Aviation Incident Analysis for Safety Optimization: *Identifying Risk Factors to Improve Aircraft Safety*

PRESENTATION BY : LYDIAH KHISA

LINKEDIN: https://www.linkedin.com/in/lydia-khisa

Project Overview

- This project focuses on assessing aviation incident and accident data to identify patterns, trends, and risk factors that contribute to aircraft safety events. By checking into structured records which includes; year, location, aircraft type and model, engine type, amateur built, flight purpose, weather conditions and injury severity.
- The analysis aims to establish insights that can enhance aviation safety.
- The findings are intended to support improvements in policy, pilot training, aircraft design, and operational procedures.
- The key stakeholders include aviation authorities (e.g., FAA, KCAA), aircraft manufacturers, airlines, flight schools, safety analysts, data scientists, and regulators.
- The project operates within the aviation and aerospace industry, specifically in the domain of safety analytics and accident investigation, serving departments such as safety oversight, risk management, and data science.

Business Problem

Even with modern improvements in aviation technology, safety incidents still continue to occur across different types of aircraft and in various parts of the world. Stakeholders don't yet have a clear, data-based understanding of which factors contribute to serious accidents. This gap in insight risks misdirecting safety interventions and limiting their effectiveness.

The objective of this project is to:

1. Identify trends in incident severity over time and geography
2. Find out which aircraft models and engine types are most often involved
3. Assess the effect of weather changes on aviation incidences
4. Explore how different aircraft features relate to accident types
5. Determine the number of aviation incidences originating from amateur built aircrafts.

The overall goal is to offer useful insights that can improve safety rules and pilot training.

Success will be measured by the clear identification of high-risk aircraft models or regions, creation of easy-to-understand visuals that show trends, get safety teams to use the findings, and a potential long-term reduction in incident rates.

Data Description

Dataset Overview

- The dataset used in this project is sourced from aviation safety reports from the National Transportation Safety Board (NTSB).
- It contains structured information including the year of each incident, the location (city and country), the type of incident (Fatal, Non-Fatal, Substantial), aircraft type (e.g., airplane, helicopter), specific aircraft models (e.g., Cessna_172, Piper_PA-28), engine types (Reciprocating, Turboprop, Jet), and other identifiers such as registration numbers or codes, models, make.
- The data spans multiple formats, including categorical variables (like incident type and aircraft model), temporal data (year), and geographic data (location), making it suitable for comprehensive analysis of aviation safety trends.
- The Data spans from 1962 -2023 with a total of 90,348 entries (31 columns)

CONT'

Data Preview

- Import pandas as pd

```
df = pd.read_csv("c:/Users/User/dsc-phase-1-project v3/data/aviation_data.csv")
```

Set low_memory=False - to tell pandas to read the file

```
df = pd.read_csv("c:/Users/User/dsc-phase-1-project-v3/data/aviation_data.csv", low_memory=False)
```

Treating columns with different data types as strings to avoid errors and ensures consistent data types for analysis and visualization

```
df = pd.read_csv("c:/Users/User/dsc-phase-1-project-v3/data/aviation_data.csv", dtype={
    'column_name_6': str,
    'column_name_7': str,
    'column_name_28': str
},
low_memory=False
)
```

CONT'

Understanding the dataframe columns (index)

```
print(df.columns.tolist())
```

Summary of the columns

To give the total entries and number of columns of the dataset

```
df.info()
```

To view the the first 5 rows of the data:

```
df.head()
```

To view the the last 5 rows of the data:

```
df.head()
```

Checking duplicated data

```
df.duplicated().sum()
```

Data Cleaning

The data will be cleaned and standardized by handling missing values, correcting inconsistent text formats, removing duplicates and ensuring uniform data types across key columns.

This is aimed at improving data quality, reduce ambiguity and enable reliable analysis of incident patterns and severity.

Data Preparation

- Standardizing column names
- Handling Duplicates
- Handling missing values

CONT’

1. Standardizing Column Names

Standardizing columns

```
aviation = pd.read_csv("aviation_data.csv", low_memory=False)
aviation['Weather.Condition'] =aviation['Weather.Condition'].str.title().str.strip()
aviation['Injury.Severity'] = aviation['Injury.Severity'].str.title().str.strip()
```

Clean column names

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

Clean location column

```
df['Country'] = df['Country'].str.strip().str.upper()
df['Location'] = df['Location'].str.strip().str.upper()
```

CONT’

Convert date column

```
df['event.date'] = pd.to_datetime(df['event.date'], errors='coerce’)
df['year'] = df['event.date'].dt.year
```

2. **Handling Duplicate Data**

Checking for duplicates

```
duplicates = df[df.duplicated()]
print(len(duplicates))
duplicates.head()
```

CONT’

Checking for duplicate rows based on Event Id subset

```
duplicates = df[df.duplicated(subset = 'Event.Id’)]
print(len(duplicates))
duplicates.tail()
```

Checking for duplicate rows based on Accident Number sub set

```
duplicates = df[df.duplicated(subset = 'Accident.Number’)]
print(len(duplicates))
duplicates.tail()
```

Drop duplicates

```
df.drop_duplicates(inplace=True)
```

CONT'

- 3. **Handling Missing Values**

This is targeting a few columns; Aircraft Model, Engine Type and Injury Severity.

- **Aircraft Model**

Replacing the missing value 'NaN' with the word 'Unknown' . This keeps the data clean and consistent label, keeping the row data hence preventing errors during modelling.

```
df['Model'] = df['Model'].fillna('Unknown')
```

```
df.fillna({'model': 'Unknown'}, inplace=True)
```

Engine Type

Replacing the missing value 'NaN' with the most common value 'Mode'. This fills the missing values with the most common category, keeping the column consistent hence improves model performance

```
df['engine.type'] = df['engine.type'].fillna(df['engine.type'].mode()[0])
```

CONT’

- **Injury Severity**

Replacing the missing value 'NaN' with the word 'Substantial'. Substantial is a reasonable word to describe the severity of the accident. This prevents data loss, hence consistency in reporting.

```
df['injury.severity'] = df['injury.severity'].fillna('Substantial’)
```

Data Analysis

The project seeks to find out;

- How have aviation incident numbers changed over time?
- Which aircraft models are most frequently involved in fatal incidents?
- Are certain engine types more likely to result in severe outcomes?
- Which countries or regions report the highest number of aviation incidents?
- Is there a correlation between adverse weather conditions and the severity of aviation accidents?
- How does aircraft type relate to incident severity?
- What proportion of aviation incidents involve amateur-built aircraft?

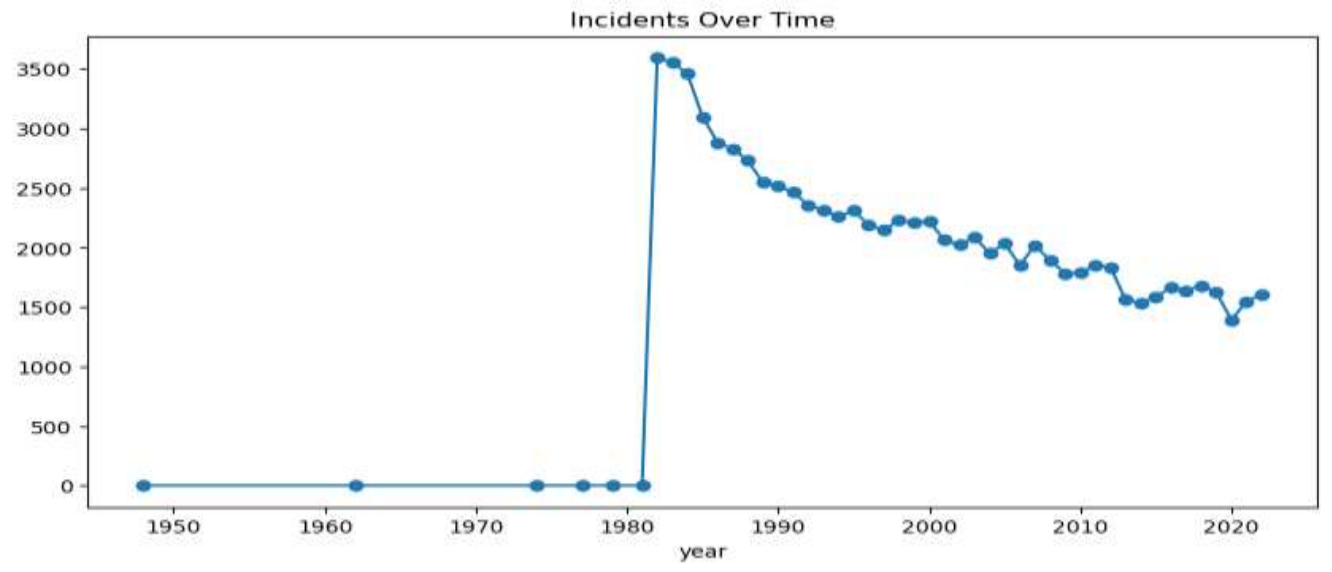
To answer these, the project uses visual tools such as time series plots to show trends over the years, bar charts to highlight high-risk aircraft models, heatmaps to explore severity by engine type, geographic maps to visualize incident density across regions, and pie charts to illustrate the distribution of incident types. These visualizations help communicate insights clearly and support data-driven decision-making for aviation safety.

CONT'

1. **Incident Trends Over Time**

incident_trend = df.groupby('year').size()

- incident_trend.plot(kind='line', marker='o', figsize=(10, 5), title='Incidents Over Time')

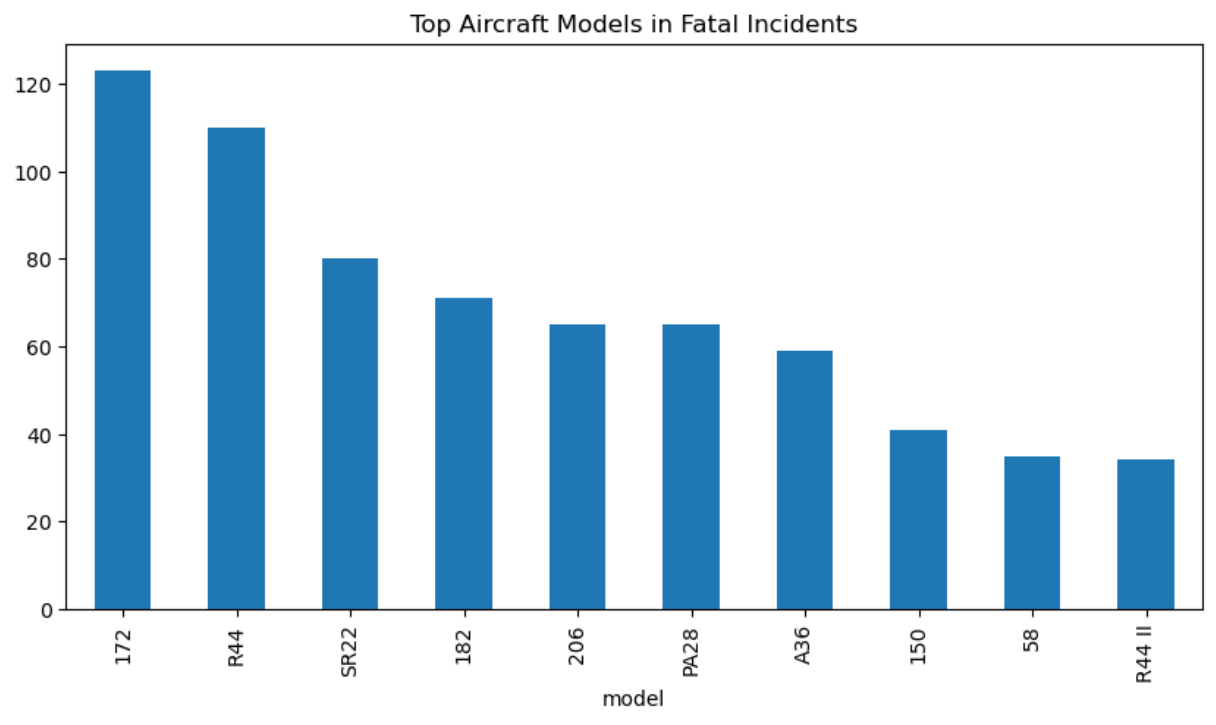


CONT’

2. Top Aircraft Models in Fatal Incidents

```
df[df['injury.severity'] == 'fatal’]

df['injury.severity'].unique()

fatal_models = df[df['injury.severity'].str.lower() == 'fatal']['model'].value_counts().head(10)

if not fatal_models.empty:

fatal_models.plot(kind='bar', figsize=(10, 5), title='Top Aircraft Models in Fatal Incidents’)

else:

print("No fatal incidents found in the dataset.")
```

CONT'


Top Aircraft Models in Fatal Incidents
120
100
80
60
40
20
0
172
R44
SR22
182
206
PA28
A36
150
58
R44 II
model

CONT'

3. Engine Type vs Severity

```
pip install seaborn
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import pandas as pd

df = pd.read_csv("c:/Users/User/dsc-phase-1-project-v3/data/aviation_data.csv")
plt.figure(figsize=(10, 5))
sns.countplot(data=df, x='Engine.Type', hue='Injury.Severity')
plt.title('Incident Severity by Engine Type')
plt.figure(figsize=(8, 4))  # Width = 8 inches, Height = 4 inches
```


Incident Severity by Engine Type
50000
40000
30000
20000
10000
0
Engine.Type
NONE
UNK

CONT'

4. Region and the fatal incidences

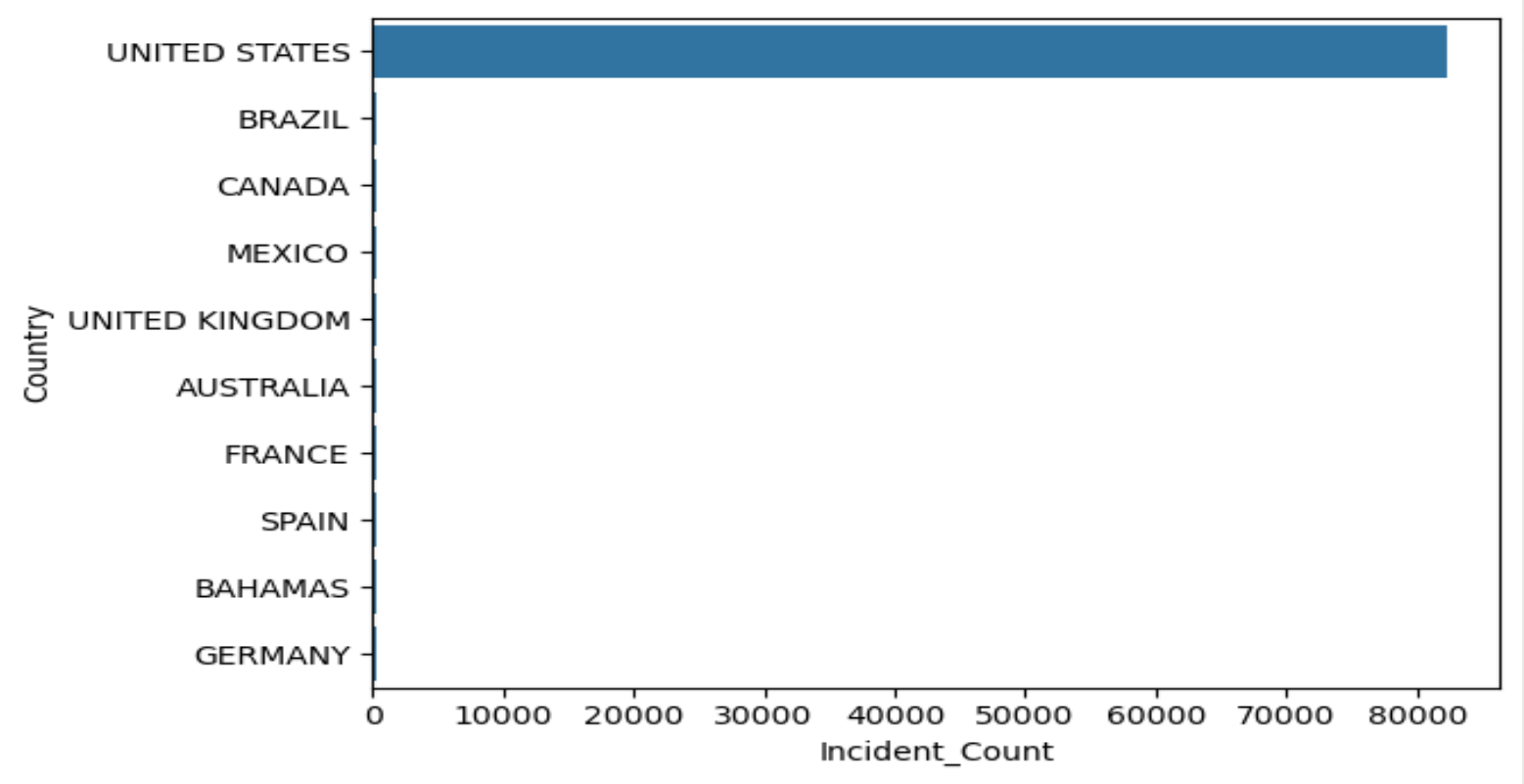
Count incidents by country

```
country_counts = df['Country'].value_counts().reset_index()
country_counts.columns = ['Country', 'Incident_Count']
```

Top 10 countries by incident count

```
top_countries = country_counts.head(10)
sns.barplot(data=top_countries, x='Incident_Count', y='Country')
print(country_counts.head())
```

CONT'


UNITED STATES
BRAZIL
CANADA
MEXICO
UNITED KINGDOM
AUSTRALIA
FRANCE
SPAIN
BAHAMAS
GERMANY
Country
0
10000
20000
30000
40000
50000
60000
70000
80000
Incident_Count

CONT'

5. **A relationship between aircraft Category and incident severity**

Create aircraft type by combining Make and Model

- df['Aircraft_Type'] = df['Make'] + " " + df['Model']

Count incidents by aircraft type and severity

- type_severity_counts = df.groupby(['Aircraft.Category', 'Injury.Severity']).size().reset_index(name='Incident_Count')

pivot_df = type_severity_counts.pivot(index='Aircraft.Category', columns='Injury.Severity', values='Incident_Count').fillna(0)

Focus on top 10 aircraft types by total incidents

- top_types = pivot_df.sum(axis=1).sort_values(ascending=False).head(10).index

  filtered_df = pivot_df.loc[top_types]
- filtered_df.plot.bar(stacked=True, colormap='Set2')

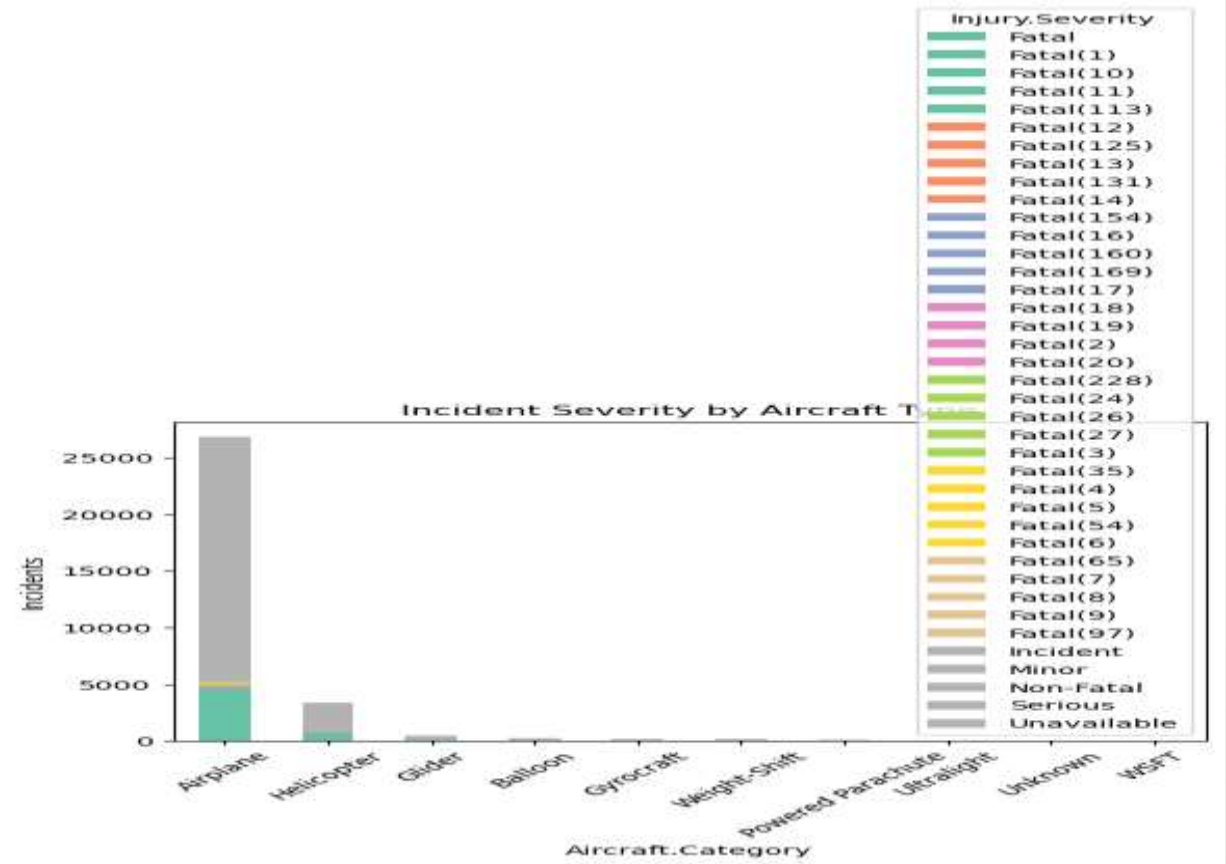
  plt.title("Incident Severity by Aircraft Type")

  plt.ylabel("Incidents")

  plt.xticks(rotation=45)

  plt.show()

CO


Injury.Severity
Fatal
Fatal(1)
Fatal(10)
Fatal(11)
Fatal(113)
Fatal(12)
Fatal(125)
Fatal(13)
Fatal(131)
Fatal(14)
Fatal(154)
Fatal(16)
Fatal(160)
Fatal(169)
Fatal(17)
Fatal(18)
Fatal(19)
Fatal(2)
Fatal(20)
Fatal(228)
Fatal(24)
Fatal(26)
Fatal(27)
Fatal(3)
Fatal(35)
Fatal(4)
Fatal(5)
Fatal(54)
Fatal(6)
Fatal(65)
Fatal(7)
Fatal(8)
Fatal(9)
Fatal(97)
Incident
Minor
Non-Fatal
Serious
Unavailable
Incident Severity by Aircraft T
Incidents
25000
20000
15000
10000
5000
0
Airplane
Helicopter
Glider
Balloon
Gyrocraft
Weight-Shift
Powered Parachute
Ultralight
Unknown
WSFT
Aircraft.Category

CONT’

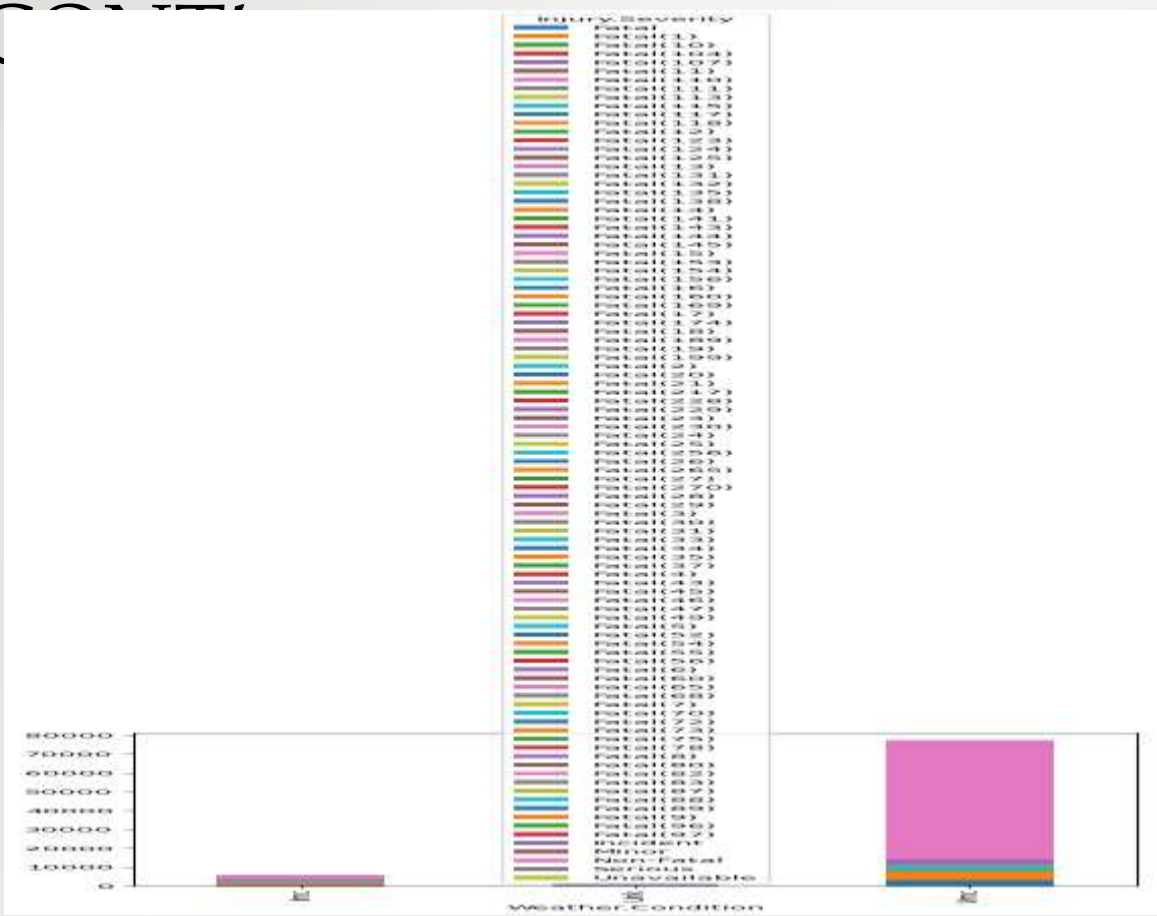
6. **The correlation between weather patterns and incident severity**

Groupby

- weather_severity = aviation.groupby(['Weather.Condition', 'Injury.Severity']).size().reset_index(name='Incident_Count')
- pivot_df = weather_severity.pivot(index='Weather.Condition', columns='Injury.Severity', values='Incident_Count').fillna(0)

```
pivot_df.plot(kind='bar', stacked=True)

plt.show()
```

CONT'


Injury_Severity
Fatal
Serious
Uninjurable
Weather.condition

CONT

7. **Aviation incidences in relation to amateur built aircrafts**

Standardize the 'Amateur_Built' column for consistent analysis

```
aviation['Amateur.Built'] = aviation['Amateur.Built'].str.title().str.strip()
```

Incidence Counts

```
build_counts = aviation['Amateur.Built'].value_counts()
print(build_counts)
```

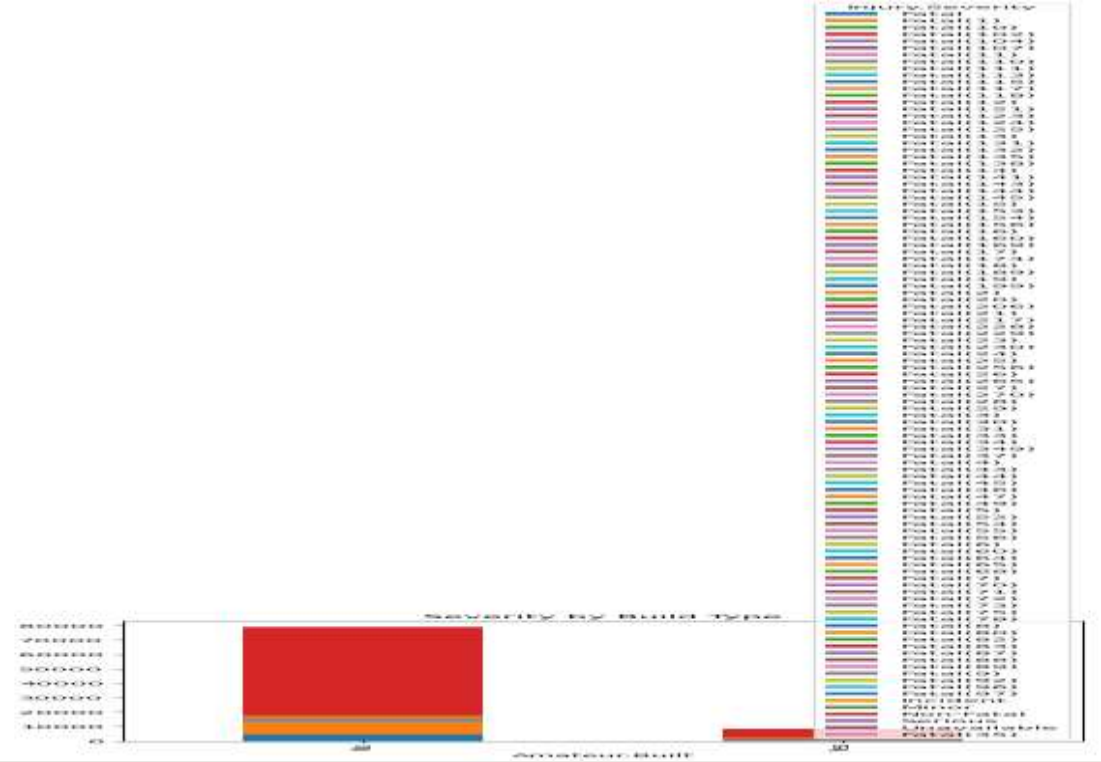
Group data by build type and injury severity, then count incidents

```
severity_by_build = aviation.groupby(['Amateur.Built', 'Injury.Severity']).size().unstack().fillna(0)
print(severity_by_build)
```

Create a stacked bar chart to visualize severity across build types

```
severity_by_build.plot(kind='bar', stacked=True)
plt.title("Severity by Build Type")
plt.show()
```

CONT'



Conclusion

Time Trends

- Incident frequency shows seasonal variation, with peaks during months of increased flight activity (e.g., summer and holiday seasons).
- Long-term trends suggest a gradual decline in total incidents, possibly due to improved safety protocols and technology—but fatal incidents persist, especially in general aviation.

Aircraft Models

- A small number of high-usage models (e.g., Cessna 172, Piper PA-28) account for a large share of incidents.
- These models are often used in training and private flights, which may correlate with less experienced pilots or less stringent maintenance oversight.

CONT'

Engine Types

- Single-engine aircraft are disproportionately represented in fatal and serious incidents.
- Turboprop and jet engines show fewer incidents per flight hour, suggesting better performance under stress and more robust safety systems.

Aircraft Categories

- Airplanes dominate the dataset, but helicopters and experimental aircraft show higher severity rates when incidents occur.
- Amateur-built aircraft have elevated risk profiles, often linked to mechanical failure or pilot error.

These insights can guide targeted safety audits, training programs, and engine modernization efforts.

Recommendations

Time-Based Safety Interventions

- Increase seasonal safety campaigns during high-traffic period, targeting private and recreational pilots.
- Use historical incident data to forecast risky periods and allocate inspection resources accordingly.

Model-Specific Oversight

- Conduct targeted audits and maintenance reviews for frequently involved models like the Cessna 172.
- Encourage manufacturers to analyze incident data and improve design or training materials for high-risk models.

Engine-Type Risk Mitigation

- Mandate additional training for pilots operating single-engine aircraft, especially in adverse weather conditions.
- Promote engine redundancy upgrades or enhanced emergency protocols for older single-engine fleets.

Aircraft Category Focus

- Require stricter certification and inspection for amateur-built and experimental aircraft.
- Develop category-specific safety dashboards for regulators to monitor trends and intervene early.

Note

How to run

- Open the notebook in Jupyter and run all cells top to bottom. No external dependencies beyond pandas, seaborn, and matplotlib.