# Kengele Aviation

**Introduction**

In this project we are studying 'Kengele Aviation' a fictious company that is interested in venturing into the aviation industry as a diversification from it's main investment portfolio.

The dataset includes information on various aspects, such as aircraft make and model, injury severity, aircraft damage and the purpose of the flight.

In our approach we shall focus on the key insights for the aviation company in starting up its local business.

```
#importing libraries

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

```
df = pd. read_csv('/content/AviationData 3.csv', encoding='latin-1')
```

```
<ipython-input-206-be3a3b0a7ddc>:1: DtypeWarning: Columns (6,7,28) have mixed types. Specify dtype option on import or s
  df = pd. read_csv('/content/AviationData 3.csv', encoding='latin-1')
```

```
df.head()
```

| | Event.Id | Investigation.Type | Accident.Number | Event.Date | Location | Country | Latitude | Longitude | Airport.Code |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 20001218X45444 | Accident | SEA87LA080 | 1948-10-24 | MOOSE CREEK, ID | United States | NaN | NaN | NaN |
| **1** | 20001218X45447 | Accident | LAX94LA336 | 1962-07-19 | BRIDGEPORT, CA | United States | NaN | NaN | NaN |
| **2** | 20061025X01555 | Accident | NYC07LA005 | 1974-08-30 | Saltville, VA | United States | 36.922223 | -81.878056 | NaN |
| **3** | 20001218X45448 | Accident | LAX96LA321 | 1977-06-19 | EUREKA, CA | United States | NaN | NaN | NaN |
| **4** | 20041105X01764 | Accident | CHI79FA064 | 1979-08-02 | Canton, OH | United States | NaN | NaN | NaN |

5 rows × 31 columns

```
df.tail()
```

| | Event.Id | Investigation.Type | Accident.Number | Event.Date | Location | Country | Latitude | Longitude | Airport.Code |
|---|---|---|---|---|---|---|---|---|---|
| **88884** | 20221227106491 | Accident | ERA23LA093 | 2022-12-26 | Annapolis, MD | United States | NaN | NaN | NaN |
| **88885** | 20221227106494 | Accident | ERA23LA095 | 2022-12-26 | Hampton, NH | United States | NaN | NaN | NaN |
| **88886** | 20221227106497 | Accident | WPR23LA075 | 2022-12-26 | Payson, AZ | United States | 341525N | 1112021W | PAN |
| **88887** | 20221227106498 | Accident | WPR23LA076 | 2022-12-26 | Morgan, UT | United States | NaN | NaN | NaN |
| **88888** | 20221230106513 | Accident | ERA23LA097 | 2022-12-29 | Athens, GA | United States | NaN | NaN | NaN |

5 rows × 31 columns

```
#this data contains a big set of unnecessary information for the project in mind therefore it requires data cleaning
df.shape
```

```
(88889, 31)
```

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

```
                           0
              Event.Id     0
    Investigation.Type     0
       Accident.Number     0
            Event.Date     0
              Location    52
               Country   226
              Latitude 54507
             Longitude 54516
          Airport.Code 38757
          Airport.Name 36185
       Injury.Severity  1000
       Aircraft.damage  3194
     Aircraft.Category 56602
   Registration.Number  1382
                  Make    63
                 Model    92
         Amateur.Built   102
     Number.of.Engines  6084
           Engine.Type  7096
       FAR.Description 56866
              Schedule 76307
     Purpose.of.flight  6192
           Air.carrier 72241
    Total.Fatal.Injuries 11401
  Total.Serious.Injuries 12510
    Total.Minor.Injuries 11933
       Total.Uninjured  5912
     Weather.Condition  4492
  Broad.phase.of.flight 27165
         Report.Status  6384
      Publication.Date 13771

dtype: int64
```

## Data Cleaning

This invloves dropping unnecessary data that doesnt support Kengeles Aviation decision to purchase aircrafts.

```
#dropping the unncessary columns in whole as this research mainly focuses on finding out what are best aircraft with the lea

relevant_colums = [
    'Event.Id',
    'Investigation.Type',
    'Injury.Severity',
    'Aircraft.damage',
    'Aircraft.Category',
    'Make',
    'Model',
    'Purpose.of.flight',
    'Air.carrier',
    'Total.Fatal.Injuries',
    'Total.Serious.Injuries',
    'Total.Minor.Injuries',
    'Total.Uninjured'
]
```

```
df = df [relevant_colums]
```

```
df.shape
```

```
(88889, 13)
```

```
df.head()
```

| | Event.Id | Investigation.Type | Injury.Severity | Aircraft.damage | Aircraft.Category | Make | Model | Purpose.of.flig |
|---|---|---|---|---|---|---|---|---|
| **0** | 20001218X45444 | Accident | Fatal(2) | Destroyed | NaN | Stinson | 108-3 | Persoı |
| **1** | 20001218X45447 | Accident | Fatal(4) | Destroyed | NaN | Piper | PA24-180 | Persoı |
| **2** | 20061025X01555 | Accident | Fatal(3) | Destroyed | NaN | Cessna | 172M | Persoı |
| **3** | 20001218X45448 | Accident | Fatal(2) | Destroyed | NaN | Rockwell | 112 | Persoı |
| **4** | 20041105X01764 | Accident | Fatal(1) | Destroyed | NaN | Cessna | 501 | Persoı |

Next steps: Generate code with df View recommended plots New interactive sheet

```
#mainly focusing on the accident type of investigation as incidents are assumed to be caused by human error
df =df[df["Investigation.Type"] == "Accident"]
df
```

| | Event.Id | Investigation.Type | Injury.Severity | Aircraft.damage | Aircraft.Category | Make | Model | Purpose.( |
|---|---|---|---|---|---|---|---|---|
| **0** | 20001218X45444 | Accident | Fatal(2) | Destroyed | NaN | Stinson | 108-3 | |
| **1** | 20001218X45447 | Accident | Fatal(4) | Destroyed | NaN | Piper | PA24-180 | |
| **2** | 20061025X01555 | Accident | Fatal(3) | Destroyed | NaN | Cessna | 172M | |
| **3** | 20001218X45448 | Accident | Fatal(2) | Destroyed | NaN | Rockwell | 112 | |
| **4** | 20041105X01764 | Accident | Fatal(1) | Destroyed | NaN | Cessna | 501 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | |
| **88884** | 20221227106491 | Accident | Minor | NaN | NaN | PIPER | PA-28-151 | |
| **88885** | 20221227106494 | Accident | NaN | NaN | NaN | BELLANCA | 7ECA | |
| **88886** | 20221227106497 | Accident | Non-Fatal | Substantial | Airplane | AMERICAN CHAMPION AIRCRAFT | 8GCBC | |
| **88887** | 20221227106498 | Accident | NaN | NaN | NaN | CESSNA | 210N | |
| **88888** | 20221230106513 | Accident | Minor | NaN | NaN | PIPER | PA-24-260 | |

85015 rows × 13 columns

Next steps: Generate code with df View recommended plots New interactive sheet

```
df.shape
```

```
(85015, 13)
```

```
#droping all null values to reduce the data set
df.dropna (inplace=True)
```

```
df.shape
```

```
(10658, 13)
```

```
df.head()
```

| | Event.Id | Investigation.Type | Injury.Severity | Aircraft.damage | Aircraft.Category | Make | Model | Purpose.of.flig |
|---|---|---|---|---|---|---|---|---|
| **22** | 20020917X01657 | Accident | Non-Fatal | Substantial | Helicopter | Bell | 206L-1 | Unknow |
| **33** | 20020917X02254 | Accident | Fatal(2) | Destroyed | Airplane | Piper | PA-34-200T | Unknow |
| **39** | 20020917X01906 | Accident | Fatal(2) | Destroyed | Airplane | Piper | PA-31 | Unknow |
| **43** | 20020917X02156 | Accident | Non-Fatal | Substantial | Airplane | Cessna | TR182 | Unknow |
| **44** | 20020917X01651 | Accident | Non-Fatal | Substantial | Airplane | Cessna | A185F | Unknow |

Next steps: Generate code with df View recommended plots New interactive sheet

```
# Filter for aircraft category 'Airplane'

df = df[df['Aircraft.Category'] == 'Airplane']
df.shape
```

```
(8725, 13)
```

```
df['Purpose.of.flight'].unique()
```

```
array(['Unknown', 'Personal', 'Instructional', 'Business', 'Skydiving',
       'Positioning', 'Aerial Application', 'Other Work Use',
       'Glider Tow', 'Ferry', 'Executive/corporate', 'Public Aircraft',
       'Aerial Observation', 'Flight Test', 'Banner Tow',
       'Public Aircraft - Federal', 'Air Race show', 'Firefighting',
       'Public Aircraft - State', 'Public Aircraft - Local', 'Air Drop',
       'PUBS', 'ASHO'], dtype=object)
```

```
#We will only use planes who's purpose of flight include Business, and Executive as they align with the business needs.
df = df[df['Purpose.of.flight'].isin(['Business', 'Executive/corporate'])]
df.shape
```

```
(351, 13)
```

## Key Business

This is where the data has been cleaned to mainly contain what is relevant for the business

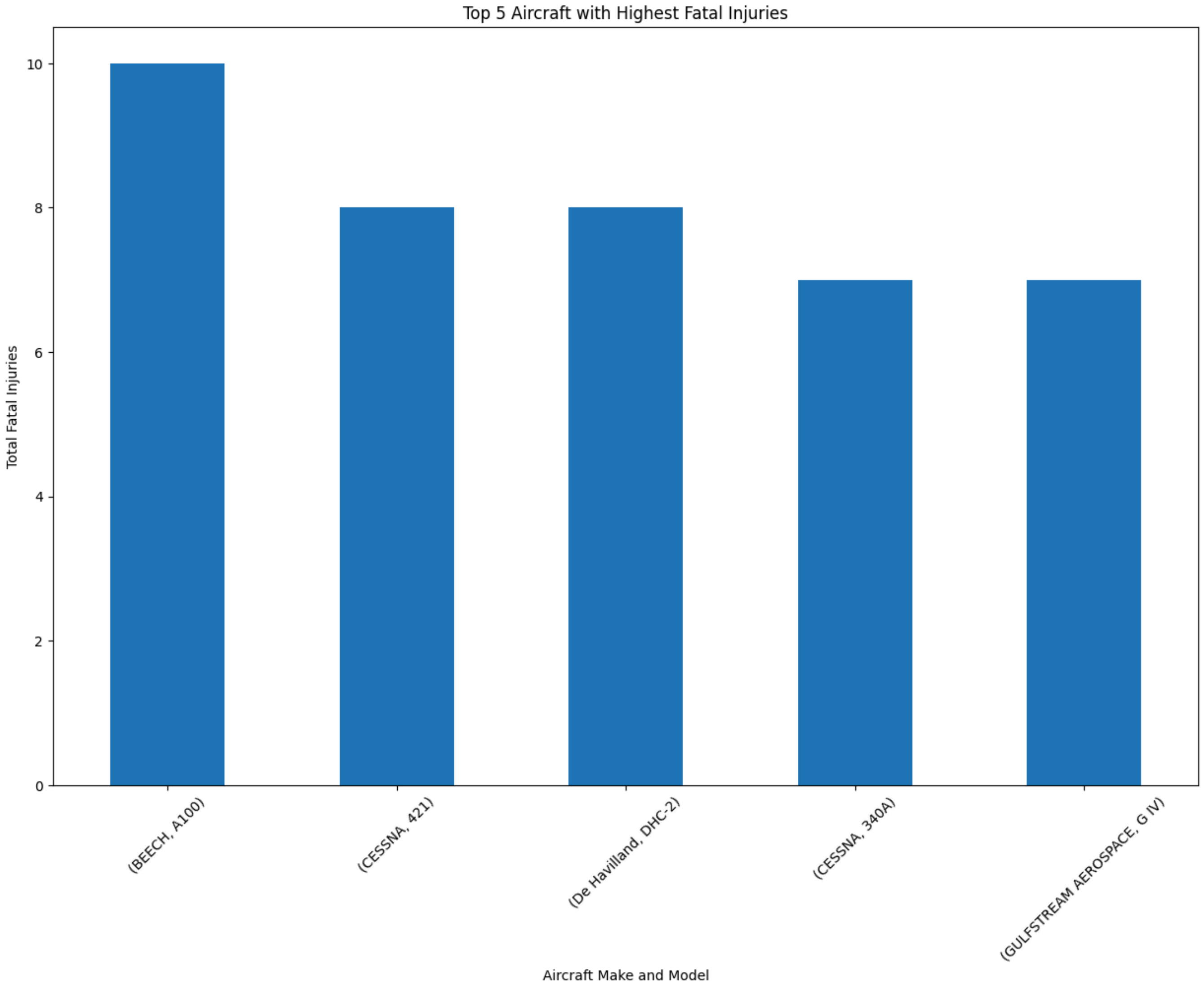
```
# 5 aircrafts that have the highest fatal injuries

top_fatal_aircraft = df.groupby(['Make', 'Model'])['Total.Fatal.Injuries'].sum().nlargest(5)
print(top_fatal_aircraft)
```

```
Make                   Model
BEECH                  A100     10.0
CESSNA                 421       8.0
De Havilland           DHC-2     8.0
CESSNA                 340A      7.0
GULFSTREAM AEROSPACE   G IV      7.0
Name: Total.Fatal.Injuries, dtype: float64
```

```
# A graph containing the 5 most dangerous planes makes and model

top_fatal_aircraft.plot(kind='bar', figsize=(15, 10))
plt.title('Top 5 Aircraft with Highest Fatal Injuries')
plt.xlabel('Aircraft Make and Model')
plt.ylabel('Total Fatal Injuries')
plt.xticks(rotation=45)
plt.show()
```



```
#The 10 aircrafts make and model that have the least fatal injuries

least_fatal_aircraft = df.groupby(['Make','Model'])['Total.Fatal.Injuries'].sum().nsmallest(10)
print(least_fatal_aircraft)
```

```
Make              Model
AERO COMMANDER    100       0.0
                  500       0.0
                  685       0.0
AIR TRACTOR       AT502     0.0
                  AT802     0.0
AIR TRACTOR INC   AT-302    0.0
Aero Commander    695       0.0
BAE               BAE125    0.0
BEECH             1900      0.0
                  200       0.0
Name: Total.Fatal.Injuries, dtype: float64
```

```
# TOP 10 MAKE AND MODEL WITH HIGHEST TOTAL UNINJURED

top_uninjured_aircraft = df.groupby(['Make', 'Model'])['Total.Uninjured'].sum().nlargest(10)
print(top_uninjured_aircraft)
```

```
Make                       Model
CESSNA                     560XL         27.0
                           208B          24.0
                           525A          16.0
Raytheon Aircraft Company  1900D         15.0
Gulfstream                 GIV           14.0
PILATUS                    PC12          12.0
BOMBARDIER                 BD100 1A10    11.0
DEHAVILLAND                DHC-3         11.0
Piper                      PA46          11.0
BEECH                      58            10.0
Name: Total.Uninjured, dtype: float64
```

```
# 5 MAKES WITH MOST AIRCRAFT DAMAGE

top_damage_makes = df.groupby('Make')['Aircraft.damage'].count().nlargest(5)
print(top_damage_makes)
```

```
Make
CESSNA                109
PIPER                  43
BEECH                  38
CIRRUS DESIGN CORP     11
Cessna                 11
Name: Aircraft.damage, dtype: int64
```

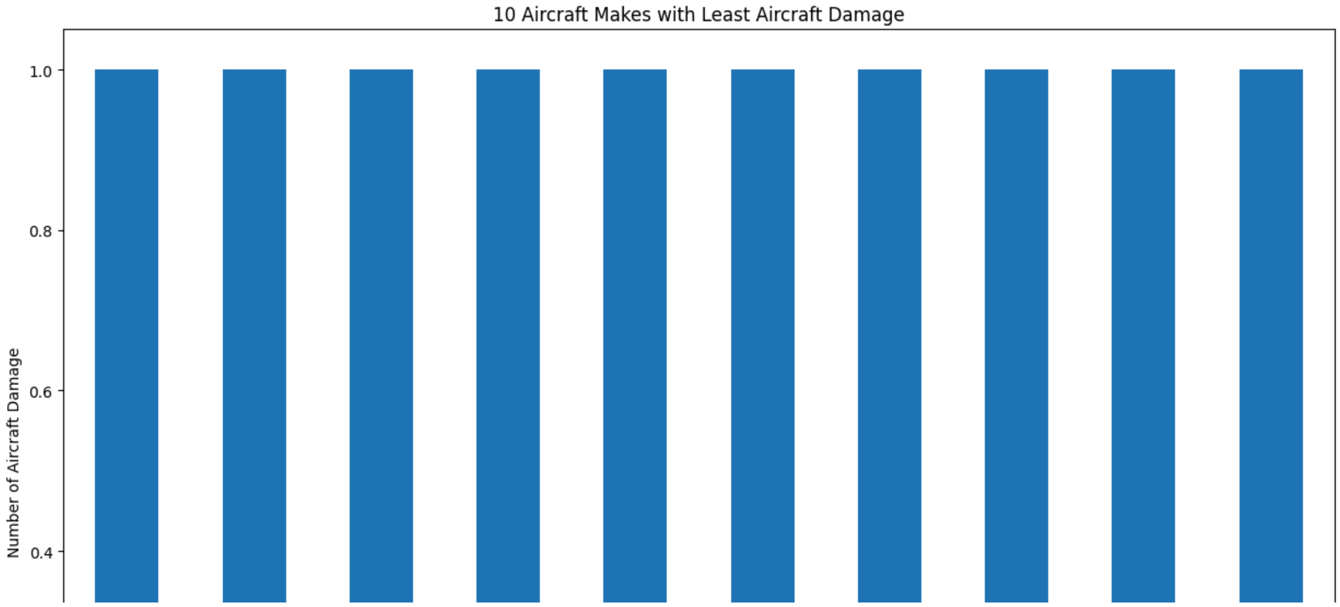
```
#5 MAKES WITH LEAST AIRCRAFT DAMAGE

least_damage_makes = df.groupby('Make')['Aircraft.damage'].count().nsmallest(10)
print(least_damage_makes)
```

```
Make
AERO VODOCHODY     1
AIR TRACTOR INC    1
Aero Commander     1
Air Tractor        1
BAE                1
BEECHCRAFT         1
BELLANCA           1
BOMBARDIER         1
BOMBARDIER INC     1
BRISTELL           1
Name: Aircraft.damage, dtype: int64
```

```
#GRAPH WITH LEAST AIRCRAFT DAMAGE

least_damage_makes.plot(kind='bar', figsize=(15, 10))
plt.title('10 Aircraft Makes with Least Aircraft Damage')
plt.xlabel('Aircraft Make')
plt.ylabel('Number of Aircraft Damage')
plt.show()
```



```
# Filter for AERO COMMANDER aircraft
aero_commander_df = df[df['Make'] == 'AERO COMMANDER']

# Get the top 5 models with the highest count
top_aero_commander_models = aero_commander_df.groupby('Model')['Model'].count().nlargest(5)
print(top_aero_commander_models)
```

```
Model
100    1
500    1
685    1
Name: Model, dtype: int64
```