

Dataset Filtering and Cleaning

21/03/2025

stratified_history.csv
8.1 MB

Analyze the provided dataset to identify its key characteristics for data-driven decision-making. Upon receiving a specific analytical question, address it strictly using the dataset. Your deliverables must include a clear, concise answer to the question and the Python code that extracts and processes the data used to derive your answer. The dataset should be cleaned by filtering admission_period to include only values between 2006.1 and 2019.2, removing entries where enrollment_type is labeled as "Exempted", and excluding those where situation is "In Progress". The resulting cleaned dataset should be stored in a new DataFrame named df_filtered.

qwen-max-latest

To analyze the dataset effectively and address your requirements, I will perform the following steps:

1. **Load the Dataset** : Read the CSV file into a pandas DataFrame.
2. **Filter the Data** :
 - Retain only rows where admission_period is between 2006.1 and 2019.2 (inclusive).
 - Remove rows where enrollment_type is labeled as "Exempted".
 - Exclude rows where situation is "In Progress".
3. **Store the Cleaned Data** : Save the filtered data into a new DataFrame named df_filtered .
4. **Provide Python Code** : Include the Python code used for these operations.

Here is the Python code to achieve this:

```
python
1 import pandas as pd
2
3 # Load the dataset
4 file_path = 'stratified_history.csv'
5 df = pd.read_csv(file_path)
6
7 # Filter the data based on the specified conditions
8 df_filtered = df[
9     (df['admission_period'] >= 2006.1) &
10    (df['admission_period'] <= 2019.2) &
11    (df['enrollment_type'] != 'Exempted') &
12    (df['situation'] != 'In Progress')
13 ]
14
15 # Display basic information about the cleaned dataset
16 print("Shape of the filtered dataset:", df_filtered.shape)
17 print("\nColumns in the filtered dataset:")
18 print(df_filtered.columns.tolist())
19
20 # Optionally, save the cleaned dataset to a new CSV file
21 df_filtered.to_csv('filtered_dataset.csv', index=False)
```

Explanation of the Code:

1. **Loading the Dataset** : The dataset is loaded using pd.read_csv() .
2. **Filtering Conditions** :
 - (df['admission_period'] >= 2006.1) & (df['admission_period'] <= 2019.2) ensures that only rows with admission_period within the specified range are retained.
 - (df['enrollment_type'] != 'Exempted') removes rows where enrollment_type is "Exempted".
 - (df['situation'] != 'In Progress') excludes rows where situation is "In Progress".
3. **Storing the Cleaned Data** : The filtered data is stored in df_filtered .
4. **Saving the Cleaned Dataset** : Optionally, the cleaned dataset is saved to a new CSV file (filtered_dataset.csv) for further use.

Key Characteristics of the Filtered Dataset:

- The dataset is now limited to records from the academic periods 2006.1 to 2019.2.
- It excludes students who were exempted or whose enrollment status was marked as "In Progress".
- This cleaned dataset is suitable for further analysis, such as identifying trends, patterns, or making data-driven decisions.

If you have a specific analytical question, please provide it, and I will address it using the cleaned dataset (df_filtered).

What are the maximum and minimum ages recorded for students at the time of dropout, and how do these values compare to the mean and standard deviation of student ages in the program?

To answer the question, we need to analyze the cleaned dataset (`df_filtered`) to compute the following statistics related to student ages at the time of dropout:

1. **Maximum Age** : The highest age recorded for students who dropped out.
2. **Minimum Age** : The lowest age recorded for students who dropped out.
3. **Mean Age** : The average age of students who dropped out.
4. **Standard Deviation** : A measure of the spread or variability of student ages around the mean.

Steps to Solve:

1. **Filter for Dropouts** : Extract rows where the `situation` column indicates "Dropout".
2. **Extract Age Data** : Use the `age_at_dropout` column to calculate the required statistics.
3. **Compute Statistics** :
 - Maximum and minimum values using `.max()` and `.min()`.
 - Mean and standard deviation using `.mean()` and `.std()`.

Here is the Python code to compute these values:

```
python
1 # Filter for rows where the situation is "Dropout"
2 dropout_data = df_filtered[df_filtered['situation'] == 'Dropout']
3
4 # Extract the age_at_dropout column
5 ages_at_dropout = dropout_data['age_at_dropout']
6
7 # Compute the required statistics
8 max_age = ages_at_dropout.max()
9 min_age = ages_at_dropout.min()
10 mean_age = ages_at_dropout.mean()
11 std_dev_age = ages_at_dropout.std()
12
13 # Display the results
14 print("Maximum Age at Dropout:", max_age)
15 print("Minimum Age at Dropout:", min_age)
16 print("Mean Age at Dropout:", mean_age)
17 print("Standard Deviation of Age at Dropout:", std_dev_age)
```

Explanation of the Code:

1. **Filtering for Dropouts** : We use `df_filtered[df_filtered['situation'] == 'Dropout']` to isolate rows where the `situation` column equals "Dropout".
2. **Extracting Ages** : The `age_at_dropout` column contains the ages of students at the time of dropout.
3. **Computing Statistics** :
 - `.max()` computes the maximum value in the column.
 - `.min()` computes the minimum value in the column.
 - `.mean()` calculates the average (mean) of the column.
 - `.std()` calculates the standard deviation of the column.

Results:

After running the code, you will obtain the following outputs:

- **Maximum Age at Dropout** : The oldest age recorded for a student who dropped out.
- **Minimum Age at Dropout** : The youngest age recorded for a student who dropped out.
- **Mean Age at Dropout** : The average age of students who dropped out.
- **Standard Deviation of Age at Dropout** : A measure of how much the ages vary around the mean.

Interpretation:

- Compare the **maximum** and **minimum** ages to understand the range of ages among dropouts.
- Use the **mean** and **standard deviation** to assess the central tendency and variability of the age distribution.

If you run the code with the provided dataset, I can help interpret the results further. Let me know if you'd like me to proceed with any additional analysis!

