

# Dataset Cleaning and Analysis

22/03/2025



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 and prepare it for data-driven decision-making, we will first load the data, clean it according to the specified criteria, and store the cleaned dataset in a new DataFrame named `df_filtered`. Below is the Python code that performs these steps:

## Key Characteristics of the Dataset:

- Columns** : The dataset contains information about students, including their admission period, enrollment type, academic situation, demographics, grades, and course details.
- Timeframe** : Admission periods range from 1979 to 2020.
- Enrollment Types** : Includes Entrance Exam, Unified Selection System (SISU), ENEM, Transfer, and others.
- Academic Situations** : Includes "Approved", "Failed", "Dropout", "In Progress", etc.
- Demographics** : Includes gender, race, school background, and quota reservation status.

## Cleaning Criteria:

- Filter `admission_period` to include values between `2006.1` and `2019.2`.
- Exclude rows where `enrollment_type` is labeled as "Exempted".
- Exclude rows where `situation` is "In Progress".

## Python Code for Data Cleaning and Analysis

```
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 # Display initial dataset info
8 print("Initial Dataset Shape:", df.shape)
9
10 # Cleaning Step 1: Filter admission_period to include only values between 2006.1 and 2019.2
11 df_filtered = df[(df['admission_period'] >= 2006.1) & (df['admission_period'] <= 2019.2)]
12
13 # Cleaning Step 2: Remove entries where enrollment_type is labeled as "Exempted"
14 df_filtered = df_filtered[df_filtered['enrollment_type'] != 'Exempted']
15
16 # Cleaning Step 3: Exclude rows where situation is "In Progress"
17 df_filtered = df_filtered[df_filtered['situation'] != 'In Progress']
18
19 # Display cleaned dataset info
20 print("Filtered Dataset Shape:", df_filtered.shape)
21
22 # Save the cleaned dataset to a new DataFrame
23 df_filtered.head()
```

## Explanation of the Code:

- Loading the Dataset** : The dataset is loaded using `pd.read_csv()`.
- Filtering Admission Period** : Rows are filtered to include only those with `admission_period` values between `2006.1` and `2019.2`.
- Removing Exempted Entries** : Rows where `enrollment_type` is "Exempted" are removed.
- Excluding In Progress Situations** : Rows where `situation` is "In Progress" are excluded.

5. **Resulting Data-frame** : The cleaned dataset is stored in `df_filtered`.

**Deliverables:**

- 1. **Cleaned Dataset** : The resulting `df_filtered` contains only relevant rows based on the cleaning criteria.
- 2. **Shape Comparison** : The shape of the initial dataset and the filtered dataset is printed to show the impact of filtering.

**Example Output:**

If the initial dataset has 10,000 rows and the filtering reduces it to 8,000 rows, the output might look like this:

```
1 Initial Dataset Shape: (10000, 50)
2 Filtered Dataset Shape: (8000, 50)
```

This cleaned dataset (`df_filtered`) can now be used for further analysis or to address specific analytical questions. If you have a specific question, please provide it, and I will extract and process the data accordingly.

For students who entered through affirmative action, what is the 70th percentile of the age of entry?



qwen-max-latest

To calculate the 70th percentile of the age of entry for students who entered through affirmative action, we need to filter the cleaned dataset (`df_filtered`) to include only rows where `affirmative_action` is "Yes". Then, we compute the 70th percentile of the `admission_age` column for this subset.

Here is the Python code to achieve this:

```
python
1 import numpy as np
2
3 # Filter the dataset for students who entered through affirmative action
4 affirmative_action_students = df_filtered[df_filtered['affirmative_action'] == 'Yes']
5
6 # Calculate the 70th percentile of admission age
7 age_70th_percentile = np.percentile(affirmative_action_students['admission_age'].dropna(), 70)
8
9 # Print the result
10 print(f"The 70th percentile of the age of entry for students who entered through affirmative act
```

**Explanation of the Code:**

- 1. **Filtering Affirmative Action Students** :
  - We use `df_filtered[df_filtered['affirmative_action'] == 'Yes']` to select rows where the `affirmative_action` column is "Yes".
- 2. **Handling Missing Values** :
  - The `.dropna()` method ensures that missing values in the `admission_age` column are excluded from the calculation.
- 3. **Calculating the 70th Percentile** :
  - The `np.percentile()` function computes the 70th percentile of the `admission_age` column.
- 4. **Formatting the Output** :
  - The result is printed with two decimal places for clarity.

**Deliverable:**

After running the code, you will get the 70th percentile of the age of entry for students who entered through affirmative action.

Let me know if you'd like me to execute this code or provide further assistance!

