Student Performance Factors

January 30, 2025

1 Role of Family on Student Performance

1.1 Purpose and Questions

Throughout our society today, exams have become woven into almost every part of our life. Almost every child in the United States spends at least 13 years taking exams and having their performance evaluated to determine their future pursuits. However, performance on exams can vary heavily. Many people blame various factors including gender, race, parental involvement, money, and more. Despite various explanations the debate still continues. Throughout this paper, I hope to provide some understanding to the reasons behind student performance on exams and to discern whether correlation is equal to causation or if there is some deeper reason.

1.1.1 Questions to Answer

- Does family income play a significant role in the performance of students?
- Does tutoring, typically only afforded by the well-off families, increase performance?
- What role does the family of a student have in their performance on exams?

We will be using a dataset from Kaggle with close to 20 features. You can access the dataset with this link https://www.kaggle.com/datasets/lainguyn123/student-performance-factors/data. We will be focusing on the role of family in student performance.

[4]:
```
#first, let's load the dataset, we'll need pandas
import pandas as pd

df = pd.read_csv('StudentPerformanceFactors.csv')
```

[5]:
```
df.shape #rows x columns
```

[5]:
```
(6607, 20)
```

1.2 Preprocessing

Before we start visualizing data, we should evaluate the dataset and determine whether the dataset needs any cleaning done.

[7]:
```
print(df.isnull().sum())   #check for nulls in every column
print(df.isnull().sum().sum())
```

```
Hours_Studied                   0
Attendance                      0
Parental_Involvement            0
Access_to_Resources             0
Extracurricular_Activities      0
Sleep_Hours                     0
Previous_Scores                 0
Motivation_Level                0
Internet_Access                 0
Tutoring_Sessions               0
Family_Income                   0
Teacher_Quality                78
School_Type                     0
Peer_Influence                  0
Physical_Activity               0
Learning_Disabilities           0
Parental_Education_Level       90
Distance_from_Home             67
Gender                          0
Exam_Score                      0
dtype: int64
235
```

[8]:
```
df = df.copy()

# Fill categorical columns with their most frequent value (mode)
categorical_cols = ['Teacher_Quality', 'Parental_Education_Level',␣
 ↪'Distance_from_Home']
df[categorical_cols] = df[categorical_cols].apply(lambda col: col.fillna(col.
 ↪mode()[0]))

# Check if nulls are removed
print(df.isnull().sum())
```

```
Hours_Studied                   0
Attendance                      0
Parental_Involvement            0
Access_to_Resources             0
Extracurricular_Activities      0
Sleep_Hours                     0
Previous_Scores                 0
Motivation_Level                0
Internet_Access                 0
Tutoring_Sessions               0
Family_Income                   0
Teacher_Quality                 0
School_Type                     0
Peer_Influence                  0
```

```
Physical_Activity             0
Learning_Disabilities         0
Parental_Education_Level      0
Distance_from_Home            0
Gender                        0
Exam_Score                    0
dtype: int64
```

[9]:
```
print(df.duplicated().sum())  # Count duplicate rows
```

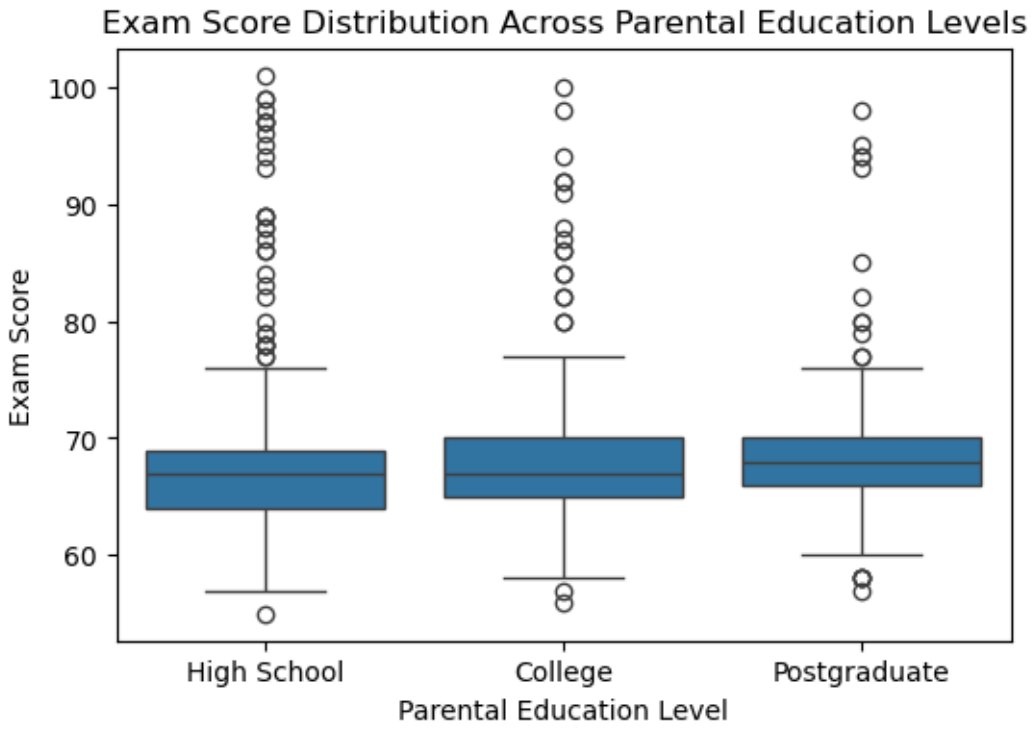
```
0
```

[10]:
```
bins = [0, 60, 80, 100]  # Chose Ranges: 0-60, 61-80, 81-100
labels = ['Low', 'Average', 'High']

# Used feature engineering to bin exam scores
df['Score_Category'] = pd.cut(df['Exam_Score'], bins=bins, labels=labels,␣
 ↪right=True)
```

So far, we've checked for any null values and checked for duplicates. I chose to replace missing values with the mode because the amount of nulls weren't considerable enough for a dataset with almost 7000 datapoints. At this point, I am comfortable with our preprocessing steps because it looks like this dataset is ready to go. I chose to create a binned feature called Score Category so that it would be easier for me to analyze the overll performance. I personally don't believe the difference between a 65 and a 67 is necessarily significant enough to evaluate performance.

Oftentimes, many students emphasize how their family's background is a big part in how they view school. Students whose families have less formal education may have a different view on the importance of school and that might correlate to the effort students put in towards their exams. Therefore, I would like to create a correlation matrix to analyze the correlation between parental education and exam performance.

[13]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(6, 4))
sns.boxplot(x=df['Parental_Education_Level'], y=df['Exam_Score'])
plt.xlabel("Parental Education Level")
plt.ylabel("Exam Score")
plt.title("Exam Score Distribution Across Parental Education Levels")
plt.show()
```



As we see in these boxplots, the median score is about the same. Median is a good measure of the central tendency of each group. We notice that each group has a very similar distribution. The only thing that is noticeable is that students whose parents had postgraduate education have a higher floor than the others. I think this goes to show that there is likely more than just your parents' education that affects your exam performance. Maybe what matters more is the individual, not just family life

[ ]: