Politecnico di Torino

Data Science and Engineering

Mathematics in Machine Learning

Ali Ghasemi (S289223)

Dataset: Higher Education Students Performance Evaluation Dataset

Professor: Francesco Vaccarino, Mauro Gasparini

TABLE OF CONTENTS

Introduction

[1] Education has a vital and increasing importance almost for all countries to accelerate their development. Well-educated people provide more benefits to their countries and for that reason, classification of students' performance before they enter exams or taking courses is also gained an importance. Improvement of education quality must be performed during the active semester to improve students' personal performance to response to this expectation. To provide this, some of the main indicators are students' personal information, educational preferences, and family properties. The dataset is gathered using questionnaire results that consists of these main indicators, of three different courses of two faculties to classify students' final grade performances and to determine some efficient machine learning algorithms for this task.

Dataset Description

Dataset is consisted of 33 attributes and 145 instances. Each row of the dataset represents a student and his/her/their information.

Student ID

1- Student Age (1: 18-21, 2: 22-25, 3: above 26)

2- Sex (1: female, 2: male)

3- Graduated high-school type: (1: private, 2: state, 3: other)

4- Scholarship type: (1: None, 2: 25%, 3: 50%, 4: 75%, 5: Full)

5- Additional work: (1: Yes, 2: No)

6- Regular artistic or sports activity: (1: Yes, 2: No)

7- Do you have a partner: (1: Yes, 2: No)

8- Total salary if available (1: USD 135-200, 2: USD 201-270, 3: USD 271-340, 4: USD 341-410, 5: above 410)

9- Transportation to the university: (1: Bus, 2: Private car/taxi, 3: bicycle, 4: Other)

10- Accommodation type in Cyprus: (1: rental, 2: dormitory, 3: with family, 4: Other)

11- Mother’s education: (1: primary school, 2: secondary school, 3: high school, 4: university, 5: MSc., 6: Ph.D.)

12- Father’s education: (1: primary school, 2: secondary school, 3: high school, 4: university, 5: MSc., 6: Ph.D.)

13- Number of sisters/brothers (if available): (1: 1, 2: 2, 3: 3, 4: 4, 5: 5 or above)

14- Parental status: (1: married, 2: divorced, 3: died - one of them or both)

15- Mother’s occupation: (1: retired, 2: housewife, 3: government officer, 4: private sector employee, 5: self-employment, 6: other)

16- Father’s occupation: (1: retired, 2: government officer, 3: private sector employee, 4: self-employment, 5: other)

17- Weekly study hours: (1: None, 2: <5 hours, 3: 6-10 hours, 4: 11-20 hours, 5: more than 20 hours)

18- Reading frequency (non-scientific books/journals): (1: None, 2: Sometimes, 3: Often)

19- Reading frequency (scientific books/journals): (1: None, 2: Sometimes, 3: Often)

20- Attendance to the seminars/conferences related to the department: (1: Yes, 2: No)

21- Impact of your projects/activities on your success: (1: positive, 2: negative, 3: neutral)

22- Attendance to classes (1: always, 2: sometimes, 3: never)

23- Preparation to midterm exams 1: (1: alone, 2: with friends, 3: not applicable)

24- Preparation to midterm exams 2: (1: closest date to the exam, 2: regularly during the semester, 3: never)

25- Taking notes in classes: (1: never, 2: sometimes, 3: always)

26- Listening in classes: (1: never, 2: sometimes, 3: always)

27- Discussion improves my interest and success in the course: (1: never, 2: sometimes, 3: always)

28- Flip-classroom: (1: not useful, 2: useful, 3: not applicable)

29- Cumulative grade point average in the last semester (/4.00): (1: <2.00, 2: 2.00-2.49, 3: 2.50-2.99, 4: 3.00-3.49, 5: above 3.49)

30- Expected Cumulative grade point average in the graduation (/4.00): (1: <2.00, 2: 2.00-2.49, 3: 2.50-2.99, 4: 3.00-3.49, 5: above 3.49)

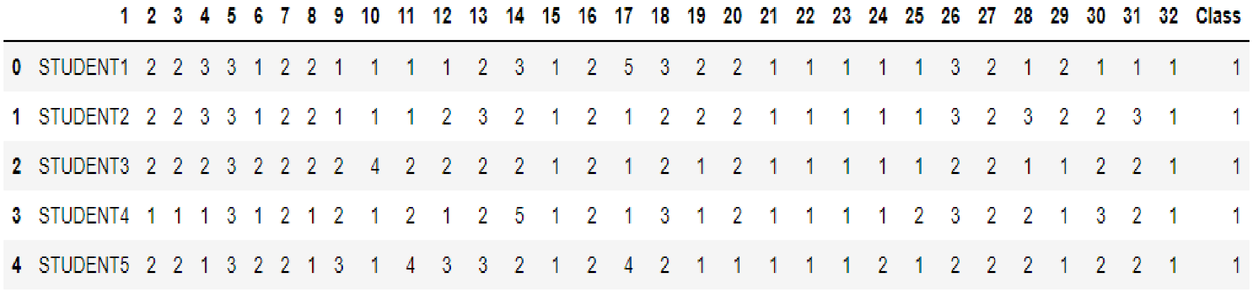
31- Course ID

32- OUTPUT Grade (0: Fail, 1: DD, 2: DC, 3: CC, 4: CB, 5: BB, 6: BA, 7: AA)

32nd attribute is the target. It represents the grade of the student.

There are classed that students must be classified into.

The dataset does not contain any NA (not available) or missing data.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Class
0	STUDENT1	2	2	3	3	1	2	2	1	1	1	1	2	3	1	2	5	3	2	2	1	1	1	1	1	3	2	1	2	1	1	1	1
1	STUDENT2	2	2	3	3	1	2	2	1	1	1	2	3	2	1	2	1	2	2	2	1	1	1	1	1	3	2	3	2	2	3	1	1
2	STUDENT3	2	2	2	3	2	2	2	2	4	2	2	2	2	1	2	1	2	1	2	1	1	1	1	1	2	2	1	1	2	2	1	1
3	STUDENT4	1	1	1	3	1	2	1	2	1	2	1	2	5	1	2	1	3	1	2	1	1	1	1	2	3	2	2	1	3	2	1	1
4	STUDENT5	2	2	1	3	2	2	1	3	1	4	3	3	2	1	2	4	2	1	1	1	1	1	2	1	2	2	2	1	2	2	1	1

Figure 1: a small representation of the dataset

Data Exploration

As the first step, data must be checked for being imbalanced. One of the best ways to come to a better understanding of the data, is to use visualization.

Imbalanced data refers to those types of datasets where the target class has an uneven distribution of observations. An imbalanced dataset leads to a low performance of the model

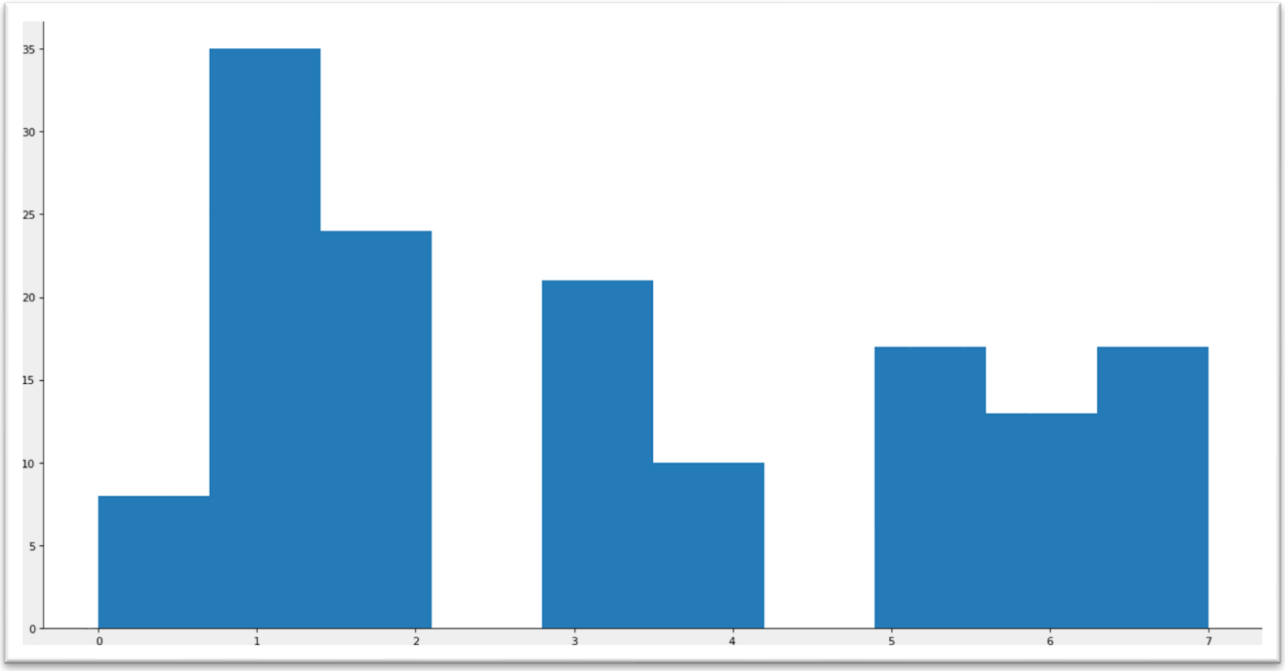


Figure 2: Number of instances of different labels across the dataset

As it can be seen in figure 2, it’s obvious that this dataset is imbalanced, and this issue must be handled.

There are various ways of overcoming the issue of an imbalanced dataset such as:

Undersampling (balances the dataset by reducing the size of the abundant class),

Oversampling (used to balance the dataset when the quantity of data is insufficient), Etc.

The technique used in this thesina is random oversampling. Random Oversampling is a resampling method. Resampling involves creating a new transformed version of the training dataset in which the selected examples have a different class distribution. Random oversampling Randomly duplicate examples in the minority class.

Figure 3 demonstrates the labels after performing random oversampling on dataset

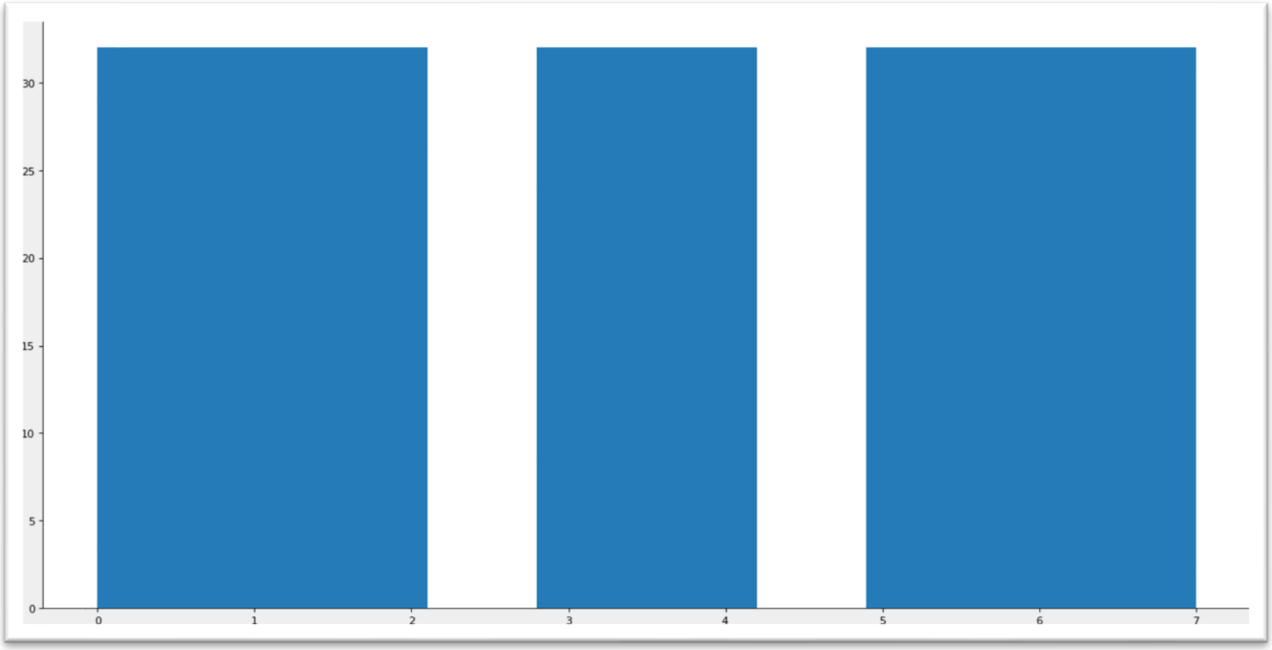
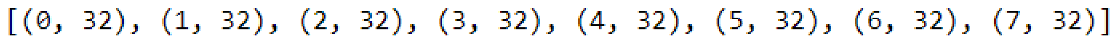


Figure 3: dataset labels after random oversampling

As it can be seen in figure 3, it’s obvious that dataset labels are balanced after performing random oversampling.

Also, the number of instances from each label are mentioned in figure 4:



```
[(0, 32), (1, 32), (2, 32), (3, 32), (4, 32), (5, 32), (6, 32), (7, 32)]
```

Figure 4: instances of each label in the dataset

Standardization

Standardization of a dataset is a common requirement for many machine learning estimators. Estimators might behave badly if the individual features do not more or less look like standard normally distributed data (e.g. Gaussian with 0 mean and unit variance).

For instance, many elements used in the objective function of a learning algorithm (such as the RBF kernel of Support Vector Machines or the L1 and L2 regularizers of linear models) assume that all features are centered around 0 and have variance in the same order. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

The result of standardization (or Z-score normalization) is that the features will be rescaled to ensure the mean and the standard deviation to be 0 and 1, respectively.

The below formula describes this method.

$$x_{stand} = \frac{x - u(x)}{sd(x)}$$

In which “$u(x)$” represent the mean of the training samples and “$sd(x)$” represent the standard deviation of the training samples.

Correlation

Correlation between two variables demonstrates the relation between those two variables. One way to quantify this relationship is to use the Pearson correlation coefficient, which is a measure of the linear association between two variables. It has a value between -1 and 1 where:

-1 indicates a perfectly negative linear correlation between two variables

0 indicates no linear correlation between two variables

1 indicates a perfectly positive linear correlation between two variables-1 indicates a perfectly negative linear correlation between two variables

0 indicates no linear correlation between two variables

1 indicates a perfectly positive linear correlation between two variables

The Pearson product-moment correlation coefficient (or Pearson correlation coefficient, for short) is a measure of the strength of a linear association between two variables and is denoted by “r”. Basically, a Pearson product-moment correlation attempts to draw a line of best fit through the data of two variables, and the Pearson correlation coefficient, r, indicates how far away all these data points are to this line of best fit (i.e., how well the data points fit this new model/line of best fit).

The formula below represents the Pearson’s Correlation Coefficient:

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 - \sum(y_i - \bar{y})^2}}$$

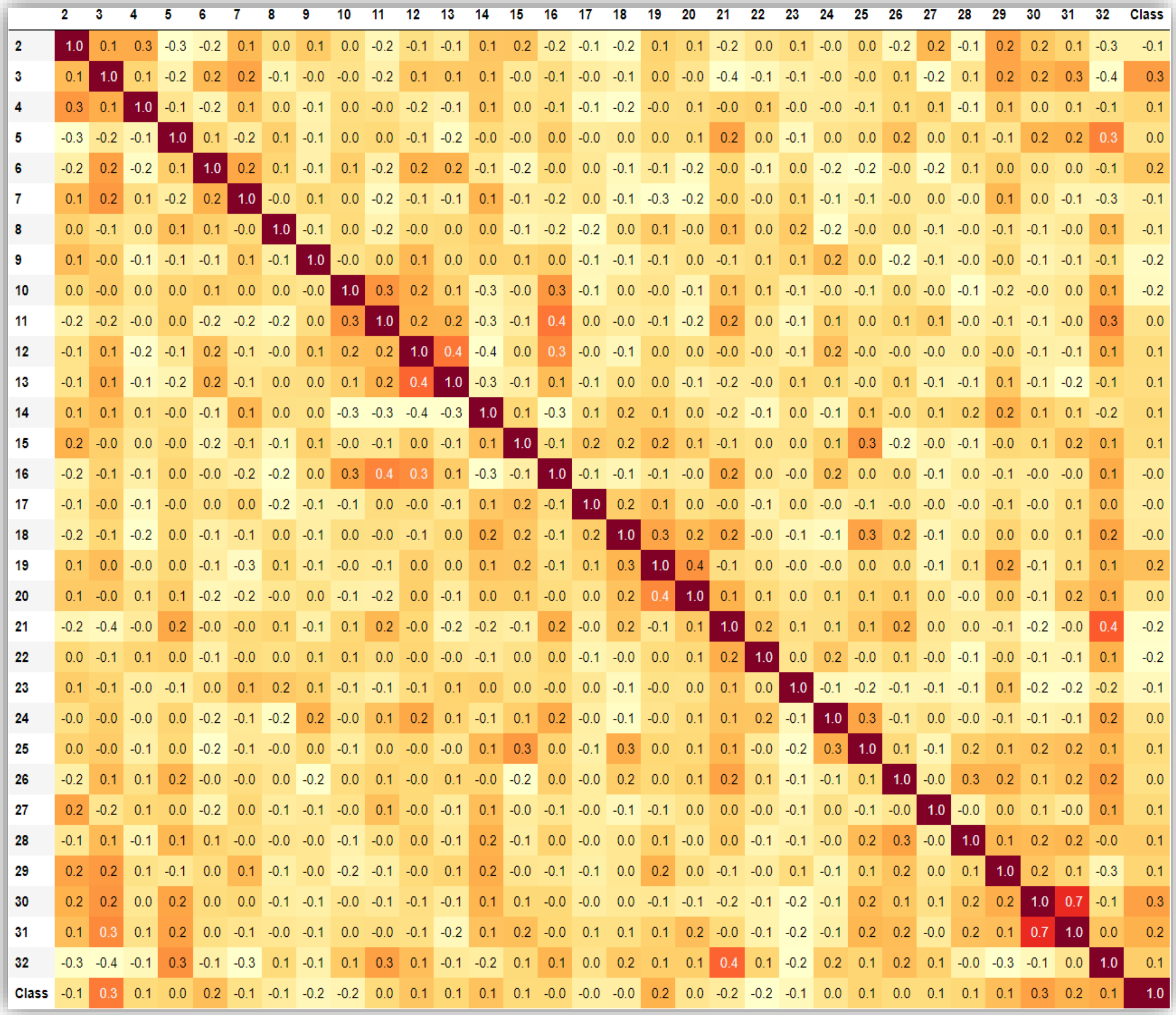
In which r represents the correlation coefficient, x_i represents the values of the x-variable in a sample, $\bar{x}$ represents the mean of the values of the x-variable, y_i represents the values of the y-variable in a sample, and $\bar{y}$ represents the mean of the values of the y-variable

The correlation between different variables of the dataset must be found.

Figure 4 demonstrates the correlation between different attributes of the dataset.

Highly correlated variables lead to redundant classification, and they affect each model in a different way. For linear models (e.g., linear regression or logistic regression), Multicollinearity can yield solutions that are wildly varying and possibly numerically unstable. Random forests can be good at detecting interactions between different features, but highly correlated features can mask these interactions. Let’s have examine an example for sake of intuition, we can see that there is a high correlation between 30 (Expected Cumulative grade point average in the graduation) and 31 (Course ID). and for example, there's no correlation between 14 (Parental status) and 4(Scholarship type)

Based on Figure 4, there aren’t much highly correlated variables in this dataset. But still PCA will be tested to get a better result from models compared to the output of the results without applying PCA on the dataset.



	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Class
2	1.0	0.1	0.3	-0.3	-0.2	0.1	0.0	0.1	0.0	-0.2	-0.1	-0.1	0.1	0.2	-0.2	-0.1	-0.2	0.1	0.1	-0.2	0.0	0.1	-0.0	0.0	-0.2	0.2	-0.1	0.2	0.2	0.1	-0.3	-0.1
3	0.1	1.0	0.1	-0.2	0.2	0.2	-0.1	-0.0	-0.0	-0.2	0.1	0.1	0.1	-0.0	-0.1	-0.0	-0.1	0.0	-0.0	-0.4	-0.1	-0.1	-0.0	-0.0	0.1	-0.2	0.1	0.2	0.2	0.3	-0.4	0.3
4	0.3	0.1	1.0	-0.1	-0.2	0.1	0.0	-0.1	0.0	-0.0	-0.2	-0.1	0.1	0.0	-0.1	-0.1	-0.2	-0.0	0.1	-0.0	0.1	-0.0	-0.0	-0.1	0.1	0.1	-0.1	0.1	0.0	0.1	-0.1	0.1
5	-0.3	-0.2	-0.1	1.0	0.1	-0.2	0.1	-0.1	0.0	0.0	-0.1	-0.2	-0.0	-0.0	0.0	-0.0	0.0	0.0	0.1	0.2	0.0	-0.1	0.0	0.0	0.2	0.0	0.1	-0.1	0.2	0.2	0.3	0.0
6	-0.2	0.2	-0.2	0.1	1.0	0.2	0.1	-0.1	0.1	-0.2	0.2	0.2	-0.1	-0.2	-0.0	0.0	-0.1	-0.1	-0.2	-0.0	-0.1	0.0	-0.2	-0.2	-0.0	-0.2	0.1	0.0	0.0	0.0	-0.1	0.2
7	0.1	0.2	0.1	-0.2	0.2	1.0	-0.0	0.1	0.0	-0.2	-0.1	-0.1	0.1	-0.1	-0.2	0.0	-0.1	-0.3	-0.2	-0.0	-0.0	0.1	-0.1	-0.1	-0.0	0.0	-0.0	0.1	0.0	-0.1	-0.3	-0.1
8	0.0	-0.1	0.0	0.1	0.1	-0.0	1.0	-0.1	0.0	-0.2	-0.0	0.0	0.0	-0.1	-0.2	-0.2	0.0	0.1	-0.0	0.1	0.0	0.2	-0.2	-0.0	0.0	-0.1	-0.0	-0.1	-0.1	-0.0	0.1	-0.1
9	0.1	-0.0	-0.1	-0.1	-0.1	0.1	-0.1	1.0	-0.0	0.0	0.1	0.0	0.0	0.1	0.0	-0.1	-0.1	-0.1	0.0	-0.1	0.1	0.1	0.2	0.0	-0.2	-0.1	-0.0	-0.0	-0.1	-0.1	-0.1	-0.2
10	0.0	-0.0	0.0	0.0	0.1	0.0	0.0	-0.0	1.0	0.3	0.2	0.1	-0.3	-0.0	0.3	-0.1	0.0	-0.0	-0.1	0.1	0.1	-0.1	-0.0	-0.1	0.0	-0.0	-0.1	-0.2	-0.0	0.0	0.1	-0.2
11	-0.2	-0.2	-0.0	0.0	-0.2	-0.2	-0.2	0.0	0.3	1.0	0.2	0.2	-0.3	-0.1	0.4	0.0	-0.0	-0.1	-0.2	0.2	0.0	-0.1	0.1	0.0	0.1	0.1	-0.0	-0.1	-0.1	-0.0	0.3	0.0
12	-0.1	0.1	-0.2	-0.1	0.2	-0.1	-0.0	0.1	0.2	0.2	1.0	0.4	-0.4	0.0	0.3	-0.0	-0.1	0.0	0.0	-0.0	-0.0	-0.1	0.2	-0.0	-0.0	-0.0	0.0	-0.0	-0.1	-0.1	0.1	0.1
13	-0.1	0.1	-0.1	-0.2	0.2	-0.1	0.0	0.0	0.1	0.2	0.4	1.0	-0.3	-0.1	0.1	-0.1	0.0	0.0	-0.1	-0.2	-0.0	0.1	0.1	-0.0	0.1	-0.1	-0.1	0.1	-0.1	-0.2	-0.1	0.1
14	0.1	0.1	0.1	-0.0	-0.1	0.1	0.0	0.0	-0.3	-0.3	-0.4	-0.3	1.0	0.1	-0.3	0.1	0.2	0.1	0.0	-0.2	-0.1	0.0	-0.1	0.1	-0.0	0.1	0.2	0.2	0.1	0.1	-0.2	0.1
15	0.2	-0.0	0.0	-0.0	-0.2	-0.1	-0.1	0.1	-0.0	-0.1	0.0	-0.1	0.1	1.0	-0.1	0.2	0.2	0.2	0.1	-0.1	0.0	0.0	0.1	0.3	-0.2	-0.0	-0.1	-0.0	0.1	0.2	0.1	0.1
16	-0.2	-0.1	-0.1	0.0	-0.0	-0.2	-0.2	0.0	0.3	0.4	0.3	0.1	-0.3	-0.1	1.0	-0.1	-0.1	-0.1	-0.0	0.2	0.0	-0.0	0.2	0.0	0.0	-0.1	0.0	-0.1	-0.0	-0.0	0.1	-0.0
17	-0.1	-0.0	-0.1	-0.0	0.0	0.0	-0.2	-0.1	-0.1	0.0	-0.0	-0.1	0.1	0.2	-0.1	1.0	0.2	0.1	0.0	-0.0	-0.1	0.0	-0.0	-0.1	-0.0	-0.0	-0.0	-0.1	-0.0	0.1	0.0	-0.0
18	-0.2	-0.1	-0.2	0.0	-0.1	-0.1	0.0	-0.1	0.0	-0.0	-0.1	0.0	0.2	0.2	-0.1	0.2	1.0	0.3	0.2	0.2	-0.0	-0.1	-0.1	0.3	0.2	-0.1	0.0	0.0	0.0	0.1	0.2	-0.0
19	0.1	0.0	-0.0	0.0	-0.1	-0.3	0.1	-0.1	-0.0	-0.1	0.0	0.0	0.1	0.2	-0.1	0.1	0.3	1.0	0.4	-0.1	0.0	-0.0	-0.0	0.0	0.0	-0.1	0.1	0.2	-0.1	0.1	0.1	0.2
20	0.1	-0.0	0.1	0.1	-0.2	-0.2	-0.0	0.0	-0.1	-0.2	0.0	-0.1	0.0	0.1	-0.0	0.0	0.2	0.4	1.0	0.1	0.1	0.0	0.1	0.1	0.1	0.0	-0.0	0.0	-0.1	0.2	0.1	0.0
21	-0.2	-0.4	-0.0	0.2	-0.0	-0.0	0.1	-0.1	0.1	0.2	-0.0	-0.2	-0.2	-0.1	0.2	-0.0	0.2	-0.1	0.1	1.0	0.2	0.1	0.1	0.1	0.2	0.0	0.0	-0.1	-0.2	-0.0	0.4	-0.2
22	0.0	-0.1	0.1	0.0	-0.1	-0.0	0.0	0.1	0.1	0.0	-0.0	-0.0	-0.1	0.0	0.0	-0.1	-0.0	0.0	0.1	0.2	1.0	0.0	0.2	-0.0	0.1	-0.0	-0.1	-0.0	-0.1	-0.1	0.1	-0.2
23	0.1	-0.1	-0.0	-0.1	0.0	0.1	0.2	0.1	-0.1	-0.1	-0.1	0.1	0.0	0.0	-0.0	0.0	-0.1	-0.0	0.0	0.1	0.0	1.0	-0.1	-0.2	-0.1	-0.1	-0.1	0.1	-0.2	-0.2	-0.2	-0.1
24	-0.0	-0.0	-0.0	0.0	-0.2	-0.1	-0.2	0.2	-0.0	0.1	0.2	0.1	-0.1	0.1	0.2	-0.0	-0.1	-0.0	0.1	0.1	0.2	-0.1	1.0	0.3	-0.1	0.0	-0.0	-0.1	-0.1	-0.1	0.2	0.0
25	0.0	-0.0	-0.1	0.0	-0.2	-0.1	-0.0	0.0	-0.1	0.0	-0.0	-0.0	0.1	0.3	0.0	-0.1	0.3	0.0	0.1	0.1	-0.0	-0.2	0.3	1.0	0.1	-0.1	0.2	0.1	0.2	0.2	0.1	0.1
26	-0.2	0.1	0.1	0.2	-0.0	-0.0	0.0	-0.2	0.0	0.1	-0.0	0.1	-0.0	-0.2	0.0	-0.0	0.2	0.0	0.1	0.2	0.1	-0.1	-0.1	0.1	1.0	-0.0	0.3	0.2	0.1	0.2	0.2	0.0
27	0.2	-0.2	0.1	0.0	-0.2	0.0	-0.1	-0.1	-0.0	0.1	-0.0	-0.1	0.1	-0.0	-0.1	-0.0	-0.1	-0.1	0.0	0.0	-0.0	-0.1	0.0	-0.1	-0.0	1.0	-0.0	0.0	0.1	-0.0	0.1	0.1
28	-0.1	0.1	-0.1	0.1	0.1	-0.0	-0.0	-0.0	-0.1	-0.0	0.0	-0.1	0.2	-0.1	0.0	-0.0	0.0	0.1	-0.0	0.0	-0.1	-0.1	-0.0	0.2	0.3	-0.0	1.0	0.1	0.2	0.2	-0.0	0.1
29	0.2	0.2	0.1	-0.1	0.0	0.1	-0.1	-0.0	-0.2	-0.1	-0.0	0.1	0.2	-0.0	-0.1	-0.1	0.0	0.2	0.0	-0.1	-0.0	0.1	-0.1	0.1	0.2	0.0	0.1	1.0	0.2	0.1	-0.3	0.1
30	0.2	0.2	0.0	0.2	0.0	0.0	-0.1	-0.1	-0.0	-0.1	-0.1	-0.1	0.1	0.1	-0.0	-0.0	0.0	-0.1	-0.1	-0.2	-0.1	-0.2	-0.1	0.2	0.1	0.1	0.2	0.2	1.0	0.7	-0.1	0.3
31	0.1	0.3	0.1	0.2	0.0	-0.1	-0.0	-0.1	0.0	-0.0	-0.1	-0.2	0.1	0.2	-0.0	0.1	0.1	0.1	0.2	-0.0	-0.1	-0.2	-0.1	0.2	0.2	-0.0	0.2	0.1	0.7	1.0	0.0	0.2
32	-0.3	-0.4	-0.1	0.3	-0.1	-0.3	0.1	-0.1	0.1	0.3	0.1	-0.1	-0.2	0.1	0.1	0.0	0.2	0.1	0.1	0.4	0.1	-0.2	0.2	0.1	0.2	0.1	-0.0	-0.3	-0.1	0.0	1.0	0.1
Class	-0.1	0.3	0.1	0.0	0.2	-0.1	-0.1	-0.2	-0.2	0.0	0.1	0.1	0.1	0.1	-0.0	-0.0	-0.0	0.2	0.0	-0.2	-0.2	-0.1	0.0	0.1	0.0	0.1	0.1	0.1	0.3	0.2	0.1	1.0

Figure 4: correlation map

Outliers

[3] Outliers that once upon a time regarded as noisy data in statistics, have turned out to be an important problem which is being researched in diverse fields of research and application domains. Many outlier detection techniques have been developed specific to certain application domains. Several surveys, research and review articles and books cover outlier detection techniques in machine learning

and statistical domains individually in great details. Outlier detection aims to find patterns in data that do not conform to expected behavior.

In this thesina, violin plots have been used to analyze the dataset regarding the existence of outliers. Figure 5 demonstrates the violin plots that are used to detect outliers.

One the methods that can be used to deal with the outliers' problem is the Z-score method.

Z-score (also called a standard score) gives an idea of how far from the mean a data point is. But more technically it's a measure of how many standard deviations below or above the population mean a raw score is.

The formula for Z-score method is available below:

$$z = \frac{x - u}{\sigma}$$

In which z represent "standard score", x represents "observed value", μ represents "mean of the sample" and, σ represents "standard deviation of the sample"

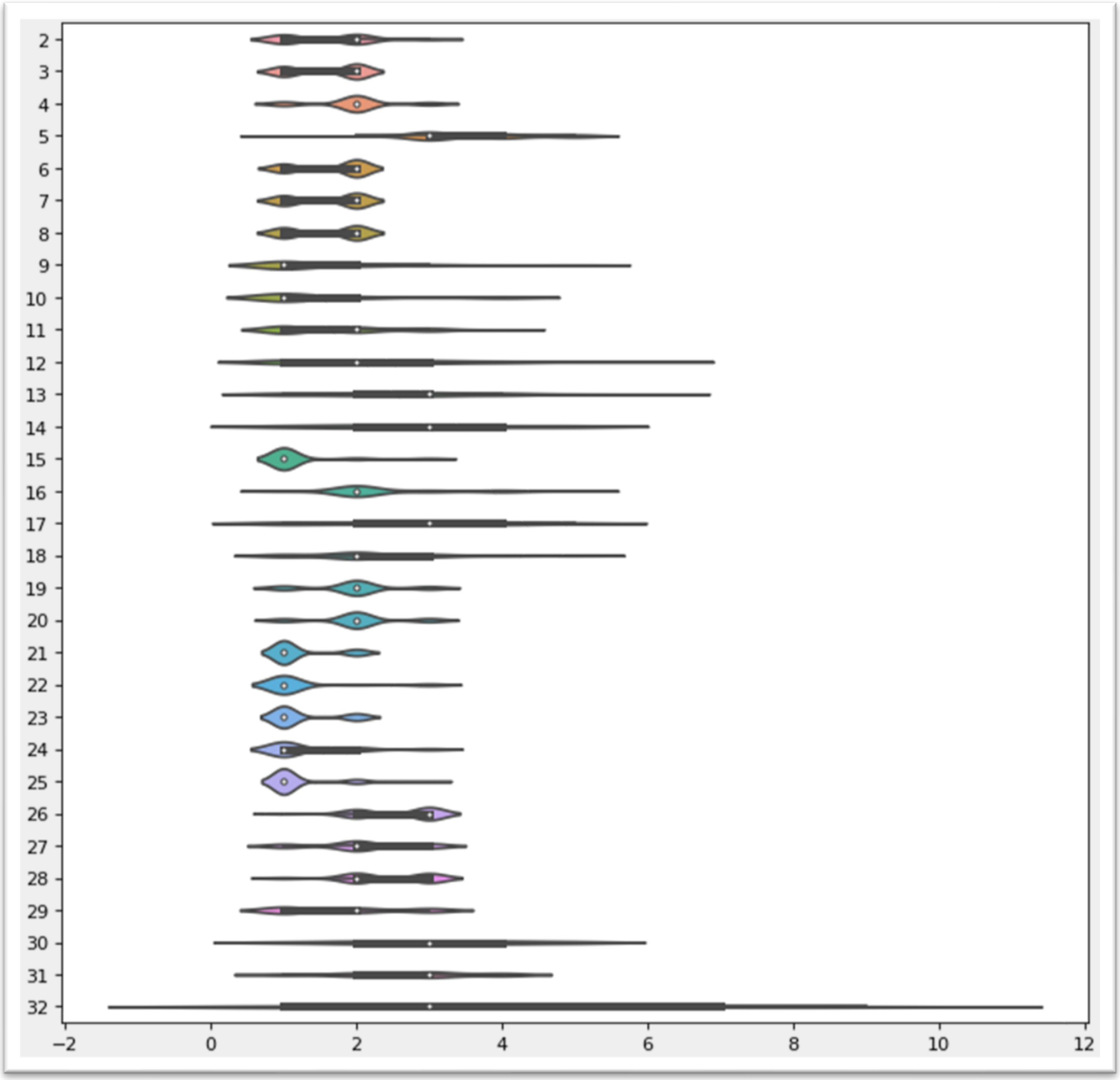


Figure 5

Z-score outlier removal technique removes all points located outside of normal distribution which is higher than 3 standard deviation area. This method will eliminate around 30 rows, but no difficulties will be faced due to low amount of training data since random oversampling is used to both deal with data being imbalanced and data being of low amount.

PCA

Principal component analysis (PCA) is a technique that transforms high-dimensions data into lower-dimensions while retaining as much information as possible. PCA is extremely useful when working with datasets that have a lot of features. Common applications such as image processing, genome research always must deal with thousands-, if not tens of thousands of columns.

While having more data is always great, sometimes they have so much information in them, we would have impossibly long model training time and the curse of dimensionality starts to become a problem. Sometimes, less is more.

One of the main steps for implementing PCA on the data is to find the right number of principal components which is number of features that should be

used in the mode. One the best ways to find that is to visualize dependency of ratio from number of components which is shown in figure 6. After that, we can use the “elbow method” to understand the right number of components to use.

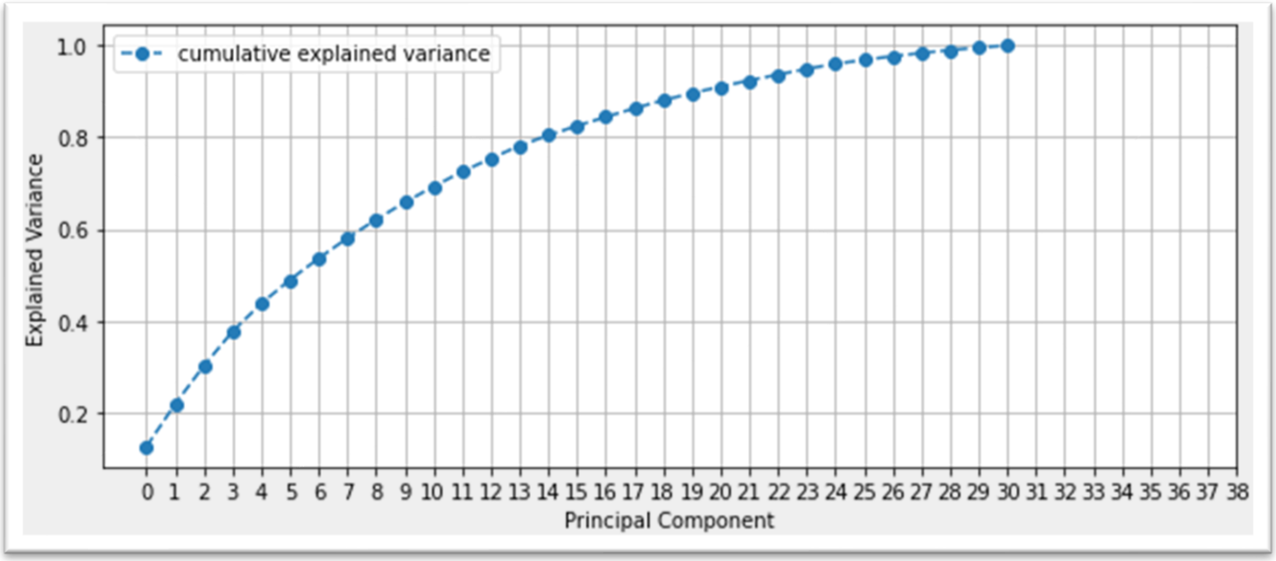


Figure 6: dependency of ratio from number of components

As it can be seen in figure 6, if you increase the number of components more than 14, the slope flattens, and the amount of captured variance does not increase that much. So, 14 is a good choice for the number of principal components.

But since the variables in data are not that correlated, PCA only makes the results worse. This issue will be addressed in the next parts of the thesina.

Model selection and Train/Test procedure

Metrics and Cross Validation

the metric used in this thesina is F1 score. The data we have in the dataset does not completely have the characteristics of the dataset which precision or recall is used on, so F1-score is the best metric for this case since it creates a balance between precision and recall.

F1-score is usually used on imbalanced datasets but the issue data of being imbalanced is solved using random oversampling.

$$F1 = 2 \times \frac{Precision\ \times\ Recall}{Precision\ +\ Recall}$$

One of the other techniques used in this thesina is cross validation.

This process of deciding whether the numerical results quantifying hypothesized relationships between variables, are acceptable as descriptions of the data, is known as validation. Generally, an error estimation for the model is made after training, better known as evaluation of residuals. In this process, a numerical estimate of the difference in predicted and original responses is done, also called the training error. We need to have to data that is not seen by the model to completely evaluate the model, in order to do that, a technique called Cross Validation is used.

There are different ways of using Cross Validation such as the Holdout Method or K-Fold Cross Validation.

The method used in this thesina is the Holdout Method which is removing a part of the training data and using it to get predictions from the model trained on rest of the data. The error estimation then tells how our model is doing on unseen data or the validation set.

This method is demonstrated in figure 7.

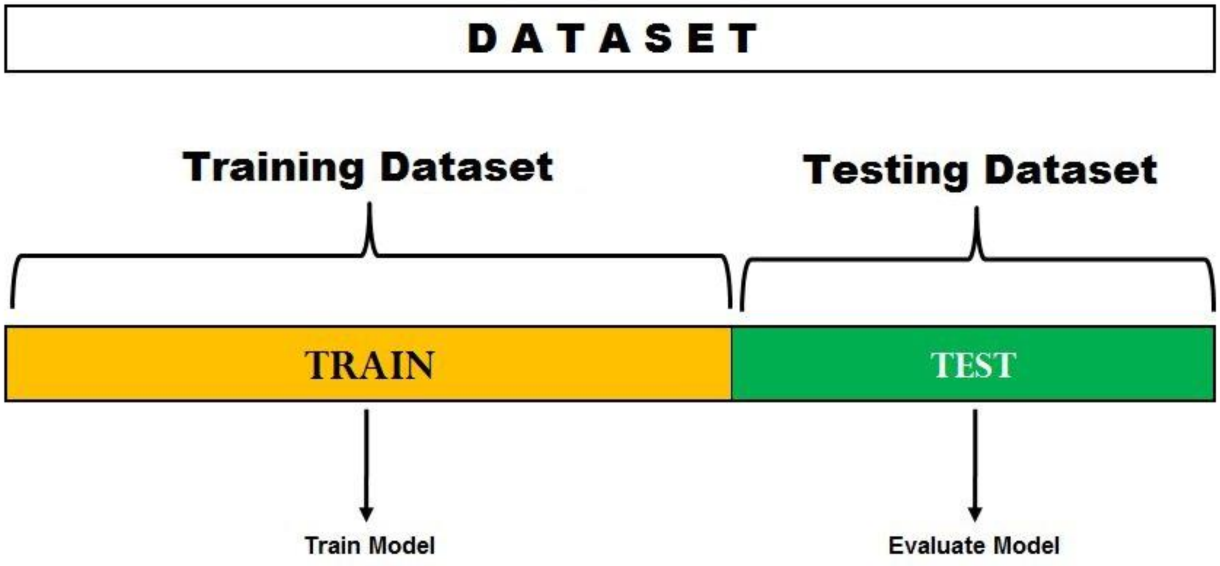


Figure 7: Holdout method

Models

In this part of the thesina we’re going through different models that has been trained on the dataset. All the outputs of the models will be demonstrated both before applying PCA and after applying PCA.

It’s important to mention that since it’s a case of multiclass classification, other algorithms such as **K nearest neighbors** could be used, but to avoid having a lengthy thesina, it hasn’t been mentioned here but the code for testing it is available in the Python notebook.

SVM

Support Vector Machines is considered to be a classification approach, it but can be employed in both types of classification and regression problems. SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplane in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane (MMH) that best divides the dataset into classes.

Support Vectors

Support vectors are the data points, which are closest to the hyperplane. These points will define the separating line better by calculating margins. These points are more relevant to the construction of the classifier.

Hyperplane

A hyperplane is a decision plane which separates between a set of objects having different class memberships.

Margin

A margin is a gap between the two lines on the closest class points. This is calculated as the perpendicular distance from the line to support vectors or closest points. If the margin is larger in between the classes, then it is considered a good margin, a smaller margin is a bad margin.

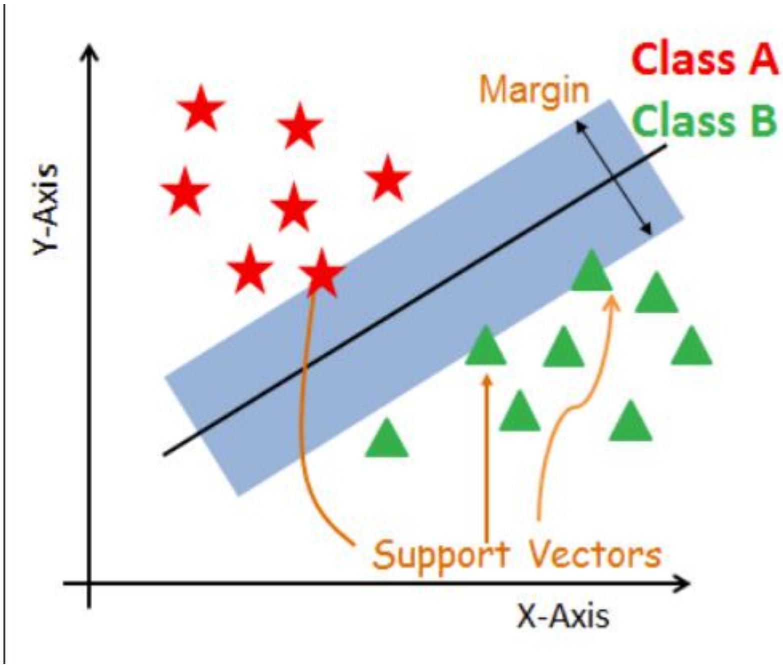
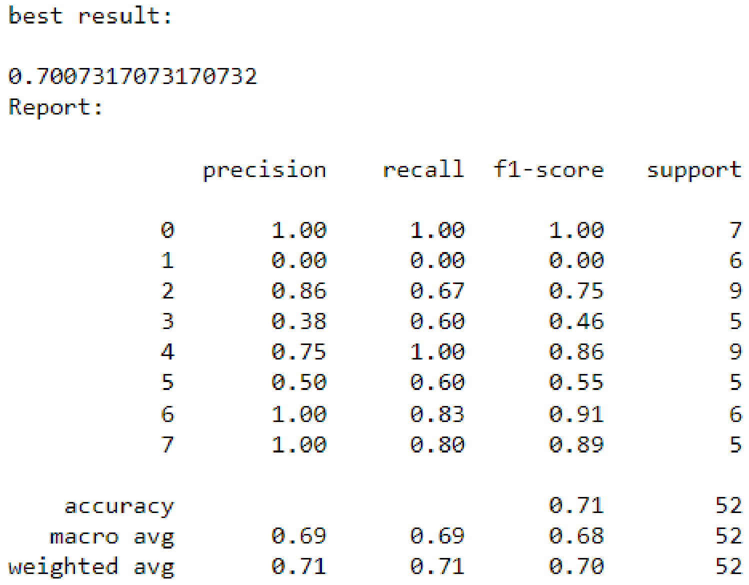


Figure 7

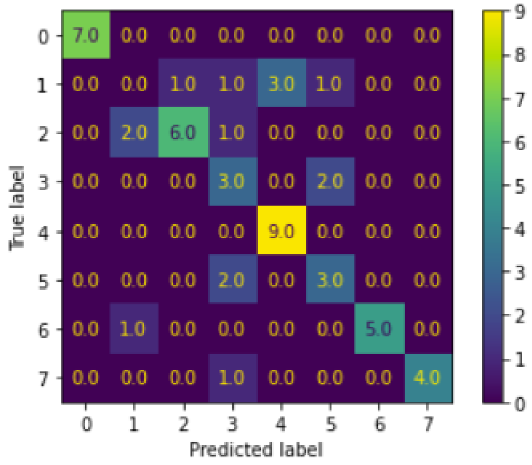
Without using PCA, the loss function used to train the model is "Squared_Hinge" and the maximum number of iterations was set to 100. Using PCA, the loss function used to train the model is "hinge" and the maximum number of iterations was set to 1000. hyperparameters were obtained using grid search technique.

You can see the results with PCA and without PCA below:

Without PCA:



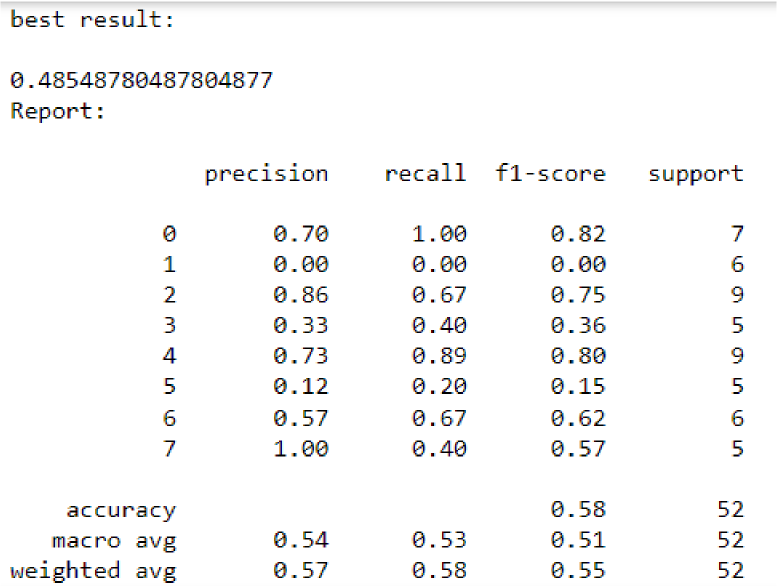
```
best result:

0.7007317073170732
Report:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         7
           1       0.00      0.00      0.00         6
           2       0.86      0.67      0.75         9
           3       0.38      0.60      0.46         5
           4       0.75      1.00      0.86         9
           5       0.50      0.60      0.55         5
           6       1.00      0.83      0.91         6
           7       1.00      0.80      0.89         5

    accuracy                           0.71        52
   macro avg       0.69      0.69      0.68        52
weighted avg       0.71      0.71      0.70        52
```



	0	1	2	3	4	5	6	7
0	7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.0	1.0	1.0	3.0	1.0	0.0	0.0
2	0.0	2.0	6.0	1.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	3.0	0.0	2.0	0.0	0.0
4	0.0	0.0	0.0	0.0	9.0	0.0	0.0	0.0
5	0.0	0.0	0.0	2.0	0.0	3.0	0.0	0.0
6	0.0	1.0	0.0	0.0	0.0	0.0	5.0	0.0
7	0.0	0.0	0.0	1.0	0.0	0.0	0.0	4.0

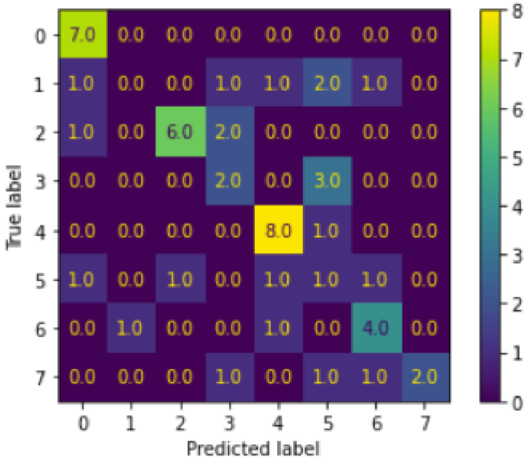
Figure 8

With PCA:



```
best result:

0.48548780487804877
Report:

              precision    recall  f1-score   support

           0       0.70      1.00      0.82         7
           1       0.00      0.00      0.00         6
           2       0.86      0.67      0.75         9
           3       0.33      0.40      0.36         5
           4       0.73      0.89      0.80         9
           5       0.12      0.20      0.15         5
           6       0.57      0.67      0.62         6
           7       1.00      0.40      0.57         5

    accuracy                           0.58        52
   macro avg       0.54      0.53      0.51        52
weighted avg       0.57      0.58      0.55        52
```

Figure 9



	0	1	2	3	4	5	6	7
0	7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	1.0	0.0	0.0	1.0	1.0	2.0	1.0	0.0
2	1.0	0.0	6.0	2.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	2.0	0.0	3.0	0.0	0.0
4	0.0	0.0	0.0	0.0	8.0	1.0	0.0	0.0
5	1.0	0.0	1.0	0.0	1.0	1.0	1.0	0.0
6	0.0	1.0	0.0	0.0	1.0	0.0	4.0	0.0
7	0.0	0.0	0.0	1.0	0.0	1.0	1.0	2.0

Figure 10

As you can see, F1 score highly dropped after using PCA

Random Forests

Random forest is a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forest creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance. Random forest has a variety of applications, such as recommendation engines, image classification and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity, and predict diseases. It lies at the base of the Boruta algorithm, which selects important features in a dataset.

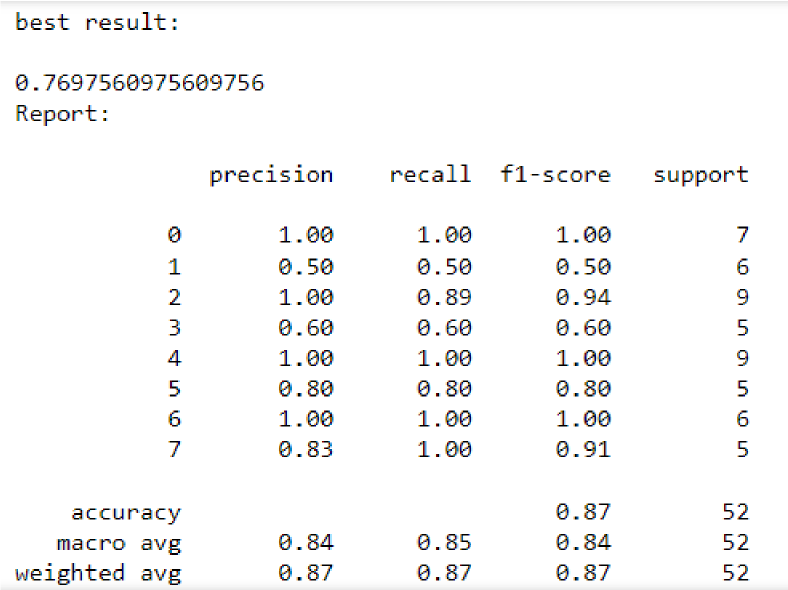
Without using PCA, criterion was set to 'gini', maximum number of features w 'log2' which means the maximum number of features will be equal to,

$$log_2 = (number\ of\ features)$$

And number of estimators which is the number of trees in the forest is equal to 1000

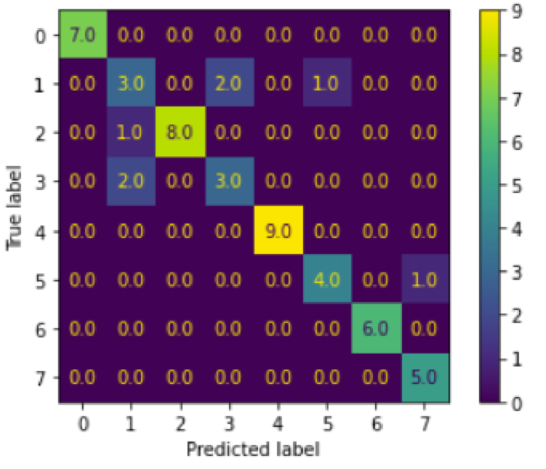
Using PCA, criterion was set to 'entropy 'and the number of estimators was set to 100 and maximum number of features were set to auto.

Results without PCA:



```
best result:

0.7697560975609756
Report:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         7
           1       0.50      0.50      0.50         6
           2       1.00      0.89      0.94         9
           3       0.60      0.60      0.60         5
           4       1.00      1.00      1.00         9
           5       0.80      0.80      0.80         5
           6       1.00      1.00      1.00         6
           7       0.83      1.00      0.91         5

    accuracy                           0.87        52
   macro avg       0.84      0.85      0.84        52
weighted avg       0.87      0.87      0.87        52
```

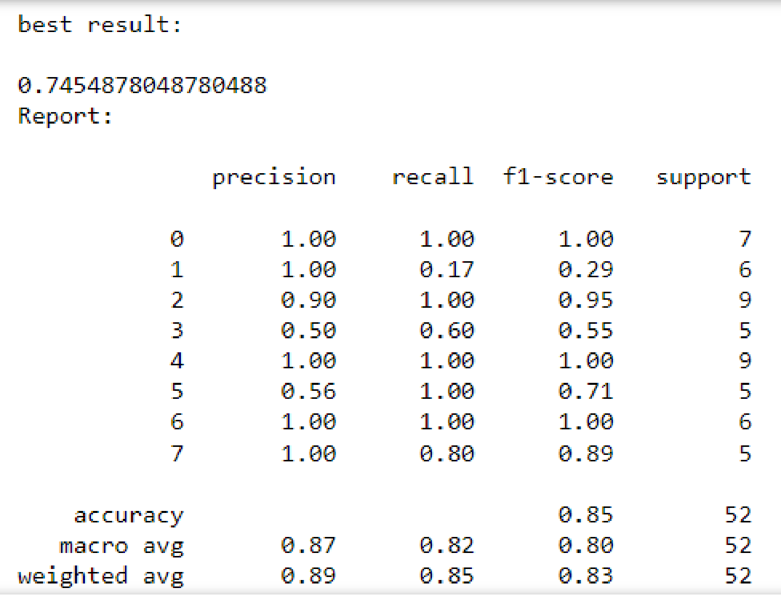
Figure 11



True label \ Predicted label	0	1	2	3	4	5	6	7
0	7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	3.0	0.0	2.0	0.0	1.0	0.0	0.0
2	0.0	1.0	8.0	0.0	0.0	0.0	0.0	0.0
3	0.0	2.0	0.0	3.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	9.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	4.0	0.0	1.0
6	0.0	0.0	0.0	0.0	0.0	0.0	6.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0

Figure 12

Results with PCA:



```
best result:

0.7454878048780488
Report:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00         7
           1       1.00      0.17      0.29         6
           2       0.90      1.00      0.95         9
           3       0.50      0.60      0.55         5
           4       1.00      1.00      1.00         9
           5       0.56      1.00      0.71         5
           6       1.00      1.00      1.00         6
           7       1.00      0.80      0.89         5

    accuracy                           0.85        52
   macro avg       0.87      0.82      0.80        52
weighted avg       0.89      0.85      0.83        52
```

Figure 13

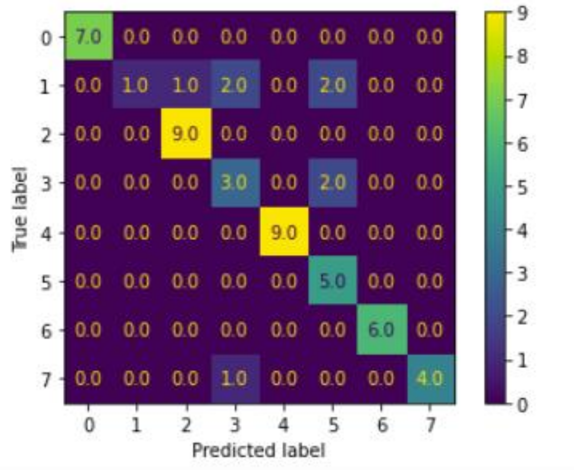


Figure 14

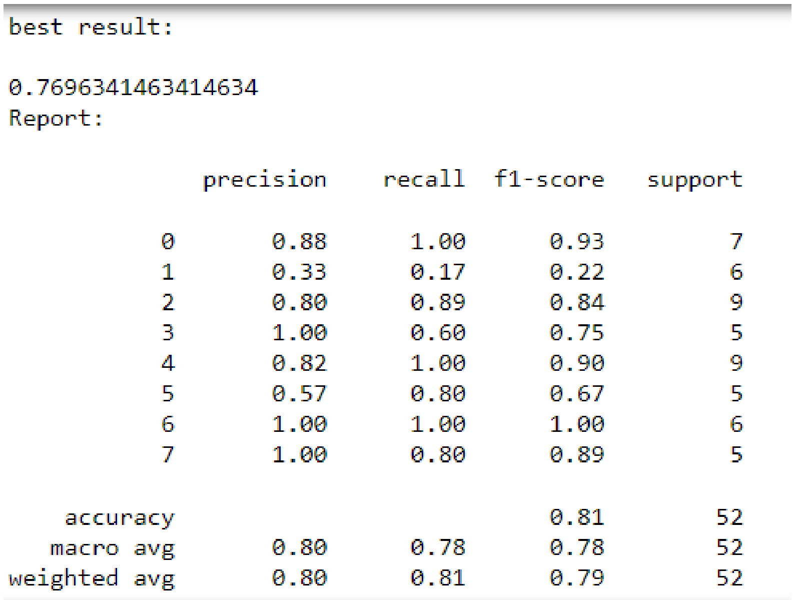
Decision Tress

[4]A decision tree is a classifier expressed as a recursive partition of the instance space. The decision tree consists of nodes that form a rooted tree,meaning it is a directed tree with a node called "root" that has no incoming edges. All other nodes have exactly one incoming edge. A node with outgoing edges is called an internal or test node. All other nodes are called leaves (also known as terminal or decision nodes). In a decision tree, each internal node splits the instance space into two or more sub-spaces according to a certain discrete function of the input attributes values. In the simplest and most frequent case, each test considers a single attribute, such that the instance space is partitioned according to the attribute's value. In the case of numeric attributes, the condition refers to a range. Each leaf is assigned to one class representing the most appropriate target value. Alternatively, the leaf may hold a probability vector indicating the probability of the target attribute having a certain value. Instances are classified by navigating them from the root of the tree down to a leaf, according to the outcome of the tests along the path.

without PCA, criterion was set to 'gini' 'and maximum number of features were set to log2(it is explained in the previous section) and splitter which is the strategy used to choose the split at each node is set as “random”

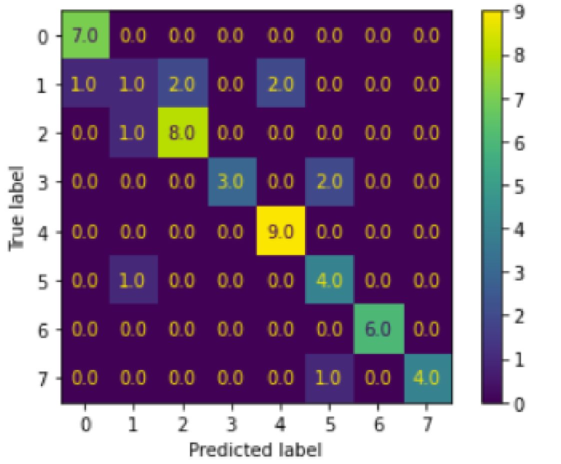
With PCA, criterion was set to 'entropy' 'and maximum number of features were set to auto and splitter is set as “best”.

Results without PCA



```
best result:

0.7696341463414634
Report:

              precision    recall  f1-score   support

           0       0.88      1.00      0.93         7
           1       0.33      0.17      0.22         6
           2       0.80      0.89      0.84         9
           3       1.00      0.60      0.75         5
           4       0.82      1.00      0.90         9
           5       0.57      0.80      0.67         5
           6       1.00      1.00      1.00         6
           7       1.00      0.80      0.89         5

    accuracy                           0.81        52
   macro avg       0.80      0.78      0.78        52
weighted avg       0.80      0.81      0.79        52
```

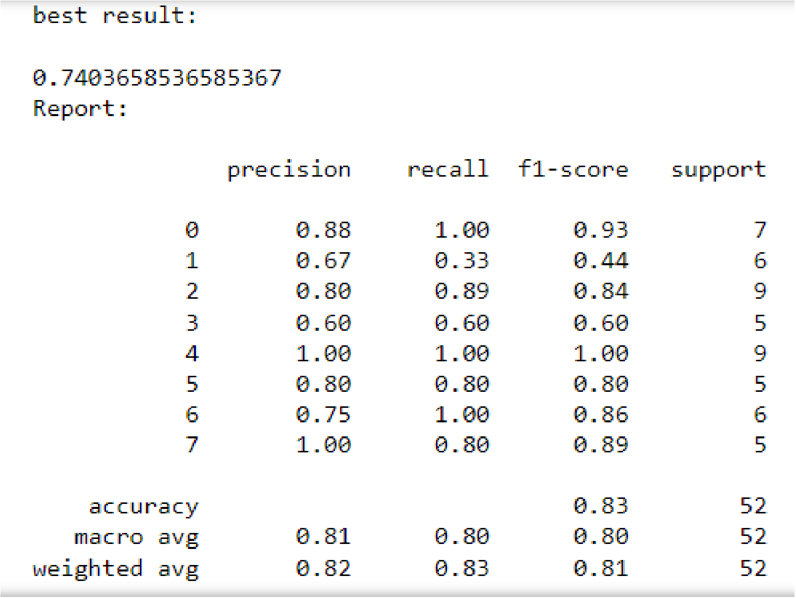
Figure 15



	0	1	2	3	4	5	6	7
0	7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	1.0	1.0	2.0	0.0	2.0	0.0	0.0	0.0
2	0.0	1.0	8.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	3.0	0.0	2.0	0.0	0.0
4	0.0	0.0	0.0	0.0	9.0	0.0	0.0	0.0
5	0.0	1.0	0.0	0.0	0.0	4.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	6.0	0.0
7	0.0	0.0	0.0	0.0	0.0	1.0	0.0	4.0

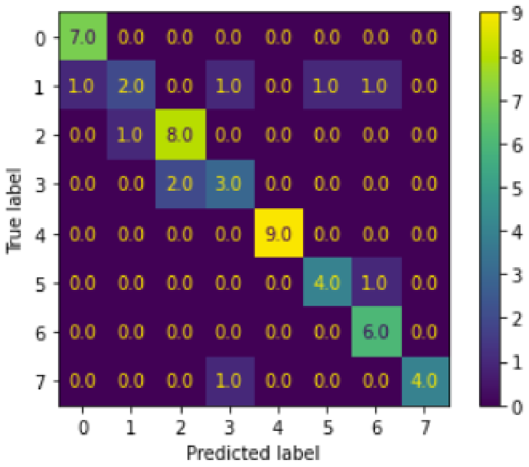
Figure 16

Results with PCA



```
best result:

0.7403658536585367
Report:

              precision    recall  f1-score   support

           0       0.88      1.00      0.93         7
           1       0.67      0.33      0.44         6
           2       0.80      0.89      0.84         9
           3       0.60      0.60      0.60         5
           4       1.00      1.00      1.00         9
           5       0.80      0.80      0.80         5
           6       0.75      1.00      0.86         6
           7       1.00      0.80      0.89         5

    accuracy                           0.83        52
   macro avg       0.81      0.80      0.80        52
weighted avg       0.82      0.83      0.81        52
```

Figure 17



	0	1	2	3	4	5	6	7
0	7.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	1.0	2.0	0.0	1.0	0.0	1.0	1.0	0.0
2	0.0	1.0	8.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	2.0	3.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	9.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	4.0	1.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	6.0	0.0
7	0.0	0.0	0.0	1.0	0.0	0.0	0.0	4.0

Figure 18

Model Review and evaluation

As seen in the previous sections of this paper, we can see that the best results were obtained using the Random Forest classifier and Decision Trees classifier since it’s a multiclass classification task but since in our case, we need interpretability, decision trees classifier is a better option than Random Forest classifier. We need interpretability because for example we need to know the reason if a student is having low grades. Interpretability is way more difficult in Random Forest classifiers but not in Decision Trees.

It’s important to mention that PCA was used in all the models and the outputs were compared with output of these models without applying PCA.

PCA does not improve the F1 score of the models and in the case of SVM it decreases the F1 score. Table 1 demonstrates the output of the models with and without applying PCA

Model Name	F1 Score	PCA
SVM	0.70	No
SVM	0.48	Yes
Random Forests	0.76	No
Random Forests	0.74	Yes
Decision Trees	0.76	No
Decision Trees	0.74	Yes

Table 1

Summary

In this section, we give a summary of the procedure and methods used in this paper.

After importing the dataset and cleansing it, standard Scaler has been used, then outliers have been removed and after that, random oversampling has been applied on the data. Then PCA method has been used.

After all, different models have been tested using the dataset both with and without PCA.

Citations

1- Yılmaz, N., Sekeroglu, B. (2020). Student Performance Classification Using Artificial Intelligence Techniques. In: Aliev, R., Kacprzyk, J., Pedrycz, W., Jamshidi, M., Babanli, M., Sadikoglu, F. (eds) 10th International Conference on Theory and Application of Soft Computing, Computing with Words and Perceptions - ICSCCW-2019. ICSCCW 2019. Advances in Intelligent Systems and Computing, vol 1095. Springer, Cham. https://doi.org/10.1007/978-3-030-35249-3_76

2- Goldberger, Arthur S. (1991). "Multicollinearity". A Course in Econometrics. Cambridge: Harvard University Press. pp. 245–53. ISBN 9780674175440.

3- Belhadi, Asma, et al. "Machine learning for identifying group trajectory outliers." ACM Transactions on Management Information Systems (TMIS) 12.2 (2021): 1-25.

4- Rokach, L., Maimon, O. (2005). Decision Trees. In: Maimon, O., Rokach, L. (eds) Data Mining and Knowledge Discovery Handbook. Springer, Boston, MA. https://doi.org/10.1007/0-387-25465-X_9