

Instituto Tecnológico y de Estudios Superiores de Occidente

Department of Electronics, Systems, and Informatics

Computing Systems Engineering

Machine Learning Course

Teacher: Edna L. Guevara Rivera

Bird strike fatality prediction on Airplane crashes

DATASET CLEANING

Presented by:

Marco Ricardo Cordero Hernández, 727272

Carlos Eduardo Rodríguez Castro, 727366

October 06th, 2022

AUTUMN, 2022

Tlaquepaque, México

# Index

[Introduction 1](#_Toc115937596)

[Cleaning process walkthrough 2](#_Toc115937597)

[Conclusions and pending work 8](#_Toc115937598)

[References 9](#_Toc115937599)

# Introduction

Once the project, goals and resources are defined, and as stated in the previous document, the next step toward completion it’s the *dataset cleaning*.

This process may vary depending on dataset structure, having multiple types of data crammed into several columns (referred as *features* from now on). For this particular project, additional measures had to be taken in order to transform string to numerical data.

The steps to clean the project’s dataset are described below. Just as a reminder, the dataset comes from an external source [1].

# Cleaning process walkthrough

First, libraries have to be imported in order to use their methods for data loading, manipulation, and visualization.

*# Libraries*

*import* pandas *as* pd

*import* numpy *as* np

*import* seaborn *as* sns

*import* matplotlib.pyplot *as* plt

Next, the file containing the data itself has to be loaded. The file name it’s *bird\_strikes.csv*.

*# Read dataset*

ds\_bst = pd.**read\_csv**('bird\_strikes.csv')

After the previous step, the dataset can be visualized just by invoking the store valuable.

Relevant information related to data types for each feature has to be displayed to determine which columns could be kept.

*# Dataset info*

ds\_bst.**info**()

Texto

Descripción generada automáticamente

Fig. 1 Dataset original features

Although output has been trimmed by the method containing library, critical information it’s displayed at the bottom, indicating that 16 object type features (most likely strings) are present. Also, several boolean features are contained within other features, and although they could work in their original state, it’s better to transform them into pure dichotomic values.

Before transforming present values, presence of null fields has to be taken into consideration.

*# Null values identification*

ds\_bst.**isna**().**sum**(*axis* = 0)

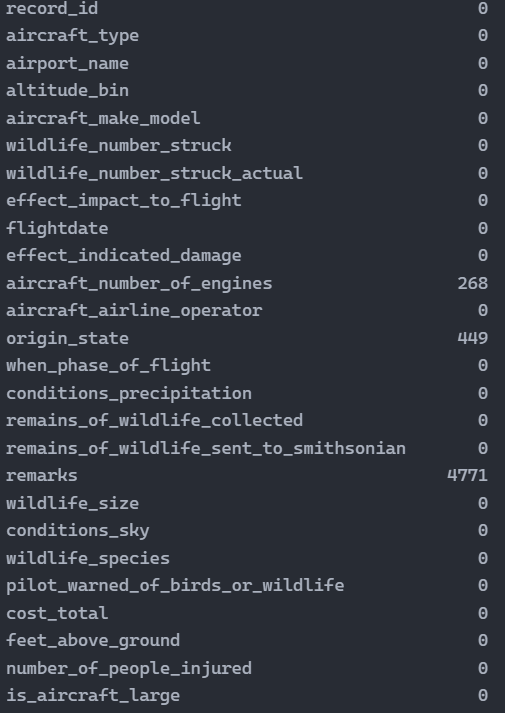


Fig. 2 Null presence in features

Through preliminary analysis, only one of the three null value containing features will have to be transformed into full data feature, this being *aircraft\_number\_of\_engines*, as the other ones will be suppressed later on.

*# Null values replacement*

ds\_bst['aircraft\_number\_of\_engines'].**fillna**(*value* = int(ds\_bst['aircraft\_number\_of\_engines'].**mean**()), *inplace* = True)

**print**(f"Null qty remaining: {ds\_bst.**isna**().**sum**(*axis* = 0)['aircraft\_number\_of\_engines']}")



Fig. 3 Null absence verification

Just as the previous step it’s completed, dropping the irrelevant features will take place. These droppable features are selected by looking at the information their data holds. Remaining features are shown below the step code.

*# Non-relevant columns dropping*

nrc = [

    'record\_id',

    'airport\_name',

    'wildlife\_number\_struck',

    'flightdate',

    'aircraft\_airline\_operator',

    'origin\_state',

    'remains\_of\_wildlife\_sent\_to\_smithsonian',

    'remarks',

    'wildlife\_species',

    'cost\_total'

]

'''

    Although 'cost\_total' could be used, the information related to that feature

    it's only obtained after the accident has occurred

'''

ds\_bst.**drop**(nrc, *inplace* = True, *axis* = 1)

ds\_bst.**info**()

Texto

Descripción generada automáticamente

Fig. 4 Remaining features after deletion

Now, just as stated before, object/string and bool features have to be transformed to numerical values. Two functions were made, one to transform categorical values, and another to turn boolean values to their binary representation.

def **categorize**(*dataset*, *feature*):

    holder = {}

    index = 0

*for* row *in* *dataset*[*feature*]:

*if* (row *not* *in* holder):

            holder[row] = index

            index += 1

*for* val *in* holder:

*dataset*[*feature*] = *dataset*[*feature*].replace([f'{val}'], holder[val])

def **to\_binary**(*dataset*, *feature*):

*dataset*[*feature*] = *dataset*[*feature*].apply(lambda *x* : 1 *if* *x* *else* 0)

With the aid of [Fig. 4](#Fig4), indexes of each feature and their corresponding transformation can be done easily.

features = ds\_bst.columns.values

to\_modify = (0, 1, 2, 4, 5, 7, 8, 10, 11)

to\_bin = (9, 12, 15)

*# Implementation not recommended for long features lenght (<50)*

*for* i *in* range(16):

*if* (i *in* to\_modify):

**categorize**(ds\_bst, features[i])

*elif* (i *in* to\_bin):

**to\_binary**(ds\_bst, features[i])

ds\_bst.**info**()

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Fig. 5 Transformed remaining features

Heading towards end of cleaning process, a visualization approach has to be taken in order to detect repeated values.

ds\_bst.**hist**(*bins* = 30, *figsize* = (20, 20), *color* = 'r')

Gráfico

Descripción generada automáticamenteGráfico

Descripción generada automáticamente

Fig. 6 Histograms

This extra analysis was helpful, because it ultimately helped in the exclusion of another two useless features.

ds\_bst.**drop**(['aircraft\_type', 'number\_of\_people\_injured'], *inplace* = True, *axis* = 1)

With that last step, the process of dataset cleaning has concluded. Additional analysis and clean dataset file generation can be found inside the jupyter notebook file.

# Conclusions and pending work

Now that the cleaning of the dataset has been made, the next step regarding the project would be the application of ML algorithms to predict an outcome given the relevant features.

# References

|  |  |
| --- | --- |
| [1] | J. Shih, «data.world,» 2016. [Online]. Available: https://data.world/shihzy/2000-2011-birds-strikes-planes. [Last access: 02 October 2022]. |