

Data Cleaning and Preparation

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 dataset and ensure it meets the criteria for data-driven decision-making, we will first load the dataset, clean it according to the specified conditions, and store the cleaned data in a new DataFrame named `df_filtered`. Then, we will proceed to address any specific analytical question using this cleaned dataset.

Below is the Python code that performs the necessary data cleaning and prepares the `df_filtered` DataFrame:

```
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 basic information about the dataset
8 print("Original Dataset Shape:", df.shape)
9 print("Columns in the Dataset:", df.columns.tolist())
10
11 # Cleaning Step 1: Filter admission_period to include only values between 2006.1 and 2019.2
12 df_filtered = df[(df['admission_period'] >= 2006.1) & (df['admission_period'] <= 2019.2)]
13
14 # Cleaning Step 2: Remove entries where enrollment_type is labeled as "Exemption"
15 df_filtered = df_filtered[df_filtered['enrollment_type'] != 'Exemption']
16
17 # Cleaning Step 3: Exclude entries where situation is "In Progress"
18 df_filtered = df_filtered[df_filtered['situation'] != 'In Progress']
19
20 # Display the shape of the cleaned dataset
21 print("Filtered Dataset Shape:", df_filtered.shape)
22
23 # Verify the cleaning steps
24 print("Unique Values in 'enrollment_type':", df_filtered['enrollment_type'].unique())
25 print("Unique Values in 'situation':", df_filtered['situation'].unique())
26 print("Range of 'admission_period':", df_filtered['admission_period'].min(), "-", df_filtered['admission_period'].max())
27
28 # Store the cleaned dataset in df_filtered
```

Explanation of the Code:

- Loading the Dataset :** The dataset is loaded into a DataFrame using `pd.read_csv()`.
- Filtering `admission_period` :** Entries are filtered to include only those where `admission_period` falls within the range `[2006.1, 2019.2]`.
- Removing "Exemption" Entries :** Rows where `enrollment_type` is labeled as `"Exemption"` are excluded.
- Excluding "In Progress" Situations :** Rows where `situation` is `"In Progress"` are removed.
- Verification :** The cleaned dataset is verified by checking its shape, unique values in relevant columns, and the range of `admission_period`.

Key Characteristics of the Cleaned Dataset:

- The cleaned dataset (`df_filtered`) includes only entries from the specified time range (`2006.1` to `2019.2`).
- It excludes rows with `enrollment_type` labeled as `"Exemption"` and `situation` labeled as `"In Progress"`.
- The resulting dataset is ready for further analysis.

Next Steps:

Once the dataset is cleaned and stored in `df_filtered`, you can provide a specific analytical question, and I will address it using the cleaned dataset. Let me know the question, and I will proceed with the analysis and provide the corresponding

Python code.

Analyze how a student's age of entry and dropout influences their overall average (mean of non-null grades). Apply one statistical method (e.g., regression or correlation) to measure significance, considering confounding factors.

 qwen-max-latest

To analyze how a student's **age of entry** and **dropout** influences their **overall average grade**, we will follow these steps:

Step 1: Define Variables

- **Dependent Variable** : The overall average grade (`mean_grade`) for each student, calculated as the mean of non-null grades.
- **Independent Variables** :
 - **Age of Entry** : Derived from the `admission_period` and the student's birth year (if available).
 - **Dropout Status** : A binary variable where `1` indicates dropout and `0` indicates no dropout.
- **Confounding Factors** :
 - **Gender** : Male or Female.
 - **Quota Reservation** : Whether the student benefited from quota reservation.
 - **School Type** : Public, Private, or Mixed schooling background.

Step 2: Data Preparation

We need to clean and transform the dataset to calculate the required variables:

1. Compute the **age of entry** by subtracting the birth year from the admission year.
2. Compute the **mean grade** for each student by averaging non-null grades.
3. Encode categorical variables like `gender`, `quota_reservation`, and `school_type` into numerical values for regression analysis.
4. Filter out rows with missing critical data (e.g., `grade`, `birth_year`, `admission_period`).

Step 3: Statistical Method

We will use **multiple linear regression** to measure the influence of age of entry and dropout status on the overall average grade, while controlling for confounding factors. The regression model can be expressed as:

$$\text{mean_grade} = \beta_0 + \beta_1(\text{age_of_entry}) + \beta_2(\text{dropout_status}) + \beta_3(\text{gender}) + \beta_4(\text{quota_reservation}) + \beta_5(\text{school_type}) + \epsilon$$

Where:

- β_0 is the intercept.
- $\beta_1, \beta_2, \dots, \beta_5$ are coefficients representing the influence of each variable.
- ϵ is the error term.

Step 4: Implementation in Python

Below is the Python code to perform the analysis:

```
python
1 import pandas as pd
2 import numpy as np
3 from sklearn.linear_model import LinearRegression
4 from sklearn.preprocessing import LabelEncoder
5 from sklearn.model_selection import train_test_split
6 from sklearn.metrics import r2_score
7
8 # Load the cleaned dataset
9 file_path = 'stratified_history.csv'
10 df = pd.read_csv(file_path)
11
12 # Step 1: Filter and clean the dataset
13 df_filtered = df[(df['admission_period'] >= 2006.1) & (df['admission_period'] <= 2019.2)]
14 df_filtered = df_filtered[df_filtered['enrollment_type'] != 'Exemption']
15 df_filtered = df_filtered[df_filtered['situation'] != 'In Progress']
16
17 # Step 2: Calculate age of entry
18 # Assuming 'admission_period' is in format YYYY.S (e.g., 2010.1 for Spring 2010)
19 df_filtered['admission_year'] = df_filtered['admission_period'].apply(lambda x: int(x))
20 df_filtered['age_of_entry'] = df_filtered['admission_year'] - df_filtered['birth_year']
21
22 # Step 3: Compute mean grade for each student
23 df_filtered['mean_grade'] = df_filtered['grade'].groupby(df_filtered['student_id']).transform('mean')
```

```

25 # Step 4: Encode categorical variables
26 label_encoders = {}
27 categorical_vars = ['gender', 'quota_reservation', 'school_type']
28 for col in categorical_vars:
29     le = LabelEncoder()
30     df_filtered[col] = le.fit_transform(df_filtered[col])
31     label_encoders[col] = le
32
33 # Step 5: Create binary dropout status
34 df_filtered['dropout_status'] = (df_filtered['situation'] == 'Dropout').astype(int)
35
36 # Step 6: Prepare data for regression
37 X = df_filtered[['age_of_entry', 'dropout_status', 'gender', 'quota_reservation', 'school_type']]
38 y = df_filtered['mean_grade']
39
40 # Split into training and testing sets
41 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
42
43 # Step 7: Fit the regression model
44 model = LinearRegression()
45 model.fit(X_train, y_train)
46
47 # Step 8: Evaluate the model
48 y_pred = model.predict(X_test)
49 r2 = r2_score(y_test, y_pred)
50
51 # Print results
52 print("Regression Coefficients:")
53 for feature, coef in zip(X.columns, model.coef_):
54     print(f"{feature}: {coef}")
55 print(f"Intercept: {model.intercept_}")
56 print(f"R-squared: {r2}")
57
58 # Analyze significance
59 significant_features = [feature for feature, coef in zip(X.columns, model.coef_) if abs(coef) >
60 print("Significant Features:", significant_features)

```

Step 5: Interpretation

1. Coefficients :

- The coefficient of `age_of_entry` indicates how much the mean grade changes for each additional year of age at entry.
- The coefficient of `dropout_status` shows the effect of dropping out on the mean grade.
- Other coefficients represent the influence of confounding factors.

2. R-squared :

- This value measures how well the model explains the variance in the mean grade. A higher R-squared value indicates a better fit.

3. Significance :

- Features with coefficients significantly different from zero (e.g., absolute value > 0.1) are considered significant predictors of the mean grade.

Expected Results

- If `age_of_entry` has a negative coefficient, it suggests that older students tend to have lower grades.
- If `dropout_status` has a strong negative coefficient, it implies that students who drop out tend to have lower grades.
- Confounding factors like `gender`, `quota_reservation`, and `school_type` help adjust for biases in the analysis.

Conclusion