**Bird Strikes Dataset - Data Cleaning Process Report**

**👋 Introduction**

**Hello, I'm Avinash Rai!**

As a **Data Analyst**, I specialize in extracting actionable insights from raw data to help businesses make informed decisions. In this report, I will guide you through the comprehensive **data cleaning** process applied to the **Bird Strikes Dataset**. This dataset captures the instances of bird strikes involving airplanes, and we aim to prepare it for further analysis.

**"Data is the new oil, but like oil, it must be refined to be valuable." – Peter Sondergaard**

**📊 About the Project**

The **Bird Strikes Dataset** involves records that capture instances where birds collided with aircraft. This dataset is vital for analyzing trends, patterns, and the impact of bird strikes on aviation safety. Cleaning and transforming the data is a critical first step in making this dataset ready for further exploration and predictive modeling.

**🔧 Data Cleaning Steps**

The following sections break down the key cleaning steps and explain the importance of each transformation. These steps ensure that the dataset is ready for meaningful analysis.

**1. Loading the Dataset**

The dataset was loaded using **Pandas**, a powerful library in Python for data manipulation.

import pandas as pd

# Load the dataset

df = pd.read\_csv('bird\_strikes\_data.csv')

**2. Removing Irrelevant Columns**

**'Altitude bin'** was identified as an irrelevant column with no useful data. Therefore, it was removed from the dataset to streamline it.

df.drop(columns=['Altitude bin'], inplace=True)

* **Why remove it?**: Removing irrelevant columns improves dataset efficiency and ensures we focus on the most impactful variables for analysis.

**3. Handling Missing Values**

**a. Imputing Categorical Columns**

We handled missing categorical values by filling them with either the most frequent value (mode) or a predefined category like **'Unknown'**.

**Affected Columns:**

* **Aircraft: Type**
* **Airport: Name**
* **Effect: Impact to flight**
* **Wildlife: Number struck**
* **Conditions: Precipitation**
* **Remarks**

df['Aircraft: Type'].fillna(df['Aircraft: Type'].mode()[0], inplace=True)

df['Airport: Name'].fillna(df['Airport: Name'].mode()[0], inplace=True)

df['Wildlife: Size'].fillna(df['Wildlife: Size'].mode()[0], inplace=True)

df['Remarks'].fillna('No remark', inplace=True)

df['Effect: Impact to flight'].fillna('Unknown', inplace=True)

**b. Specific Categorical Columns with 'Unknown'**

Certain columns were assigned **'Unknown'** where data was missing, indicating an unknown or undefined value.

**Affected Columns:**

* **Aircraft: Number of engines?**
* **Origin State**
* **Wildlife: Number struck**

df['Aircraft: Number of engines?'] = df['Aircraft: Number of engines?'].cat.add\_categories('Unknown').fillna('Unknown')

df['Origin State'] = df['Origin State'].cat.add\_categories('Unknown').fillna('Unknown')

df['Wildlife: Number struck'] = df['Wildlife: Number struck'].cat.add\_categories('Unknown').fillna('Unknown')

**c. Remarks Imputation**

For missing **'Remarks'**, we replaced empty values with **'No remarks'** to ensure the column had meaningful entries.

df['Remarks'] = df['Remarks'].cat.add\_categories('No remarks').fillna('No remarks')

**4. Converting Data Types**

**a. Numeric Conversion**

Columns like **'Feet above ground'** and **'Cost: Total $'** were converted to numeric types for accurate calculations. Invalid entries were coerced into missing values (NaN), which were then handled accordingly.

df['Feet above ground'] = pd.to\_numeric(df['Feet above ground'], errors='coerce')

df['Cost: Total $'] = df['Cost: Total $'].replace({',': ''}, regex=True)

df['Cost: Total $'] = pd.to\_numeric(df['Cost: Total $'], errors='coerce')

**b. Categorical Columns Optimization**

Several columns that were stored as objects were converted to **'category'** type for memory optimization, improving performance and storage.

categorical\_columns = ['Aircraft: Type', 'Airport: Name', 'Aircraft: Make/Model', 'Wildlife: Number struck',

'Effect: Impact to flight', 'Effect: Indicated Damage', 'Aircraft: Number of engines?',

'Aircraft: Airline/Operator', 'Origin State', 'When: Phase of flight', 'Conditions: Precipitation',

'Remarks', 'Wildlife: Size', 'Conditions: Sky', 'Wildlife: Species', 'Pilot warned of birds or wildlife?',

'Is Aircraft Large?']

df[categorical\_columns] = df[categorical\_columns].apply(lambda x: x.astype('category'))

**5. Year Extraction**

We extracted the **'Year'** from the **'FlightDate'** column. This is an important step for time-based analysis, allowing us to explore trends across different years.

df['Year'] = df['FlightDate'].dt.year.astype('Int64')

**6. Handling Duplicates**

We checked for duplicates in the dataset and removed them. We also dropped rows with missing **'Record ID'** as it is essential for identifying individual entries.

df.drop\_duplicates(inplace=True)

df.dropna(subset=['Record ID'], inplace=True)

df.reset\_index(drop=True, inplace=True)

* **Why remove duplicates?**: Duplicate entries could skew the analysis and result in misleading insights.

**7. Final Dataset Review**

After applying all necessary transformations, we reviewed the dataset using **df.info()** and **df.head()** to ensure that the data was clean and structured for analysis.

print(df.info())

print(df.head())

**🚀 Project Insights**

By applying these data cleaning steps, the dataset is now ready for in-depth analysis. We can now explore patterns in bird strikes, the financial impact of such events, and the correlation between bird strikes and flight safety.

**💡 Key Takeaways**

* Cleaning and transforming data is critical to making it ready for analysis.
* Handling missing values, ensuring correct data types, and removing irrelevant information improve the quality of the dataset.
* Extracting important time features like **Year** and addressing issues like duplicates allows for more accurate insights.

**📈 Conclusion**

This **Bird Strikes Dataset** is now prepared for further analysis. The cleaned data will help identify trends and patterns in bird strike incidents, ultimately contributing to improved aviation safety practices.

**📫 Contact Me**

Feel free to connect with me if you're interested in discussing data projects or collaborations:

* **Email**: [masteravinashrai@gmail.com](mailto:masteravinashrai@gmail.com)
* **LinkedIn**: [Avinash Analytics](https://www.linkedin.com/in/avinashanalytics/)
* **Twitter (X)**: [@AvinashAnalytiX](https://x.com/AvinashAnalytiX)

**💬 My Philosophy**

**"Without data, you're just another person with an opinion." – W. Edwards Deming**

**Thank you for reading this data cleaning report! Feel free to reach out for any further discussions or queries.**

**Appendix**

* **Full Code Implementation**: The full Python code for the cleaning process can be accessed in my project’s GitHub repository
* **[**[**Bird-Strike-Data-Visualization-2000-2011/notebooks/datacleaning.ipynb at main · AvinashAnalytics/Bird-Strike-Data-Visualization-2000-2011**](https://github.com/AvinashAnalytics/Bird-Strike-Data-Visualization-2000-2011/blob/main/notebooks/datacleaning.ipynb)**]**