Periodic Performance Analysis to Predict Student Success Rates

by

Nurettin Selcuk SENOL

Submitted in Partial Fulfillment of the Requirements

for the Degree of

Master of Computing and Information Systems

YOUNGSTOWN STATE UNIVERSITY
May, 2020

Periodic Performance Analysis to Predict Student Success Rates

Nurettin Selcuk SENOL

Signature:

Nurettin Selcuk SENOL, Student Date

Approvals:

Abdu Arslanyilmaz, Ph.D., Thesis Advisor Date

Alina Lazar, Ph.D., Committee Member Date

Yong Zhang, Ph.D., Committee Member Date

Dr. Salvatore A. Sanders, Dean of Graduate Studies Date

ACKNOWLEDGMENT

I would like to express my deepest thanks to, Coskun Bayrak who recommended the main idea of this research and facilitate the needs for this study as well as his advising through the whole phases of the study. My gratefulness also goes to my committee members (Dr. Alina Lazar, Dr. Yong Zhang, Dr. Abdu Arslanyilmaz) for their invaluable experience and knowledge.

I would like to thank my family, my mother Gulsen, my father Abdil and my brother Serhat for their support and love during this study, and I would like to thank my friends for accepting nothing less than excellence from me.

Finally, I would like to dedicate this work to Dr.Bayrak who helped and supported me during my studies at Youngstown State University.

Abstract

There are many students who have difficulties in their higher education. There may be several reasons why students' performances may not be as intended. The system showcased in this study will enable students' performances be monitored, and it will be very beneficial for both academic staff and students who can be intervened promptly. The monitoring system tracks student's progress. Once an advisor notices a student's performance is beginning to deteriorate, they can step in and aid. Doing this helps prevent negative consequences occurring like failing a semester or dropping out of school.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Dedication

To Coskun Bayrak

Rest in Peace...

Chapter One

INTRODUCTION

A significant increase has been observed in the number of students in higher education with many new universities. According to research, there are some 5,300 colleges and universities in the United States [1]. Another study shows that in 2017, there were 14.56 million students in the U.S. enrolled in public colleges and 5.1 million students enrolled in private colleges [2]. This data is expected to increase to 14.98 million and 5.33 million, respectively, by the year 2028, according to Erin Duffin [2]. As seen, there is a substantial increase in the number of students and universities. This increase has the potential to cause some problems when performance is closely monitored.

Monitoring the performances of students who continue their higher education is essential for a number of reasons such as grade rate, graduation, enrollment, retention etc. Because the performance of the students shed light on their future performance, poor performance may herald future worse performances. There are many reasons why a student's initial performance maybe bad. These reasons may be caused by environmental or per-sonal factors. It can be clearly said that different environmental factors influencing during the early stages affect the performance of the student. For example, for higher education, students can choose from different places of residence, and leaving one's family requires an adaptation period which can negatively affect their academics. Due to this transitional period, it is not possible to fully understand the performance of the student in the first term. However, the performance(s) after the first semester, play a vital role in the formation of the future academic success of the student.

Different performances may be observed between periods, but obvious differences may raise questions about whether the student is in the right department. About 80 percent of students in the United States end up changing their major at least once, according to the National Center for Education Statistics [2]. On average, college students change their major at least three times over the course of their college career [2]. On the other hand, students whose performances are not followed may have lower grade scores than students who are followed. For these reasons, monitoring student performance is essential in both academia and industry.

1.1 Motivation

Most students make many important decisions before they start their academic life, and these decisions not only determine the future of students but also make the most of their lives. For this reason, the choice of profession is important when progressing to higher education. In addition, how their performance improves is going to be important for both individuals and academic staff. Therefore, a few predictions can be made using current students' data and personal performances of past students. These predictions may bring us one step closer to develop a monitoring system that solves the student's problem, helps their performance, and indicates how they can be better in their academia.

1.2 Research problem

With respect to the factors outlined above, it is necessary to maintain a monitoring process performance at least for three semesters. Therefore, we developed a performance monitoring system that uses data collected from students and provides an assessment of student academic performance. Thus, we can adjust the development of students' performance. Also, with the collected data, not only the student's data, but also semesters will be analyzed, classified, or clustered to make meaningful information.

1.3 Objectives of the study

O1 There will be class registration part that collects student classes information.

O2 There will be detailed interface design that collects student detailed information about their family's size, their habits, incomes, city, and family qualifications.

O3 There will be semester evaluation part that contents statistical information about each semester by cohort type.

Chapter Two

LITERATURE REVIEW

The amounts of data stored in the educational database is growing quickly. This data represents different types of hidden information about the improvements of students' performances. Some studies focused on kind of situations/factors may affect academic performance [3]. However, the data used was not limited to the academic life of the students, it also included their daily life, father's education (elementary, graduate, or doctorate), food habits, where they live, and family size [3].

There are many algorithms and ways such as clustering, classification, regression, Neural Networks, and Decision Trees for analysis. These algorithms are used for different types of data to discover knowledge from the data file. The Bayesian classification is proposed as a powerful algorithm technique in [3]. An advantage of the Naive Bayes classifier is that it requires a small training data to approximate the parameters (means and variances of the variables) necessary for classification.

Researchers in [4] investigated monitoring student progress using virtual appliances. A significant factor toward the strength of improving performance is an instructor's ability to monitor overall student progress and potentially act based on the observed events. An instructor monitoring all events taking place in student performance would have an advantage in improving the overall performance of students.

The main scope of concentration in [5] shows that, unfortunately, data mining algorithms work best with large data sets, while student data presented to higher education institutions, associated to courses, are limited and fall into the category of small data sets. Therefore, the study focuses on data mining for small student data sets and aims to answer the above research questions by comparing two different data mining tools: building student success rate data mining model using the MS Excel tool and building student data mining model with Weka.

Educational Data Mining (EDM) and Knowledge Discovery in Database (KDD) processes were used in [6]. The data set was obtained from a students' database used in one of the educational institutions. The EDM and KDD processes applied the data from 2005 to 2010, the initial size of data is 1,548 records. Decision Tree method is used on the student data to predict the students' performance successfully. In this way, the study helps the specific students improve their performances and identify those students who need more attention. Figure 2.1 shows the steps involved in KDD process. This process is an iterative mechanism where evaluation measures can be enhanced, data mining can be refined, and new data can be integrated and transformed to get different and more proper results.

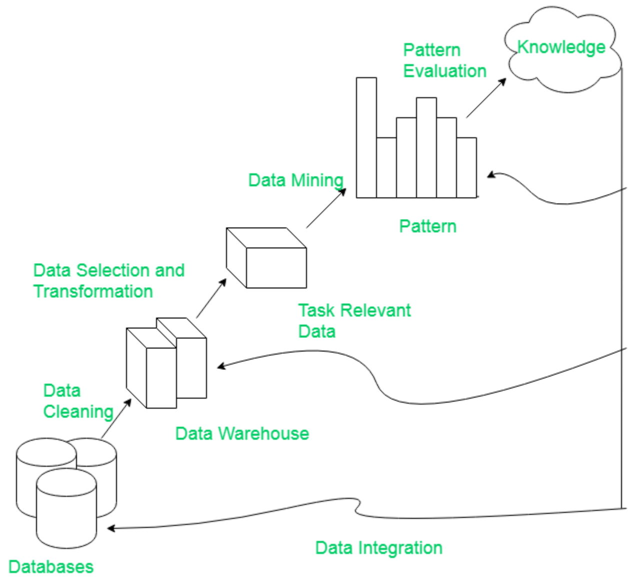


Figure 2.1 Knowledge Discovery Database (KDD) Process [6]

Artificial Neural Network (ANN), Decision Tree and linear regression methods have been compared in [7]. Research in examining students' performance has been done in [8], [9], and [10] using statistical analysis. This study examines these methods using SAS Enterprise miner in predicting the final cumulative grade point average (CGPA) of the students upon graduation. 206 students' data were obtained for research. Considering the above information, ANN is the best model in predicting the final CGPA of the students upon graduation.

Authors in [11] discuss predicting students' performance using data mining techniques. This study provides common attributes and methods that are used in predicting students' performance. Under the classification techniques, Neural Network and Decision Tree, are two methods highly used for predicting student performance. K-means clustering is a technique of vector quantization, originally from signal processing that is also famous for cluster analysis in data mining. The term 'k-means' was first used by James MacQueen in 1967 [12]. The basic algorithm was first proposed by Stuart Lloyd of Bell Labs in 1957 as a method for pulse-code modulation, but it was not brought out as an article until 1982 [13]. The most popular algorithm is the standard algorithm of k-mean. It is also known as Lloyd algorithm, particularly in the computer science community. Also, it is sometimes referred to as a 'native k-means' because there are better choices available [14]. Clustering is a grouping of data that has similar properties/features in a data set. Within the same cluster, similarities should be high, and similarities between clusters should be low. There is Unsupervised Learning, so no prior information is given. K-Means and Hierarchical Segmentation are commonly used clustering algorithms. These algorithms are frequently used in areas such as customer segmentation, student segmentation, student performance, market segmentation, and computer vision [15]. Vance Faber discusses in [16], a typical example of clustering is the consolidation of a set of student's academic test results, expressed as a percentage, into five clusters, one for each letter grade represents A, B, C, D, and F (see Figure 2.2). Academic test scores are the data points, and each individual cluster is a reference point of the average of the test scores in that cluster. Figure 2.2 illustrates partitioning of 20 test scores and their clusters. Letter grades can be thought of as symbolic substitutions for numerical reference points.

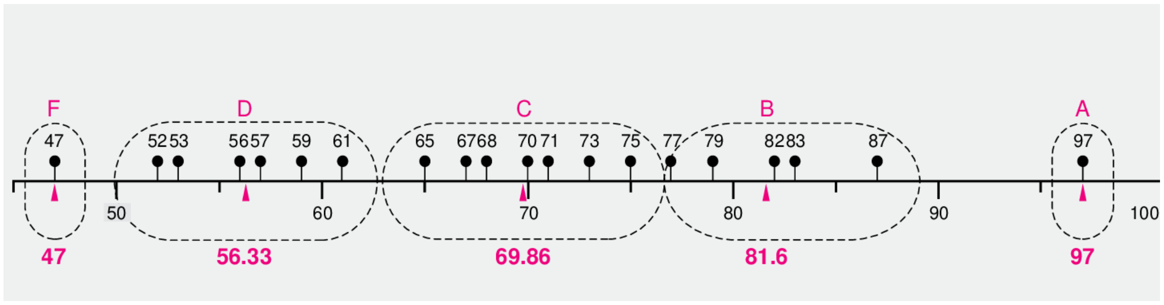


Figure 2.2 Clustering Test Scores, Total 20 test scores into non-over-lapping clusters (dashed lines), corresponding to 5 letter grades [16]

Sinan Aydin provides a definition of clustering algorithms as clustering divides data into meaningful and/or useful clusters (groups) [17]. If the goal is to create meaningful clusters, then the clusters should reflect the natural structure of the data. Clustering is a useful algorithm for having similar numerical values, such as summarizing data in [17]. For example, if a national chain of stores wants to create catalogs targeting various demographic groups based on characteristics such as the physical characteristics of their customers (age, height, weight, etc.), in- come, and residence. To this end, the company can create clusters of potential customers based on the values of the specified features to identify target customers of various catalogs and help create catalogs that address new and more specific groups [17].

Universities are higher education insti-tutions established to educate students. Students can continue their academic careers if they wish, or they can be part of the industry. Thus, monitoring students' performance is critical to ensure that the supply chain is fulfilled. Many studies on student performances are still available. In fact, data mining methods are not used only within the educational sciences as mentioned above. It is possible to see examples of this in different industries, including vehicle performance anal-ysis, customer satisfaction, and work-employment analysis. In light of the researches published above, the proposed monitoring system that uses these methods and provides more meaningful in-formation will be more beneficial for both advisors and students. Some of the studies could show similarities. However, depending on the variety of data available and the method used the results may be different.

Chapter Three

DATA COLLECTION, PROCESSING AND ANALYSIS

In this study, the data is collected at Youngstown State University with the consent of the subjects. The first set of recordings started from spring 2008 through fall 2017 and was calculated on 03/23/2018. Student Cumulative GPA data file was prepared by Steve Taraszewski at the office of Institutional Research & Analytics at Youngstown State University. The attribute names and their descriptions are provided in Table 3.1.

3.1 Data Cleansing

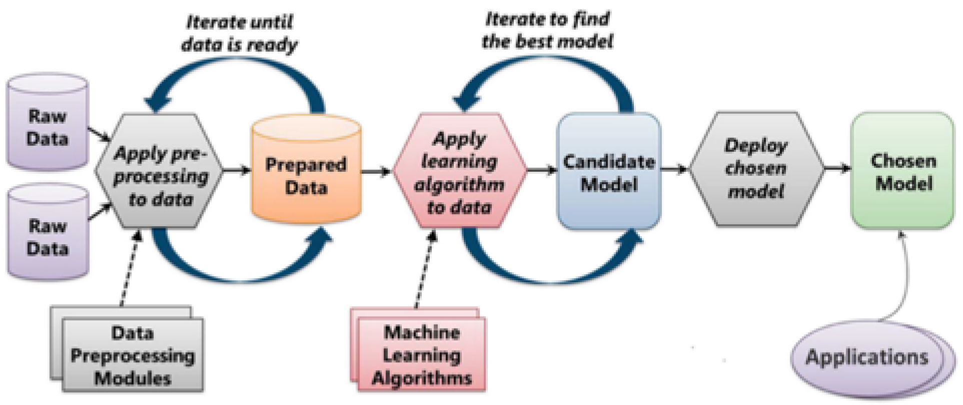
Data cleaning is done to detect corrupt or erroneous recordings from a database or any data table from a set of records and the whole of the smoothing operations. Identifying incorrect, incomplete, or irrelevant parts of the data, followed by a tool change or deletion are generally applied. Data cleaning operations can be implemented interactively with data cleaning tools. In this study, numerical values are used in all but two attributes.

Attribute	Description
COHORT	First-time (FU) or new transfer (XFER) undergraduate students that were in CSIS majors during their first semester at YSU
PERSON_UID	Personal User Identification Number
BANNERID	Youngstown State University Banner Identification the number starts with Y00
200830_Cum_GPA 200840_Cum_GPA 200920_Cum_GPA 200930_Cum_GPA 200940_Cum_GPA through 201620_Cum_GPA 201630_Cum_GPA 201640_Cum_GPA 201420_Cum_GPA 201730_Cum_GPA 201740_Cum_GPA	First 4 digit represents year after 2 digit represents of the semesters such as 30 is Spring, 40 is Fall and 20 is Summer

Table 3.1 Students Data File Attributes and Descriptions

3.2 Data Preprocessing

If there is prior data to process the machine learning model before, the results may not work properly, and the results may be discussed. An example can be given about this. The machine algorithm must be implemented as a preliminary preparation of the test file. Figure 3.1 represents the machine learning process. The data file needs to be prepared to be able to apply a classification or clustering algorithms. In order to have the data file ready for the algorithm, the data preprocessing steps must be performed repeatedly.



Iterate until data is ready
Iterate to find the best model
Raw Data
Raw Data
Apply pre-processing to data
Prepared Data
Apply learning algorithm to data
Candidate Model
Deploy chosen model
Chosen Model
Data Preprocessing Modules
Machine Learning Algorithms
Applications

Figure 3.1 Machine Learning Process [34]

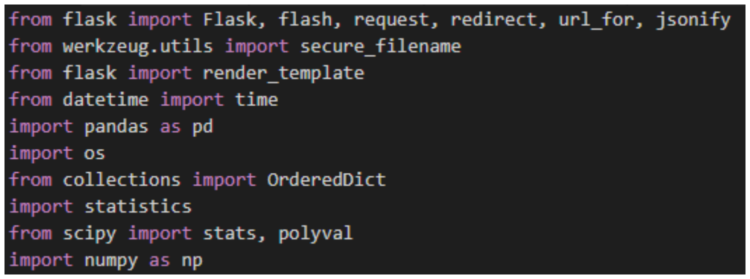
3.2.1 Step 1: Import Libraries

Library is a tool thatis used to make a specific job. User needs to give input, then library will do the job for the user and return the results. Figure 3.2 shows the part of the libraries used in this study.

- NumPy is the fundamental package for scientific computing with Python. It contains, among other things such as strong N-dimensional array objects, functional linear algebra, transform, and random number capabilities.

- Pandas is for data manipulation and data analysis. Pandas is an open-source library providing high performance, easy to use, data analysis tools for the Python programming language.
- Statistics is a Python module that allows users to explore data, estimate statistical models, and mathematical tests.
- SciPy is a Python-based ecosystem of open-source software for mathematics, science, and engineering.

Figure 3.2 shows a part of libraries that used in this study such as pandas, numpy, statistics etc.

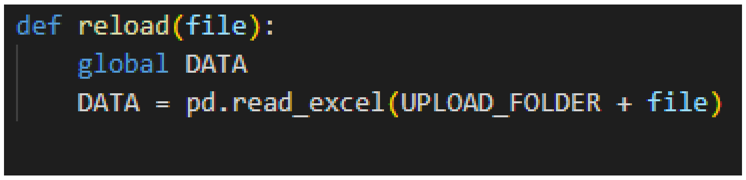


```
from flask import Flask, flash, request, redirect, url_for, jsonify
from werkzeug.utils import secure_filename
from flask import render_template
from datetime import time
import pandas as pd
import os
from collections import OrderedDict
import statistics
from scipy import stats, polyval
import numpy as np
```

Figure 3.2 A part of libraries that used in this study

3.2.2 Step 2: Import the Data - Set

The data that will be analyzed, needs to be imported to check out the missing values. Figure 3.3 shows how to import the data file. Importing by type may vary such as .xlsx, .csv etc.

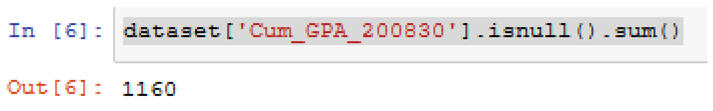


```
def reload(file):
    global DATA
    DATA = pd.read_excel(UPLOAD_FOLDER + file)
```

Figure 3.3 Importing Excel Data file to DATA frame.

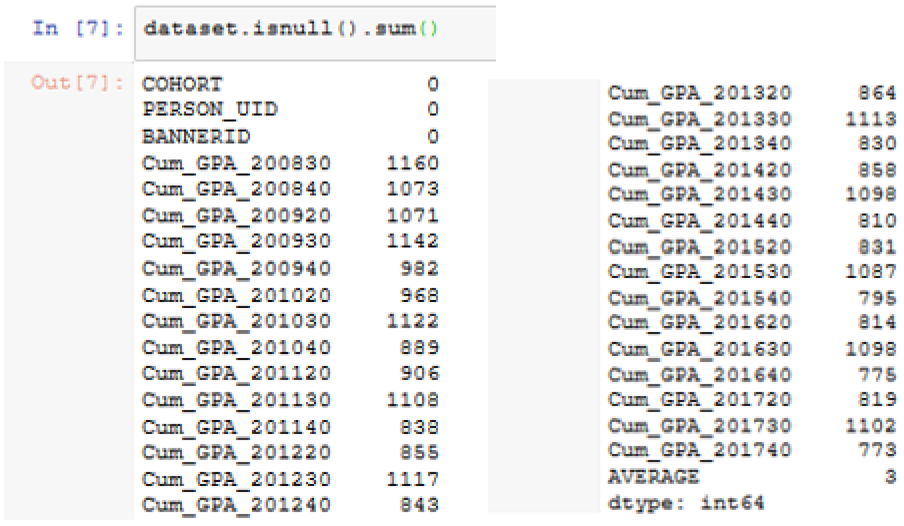
3.2.3 Step 3: Check out the Missing Values

The concept of missing values is essential to understand how to successfully manipulate the data. If the missing values are not appropriately handled by the scientists, there will be other problems in the next step. Figure 3.4 shows the total number of missing values of student's cumulative GPAs of 2008 spring. Figure 3.5 shows the total number of missing values of each attribute of the dataset. Handling with missing values for all attributes and dataset shape are shown Figure 3.6.



```
In [6]: dataset['Cum_GPA_200830'].isnull().sum()

Out[6]: 1160
```

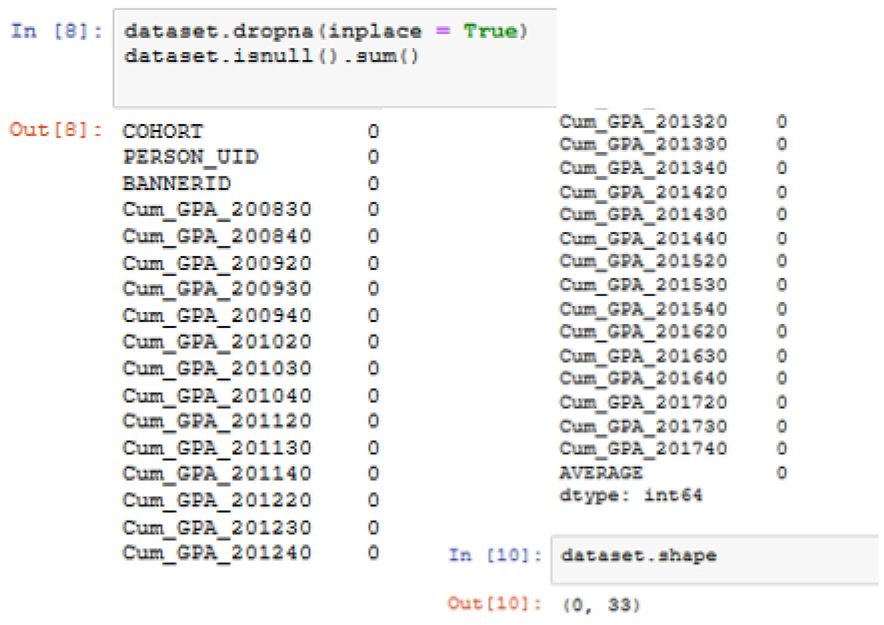
Figure 3.4 Total number of missing values of student's cumulative GPA's of 2008 Spring.



```
In [7]: dataset.isnull().sum()

Out[7]: COHORT               0
        PERSON_UID           0
        BANNERID             0
        Cum_GPA_200830    1160
        Cum_GPA_200840    1073
        Cum_GPA_200920    1071
        Cum_GPA_200930    1142
        Cum_GPA_200940     982
        Cum_GPA_201020     968
        Cum_GPA_201030    1122
        Cum_GPA_201040     889
        Cum_GPA_201120     906
        Cum_GPA_201130    1108
        Cum_GPA_201140     838
        Cum_GPA_201220     855
        Cum_GPA_201230    1117
        Cum_GPA_201240     843
        Cum_GPA_201320     864
        Cum_GPA_201330    1113
        Cum_GPA_201340     830
        Cum_GPA_201420     858
        Cum_GPA_201430    1098
        Cum_GPA_201440     810
        Cum_GPA_201520     831
        Cum_GPA_201530    1087
        Cum_GPA_201540     795
        Cum_GPA_201620     814
        Cum_GPA_201630    1098
        Cum_GPA_201640     775
        Cum_GPA_201720     819
        Cum_GPA_201730    1102
        Cum_GPA_201740     773
        AVERAGE              3
        dtype: int64
```

Figure 3.5 Total number of missing values of each attributes of the dataset

Abstracting the data will lead to loss of information, which will not give the expected results while prognosticating the output.



```
In [8]:  dataset.dropna(inplace = True)
         dataset.isnull().sum()

Out[8]:  COHORT            0        Cum_GPA_201320    0
         PERSON_UID        0        Cum_GPA_201330    0
         BANNERID          0        Cum_GPA_201340    0
         Cum_GPA_200830    0        Cum_GPA_201420    0
         Cum_GPA_200840    0        Cum_GPA_201430    0
         Cum_GPA_200920    0        Cum_GPA_201440    0
         Cum_GPA_200930    0        Cum_GPA_201520    0
         Cum_GPA_200940    0        Cum_GPA_201530    0
         Cum_GPA_201020    0        Cum_GPA_201540    0
         Cum_GPA_201030    0        Cum_GPA_201620    0
         Cum_GPA_201040    0        Cum_GPA_201630    0
         Cum_GPA_201120    0        Cum_GPA_201640    0
         Cum_GPA_201130    0        Cum_GPA_201720    0
         Cum_GPA_201140    0        Cum_GPA_201730    0
         Cum_GPA_201220    0        Cum_GPA_201740    0
         Cum_GPA_201230    0        AVERAGE           0
         Cum_GPA_201240    0        dtype: int64

In [10]:  dataset.shape

Out[10]:  (0, 33)
```

Figure 3.6 Handling with missing values for all attributes and dataset shape.

3.2.4 Step 4: Categorical Values

Machine learning models are based on mathematical equations. If the categorical data is kept in equations, it can be intuitively understood that the problem may arise because only numbers were desired in the equations. GPA columns include numeric values, but Cohort column will cause a problem, so the values needed to be converted into numerical values. If the Categorical variable needs to be converted into numerical data, LabelEncoder() class can be used from the preprocessing library.

label_encoder is used for helping for categorical data into numerical data. Cohort column will be transformed as numerical data, which are 0 and 1. Machine learning models are based on mathematical and statistical equations. Label encoder replaced text by numbers so that it can be included among the numbers in equations. Column values would include different types of strings so that numerical values could be 0, 1, 2, etc.

3.2.5 Step 5: Splitting the data-set into Training and Test Set

When a big amount of data is used, it can be divided as a test and train data. Generally, the data splits as into 70:30 ratio or 80:20 test and train data. What it means is that 70 percent of data is taken in the train, and 30 percent of data is taken for the test. The same rule applies for 80:20 ratio, 80 percent of the data is taken for the train, and 20 is taken for the test. Nevertheless, this kind of splitting can vary depending on the dataset shape and size. When classification and regression is done, usually set testing and training sets are used to help build and improve models. However, when clustering is done, occasionally there is a need for set testing and training sets depending on the situation because clustering also suffers from over-fitting problem.

Chapter Four

Introduction to Performance Monitoring System: A Solution for Success in Higher Education

In this chapter, the structures of the Performance Monitoring System and its components will be discussed. First, the interior design of each component will be explained. Then the mechanical design of the dispenser will be described in detail. Finally, student class and grade registration parts will be explained. Student Performance Monitoring System is a promising solution to improve student performance for both academia and industry. It also provides a secure communication channel between students and their departments. The importance of overall performance is determined by the individual performance. Although individual students are good, general student performance may differ from each other. According to the IDs of the students, their personal information (such as banner ID or cohort type etc.) and their academic performances can be followed on individual pages.

A typical student is expected to attend eight semesters (fall, spring, and summer) in undergraduate courses. Upon successful completion of the courses, students should graduate. During this period, the student's condition, semester by semester, is monitored through this study and may indicate that it is necessary to supervise as needed.

4.1 Technical Review

4.1.1 Introduction to Python

Python is an object-oriented, interpretive, modular and interactive, high-level programming language. Programming languages function as a bridge between machine logic and human logic. It is a programming language that is easier to learn and implement than many programming languages. The reason python was selected for this study is that it accommodates many libraries that will be used for our needs. The libraries to be used for analysis are more than other programming languages and can be implemented more efficiently. At the same time, we can easily visualize the analyzed data using the web frameworks (other details related to visualization, implementation, and frameworks can be seen more comprehensively in the following sections.). Conversely to C and Java, Python focuses on readability. Hence it attracts lots of developers and has a very substantial developer community, which in turn brings out greater support in the industry. Python also has a very large library, which eases lots of tasks. These libraries and framework details can be seen in oncoming sections. With Python, data analyzers can do most of the work with a small number of lines of code. Desktop, web applications, data analysis, and signal processing visualization applications can easily be written with Python [18], [19], and [20]. Lastly, Machine Learning is a complex computational technique that involves mathematics over thousands of rows and columns. All this becomes very simple and efficient with the help of the Python machine learning library which is scikit-learn.

4.1.2 Introduction to Flask

Flask is a very powerful and easy to learn microframework based on WSGI toolkit and jinja2 template engine for Python. Flask gives speed for different projects and prevents programmers from struggling with unnecessary configurations. It provides the flexible Python programming language and provides an effective template for web development. When programmers import into Python, it can be used to save time building for web applications. Because of the advantages it provides, flask microframework can be an alternative for this study. Especially with the help of the Jinja2 template, it is possible to manage the analyzed data successfully in the web application. It keeps the core simple but renewable [22].

4.1.3 Introduction to Jinja2

Jinja2 is a design engine for Python programming language. It is like the Django template engine, but uses expressions similar to Python, and the template files are put into a Sandbox. Jinja2 is a modern and user-friendly, flexible, fast, and secure templating language for Python that is modeled after Django templates. Jinja2 is more readable because of the syntax structure, which is the basis for visually distinguishing it from the HTML structure, is simpler. [23].

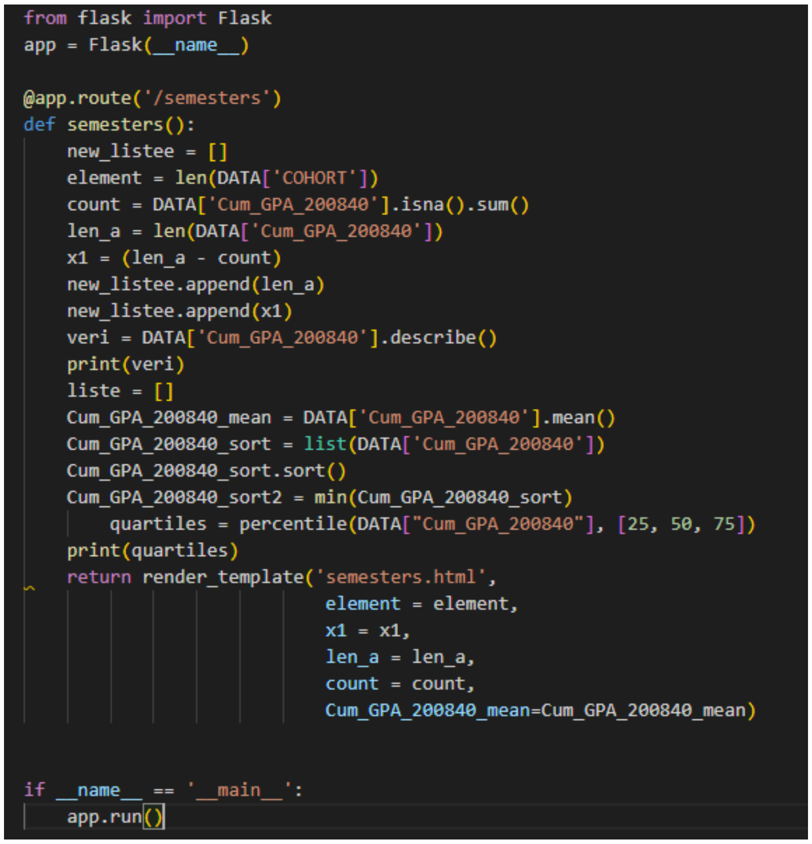
4.1.4 Introduction to Chart.js

Chart.js is an English-based software and designed by designer Nick Downie and presented free of charge with this javascript (Chart.js) library. Programmers can make beautiful and animated visual analysis from each other. In fact, there are many visualization libraries provided by javascript. It's comparable to Chartist and Google Charts. D3js, ChartJS, ThreeJS Echarts & Highcharts libraries can be given as examples. A suitable library can be selected to meet our needs. It supports six different chart types (including bars, lines, pies, bubble, radar, and scatter), and they are all reactive. In other words, we set up your chart once, and Chart.js will visualize our data effectively, and it's always legible [25] [26].

4.1.5 Implementation

4.1.5.1 General template structures of Flask

For example, Flask library needs to be imported with import function, then we are defining semesters function, and it returns the desired semester informations, sometimes data needs to be organized or sometimes the data is messy. We need to make some adjustments first, and after that, we return the result to .html file.



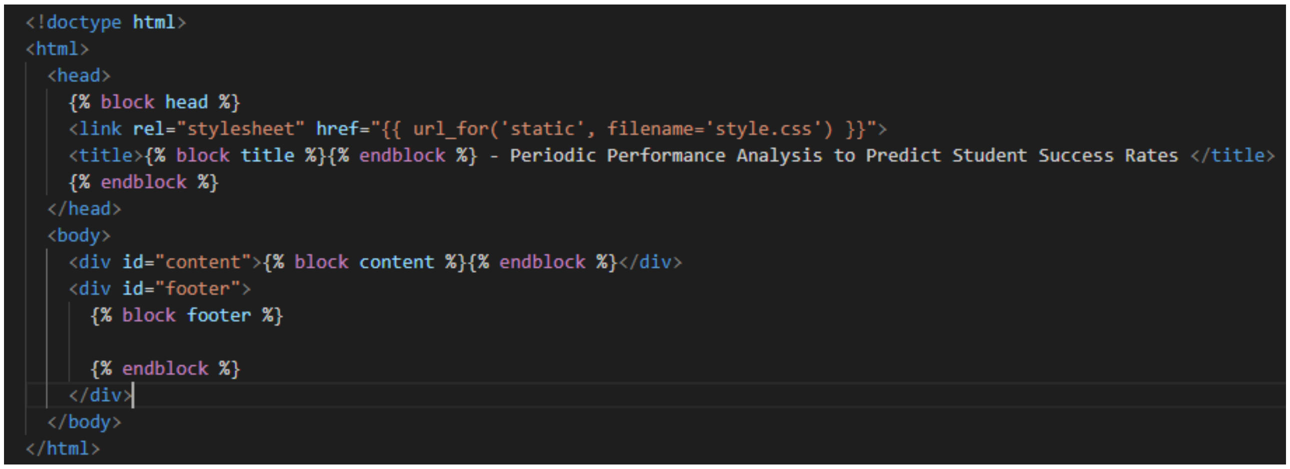
```python
from flask import Flask
app = Flask(__name__)

@app.route('/semesters')
def semesters():
    new_listee = []
    element = len(DATA['COHORT'])
    count = DATA['Cum_GPA_200840'].isna().sum()
    len_a = len(DATA['Cum_GPA_200840'])
    x1 = (len_a - count)
    new_listee.append(len_a)
    new_listee.append(x1)
    veri = DATA['Cum_GPA_200840'].describe()
    print(veri)
    liste = []
    Cum_GPA_200840_mean = DATA['Cum_GPA_200840'].mean()
    Cum_GPA_200840_sort = list(DATA['Cum_GPA_200840'])
    Cum_GPA_200840_sort.sort()
    Cum_GPA_200840_sort2 = min(Cum_GPA_200840_sort)
        quartiles = percentile(DATA["Cum_GPA_200840"], [25, 50, 75])
    print(quartiles)
    return render_template('semesters.html',
                            element = element,
                            x1 = x1,
                            len_a = len_a,
                            count = count,
                            Cum_GPA_200840_mean=Cum_GPA_200840_mean)


if __name__ == '__main__':
    app.run()
```

Figure 4.1 General Structure of Flask

Flask maps HTTP requests to Python functions. In this case, we've mapped one URL path ('/semesters') to one function, which is semesters. When we connect to the Flask server at http://127.0.0.1:5000/semester.html, Flask checks if there is a match between the path provided and a defined function. Flask runs the code in the function and displays the returned result in the browser. In this case, the returned result is HTML markup for a semesters page welcoming visitors to the site hosting the API [28]. app = Flask(__name__) if __name__ == '__name__': app.run () creates the Flask app object that contains data about application and methods (object functions) and tells the application the exact action it needs to do and app.run() is a method that runs the application server [28].

4.1.5.2 Template Inheritance

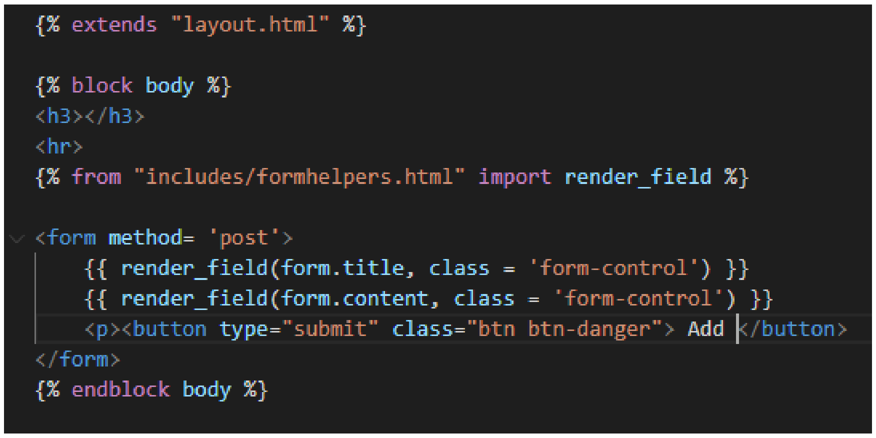
Flask permits software developers to build a base template that can be overridden by the child template. Flask uses % block %, and % endblock % tags define blocks that child templates can fill in it. All the block structure does is tell the template engine that the child template may override those portions of the template. On the other hand, % extends % section clarifies that this template can 'extend' another template [29]. Figure 4.2 shows that some blocks (like title, navbar or head) are "convenience blocks". Obviously they would not be necessary but are added to save typing effort.



```
<!doctype html>
<html>
  <head>
    {% block head %}
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    <title>{% block title %}{% endblock %} - Periodic Performance Analysis to Predict Student Success Rates </title>
    {% endblock %}
  </head>
  <body>
    <div id="content">{% block content %}{% endblock %}</div>
    <div id="footer">
      {% block footer %}

      {% endblock %}
    </div>
  </body>
</html>
```

Figure 4.2 % block % and % endblock % structures of Flask

When we execute the system, the system locates the first parent template. The extends tag must be the first tag in the template, and this type of templates can be named as child template (please see Figure 4.3). To render the contents of block defined in the parent template, % super() % can be used[30][31]. Figure 4.3 shows extend tag in Flask framework. It tells the template engine that this template “extends” another template. When the template system reclaims this template, first, it locates the parent template. The extends tag must be the first tag in the template.

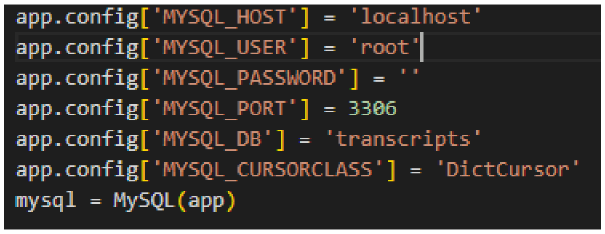


```
{% extends "layout.html" %}

{% block body %}
<h3></h3>
<hr>
{% from "includes/formhelpers.html" import render_field %}

<form method= 'post'>
    {{ render_field(form.title, class = 'form-control') }}
    {{ render_field(form.content, class = 'form-control') }}
    <p><button type="submit" class="btn btn-danger"> Add </button>
</form>
{% endblock body %}
```

Figure 4.3 % extends % structures of Flask.

4.1.5.3 Database Configuration

We need to install packages, as always, before we make some adjustments. After the required packages are installed, the MySQL configuration must be configured to establish a connection to the database. MYSQL_HOST represents the name of the host to connect to. MYSQL_USER is the user to authenticate with. MYSQL_PASSWORD is the password to authenticate with, for our condition, there will be no password, MYSQL_PORT is TCP (Transmission Control Protocol) port of the MySQL server. MYSQL_DB is the database to use, and MYSQL_CURSORCLASS is the cursor class will be set to the given string, and in our case, it is ‘DictCursor’. Figure 4.4 shows the necessary library and MySQL configuration [32].



```
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = ''
app.config['MYSQL_PORT'] = 3306
app.config['MYSQL_DB'] = 'transcripts'
app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
mysql = MySQL(app)
```

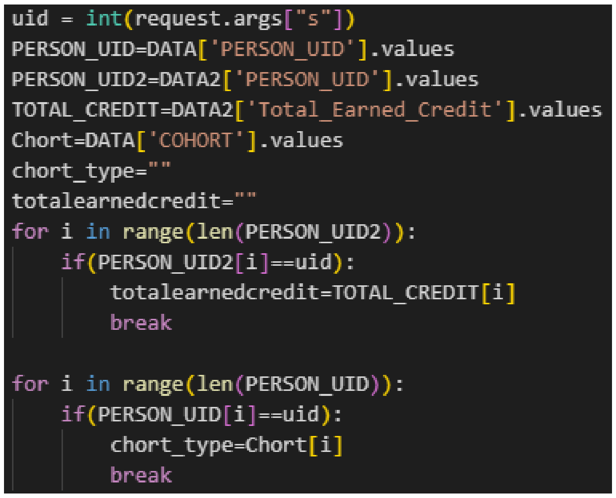
Figure 4.4 MySQL package and Database configuration

4.2 Introduction to Student Performance Monitoring System

To monitor the performance of each student, the first and the last semesters of student GPAs are important in terms of performance monitoring. Tracking the students' GPAs between the first and last semesters gives important information about the students' current status or their future performance. When performing data analysis, it is sometimes necessary to clear empty data or to perform pretreatment such as correction of incorrectly entered data or incompatible data. However, this study also makes sense for us in areas where there is no data or empty data. In places where there is no data, there is no GPA grade for the student. This means that the student either did not come to school (pass the period), dropped out (or changed schools) or graduated. For the student process, we have used all of the data in the data frame using the pandas library to be able to do the analysis more efficiently. We use the chart.js visualization tools using the jinja2 template. In fact, the process we do in the data frame is to check if there is any data in the attributes representing the semesters. It provides information about one of the conditions mentioned above may have occurred where there is no data.

4.2.1 General Distribution of Student GPA

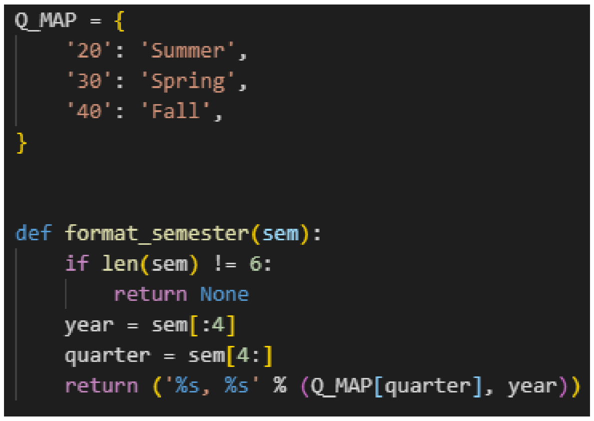
We need to get information about the students periodically to gain a better into who they are. So, the excel file has been imported into the system. The system kept the attributes in the Excel file separately in different numpy arrays. Thus, the unique IDs of the student can easily be traced to the information of other characteristics, such as their cohort type or total earned credits. Figure 4.5 shows that 'PERSON UID' and 'Total Earned Credit' are kept in two different numpy arrays so we will able to acquire student personal data with person ID.



```python
uid = int(request.args["s"])
PERSON_UID=DATA['PERSON_UID'].values
PERSON_UID2=DATA2['PERSON_UID'].values
TOTAL_CREDIT=DATA2['Total_Earned_Credit'].values
Chort=DATA['COHORT'].values
chort_type=""
totalearnedcredit=""
for i in range(len(PERSON_UID2)):
    if(PERSON_UID2[i]==uid):
        totalearnedcredit=TOTAL_CREDIT[i]
        break

for i in range(len(PERSON_UID)):
    if(PERSON_UID[i]==uid):
        chort_type=Chort[i]
        break
```

Figure 4.5 To access student personal information using numpy arrays

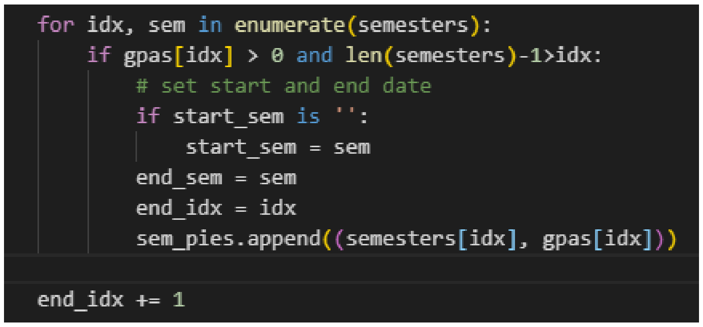
To use the GPAs of students, we must go through a series of processes. The numbers that come after "Cum GPA", as mentioned before, tells about the year and is the semester. Since we know that the year is 4-digits and the semester represents a 2-digit number, we should not consider non-6-digit numbers. Apart from this, it is possible to see the information of the attributes by repeating the data discovery steps. One of the easiest methods to do this is to use the len () function. To make the last two digits more meaningful by the user, the semester names of these numbers were arranged by the system. Figure 4.6 shows the steps of making the numerical data more meaningful for the user.



```python
Q_MAP = {
    '20': 'Summer',
    '30': 'Spring',
    '40': 'Fall',
}


def format_semester(sem):
    if len(sem) != 6:
        return None
    year = sem[:4]
    quarter = sem[4:]
    return ('%s, %s' % (Q_MAP[quarter], year))
```

Figure 4.6 Changing numerical data into meaningful data steps for semesters

The graph shown in Figure 4.8, it gives information about the semesters in which the student has a GPA and the semesters the student does not have a GPA. For this study, the semester range has been recorded from Spring 2008 to Fall 2017. For this reason, Figure 4.8 offers a broad perspective to make a better observation. It is of great importance for advisors in which semester the student's GPA starts and when the GPA ends. To observe this, we need to check the values in the file that are taken from the data. Semesters must be checked one by one to see if there is a numerical value different from 0. These steps can be easily done with the Python enumerate function. Figure 4.7 shows the enumerate function to show the start semester and end semester with its index.



```python
for idx, sem in enumerate(semesters):
    if gpas[idx] > 0 and len(semesters)-1>idx:
        # set start and end date
        if start_sem is '':
            start_sem = sem
        end_sem = sem
        end_idx = idx
        sem_pies.append((semesters[idx], gpas[idx]))

end_idx += 1
```

Figure 4.7 Start and End semesters

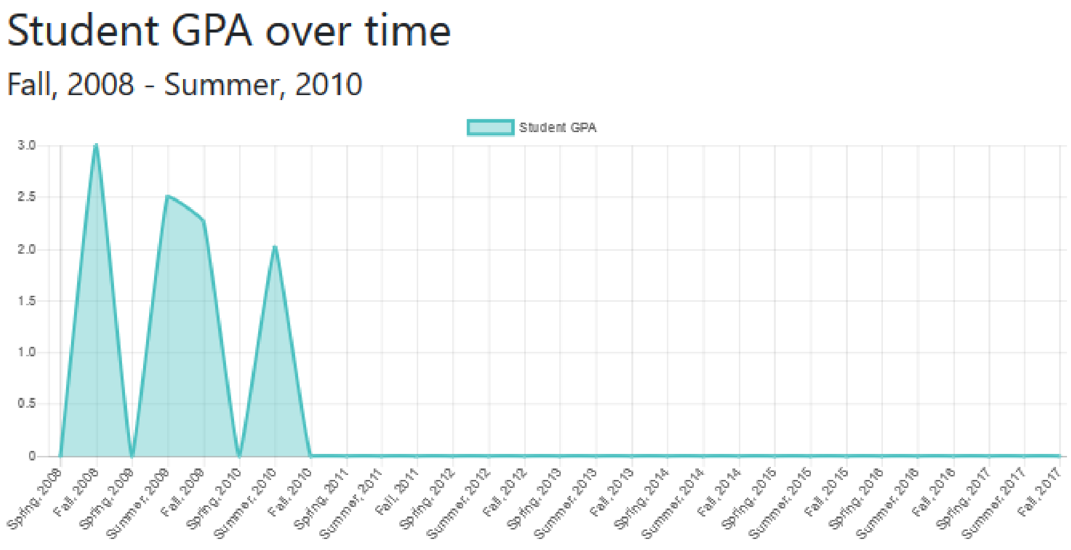


Figure 4.8 GPA distribution for a particular student

Figure 4.9 provides a better opportunity to see and compare semester-end scores with other semesters. Increases or decreases in GPA scores between semesters indicate some information about the future of the student. This information can help create some ideas about the student. For example, the student shown in Figure 4.9, GPA is decreasing between semesters. Intervention may be required when it is below a specific limit. Any assistance provided in this process will be of great benefit both in the academic life of the student and industry.

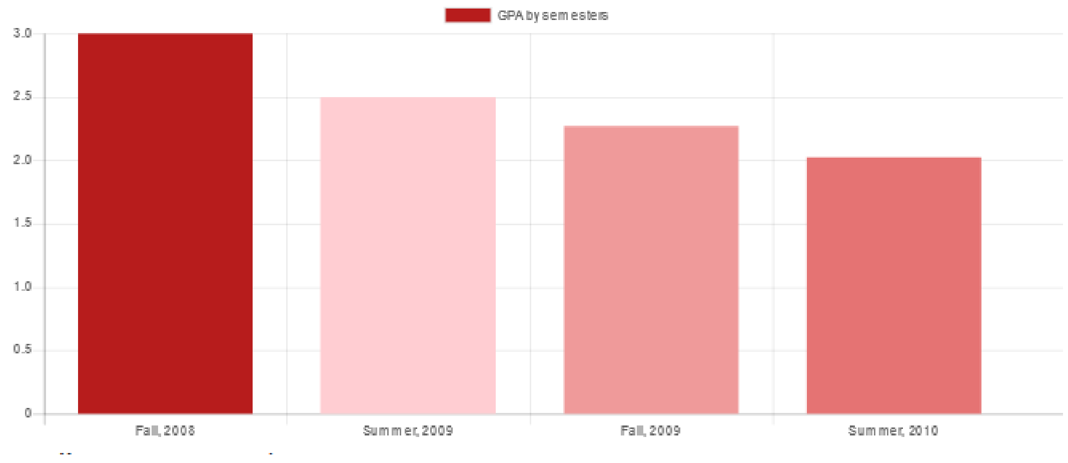


Figure 4.9 Student GPAs Chart

4.2.2 Prediction of Semesters

The number of semesters alone is not enough when the student graduates. According to [33], the semester hours of credit a student carries per term depends on the degree sought and on the curriculum followed. A minimum of 120 semester hours must be satisfactorily completed to earn a baccalaureate degree. Students expecting to complete a bachelor's degree in four years should average 16 credits per term. Students interested in taking 21 credit hours or more per term must seek approval from the Dean of their college. The number of credits completed by the student gives more robust information about the student's current and upcoming semesters. Students who have completed 120 credits or more and have successfully completed all courses are eligible to graduate. Students between 0-29 credits are classified as freshman, students between 30-59 credits are classified as sophomores, 60-89 credit classified as juniors, students with 90 or more credits are classified as seniors. Figure 4.10 shows that the estimated date of graduation of one of the random students. As of Fall 2014, the student had 94 completed credits and the student is currently classified as a senior. Even if he/she receives the maximum amount of credits he/she can get under normal conditions, he cannot fill the 120-credit limit. It is not foreseeable to graduate in Spring 2015, which is the next semester, unless there are additional requirements (overload permission by the Dean). With the overload allowed by the Dean, the GPA of the student is also important. When there is an increase in the number of courses for students without a good average, the completion of these courses is also a matter of discussion for the student. Having a good GPA can mean that the student can handle an overload. However, this does not apply to students who are below a certain GPA, because the problem with the courses that the student must take during the semester may indicate that he/she cannot successfully overcome the overloaded classes.

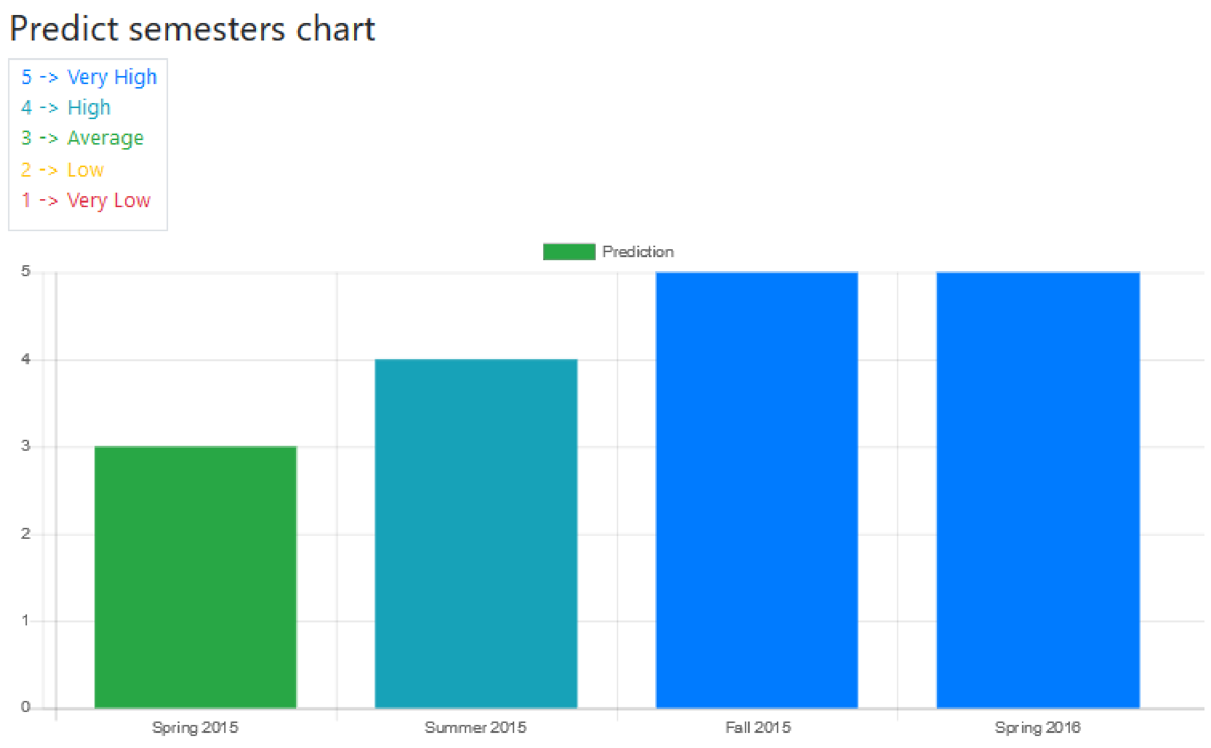


Figure 4.10 Prediction of Student Graduation

4.2.3 Student Assessment Section

Sometimes interpretation of graphics can be strengthened. A Student Assessment provides more convenience to the users where the information belonging to the student is processed, and the information extracted from the student's data is included. In this part, we are doing the operations done before, for example, data extraction. First, the data of the student is taken directly from the data file without processing and is shown in the Student Assessment section. This data was available at the beginning of this study. This data is kept directly in different variables, and then the variable names were put into '{{}}' to provide easy access from the Python script. This is one of the useful features of jinja2 child template structure. This can be done with the return render template structure in the app.py file. Previously showed are the student GPA grades on the graph, this was done on numpy.array, (please see Figure 4.9). The student's semester GPAs are important, and the average of these GPAs can provide insight into their future performance, this average has been used elsewhere in the Student Assessment section as well. The numerator of the equation represents the sum of the GPAs for the student and represents the number of semesters in the denominator. The result of this equation gives the student's average GPA. The formula 4.1 for the average is calculated by the following formula.

$$A = \frac{1}{n}\sum_{i=1}^{n} a_i = \frac{a_1 + a_2 + \cdots + a_n}{n} \tag{4.1}$$

After the average of the student's semesters, an assessment should be made about the student's condition. Youngstown State University uses the 0-4.0 GPA range. It is possible to have information about the student's situation by using the threshold points. If the 4.0 grade is the best, different evaluations can be made about the GPA grade of the student. Students who are below a certain limit can be considered critical. Figure 4.11 explains of determination of student status. If the number of semesters is bigger than three and the total GPA is bigger than 2.0, the student status is considered as not dangerous. However, a total GPA lower than 2.0, means there is a problem with the student, and it is considered as dangerous. If the number of semesters is lower than three it means, the semester number is not enough to tell something.

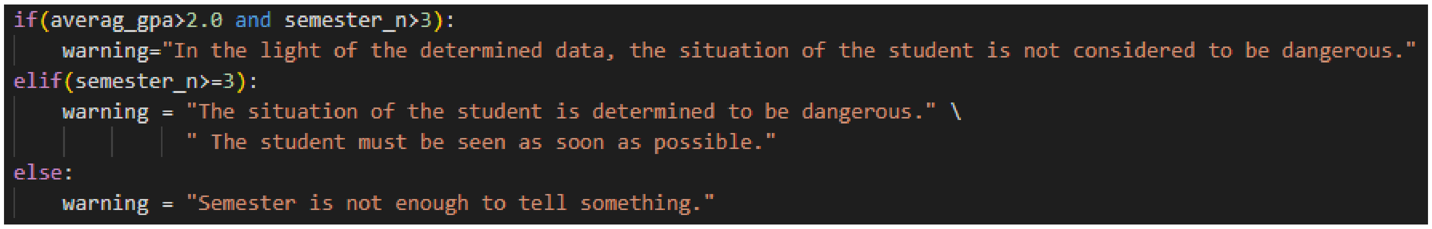
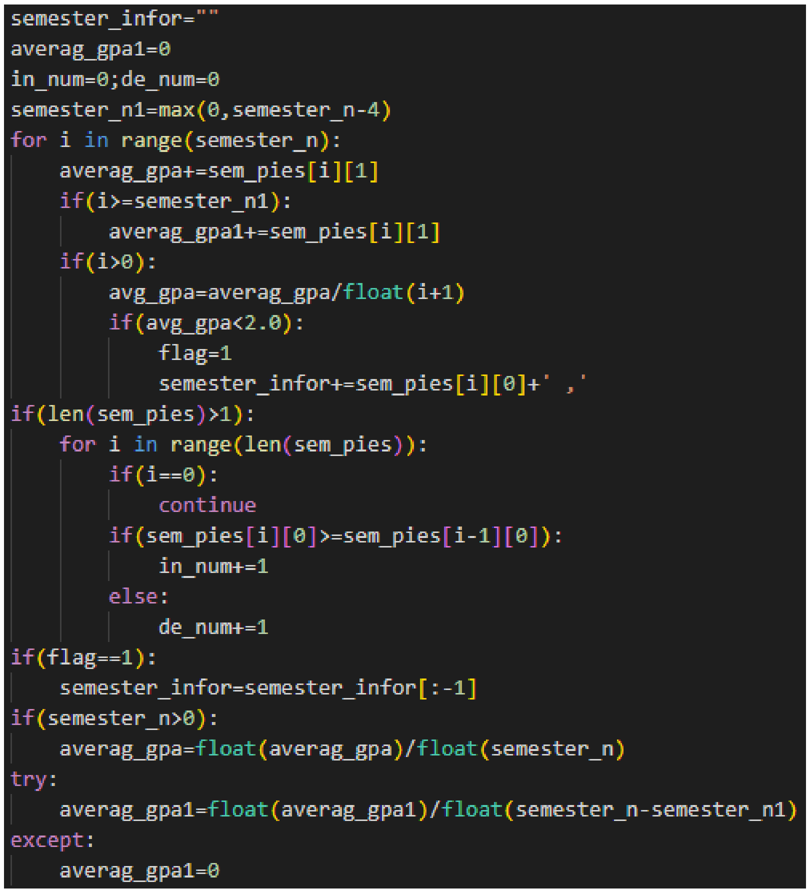

```
if(averag_gpa>2.0 and semester_n>3):
    warning="In the light of the determined data, the situation of the student is not considered to be dangerous."
elif(semester_n>=3):
    warning = "The situation of the student is determined to be dangerous." \
              " The student must be seen as soon as possible."
else:
    warning = "Semester is not enough to tell something."
```

Figure 4.11 Determination of Student Status

After a semester, it may be necessary to help the student if his/her situation is still in danger. It is a critical situation for students whose semester number is above three and who are above the average 2.0 GPA. However, the situation is critical for students below a 2.0 GPA and students whose semester number is three or more. In these cases, the semester number is significant. For students whose number of semesters is less than three, regardless of their term GPA.

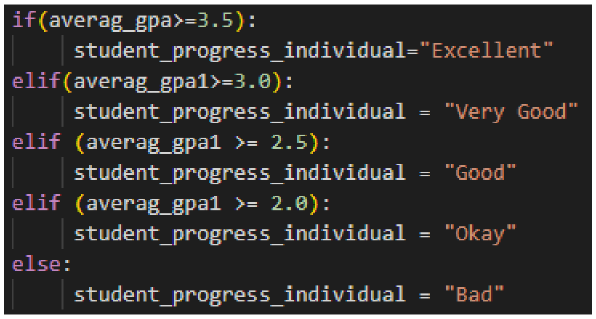
Any intervention may be required for students with poor status. For students who are in critical condition, the next step may be to identify the semesters that need to be assisted. In order to determine the semester, the student's current semester must be re-evaluated. Students whose semester number is more than three and whose average is below 2.0 should be intervened by the advisor. Situation monitoring is also vital for students. Because students whose condition is not followed in a healthy way may deteriorate again without improving their condition. The stu-dents who are below the required limit, or whose semester does not meet the requirements, will be in the semester that needs to be interfered. Critical semester of students whose status does not show any improvement continues to be displayed.



```python
semester_infor=""
averag_gpa1=0
in_num=0;de_num=0
semester_n1=max(0,semester_n-4)
for i in range(semester_n):
    averag_gpa+=sem_pies[i][1]
    if(i>=semester_n1):
        averag_gpa1+=sem_pies[i][1]
    if(i>0):
        avg_gpa=averag_gpa/float(i+1)
        if(avg_gpa<2.0):
            flag=1
            semester_infor+=sem_pies[i][0]+' ,'
if(len(sem_pies)>1):
    for i in range(len(sem_pies)):
        if(i==0):
            continue
        if(sem_pies[i][0]>=sem_pies[i-1][0]):
            in_num+=1
        else:
            de_num+=1
if(flag==1):
    semester_infor=semester_infor[:-1]
if(semester_n>0):
    averag_gpa=float(averag_gpa)/float(semester_n)
try:
    averag_gpa1=float(averag_gpa1)/float(semester_n-semester_n1)
except:
    averag_gpa1=0
```

Figure 4.12 Determination of Student semesters that need to be interfered

Student Assessment part also provides some important information on the student's follow-up processes between the semesters. The average of the student was calculated in the previous sections (Please see Equation 4.1). The individual performance of the student can be evaluated as the best with 4.0 GPA, and the student's condition is excellent in this regard. But when the individual performance of these students is within a certain range, the individual performance of the student is considered differently. Average GPA is considered excellent if the individual performances of the student are over 3.5. On the other hand, the individual performances of a student whose average is between 3.4 and 3.0 are considered very good. The performances of students between 2.9 and 2.5 are considered good. The performance of students between 2.4 and 2.0 is okay. The individual performances of all students below 2.0 are considered to be poor.

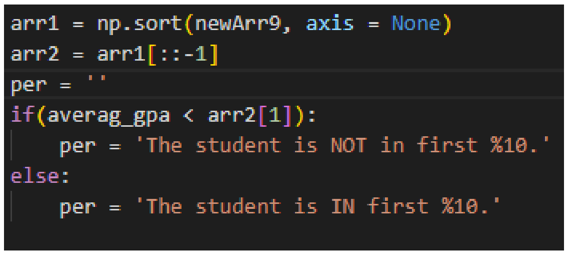


```
if(averag_gpa>=3.5):
    student_progress_individual="Excellent"
elif(averag_gpa1>=3.0):
    student_progress_individual = "Very Good"
elif (averag_gpa1 >= 2.5):
    student_progress_individual = "Good"
elif (averag_gpa1 >= 2.0):
    student_progress_individual = "Okay"
else:
    student_progress_individual = "Bad"
```

Figure 4.13 Individual Process Threshold limits

In addition to the individual performances of the students, information about the general performances of the students is also given in the Student Assessment section. The general data discovery steps in chapter 4 were followed to calculate the student's overall performance. The total credits earned by the student is kept in another excel file. We take the same steps as the student's school number or cohort type for student's total earned credit. The total credits earned by the student gives a lot of extra information about the student in the above sections.

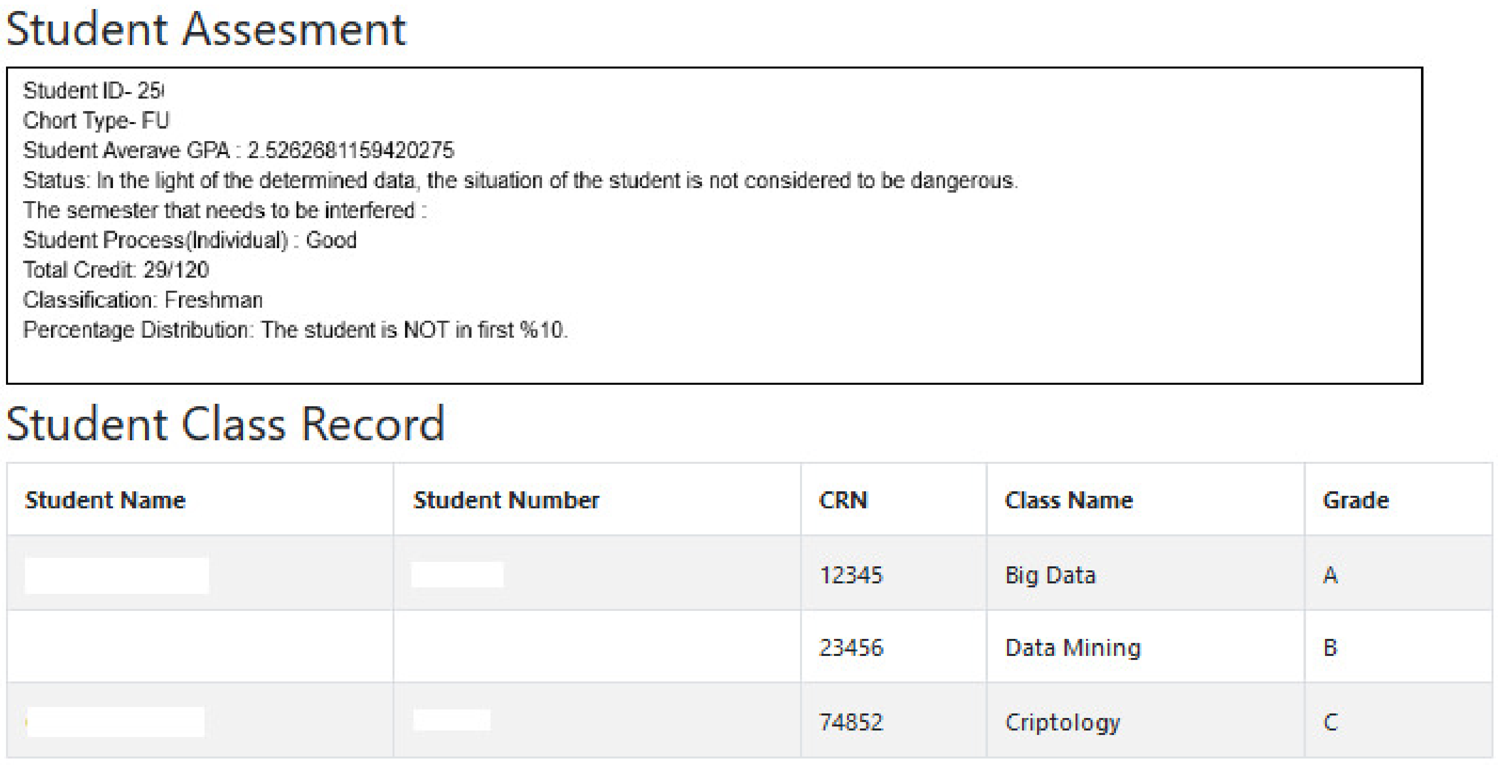
Another way of evaluating a student is a sign of success first 10% percent of GPAs in the total number of students. 1169 students were used for this study. When ranking the top 10% of students according to their GPAs, students with 3.75 GPAs are included in the first 10%. This can be considered as a criterion for success evaluation. To determine range, all the students were ranked according to their GPAs. Next, all the students in first 10% were gathered in an array. The first percentile is calculated as 4.0 to 3.75 slices. All students calculated within this range can also be considered successful.



```
arr1 = np.sort(newArr9, axis = None)
arr2 = arr1[::-1]
per = ''
if(averag_gpa < arr2[1]):
    per = 'The student is NOT in first %10.'
else:
    per = 'The student is IN first %10.'
```

Figure 4.14 Calculation of Percentage Distribution

In the light of the above calculations and definitions, Figure 4.15 shows an example of the Student Assessment section with the jinja2 template structure. Figure 4.15 provides an assessment of the status of one random student.



Student Assesment

Student ID- 25
Chort Type- FU
Student Averave GPA : 2.5262681159420275
Status: In the light of the determined data, the situation of the student is not considered to be dangerous.
The semester that needs to be interfered :
Student Process(Individual) : Good
Total Credit: 29/120
Classification: Freshman
Percentage Distribution: The student is NOT in first %10.

Student Class Record

Student Name	Student Number	CRN	Class Name	Grade
[redacted]	[redacted]	12345	Big Data	A
		23456	Data Mining	B
[redacted]	[redacted]	74852	Criptology	C

Figure 4.15 Student Assessment

4.2.4 Student Class Record

In order to make the assessment of the student in a healthy way, the student also needs the course information because it is important whether students are successful or not in addition to GPAs. Because most of the courses that make up the overall GPA are related to the field of study, but in the first and second year, there are a number of non-field courses that students must complete. There is no database or an excel file for the student's courses in this study, but a prototype has been developed. Thanks to a different interface, the information obtained from the student is shown on the student performance page under the Student Assessment section.

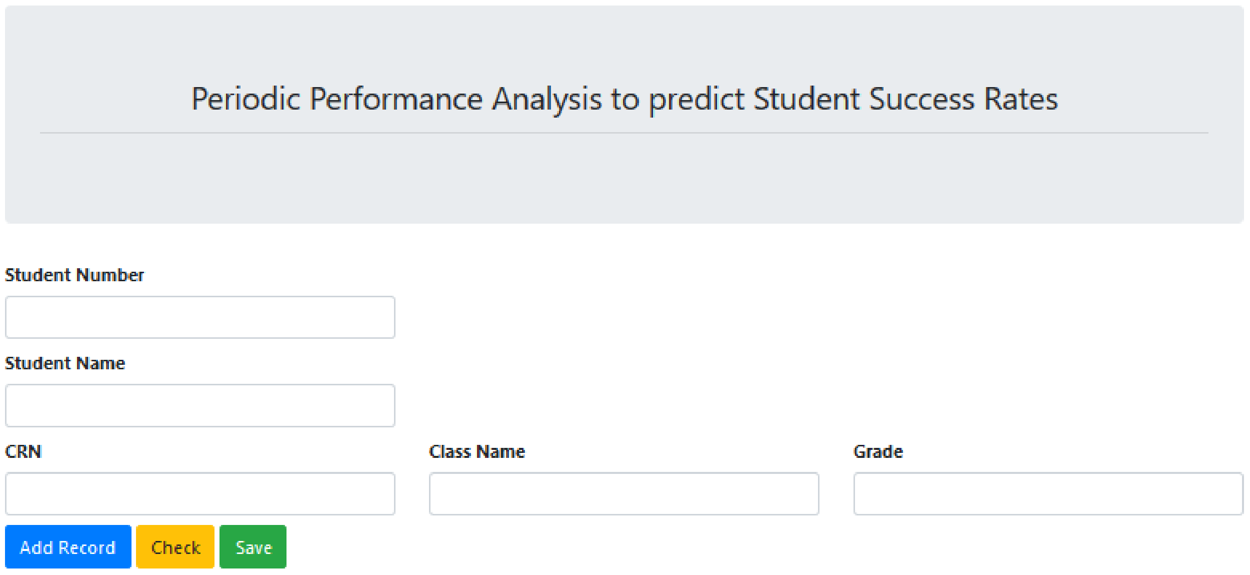


Figure 4.16 Student Class Record Adding Page

After the number of courses are entered manually, all the courses entered are seen by the student monitoring system. Thanks to the designed interface, students can check the courses they have entered as well.

4.3 General Report

In this section, the information belonging to the student and the data analyzed are shown as a report. Only data analysis can be considered as an incomplete study in terms of information acquisition. It has great importance in terms of gaining information by using some visualization tools in the analyzed data. Therefore, the data analyzed should be prioritized to obtain more understandable information by the user using various visualization techniques such as pie chart, bar chart or line chart. Thanks to this acquired knowledge, students can be given a different and effective consultation. One of the first steps to be taken is to determine which kind of student they are. The student can be a first time or a transfer student. Figure 4.17 shows how many students are in this study, and at the same time, it shows the percentages of the student types in total.

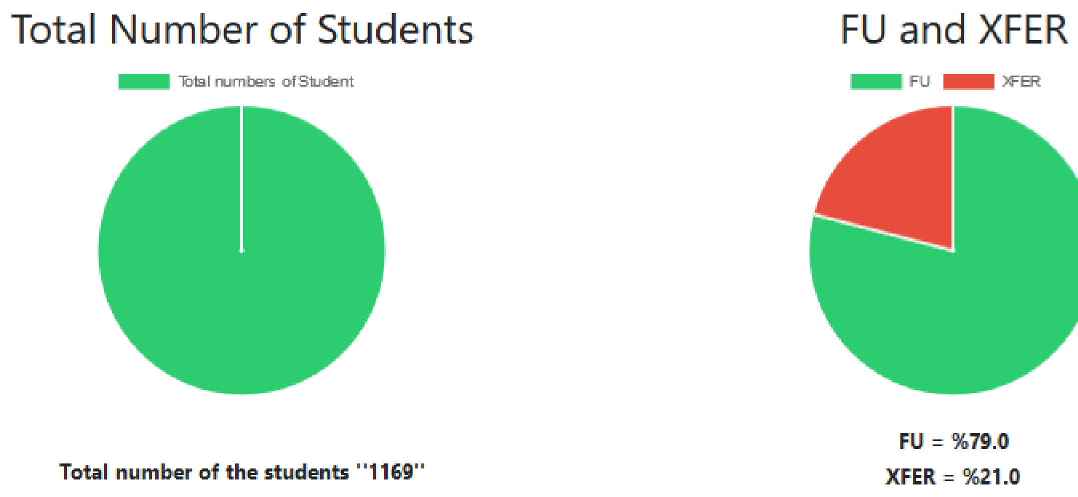
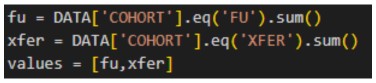


Figure 4.17 Total number of Student and Student Type

To find out the number of students in the study, the len() function of the column name of the student number can easily be used. Using the render template structure, the total number of students can be accessed. Likewise, this process can be done to find the number of student types. Figure 3.19 shows how the cohort types of undergraduate students are calculated. Here two different functions are used. The value is checked with the .eq function in DATA data frame and with .sum function, the total number of cohort student types can be found.



```
fu = DATA['COHORT'].eq('FU').sum()
xfer = DATA['COHORT'].eq('XFER').sum()
values = [fu,xfer]
```

Figure 4.18 Calculation of total number of student types

Observing successful students as well as students with problems is another alternative to assess student performance because some successful determinations of successful students can also be used for students who have issues. It would be more beneficial for both the advisor and the student to know what kind of difficulties they are focusing on and to identify an approach type according to it. The first four students we have obtained by sorting the students according to their performances are shown in Figure 4.19.

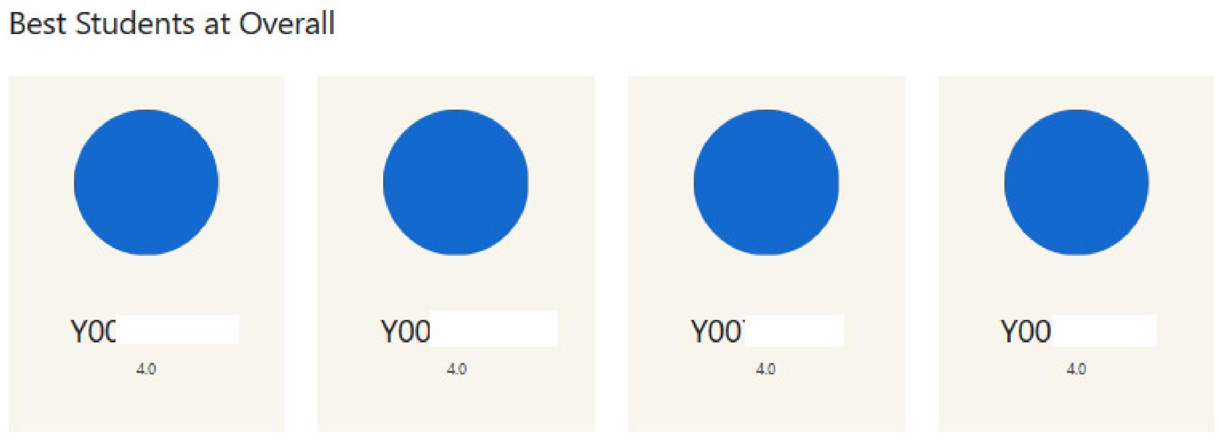


Figure 4.19 The best students at Overall

Knowing the variance in student GPA ranges provides a better assessment for target students. Student GPA distributions are shown in Figure 4.20. This demonstration is also adjusted for the student type. When the charts are examined carefully, it is possible to see that there are many students whose averages are below 2.0. If the target students are firmly determined, the assessment will be as successful.

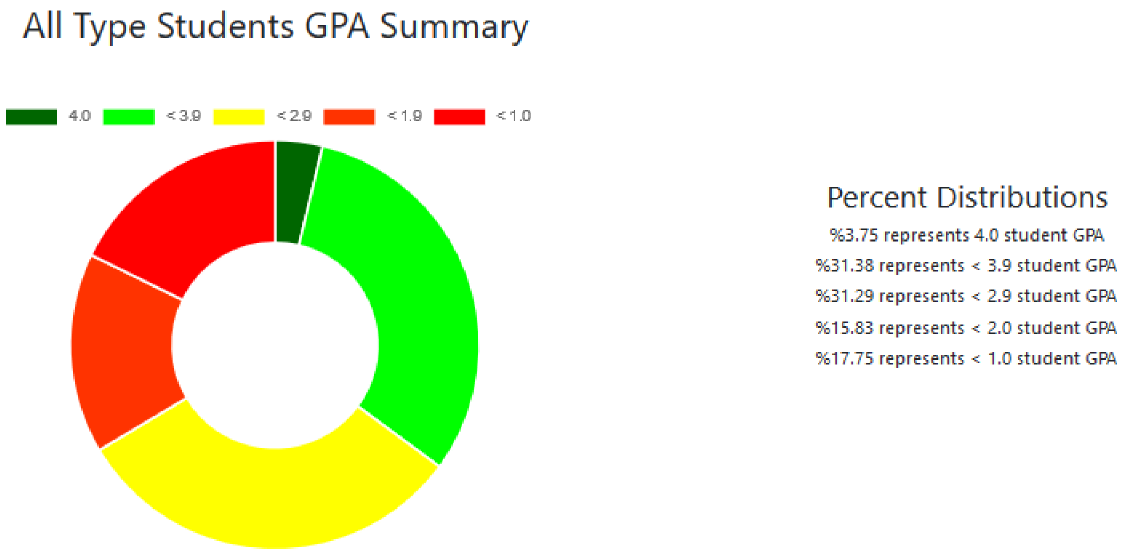
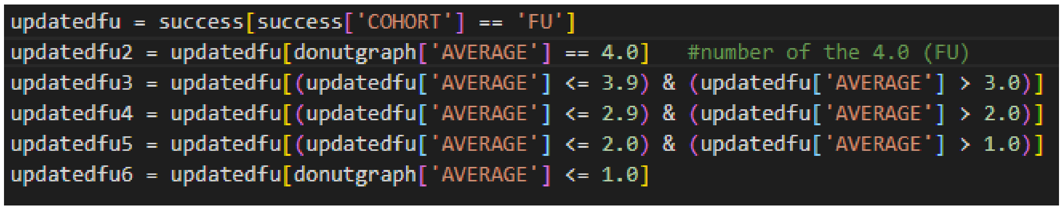


Figure 4.20 All types of Student Distribution

Data frame structures of the pandas library were used for calculations in the previous chapters. Here the same process is repeated. However, different types of data such as student type or average student GPA were used. For example, instead of the FU student type as shown in Figure 4.21, the XFER student type was input. This way the distribution of both student types can be seen.



```
updatedfu = success[success['COHORT'] == 'FU']
updatedfu2 = updatedfu[donutgraph['AVERAGE'] == 4.0]   #number of the 4.0 (FU)
updatedfu3 = updatedfu[(updatedfu['AVERAGE'] <= 3.9) & (updatedfu['AVERAGE'] > 3.0)]
updatedfu4 = updatedfu[(updatedfu['AVERAGE'] <= 2.9) & (updatedfu['AVERAGE'] > 2.0)]
updatedfu5 = updatedfu[(updatedfu['AVERAGE'] <= 2.0) & (updatedfu['AVERAGE'] > 1.0)]
updatedfu6 = updatedfu[donutgraph['AVERAGE'] <= 1.0]
```

Figure 4.21 All types of Student Distribution

In the Student Assessment section, the information about whether the first ten percentile of the student is in or not was provided. In this section, the other percentage slices were calculated. To make this calculation, all students were ranked from the best to the worst. The total number of students is divided into percentile slices. In this way, the student can be discussed as to the percentile he/she is in and how the process can be improved.

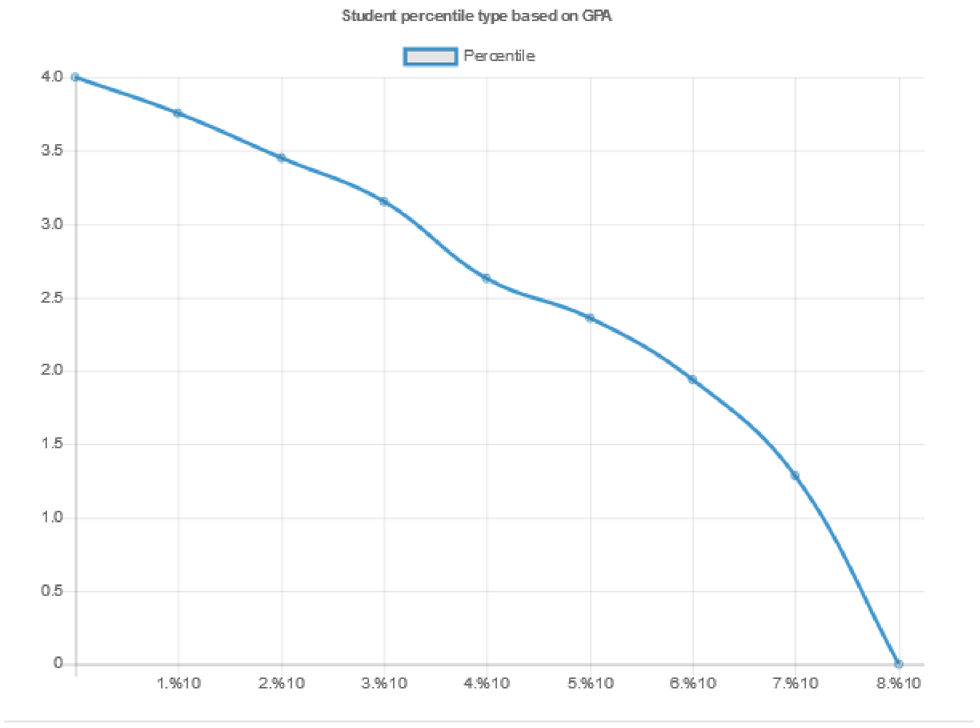
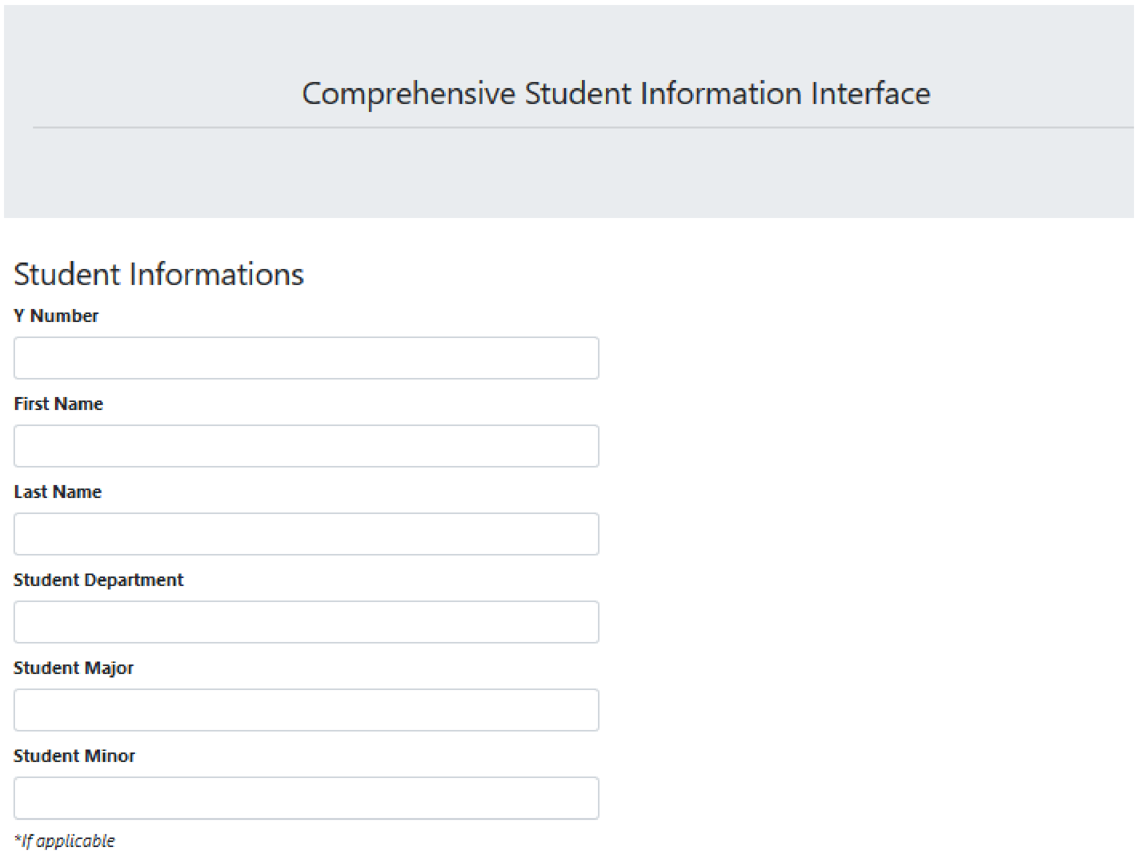


Figure 4.22 Student percentile type based on GPA

4.3.1 Discussion

Since the study does not have data of a graduate student, and the course records of the students, it cannot be determined the use of a machine learning algorithms at this point for better prediction models. The accuracy of the algorithms that can be used for the existing data can be discussed. If more data about students can be gathered, it will be possible to identify the area to help them in this process. There are many external factors that affect students' success. These factors can sometimes be caused by the student herself/himself, by family, and sometimes by environmental factors. In order to meet this data and to solve the problems of the student, another interface was also designed with some questions for students. In case the data needs are met, it is aimed to solve the other factors that affect the process of the students. Figure 4.23 contains a set of basic questions of students.



Comprehensive Student Information Interface

Student Informations

Y Number

First Name

Last Name

Student Department

Student Major

Student Minor

*If applicable

Figure 4.23 Student Personal Information

Chapter Five

Conclusion and Future Work

This study aimed to help identify students who have difficulty in the learning process, to monitor their performances, and to train more competent students in academia and industry. Within the scope of this investigation, a new performance monitoring system was introduced, the necessary interface was designed to identify the factors that would harm students during their education rather than their personal problems. Although there was not much information about the student, some results can still be determined. In addition to the education life, general life information about the student can also play a major role in improving the performance of the student. When looking at the current studies, there are different types of employee monitoring systems used for improving the industry.

Data has a very important place today. It gives information about the present day and the future for companies, schools, or governments. At this point, with more data about the student, more information could be calculated. The computations made in this study will yield more healthy results depending on the size of the data used. At this point, more effective solutions can be developed for the problems faced by students through the interface in which their information is collected. With the results obtained, students and advisors can intervene faster with the automatic notification system.

REFERENCES

[1] J. J. Selingo. (2015). How many colleges and universities do we really need? [Online]. Available: https://www.washingtonpost.com/news/grade-point/wp/2015/07/20/how-many-colleges-and-universities-do-we-really-need/?noredirect=%20on&utm_term=.084130b3a537.

[2] E. Duffin. (2015). U.s. college enrollment statistics 1965-2028, [Online]. Available: https://www.statista.com/statistics/183995/us-college-enrollment-and-projections-in-public-and-private-institutions/.

[3] Bhardwaj, B. and Pal S., "Data mining: A prediction for performance improvement using classification," *(IJCSIS) International Journal of Computer Science and Information Security*, vol. 25, no. 4, 2011.

[4] V. Arturo, A. Zaldivar, D. B. Pardo, and C. Kloos, "Monitoring student progress using virtual appliances: A case study," *Computers and Education*, vol. 58, pp. 1058–1067, 2012.

[5] Natek, S and Zwilling, M, "Student data mining solution–knowledge management system related to higher education institutions," *Expert systems with applications*, vol. 41, pp. 6400–6407, 2014.

[6] Abeer Badr El Din Ahmed and Elaraby, I S, "Data mining: A prediction for student's performance using classification method," *World Journal of Computer Application and Technology*, vol. 2, pp. 43–47, 2014.

[7] IBRAHIM, ZAIDAH and RUSLI, DALIELA, "Predicting students' academic performance: Comparing artificial neural network, decision tree and linear regression," *21st Annual SAS Malaysia Forum*, 2007.

[8] Alfan, E and Othman, M N, "Undergraduate students' performance: The case of university of malaya," *Quality Assurance in Education*, vol. 13, no. 4, pp. 329–343, 2005.

[9] Nelson, C Van andNelson, J S and Malone, B G, "Predicting success of international graduate students in an american university, college and university," vol. 80, no. 1, pp. 19–27, 2004.

[10] Frantz, Paul L., and Alex H. Wilson, "Frantz, paul l., and alex h. wilson. "student performance in the legal environment course: Determinants and comparisons," *Journal of Legal Studies Education*, vol. 21, no. 2, pp. 225–239, 2004.

[11] Shahiri, A M and Rashid, N and Husain, W A, "A review on predicting student's performance using data mining techniques," *The Third Information Systems International Conference*, vol. 72, pp. 414–422, 2015.

[12] MacQueen, James, "Some methods for classification and analysis of multivariate observations," *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14, 1967.

[13] Lloyd, Stuart P., "Lloyd, stuart. "least squares quantization in pcm," *IEEE transactions on information theory 28.2*, pp. 129–137, 1982.

[14] Pelleg, Dan, and Andrew Moore, "Accelerating exact k-means algorithms with geometric reasoning.," *CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE*, 2000.

[15] E. Hatipoglu. (2018). Machine learning - clustering (kumeleme)- k-means algorithm -hierarchical clustering (hiyerarsik kumeleme part13, [Online]. Available: https://medium.com/@ekrem.hatipoglu/machine-learning-clustering-k%C3%BCmeleme-k-means-algorithm-part-13-be33aeef4fc8..

[16] Faber, V, "Clustering and the continuous k-means algorithm," *Los Alamos Science*, no. 22, 1994.

[17] Aydin, S, "Veri madenciliği ve anadolu üniversitesiuzaktan eğitim sisteminde bir uygulama," 2007.

[18] (). The python tutorial, the python software foundation is a non-profit corporation, [Online]. Available: https://docs.python.org/3/tutorial/index.html,March16th%202015..

[19] F. Aslam, H. Mohammed, and P. Lokhande. (2015). Efficient way of web development using python and flask. international journal of advanced research in computer science, [Online]. Available: http://www.ijarcs.info.

[20] B. Gulay, Y. Degirmenci, H. Dirik, and A. Gorur. (2019). Visual assistance for blind people, [Online]. Available: https://github.com/CankayaUniversity/ceng-407-408-2018-2019-Visual-Assistance-for-Blind-People/blob/master/Project%20Report%20V2.pdf.

[21] P. Guo. (2014), [Online]. Available: http://www.pgbovine.net.

[22] A. Ronacher. (2015). Flask web development, one drop at a time, [Online]. Available: http://flask.pocoo.org/,.

[23] ——, (2015). Jinja is beautiful, [Online]. Available: http://jinja.pocoo.org/.

[24] Pallets. (2019), [Online]. Available: http://jinja.pocoo.org/.

[25] G. André. (2018). How to use chart.js, [Online]. Available: https://medium.com/javascript-in-plain-english/exploring-chart-js-e3ba70b07aa4.

[26] S. Jonathan. (2018). 11 javascript data visualization libraries for 2019, [Online]. Available: https://blog.bitsrc.io/11-javascript-charts-and-data-visualization-libraries-for-2018-f01a283a5727.

[27] (2019). Chart.js — open source html5 charts for your website, [Online]. Available: https://www.chartjs.org/.

[28] P. Smyth. (2019). Creating web apis with python and flask, [Online]. Available: https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask#what-flask-does..

[29] Patricksoftwareblog.com. (2019). Template inheritance – patrick's software blog, [Online]. Available: http://www.patricksoftwareblog.com/tag/template-inheritance/.

[30] S. Abuse. (2019). Serving static files with flask, [Online]. Available: https://stackabuse.com/serving-static-files-with-flask/.

[31] Patricksoftwareblog.com. (2019). Template inheritance – patrick's software blog, [Online]. Available: http://www.patricksoftwareblog.com/tag/template-inheritance//.

[32] Flask-mysqldb.readthedocs.io. (2019). Welcome to flask-mysqldb's documentation! — flask-mysqldb 0.2.0 documentation, [Online]. Available: https://flask-mysqldb.readthedocs.io/en/latest/..

[33] Y. S. University. (). Course catalog 2019-2020, [Online]. Available: https://catalog.ysu.edu/undergraduate/general-information/academic-policies-procedures/credit-hours-class-standing-majors/.

[34] M. Sharma. (2019). What steps should one take while doing data preprocessing? [Online]. Available: https://hackernoon.com/what-steps-should-one-take-while-doing-data-preprocessing-502c993e1caa.

[35] K. A. Wetterstrand. (2016). Dna sequencing costs: Data from the NHGRI genome sequencing program (GSP), [Online]. Available: www.genome.gov/sequencingcosts.

APPENDIX

May 7, 2020

Dr. Abdu Arslanyilmaz, Principal Investigator
Mr. Nurettin Senol, Co-investigator
Department of Computer Science and Information Systems
UNIVERSITY

RE: HSRC PROTOCOL NUMBER: 162-2020
TITLE: Periodic Performance Analysis to Predict Student Success Rates

Dear Dr. Arslanyilmaz and Mr. Senol:

The Institutional Review Board has reviewed the abovementioned protocol and determined that it meets the expectations of DHHS 45 CFR 46.104(d)(4) and therefore is exempt from full committee review and oversight. Your project is approved.

Any changes in your research activity should be promptly reported to the Institutional Review Board and may not be initiated without IRB approval except where necessary to eliminate hazard to human subjects. Any unanticipated problems involving risks to subjects should also be promptly reported to the IRB.

The IRB would like to extend its best wishes to you in the conduct of this study.

Sincerely,

Dr. Severine Van Slambrouck
Director Research Services, Compliance and Initiatives
Authorized Institutional Official

SVS:cc

c: Dr. Abdu Arslanyilmaz, Acting Chair
Department of Computer Science and Information Systems