Dataset Cleaning and Analysis

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 provided dataset and answer specific analytical questions, we will first clean the dataset according to the specified criteria:

1. Filter `admission_period` to include only values between 2006.1 and 2019.2.
2. Remove entries where `enrollment_type` is labeled as "Exempted".
3. Exclude entries where `situation` is "In Progress".

The cleaned dataset will be stored in a new DataFrame named `df_filtered`. Below is the Python code to perform these steps, followed by an example of how to address a specific analytical question using the cleaned dataset.

Python Code for Data Cleaning

python

```python
import pandas as pd

# Load the dataset
file_path = 'stratified_history.csv'
df = pd.read_csv(file_path)

# Clean the dataset based on the specified criteria
df_filtered = df[
    (df['admission_period'] >= 2006.1) &
    (df['admission_period'] <= 2019.2) &
    (df['enrollment_type'] != 'Exempted') &
    (df['situation'] != 'In Progress')
]

# Display the first few rows of the cleaned dataset
print(df_filtered.head())
```

Example Analytical Question and Solution

Question:

What is the average grade (`final_grade`) of students who graduated (`situation == 'Graduate'`) from the "Academic Unit of Systems and Computing" (`academic_unit`) in the cleaned dataset?

Solution:

We will filter the `df_filtered` DataFrame to include only rows where `situation` is "Graduate" and `academic_unit` is "Academic Unit of Systems and Computing". Then, we calculate the mean of the `final_grade` column.

python

```python
# Filter for graduates from the specified academic unit
graduates_systems_computing = df_filtered[
    (df_filtered['situation'] == 'Graduate') &
    (df_filtered['academic_unit'] == 'Academic Unit of Systems and Computing')
]

# Calculate the average final grade
average_grade = graduates_systems_computing['final_grade'].mean()

# Print the result
print(f"The average grade of graduates from the Academic Unit of Systems and Computing is: {aver
```

Explanation of the Code

1. **Data Cleaning** :
    - The dataset is filtered to include only rows where `admission_period` is between 2006.1 and 2019.2.

   - Rows with `enrollment_type` labeled as "Exempted" are removed.
   - Rows where `situation` is "In Progress" are excluded.
2. **Analytical Query** :
   - The cleaned dataset is further filtered to focus on students who graduated (`situation == 'Graduate'`) and belong to the "Academic Unit of Systems and Computing".
   - The mean of the `final_grade` column is calculated to determine the average grade.

This approach ensures that the analysis is performed on a clean and relevant subset of the data, providing accurate insights for data-driven decision-making.

What is the proportion of student dropouts for each year of admission, considering dropout rate as a proportion of total admissions in that year?

qwen-max-latest

To calculate the proportion of student dropouts for each year of admission, we need to:

1. Group the data by `admission_year`.
2. Count the total number of students admitted in each year.
3. Count the number of students who dropped out (`situation == 'Dropout'`) in each year.
4. Compute the dropout rate as the ratio of dropouts to total admissions for each year.

Here is the Python code to compute this:

```python
import pandas as pd

# Load the dataset
file_path = 'stratified_history.csv'
df = pd.read_csv(file_path)

# Clean the dataset based on the specified criteria
df_filtered = df[
    (df['admission_period'] >= 2006.1) &
    (df['admission_period'] <= 2019.2) &
    (df['enrollment_type'] != 'Exempted') &
    (df['situation'] != 'In Progress')
]

# Extract the admission year from admission_period (assuming it's in the format YYYY.X)
df_filtered['admission_year'] = df_filtered['admission_period'].astype(str).str.split('.').str[0

# Group by admission_year and calculate total admissions and dropouts
dropout_rate_by_year = df_filtered.groupby('admission_year').apply(
    lambda group: {
        'total_admissions': len(group),
        'dropouts': (group['situation'] == 'Dropout').sum(),
        'dropout_rate': (group['situation'] == 'Dropout').mean()
    }
).apply(pd.Series)  # Convert the result into a DataFrame

# Reset index for better readability
dropout_rate_by_year = dropout_rate_by_year.reset_index()

# Display the resulting DataFrame
print(dropout_rate_by_year)
```

Explanation of the Code:

1. **Data Cleaning** :
   - The dataset is filtered to include only rows where `admission_period` is between 2006.1 and 2019.2.
   - Rows with `enrollment_type` labeled as "Exempted" are removed.
   - Rows where `situation` is "In Progress" are excluded.
2. **Extract Admission Year** :
   - The `admission_period` column is split to extract the year (e.g., `2006.1` becomes `2006`).
3. **Group by Admission Year** :
   - For each admission year, the total number of students is counted.
   - The number of students who dropped out (`situation == 'Dropout'`) is counted.
   - The dropout rate is calculated as the proportion of dropouts to total admissions.
4. **Output** :
   - A DataFrame is created with columns: `admission_year`, `total_admissions`, `dropouts`, and `dropout_rate`.

Example Output:

Assuming the dataset contains sufficient data for years 2006 to 2019, the output might look like this:

ADMISSION_YEAR	TOTAL_ADMISSIONS	DROPOUTS	DROPOUT_RATE
2006	500	150	0.30
2007	600	200	0.33
2008	550	180	0.33
...	...	...	...

This table shows the dropout rate for each admission year, enabling analysis of trends over time.

copyright © 2025

Potencializado por Open WebUI