Homework 2

Frimu Aurel-Viorel 1231EC – IoT

Workflow for “Homework 2” -> Create Kaggle account -> Download dataset without looking at anything else -> Create Jupyter Notebook -> Get to coding while doing research online as it can be seen from the notebook (the links), most of my thoughts at the time can be seen in the comments as well -> Finish coding -> Create PowerPoint -> Forget about the report (thinking the PowerPoint and report are the same) -> Find out from a friend they’re different -> Panic and do the report after the deadline, starting at around 00:30 D:

Python Version used: 3.12.3

1. Introduction

Definition: In this dataset there is the data for a bunch of students’ performance.

Objectives: Perform EDA to get a better understanding of the dataset, clean and preprocess the data if needed, smooth out class imbalance using SMOTE, ADASYN, undersampling, etc. In this Jupyter Notebook I also test out various Machine Learning models on the data to see which one gives the best results. The ultimate objective is to build a model that can predict the GPA / GradeClass of new students based on the other data in the set.

Dataset Description: The dataset is segmented in: Age, Gender, Ethnicity, ParentalEducation, StudyTimeWeekly, Absences, Tutoring, ParentalSupport, Extracurricular, Sports, Music, Volunteering, GPA and GradeClass.

2. EDA

Insights from data exploration: The data is already in numerical form, as in no words in the rows of the table except the header. After doing the heatmap (Figure 1), I realized the GPA and GradeClass are way too close to not be derived from one another.

- Absences are closely correlated with GPA and implicitly with GradeClass.
- Data distribution histograms (figure 2) show that there are data imbalances in a few columns: Ethnicity, Tutoring, Music, Sports, Volunteering, Extracurricular.
- I took GPA as the target because it made the most sense to me at the time.

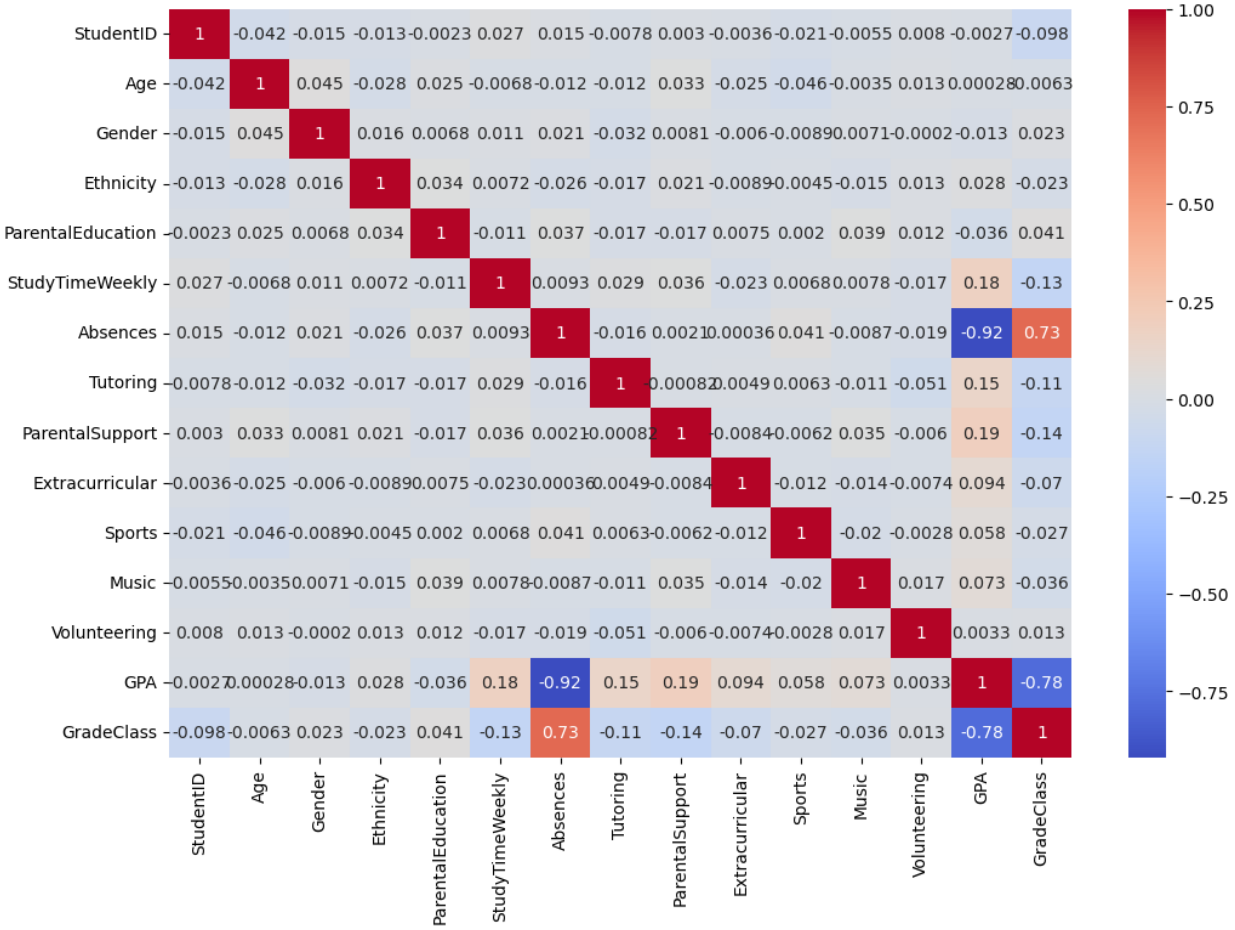

Figure 1 Correlation Heatmap

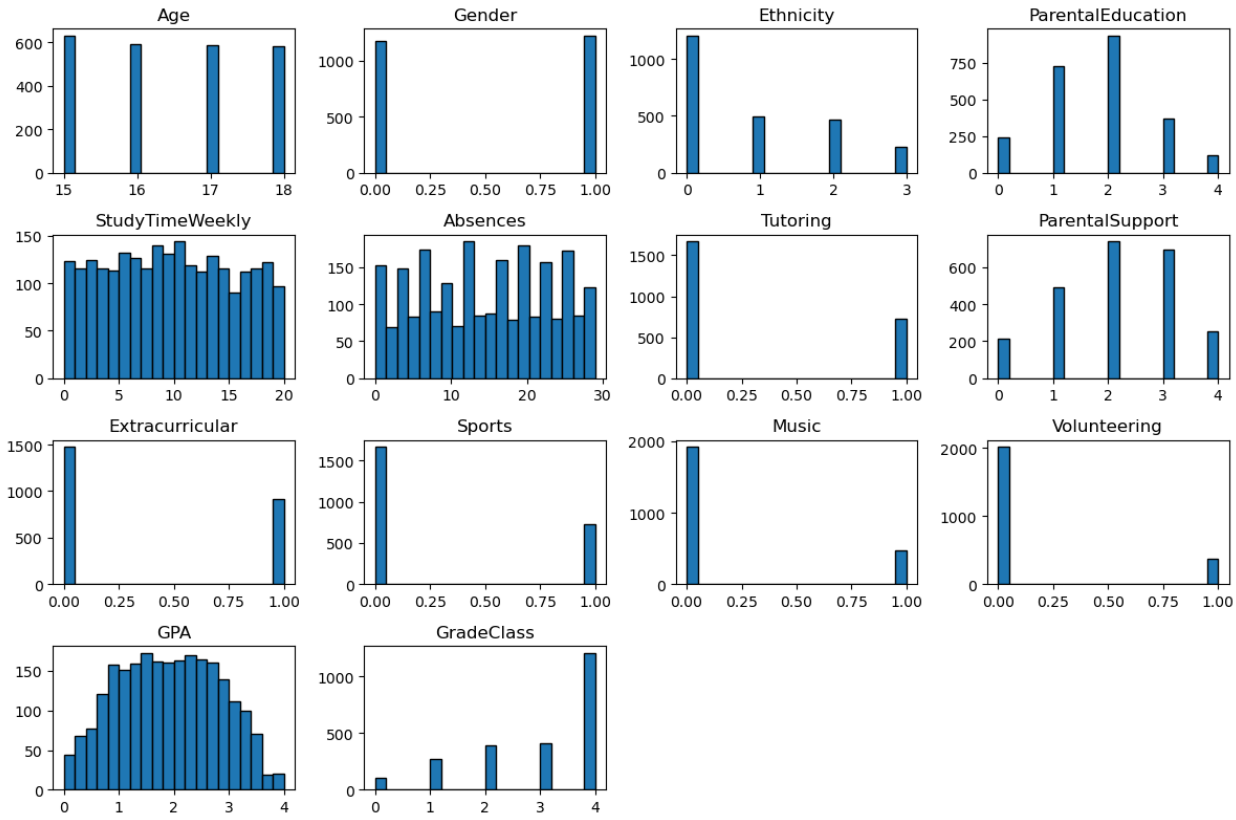

Figure 2 Data Histograms for All Columns

3. Data Preprocessing

Techniques applied and justification of my choice: Most of the data was already preprocessed, as in it was already in numeral form (no words), no outliers could be found using IQR (Figure 3), no values were missing (Figure 4), all categorical data was already encoded into ints (for example ethnicity was encoded into 0, 1, 2, 3). The only preprocessing needed before doing the PCA was to standardize (Figure 5) (because as far as I could find, PCA (Figure 6) wants standardized) the numerical variables that had big differences between them (Absences, StudyTimeWeekly and GPA).

```
Outliers in StudyTimeWeekly:
None


Outliers in Absences:
None


Outliers in GPA:
None
```

Figure 3 Outliers

```
No missing data. Yayyy!
```

Figure 4 Missing Data

```
Data after standardization:
   Age  Gender  Ethnicity  ParentalEducation  StudyTimeWeekly  Absences  \
0   17       1          0                  2         1.780336 -0.890822
1   18       0          0                  1         0.997376 -1.717694
2   15       0          2                  3        -0.984045  1.353542
3   17       1          0                  3         0.045445 -0.063951
4   17       1          0                  2        -0.902311  0.290422

   Tutoring  ParentalSupport  Extracurricular  Sports  Music  Volunteering  \
0         1                2                0       0      1             0
1         0                1                0       0      0             0
2         0                2                0       0      0             0
3         0                3                1       0      0             0
4         1                3                0       0      0             0

        GPA
0  1.118086
1  1.242374
2 -1.960277
3  0.161790
4 -0.675573
```

Figure 5 Standardized Data

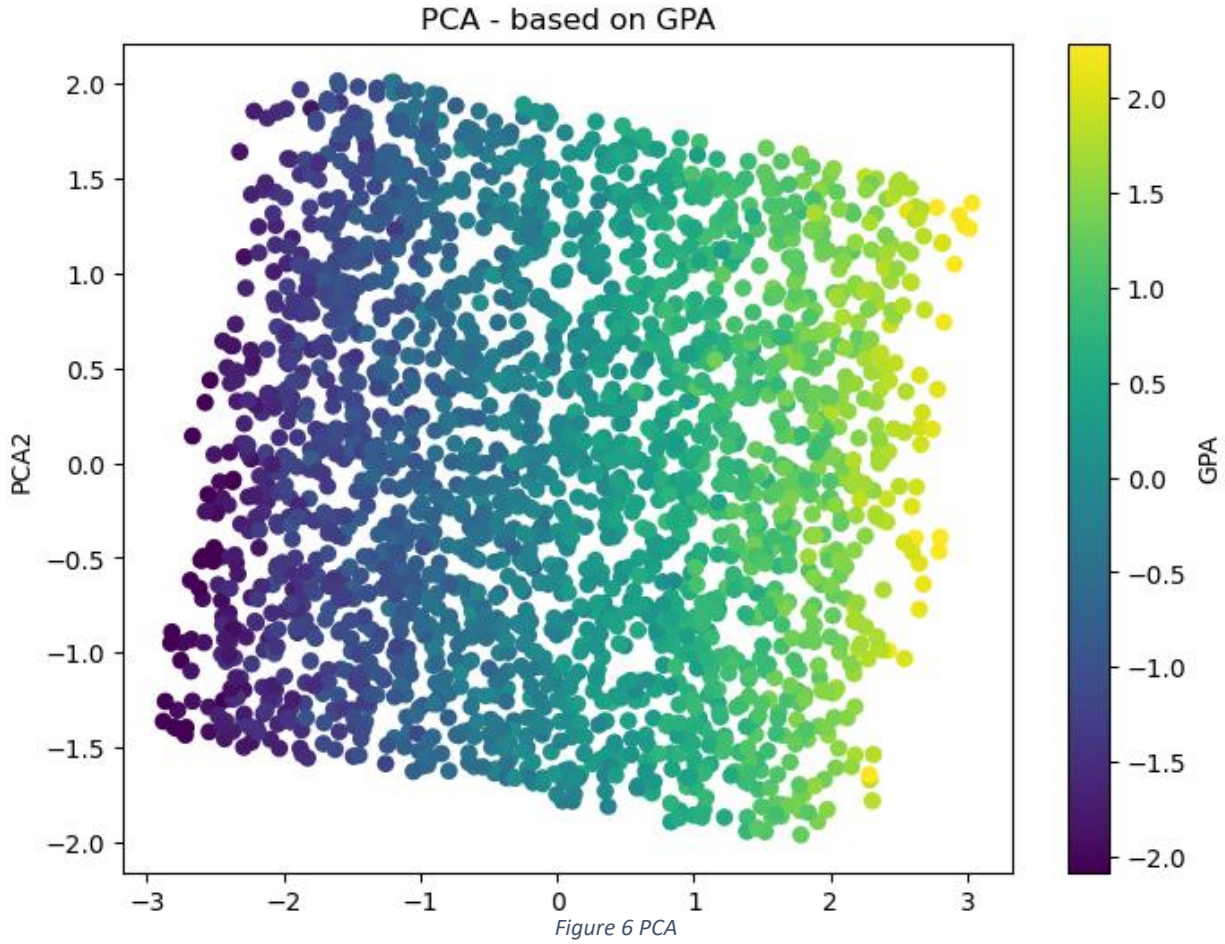

Figure 6 PCA

4. Handling Imbalance

Methods tested and their impact on results: I used both SMOTE and ADASYN to fix the data imbalance with different results as seen in Figure 7 and 8. This will lead to model performance differences later on as well.

```
Class distribution before SMOTE:
GradeClass
4    974
3    328
2    306
1    220
0     85
Name: count, dtype: int64

Class distribution after SMOTE:
GradeClass
4    974
1    974
2    974
3    974
0    974
Name: count, dtype: int64
```

Figure 7 Class Distribution SMOTE

```
Class distribution before ADASYN:
GradeClass
4    974
3    328
2    306
1    220
0     85
Name: count, dtype: int64

Class distribution after ADASYN:
GradeClass
2    1001
0    1001
4     974
3     954
1     915
Name: count, dtype: int64
```

Figure 8 Class Distribution ADASYN

5. ML Models

Model descriptions:

- **Linear regression** is a type of supervised machine learning algorithm that computes the linear relationship between the dependent variable and one or more independent features by fitting a linear equation to observed data.
- **A random forest** is a meta estimator that fits a number of decision tree regressors on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
- **Support vector regression** is a type of support vector machine that is used for regression tasks. It tries to find a function that best predicts the continuous output value for a given input value.
- **K-Nearest Neighbors** is one of the simplest and most intuitive machine learning algorithms. While it is commonly associated with classification tasks, KNN can also be used for regression.
- **Gradient Boosting** is a popular boosting algorithm in machine learning used for classification and regression tasks. Boosting is one kind of ensemble Learning method which trains the model sequentially and each new model tries to correct the previous model.
- **A Voting Regressor** can be defined as a special method that combines or 'ensembles' multiple regression models and overperforms the individual models present as its estimators.

Parameters Tested: I didn't give the models any parameters (so I used the default ones). SMOTE and ADASYN got the random_state = 17 (my birthday) and ADASYN also got n_neighbours = 7 (my birth month).

Performance Results: The performance results can be seen in figures 9 and 10 for SMOTE, figures 11 and 12 for ADASYN

```
Model Performance Comparison (Ranked by Accuracy):
SVR: Accuracy=0.6889, Precision=0.7174, Recall=0.6889, F1=0.6978
Random Forest Regressor: Accuracy=0.5637, Precision=0.6580, Recall=0.5637, F1=0.5839
Voting Regressor: Accuracy=0.5637, Precision=0.6649, Recall=0.5637, F1=0.5852
Gradient Boosting Regressor: Accuracy=0.4760, Precision=0.6483, Recall=0.4760, F1=0.5016
KNN Regressor: Accuracy=0.4593, Precision=0.5866, Recall=0.4593, F1=0.4815
Linear Regression: Accuracy=0.4405, Precision=0.6131, Recall=0.4405, F1=0.4655
```

Figure 9 Models with SMOTE Accuracy

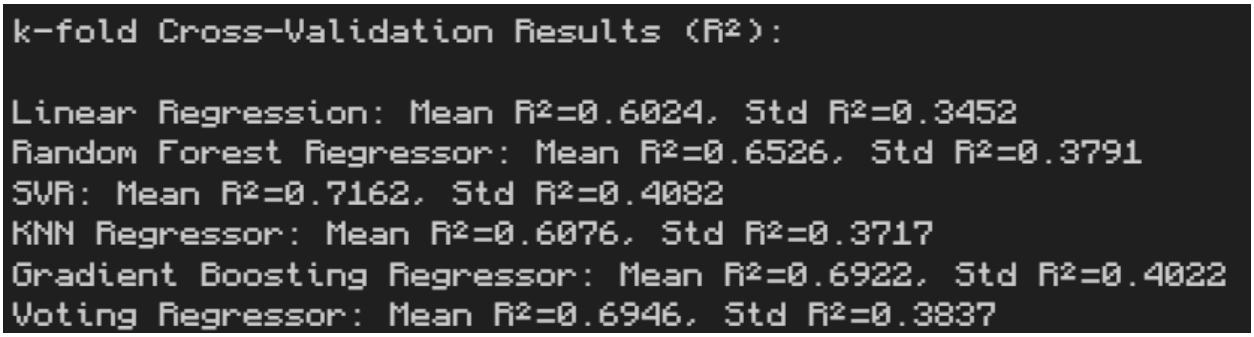

Figure 10 Models with SMOTE Cross-Validation

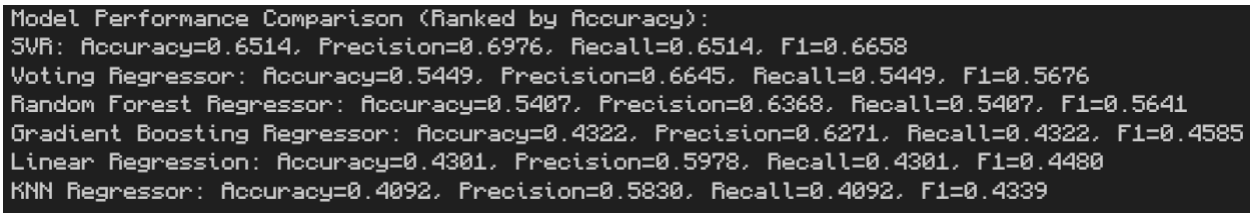

Figure 11 Models with ADASYN Accuracy

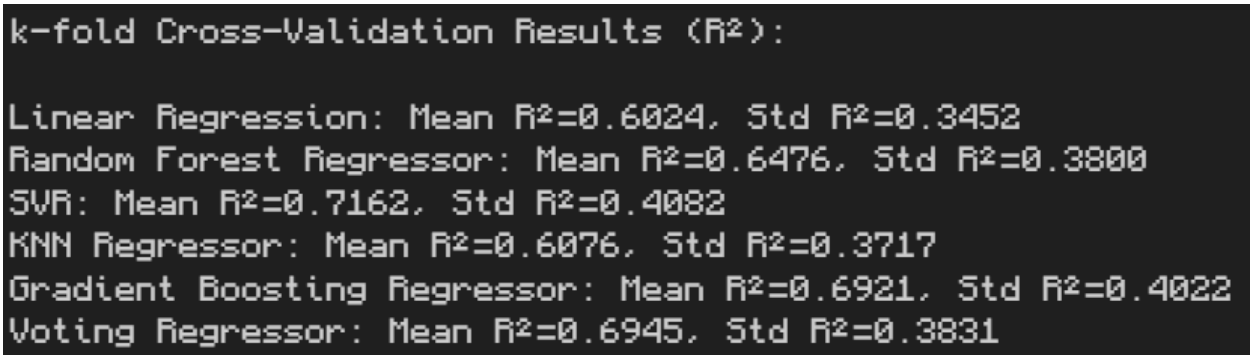

Figure 12 Models with ADASYN Cross-Validation

6. Conclusion

Summary of Findings: SMOTE seems slightly better than ADASYN for this particular dataset. The dataset was mostly preprocessed for machine learning purposes, being complete, already numerical and had no outliers (at least according to IQR)

Best-Performing Model: SVR seems to be the best model for this particular dataset in terms of Accuracy, Precision, Recall, F1 and R^2 mean for the Cross-Validation with 7 folds (so all the ones necessary for the homework)

Future Ways of Improving: Spending more time on the code will always lead to improvements, optimizing parameters and so on. Using other models might also lead to better results.

Webography Used:

https://www.geeksforgeeks.org/ml-linear-regression/#what-is-linear-regression

https://scikit-learn.org/1.5/modules/generated/sklearn.ensemble.RandomForestRegressor.html

https://www.geeksforgeeks.org/support-vector-regression-svr-using-linear-and-non-linear-kernels-in-scikit-learn/

https://www.geeksforgeeks.org/k-nearest-neighbors-knn-regression-with-scikit-learn/

https://www.geeksforgeeks.org/ml-gradient-boosting/

https://www.geeksforgeeks.org/voting-regressor/

https://ca.indeed.com/career-advice/career-development/outliers-in-statistics

https://www.datacamp.com/tutorial/normalization-vs-standardization

https://stackoverflow.com/questions/40795141/pca-for-categorical-features

https://saturncloud.io/blog/what-is-sklearn-pca-explained-variance-and-explained-variance-ratio-difference/

https://builtin.com/machine-learning/pca-in-python

https://realpython.com/linear-regression-in-python/

https://www.geeksforgeeks.org/random-forest-regression-in-python/

https://medium.com/data-and-beyond/voting-regressor-intuition-and-implementation-0359771b5204