1. Business Understanding.

This project focuses on analyzing aviation accident data to uncover the key factors contributing to these incidents. Our dataset includes critical information such as aircraft make and model, weather conditions.

Dataset Overview

- Aircraft Make and Model: Details of the aircraft involved.
- Weather Conditions: Insights into the weather during accidents.

Exploratory Data Analysis (EDA)

- Aircraft Analysis: Investigate patterns in accident distributions among aircraft types.
- Weather Analysis: Examine the impact of weather on accidents to identify trends.

Key Steps

1. Utilize factors such as aircraft type and weather for clustering.
2. Explore incidents for actionable insights.

Goal The goal is to reveal patterns in aviation accidents and provide recommendations to enhance safety. By using EDA we aim to significantly contribute to aviation safety efforts.

In [255…

```
# Import Libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
from scipy.stats import linregress
%matplotlib inline
```

In [256…

```
df = pd.read_csv("AviationData.csv", encoding="latin1")
```

```
/var/folders/77/0jwr1_4d5p1g1g_rb4gqchhh0000gn/T/ipykernel_12910/20481393
34.py:1: DtypeWarning: Columns (6,7,28) have mixed types. Specify dtype o
ption on import or set low_memory=False.
  df = pd.read_csv("AviationData.csv", encoding="latin1")
```

The data is encoded in Latin. Used the above code to identify.

2.Data Understanding

In [257...
```
df.head()
```

Out[257]:

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

5 rows × 31 columns

In [258...
```
df.columns
```

Out[258]:
```
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.Descrip
tion',
       'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Inju
ries',
       'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjure
d',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

In [259...
```
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
```

Data Cleaning

Identifying missing values

In [210…

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

Out[210]:
```
Event.Id                       0
Investigation.Type             0
Accident.Number                0
Event.Date                     0
Location                      52
Country                      226
Latitude                   54507
Longitude                  54516
Airport.Code               38640
Airport.Name               36099
Injury.Severity             1000
Aircraft.damage             3194
Aircraft.Category          56602
Registration.Number         1317
Make                          63
Model                         92
Amateur.Built                102
Number.of.Engines           6084
Engine.Type                 7077
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               6381
Publication.Date           13771
dtype: int64
```

In [260…
```
df.shape
```

Out[260]:
```
(88889, 31)
```

In [261…
```
missing_columns = df.columns[df.isna().any()]
print(missing_columns)
```

```
Index(['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.Descript
ion',
       'Schedule', 'Purpose.of.flight', 'Air.carrier', 'Total.Fatal.Injur
ies',
       'Total.Serious.Injuries', 'Total.Minor.Injuries', 'Total.Uninjured
',
       'Weather.Condition', 'Broad.phase.of.flight', 'Report.Status',
       'Publication.Date'],
      dtype='object')
```

Dropping Missing values

In [262…
```
# Dropping the columns having more missing values
threshold = 0.3
drop_cols = df.columns[df.isna().mean() > threshold]
df.drop(columns=drop_cols, inplace=True)
df.shape
```

Out[262]:
```
(88889, 22)
```

In [263…
```
unique_values = pd.DataFrame({
    'Column': df.columns,
    'unique_val': [df[col].nunique() for col in df.columns],
    'missing_values': df.isna().sum().values
})
unique_values
```

Out[263]:

	Column	unique_val	missing_values
0	Event.Id	87951	0
1	Investigation.Type	2	0
2	Accident.Number	88863	0
3	Event.Date	14782	0
4	Location	27758	52
5	Country	219	226
6	Injury.Severity	109	1000
7	Aircraft.damage	4	3194
8	Registration.Number	79105	1317
9	Make	8237	63
10	Model	12318	92
11	Amateur.Built	2	102
12	Number.of.Engines	7	6084
13	Engine.Type	13	7077
14	Purpose.of.flight	26	6192
15	Total.Fatal.Injuries	125	11401
16	Total.Serious.Injuries	50	12510
17	Total.Minor.Injuries	57	11933
18	Total.Uninjured	379	5912
19	Weather.Condition	4	4492
20	Report.Status	17075	6381
21	Publication.Date	2924	13771

In [264…
```
# Dropping rows that have missing values
subset_cols = unique_values.loc[unique_values['missing_values'] > 1000, '
df.dropna(subset=subset_cols, inplace=True)
df.shape
```

Out[264]:
```
(51339, 22)
```

In [265…
```
df.isna().sum()
```

Out[265]:
```
Event.Id                    0
Investigation.Type          0
Accident.Number             0
Event.Date                  0
Location                   11
Country                   145
Injury.Severity            11
Aircraft.damage             0
Registration.Number         0
Make                        8
Model                      21
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
Report.Status               0
Publication.Date            0
dtype: int64
```

In [266…
```
# Handling null values

df[['Location', 'Country', 'Injury.Severity', 'Model', 'Make']] = df[['Lo
df.isna().sum()
```

Out[266]:
```
Event.Id                  0
Investigation.Type        0
Accident.Number           0
Event.Date                0
Location                  0
Country                   0
Injury.Severity           0
Aircraft.damage           0
Registration.Number       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
Report.Status             0
Publication.Date          0
dtype: int64
```

Data Analysis

In [267…
```
# Identify top makes by accident count
top_makes = df['Make'].value_counts().head(5).index  # Top 5 makes
df_top_makes = df[df['Make'].isin(top_makes)]
```

In [268…
```
# Group by 'Make' and 'Model' to find accident counts
make_model_accidents = df_top_makes.groupby(['Make', 'Model']).size().res
make_model_accidents.sort_values(by='Accident_Count', ascending=False, in
```

Examine Key Factors

a) Engine type

In [269…
```
engine_type_analysis = df_top_makes.groupby(['Make', 'Engine.Type']).size
```

b)Weather

In [270…
```
weather_analysis = df_top_makes.groupby(['Make', 'Weather.Condition']).si
```

c) Location

In [271…
```
location_analysis = df_top_makes.groupby(['Make', 'Location']).size().res
```

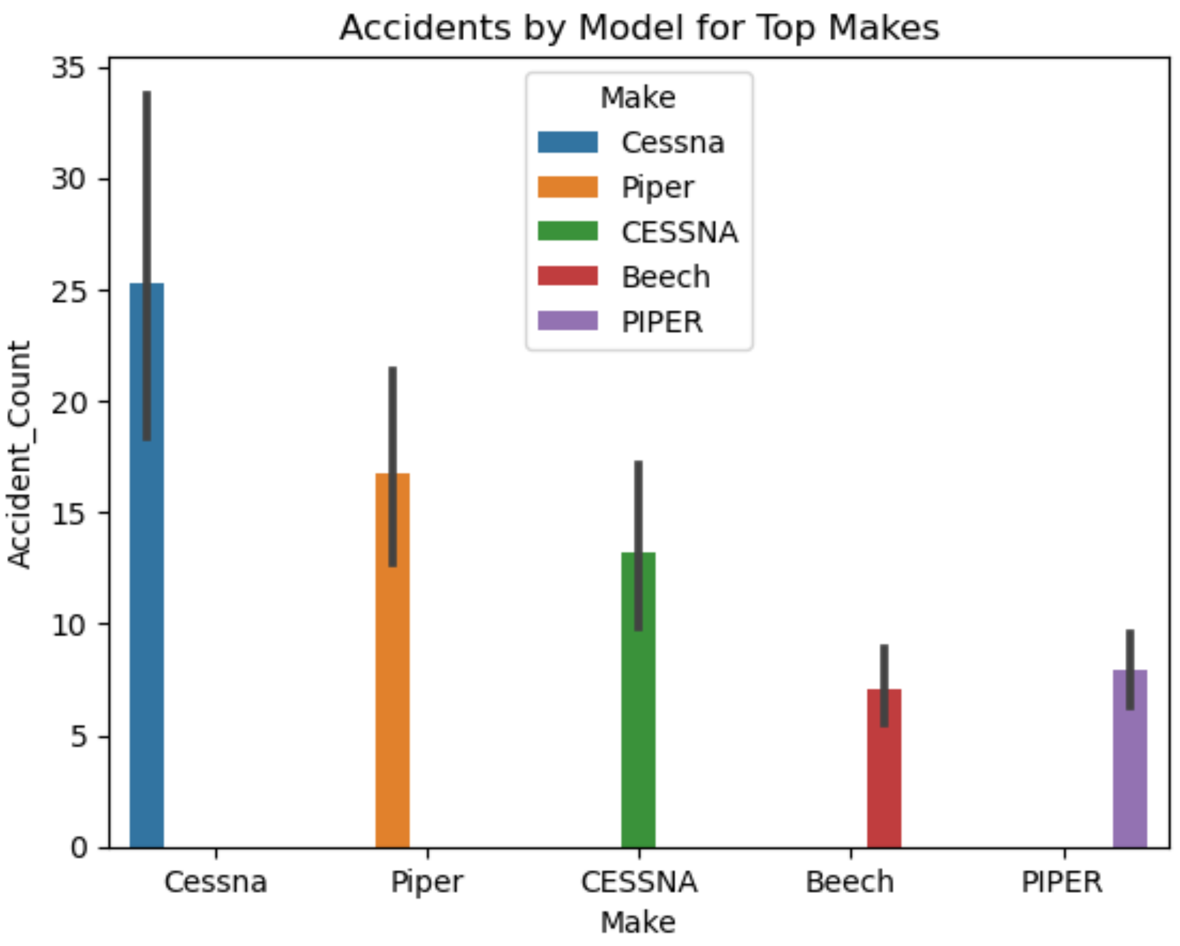
d) Amateur built status

In [272...

```
amateur_built_analysis = df_top_makes.groupby(['Make', 'Amateur.Built']).
```

In [273...

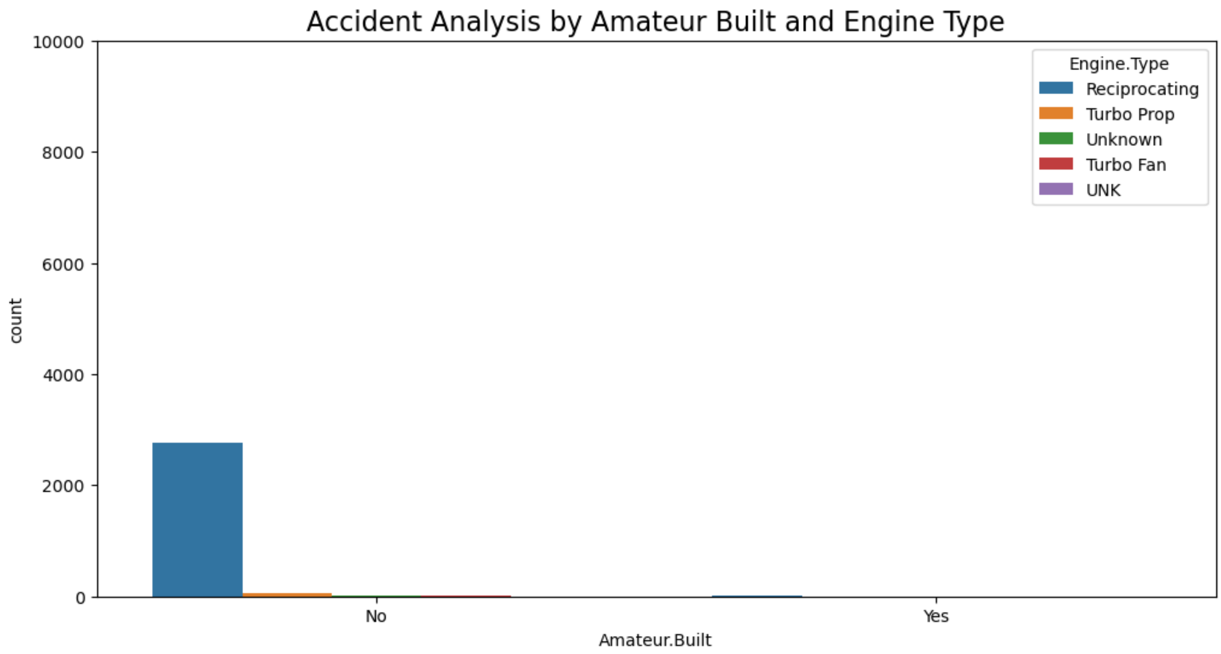
```
sns.barplot(data=make_model_accidents, x='Make', y='Accident_Count', hue=
plt.title('Accidents by Model for Top Makes')

plt.show()
```



From the above graph, Cessna Aircraft model has the highest number of accidents registered.

In [274...

```
plt.figure(figsize=(12, 6))
sns.countplot(data=df_top_makes.sample(frac=0.1, replace=True), x='Amateu
plt.title('Accident Analysis by Amateur Built and Engine Type', fontsize=
plt.ylim(0, 10000)
plt.show()
```

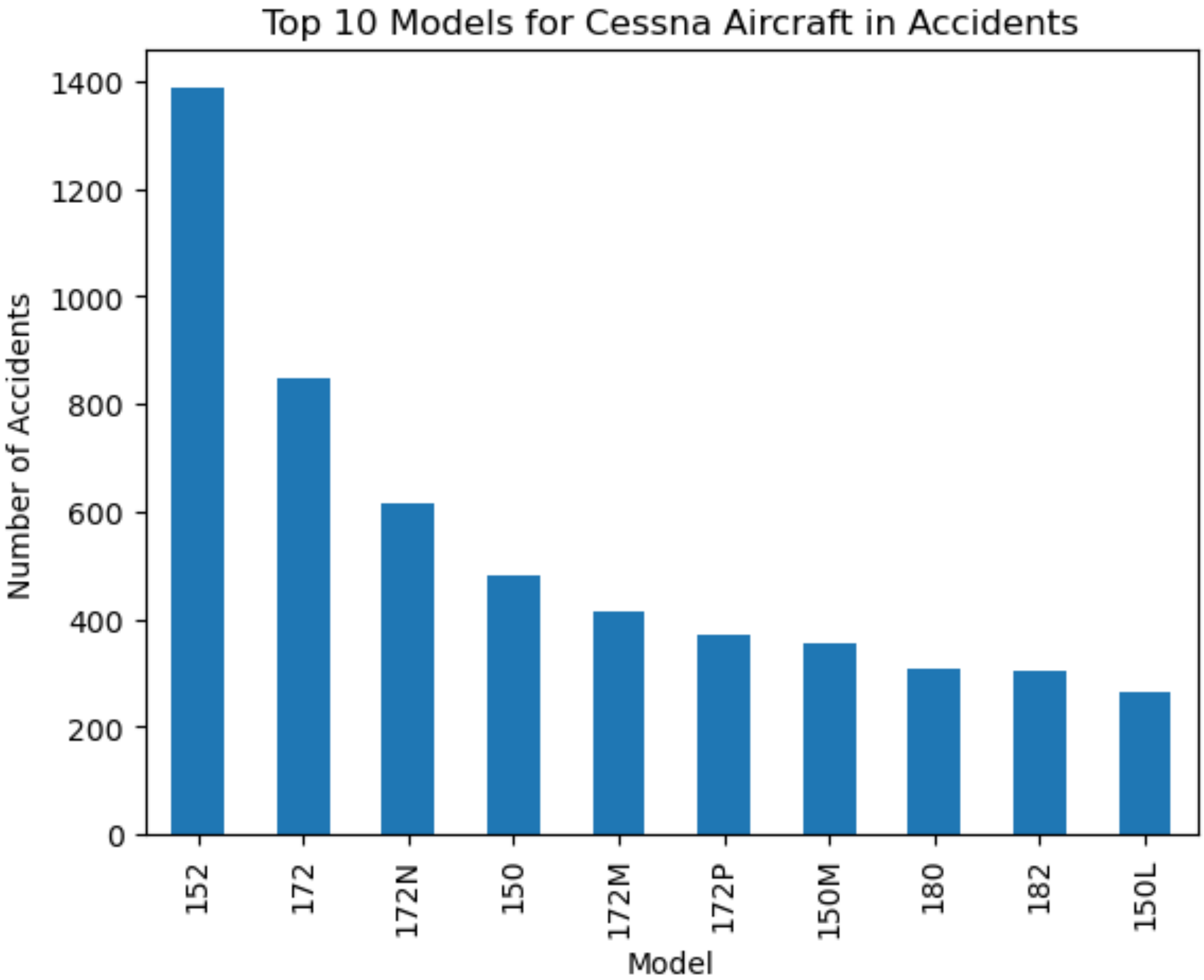


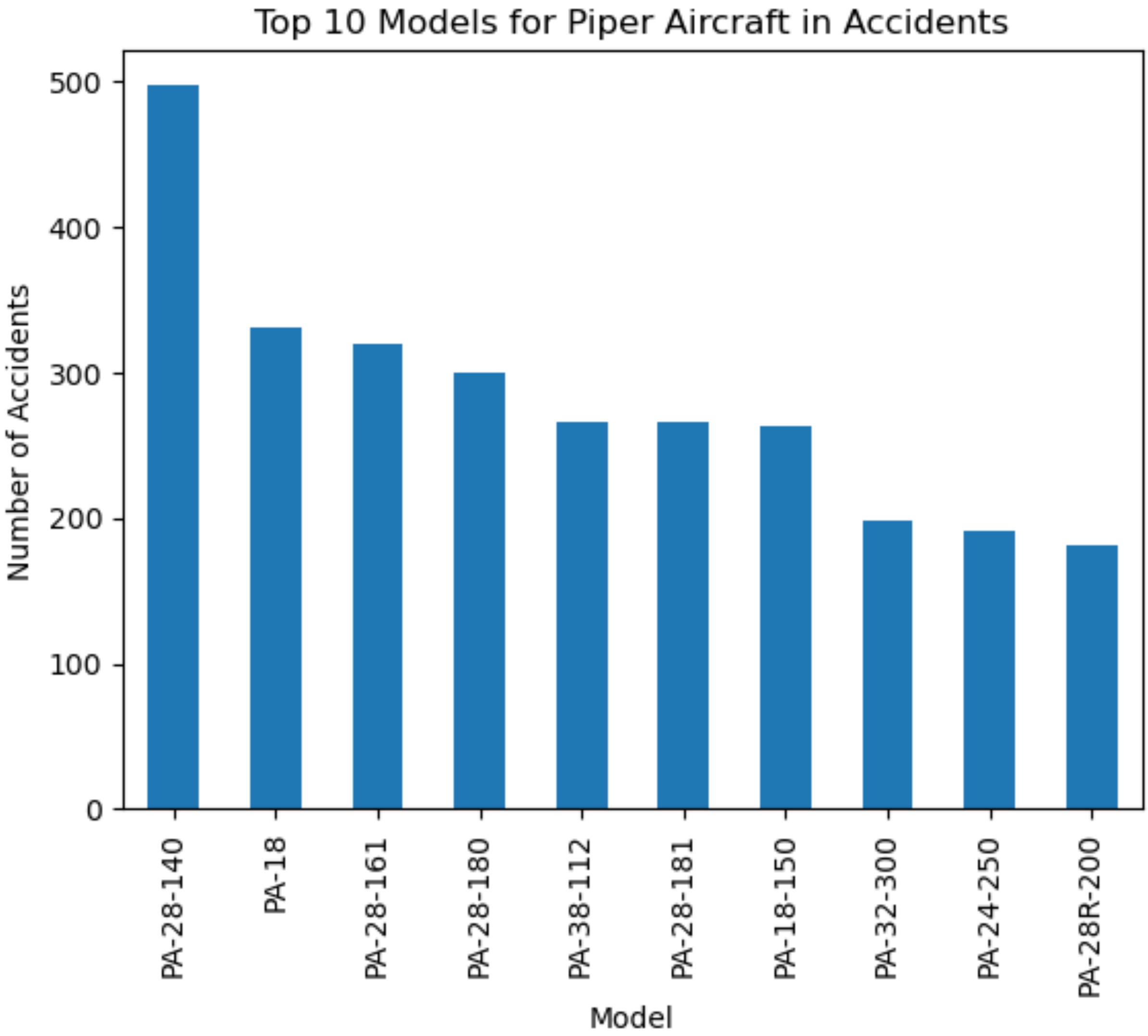
In [275...

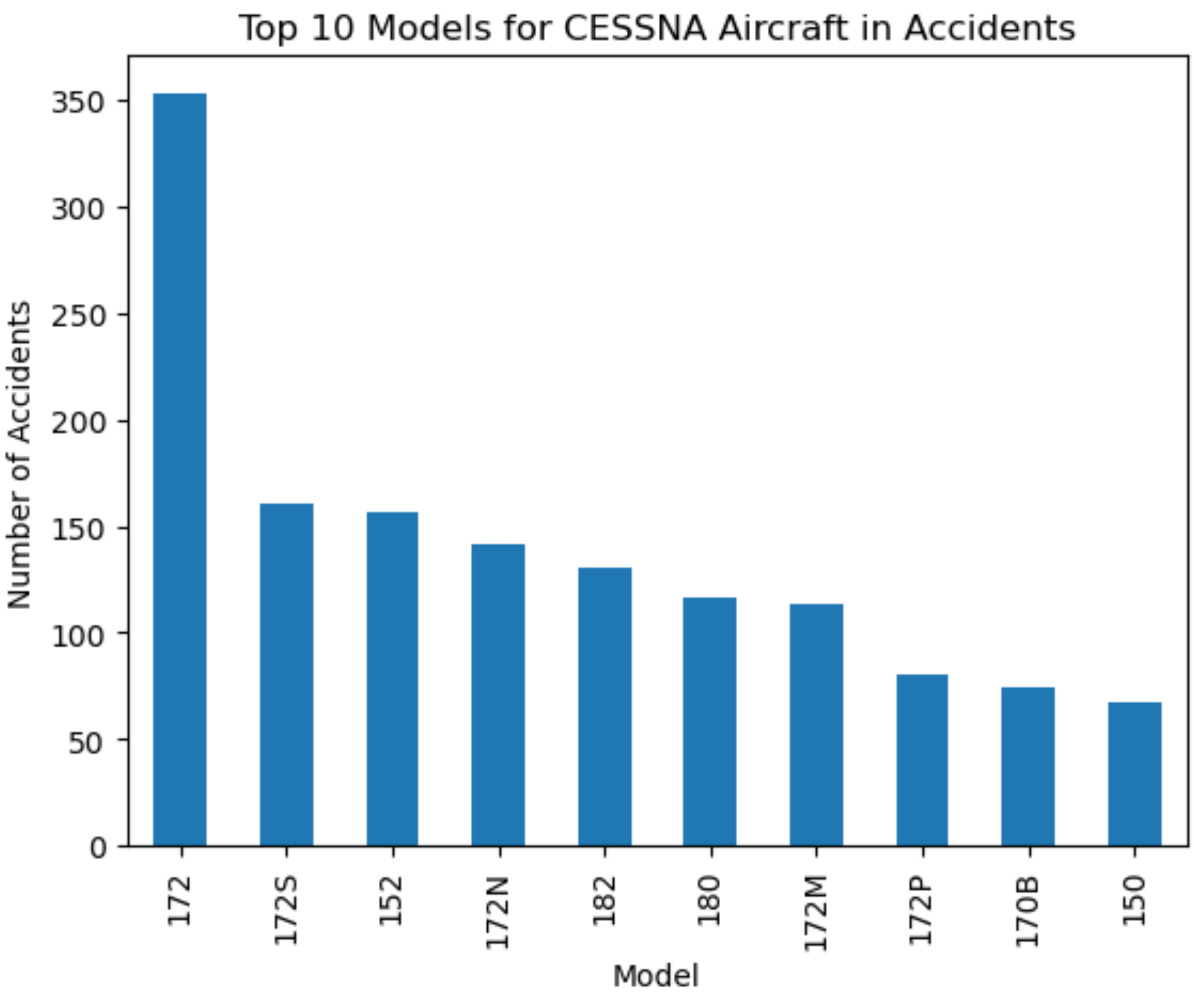
```
class AircraftAccidentVisualizer:
    def __init__(self, data):
        self.data = data

    def visualize_top_10_models_by_make(self):
        top_10_makes = self.data['Make'].value_counts().head(10).index
        for make in top_10_makes:
            make_data = self.data[self.data['Make'] == make]
            top_10_models = make_data['Model'].value_counts().head(10)
            top_10_models.plot(kind='bar', title=f'Top 10 Models for {mak
            plt.xlabel('Model')
            plt.ylabel('Number of Accidents')
            plt.show()

    def visualize_weather(self):
        weather_counts = self.data['Weather.Condition'].value_counts()
        weather_counts.plot(kind='bar', title='Weather Conditions in Airc
        plt.xlabel('Weather Conditions')
        plt.ylabel('Number of Accidents')
        plt.show()

    def visualize_engines(self):
        engine_weather_counts = self.data.groupby(['Engine.Type', 'Weathe
        plt.figure(figsize=(20,12))
        sns.barplot(x='Engine.Type', y='Count', hue='Weather.Condition',
        plt.xlabel('Type of Engine')
        plt.ylabel('Count of Engine')
        plt.title('Count of Engine Types by Weather Condition')
        plt.legend(title='Weather Condition', loc='upper right')
        plt.show()
```

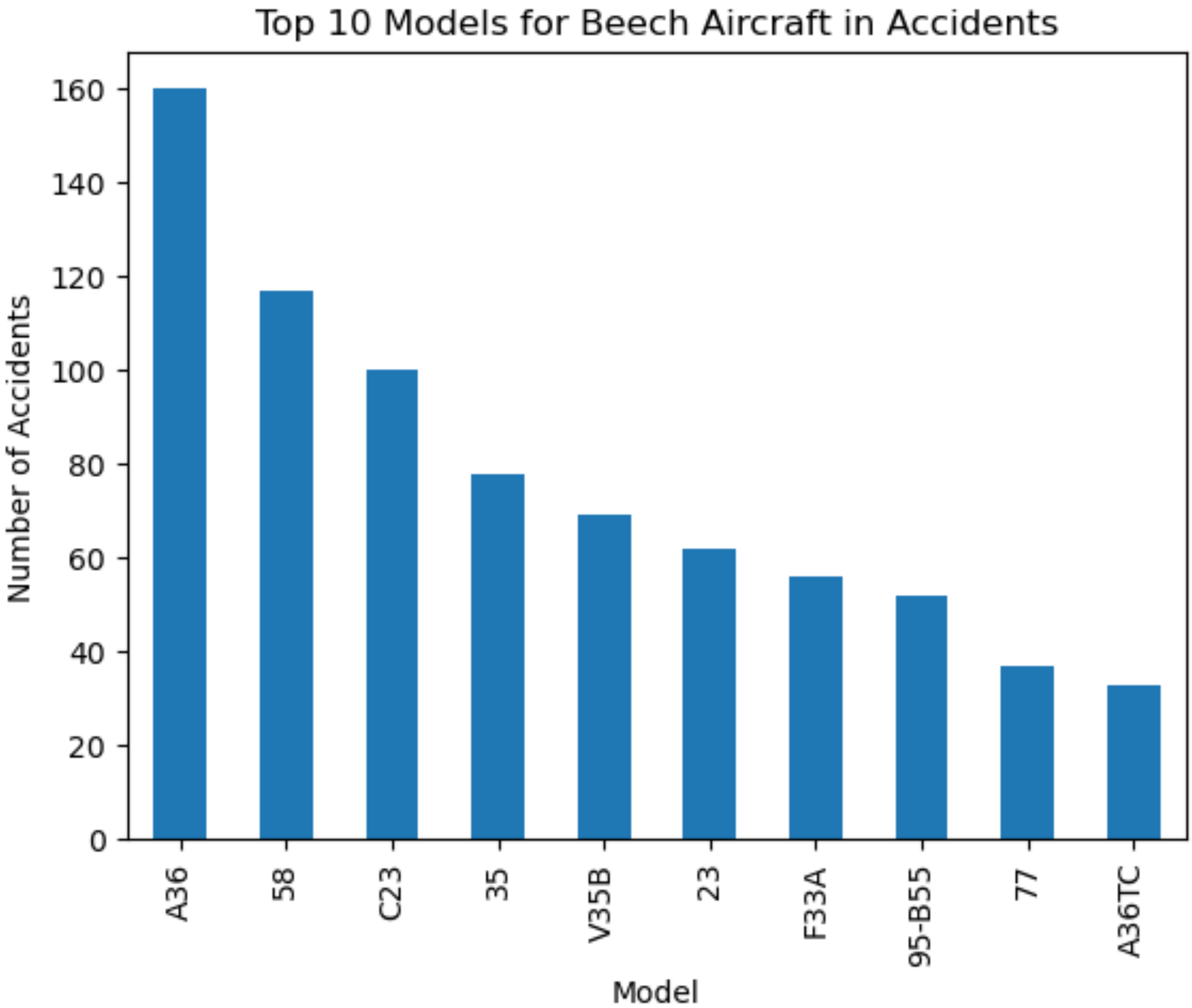
In [276...

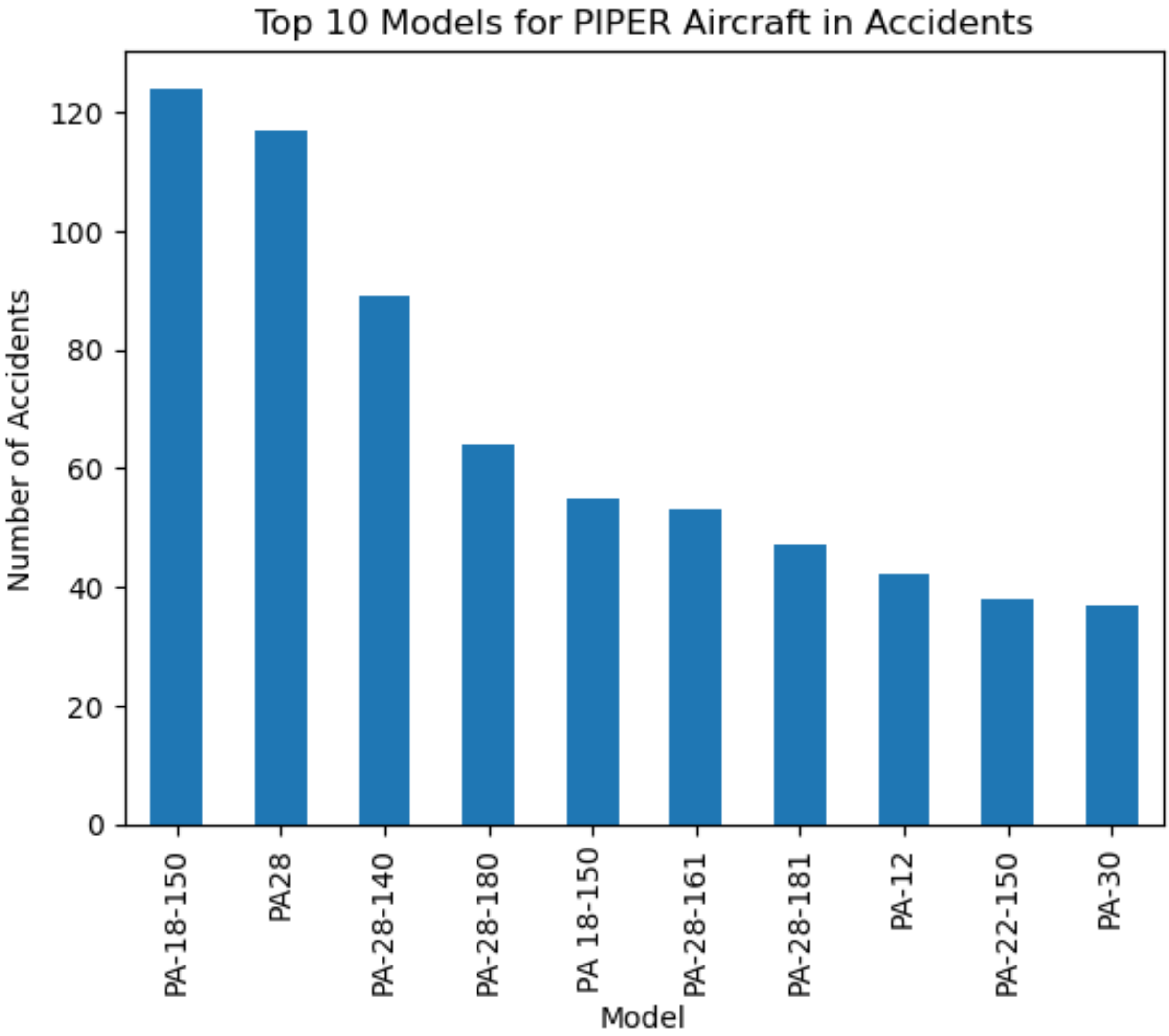
```
visualizer = AircraftAccidentVisualizer(df)
visualizer.visualize_top_10_models_by_make()
```

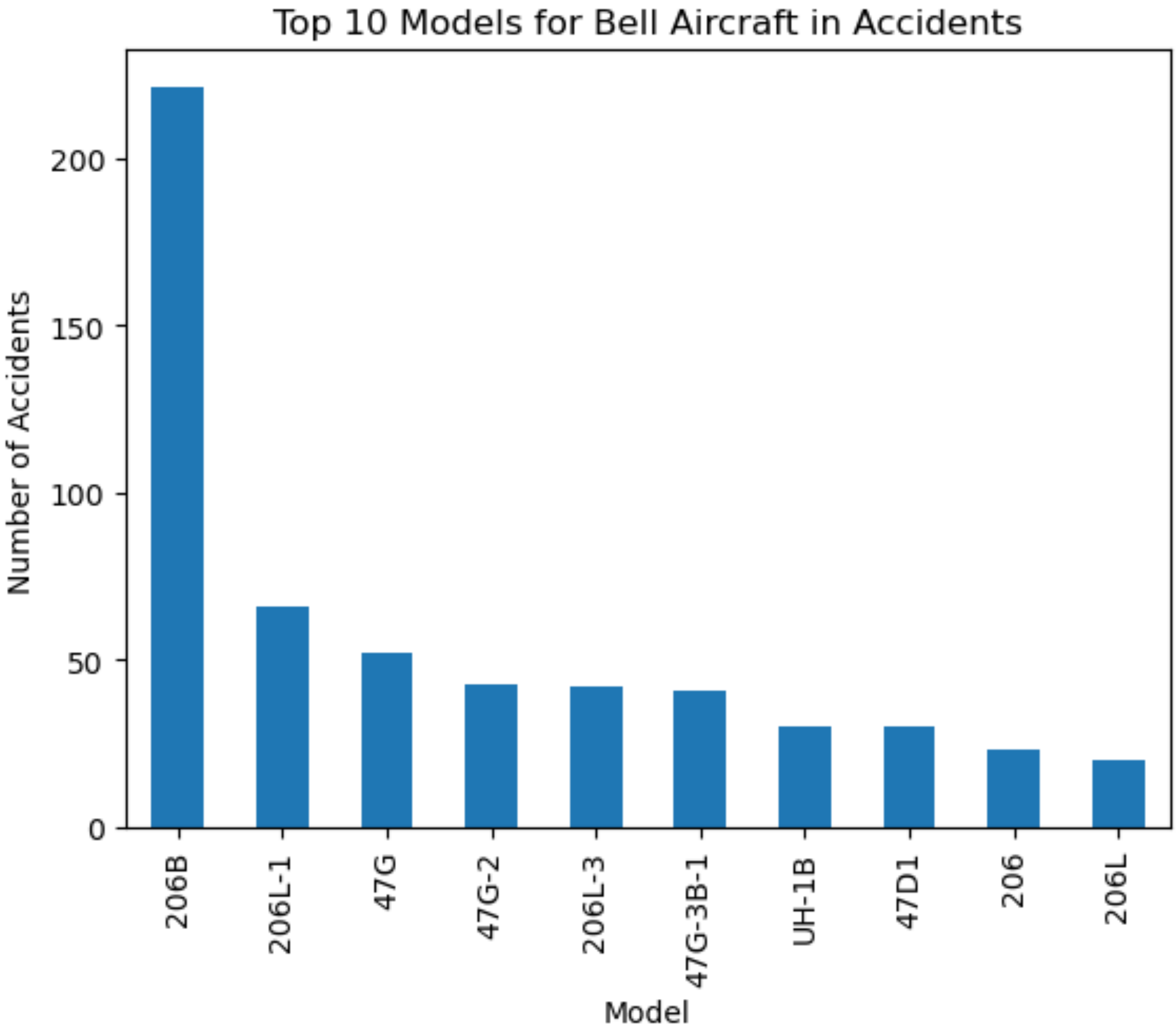

Top 10 Models for Cessna Aircraft in Accidents
Number of Accidents
0
200
400
600
800
1000
1200
1400
152
172
172N
150
172M
172P
150M
180
182
150L
Model

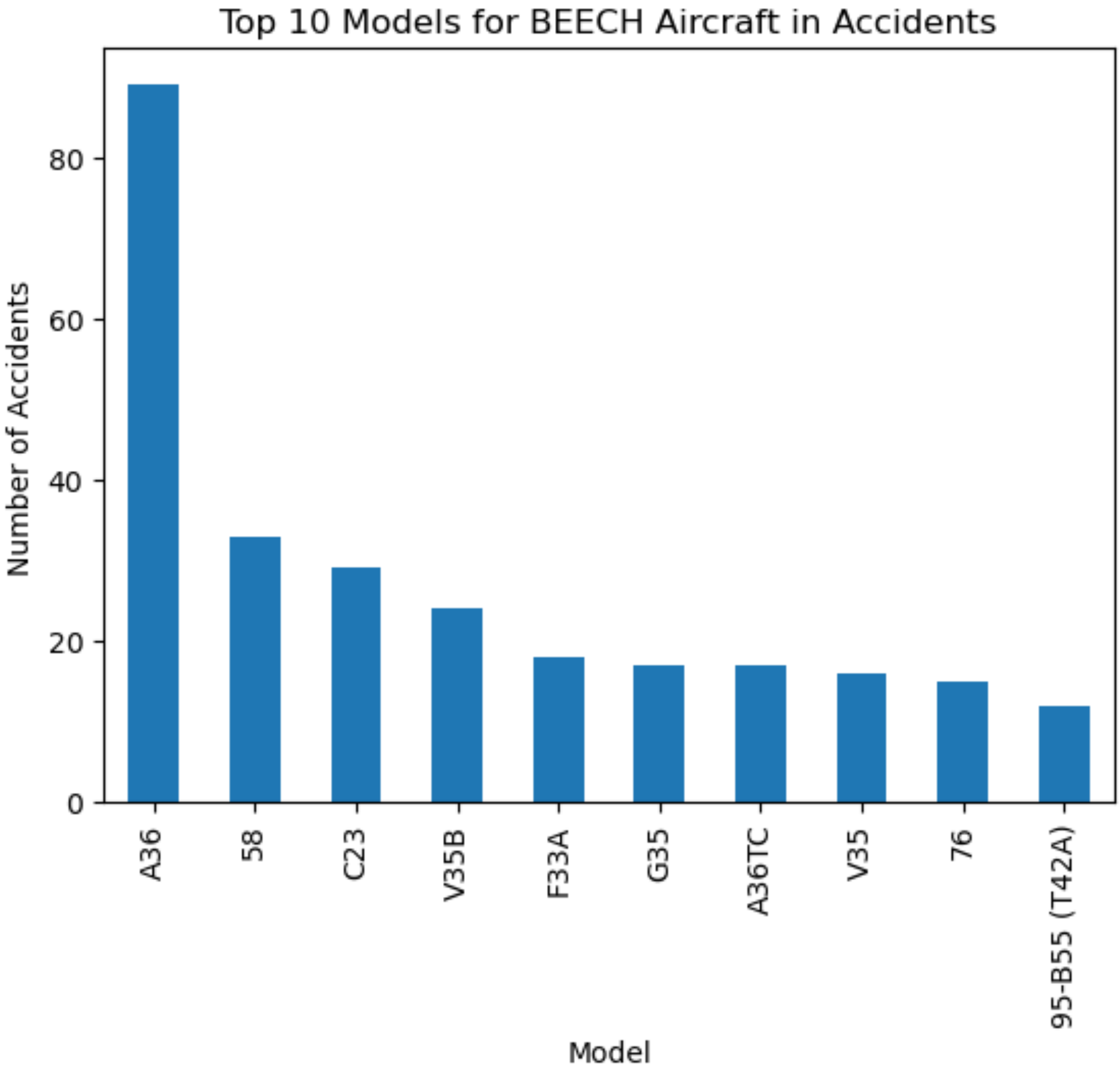

Top 10 Models for Piper Aircraft in Accidents
Number of Accidents
500
400
300
200
100
0
PA-28-140
PA-18
PA-28-161
PA-28-180
PA-38-112
PA-28-181
PA-18-150
PA-32-300
PA-24-250
PA-28R-200
Model

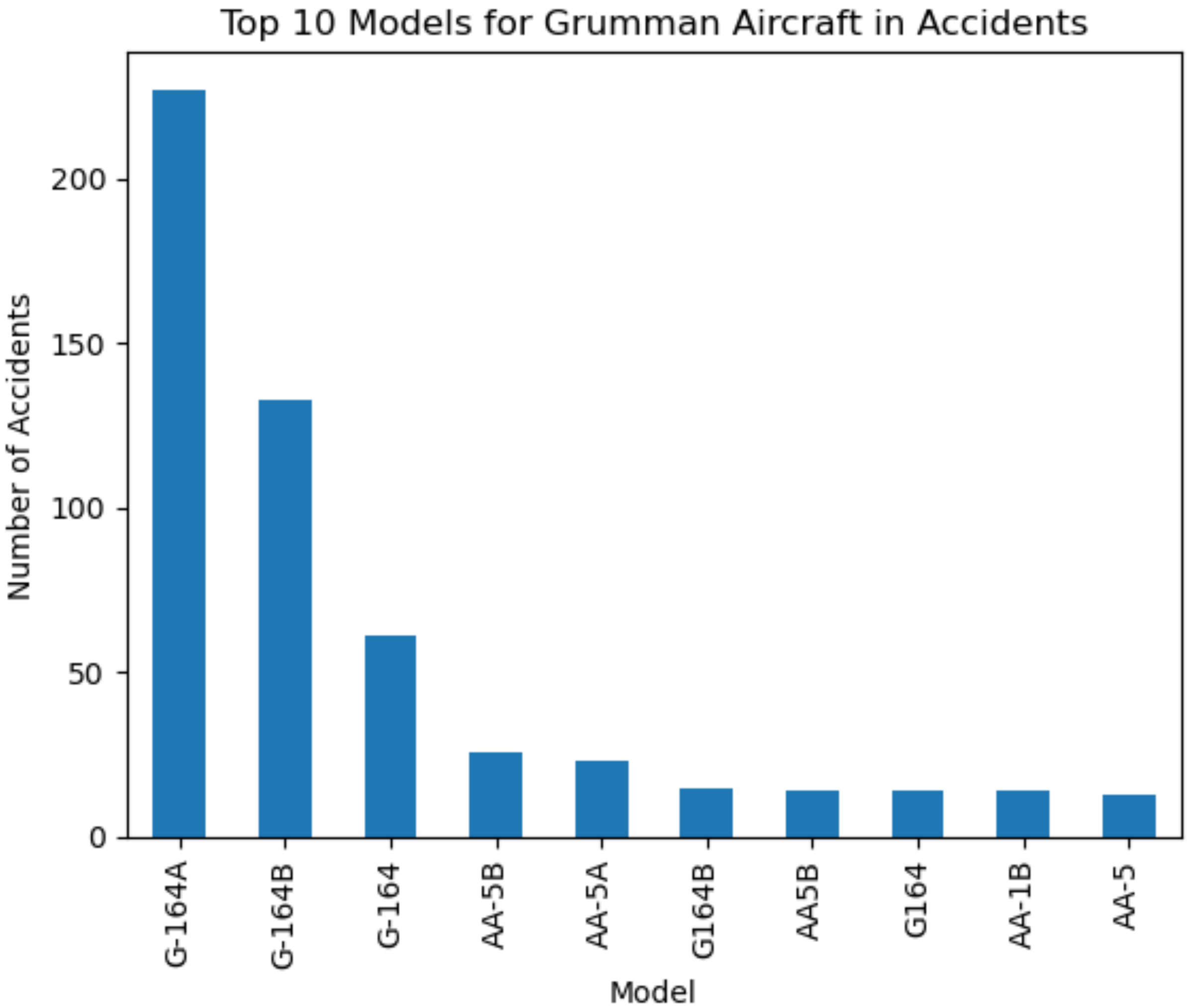

Top 10 Models for CESSNA Aircraft in Accidents
Number of Accidents
0
50
100
150
200
250
300
350
172
172S
152
172N
182
180
172M
172P
170B
150
Model

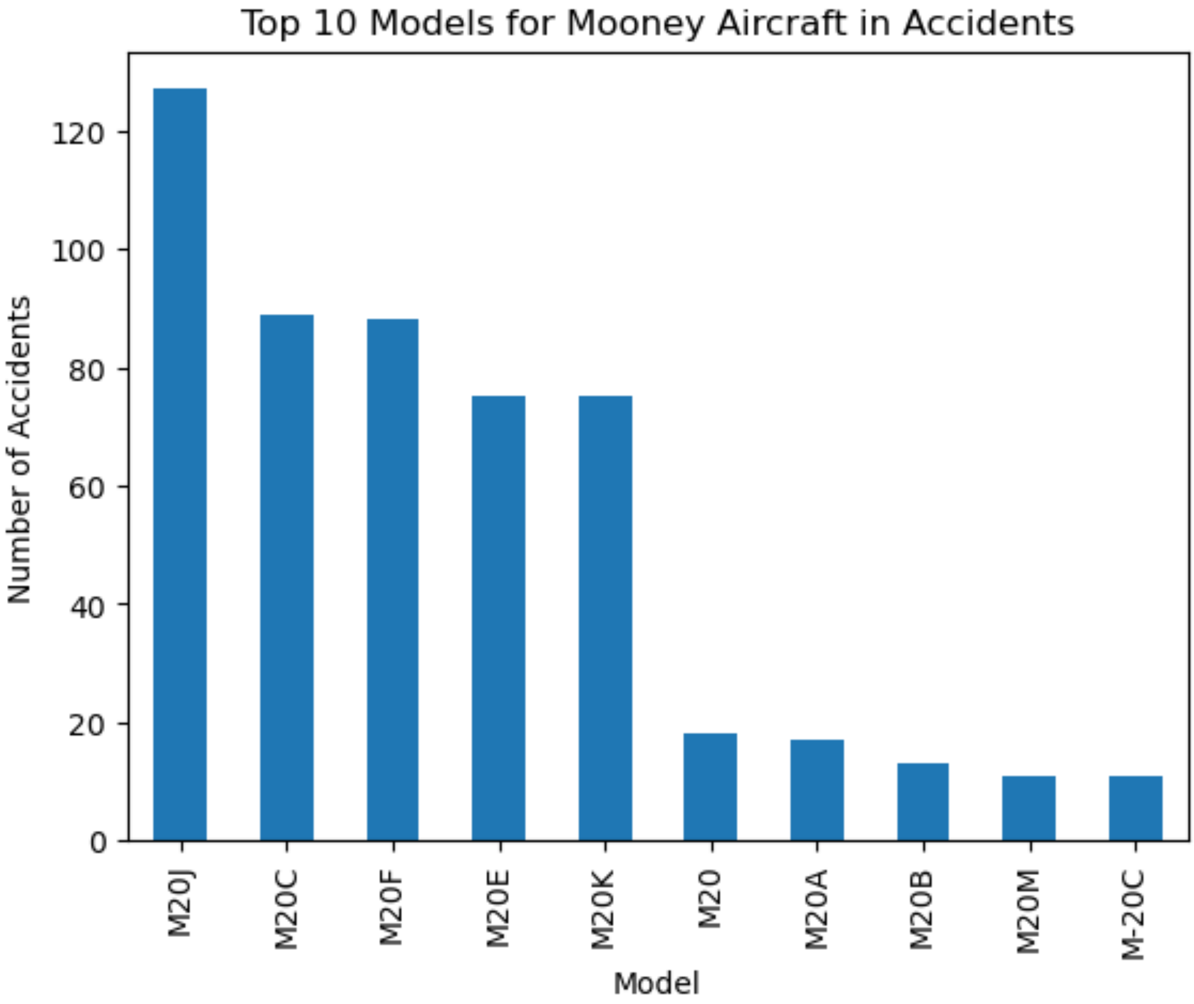

Top 10 Models for Beech Aircraft in Accidents
Number of Accidents
0
20
40
60
80
100
120
140
160
A36
58
C23
35
V35B
23
F33A
95-B55
77
A36TC
Model

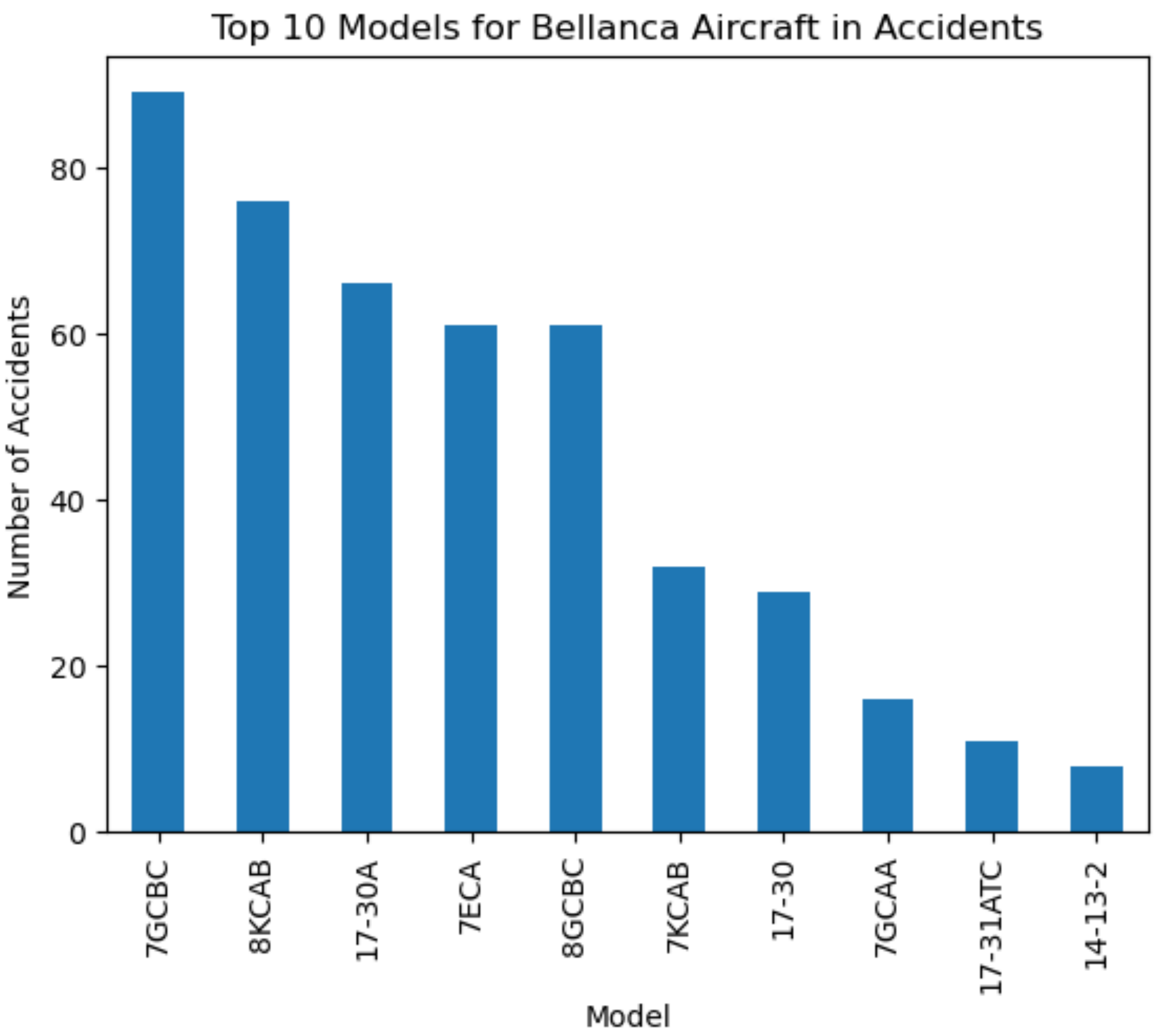

Top 10 Models for PIPER Aircraft in Accidents
Number of Accidents
0
20
40
60
80
100
120
PA-18-150
PA28
PA-28-140
PA-28-180
PA 18-150
PA-28-161
PA-28-181
PA-12
PA-22-150
PA-30
Model


Top 10 Models for Bell Aircraft in Accidents
Number of Accidents
0
50
100
150
200
206B
206L-1
47G
47G-2
206L-3
47G-3B-1
UH-1B
47D1
206
206L
Model


Top 10 Models for BEECH Aircraft in Accidents
Number of Accidents
0
20
40
60
80
A36
58
C23
V35B
F33A
G35
A36TC
V35
76
95-B55 (T42A)
Model


Top 10 Models for Grumman Aircraft in Accidents
Number of Accidents
0
50
100
150
200
G-164A
G-164B
G-164
AA-5B
AA-5A
G164B
AA5B
G164
AA-1B
AA-5
Model

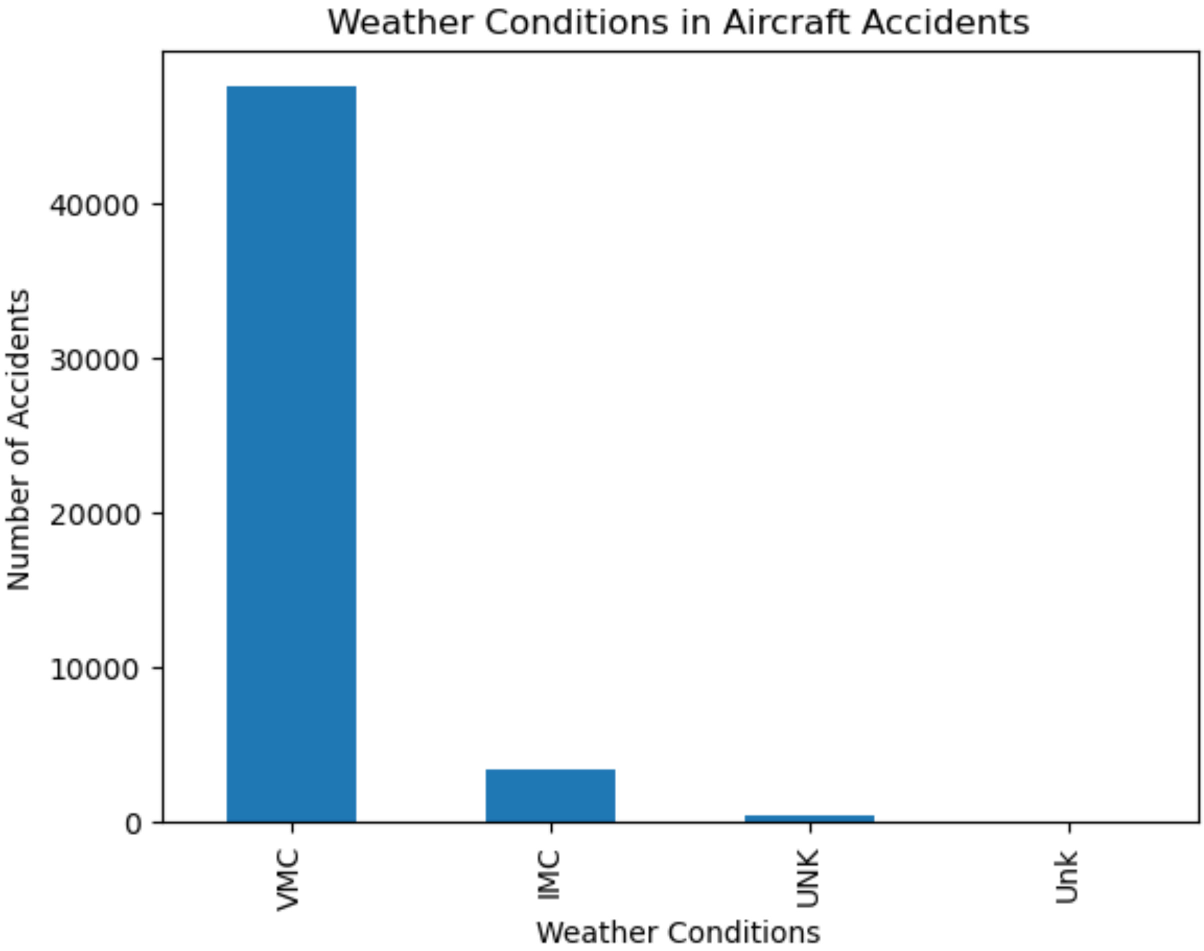

Top 10 Models for Mooney Aircraft in Accidents
Number of Accidents
0
20
40
60
80
100
120
M20J
M20C
M20F
M20E
M20K
M20
M20A
M20B
M20M
M-20C
Model



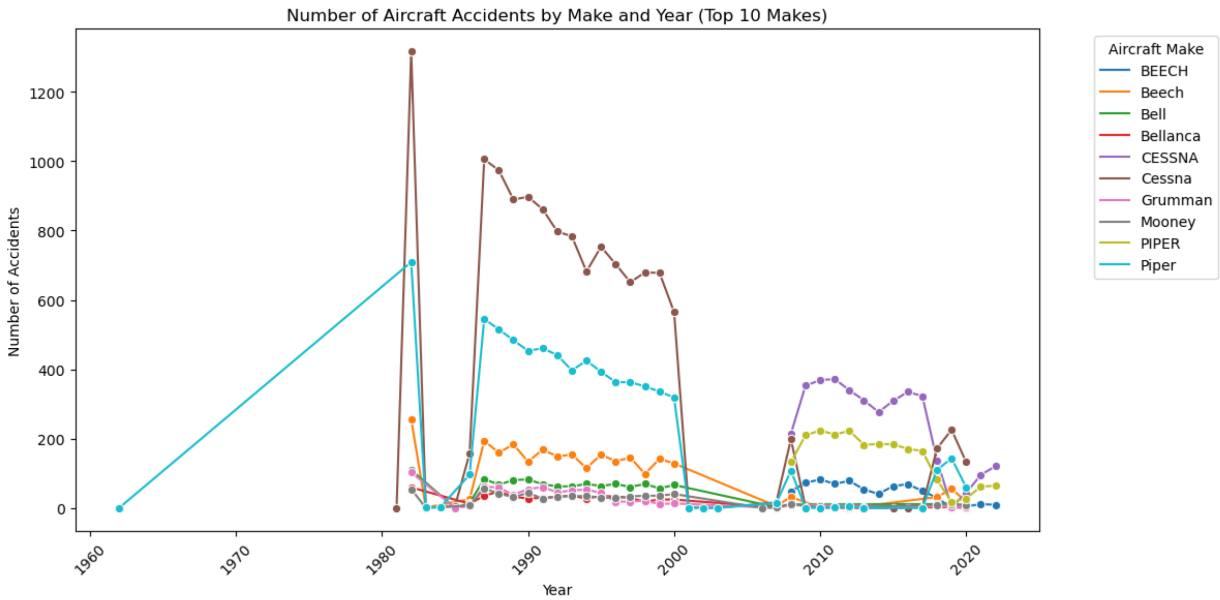
We start by identifying the top aircraft makes with the most accidents. Then, within each make, we focus on the models that have the highest number of incidents. This approach gives us a clear and detailed understanding of the data.

In [277…

```
# Explore Number of Accidents against Weather Conditions
visualizer.visualize_weather()
```


Weather Conditions in Aircraft Accidents
Number of Accidents
40000
30000
20000
10000
0
VMC
IMC
UNK
Unk
Weather Conditions

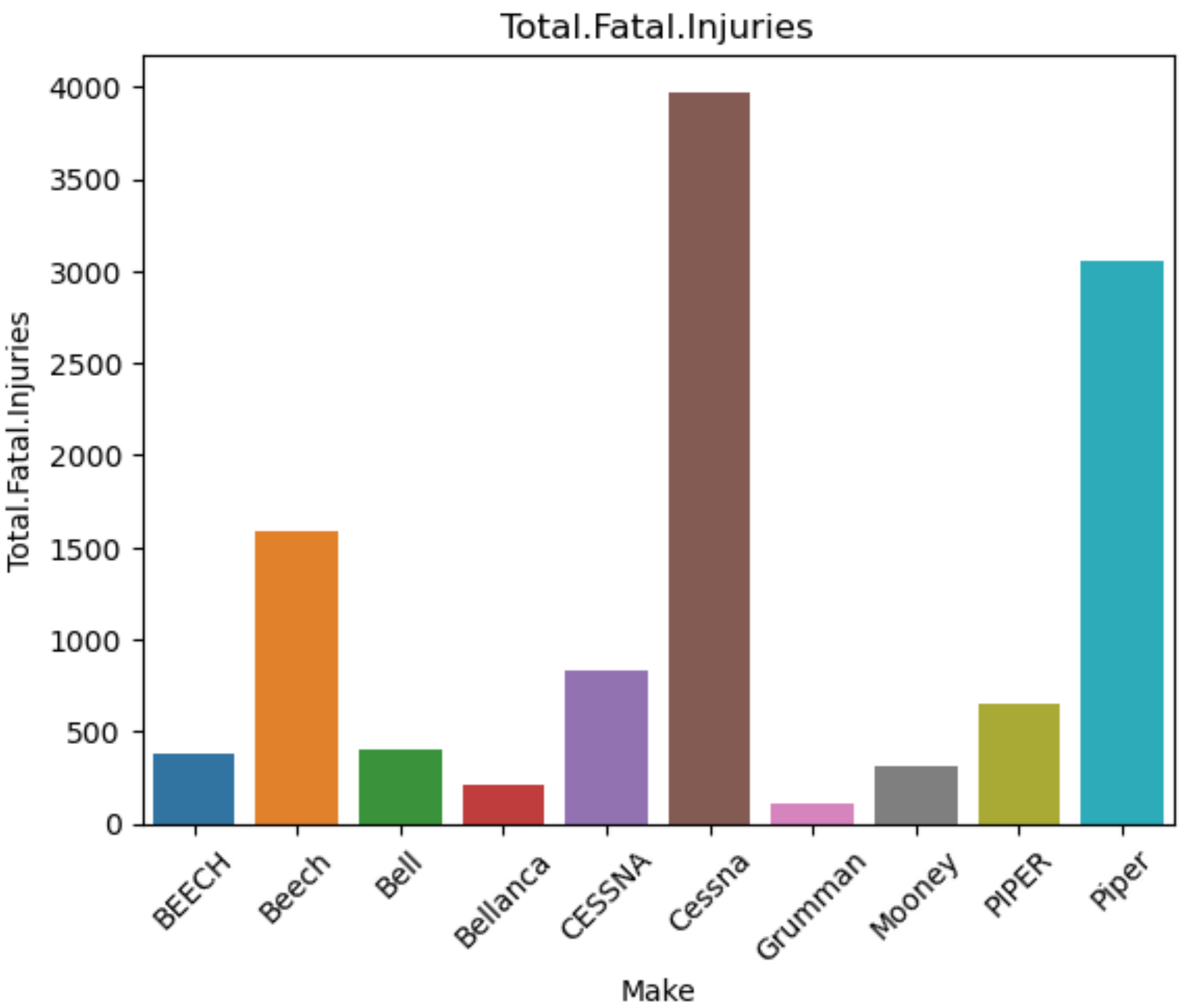
In [ ]:
```
# Exploring Number of Aircraft Accidents by Make and Year
class AircraftAccidentVisualizer:
    def __init__(self, data):
        self.data = data
        self.data['Make'] = self.data['Make'].str.strip()

        self.data['Event.Date'] = pd.to_datetime(self.data['Event.Date'],

        self.data['Year'] = self.data['Event.Date'].dt.year

    def accidents_by_make_and_year(self):
        accident_counts = self.data.groupby(['Make', 'Year']).size().rese
        return accident_counts

    def plot_accidents_line(self, top_n=10):
        accident_counts = self.accidents_by_make_and_year()

        top_makes = accident_counts.groupby('Make')['Accident_Count'].sum
        accident_counts = accident_counts[accident_counts['Make'].isin(to

        plt.figure(figsize=(12, 6))
        sns.lineplot(x='Year', y='Accident_Count', hue='Make', data=accid

        plt.title(f'Number of Aircraft Accidents by Make and Year (Top {t
        plt.xlabel('Year')
        plt.ylabel('Number of Accidents')
        plt.xticks(rotation=45)
        plt.legend(title='Aircraft Make', bbox_to_anchor=(1.05, 1), loc='
        plt.tight_layout()
        plt.show()

visualizer = AircraftAccidentVisualizer(df)
visualizer.plot_accidents_line(top_n=10)
```

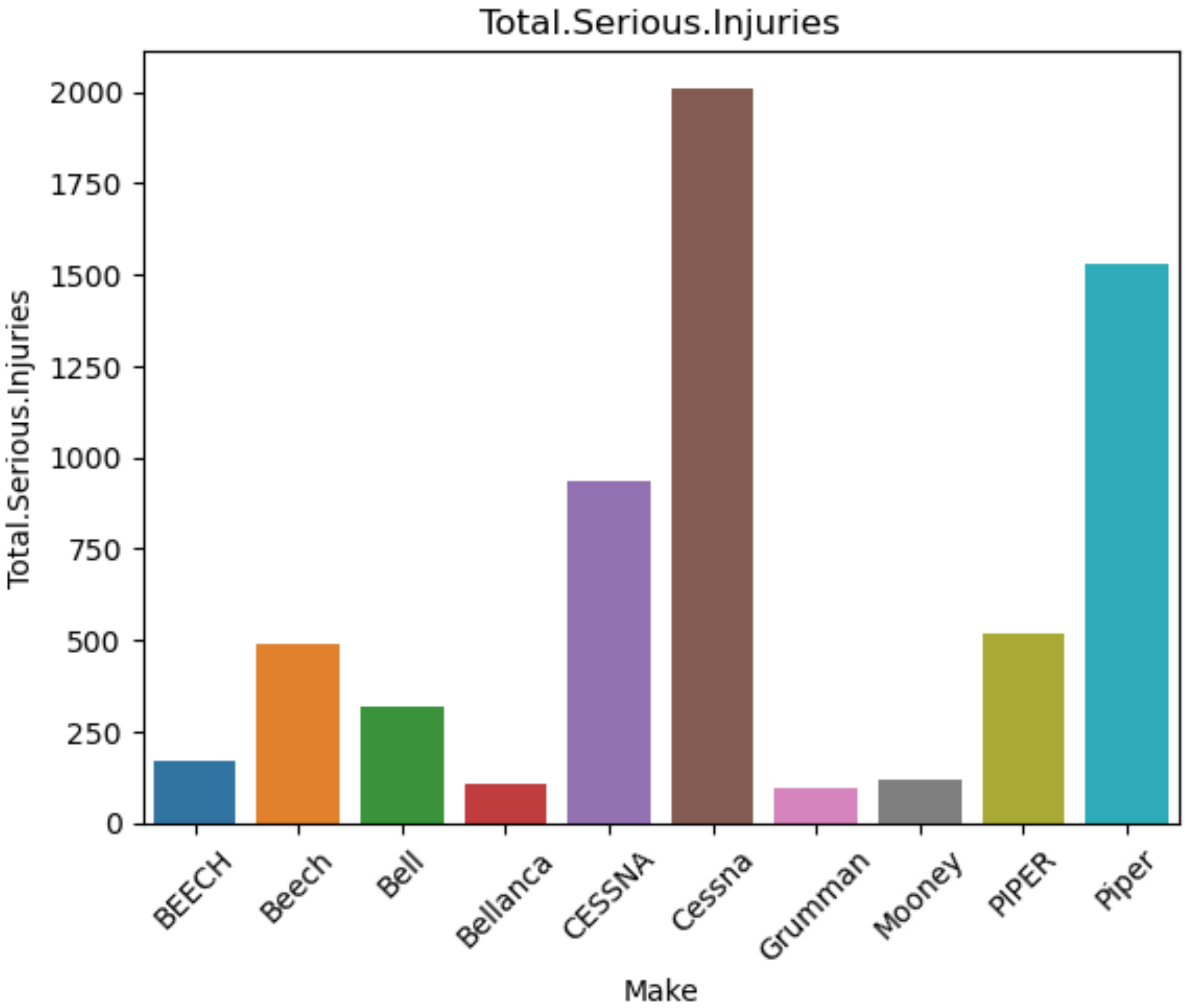


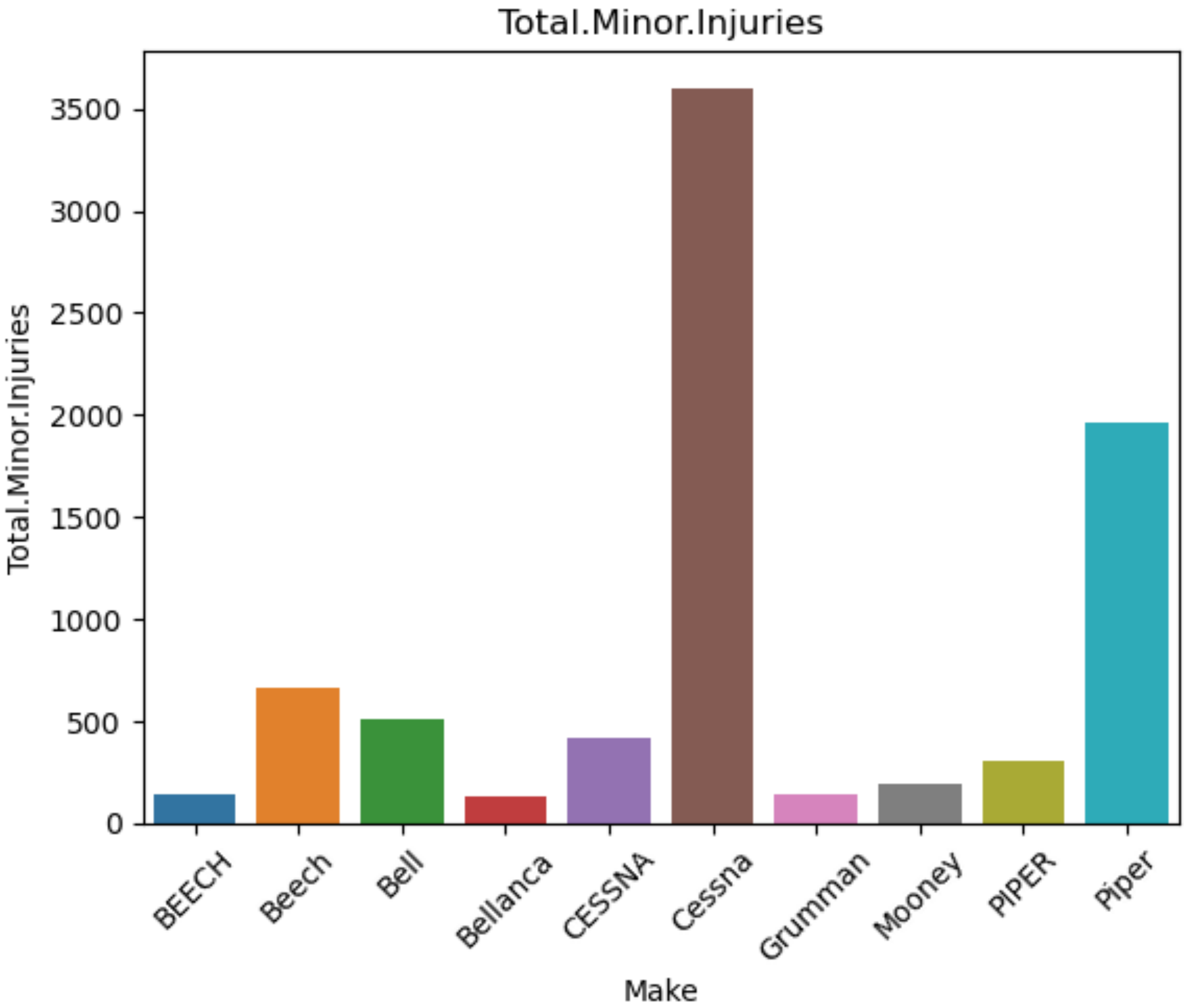
We note that across years, Cessna has been had the highest number of accidents in comparison to the other makes with 1983 being the year with the most accidents. Robinson has maintained a low accident rate across tha years since its manufacture in the 1980s.

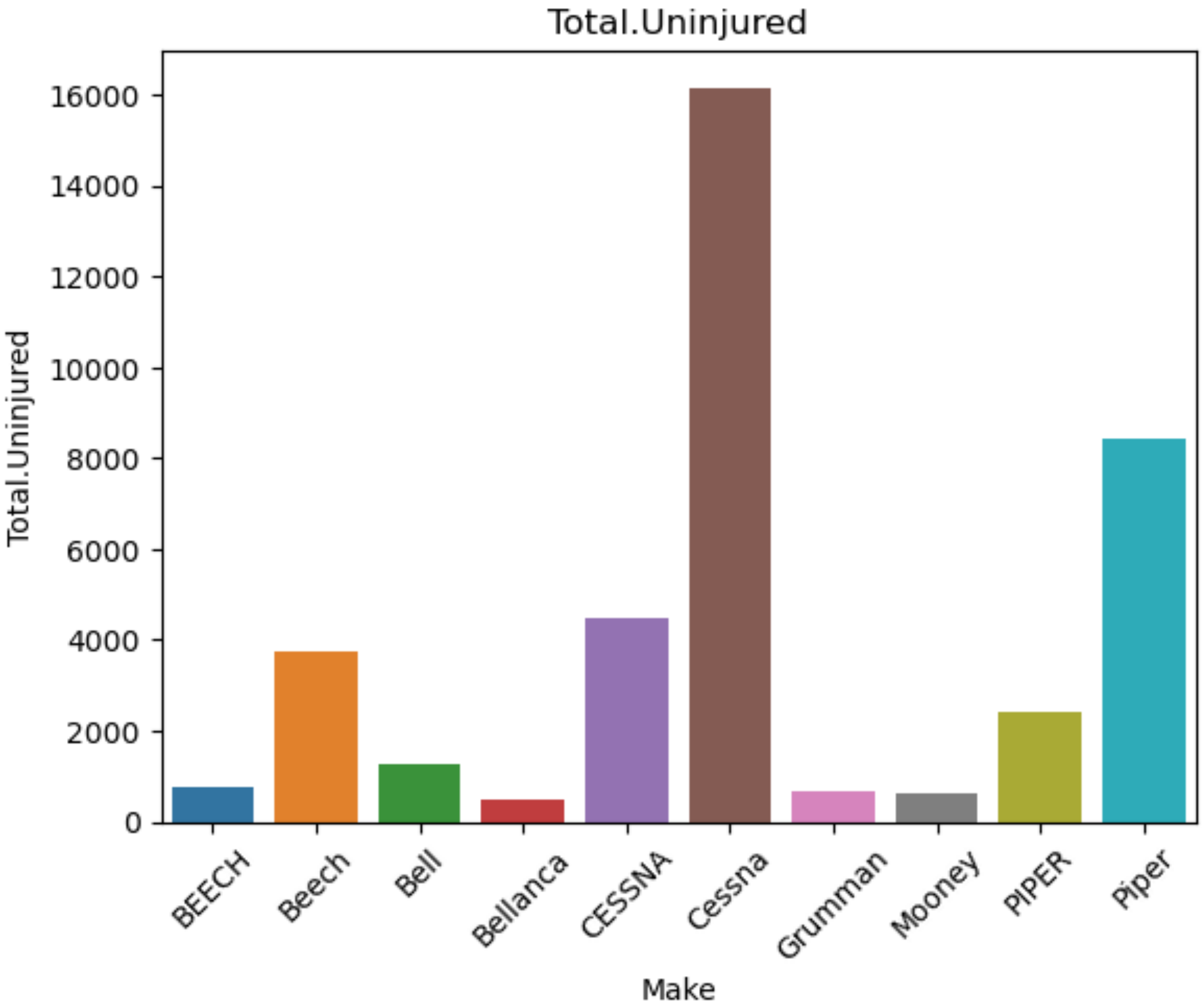
In [285…

```
# Exploring Injuries against the Make
top10_model=df["Make"].value_counts().head(10)
top10_model.index

injuries = ['Total.Fatal.Injuries', 'Total.Serious.Injuries', 'Total.Mino
df_selected = df[['Make'] + injuries]
top_10_make_injuries = df_selected[df_selected['Make'].isin(top10_model.index)]
inj_pivot = top_10_make_injuries.groupby('Make')[injuries].sum()

for injury_type in injuries:
    sns.barplot(x=inj_pivot.index, y=inj_pivot[injury_type])
    plt.title(injury_type)
    plt.xticks(rotation=45)
    plt.show()
```




Total.Serious.Injuries
Total.Serious.Injuries
2000
1750
1500
1250
1000
750
500
250
0
BEECH
Beech
Bell
Bellanca
CESSNA
Cessna
Grumman
Mooney
PIPER
Piper
Make


Total.Minor.Injuries
Total.Minor.Injuries
0
500
1000
1500
2000
2500
3000
3500
BEECH
Beech
Bell
Bellanca
CESSNA
Cessna
Grumman
Mooney
PIPER
Piper
Make



In this visualisation we note that Cessna accounts for the highest numbers across total fatal, serious, and minor injuries, whereas Boeing demonstrates the largest proportion of uninjured individuals.

Conclusion

When considering an aircraft purchase, safety should be a primary factor. The data indicates that Cessna aircraft are associated with higher counts of fatal, serious, and minor injuries, which may warrant a closer examination of their safety features, maintenance history, and pilot training requirements. On the other hand, Boeing's record of having the highest number of uninjured individuals suggests a strong safety track record in specific scenarios, potentially making it a more reliable choice for buyers prioritizing safety outcomes. Buyers should also assess the intended use of the aircraft, its operational history, and any available safety enhancements or upgrades to make an informed decision. Aircraft/engine type resilience against bad weather should be a afactor to consider.