

Student Dataset – 10 Detailed Case Studies

Dataset Columns:

Gender, EthnicGroup, ParentEduc, LunchType, TestPrep, ParentMaritalStatus, PracticeSport, IsFirstChild, NrSiblings, TransportMeans, WklyStudyHours, MathScore, ReadingScore, WritingScore

Case Study 1: Data Cleaning & Standardization

```
df2 = pd.read_csv('/content/student_data.csv')
```

```
df2.info()
```

Objective: Standardize categorical values, handle missing values, and convert data types.

Drop unnecessary column

```
df.drop(columns=['Unnamed: 0'], inplace=True)
```

Convert weekly study hours to numeric

```
df['WklyStudyHours'] = pd.to_numeric(df['WklyStudyHours'], errors='coerce')
```

Standardize categorical columns

```
df['Gender'] = df['Gender'].str.title()
```

```
df['LunchType'] = df['LunchType'].str.title()
```

```
df['TestPrep'] = df['TestPrep'].str.title()
```

Fill missing ParentEduc with mode

```
df['ParentEduc'].fillna(df['ParentEduc'].mode()[0], inplace=True)
```

Insights:

- Clean data ensures accurate analysis and model building.
- Standardized values avoid errors in grouping and encoding.

Case Study 2: Handling Missing Values

Objective: Impute missing values for EthnicGroup, ParentMaritalStatus, TransportMeans, and NrSiblings.

```
df['EthnicGroup'].fillna('Unknown', inplace=True)
df['ParentMaritalStatus'].fillna('Unknown', inplace=True)
df['TransportMeans'].fillna('Other', inplace=True)
df['NrSiblings'].fillna(df['NrSiblings'].median(), inplace=True)
```

Insights:

- Proper imputation prevents data loss.
- Median imputation for numeric columns avoids bias.

Case Study 3: Exploratory Data Analysis – Gender & Scores

Objective: Compare performance by gender.

```
import seaborn as sns
import matplotlib.pyplot as plt

sns.boxplot(x='Gender', y='MathScore', data=df)
plt.title("Math Score by Gender")
plt.show()

sns.boxplot(x='Gender', y='ReadingScore', data=df)
plt.title("Reading Score by Gender")
plt.show()

sns.boxplot(x='Gender', y='WritingScore', data=df)
plt.title("Writing Score by Gender")
```

```
plt.show()
```

Insights:

- Observe differences in score distribution between male and female students.
 - Helps identify performance gaps.
-

Case Study 4: Effect of Test Preparation on Scores

Objective: Analyze impact of completing test preparation course.

```
sns.boxplot(x='TestPrep', y='MathScore', data=df)
```

```
plt.title("Math Scores by Test Preparation Status")
```

```
plt.show()
```

Insights:

- Students who completed test prep generally have higher scores.
 - Reinforces the importance of preparation programs.
-

Case Study 5: Parent Education & Student Performance

Objective: Explore relationship between parental education and student scores.

```
sns.barplot(x='ParentEduc', y='MathScore', data=df)
```

```
plt.xticks(rotation=45)
```

```
plt.title("Math Score vs Parent Education")
```

```
plt.show()
```

Insights:

- Higher parental education is correlated with better student performance.
 - Useful for targeted support programs.
-

Case Study 6: Sibling Count & Study Hours vs Performance

Objective: Examine impact of siblings and weekly study hours.

```
sns.scatterplot(x='NrSiblings', y='MathScore', data=df)
plt.title("Math Score vs Number of Siblings")
plt.show()
```

```
sns.scatterplot(x='WklyStudyHours', y='MathScore', data=df)
plt.title("Math Score vs Weekly Study Hours")
plt.show()
```

Insights:

- More siblings may slightly reduce individual attention, affecting scores.
 - Increased study hours positively correlate with performance.
-

Case Study 7: Correlation Analysis

Objective: Study relationships between scores.

```
corr_matrix = df[['MathScore', 'ReadingScore', 'WritingScore']].corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm')
plt.title("Correlation between Scores")
plt.show()
```

Insights:

- Strong positive correlation among Math, Reading, and Writing scores.
 - Indicates overall academic performance trend.
-

Case Study 8: Clustering Students Based on Scores

Objective: Segment students into performance clusters using KMeans.

```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
```

```
scaler = StandardScaler()

scores_scaled = scaler.fit_transform(df[['MathScore', 'ReadingScore', 'WritingScore']])

kmeans = KMeans(n_clusters=3, random_state=42)
df['PerformanceCluster'] = kmeans.fit_predict(scores_scaled)

sns.scatterplot(x='MathScore', y='ReadingScore', hue='PerformanceCluster', data=df)
plt.title("Student Performance Clusters")
plt.show()
```

Insights:

- Clusters represent low, medium, and high performers.
 - Enables targeted interventions.
-

Case Study 9: Predictive Modeling – Predict Math Scores

Objective: Predict Math Scores using student features.

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
```

Encode categorical features

```
df_model = pd.get_dummies(df.drop(columns=['ReadingScore', 'WritingScore']),
drop_first=True)
```

```
X = df_model.drop('MathScore', axis=1)
```

```
y = df_model['MathScore']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
rf = RandomForestRegressor(n_estimators=200, random_state=42)
```

```
rf.fit(X_train, y_train)
```

```
y_pred = rf.predict(X_test)
```

```
print("RMSE:", mean_squared_error(y_test, y_pred, squared=False))
```

Insights:

- Important features: WklyStudyHours, TestPrep, ParentEduc.
 - Can guide personalized learning plans.
-

Case Study 10: Actionable Insights & Recommendations

Objective: Provide recommendations based on analysis.

Identify students needing intervention

```
low_perf_students = df[df['PerformanceCluster']==0]
```

```
print(low_perf_students[['Gender', 'TestPrep', 'WklyStudyHours', 'MathScore']].head(10))
```

Insights:

- Students in low-performance cluster may need extra support.
- Encourage test prep courses and additional study hours.
- Target parental engagement programs to improve outcomes.