
df_filtered_cleaned['total_serious_injuries'] +
df_filtered_cleaned['total_minor_injuries'])

<ipython-input-43-b63d03d35570>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df_filtered_cleaned['total_injuries']
=(df_filtered_cleaned['total_fatal_injuries'] +
df_filtered_cleaned['total_serious_injuries'] +
df_filtered_cleaned['total_minor_injuries'])

#Sort values and take the top 10
safest_10 =
df_filtered_cleaned.sort_values('total_injuries',ascending=False).head
(10)

x = safest_10['make']
heights = safest_10['total_injuries']
```

```
#Plot
fig, ax = plt.subplots(figsize=(10,6))
ax.bar(x, heights, color='lightgreen', edgecolor='black')

ax.set_title('Top 10 Safest Aircraft Makes by total injuries')
ax.set_xlabel('Aircraft Make')
ax.set_ylabel('Total Injuries')

plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



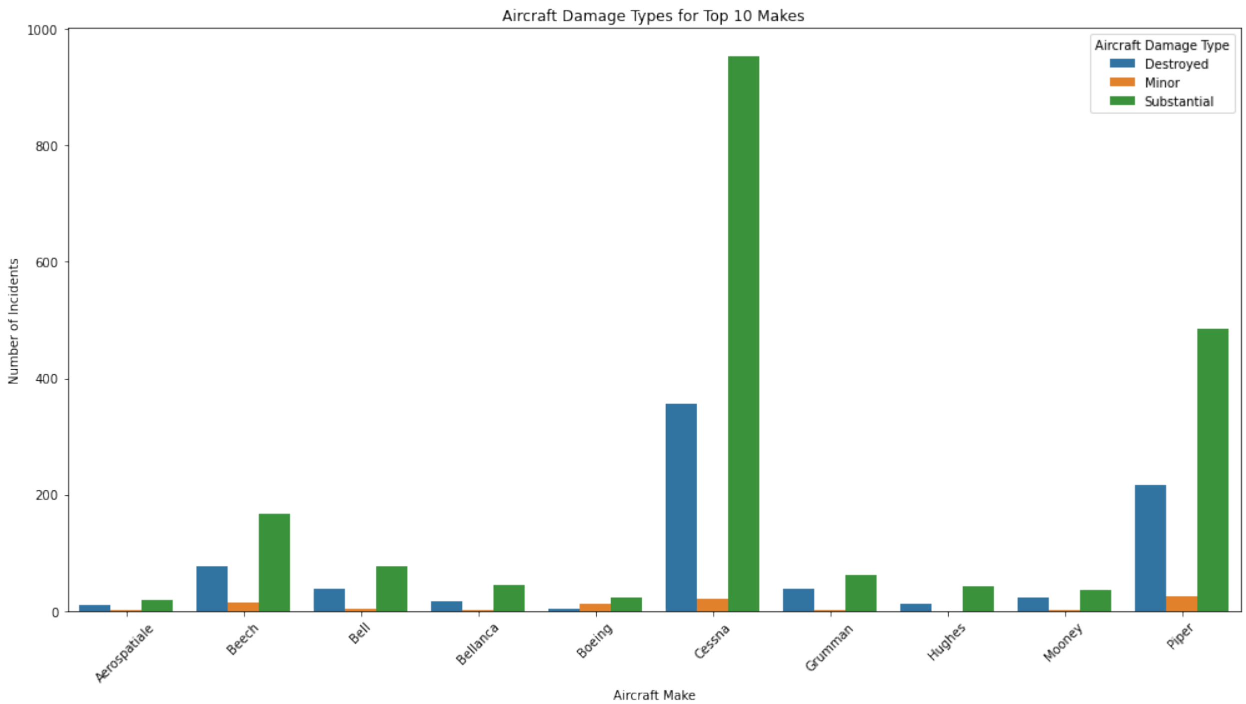

```
"""
Results:

Bell and Aeronca are the aircraft makes with the least total injuries.

Fokker is the aircraft make with the highest total injuries.

"""
```

```
'\nResults:\n\nBell and Aeronca are the aircraft makes with the least
total injuries. \nFokker is the aircraft make with the highest total
injuries.\n\n'
```

Compare Aircraft Damage Frequency by Make

```python
# Group by both 'make' and 'aircraft_damage' and count occurrences
damage_counts = df_filtered_cleaned.groupby(['make',
'aircraft_damage']).size().reset_index(name='counts')

# Check the result
print(damage_counts.head())

#Plot
import seaborn as sns

# Take only top 10 makes by overall damage counts
top_10_makes = damage_counts.groupby('make')
['counts'].sum().sort_values(ascending=False).head(10).index

# Filter damage_counts to only include top 10 makes
damage_counts_top10 =
damage_counts[damage_counts['make'].isin(top_10_makes)]

# Plot grouped bar chart
plt.figure(figsize=(14,8))
sns.barplot(data=damage_counts_top10, x='make', y='counts',
hue='aircraft_damage')

plt.title('Aircraft Damage Types for Top 10 Makes')
plt.xlabel('Aircraft Make')
plt.ylabel('Number of Incidents')
plt.xticks(rotation=45)
plt.legend(title='Aircraft Damage Type')
plt.tight_layout()
plt.show()
```

```
             make aircraft_damage  counts
0           Adams     Substantial       1
1  Aero Commander       Destroyed       9
2  Aero Commander     Substantial       8
3         Aeronca       Destroyed       8
4         Aeronca           Minor       2
```



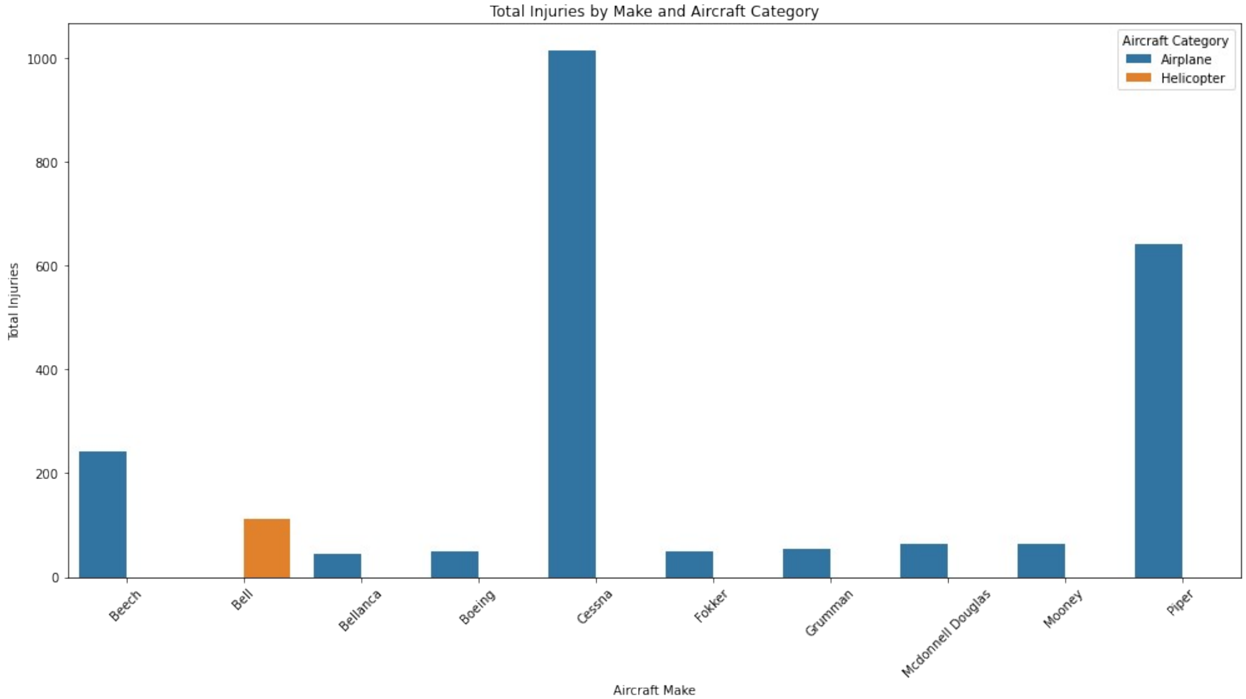

```
"""
Results:

Cessna is the aircraft make with the highest destroyed aircraft damage type while
boeing is the aircraft make with the lowest destroyed aircraft damage type.

Piper is the aircraft make with the highest minor aircraft damage type while Hughes
is the aircraft make with the lowest minor aircraft damage type.

Cessna is the aircraft make with the highest substantial aircraft damage type while
Aerospatiale is the aircraft make with the lowest substantial aircraft damage type.

"""
```

```
'\nResults:\n\nCessna is the aircraft make with the highest destroyed aircraft damage type while \nboeing is the aircraft make with the lowest destroyed aircraft damage type.\n\nPiper is the aircraft make with the highest minor aircraft damage type while Hughes\nis the aircraft make with the lowest minor aircraft damage type.\n\nCessna is the aircraft make with the highest substantial aircraft damage type while\nAerospatiale is the aircraft make with the lowest substantial aircraft damage type.\n\n'
```

Injury Rates by Aircraft Category and Make

```
# Group by both 'make' and 'aircraft_category', and sum
'total_injuries'
injury_counts = df_filtered_cleaned.groupby(['make',
'aircraft_category'])['total_injuries'].sum().reset_index()

# Check result
print(injury_counts.head())

# pick top 10 makes with highest injuries to make plot clean
top_10_makes = injury_counts.groupby('make')
['total_injuries'].sum().sort_values(ascending=False).head(10).index
injury_counts_top10 =
injury_counts[injury_counts['make'].isin(top_10_makes)]

# Plot grouped bar chart
plt.figure(figsize=(14,8))
sns.barplot(data=injury_counts_top10, x='make', y='total_injuries',
hue='aircraft_category')

plt.title('Total Injuries by Make and Aircraft Category')
plt.xlabel('Aircraft Make')
plt.ylabel('Total Injuries')
plt.xticks(rotation=45)
plt.legend(title='Aircraft Category')
plt.tight_layout()
plt.show()
```

```
               make aircraft_category  total_injuries
0             Adams          Balloon             0.0
1    Aero Commander         Airplane            12.0
2           Aeronca         Airplane            38.0
3     Aeronca Champ         Airplane             0.0
4  Aeronca Champion         Airplane             0.0
```



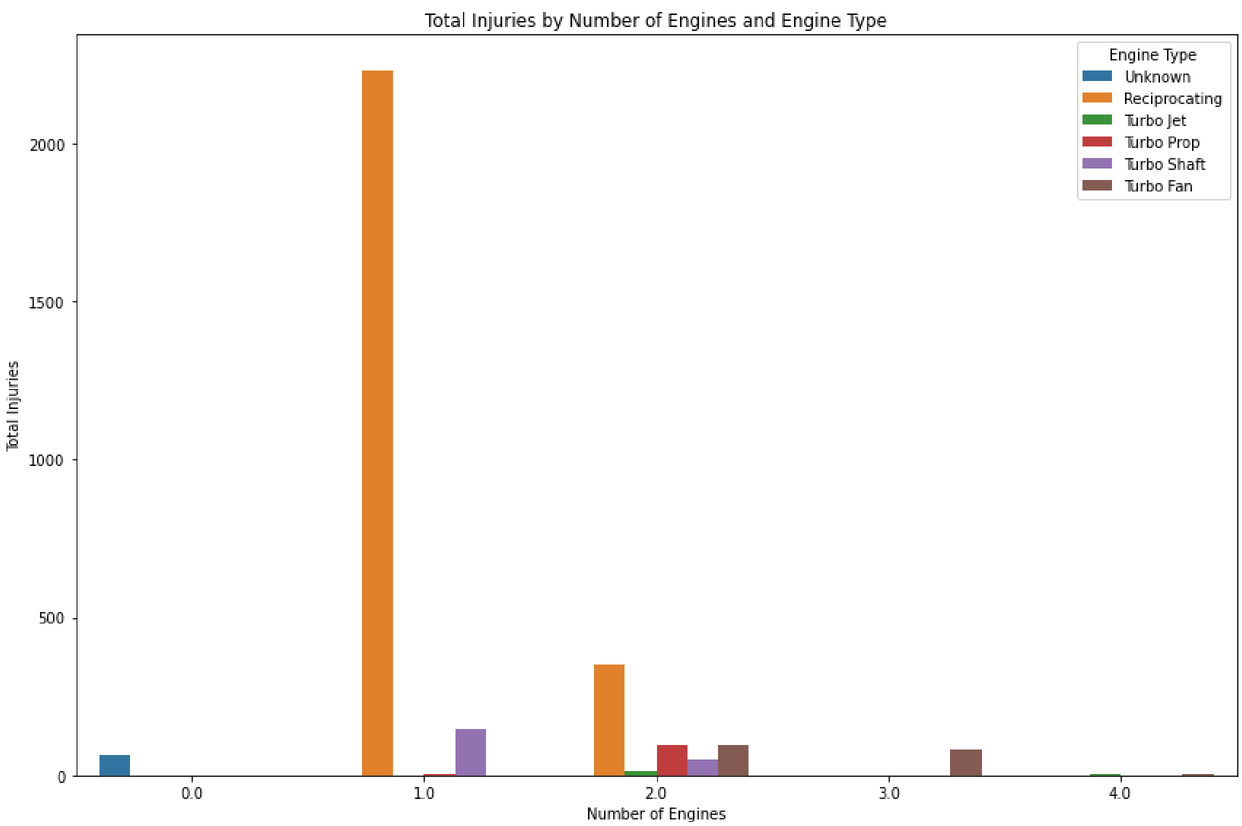

```
"""
Results:

Cessna of the aircraft category airplane has the highest number of
total injuries.
Bellanca of the aircraft category airplane has the lowest number of
total injuries.
Bell is the only one from my sample of the artifact category
helicopter and has 100 total injuries
which among the least injuries.

"""

'\nResults:\n\nCessna of the aircraft category airplane has the
highest number of total injuries.\nBellanca of the aircraft category
airplane has the lowest number of total injuries.\nBell is the only
one from my sample of the artifact category helicopter and has 100
total injuries.\n\n'
```

Effects of Number of Engines and Engine Type

```
# Group by number_of_engines and engine_type, and sum injuries
engine_counts = df_filtered_cleaned.groupby(['number_of_engines',
'engine_type'])['total_injuries'].sum().reset_index()

# See the result
print(engine_counts.head())
```

```python
#Plot grouped bar chart
plt.figure(figsize=(12,8))
sns.barplot(data=engine_counts, x='number_of_engines',
y='total_injuries', hue='engine_type')

plt.title('Total Injuries by Number of Engines and Engine Type')
plt.xlabel('Number of Engines')
plt.ylabel('Total Injuries')
plt.legend(title='Engine Type')
plt.tight_layout()
plt.show()
```

```
   number_of_engines    engine_type  total_injuries
0                0.0        Unknown            62.0
1                1.0  Reciprocating          2233.0
2                1.0      Turbo Jet             0.0
3                1.0     Turbo Prop             3.0
4                1.0    Turbo Shaft           149.0
```




```python
"""
Results:

Reciprocating is the engine type with the highest number of injuries
followed by turbo prop and
```

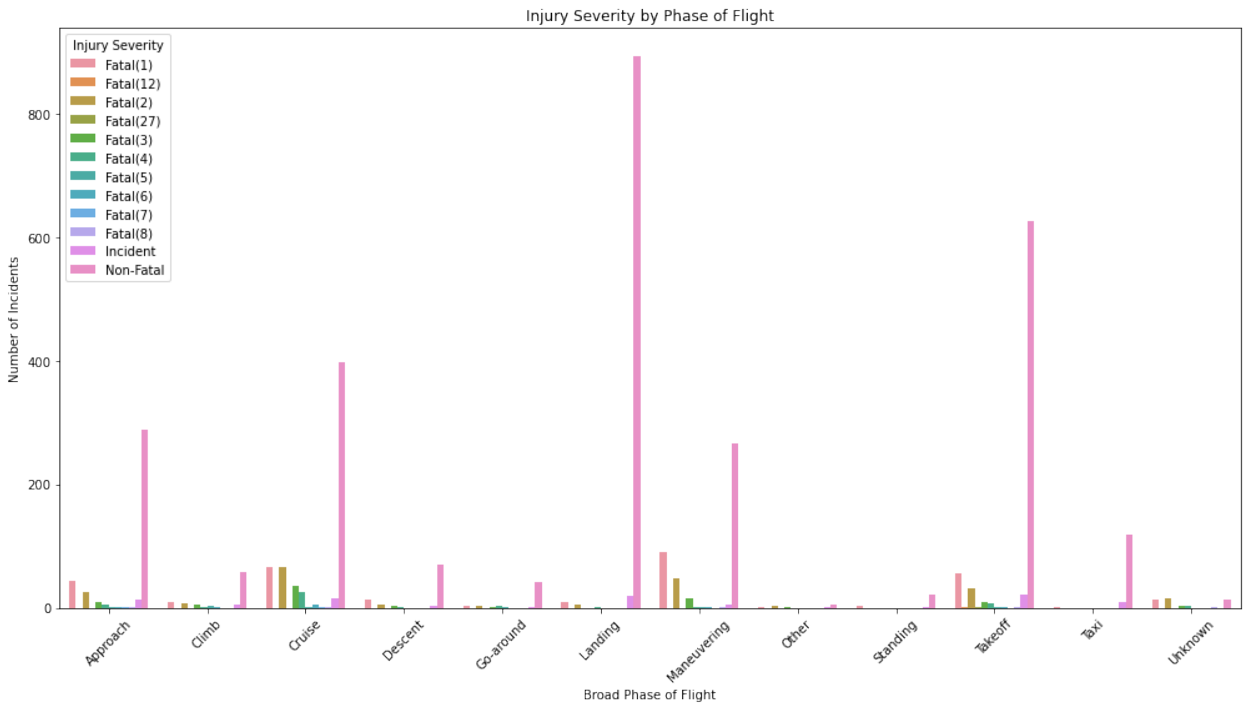
```
turbo fan then unknown then turbo shaft then lastly turbo jet.

"""
```

```
'\nResults:\n\nReciprocating is the engine type with the highest number of injuries followed by turbo prop and\nturbo fan then unknown then turbo shaft then lastly turbo jet.\n\n'
```

Relation between Injury Severity and Broad Phase of Flight

```
# Group by injury severity and broad phase of flight
injury_phase_counts = df_filtered_cleaned.groupby(['injury_severity', 'broad_phase_of_flight']).size().reset_index(name='counts')

# See the grouped data
print(injury_phase_counts.head())

#Plot grouped bar chart
plt.figure(figsize=(14,8))
sns.barplot(data=injury_phase_counts, x='broad_phase_of_flight', y='counts', hue='injury_severity')

plt.title('Injury Severity by Phase of Flight')
plt.xlabel('Broad Phase of Flight')
plt.ylabel('Number of Incidents')
plt.xticks(rotation=45)
plt.legend(title='Injury Severity')
plt.tight_layout()
plt.show()
```

```
  injury_severity broad_phase_of_flight  counts
0        Fatal(1)              Approach      43
1        Fatal(1)                 Climb       9
2        Fatal(1)                Cruise      66
3        Fatal(1)               Descent      14
4        Fatal(1)             Go-around       4
```

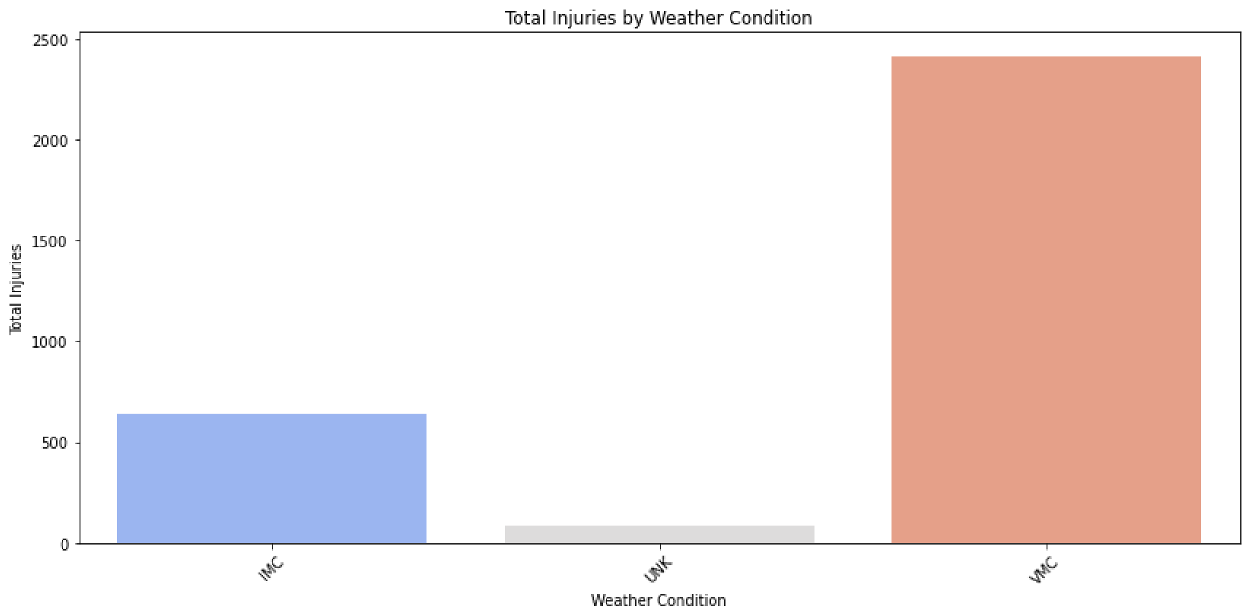



```
"""
Results:

Most of the fatal injuries happened during landing and takeoff phases.
Least of the injuries
happened during unknown, go-ground, standing and descent phases.

"""

'\nResults:\n\nFatal 1 in all the broad phases of flight has the
highest number of injuries.\n\n'
```

Weather Condition by total injuries

```
# Group by weather condition and sum total injuries
weather_injuries = df_filtered_cleaned.groupby('weather_condition')
['total_injuries'].sum().reset_index()

# See the result
print(weather_injuries.head())

plt.figure(figsize=(12,6))
sns.barplot(data=weather_injuries, x='weather_condition',
y='total_injuries', palette='coolwarm')

plt.title('Total Injuries by Weather Condition')
plt.xlabel('Weather Condition')
plt.ylabel('Total Injuries')
```

```
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

  weather_condition  total_injuries
0               IMC           644.0
1               UNK            88.0
2               VMC          2411.0
```




```
"""
Results:

VMC is the weather condition with the highest number of injuries
followed by IMC weather condtion
and lastly is the UNK weather condition.

"""

'\nResults:\n\nVMC is the weather condition with the highest number of injuries followed by IMC weather condtion \nand lastly is the UNK weather condition.\n\n'
```

Conclusion

In conclusion, after annalyzing data from the National Transpotation Safety Board, I came up with valuable insights based on factors such as make, total injuries, injury severity, aircraft damage, broad phase flight, weather conditions, number of engines, engine type and aircraft category. Aircraft makes such as Bell consistently showed lower total injury counts. Most of the fatal injuries happened during landing and takeoffs phases, highlighting the artificial importance of pilot training and rigorous maintenance protocals during these phases.

A large number of injuries occurred under Visual Meteorological Conditions (VMC), likely because more flights happen during clear weather. However, Instrument Meteorological Conditions (IMC) like fog and rain still pose significant risks that should not be ignored.Aircraft with fewer engines (especially single-engine reciprocating types) were involved in more injury incidents, suggesting that for commercial operations, investing in multi-engine, turbine-powered aircraft could improve overall safety margins.Small airplanes had higher reported injuries compared to larger categories. While small aircraft are cheaper to operate, risk mitigation strategies (such as enhanced inspection and pilot standards) are necessary.