IO-AVSTATS - User Manual

Release 23.06.15

WORK IN PROGRESS

IO Aeronautical Autonomy Labs, LLC

© IO-Aero 2023

1. Introduction

The **IO-AVSTATS-DB** database contains not only the NTSB's aviation accident data, but also a large number of supplementary data from a wide variety of data sources. To enable the most comprehensive analysis of this data, **IO-Aero** provides the following cloud-based applications as tools:

- **Association Rule Analysis**: to apply various association rule algorithms to selected aspects of event causes, such as phase of flight or cause of accident, and more,
- **Aviation Event Analysis**: allows detailed analysis of selectable event data using data profiling, maps, various chart types and more,
- **Database Profiling**: allows exploratory data analysis of all tables and views in the database **IO-AVSTATS-DB**,
- **US Aviation Fatal Accident Analysis**: this is a freely available but in terms of data and functionality very limited version of the **Aviation Event Analysis** application.

2. Getting Started

a. Application Access

The **US Aviation Fatal Accidents** application is freely available and can be accessed either through the IO-Aero website at **www.io-aero.com** under **Products** or directly through the link **stats.io-aero.com**.

The remaining functionality is only available to IO-Aero customers either via the IO-Aero website **www.io-aero.com** under **Members** or directly via one of the following links:

Association Rule Analysis:	**slara.io-aero.com**
Aviation Event Analysis:	**ae1982.io-aero.com**
Database Profiling:	**pd1982.io-aero.com**
Members Only Area:	**members.io-aero.com**

To use this functionality, you must first authenticate yourself using the **Sign in** button:

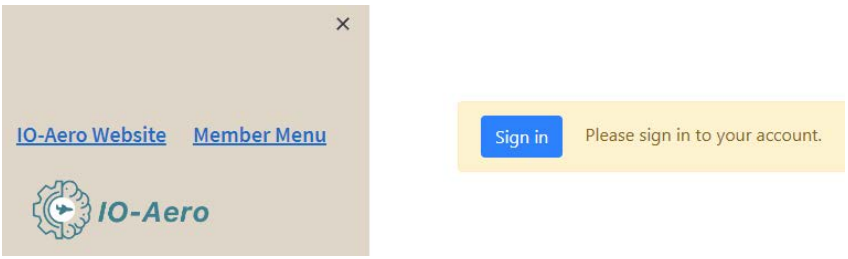


Figure 1: Application access lock

Authentication is done with username or email address and password. A process for changing or new password is available.

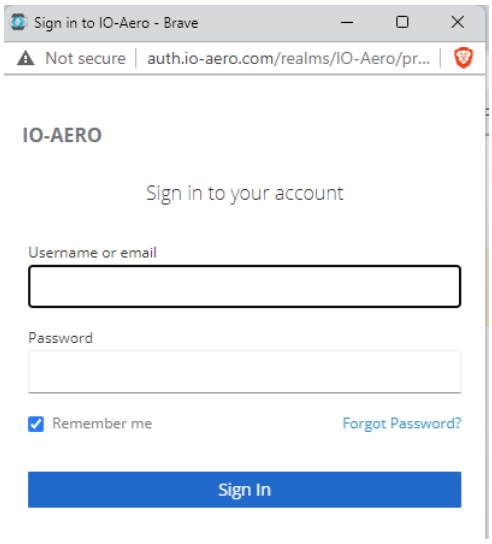


Figure 2: Authentication dialog

Depending on the personal permissions, the member menu provides the links to the applications under **`Applications`** and the database and user documentation in PDF format under **`Downloads`**. At the top left, access to the IO-Aero website is available.

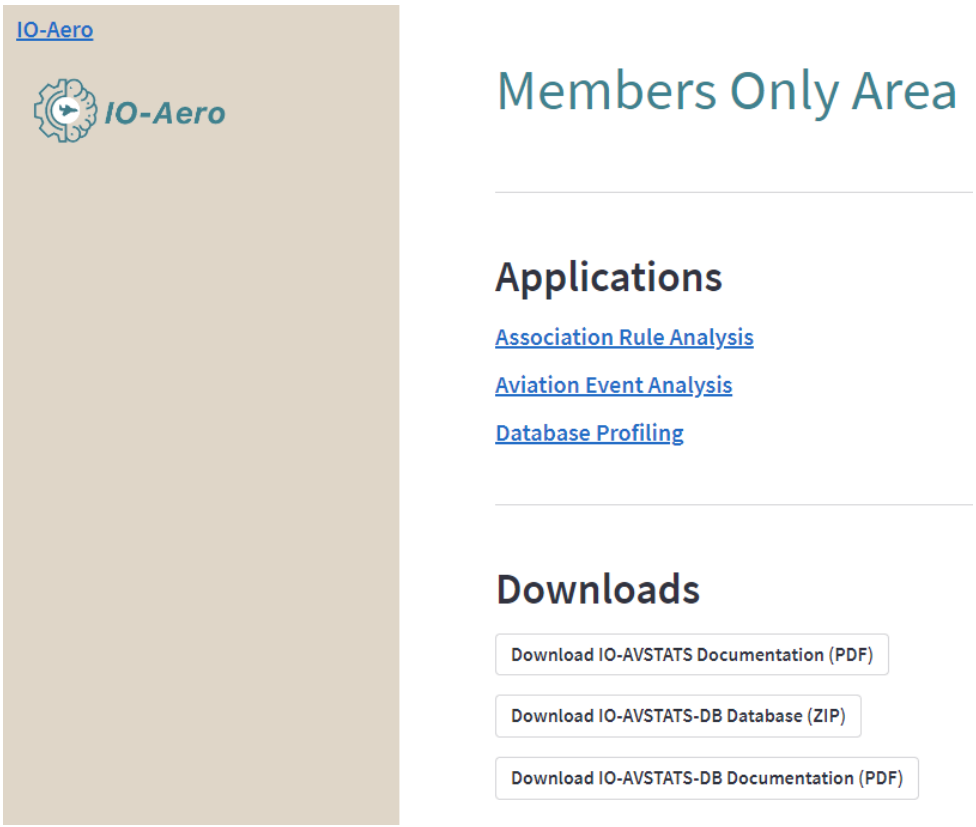


Figure 3: Sample menu in the members only area

In all applications, access to the IO-Aero website and access to the **`Members Only Area`** is available in the upper left corner.

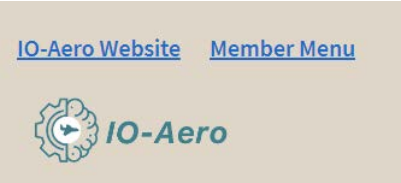


Figure 4: Application standard controls top left

b. General Usability

All applications are based on Streamlit. Streamlit is a powerful and versatile library that makes it easy to build and deploy data-driven web apps with Python. The main idea behind Streamlit is to provide a fast and efficient way to build and deploy data science and machine learning projects, without the need for extensive coding knowledge or web development skills. Streamlit comes with pre-built widgets and tools that allow you to create interactive elements such as sliders, text inputs, and dropdowns, as well as data visualizations like charts and graphs.

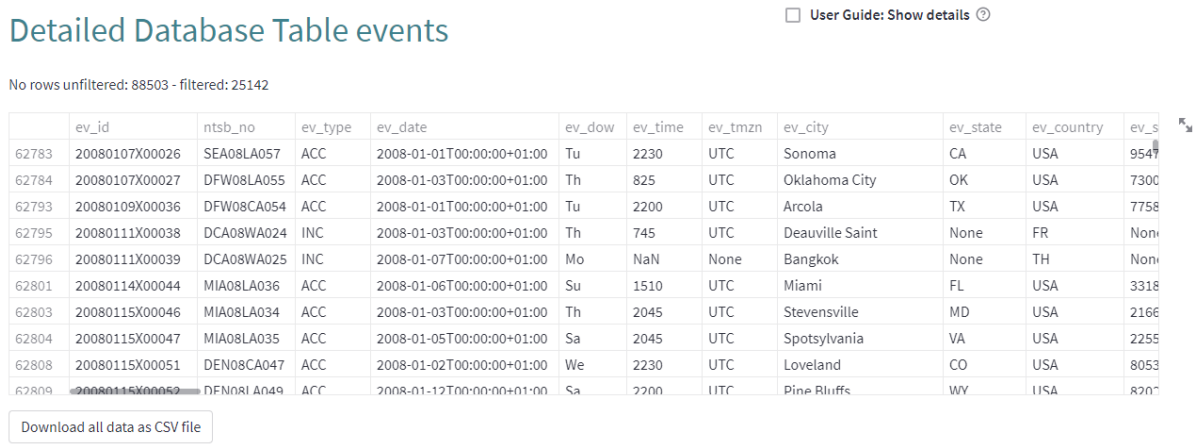
The user interface of a Streamlit application is composed of multiple elements that can be arranged in various ways depending on the design of the application. Here are some of the key elements that we use to create a Streamlit user interface:

- **`Sidebar`**: The sidebar is a container on the left-hand side of the screen that can hold various widgets, such as sliders, drop-down menus, and checkboxes. The sidebar can be used to allow users to interact with the application by adjusting input parameters.
- **`Title`**: The title is the text element that appears at the top of the page and typically describes the purpose of the application.
- **`Main page`**: The main page is the primary area of the screen where the results of the application are displayed. This can include charts, tables, or any other visualizations that the application generates.
- **`Buttons`**: Buttons are clickable elements that perform a specific action when clicked. These can be used to initiate computations or navigate between different views of the application.
- **`Text elements`**: Text elements can be used to provide instructions, descriptions, or context for the application. These can be placed throughout the interface to help guide users through the application.
- **`Layout components`**: Streamlit provides various layout components, such as columns and containers, that can be used to organize the user interface in a structured and intuitive way.

Overall, the structure of a Streamlit application is designed to provide a clear and intuitive user experience, with a focus on making it easy for users to interact with the application and understand its results.

i. Data frame

A data frame contains detailed data in an interactive table format similar to MS Excel.



	ev_id	ntsb_no	ev_type	ev_date	ev_dow	ev_time	ev_tmzn	ev_city	ev_state	ev_country	ev_s
62783	20080107X00026	SEA08LA057	ACC	2008-01-01T00:00:00+01:00	Tu	2230	UTC	Sonoma	CA	USA	9547
62784	20080107X00027	DFW08LA055	ACC	2008-01-03T00:00:00+01:00	Th	825	UTC	Oklahoma City	OK	USA	7300
62793	20080109X00036	DFW08CA054	ACC	2008-01-01T00:00:00+01:00	Tu	2200	UTC	Arcola	TX	USA	7758
62795	20080111X00038	DCA08WA024	INC	2008-01-03T00:00:00+01:00	Th	745	UTC	Deauville Saint	None	FR	Non
62796	20080111X00039	DCA08WA025	INC	2008-01-07T00:00:00+01:00	Mo	NaN	None	Bangkok	None	TH	Non
62801	20080114X00044	MIA08LA036	ACC	2008-01-06T00:00:00+01:00	Su	1510	UTC	Miami	FL	USA	3318
62803	20080115X00046	MIA08LA034	ACC	2008-01-03T00:00:00+01:00	Th	2045	UTC	Stevensville	MD	USA	2166
62804	20080115X00047	MIA08LA035	ACC	2008-01-05T00:00:00+01:00	Sa	2045	UTC	Spotsylvania	VA	USA	2255
62808	20080115X00051	DEN08CA047	ACC	2008-01-02T00:00:00+01:00	We	2230	UTC	Loveland	CO	USA	8053
62809	20080115X00052	DEN08LA049	ACC	2008-01-12T00:00:00+01:00	Sa	2200	UTC	Pine Bluffs	WY	USA	8207

Figure 5: Data frame example

The following interactive options are available for data frames:

- **Column resizing**: resize columns by dragging and dropping column header borders.
- **Column sorting**: sort columns by clicking on their headers.
- **Copy to clipboard**: select one or multiple cells, copy them to clipboard, and paste them into your favorite spreadsheet software.
- **Search**: search through data by clicking a table, using hotkeys (⌘ Cmd + F or Ctrl + F) to bring up the search bar, and using the search bar to filter data.
- **Table (height, width) resizing**: resize tables by dragging and dropping the bottom right corner of tables.

As a rule, the data contained in the data frame can be downloaded to the local system as a CSV file if requested.

3. Application Association Rule Analysis

a. Introduction

Association Rule Analysis, also known as Market Basket Analysis or Affinity Analysis, is a data mining technique that aims to identify the relationships between variables in a large dataset. Specifically, it focuses on finding the association between different items that are frequently bought together by customers in a transactional dataset.

In other words, it looks for patterns or rules that explain the co-occurrence of items in a dataset. For example, if a customer buys bread, there is a high likelihood that they will also buy butter, as these two items are often bought together. By identifying such patterns, businesses can understand the buying behavior of their customers and make better decisions about product placement, promotions, and pricing strategies.

Association Rule Analysis uses measures such as support, confidence, and lift to identify the most relevant patterns. Support refers to the frequency of occurrence of a particular itemset, confidence measures the strength of association between two items, and lift indicates the degree of association between two items beyond chance. The results of Association Rule Analysis are typically presented as a set of rules that express the relationship between items in terms of if-then statements.

This application gives an opportunity to experiment with association rule analysis algorithms. The content of the "basket" can be filled either with the occurrence or finding codes of an event or their components.

b. Processing Options

i. Run the Algorithms

Streamlit-based applications immediately recalculate each time a parameter is changed. However, since the desired setting can consist of a whole row of parameters, this checkbox allows you to explicitly trigger the recalculation after all desired settings have been made.

Figure 6: Checkbox to run the algorithms

ii. Apriori Algorithm

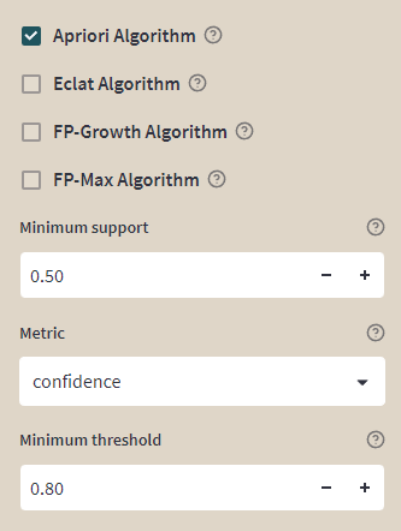


Figure 7: Settings for the Apriori algorithm

The Apriori algorithm is based on the idea that if an itemset is frequent, then all of its subsets must also be frequent. The algorithm works by scanning the dataset to identify the support (frequency) of each item, and then using that information to find frequent item sets. The support of an itemset is defined as the number of transactions in which the itemset appears. The algorithm then iterates through the dataset, looking for sets of items that appear together more often than a minimum support threshold.

The Apriori algorithm is called "Apriori" because it uses prior knowledge of frequent item sets to prune the search space and make the algorithm more efficient. In other words, the algorithm starts by looking for frequent item sets of length 1, and then uses those item sets to generate candidate sets of length 2. It continues this process, generating candidate item sets of increasing length until no more frequent item sets can be found.

The implementation of the Apriori algorithm used here is from the MLXTEND library. The following parameters are available:

- **Minimum support**: A float between 0 and 1 for minimum support of the item sets returned. The support is computed as the fraction **`transactions_where_item(s)_occur / total_transactions`**.
- **Metric:** Metric to evaluate if a rule is of interest - **A** stands for antecedent and **C** stands for consequent:
    - o - support(A->C) = support(A+C) [aka 'support'], range: [0, 1]
    - o - confidence(A->C) = support(A+C) / support(A), range: [0, 1]
    - o - lift(A->C) = confidence(A->C) / support(C), range: [0, inf]
    - o - leverage(A->C) = support(A->C) - support(A)*support(C), range: [-1, 1]
    - o - conviction = [1 - support(C)] / [1 - confidence(A->C)], range: [0, inf]
    - o - zhangs_metric(A->C) = leverage(A->C) / max(support(A->C)*(1-support(A)), support(A)*(support(C)-support(A->C))) range: [-1,1]
- **Minimum threshold**: Minimal threshold for the evaluation metric, via the metric parameter, to decide whether a candidate rule is of interest.

iii. ECLAT Algorithm

Figure 8: Settings for the ECLAT algorithm

The ECLAT (Equivalence Class Clustering and Bottom-Up Lattice Traversal) algorithm works by first finding all the individual items that occur frequently in the dataset. It then uses a depth-first search approach to efficiently find all the combinations of items that occur frequently together (i.e., the frequent item sets).

To achieve this, ECLAT creates a vertical representation of the dataset, where each column represents an item and each row represents a transaction. It then uses this representation to determine the support (i.e., frequency of occurrence) of each itemset.

The algorithm recursively combines frequent item sets of increasing size, and the resulting item sets are stored in a lattice structure. The lattice structure provides an efficient way to store and search for frequent item sets.

Overall, the ECLAT algorithm is a fast and memory-efficient method for finding frequent item sets in large transactional datasets

The implementation of the ECLAT algorithm used here is from the pyECLAT library. The following parameter is available:

- **Minimum support**: A float between 0 and 1 for minimum support of the item sets returned. The support is computed as the fraction **`transactions_where_item(s)_occur / total_transactions`**.

iv. FP-Growth Algorithm

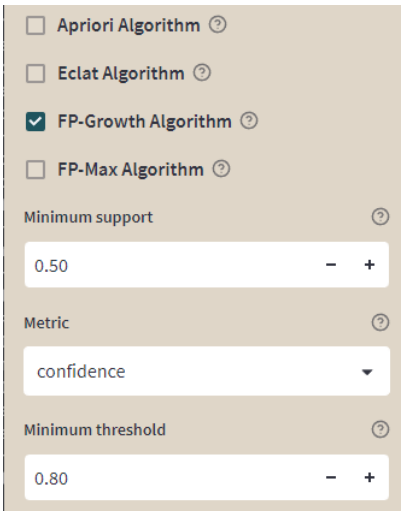


Figure 9: Settings for the FP-Growth algorithm

Here is a simple explanation of how the FP-Growth algorithm works:

1. First, the algorithm scans the dataset and calculates the frequency of each item (such as a product in a store) in the dataset.

2. Then, it builds a tree structure called an FP-Tree based on the frequent items identified in step 1. Each path in the tree represents a frequent pattern in the dataset.

3. Next, the algorithm uses the FP-Tree to mine the dataset for frequent item sets. It does this by recursively mining conditional sub-trees that are built from the original tree.

4. Finally, the algorithm returns a list of frequent item sets along with their corresponding support values (i.e., the frequency of occurrence in the dataset).

One of the benefits of the FP-Growth algorithm is that it can be faster than other algorithms that use an apriori approach, which require multiple passes over the dataset to identify frequent item sets. By building a tree structure, FP-Growth can reduce the number of scans needed to identify frequent patterns in the dataset, making it an efficient and effective method for large-scale data mining.

The implementation of the FP-Growth algorithm used here is from the MLXTEND library. The same parameters are available as for the Apriori algorithm.

v. FP-Max Algorithm

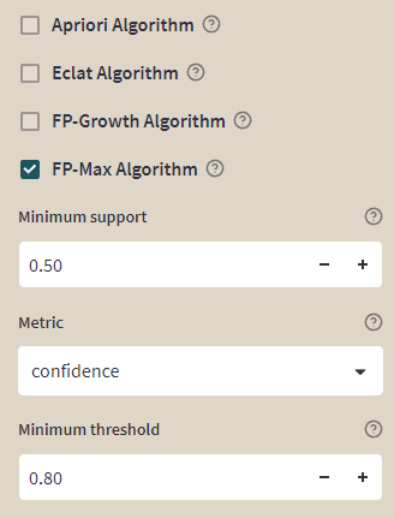


Figure 10: Settings for the FP-Max algorithm

The FP-Max Algorithm is used to find frequent patterns, which are patterns that appear frequently in a given set of data. The algorithm works by first scanning the data set to find all of the frequent items. It then uses these frequent items to build a tree structure called an FP-tree. Each branch in the tree represents a frequent item, and each node in the branch represents a transaction that contains that item.

Once the FP-tree is constructed, the algorithm recursively mines it to find all the frequent patterns. It does this by finding all the conditional patterns of each frequent item in the tree, starting with the most frequent item. A conditional pattern is a sub-pattern of a frequent pattern that appears in the same transactions as the frequent item.

The algorithm continues this process until no more frequent patterns can be found. The result is a set of frequent patterns that can be used to identify common trends and relationships in the data.

The implementation of the FP-Max algorithm used here is from the MLXTEND library. The same parameters are available as for the Apriori algorithm.

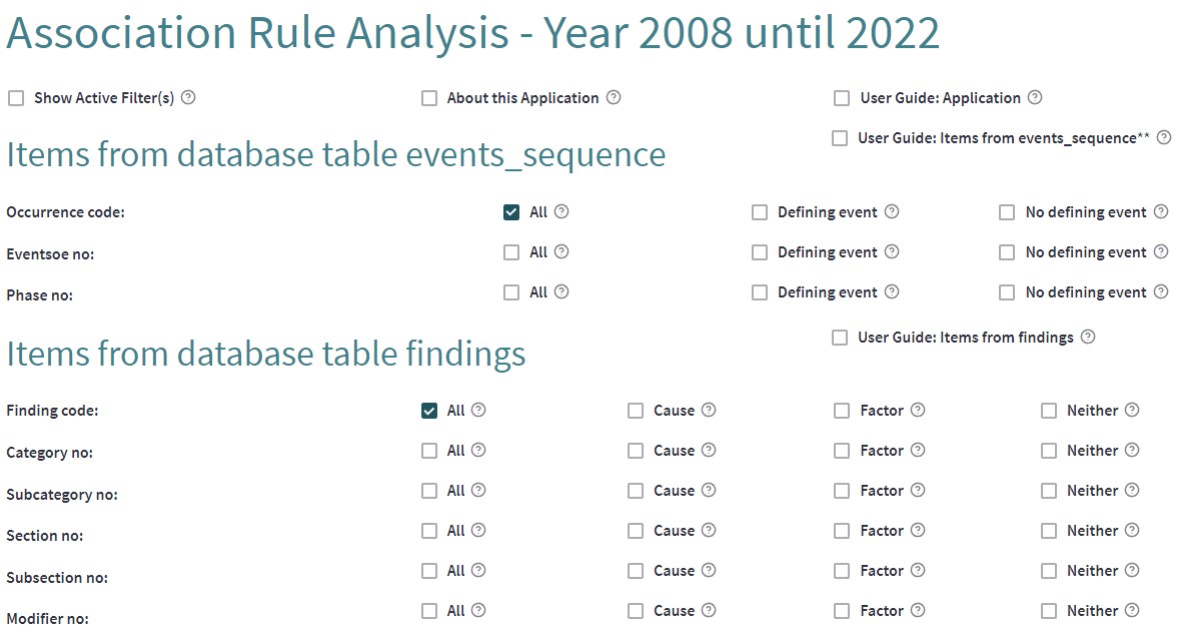
vi. Item Selection

Here the content of the "basket" can be determined. Available are the occurrence codes or their components from the database table **`events_sequence`** as well as the finding codes or their components from the database table **`findings`**.

The following rules must be adhered to when making the selection:

- For database table **`events_sequence`**, either **`Occurrence code`** or any number of its components can be selected.
- For database table **`events_sequence`**, either **`All`** or any of **`Defining event`** or **`No defining event`** can be selected.
- For database table **`findings`**, either **`Finding code`** or any number of its components can be selected.

- For database table **`findings`**, either **`All`** or any of **`Cause`**, **`Factor`** or **`Neither`** can be selected.



A finer selection of **`Occurrence code`** and **`Findings code`** can be made via their components at **`Filter Events Sequence`** and **`Filter Findings`**.

vii. Filtered Raw Data

With this checkbox the detailed raw data can be displayed.

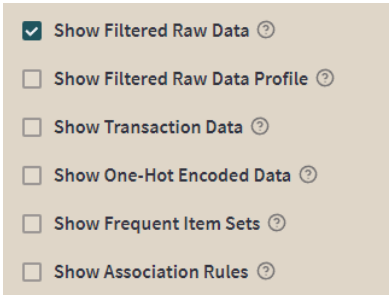
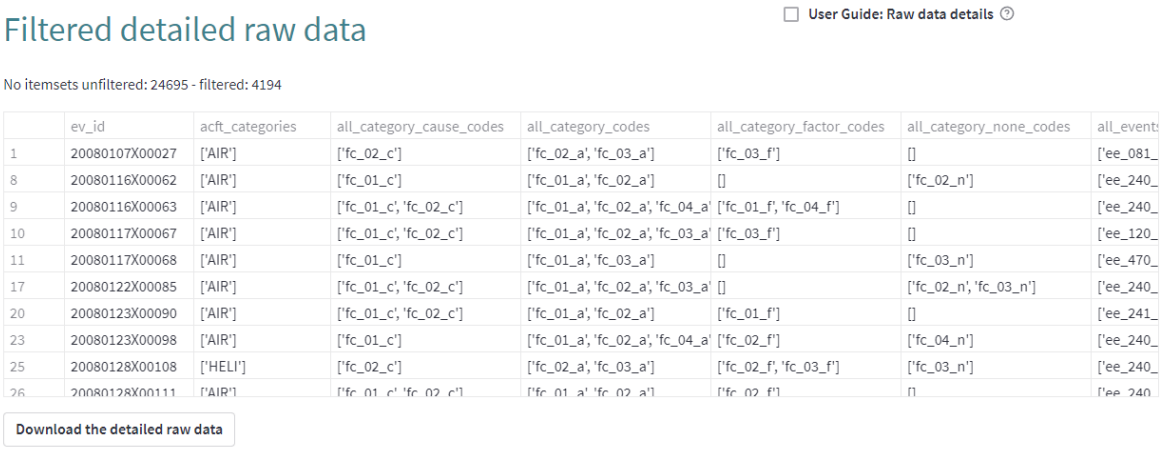


Figure 11: Checkbox to show the filtered raw data

The data is displayed in a table format, the so-called data frame. The data frame shows only the data that is still available after applying the selected filter options. The data in the data frame can also be downloaded as a CSV file using the '**`Download the detailed raw data`**' button.



More detailed information on how to use the data frame can be found in section **'2.b General Usability**'.

Filtered Raw Data Profile

This checkbox triggers profiling of the filtered raw data and creating a report.

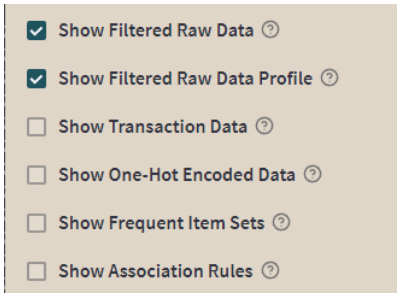


Figure 12: Checkbox to show the raw data profile

The **Overview** shows mostly global details about the dataset (number of records, number of variables, overall missingness and duplicates, memory footprint).

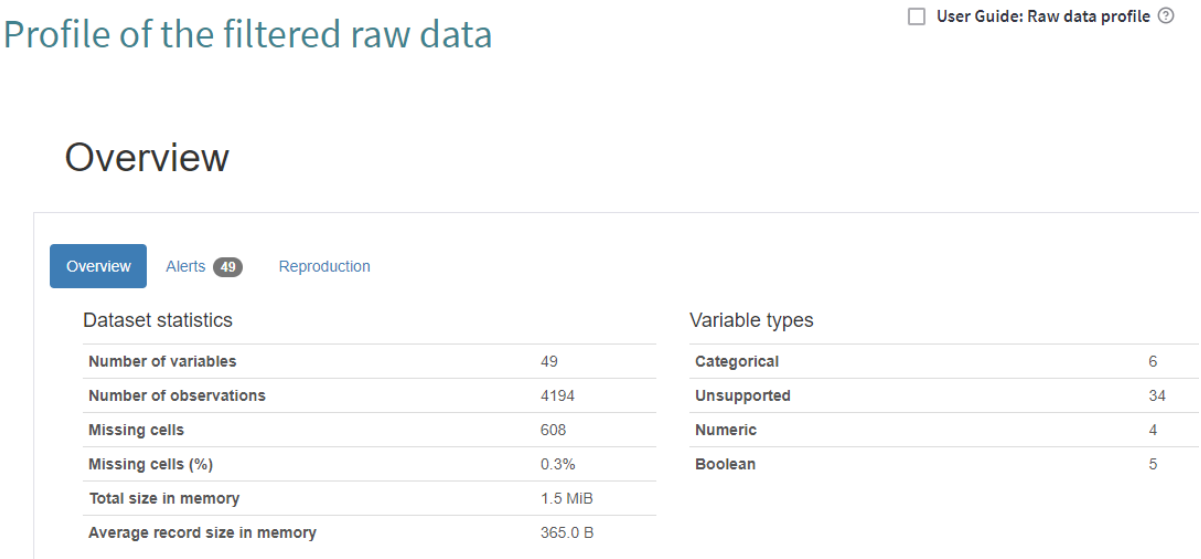


Figure 13: Raw data profile summary

The **Overview** is followed by the **Variables** section, which contains the detailed information for each column. The **Select Columns** select box at the top allows you to select a specific column.

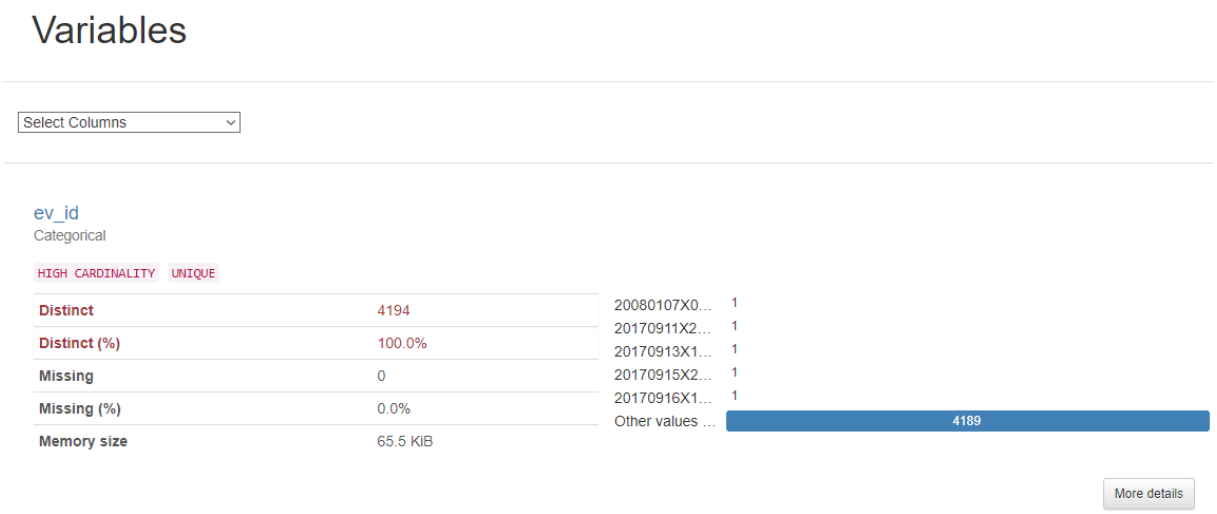


Figure 14: Simple view of column profile

More information about a specific column can be retrieved by clicking the **More Details** button. The additional information shown is located in the two tabs **Overview** and **Categories**.

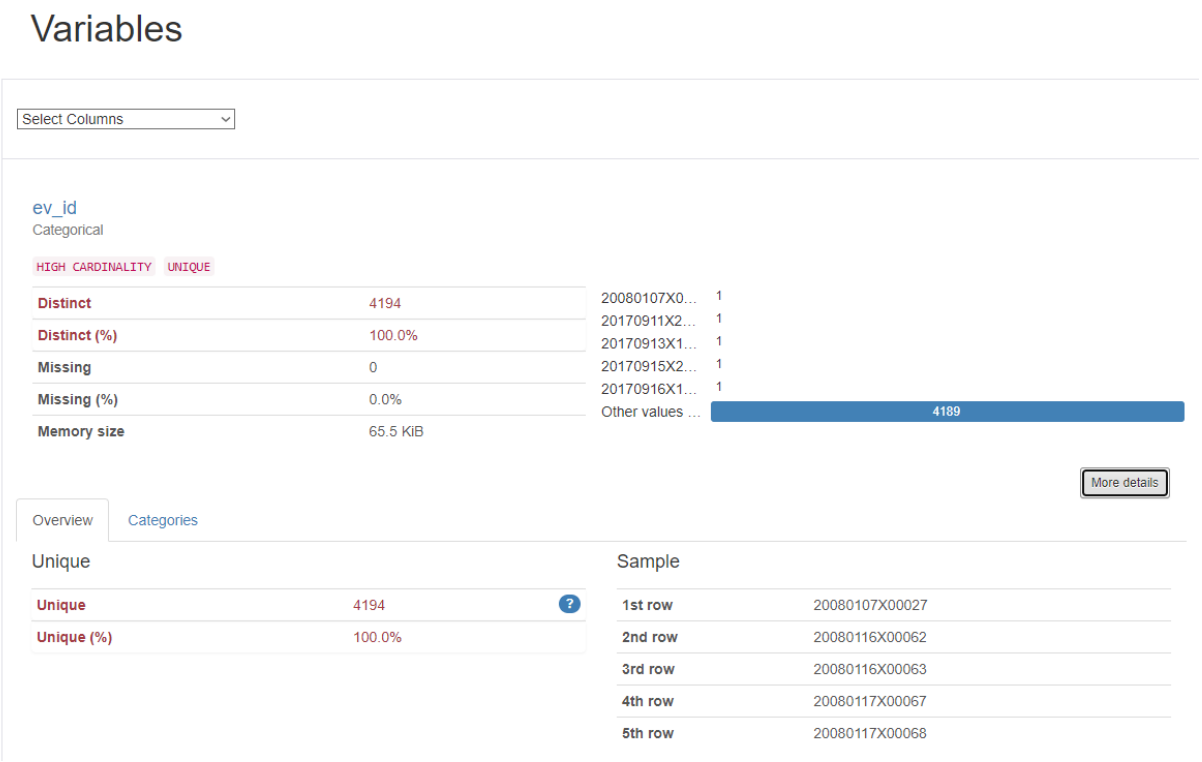


Figure 15: Extended view of column profile

Columns whose contents cannot be evaluated are shown as follows:

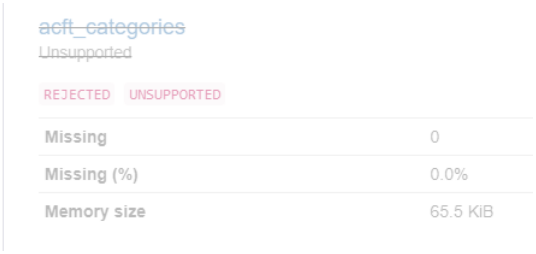


Figure 16: Example of an unsupported column

The **`Download the profile report`** button at the end of the report can be used to download an HTML version of the profiling report to the local computer.

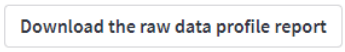


Figure 17: Download button

Further details can be found in the [ydata-profiling documentation].

viii. Transaction Data

xxxx

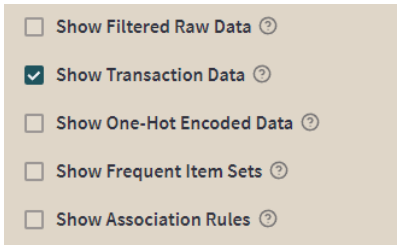
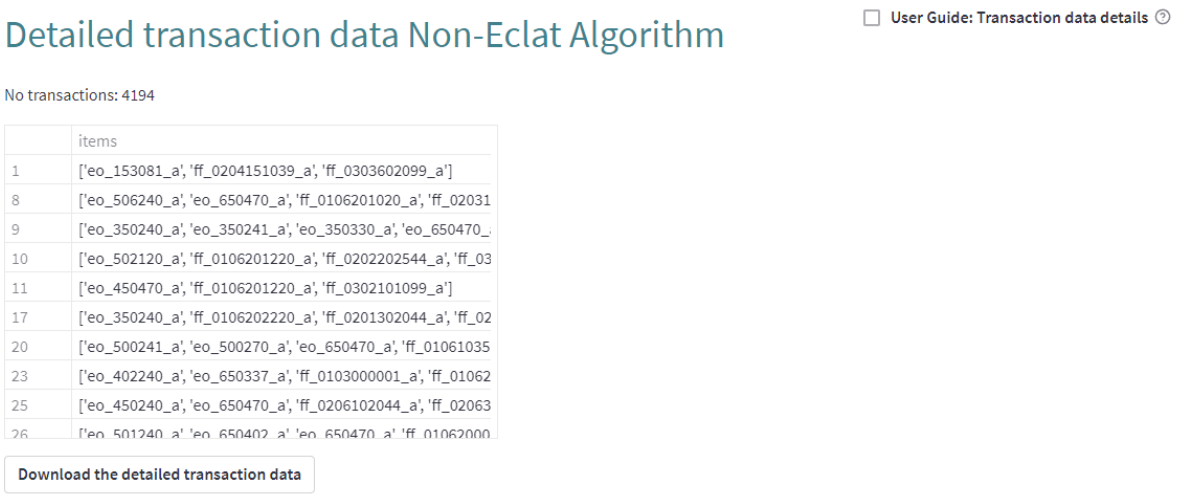
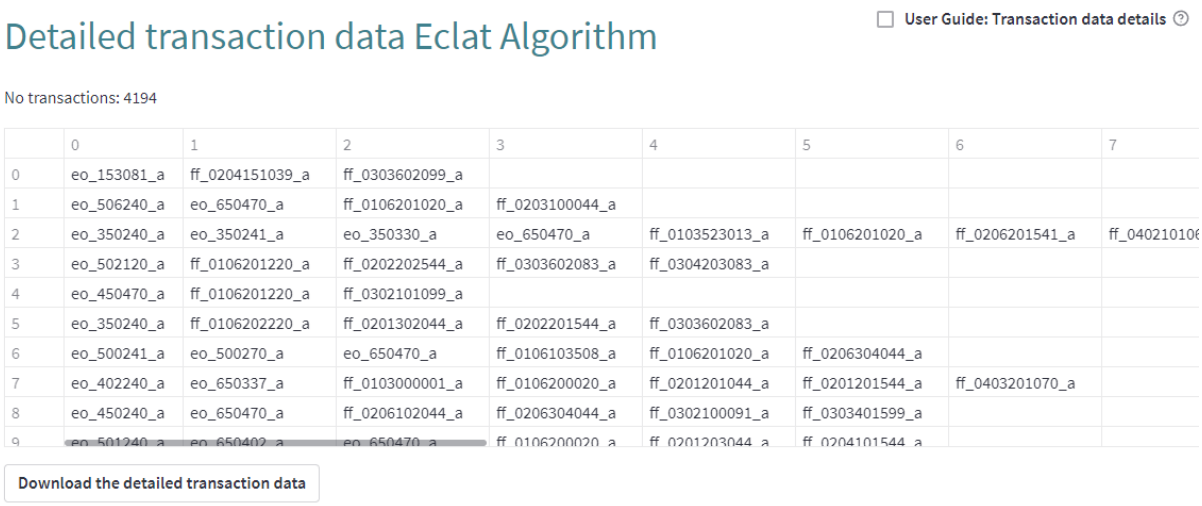


Figure 18: Checkbox to show the transaction data

xxx



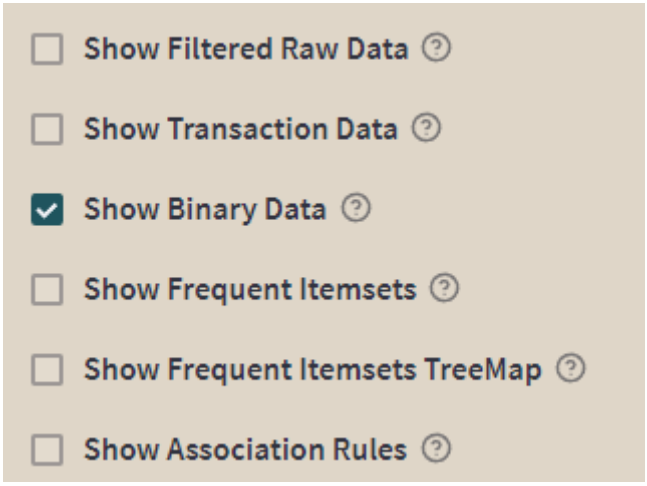


	0	1	2	3	4	5	6	7
0	eo_153081_a	ff_0204151039_a	ff_0303602099_a					
1	eo_506240_a	eo_650470_a	ff_0106201020_a	ff_0203100044_a				
2	eo_350240_a	eo_350241_a	eo_350330_a	eo_650470_a	ff_0103523013_a	ff_0106201020_a	ff_0206201541_a	ff_040210106
3	eo_502120_a	ff_0106201220_a	ff_0202202544_a	ff_0303602083_a	ff_0304203083_a			
4	eo_450470_a	ff_0106201220_a	ff_0302101099_a					
5	eo_350240_a	ff_0106202220_a	ff_0201302044_a	ff_0202201544_a	ff_0303602083_a			
6	eo_500241_a	eo_500270_a	eo_650470_a	ff_0106103508_a	ff_0106201020_a	ff_0206304044_a		
7	eo_402240_a	eo_650337_a	ff_0103000001_a	ff_0106200020_a	ff_0201201044_a	ff_0201201544_a	ff_0403201070_a	
8	eo_450240_a	eo_650470_a	ff_0206102044_a	ff_0206304044_a	ff_0302100091_a	ff_0303401599_a		
9	eo_501240_a	eo_650402_a	eo_650470_a	ff_0106200020_a	ff_0201203044_a	ff_0204101544_a		

ix. Binary Data

xxxx

xxx



Binary data Eclat Algorithm

User Guide: Binary data

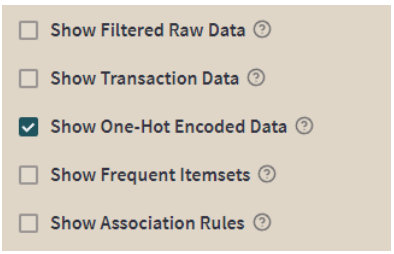
No transactions: 4194 - no columns: 2508

	eo_403337_a		eo_553490_a	eo_350282_a	ff_0102256301_a	eo_402241_a	ff_0106204221_a	ff_0102340001_a	ff_010353470
0	0	1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0
3	0	1	0	0	0	0	0	0	0
4	0	1	0	0	0	0	0	0	0
5	0	1	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0
7	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0
9	0	1	0	0	0	0	0	0	0

Download the binary data

x. One-Hot Encoded Data

xxxx



xxx

Detailed one-hot encoded data Non-Eclat Algorithm

User Guide: Binary data one-hot encoded

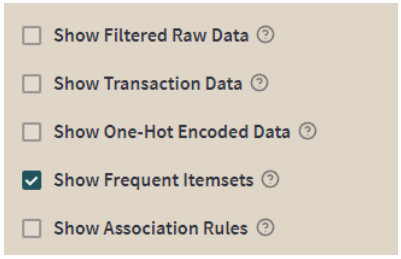
No transactions: 4194 - no columns: 2507

	eo_100000_a	eo_100010_a	eo_100020_a	eo_100030_a	eo_100040_a	eo_100050_a	eo_100080_a	eo_100081_a	eo_100110_a	eo_100193_a
1	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0

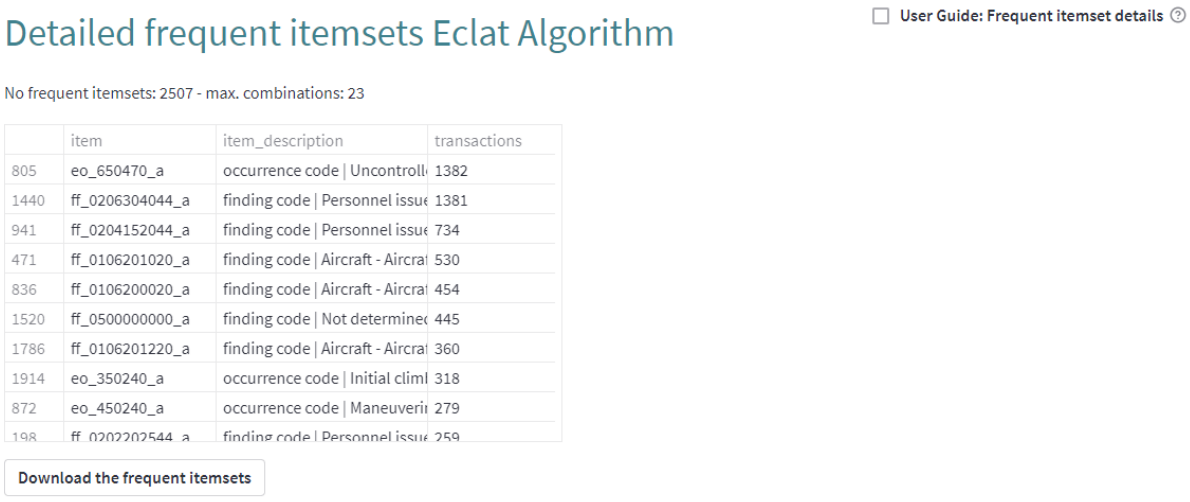
Download the one-hot encoded data

xi. Frequent Item Sets

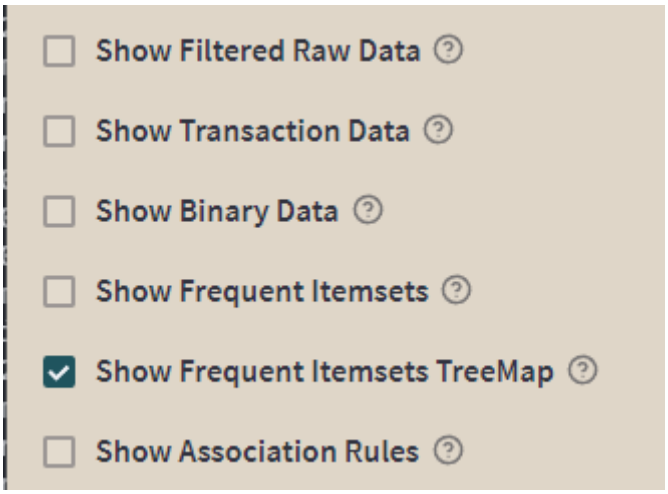
xxxx



xxx

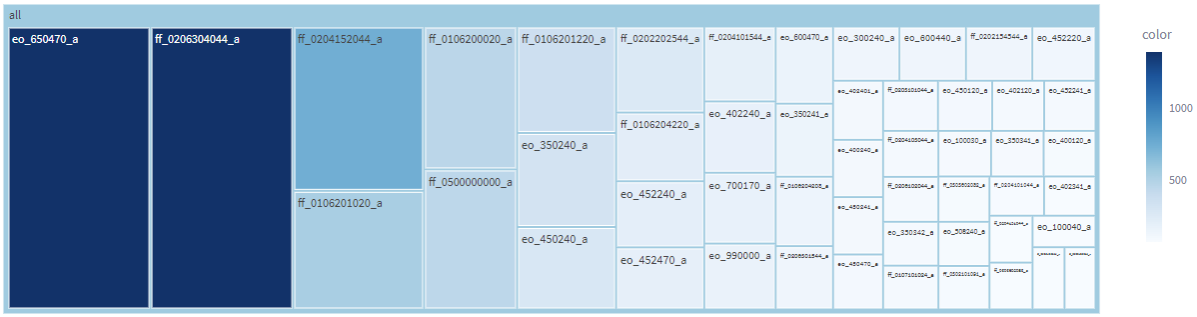


xii. Frequent Item Sets TreeMap



Frequent itemsets tree map Eclat Algorithm

User Guide: Frequent itemsets tree map



Detailed frequent itemsets Apriori Algorithm

User Guide: Frequent itemset details

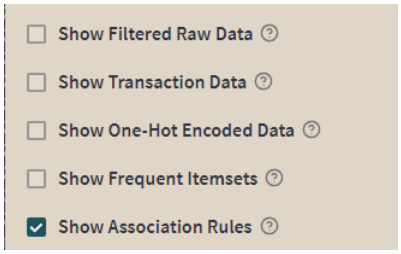
No frequent itemsets: 7

	support	itemsets	itemsets_description
0	0.3295	frozenset({'eo_650470_a'})	['occurrence code | Uncontrol
4	0.3293	frozenset({'ff_0206304044_a'	['finding code | Personnel issu
6	0.1803	frozenset({'ff_0206304044_a',	['finding code | Personnel issu
3	0.1750	frozenset({'ff_0204152044_a'	['finding code | Personnel issu
2	0.1264	frozenset({'ff_0106201020_a'	['finding code | Aircraft - Aircr
1	0.1082	frozenset({'ff_0106200020_a'	['finding code | Aircraft - Aircr
5	0.1061	frozenset({'ff_0500000000_a'	['finding code | Not determine

Download the frequent itemsets

xiii. Association Rules

XXXX



XXX

Detailed association rules Apriori Algorithm

User Guide: Association rule details

No association rules: 2

	antecedents	antecedents_description	consequents	consequents_description	antecedent support	consequent support	supp
0	frozenset({'ff_0206304044_a'	['finding code | Personnel issu	frozenset({'eo_650470_a'})	['occurrence code | Uncontro	0.3293	0.3295	0.180
1	frozenset({'eo_650470_a'})	['occurrence code | Uncontro	frozenset({'ff_0206304044_a'	['finding code | Personnel issu	0.3295	0.3293	0.180

Download the association rules

Detailed association rules Eclat Algorithm

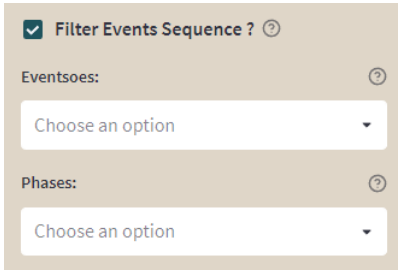
User Guide: Association rule details

No association rules: 7

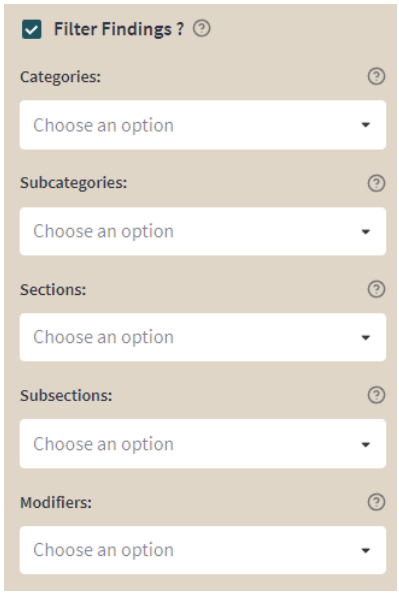
	Item	Item_description	Support
1	& eo_650470_a	['occurrence code | Uncontro	0.3295
4	& ff_0206304044_a	['finding code | Personnel issu	0.3293
6	eo_650470_a & ff_020630404	['occurrence code | Uncontro	0.1803
3	& ff_0204152044_a	['finding code | Personnel issu	0.1750
0	& ff_0106201020_a	['finding code | Aircraft - Aircr	0.1264
2	& ff_0106200020_a	['finding code | Aircraft - Aircr	0.1082
5	& ff_0500000000_a	['finding code | Not determine	0.1061

Download the association rules

c. Filter Events Sequence

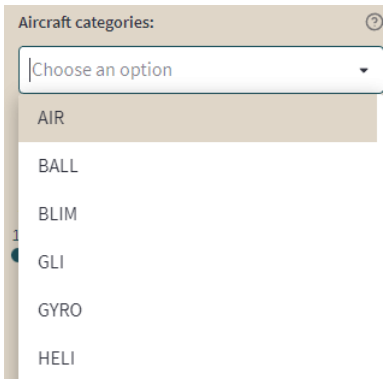


d. Filter Findings

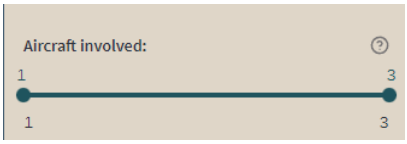


e. Filter Other Criteria

i. Aircraft categories



ii. Aircraft involved



4. Application Aviation Event Analysis

a. Introduction

US Aviation Fatal Accident Analysis is an analysis of fatal aviation accidents that have occurred in the United States. The analysis involves examining data related to fatal accidents, including the number of accidents, fatalities, and the causes of accidents.

The analysis is important because it helps to identify trends and patterns in aviation accidents, which can be used to improve aviation safety. The analysis can also be used to identify areas where additional research or training may be needed to prevent future accidents.

Aviation safety is a critical concern, and the US Aviation Fatal Accident Analysis plays an important role in improving safety by identifying areas of concern and developing strategies to prevent future accidents.

b. Processing Options

i. Run the Data Analysis

xxx

Figure 19: Checkbox to run the data analysis

xxx

ii. Extended Version

iii. Map

iv. Data Graphs- Years

Fatalities per Year by FAR Operations Parts
Fatalities per Year by Selected FAR Operations Parts
Events per Year by CICTT Codes
Events per Year by Event Types
Events per Year by Highest Injury Levels
Events per Year by Main Phases of Flight
Events per Year by Nearest Airport
Events per Year by Phases of Flight
Events per Year by Safety Systems
Events per Year by Top Level Logical Parameters

v. Data Graphs- Totals

Total Fatalities by FAR Operations Parts
Total Fatalities by Selected FAR Operations Parts
Total Events by CICTT Codes
Total Events by Event Types
Total Events by Highest Injury Levels
Total Events by Main Phases of Flight
Total Events by Nearest Airport
Total Events by Phases of Flight
Total Events by Safety Systems
Total Events by Top Level Logical Parameters

vi. Data Graphs- Distances

Distance to the Nearest Airport

vii. Data Profile

viii. Detailed Data

c. Filter Options

i. Standard Version

ii. Extended Version

5. Application Database Profiling

a. Introduction

Figure 20: Header Database Profiling

Data profiling is the process of examining and analyzing data from various sources to understand the structure, content, relationships, and quality of the data. The goal of data profiling is to create a comprehensive understanding of the data so that it can be used effectively for analysis, decision-making, and other purposes. Data profiling involves collecting information about the data, such as its size, format, data types, completeness, uniqueness, patterns, and outliers. This information can be used to identify data quality issues, such as missing values, inconsistent data, or data that does not conform to expected standards.

ydata-profiling is a Python library that generates an interactive report from a Pandas data frame. The report includes various statistics and visualizations that provide insights into the data. It helps in understanding the data distribution, identifying missing values, detecting outliers, and much more. ydata-profiling offers a fast and easy way to get an overview of a dataset. It provides a comprehensive report that includes details about the data types, distribution, missing values, correlations, and much more. The report is generated in HTML format and can be easily exported to different formats.

b. Processing Options

i. Data Profile

This checkbox triggers profiling of the previously selected database table or database view and creating a report.

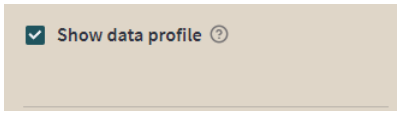


Figure 21: Checkbox to show the data profile

The **`Overview`** shows mostly global details about the dataset (number of records, number of variables, overall missingness and duplicates, memory footprint).

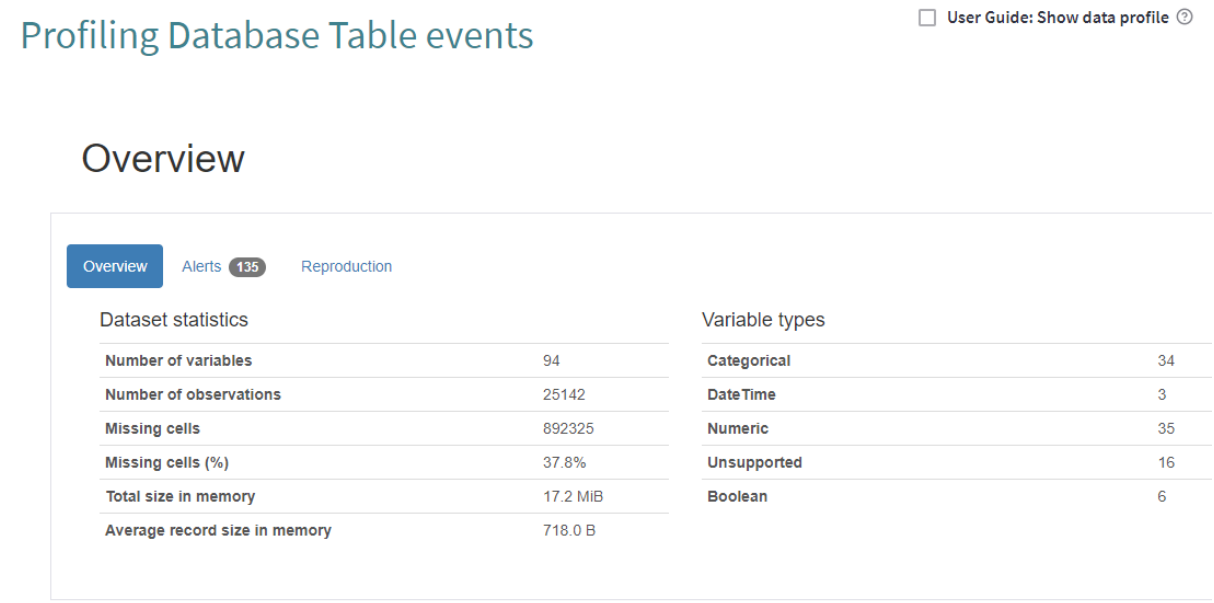


Figure 22: Data profile summary

The **`Overview`** is followed by the **`Variables`** section, which contains the detailed information for each column. The **`Select Columns`** select box at the top allows you to select a specific column.

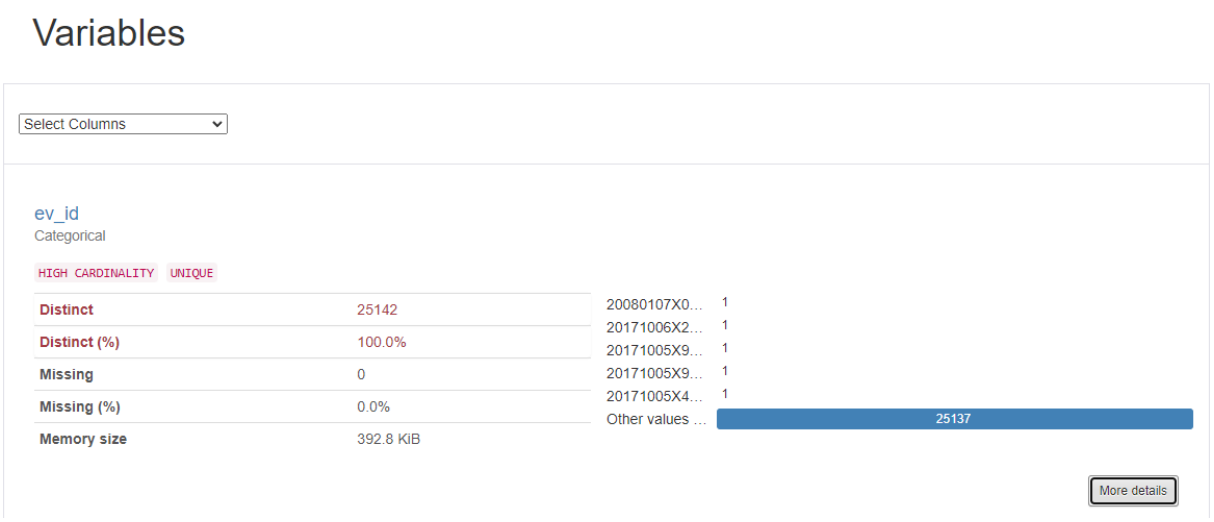


Figure 23: Simple view of column profile

More information about a specific column can be retrieved by clicking the **`More Details`** button. The additional information shown is located in the two tabs **`Overview`** and **`Categories`**.

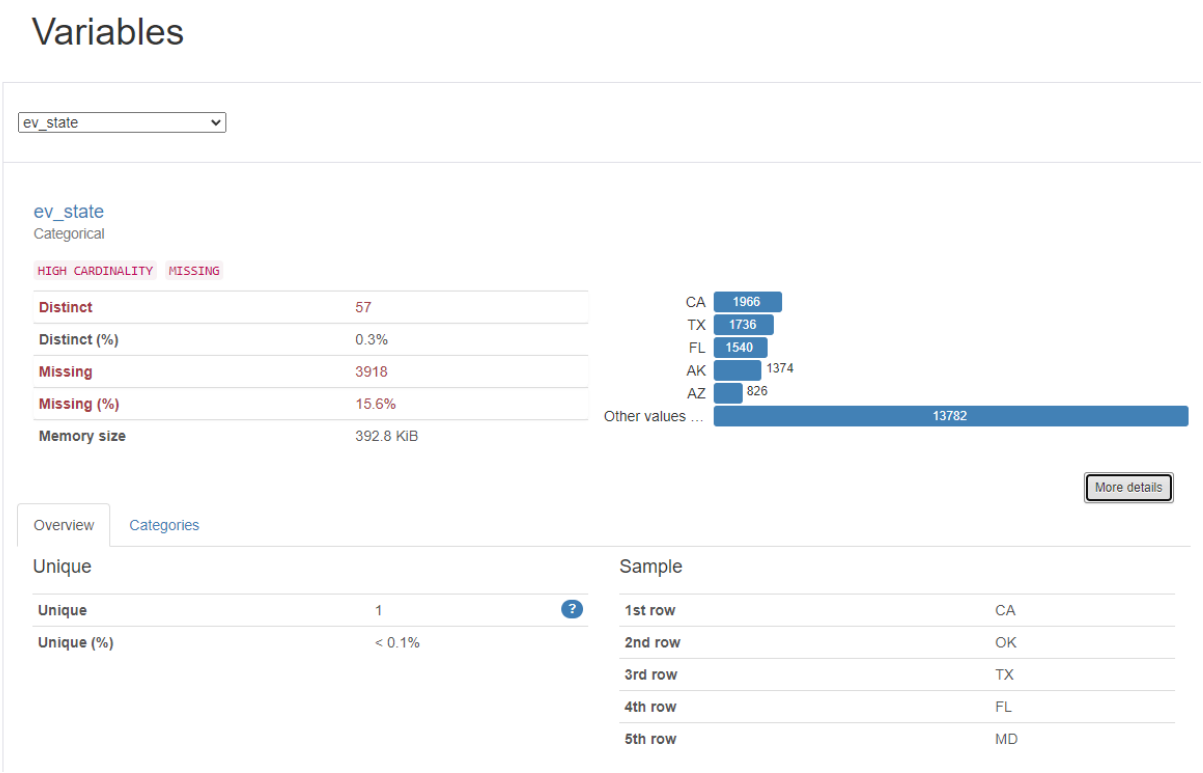


Figure 24: Extended view of column profile

Columns whose contents cannot be evaluated are shown as follows:

Figure 25: Example of an unsupported column

The **`Download the profile report`** button at the end of the report can be used to download an HTML version of the profiling report to the local computer.

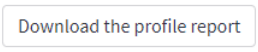


Figure 26: Download button

Further details can be found in the ydata-profiling documentation.

ii. Detailed Data

With this checkbox the detailed data for the selected database table or view can be displayed.

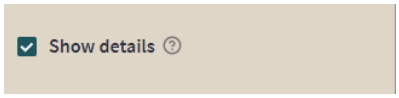


Figure 27: Checkbox to show the detailed data

The data is displayed in a table format, the so-called data frame. The data frame shows only the data that is still available after applying the selected filter options. The data in the data frame can also be downloaded as a CSV file using the '**`Download all data as CSV file`**' button.

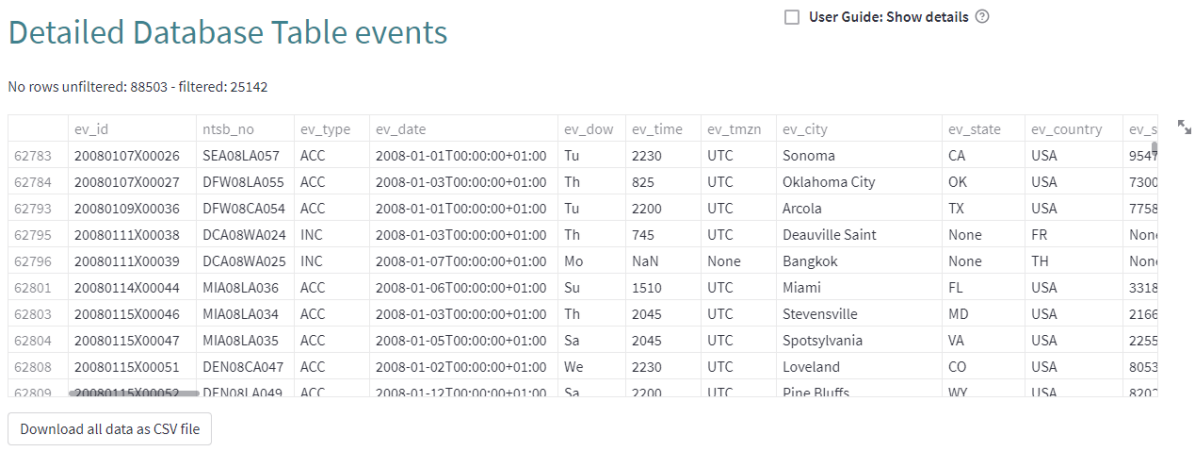


Figure 28: Data frame

More detailed information on how to use the data frame can be found in section '**`2.b General Usability`**'.

c. Filter Options

`Database tables and views`: A database table or database view to be analyzed can be selected here. The default value is the database table **`event`**.

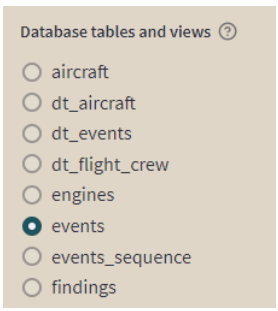


Figure 29: Radio button to select a database table or database view

`Event year(s)`: A range of years between 1982 and today can be selected here via a slider widget. However, the selection made here only has an effect on those database tables and database views that either contain a column **`ev_year`** or contain a foreign key to the database table **`event`**.

Figure 30: Slider to select an interval of event years

6. Application US Aviation Fatal Accident Analysis

a. Introduction

US Aviation Fatal Accident Analysis is an analysis of fatal aviation accidents that have occurred in the United States. The analysis involves examining data related to fatal accidents, including the number of accidents, fatalities, and the causes of accidents.

The analysis is important because it helps to identify trends and patterns in aviation accidents, which can be used to improve aviation safety. The analysis can also be used to identify areas where additional research or training may be needed to prevent future accidents.

Aviation safety is a critical concern, and the US Aviation Fatal Accident Analysis plays an important role in improving safety by identifying areas of concern and developing strategies to prevent future accidents.

This is a data and functionally limited version of the 'Aviation Event Analysis' application.

The database view **`io_app_ae1982`** is used for the data. The data limitations concern:

- only events of type accident,
- only operations of type charter services (parts 135), regular scheduled air carriers (parts 121), or general aviation (parts 091x),
- only U.S. related fatal accidents from 2008 to present, i.e., accidents on U.S. soil, departure or planned arrival in the U.S., U.S. owner, U.S. operator, or U.S. registration.

Functionality is limited to:

- total and annual views on fatalities by selected FAR operations parts,
- total and annual views on preventable accidents by safety system, and
- the fatal accident map.

The application is divided into two parts: On the left side there is the so-called sidebar and on the right side the results are displayed. The sidebar has the functional controls in the upper part and the controls for filtering the data in the lower part. Each time the functional or filtering controls are changed, all the results on the right side are recalculated.

b. Processing Options

i. Map

Figure 31: Functional control

The map shows the places where fatal aviation accidents occurred in the selected period. Each point represents at least one accident with fatalities. If you hover the mouse over such a point, you will get detailed information about the accident behind it. However, only those accidents can be shown here for which a decimal longitude and latitude have been entered. The map can be zooming in and out as needed. Furthermore, the map can also be printed out.

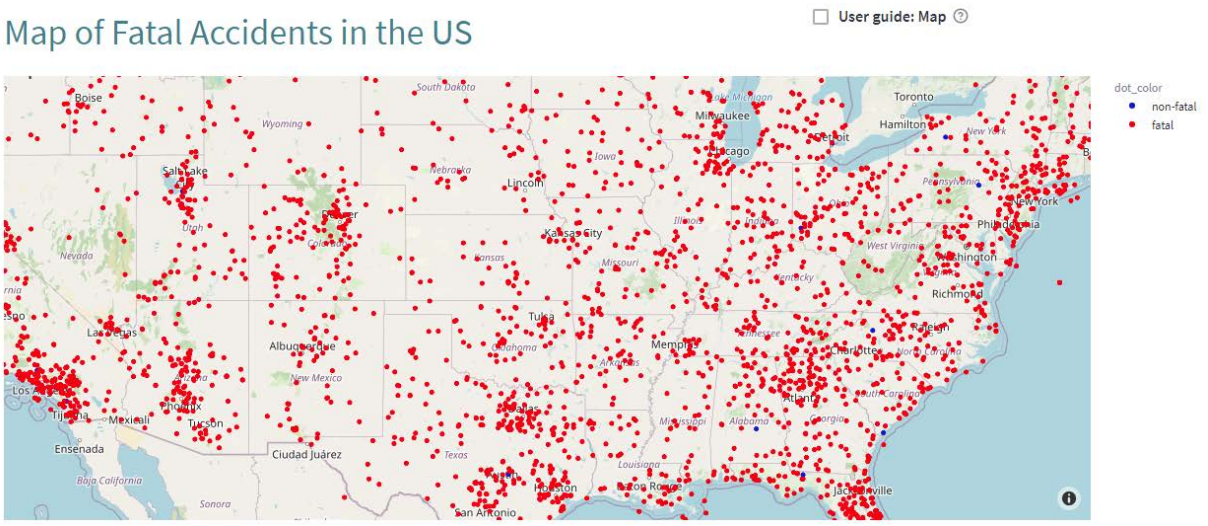


Figure 32: US map

ii. Fatality-based Charts

These analyses are based on U.S. aviation fatalities.

The selected data can be displayed in up to three different chart types:

- The vertical bar chart shows the annual values,
- Pie chart and horizontal bar chart show the total values.

Selected FAR Operations Parts

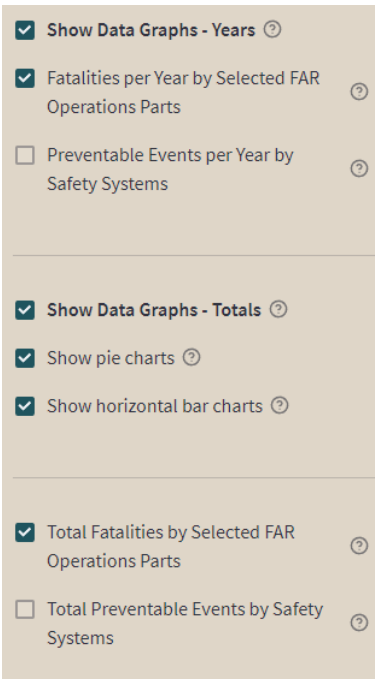


Figure 33: Functional controls

The fatalities processed here result exclusively from accidents that can be assigned to one of the following FAR operations parts:

- Parts 091x General operations
- Parts 121 Regular scheduled air carriers
- Parts 135 Charter type services

Since the assignment is made at the aircraft, accidents with multiple aircraft involved and different FAR Operations Parts may result in multiple fatality counts for the accidents involved. However, this only affects a negligible number of accidents.

Number of Fatalities per Year by Selected FAR Operations Parts

User guide: Years chart

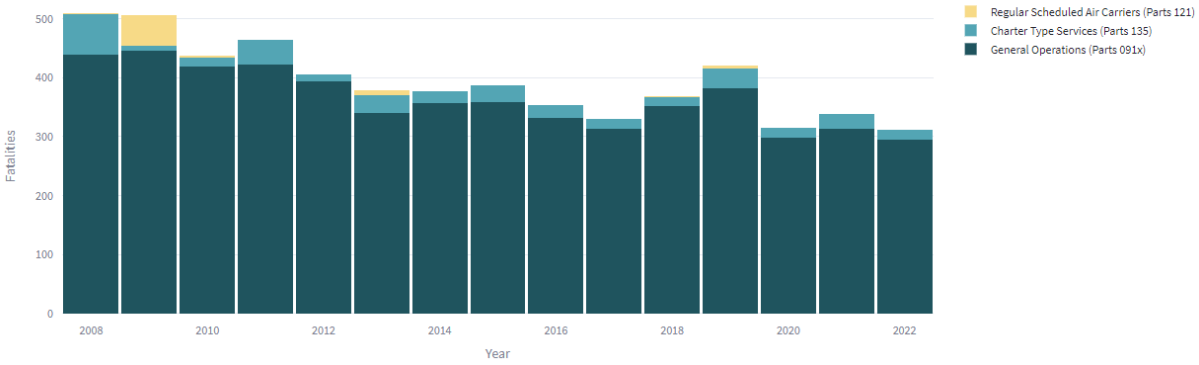


Figure 34: Bar chart representation of annual values

Total Number of Fatalities by Selected FAR Operations Parts

User guide: Totals chart

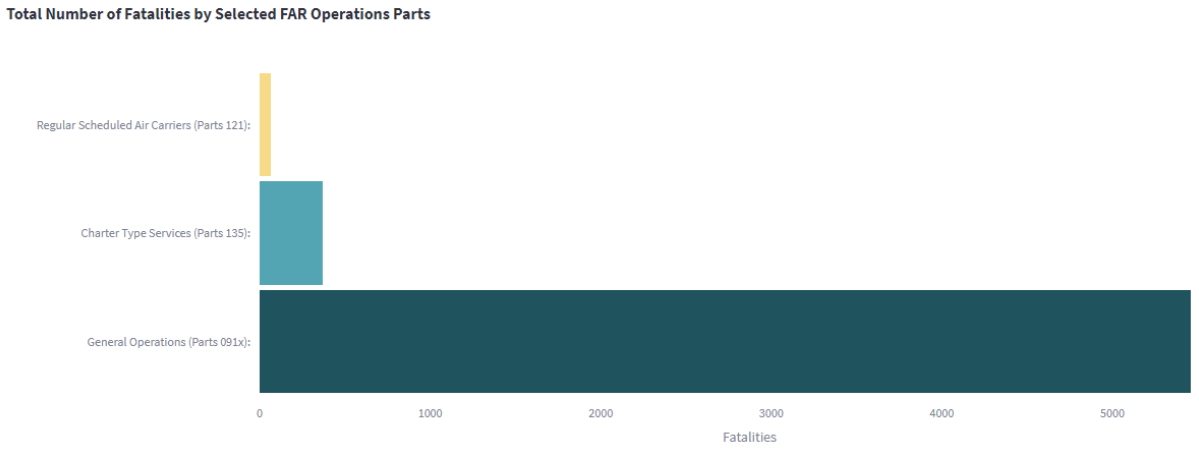


Figure 35: Horizontal bar chart representation of totals

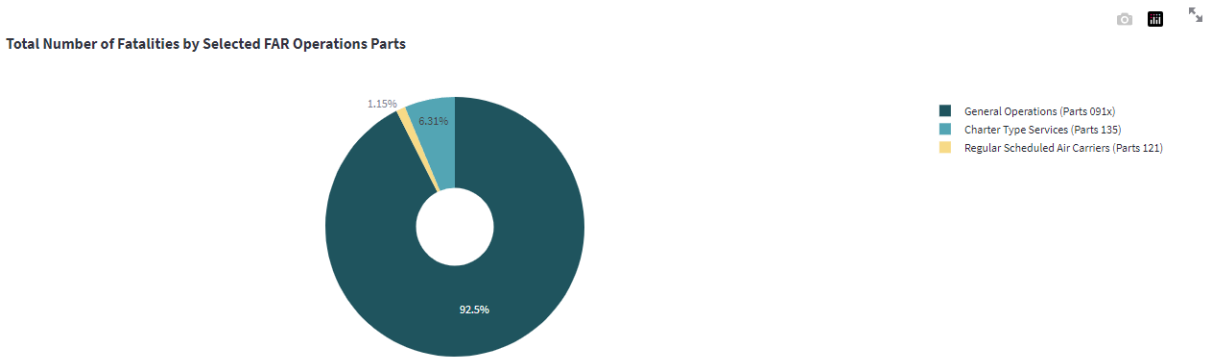


Figure 36: Pie chart representation of totals

iii. Accident-based Charts

These analyses are based on fatal accidents in U.S. aviation.

The selected data can be displayed in up to three different chart types:

- The vertical bar chart shows the annual values,
- Pie chart and horizontal bar chart show the total values.

Preventable Accidents by Safety Systems

The percentage threshold limits the individual display of the affected safety systems. The safety systems whose percentage is below the threshold are grouped in the **`below threshold`** category.

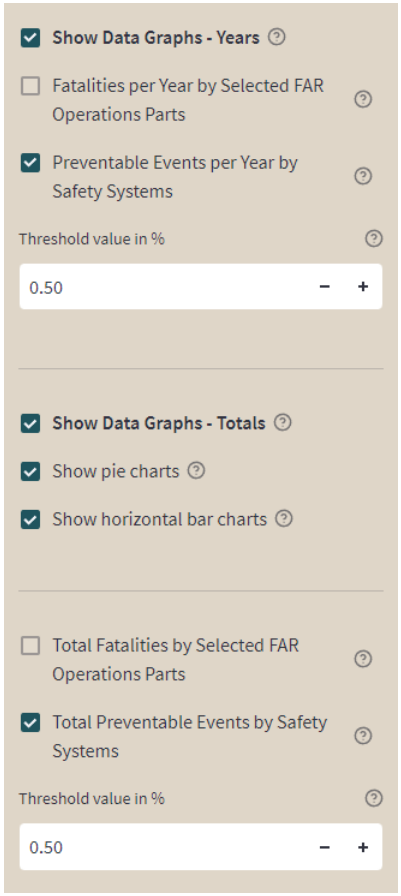


Figure 37: Functional controls

The high-level security system requirements defined in [Hook, Loyd & Sizoo, David & Fuller, Justin. (2022)] are mapped to the data in **`IO-AVSTATS-DB`** as follows:

- Airbone collision **`is_midair_collision`**
- Forced landing **`is_rss_forced_landing`** i.e., **`is_attitude controllable`** and **`is_emergency_landing`**
- Spin / stall **`is_rss_spin_stall_prevention_and_recovery`** i.e., **`is_attitude_controllable`** and **`is_spin_stall`**
- Terrain collision **`is_rss_terrain_collision_avoidance`** i.e., **`is_attitude_controllable`** and **`is_altitude_low`** and **`is_altitude_controllable`**

This data is mainly based on the database tables **`event_sequence`** and **`findings`**. However, since these data are not unique per event and moreover, several aircraft can be involved in an event, combinations of safety systems can occur, e.g.: **`Forced landing, Spin / stall`**.

Number of Preventable Accidents per Year by Safety Systems

User guide: Years chart

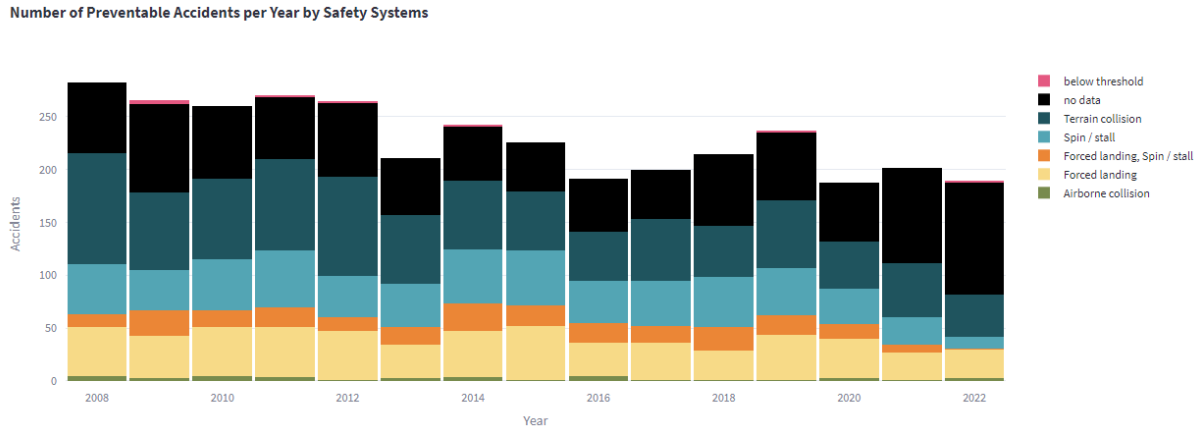


Figure 38: Bar chart representation of annual values

Total Number of Preventable Accidents by Safety Systems

User guide: Totals chart

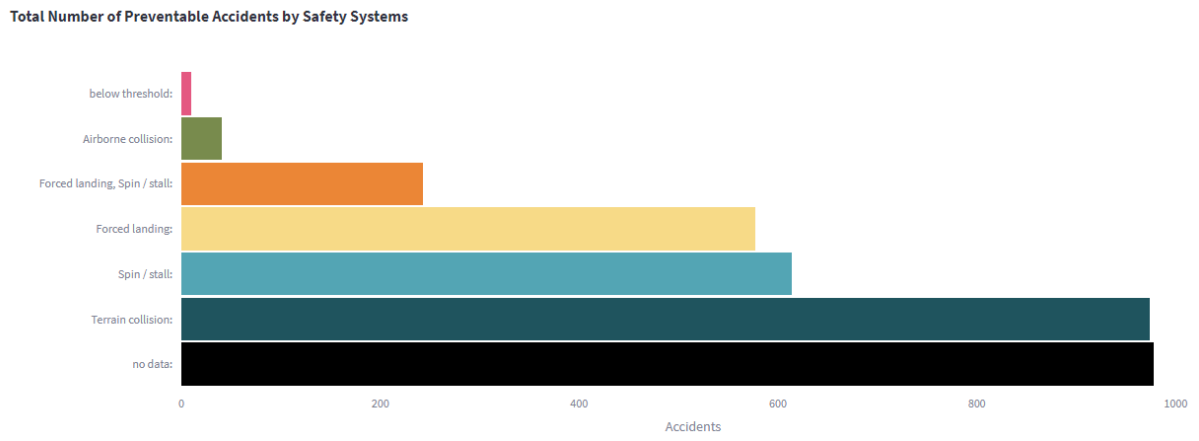


Figure 39: Horizontal bar chart representation of totals

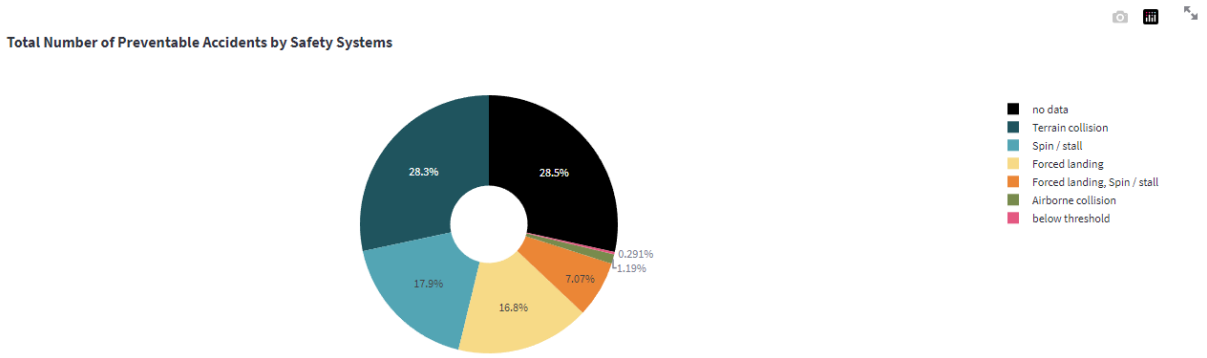


Figure 40: Pie chart representation of totals

c. Filter Options

A range of years between 2008 and today can be selected here via this slider widget:

Figure 41: Slider to select an interval of event years

7. Change Log

a. Release 23.06.15

- **NTSB file included:**
  - up15JUN.zip - 6/15/2023 3:01:48 AM

b. Release 23.06.08

- **NTSB file included:**
  - up08JUN.zip - 6/ 8/2023 3:00:25 AM

c. Release 23.06.01

- **NTSB files included:**
  - avall.zip - 6/ 1/2023 6:07:07 AM
  - up01JUN.zip - 6/ 1/2023 3:00:19 AM
- **ZIP Code Database file included:**
  - June 2023

d. Release 23.05.22

- **NTSB file included:**
  - up22MAY.zip - 5/22/2023 3:00:10 AM

e. Release 23.05.15

- **NTSB file included:**
  - up15MAY.zip - 5/15/2023 3:00:19 AM

f. Release 23.05.08

- **NTSB file included:**
  - up08MAY.zip - 5/ 8/2023 3:00:17 AM

g. Release 23.05.01

- **FAA Airport file included:**
  - 2023.04.20_Airports.csv (20. April 2023)
- **FAA Runway file included:**
  - 2023.04.20_Runways.csv (20. April 2023)
- **NTSB files included:**
  - Pre2008.zip - 9/30/2020 12:51:56 PM
  - avall.zip - 5/ 1/2023 5:58:06 AM
  - up01MAY.zip - 5/ 1/2023 3:00:26 AM

h. Release 23.04.22

- **National Plan of Integrated Airport Systems (NPIAS) included:**
  - NPIAS-2023-2027-Appendix-A.xlsx
- **NTSB file included:**
  - Up22APR.zip - 4/22/2023 3:30 AM

i. Release 23.04.15

- **FAA Airports file included:**
  - 2023.02.23_Airports.csv (23. February 2023)
- **NTSB file included:**
  - Up15APR.zip - 4/15/2023 3:00:12 AM
- **simplemaps files included:**
  - simplemaps_uscities_basicv1.76
  - simplemaps_uszips_basicv1.82
- **United States Zip Codes.org file included:**
  - zip_code_database.xls (42735 entries)

j. Release 23.04.08

- **NTSB files included:**
  - Pre2008.zip - 9/30/2020 12:51:56 PM
  - avall.zip   - 4/ 3/2023 8:13:22 AM
  - up08APR.zip - 4/ 8/2023 3:01:20 AM

k. Release 23.04.01

- **NTSB files included:**
  - Pre2008.zip - 9/30/2020 12:51:56 PM
  - avall.zip   - 3/ 1/2023 6:01:23 AM
  - up08MAR.zip - 3/ 8/2023 3:00:22 AM
  - up15MAR.zip - 3/15/2023 3:00:11 AM
  - up22MAR.zip - 3/22/2023 3:00:14 AM
  - up01APR.zip - 4/ 1/2023 3:00:31 AM

8. References

data.ntsb.gov. (n.d.). MDB Download Directory - NTSB.ADMS.DataTransfer.Web. [online] Available at: https://data.ntsb.gov/avdata.

Dias, J.R. (2023). *pyECLAT*. [online] GitHub. Available at: https://github.com/jeffrichardchemistry/pyECLAT [Accessed 14 May 2023].

Hook, Loyd & Sizoo, David & Fuller, Justin. (2022). How Digital Safety Systems Could Revolutionize Aviation Safety. pp. 1-9. 10.1109/DASC55683.2022.9925863. Available at: https://www.researchgate.net/publication/365100080_How_Digital_Safety_Systems_Could_Revolutionize_Aviation_Safety

rasbt.github.io. (n.d.). *Apriori - mlxtend*. [online] Available at: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/apriori/ [Accessed 14 May 2023].

rasbt.github.io. (n.d.). *Fpgrowth - mlxtend*. [online] Available at: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/fpgrowth/ [Accessed 14 May 2023].

rasbt.github.io. (n.d.). *Fpmax - mlxtend*. [online] Available at: https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/fpmax/ [Accessed 14 May 2023].

streamlit.io. (n.d.). *Streamlit • The fastest way to build and share data apps*. [online] Available at: https://streamlit.io/.