aviation-analysis-1

March 27, 2025

0.1 Project Title: Aviation Accident Data Analysis

Objectives: This project aims to analyze aviation accident data to identify trends, clean inconsistencies, and visualize insights. The workflow follows five structured steps:

0.1.1 Workflow & Steps

Step 1: Importing and Loading Data 1.1 Import necessary Python libraries
1.2 Load the dataset and display the first few rows

Step 2: Exploratory Data Analysis (EDA) 2.1 Check dataset shape (rows & columns)
2.2 View dataset information (column types, missing values)
2.3 Summary statistics of numerical columns
2.4 Check for duplicate rows
2.5 Check for unique values in key categorical columns

Step 3: Data Cleaning 3.1 Remove duplicates
3.2 Handle missing values appropriately
3.3 Convert data types if necessary
3.4 Standardize column names for consistency
3.5 Correct inconsistent categorical values
3.6 Save the cleaned dataset

Step 4: Data Visualization 4.1 Plot distributions of numerical variables
4.2 Visualize accident trends over time
4.3 Show accidents by aircraft type
4.4 Analyze fatalities and survivability rates
4.5 Heatmaps to show correlations

Step 5: Additional Enhancements & Insights 5.1 Feature Engineering (if necessary)
5.2 Save the final cleaned dataset for further use

0.1.2 Step 1: Importing Libraries and Loading Dataset

1.1 Import necessary libraries

[3]:
```
#  Import  library
import pandas as pd  # Data manipulation and analysis
```

1.2 loading a dataset

[4]:
```
# 1.2 Load the dataset
file_path = r"C:\Users\hp\OneDrive\Desktop\DSF-FT12\DS-Phase1\Phase 1␣
 ↪Project\aviation-accident-analysis\data\AviationData.csv"
# Load dataset with encoding to prevent character errors
df = pd.read_csv(file_path, encoding="ISO-8859-1")
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_10896\782696525.py:4: DtypeWarning:
Columns (6,7,28) have mixed types. Specify dtype option on import or set
low_memory=False.
  df = pd.read_csv(file_path, encoding="ISO-8859-1")
```

[5]:
```
# 1.3 Display first few rows
df.head()  # Preview the dataset
```

[5]:

	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	36.922223	-81.878056	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]
```

[6]:
```
# 1.4 Check dataset info
df.info()  # Summary of dataset, including column types and non-null values
```

```
<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            50132 non-null  object
 9   Airport.Name            52704 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     87507 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             81793 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           82505 non-null  object
```

```
 30  Publication.Date        75118 non-null  object
dtypes: float64(5), object(26)
memory usage: 21.0+ MB
```

[7]:
```
# 1.5 Check for missing values
df.isnull().sum()  # Count missing values in each column
```

[7]:
```
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
```

0.1.3 Step 2: Exploratory Data Analysis (EDA)

2.1 Summary statistics

[8]:
```
# 2.1 Summary statistics
df.describe()  # Generates summary statistics for numerical columns
```

[8]:
```
       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
```

2.2 Check for duplicates

[9]:
```
# 2.2 Check for duplicates
duplicates = df.duplicated().sum()
print(f"Number of duplicate rows: {duplicates}")  # Identify duplicate records
```

```
Number of duplicate rows: 0
```

2.3 Display categorical features summary

[10]:
```
# 2.3 Display categorical features summary
df.describe(include=['O'])  # Provides summary for categorical columns
```

[10]:
```
              Event.Id Investigation.Type Accident.Number  Event.Date  \
count            88889              88889           88889       88889
unique           87951                  2           88863       14782
top     20001212X19172           Accident      CEN22LA149  1984-06-30
freq                3              85015               2          25

             Location        Country Latitude Longitude Airport.Code  \
count           88837          88663    34382     34373        50132
unique          27758            219    25592     27156        10374
top     ANCHORAGE, AK  United States  332739N  0112457W         NONE
freq              434          82248       19        24         1488

       Airport.Name  … Amateur.Built    Engine.Type FAR.Description  \
count         52704  …         88787          81793           32023
unique        24870  …             2             12              31
```

```
top         Private  …            No  Reciprocating             091
freq            240  …         80312          69530           18221

       Schedule Purpose.of.flight Air.carrier Weather.Condition  \
count     12582             82697       16648             84397
unique        3                26       13590                 4
top        NSCH          Personal       Pilot               VMC
freq       4474             49448         258             77303

       Broad.phase.of.flight   Report.Status Publication.Date
count                  61724           82505            75118
unique                    12           17074             2924
top                  Landing  Probable Cause       25-09-2020
freq                  15428           61754            17019

[4 rows x 26 columns]
```

[11]:
```
df.shape # chech the  dataframe rows and column
```

[11]:
```
(88889, 31)
```

2.5 Identify missing values

[13]:
```
# 2.5 Identify missing values
df.isnull().sum()  # Count missing values in each column
```

[13]:
```
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
```

0.1.4 Step 3: Data Cleaning

3.1 Remove duplicates

[14]:
```
# 3.1 Remove duplicates
df.drop_duplicates(inplace=True)  # Remove duplicate rows
```

3.2 Handle missing values () Fill with median or mode

[15]:
```
# 3.2 Handle missing values
        # Check which columns still have missing values
missing_values = df.isnull().sum()
print("Columns with missing values after cleaning:\n",
      missing_values[missing_values > 0])
```

```
Columns with missing values after cleaning:
 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
```

[20]:
```
df.fillna(df.mean(numeric_only=True), inplace=True)  # Fill numeric columns␣
 ↪with mean
df.fillna(df.mode().iloc[0], inplace=True)  # Fill categorical columns with␣
 ↪mode
df.dropna(inplace=True) #drop missing value
```

3.3 Standardize column names

[21]:
```
# 3.4 Standardize column names
df.columns = df.columns.str.strip().str.lower().str.replace(
    ' ', '_')  # Standardize column names
```

3.4 Correct inconsistent categorical values (Example: Convert text to lowercase)

[18]:
```
#3.4 Correct inconsistent categorical values (Example: Convert text to␣
 ↪lowercase)
for col in df.select_dtypes(include=['object']).columns:
    df[col] = df[col].str.lower().str.strip()
```

3.5 Verify dataset after cleaning

[22]:
```
#3.5 Verify dataset after cleaning
print({"missing_values": df.isnull().sum().sum(), "duplicates": df.duplicated(
).sum(), "data_types": df.dtypes.to_dict(), "shape": df.shape})
```

```
{'missing_values': 0, 'duplicates': 0, 'data_types': {'event.id': dtype('O'),
'investigation.type': dtype('O'), 'accident.number': dtype('O'), 'event.date':
dtype('O'), 'location': dtype('O'), 'country': dtype('O'), 'latitude':
dtype('O'), 'longitude': dtype('O'), 'airport.code': dtype('O'), 'airport.name':
dtype('O'), 'injury.severity': dtype('O'), 'aircraft.damage': dtype('O'),
'aircraft.category': dtype('O'), 'registration.number': dtype('O'), 'make':
dtype('O'), 'model': dtype('O'), 'amateur.built': dtype('O'),
'number.of.engines': dtype('float64'), 'engine.type': dtype('O'),
'far.description': dtype('O'), 'schedule': dtype('O'), 'purpose.of.flight':
dtype('O'), 'air.carrier': dtype('O'), 'total.fatal.injuries': dtype('float64'),
'total.serious.injuries': dtype('float64'), 'total.minor.injuries':
dtype('float64'), 'total.uninjured': dtype('float64'), 'weather.condition':
dtype('O'), 'broad.phase.of.flight': dtype('O'), 'report.status': dtype('O'),
'publication.date': dtype('O')}, 'shape': (88889, 31)}
```

[24]:
```
df .info
```

[24]:
```
<bound method DataFrame.info of             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
…                  …                  …             …          …
88884   20221227106491           accident      era23la093  2022-12-26
88885   20221227106494           accident      era23la095  2022-12-26
88886   20221227106497           accident      wpr23la075  2022-12-26
88887   20221227106498           accident      wpr23la076  2022-12-26
88888   20221230106513           accident      era23la097  2022-12-29

               location        country latitude longitude airport.code  \
0      moose creek, id  united states  332739n  0112457w         none
1       bridgeport, ca  united states  332739n  0112457w         none
2        saltville, va  united states  332739n  0112457w         none
3           eureka, ca  united states  332739n  0112457w         none
4           canton, oh  united states  332739n  0112457w         none
…                  …            …       …        …          …
88884    annapolis, md  united states  332739n  0112457w         none
88885      hampton, nh  united states  332739n  0112457w         none
88886       payson, az  united states  341525n  1112021w          pan
88887       morgan, ut  united states  332739n  0112457w         none
88888       athens, ga  united states  332739n  0112457w         none

      airport.name  … purpose.of.flight         air.carrier  \
0          private  …          personal               pilot
1          private  …          personal               pilot
2          private  …          personal               pilot
3          private  …          personal               pilot
4          private  …          personal               pilot
…            …  …                 …                  …
88884      private  …          personal               pilot
88885      private  …          personal               pilot
88886       payson  …          personal               pilot
88887      private  …          personal  mc cessna 210n llc
88888      private  …          personal               pilot

      total.fatal.injuries total.serious.injuries total.minor.injuries  \
0                      2.0               0.000000             0.000000
1                      4.0               0.000000             0.000000
2                      3.0               0.279881             0.357061
3                      2.0               0.000000             0.000000
```

```
4                      1.0                2.000000              0.357061
…                      …                   …                    …
88884                  0.0                1.000000              0.000000
88885                  0.0                0.000000              0.000000
88886                  0.0                0.000000              0.000000
88887                  0.0                0.000000              0.000000
88888                  0.0                1.000000              0.000000

      total.uninjured weather.condition  broad.phase.of.flight  \
0             0.00000               unk                 cruise
1             0.00000               unk                unknown
2             5.32544               imc                 cruise
3             0.00000               imc                 cruise
4             0.00000               vmc               approach
…               …                   …                    …
88884         0.00000               vmc                landing
88885         0.00000               vmc                landing
88886         1.00000               vmc                landing
88887         0.00000               vmc                landing
88888         1.00000               vmc                landing

        report.status publication.date
0      probable cause       25-09-2020
1      probable cause       19-09-1996
2      probable cause       26-02-2007
3      probable cause       12-09-2000
4      probable cause       16-04-1980
…                 …              …
88884  probable cause       29-12-2022
88885  probable cause       25-09-2020
88886  probable cause       27-12-2022
88887  probable cause       25-09-2020
88888  probable cause       30-12-2022

[88889 rows x 31 columns]>
```

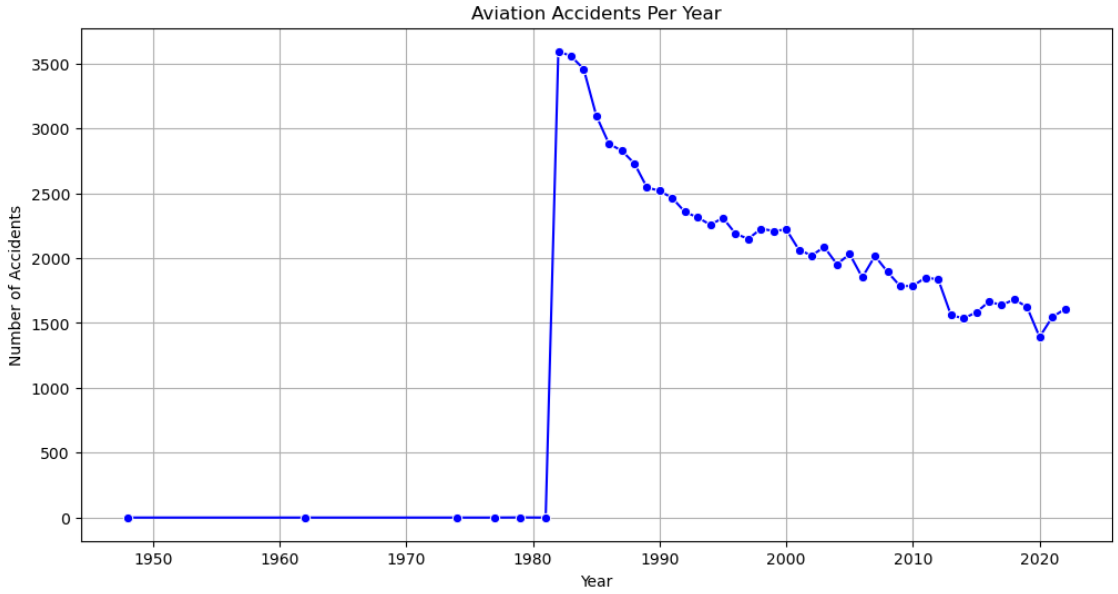
0.1.5 Step 4: Data Visualization

step 4 import matplotlib and sns

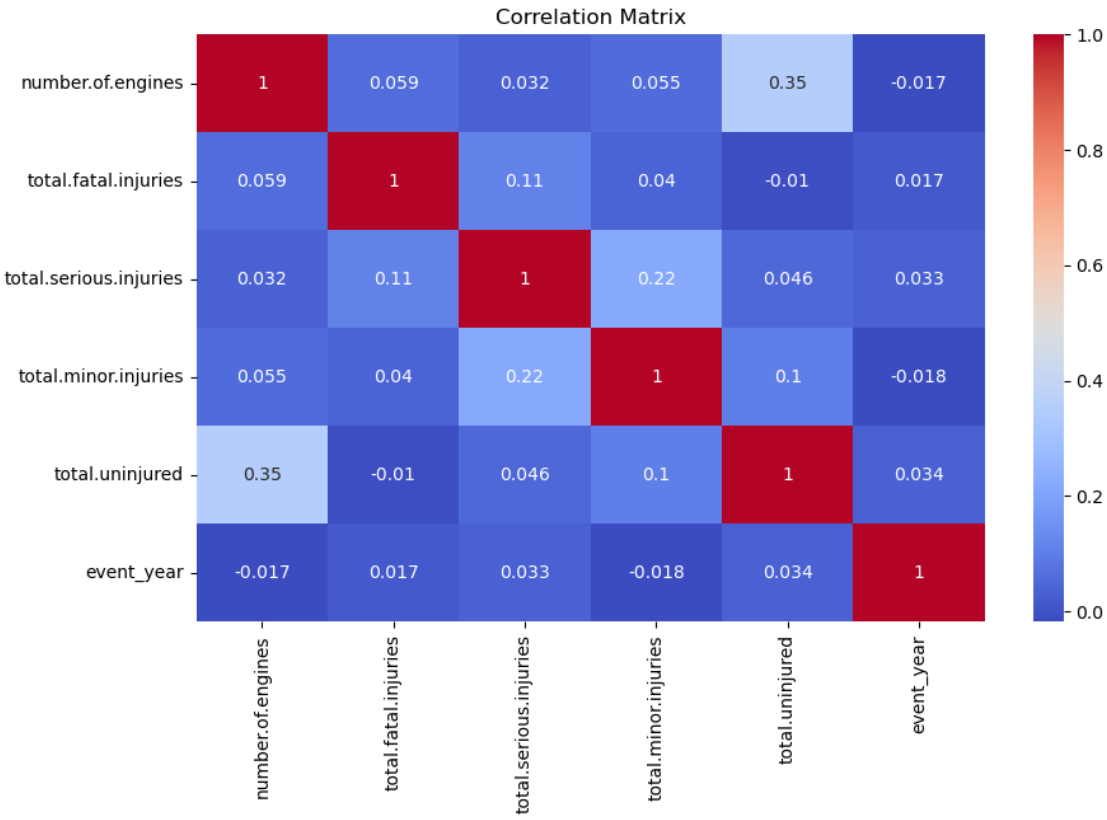
[25]:
```
# step 4.1 import matplotlib and sns
import matplotlib.pyplot as plt  # Data visualization
import seaborn as sns  # Advanced visualizatio
```

[27]:
```
# 4.1 Aviation Accidents Over the Years
df['event.date'] = pd.to_datetime(df['event.date'], errors='coerce')  # Convert␣
 ↪to datetime
```

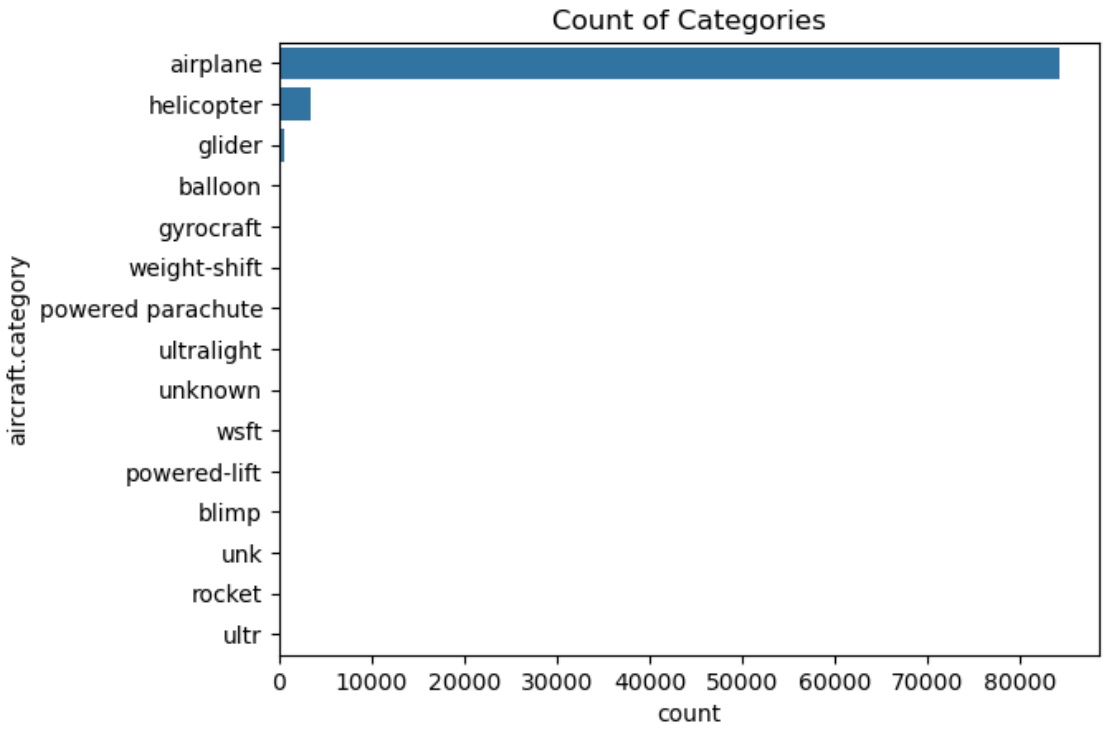
```
df['event_year'] = df['event.date'].dt.year  # Extract year

plt.figure(figsize=(12, 6))
df['event_year'] = df['event.date'].dt.year
event_counts = df.groupby('event_year').size().reset_index(name='count')
sns.lineplot(data=event_counts, x='event_year',
             y='count', marker='o', color='blue')
plt.title("Aviation Accidents Per Year")
plt.xlabel("Year")
plt.ylabel("Number of Accidents")
plt.grid()
plt.show()
```



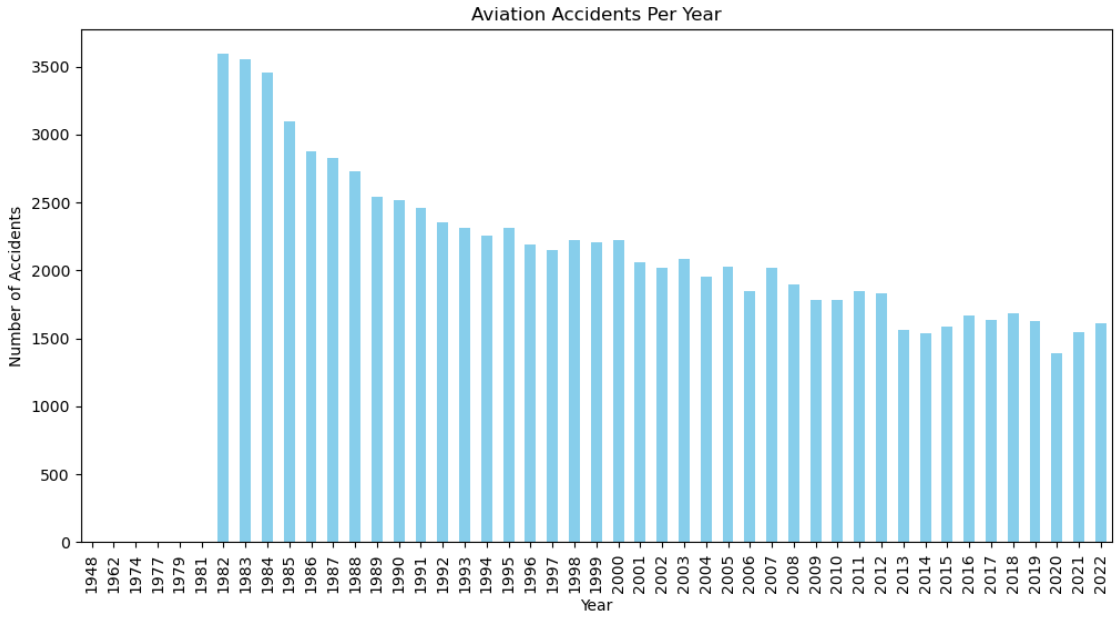
[28]:
```
# 4.2 Correlation Heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(df.select_dtypes(
    include=['number']).corr(), annot=True, cmap='coolwarm')
plt.title("Correlation Matrix")
plt.show()
```



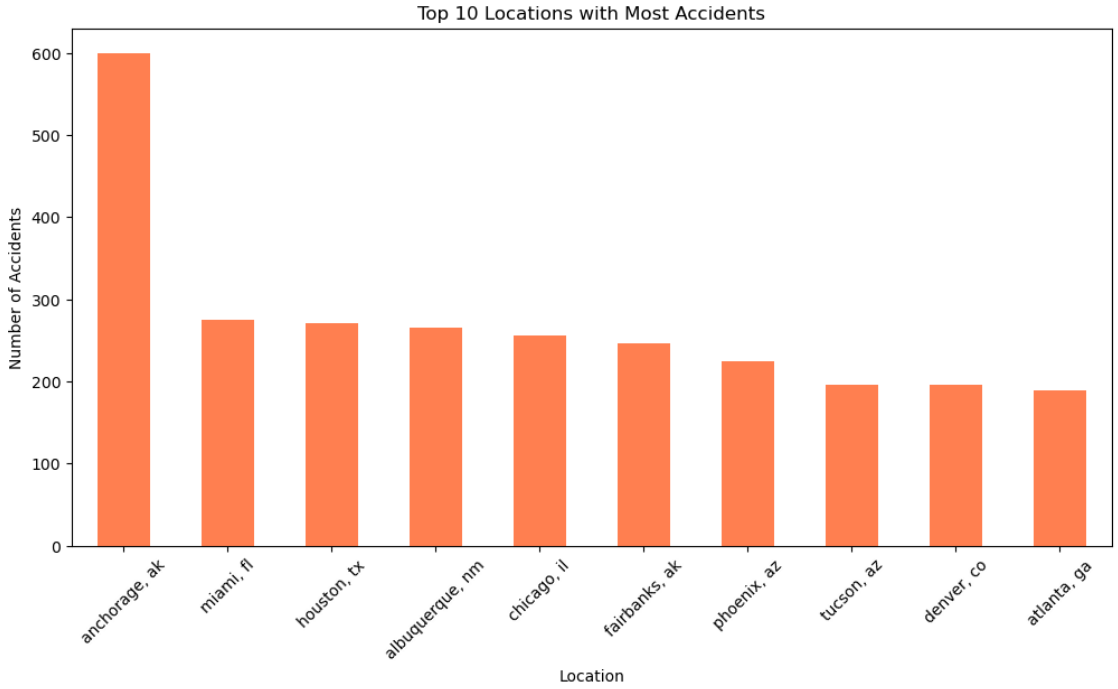
[29]:
```
#  4.3 for categorical column
sns.countplot(y=df['aircraft.category'],
              order=df['aircraft.category'].value_counts().index)
plt.title("Count of Categories")
plt.show()
```



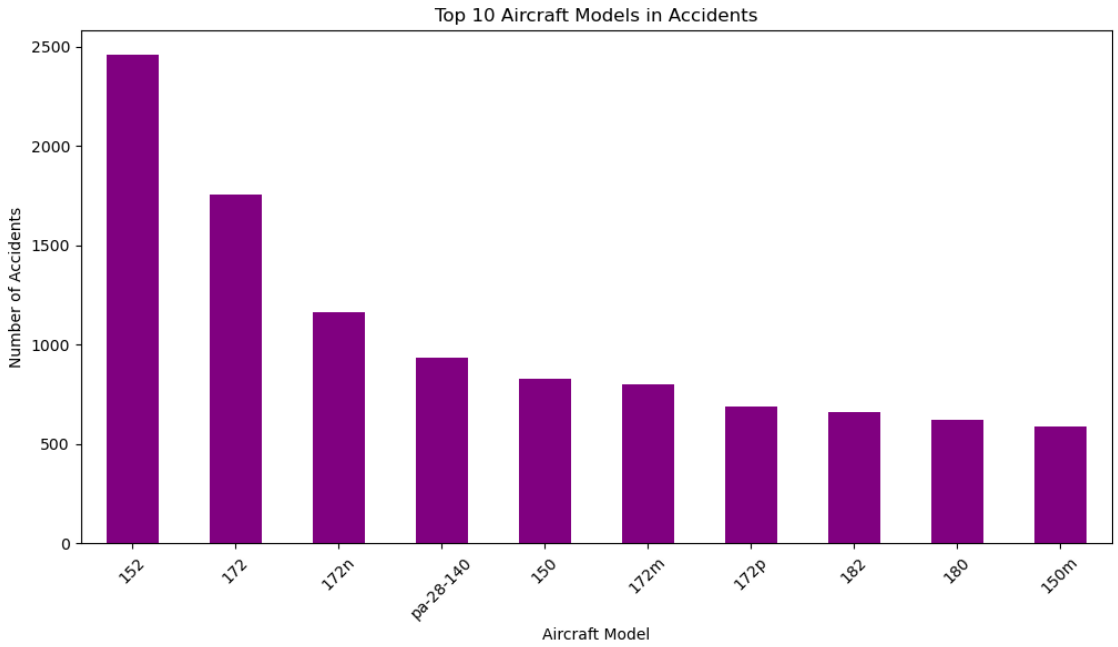
[30]:
```
# 4.4. Accidents per year
if 'event_year' in df.columns:
    plt.figure(figsize=(12, 6))
    df.groupby('event_year').size().plot(kind='bar', color='skyblue')
    plt.title("Aviation Accidents Per Year")
    plt.xlabel("Year")
    plt.ylabel("Number of Accidents")
    plt.show()
```



[31]:
```
# 4.5. Accidents by location
if 'location' in df.columns:
    plt.figure(figsize=(12, 6))
    df['location'].value_counts().nlargest(10).plot(kind='bar', color='coral')
    plt.title("Top 10 Locations with Most Accidents")
    plt.xlabel("Location")
    plt.ylabel("Number of Accidents")
    plt.xticks(rotation=45)
    plt.show()
```

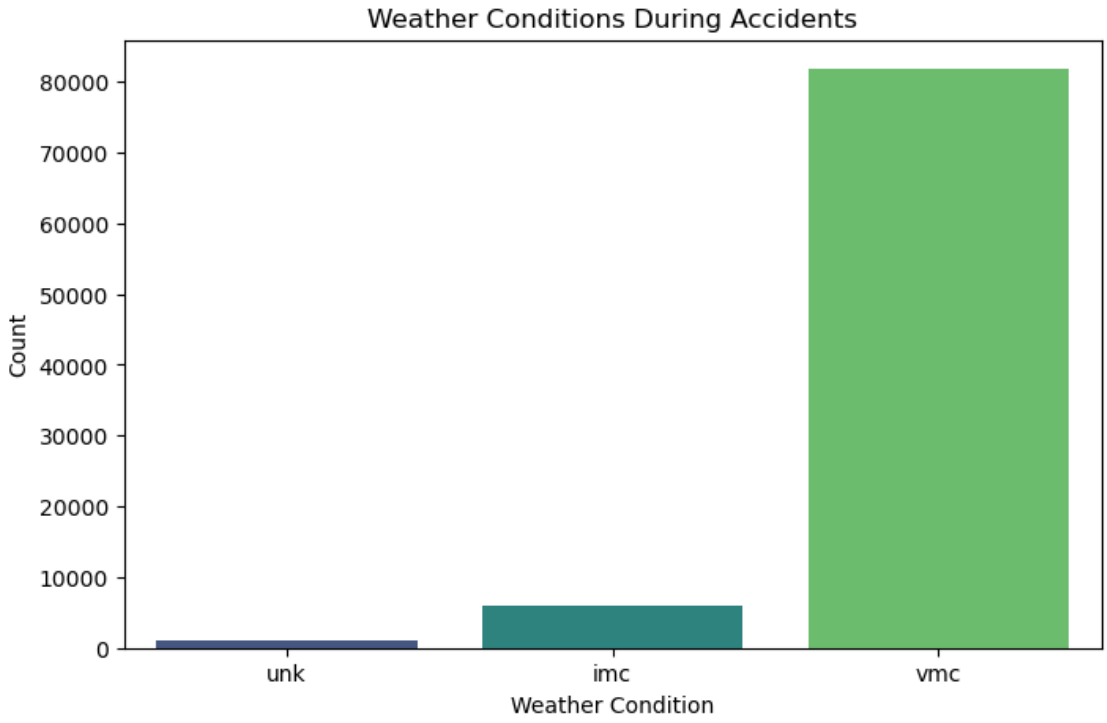


[32]:
```
# 4.6 Aircraft Models in Accidents
if 'model' in df.columns:
    plt.figure(figsize=(12, 6))
    df['model'].value_counts().nlargest(10).plot(kind='bar', color='purple')
    plt.title("Top 10 Aircraft Models in Accidents")
    plt.xlabel("Aircraft Model")
    plt.ylabel("Number of Accidents")
    plt.xticks(rotation=45)
    plt.show()
```



[33]:
```
# 4.7  Weather conditions during accidents
if 'weather.condition' in df.columns:
    plt.figure(figsize=(8, 5))
    sns.countplot(x=df['weather.condition'], palette='viridis')
    plt.title("Weather Conditions During Accidents")
    plt.xlabel("Weather Condition")
    plt.ylabel("Count")
    plt.show()
```

```
C:\Users\hp\AppData\Local\Temp\ipykernel_10896\1870508292.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same
effect.

  sns.countplot(x=df['weather.condition'], palette='viridis')
```



[35]:
```
print("Missing Values:\n", df.isnull().sum())  # Check missing values
print("\nDuplicate Rows:", df.duplicated().sum())  # Check duplicate rows
print("\nData Types:\n", df.dtypes)  # Check data types
```

```
Missing Values:
 event.id                 0
investigation.type        0
accident.number           0
event.date                0
location                  0
country                   0
latitude                  0
longitude                 0
airport.code              0
airport.name              0
injury.severity           0
aircraft.damage           0
aircraft.category         0
registration.number       0
make                      0
model                     0
amateur.built             0
number.of.engines         0
engine.type               0
```

```
far.description           0
schedule                  0
purpose.of.flight         0
air.carrier               0
total.fatal.injuries      0
total.serious.injuries    0
total.minor.injuries      0
total.uninjured           0
weather.condition         0
broad.phase.of.flight     0
report.status             0
publication.date          0
event_year                0
dtype: int64

Duplicate Rows: 0

Data Types:
 event.id                          object
investigation.type                 object
accident.number                    object
event.date                 datetime64[ns]
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
event_year                         int32
dtype: object
```

[36]:
```
df.to_csv("cleaned_dataset.csv", index=False)  # Save for Tableau
```

[37]:
```
from IPython.display import FileLink
FileLink("cleaned_dataset.csv")  # Generates a download link
```

[37]:
```
c:\Users\hp\OneDrive\Desktop\DSF-FT12\DS-Phase1\Phase 1 Project\aviation-
accident-analysis\cleaned_dataset.csv
```