Homework 5

https://www.kaggle.com/datasets/jayaantanaath/student-habits-vs-academic-performance

Overview: This dataset sets student habits against academic performance, tracking relevant lifestyle habits such as social media usage, exercise frequency, and sleep quantity. It is meant to evaluate the factors that impact and influence academic performance. Though simulated, this data is useful for educational analysis and can be used as inspiration for real data collection.

Categorical attributes:

1. gender
2. part_time_job
3. diet_quality
4. parental_education_level
5. internet_quality
6. extracurricular_participation

Numeric attributes:

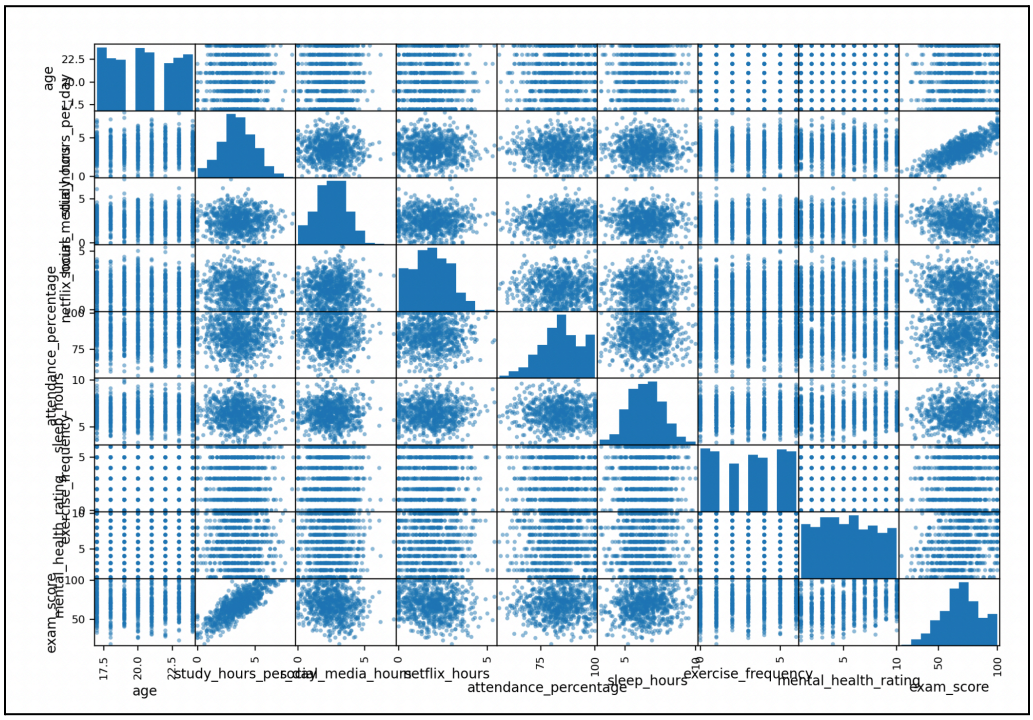
1. age
2. study_hours_per_day
3. social_media_hours
4. netflix_hours
5. attendance_percentage
6. sleep_hours
7. exercise_frequency
8. mental_health_rating
9. exam_score

Prediction/use case: This dataset is used for machine learning models to predict the academic performance, including exam scores and grades, based on an aggregation of data of an individual student.

Code Results

```
RangeIndex: 1000 entries, 0 to 999
Data columns (total 16 columns):
 #   Column                         Non-Null Count  Dtype
---  ------                         --------------  -----
 0   student_id                     1000 non-null   object
 1   age                            1000 non-null   int64
 2   gender                         1000 non-null   object
 3   study_hours_per_day            1000 non-null   float64
 4   social_media_hours             1000 non-null   float64
 5   netflix_hours                  1000 non-null   float64
 6   part_time_job                  1000 non-null   object
 7   attendance_percentage          1000 non-null   float64
 8   sleep_hours                    1000 non-null   float64
 9   diet_quality                   1000 non-null   object
 10  exercise_frequency             1000 non-null   int64
 11  parental_education_level       909 non-null    object
 12  internet_quality               1000 non-null   object
 13  mental_health_rating           1000 non-null   int64
 14  extracurricular_participation  1000 non-null   object
 15  exam_score                     1000 non-null   float64
dtypes: float64(6), int64(3), object(7)
memory usage: 125.1+ KB
gender
Female    481
Male      477
Other      42
Name: count, dtype: int64
part_time_job
No     785
Yes    215
Name: count, dtype: int64
diet_quality
Fair    437
Good    378
Poor    185
Name: count, dtype: int64
parental_education_level
High School    392
Bachelor       350
Master         167
Name: count, dtype: int64
internet_quality
Good       447
Average    391
Poor       162
Name: count, dtype: int64
              age  study_hours_per_day  social_media_hours  netflix_hours  attendance_percentage  sleep_hours  exercise_frequency  mental_health_rating   exam_score
count  1000.0000          1000.00000          1000.000000    1000.000000            1000.000000  1000.000000         1000.000000           1000.000000  1000.000000
mean     20.4980             3.55010             2.505500       1.819700              84.131700     6.470100            3.042000              5.438000    69.601500
std       2.3081             1.46889             1.172422       1.075118               9.399246     1.226377            2.025423              2.847501    16.888564
min      17.0000             0.00000             0.000000       0.000000              56.000000     3.200000            0.000000              1.000000    18.400000
25%      18.7500             2.60000             1.700000       1.000000              78.000000     5.600000            1.000000              3.000000    58.475000
50%      20.0000             3.50000             2.500000       1.800000              84.400000     6.500000            3.000000              5.000000    70.500000
75%      23.0000             4.50000             3.300000       2.525000              91.025000     7.300000            5.000000              8.000000    81.325000
max      24.0000             8.30000             7.200000       5.400000             100.000000    10.000000            6.000000             10.000000   100.000000
```

Homework 6




```
exam_score                          1.000000
study_hours_per_day                 0.825203
attendance_study_interaction        0.810264
mental_health_rating                0.308922
exercise_frequency                  0.167014
sleep_hours                         0.131732
attendance_percentage               0.082723
age                                 0.002882
netflix_hours                      -0.162496
social_media_hours                 -0.185607
screen_time_total                  -0.242562
sleep_study_ratio                  -0.331338
Name: exam_score, dtype: float64
exam_score                          1.000000
study_hours_per_day                 0.825419
mental_health_rating                0.321523
exercise_frequency                  0.160107
sleep_hours                         0.121683
attendance_percentage               0.089836
age                                -0.008907
social_media_hours                 -0.166733
netflix_hours                      -0.171779
Name: exam_score, dtype: float64
```

In this assignment, I focused on finding patterns and correlations between the data. Specifically, I tried to find what attributes were most directly linked to academic performance. Additionally, I created three attributes to better understand my dataset: screen_time_total, sleep_study_ratio, and academic_effort (more on them below). I then visualized relationships between variables using scatter plots to identify which attributes had the strongest linear relationships with exam scores (although I encountered formatting issues).

Augmentations to the dataset:

1. screen_time total - adds up social media hours and Netflix hours to better understand the total device usage
2. sleep_study_ratio - divides sleep hours by study hours to try to find a good balance
3. academic_effort = adds or multiples attendance_percentage and study_hours_per_day

Potential attributes to drop:

1. age - with a correlation rating of 0.002882, age has almost no effect on academic performance
2. attendance_percentage - with a correlation rating of 0.082723, attendance also has a minimal effect on academic performance. It could be better off combined with another attribute, as mentioned above.

Homework 7

```
Numeric: Index(['study_hours_per_day', 'social_media_hours', 'netflix_hours',
       'sleep_hours', 'exercise_frequency', 'mental_health_rating',
       'exam_score', 'academic_effort'],
      dtype='object')
Categorical: Index(['student_id', 'gender', 'part_time_job', 'diet_quality',
       'parental_education_level', 'internet_quality',
       'extracurricular_participation'],
      dtype='object')
exam_score                      1.000000
study_hours_per_day             0.825419
academic_effort                 0.809390
mental_health_rating            0.321523
exercise_frequency              0.160107
sleep_hours                     0.121683
extracurricular_participation   0.000881
gender                         -0.002406
student_id                     -0.021489
part_time_job                  -0.026608
internet_quality               -0.032560
parental_education_level       -0.044349
diet_quality                   -0.050275
social_media_hours             -0.166733
netflix_hours                  -0.171779
screen_time_total              -0.237631
sleep_study_ratio              -0.380043
Name: exam_score, dtype: float64
```

For Assignment 7, I dived into prepping data for machine learning models, specifically with transformers. (With no missing values, cleaning the data with tools such as Scikit-Learn was irrelevant). To start, I categorized my data into numerical and categorical attributes using the select_dtypes() function. Next, I used a pipeline to standardize numerical data and an ordinal encoder to standardize categorical data into integers that the machine learning model could understand. (With the help of many Google searches) I learned these were meant to convert values into numerical form for the machine learning model to better interpret. After running corr(), I analyzed the relationships between all attributes and exam_score. In the end, I did not see a major difference in the results after applying the transformers, although I understand the use case of transformers in more subjective datasets.

Potential attributes to drop:

1. student_id - Although useful for tracking individual students, it is completely irrelevant to the machine learning model.

2. sleep_hours - this one is very surprising, with a correlation rating of 0.121683. While I think I would not drop this attribute, it seems to have only a weak positive correlation with student performance and may not be the most relevant.

Homework 8

For this assignment, I started by attempting to run the linear regression model on my dataset. However, I was met with an error about NaN values (I found that it was because of sleep_study_ratio, with some values becoming undefined after being divided by 0). To fix this, I used a SimpleImputer to clean up these values and prep the data. The linear regression model worked **GREAT**, with the first five student predictions pretty accurately matching the actual exam scores (with one outlier). I ran code to calculate my RMSE score and received a 5.2 (after some research, I am content with this score due to having 1000 rows of data). Next, I tried the decision tree regressor and received a RMSE value of 0 (which, as we talked about in class, led me to believe that it was too complex for my dataset. Finally, I performed cross-validation on both models, and found that linear regression again resulted in better mean and RMSE values. Between the two models, the linear regression model is a clear winner for my dataset due to its accurate predictions and good RMSE values.

Below is the analysis of working through the code, as well as the code outputs.

What I did:

- Tried to run Linear Regression model, but was met with an error about NaN values
    - Found that it was because of sleep_study_ratio with some values becoming undefined after being divided by 0
    - Used a SimpleImputer to clean up these values and prep the data

```
exam_score                      1.000000
study_hours_per_day             0.839841
academic_effort                 0.823358
mental_health_rating            0.312560
exercise_frequency              0.192722
sleep_hours                     0.119980
sleep_study_ratio               0.042346
screen_time_total               0.024775
extracurricular_participation   0.016384
gender                          0.001894
parental_education_level       -0.027840
part_time_job                  -0.028011
student_id                     -0.032306
internet_quality               -0.038011
diet_quality                   -0.056196
social_media_hours             -0.163540
netflix_hours                  -0.178994
Name: exam_score, dtype: float64
Predictions: [48.78816204 85.99995017 81.05033474 68.41628725 67.43805174]
Labels: [49.9, 94.2, 80.9, 73.6, 67.5]
```

- The linear regression model worked **GREAT**, with the first five student predictions pretty accurately matching the actual exam scores

```
Predictions: [48.78816204 85.99995017 81.05033474 68.41628725 67.43805174]
Labels: [49.9, 94.2, 80.9, 73.6, 67.5]
5.203928129555084
```

- I ran code to calculate my rmse score and received a 5.2. After some research, I am content with this score due to having 1000 rows of data

```
exam_score                      1.000000
study_hours_per_day             0.839841
academic_effort                 0.823358
mental_health_rating            0.312560
exercise_frequency              0.192722
sleep_hours                     0.119980
sleep_study_ratio               0.042346
screen_time_total               0.024775
extracurricular_participation   0.016384
gender                          0.001894
parental_education_level       -0.027840
part_time_job                  -0.028011
student_id                     -0.032306
internet_quality               -0.038011
diet_quality                   -0.056196
social_media_hours             -0.163540
netflix_hours                  -0.178994
Name: exam_score, dtype: float64
0.0
```

- Ran the decision tree model, RMSE value of 0, probably too complicated for my dataset

```
0.0
DT_Scores: [ -95.7175    -80.73825   -91.2545    -71.538625  -86.31     -104.28325
  -88.389625  -79.445     -75.92      -67.7045  ]
DT_Mean: -84.13012499999999
DT_Standard deviation: 10.69240859091159
```

- ^ Decision Tree Cross Validation Score

```
LE_Scores: [5.49359878 5.19662502 5.45832567 5.07865217 5.20882984 5.51473476
 4.97048014 4.51532346 5.82654337 5.83706805]
LE_Mean: 5.310018125725639
LE_Standard deviation: 0.38283661836775884
```

- ^ Linear Regression Cross Validation Score

Homework 9

To start, I dropped student_id from my categorical encoding (in the real and test sets) as it hindered my model. Before working with hyperparameters, I started by using cross_val_scores from Scikit-Learn to obtain a baseline RMSE for each model. With this information, I decided to use the RandomForestRegressor model (although I had used a DecisionTree model previously). With this, I began running a GridSearchCV, using five combinations of hyperparameters to determine which would yield the lowest RMSE value, with each run being adjusted based on the results of the last. I started broadly with my first search, then narrowed the range and intervals until I was satisfied with the results. This step-by-step narrowing allowed me to focus on the most effective region of the hyperparameter space rather than testing random values. I found the best RMSE value came from a combination of 9 features and 130 estimators (decision trees). Running the final predictions, I found that my RMSE value actually increased (The baseline was 6.28, and the final was 7.45). This suggested that the tuned model overfitted the training folds during cross-validation. In the end, it still seems like linear regression is the best model to use as it consistently yields the lowest RMSE.

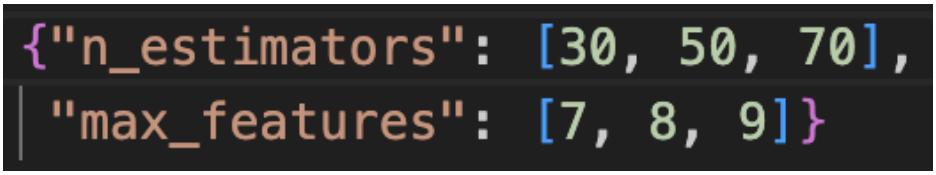
Below is my step-by-step thought and experimentation process with the code outputs.

1. Used cross_val_scores from SciKit to get a baseline RMSE from each model.
    a. Forest_reg = 6.277089288237025
    b. Lin_reg = 5.309203967649482
    c. Tree_reg = 9.366425412076904
    d. Even though I used a DecisionTree model last time, I went with a RandomForestRegressor model this time due to a better baseline RMSE
2. Run GridSearchCV to find best hyperparameters
    a. First run
        i.

```
{"n_estimators": [5, 10, 30],
 "max_features": [2, 4, 6, 8]}
```

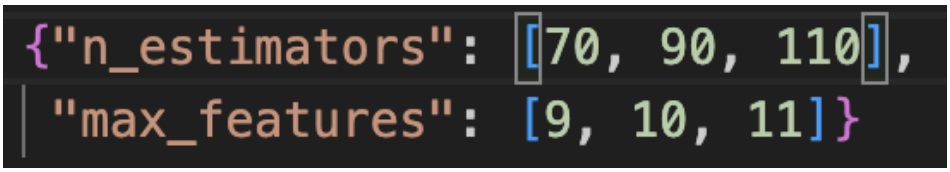
ii. 

b. Second run

i. 

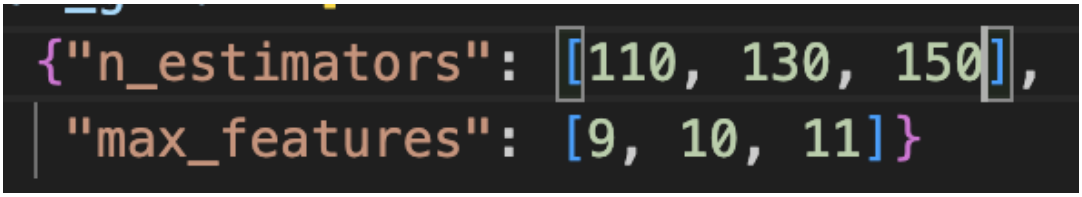
ii. 

c. Third run

i. 

ii. 

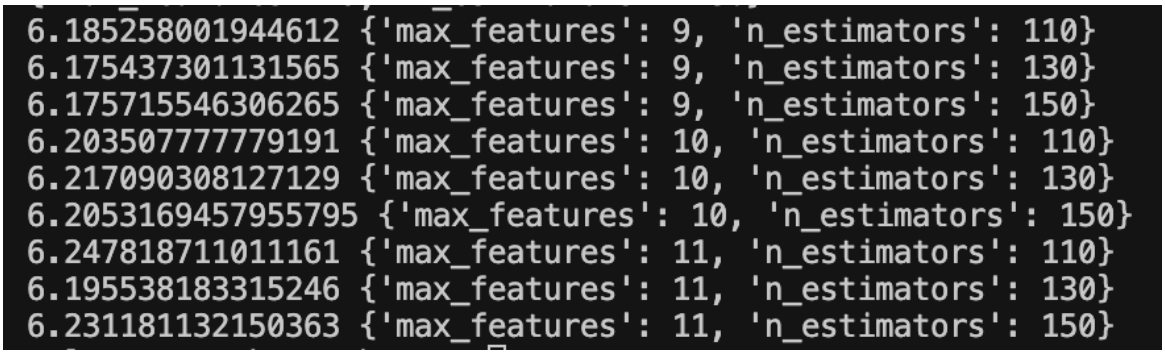
d. Fourth run

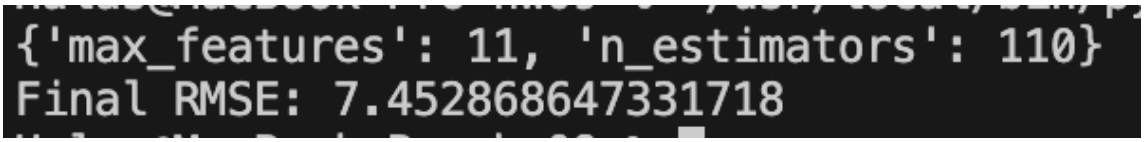
i. 

ii. 

e. I believe the best parameters are 11 and 150

3. I used the code snippet from lines 94-97 that allowed me to find the RMSE for every combination

a. 

    b. Based on this, the best combination looks to be 9 and 130.
4. Dropped student_id from my categorical encoding (in real and test set)
5. Ran final predictions, and got an RMSE that was worse
    a. 

    b. Model was overfit during GridSearchCV