

# Dataset Filtering 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.



To analyze the provided dataset and answer specific analytical questions, we'll follow a structured approach:

1. **Understand the Dataset** : The dataset contains student records with attributes such as admission period, enrollment type, situation, demographics, academic performance, and more.
2. **Filter and Clean the Data** : Apply the specified filters:
  - Include only rows where `admission_period` is between 2006.1 and 2019.2.
  - Exclude 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. **Answer the Analytical Question** : Use the cleaned dataset to address the question.

Here is the Python code to perform these steps:

```
python
1 import pandas as pd
2
3 # Load the dataset
4 df = pd.read_csv('stratified_history.csv', header=None)
5
6 # Assign column names based on the dataset structure (assuming known column order)
7 v columns = [
8     'student_id', 'year', 'university_code', 'admission_period', 'entry_mode', 'course_code',
9     'enrollment_status', 'graduation_period', 'duration', 'situation', 'is_ignored',
10    'quota_reservation', 'gender', 'has_disability', 'race', 'age_at_entry', 'age_at_situation',
11    'school_type', 'school_exclusivity', 'score', 'registration_number', 'last_period',
12    'periods_enrolled', 'periods_approved', 'academic_situation', 'grade', 'final_result',
13    'final_grade', 'subject_name', 'credits', 'workload', 'academic_unit', 'semester'
14 ]
15 df.columns = columns
16
17 # Filter the dataset
18 df_filtered = df[
19     (df['admission_period'] >= 2006.1) &
20     (df['admission_period'] <= 2019.2) &
21     (df['enrollment_status'] != 'Exempted') &
22     (df['situation'] != 'In Progress')
23 ]
24
25 # Display basic information about the cleaned dataset
26 print("Cleaned Dataset Overview:")
27 print(df_filtered.info())
28 print("\nFirst Few Rows of Cleaned Dataset:")
29 print(df_filtered.head())
30
31 # Example Analytical Question: What is the distribution of students by gender in the cleaned dataset?
32 gender_distribution = df_filtered['gender'].value_counts()
33 print("\nGender Distribution in Cleaned Dataset:")
34 print(gender_distribution)
```

**Explanation of the Code:**

1. **Loading the Dataset** : The dataset is loaded using `pd.read_csv`. Since the dataset lacks headers, we manually assign column names based on the provided structure.
2. **Filtering** :
  - `admission_period` is filtered to include values between 2006.1 and 2019.2.

- Rows where `enrollment_status` is "Exempted" are excluded.
  - Rows where `situation` is "In Progress" are excluded.
- Storing the Cleaned Data :** The filtered data is stored in `df_filtered`.
  - Analytical Question :** As an example, the distribution of students by gender is calculated using `value_counts()`.

#### Deliverables:

- Cleaned Dataset :** `df_filtered` contains the filtered data ready for analysis.
- Answer to Analytical Question :** The gender distribution is printed, showing the count of male and female students in the cleaned dataset.

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

Using multiple regression or ANCOVA, evaluate whether affirmative action, mode of admission, and gender significantly affect overall student averages, controlling for confounding factors such as academic program and socioeconomic background.



To evaluate whether affirmative action, mode of admission, and gender significantly affect overall student averages while controlling for confounding factors such as academic program and socioeconomic background, we can use a multiple regression or ANCOVA (Analysis of Covariance) approach. Below is the structured plan and Python implementation to achieve this:

#### Steps to Solve the Problem

- Understand the Variables :**
  - Dependent Variable :** Overall student averages (`final_grade`).
  - Independent Variables :**
    - Affirmative action (`quota_reservation`).
    - Mode of admission (`entry_mode`).
    - Gender (`gender`).
  - Control Variables :**
    - Academic program (`academic_unit`).
    - Socioeconomic background (e.g., `school_type`, `school_exclusivity`, `race`).
- Filter the Dataset :**
  - Use the cleaned dataset (`df_filtered`) created earlier, which excludes irrelevant rows.
- Prepare the Data :**
  - Convert categorical variables (e.g., `gender`, `entry_mode`, `quota_reservation`) into numerical representations using one-hot encoding or dummy variables.
  - Normalize continuous variables if necessary.
- Perform Multiple Regression :**
  - Fit a multiple regression model where:
    - Dependent variable: `final_grade`.
    - Independent variables: Affirmative action, mode of admission, gender, and control variables.
  - Assess the statistical significance of each independent variable.
- Interpret Results :**
  - Check the p-values for each independent variable to determine significance.
  - Evaluate the coefficients to understand the direction and magnitude of the effects.
- Python Implementation :**
  - Use libraries like `statsmodels` for regression analysis.

#### Python Code

```
python
1 import pandas as pd
2 import statsmodels.api as sm
3 from statsmodels.formula.api import ols
4
5 # Load the cleaned dataset
6 df_filtered = pd.read_csv('stratified_history.csv', header=None)
7
```

```

8 # Assign column names based on the dataset structure
9 v columns = [
10     'student_id', 'year', 'university_code', 'admission_period', 'entry_mode', 'course_code',
11     'enrollment_status', 'graduation_period', 'duration', 'situation', 'is_ignored',
12     'quota_reservation', 'gender', 'has_disability', 'race', 'age_at_entry', 'age_at_situation',
13     'school_type', 'school_exclusivity', 'score', 'registration_number', 'last_period',
14     'periods_enrolled', 'periods_approved', 'academic_situation', 'grade', 'final_result',
15     'final_grade', 'subject_name', 'credits', 'workload', 'academic_unit', 'semester'
16 ]
17 df_filtered.columns = columns
18
19 # Additional filtering (if not already applied)
20 df_filtered = df_filtered[
21     (df_filtered['admission_period'] >= 2006.1) &
22     (df_filtered['admission_period'] <= 2019.2) &
23     (df_filtered['enrollment_status'] != 'Exempted') &
24     (df_filtered['situation'] != 'In Progress')
25 ]
26
27 # Drop rows with missing final grades
28 df_filtered = df_filtered.dropna(subset=['final_grade'])
29
30 # Convert categorical variables to numerical using dummy variables
31 df_filtered['gender_numeric'] = df_filtered['gender'].apply(lambda x: 1 if x == 'Male' else 0)
32 df_filtered = pd.get_dummies(df_filtered, columns=['entry_mode', 'quota_reservation', 'academic_unit'])
33
34 # Define the regression formula
35 formula = (
36     'final_grade ~ '
37     'gender_numeric + '
38     'entry_mode_Unified Selection System (SISU) + entry_mode_Eorraine Exam + entry_mode_Eorraine '
39     'quota_reservation_School Income Race + quota_reservation_School Income + quota_reservation_ '
40     'academic_unit_Academic Unit of Systems and Computing + academic_unit_Academic Unit of Socic '
41     'school_type + school_exclusivity + race'
42 )
43
44 # Fit the multiple regression model
45 model = ols(formula, data=df_filtered).fit()
46
47 # Print the summary of the regression
48 print(model.summary())

```

### Explanation of the Code

#### 1. Dataset Preparation :

- The dataset is loaded and filtered to include only relevant rows.
- Missing values in `final_grade` are dropped since they cannot be used in regression.

#### 2. Categorical Variables :

- Categorical variables (`gender`, `entry_mode`, `quota_reservation`, `academic_unit`) are converted into numerical representations using dummy variables.

#### 3. Regression Formula :

- The dependent variable is `final_grade`.
- Independent variables include affirmative action (`quota_reservation`), mode of admission (`entry_mode`), gender (`gender_numeric`), and control variables (`school_type`, `school_exclusivity`, `race`, `academic_unit`).

#### 4. Model Fitting :

- The `ols` function from `statsmodels` is used to fit the regression model.
- The `summary()` method provides detailed statistics, including coefficients, p-values, and R-squared.

### Interpreting Results

#### 1. Significance Testing :

- Look at the p-values for each independent variable. A p-value < 0.05 indicates statistical significance.

#### 2. Effect Size :

- The coefficients indicate the magnitude and direction of the effect. For example, a positive coefficient for `gender_numeric` suggests that males have higher grades (on average) compared to females.

#### 3. R-squared :

- The R-squared value shows how much variance in `final_grade` is explained by the model.

### Deliverables

#### 1. Regression Summary :

- The output of `model.summary()` will provide detailed insights into the relationships between variables.

#### 2. Conclusion :

- Based on the results, we can conclude whether affirmative action, mode of admission, and gender significantly

affect overall student averages while controlling for confounding factors.

If you run the code and share the results, I can help interpret them further!

