České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra počítačů

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Václavek Jonáš

Studijní program: Otevřená informatika
Obor: Umělá inteligence

Název tématu: Sběr a prediktivní analýza studentských dat

Pokyny pro vypracování:

Proveďte rešerši zadaného tématu.
Seznamte se se stávajícími možnostmi evidence docházky na ČVUT.
Seznamte se s problematikou Learning Analytics. Navrhněte řešení pro elektronickou evidenci docházky na ČVUT. Implementujte řešení elektronické docházky. Toto řešení ve spolupráci s FS ČVUT otestujte a implementujte případné rozšíření a opravy. Navrhněte systém pro analýzu a modelování studijních výsledků jednotlivých studentů. K analýze a modelování využijte stávajících dat z FS ČVUT a dat, která sesbíráte pomocí evidenčního systémů. Implementujte systém v jazyce R za pomoci technologie Shiny. Otestujte funkcionalitu systému. Analyzujte výsledky modelování. Je možné formulovat nějaká doporučení pro výuku? Aplikujte vytvořené modely pro predikci studijních výsledků. Ověřte, zda lze data od docházce spolu se základními demografickými údaji použít k predikci studijních výsledků studenta v jednotlivých předmětech. Predikce porovnejte s reálnými výsledky a výstupy porovnání vyhodnoťte.

Seznam odborné literatury:

[1] Matloff, Norman: The Art of R programming: A tour of statistical software design - San Francisco, 2011
[2] Wickham, Hadley: Advanced R. New York, 2014
[3] Schlesinger Michail I., Hlaváč, Václav: Deset přednášek z teorie statistického a strukturního rozpoznávání. Praha, 1999
[4] Kuhn, Max, Johnson, Kjell: Applied predictive modelling. New York, 2013
[5] Lord Norton, Porter, Sarah: From Bricks to Clicks: The potential of data and analytics in higher education. London, 2016

Vedoucí: Ing. Jakub Kužílek, Ph.D.

Platnost zadání do konce zimního semestru 2018/2019

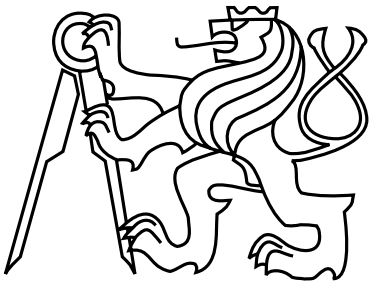
L.S.

prof. Dr. Michal Pěchouček, MSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 21.7.2017

Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Science and Engineering



Diploma thesis

Collection and predictive analysis of student data

Bc. Jonas Vaclavek

Supervisor: Dr. Jakub Kuzilek

Study Programme: Open informatics

Field of Study: Artificial intelligence

May 23, 2018

Aknowledgements

I thank my supervisor, Dr. Jakub Kuzilek, for many valuable advices and constructive conversations during the implementation of solutions in this thesis, and for helpful comments on the text.

My sincere thanks also goes to my family, which has been always supportive throughout the whole studies.

Declaration

I hereby declare that I have completed this thesis independently and that I have listed all the literature and publications used.
I have no objection to usage of this work in compliance with the act §60 Zákon č. 121/2000Sb. (copyright law), and with the rights connected with the copyright act including the changes in the act.

In Prague, 23. 5. 2018 .....

Abstract

In past decade, the higher education institutions face the problem of low student retention. At the same time, the education has become digitalized as various ICT systems such as Virtual Learning Environments have been employed. These systems collect a student-related data, which can be further analysed. This thesis presents results achieved when solving a task of identification first-year engineering students at risk of failing their studies at Faculty of Mechanical Engineering (FME), Czech Technical University in Prague (CTU). For the analysis, data from CTU study information system KOS combined with the records of student attendance and in-semester tests have been used. Additionally, web application - *Seminarist*, has been implemented to provide lecturers with a comfortable way of a collection of the student attendance and results. The collected data has been analysed using machine learning methods, and the results have been communicated back to FME lecturers, to help them identify those students, who may gain from an early help. To provide FME staff convenient way of accessing analysis results, web application - *Analyst* has been created. The FME reported improvement of the retention rate in the first study year.

Abstrakt

V posledních letech, vysokoškolské instituce čelí problému vysokého počtu studentů, kteří nedokončí studium. Paralelně dochází k rozmachu využití ICT technologií ve výuce a využívání různých online systémů výuky Tyto systémy ukládají informace o studentech a ty pak mohou být dále analyzována. Tato práce prezentuje výsledky řešení problému retence studentů prvního ročníku na Fakultě strojní (FS), Českého vysokého učení technického v Praze. Pro analýzu byla použita data z informačního systému KOS kombinovaná s informacemi o docházce a výsledcích semestrálních testů studentů. Pro zjednodušení evidence docházky učitelem byla vytvořena webová aplikace - *Seminarist*. Výsledky analýzy metodami strojového učení byly komunikovány zpět zaměstnancům strojní fakulty, kde byly použity pro identifikaci studentů, kteří mohou mít problémy se studiem. Pro snadnější a rychlejší přístup k výsledkům analýzy byl vytvořen online nástoj - *Analyst*. FS následně reportovala zlepšení retence studentů prvního ročníku.

Contents

Chapter 1

Introduction

Higher education has experienced extensive growth of educational systems based on Information and Communication Technologies (ICT). Students have started using Virtual Learning Environments (VLE) such as Moodle (Moodle 2018), which are used by the educators to achieve various learning goals such as providing study materials, tests, course management and overall improvement of the learning process. Also, a new trend of Massive Open Online Courses (MOOC) has emerged. Platforms like Coursera[1] or edX[2] provide a lot of courses from diverse fields to everybody in an online form. Both VLEs and MOOCs collect a vast amount of student-related data, which can be further analysed. At the same time, universities started to face a problem of low student retention, especially in the first year of a university degree. In EU countries between 20% and 54% of students fail to complete their degree (Quinn 2013).

Potential of student data has been analysed in a broad number of studies (Papamitsiou and Economides 2014). As Shacklock (2016) and Ferguson et al. (2016) show *Learning Analytics* (LA) seems to be one of the most promising fields regarding the potential of such analysis. It is defined as 'The measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimising learning and the environments in which it occurs' (Ferguson 2012). LA is a fast-growing field, which uses and synthesizes techniques from different related fields such as Educational Data Mining, Big Data, Machine Learning, Business Analytics and Web Analysis to convert the student-related data into a useful information leading into improvement and a better understanding of learning processes and environments students and educators work with.

One of the typical ways of visualisation of the results is using learning dashboards (Jivet et al. 2018). For example, the Open University one of the biggest universities in the United Kingdom extensively uses a system called OU Analyse. The system is the main topic of our previous work Vaclavek (2015) and aims at an early prediction of students at risk of failing a course. Multiple dashboard views demonstrating the predictions are available to Student Support Teams which can decide how to intervene in student's educational plan (Kuzilek et al. 2015). The use of systems providing similar dashboards is growing and OU Analyse is only one of them (Jivet et al. 2018).

The Faculty of Mechanical Engineering, Czech Technical University in Prague (FME) is no exception among the universities in terms of drop-out. They deal with low retention of students and especially in the first year of degree where student drop-out is approximately 15%. There is a need to reduce this number and for this purpose, we have started cooperation with the FME. Our main task in the analysis is to address the problem of first-year student retention. In order to do so, we have analysed these students by applying methods from Learning Analytics and Education Data Mining.

In this thesis, we focus on a collection of student-related data and creation of views on the collected data from different perspectives. These views can be used by the teaching staff to better understand patterns in student behaviour. The data is also analysed by supervised learning methods to create predictive models,

[1] https://www.coursera.org/
[2] https://www.edx.org/

which aim to identify students at risk of failure in studies so that the staff can intervene them and help them to successfully complete the degree.

In the first chapter, existing system for data collection is demonstrated. In Chapter 3, a web application for collection of students' attendance and course results - *Seminarist* is presented. Then, available datasets for the analysis are described in Chapter 4. Before the analytical part of the thesis, basics of pattern recognition theory and technologies used during the analysis are explained in Chapter 5. Then, analysis and predictive modelling are applied to the datasets in Chapter 6. Finally, the learning analytics dashboard - *Analyst* is shown in Chapter 7.

Chapter 2

Data collection

The first chapter of the thesis addresses the problem of data collection, which is the first step before the data analysis can be made. For this purpose, we analysed systems used at FME to collect data about students and their possibilities to provide the data for the analysis.

2.1 Student information system

The FME uses a system, which is used at all faculties of the Czech Technical University in Prague (CTU). The system - KOS - provides tools to students and university staff. The tools help students for example with registration into courses, planning schedule for an academic year, signing up for exams or management of personal information. It helps to teachers when providing information about courses, assigning grades to students or planning exam dates.

KOS is a valuable source of the information regarding the student-related data. There exists a service KOSApi allowing to access part of the data, but its functionality is limited and it is not officially supported across the university. It also requires a dedicated server managed by the faculty, which the faculty does not have resources for.

However, the data can be exported into CSV files either directly from the KOS or using an iKOS system, which is an extension of the KOS system. The former approach does not allow to export full range of data. The latter approach provides an option to export all kinds of data, but it requires a special application for that. Both systems export data into Comma Separated Values (CSV) files.

For the needs of the analysis, we have exported personal and demographic data of the students and their results in courses. The available attributes are student's:

- CTU personal identifier
- year of birth
- form of study (full-time or combined)
- study year
- year of graduation on secondary school
- postcode
- gender
- identification number of secondary school
- grade in a course (including a date in which the grade has been assigned to the students)
- parallel in a course

The data is split into multiple files. There is one file in which personal and demographic data of the students are defined. The other files represent courses of an academic year and contain information about attending students and their results with a date of assignment of the grade.

2.2 Other systems

The data from the KOS system gives information only about the final results of students in courses. The FME does not dispose of any system, which allows the staff to collect attendance and intermediate results of the students. Teachers keep notes about the progress of students in a course in paper form or electronic form of their choice. However, there is no single interface for all of them. It implies that these data cannot be collected either by the faculty or by the university for further use. For this purpose, we have implemented our system *Seminarist*, which is presented in the next chapters.

Chapter 3

Seminarist

In this chapter, the attendance application *Seminarist* is presented. Firstly the specification of the system is defined, then short introduction about web applications is provided and finally, the application is shown.

3.1 Seminarist requirements

With an assist from the FME staff, a specification of the application for the collection of course-related data has been created. The application is called *Seminarist*. The process of specification making went through a lot of changes but always existing structure of the courses at CTU (and especially at FME) has been taken into account to create a system, which can be used (with small modifications) at any faculty of the CTU. The final specification or rather set of requirements is presented in the following sections.

3.1.1 General system requirements

In the beginning, the set of key requirements has been defined:

- **Web application** - the application should be available from the Internet so that its users do not have to install the application locally and data stored in the application are available constantly
- **Responsive design** - it is possible to use the application from various types of devices (notebooks, tablets, mobile phones).
- **Security** - Since the application will be available from the Internet and contains sensitive data about students (personal data, results in courses and so on), the user needs to log in to the application.
- **REST API** - Communication between browser and server will be accomplished using REST API (Maly 2009)

3.1.2 Entity-relationship diagram

Firstly, we have identified the data units (entities) in the application. The goal of the task was to create a model, which is able to represent all data existing in the application and also the database is able to represent it without difficulties. The following list demonstrates the identified entities:

- **Student** - represent information collected about student. A student is defined by CTU identification number and full name.
- **Course** - represents a single run of a course taught in one academic year. Each course is defined by name, academic year, semester and type of final assessment.

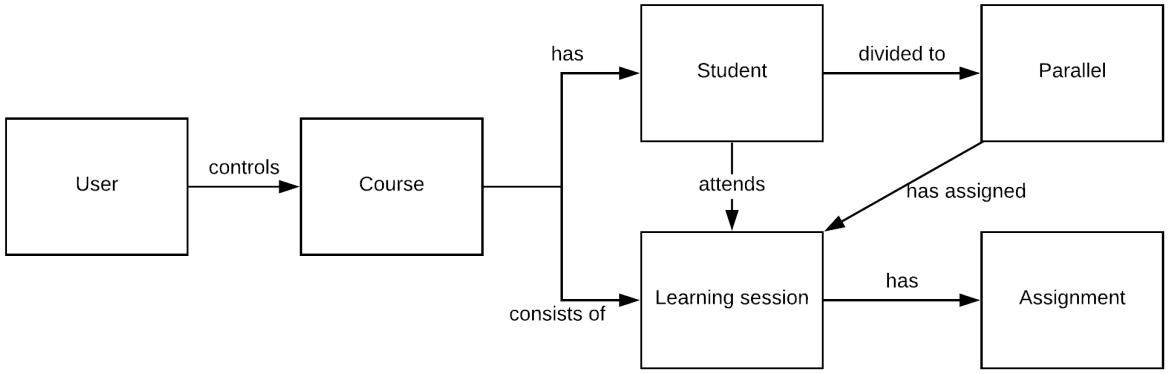


Figure 3.1: Basic model of the connection between entities of the Seminarist application

- **Parallel** - a study group into which students of a course are divided into. A parallel is defined by the following attributes: name, number of study weeks, day, start and end time of a learning session and odd/even flag in case the course is not taught weekly.
- **Learning session** - stands for one lesson of a course. It can be lecture, seminar or laboratory. Each learning session can be formative or summative. The entity is also used for collection of students' attendance.
- **Assignment** - represents activities of the students during the academic year. An assignment can be homework, test, drawing or technical protocol from a laboratory seminar. An assignment might have the result represented as yes-no value, mark or points.
- **User** - application user, typically lecturer, who collects the attendance. The entity is defined by the following attributes: name, username and its role.

Figure 3.1 demonstrates the general data model consisting of entities and their relations. The diagram can be interpreted as follows:

- Users of the application are in control of courses
- The courses consists of multiple learning sessions, which may also have an assignment
- Courses are attended by a set of students, which are divided into parallels
- Students attend learning sessions based on parallel they are assigned to

3.1.3 User roles

Each application user has assigned a user role. We distinguish three different roles:

- **Administrator** has access to all data in the application, can arbitrarily manipulate with them and can create a new user. In other words, the administrator can use any feature of the Seminarist
- **Garant** is allowed to perform all actions as the administrator. Only creating new users is not permitted
- **Teacher** has an option to change attendance records, assignment results, and final grades. A teacher can also view detail of courses, parallels or students but only for entities which administrator or garant made accessible for the user. Application user with this role cannot create or manipulate other entities

The set of user roles may be extended in the future hence the system has to be prepared for it.

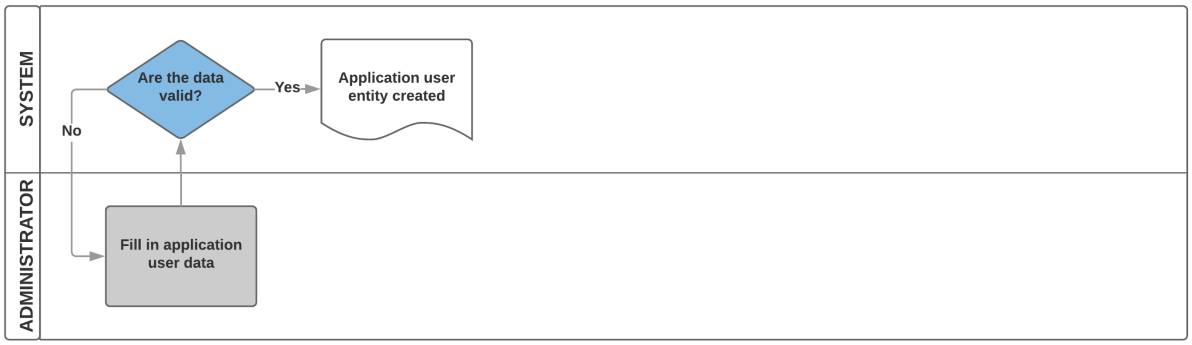


Figure 3.2: Seminarist process flow from Administrator user role perspective

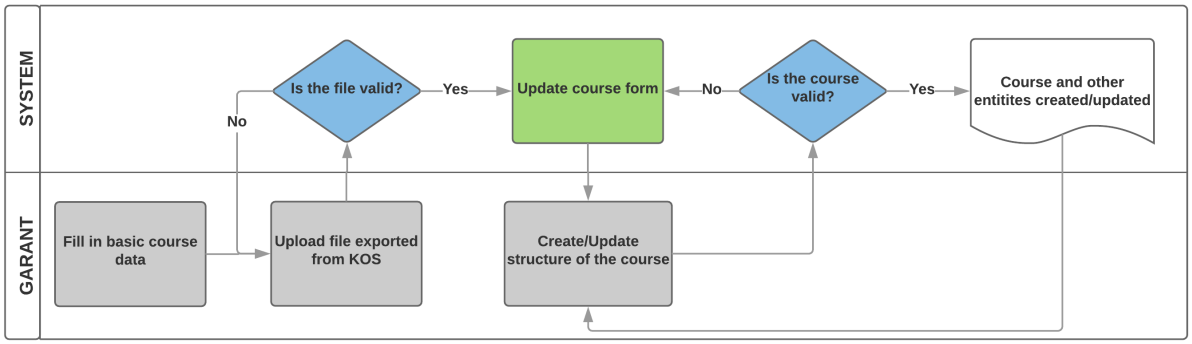


Figure 3.3: Seminarist process flow from Garant user role perspective

3.1.4 Process flow

Each user role is expected to behave differently. To reflect that behaviour, we have developed different process flows. Figures 3.2, 3.3, 3.4 show the general flow divided into three parts with respect to the user roles. Each part reflects what users with a specific role are expected to do with the application. Common scenarios have been modelled.

Administrator focuses on creating new users of the application. The process of creating new users requires to fill in information about the user and if valid, the user can be created.

Garants are in charge of creating and managing courses. To create a course, the garant exports data from the KOS system. The data contains information about students assigned to the course and their division into parallels. When the file is exported, the process of creating a course continues as follows:

1. The garant defines basic course attributes - course name, academic year, semester, types of learning sessions present in the course, number of study weeks, and type of final assessment
2. The system validates the imported file
3. The user can create a course structure. It is update the division of students into parallels, define learning sessions and assignments
4. The system validates the course structure. If the structure is valid, a new course is created

The user can also update the previously created course and save the changes. The system validates the course structure and if it valid the course is updated.

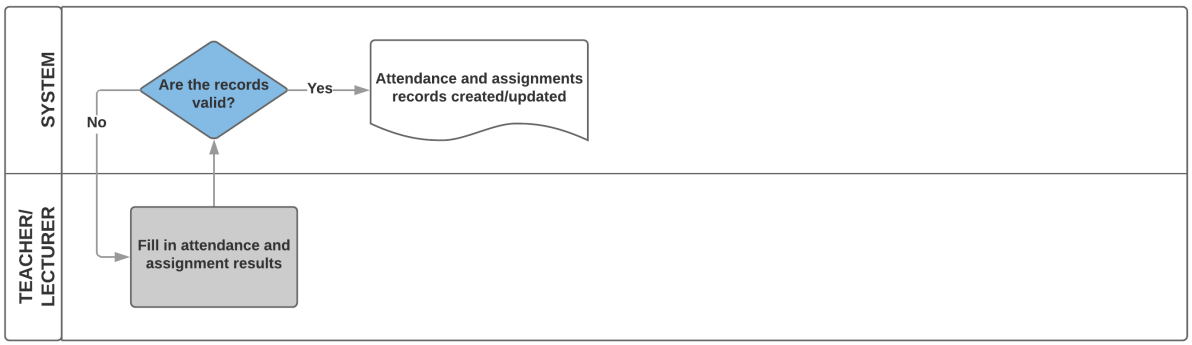


Figure 3.4: Seminarist process flow from Teacher user role perspective

Teachers collect the attendance. For this purpose, the application provides a form, which lists students of a course parallel and enables the teacher to manage the attendance and assignments.

3.2 Development stack

The specification of the Seminarist requires the system to be created as a web application. The web application is a client-server computer program, which users typically run in a web browser (Vaclavek 2015). Its architecture is usually divided into two parts - front-end and back-end (Vaclavek 2015). On the following lines, we show components of both parts and technologies we used for development of the Seminarist.

3.2.1 Front-end

The client side of the application is called front-end. It is basically any content the user can see and interact with. Hypertext Markup Language (HTML), Cascading Style Sheet (CSS) and JavaScript (JS) languages are used to create and implement the content and interactions (Vaclavek 2015). Each of them serves a different purpose. The HTML defines the structure of a web page, the CSS is used to define a look of the web page and JS enables the application to dynamically react on the interactions of the user. Browsers interpret and control the code and as a result a web page, which the user can interact with is created (Vaclavek 2015).

It is recommended to use a JavaScript framework, which helps with the development of the front-end (Andras 2017). The number of available frameworks is large although at the beginning of the Seminarist development there were two major frameworks - Angular.js ("AngularJS — Superheroic Javascript Mvw Framework" 2010) and React ("React - a Javascript Library for Building User Interfaces" 2013). Both are used to create so-called Single page applications. These frameworks are designed in a way that parts of the web page are created dynamically as the user interacts with the application. The page does not have to be reloaded entirely, which increases the user experience. The process of dynamic changes of the web page is called routing. Angular.js and React apply different approaches to the implementation of the Single Page Application features. Nevertheless, both can accomplish the same thing. We have chosen to use Angular.js as it had a larger community of users. which could have helped with any difficulties during the development.

Angular.js is a framework developed by the Google company. Its main idea is to create small reusable components which consist of JS code and HTML templates. The components can be nested into each other and their composition creates an Angular application. Angular also provides many service Application Programming Interfaces (API) which can be used to load data into the components, solve routing or create application custom services.

When the development of the Seminarist application has started, the most recent and stable version was

1.5.8. Next version Angular 2[1] was in Beta version and promised different approach to the Angular application architecture as the architecture of 1.5.8 version was too complex. However the version was still in experimental phase, hence we have decided to stick with the stable version. From today's perspective, it was not the best decision as the new version shows better performance (Sharma 2014) and is simpler to use.

3.2.2 Back-end

The server side of the application is called back-end and typically consists of three parts - web server, database and application scripts and services (Vaclavek 2015). Web server is a computer running all software necessary for communication between a user and web application.

A database is a place where the majority of data is stored. The database enables manipulation with the data using create, read, update and delete operations (CRUD).

Application scripts and services decide what actions should be carried out and how they should be done with respect to a current state of an application.

Application scripts and services of the Seminarist are developed in Node.js ("Node.js" 2010), which is a JavaScript engine enabling developers to write server-side code in JavaScript. The whole process of communication with client and database can be established via the Node.js and there is no need to use other languages. Basic functionalities of the Node.js can be extended by importing packages from npm package system ("Npm Package Manager" 2010). One of the packages is Express ("Express - Node.js Web Application Framework" 2010), which makes from the Node.js application Node.js web server. We use the Express web server to implement REST API, which is consumed by the front-end in order to deliver data to the application user. The REST API can be used by other software to communicate with the *Seminarist* to either update its own data or data in the *Seminarist*.

As a database solution, we chose to use MySQL database ("MySQL" 1995). It is an open source relational database management system, which uses language similar to Structured Query Language. MySQL is often associated with web applications or online publishing, although it is not the most typical choice together with Node.js ("9 Best Databases to Use for Node.js Applications as of 2018" 2018). We expected that data coming from the Seminarist application will be used in the analytical part as well and conversion of the data into the analytical database would be easier.

To increase the security and performance of the web application reverse proxy server is implemented. Requests from the Internet are checked and the check may include whether the request is authenticated, whether it comes from secured network and so on. When the check is successfully finished, the request is sent to the web application (Figure 3.5).

3.3 Seminarist implementation

Based on the specification and using the presented technologies we have implemented the Seminarist application and the following sections present it.

3.3.1 Architecture overview

The Seminarist is hosted on a server owned by Czech Institute of Informatics, Robotics, and Cybernetics (CIIRC). It is divided into front-end application, back-end application and MySQL database. The front-end application is in charge of the static content of the Seminarist (HTML, CSS, JS and images) and also forwarding requests to the back-end application on data from the database. The back-end application controls data flow from the database to the front-end application and vice-versa. Both applications run

[1] The version is oficially called only Angular as it is basically new framework. However we use the notaion Angular 2 to show the timeline of the Angular framework development

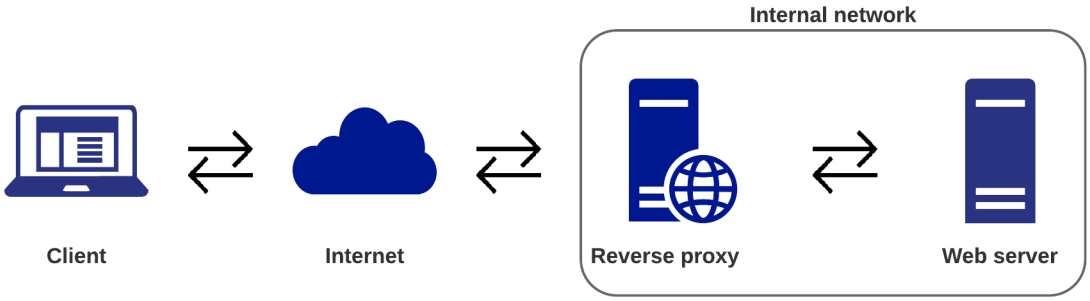


Figure 3.5: Diagram of the reverse proxy method

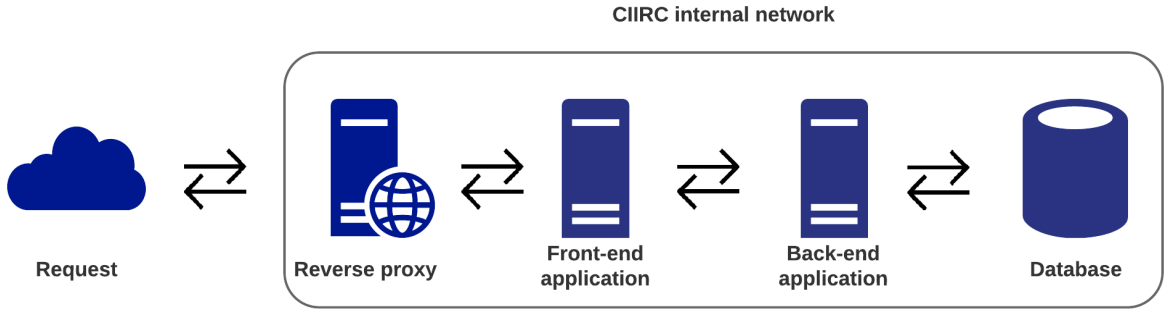


Figure 3.6: Seminarist architecture overview

on the Express web server and they are not exposed outside of the computer they run on. To make the Seminarist available from the Internet, we made use of the Apache HTTP server (Brian Behlendorf 1995), which provides reverse proxy capabilities and its easy configure. The server is configured so that user request is authenticated against the database and if the authentication succeeds, the user is shown the content of the web application, otherwise the user request is rejected. Overview of the architecture is presented in Figure 3.6.

3.3.2 Database model

Based on the specification (Chapter 3.1), we have designed a database schema to cover all the entities and processes in the application. The database schema is shown in Appendix A[2]. The database contains tables of five types:

- **Course tables** - represent course related data including learning sessions of each course
- **Results and attendance tables** - used to store results of students in courses and their attendance
- **Student tables** - hold data about students and parallels they are assigned to
- **Assignment tables** - collect data of students' assignments
- **User tables** - used to define application users and their roles

[2]As opposed to the specification the database model uses credit instead of assessment and assessment instead of assignment.

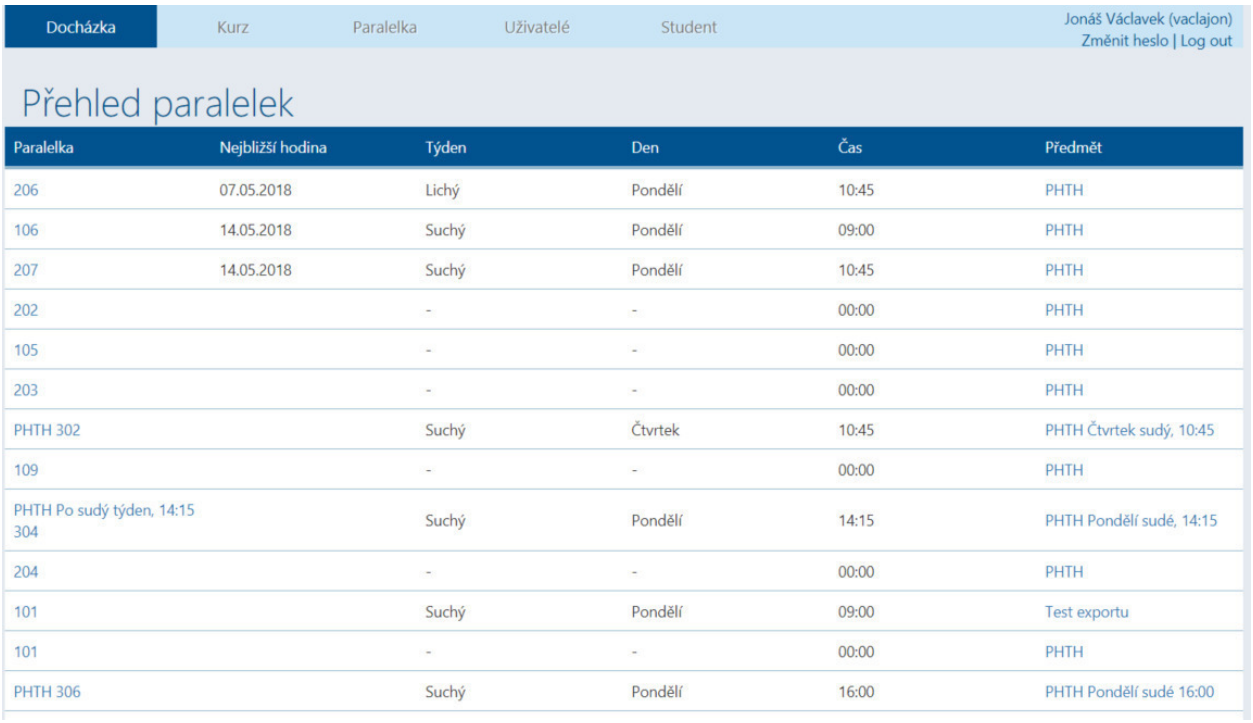


Figure 3.7: Seminarist default page

3.3.3 User roles and rights

The specification defines user roles. The set of available features of the *Seminarist* depend on the role assigned to the user. We have extended the specification and added a notion of user rights. These rights are defined for each user separately. Each action in the application is assigned a specific right and user needs to have the right in the set of rights to perform the action. For example, when a new student needs to be created, the user must have *SEM_STUDENT_UPDATE* right. By default, there is a set of rights assigned to each user based on the user role. However, the rights can be added or removed for a particular user. It adds more variability in available features to users of the application.

3.3.4 Front-end and back-end applications

On top of the database model, the front-end and back-end applications have been implemented to create the *Seminarist*. On the following lines, we demonstrate the *Seminarist* from client's perspective (front-end).

When the user accesses the application and logs in successfully default page is shown (Figure 3.7). The page consists of two components. At the top, there is a navigation menu component, which provides a user with the option to browse different views of the application. Set of buttons depends on the user role of the user. Besides the navigation buttons, the menu provides information about the currently logged in user, a button used to change a password of the logged in user and log out button. The component has an alternative version for devices with a smaller screen (Figure 3.8).

The rest of the page contains a table component, which is a part of the page body. Optionally, there is an action bar component located in the lower part of the page and contains buttons which serve for the interaction of the user with the application. Set of action buttons in the action bar differs from page to page and user role of the logged in user (Figure 3.9).

The content of the page body differs from page to page based on the selected view in the navigation menu. The views are of three types:

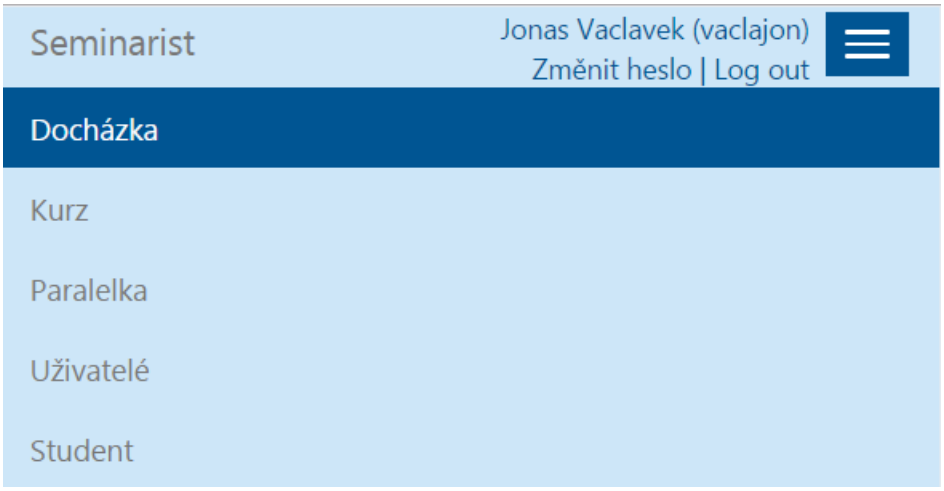


Figure 3.8: Seminarist navigation menu displayed from small screen devices

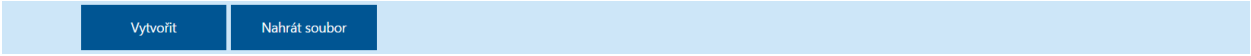


Figure 3.9: Example of an action bar with two action buttons

- list views
- data collection views
- data administration views

3.3.4.1 List views

The Figure 3.7 contains a table, where parallels are listed. Page body of the figure is an example of the list view. Purpose of the view is to list all entities of a specific type available to a logged in user. The view consists only of a table. For the sake of reusability, we have created a specialized Angular component.

The component needs to be specified two mandatory parameters. Firstly, a data source, which can be either a content already available in the user's browser or link to a data which needs to be downloaded from the server. The second parameter is a set of column definitions, where each column definition represents one column of the table. It describes a type of the record, a name of the attribute in the data, column name and other optional attributes. Other optional parameters can be also specified to modify the behaviour of the component. For example, set text, which is shown in case no data is present in the table.

The component takes these all the parameters and takes care of the data retrieval and data rendering. The following snippet of HTML code shows the required markup:

```
<CTM-TABLE
  data-table-header="vm.attendanceHeader"
  data-data-source="attendanceService"
  data-data-source-params="{method: 'getAvailable'}"
>
</CTM-TABLE>
```

The code defines that the column definitions are stored in the variable *attendanceHeader*, the source of the data is *attendanceService*[3] and method name, which should be used from the service is *getAvailable*. Example of the *tableDefinition* variable is shown in the following snippet:

[3]See https://docs.angularjs.org/guide/services for details about services

```
[
 {
   name: 'parallel.code', columnName: 'Paralelka',
   type: 'link', linkUI: 'attendance.detail'
 },
 {
   name: 'parallelLearningSession.date', columnName: 'Nejbližší hodina',
   type: 'date'
 },
 {
   name: 'parallel.week', columnName: 'Týden',
   type: 'week'
 },
 {
   name: 'parallel.day', columnName: 'Den',
   type: 'day'
 },
 {
   name: 'parallel.time', columnName: 'Čas',
   type: 'time'
 },
 {
   name: 'course.name', columnName: 'Předmět',
   type: 'link', linkUI: 'course.detail'
 }
]
```

As mentioned above, using these two code snippets (or their variants) results in a table, which is similar to the table in Figure 3.7.

3.3.4.2 Data collection views

The main purpose of the Seminarist is the collection of student attendance and results in courses. For that purpose, data collection views have been created. They can be accessed from the default page by clicking a name of a parallel in the table (Figure 3.7). There are three different views but only one of them is shown at a time. The user can switch between them in the top of the page. The views are:

- **Attendance table** - The table lists all students registered to a course and have following columns (Figure 3.10):
  - name of a student - full name of a student
  - attendance summary - shows how many learning sessions a student has attended so far from the total number
  - attendance columns - for each week of a semester one column is generated. It contains a checkbox, which has three states. The states represent whether a student has attended a learning session, substituted it somewhere else or has not attended a learning session at all
  - assessment - indicates whether a student has been assigned an assessment of not
  - grade - shows a result of a student in a course

- **Assessment table** - enables the user to fill in results of assignments given to the students during a semester. The table has a similar structure as the attendance table, it contains following columns (Figure 3.11):

Figure 3.10: Attendance table.

Figure 3.11: Assessment table.

  - name of a student
  - attendance summary
  - assignment summary - shows how many assignments a student has finished so far
  - assignment columns - for each assignment one column is generated. The user can fill in values based on a type of the assignment
  - assessment
  - grade

- **Mobile view** - the view is designed mainly for the users accessing the application from devices with a narrow screen. The two previous tables are wide and it is not simple to fill in data from the narrow screens. The mobile view shows attendance data week by week. To change a week, there is a component at the top of the view. Based on the selected week, the view shows a table of students with the following columns (Figure 3.12):
- attendance column - contains the same checkbox component as in the attendance table
- name of a student - full name of a student
- date - used to select a date, when a student has attended a learning session

3.3.4.3 Administration views

The administration views are used to manage the course, parallel, student and user entities of the application. Since workflow with parallel, student and user entities is very similar, because they do not have complicated structure, their set of views is explained together.

The *Seminarist* provides a simple form (Figure 3.13), which can be used to manage an entity. For each of the entities, the form differs based on the attributes of the entity. In the form, attributes need to be filled

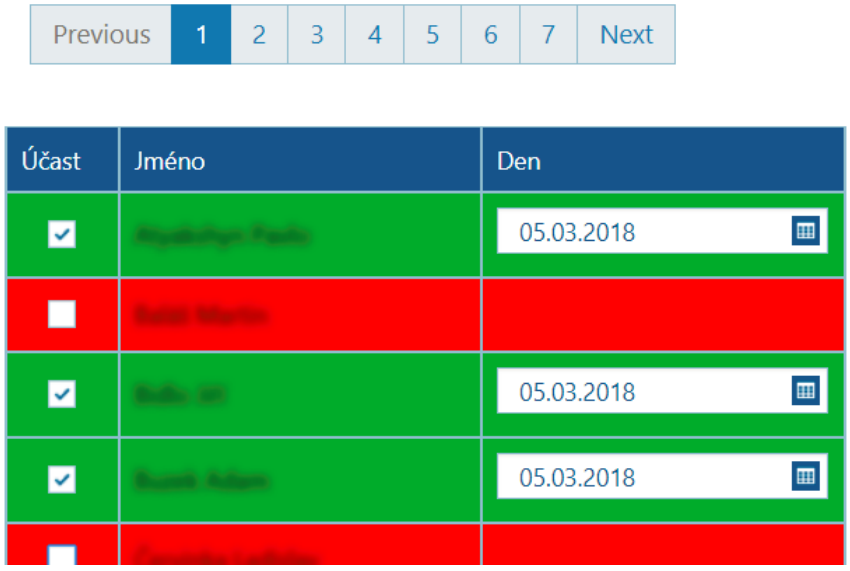


Figure 3.12: Mobile view

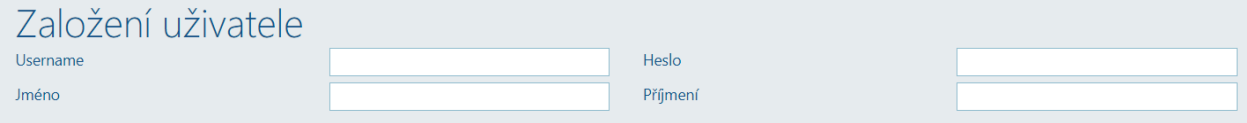


Figure 3.13: Form used to set basic attributes of an user entity

in, the user can click a save button in the action bar component and the entity is created. A similar flow is also used to update the attributes of the existing entity.

Next, administration views have been created for a course entity. When a new course is being defined, the situation is more complex compared to creating other entities. The user needs to follow steps defined in the process flow (Section 3.1.4):

1. Fill in basic course attributes (Form similar to Figure (3.13)
2. Open dialog window by clicking a button from the action bar (Figure 3.14)
3. Select and upload a file exported from the KOS
4. As specified in the process flow, when the file is uploaded and successfully processed, the application form is updated to enable the user to define the learning sessions and alternatively the assignments (Figure 3.15).

When the user completes the definition of the course attributes and structure, save button in the action bar can be clicked. The application creates a new course based on the specified parameters including relations between students and parallels as specified in the KOS. If all the actions complete successfully the user is redirected to the newly created course detail page.

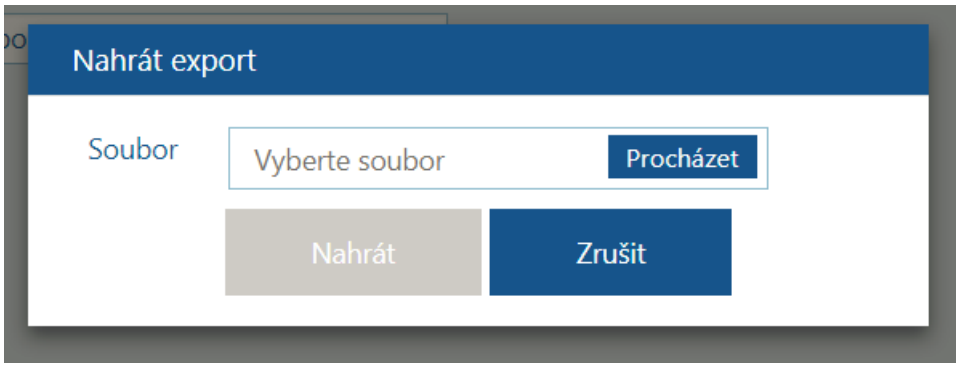


Figure 3.14: Form used to select a file to be uploaded while creating a course

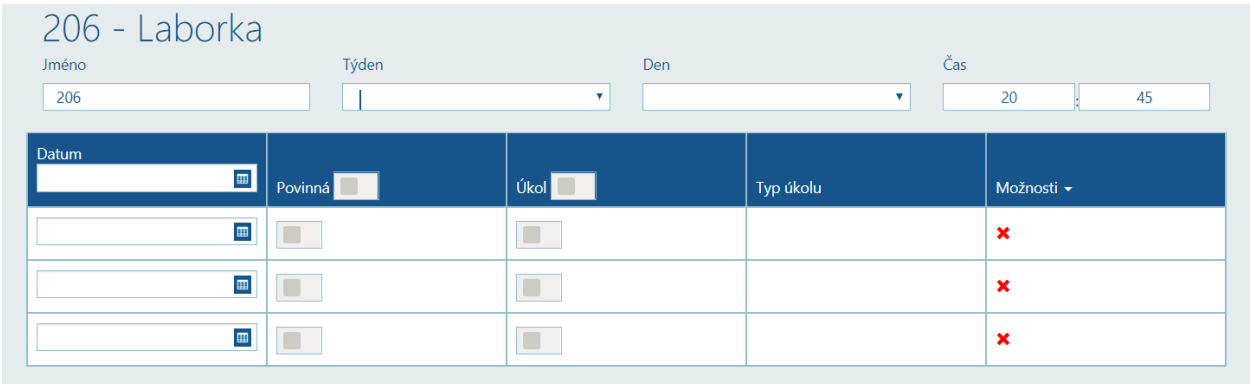


Figure 3.15: Form used to specify attributes of a learning session

Chapter 4

Datasets

In this chapter, the datasets, which we have created for the analytical part are described. The data comes from two sources: files exported from the KOS system and the Seminarist application. The files exported by the FME staff provides information about all courses taught for first and second-year students and are available for four consecutive academic years starting from year 2013/2014. The *Seminarist* application is still in its testing phase and contains data only for one course.

From the available data, we have created two types of datasets. The first is used to analyse the performance of a student in a whole academic year based on final grade. The second analyses performance in a single course based on final grade and student's attendance and performance during the semester. Possibilities of the second approach are limited by the fact that attendance data and other course-related data (e.g. results from tests) are available only for a single course.

4.1 Dataset for academic year performance analysis

In the analysis presented later in this thesis, we focus on the analysis of first-year students. Identification of those students is actually not straightforward from the available data. Even though the data contains information about student's *study year*, which could be used to identify the first-year students, the value of the *study year* is oftentimes not correct. Therefore another approach has to be applied.

We have used a white book from FME[1] as next source of information. The white book is a document, which contains information about study plans. Sets of mandatory courses have been extracted from it for first-year students of each academic year. Then, we selected only those students which have results from all the mandatory courses. Using this approach, we have created four datasets for each of the academic years - AC2013, AC2014, AC2015, AC2016. Each of the datasets contains approximately four hundred of first-year students along with course results. In the thesis, we call them *AC* (ACademic) datasets. One more dataset has been created by a combination of all the AC datasets into one ACFULL dataset with more than sixteen hundred students. The exact numbers of students in the datasets are shown in Table 4.1.

Datasets consist of the following attributes: * CTU identification numbers of student * Student's gender * Student's postcode * Student's study form * Information whether a student is from the first year or not

- Course name
- Number of ECTS credits
- Result in a course
- Date when a course has been finished (result has been given to a student)

[1] https://dms.fs.cvut.cz/12922_Public/%2FB%C3%ADl%C3%A1%20kniha/

Table 4.1: Overview of number of students in datasets

	Number of students	Without passive withdrawals
AC2013	402	352
AC2014	422	376
AC2015	429	374
AC2016	413	365
ACFULL	1652	1458
MECH2016	201	NA

4.2 Dataset for single course analysis

As mentioned above, the amount of data available for the analysis of students in a single course is limited. The Seminarist contains attendance information of more than only one courses. However, only for one course, there are also final results of students in the course. From those data, we have created a dataset - MECH2016. The dataset consists of 201 students and each student record gives information about the attendance at seminars (there is thirteen study weeks - thirteen attendance records), dates and results of tests (there are three tests) and final grade.

Chapter 5

Theory and technologies

Before we dive into the analysis of student-related data, we provide a brief theoretical introduction of the methods used later in the analysis. All the methods fall into the area of pattern recognition, which can be divided into two basic groups - supervised and unsupervised learning. Both are explained in the following two sections. The last section of this chapter presents technologies used to perform the analysis and visualize the data.

5.1 Unsupervised learning

Unsupervised learning, also often referred as clustering (Xu and Wunsch 2005), works with unlabeled data. In general, it is a task where data are divided into multiple groups (clusters) based on observed patterns. Data in one group are more relevant/similar to each other than to data in the other groups. There are various algorithms for performing the clustering and it cannot be said that only one is the correct one to use. The choice of the algorithm depends on input data and expected results of the unsupervised learning.

5.1.1 *k*-means

k-means is one of the clustering algorithms (Xu and Wunsch 2005). It aims to find k points (centroids) to which each point of the input dataset can be assigned and at the same time sum of distances of points in each cluster to its centroid is minimal. The problem of finding such centroids is NP-hard (Xu and Wunsch 2005). Nevertheless, there is a heuristic approach which converges quickly to a local minimum (Lloyd 1982). Typical methods for evaluation of the local minima can be used to ensure that the result is the best possible - run the k means algorithm multiple times, start from different points and so on.

For a given set of samples $(\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n})$, where each sample has same dimmension, k-means creates k ($k \leq n$) sets $S = \{S_1, S_2, ..., S_k\}$ into which the samples are divided. During the partitioning the algortihm minimizes the following objective function

$$\sum_{i=1}^{k} \sum_{\boldsymbol{x_j} \in S_i} \|\boldsymbol{x_j} - \boldsymbol{\mu_i}\|^2$$

where μ_i is the mean value of samples in a cluster $\boldsymbol{S}_i$. The μ values define the cluster centroids. The standard algorithm uses the iterative refinement approach to find them It works as follows:

1. Choose the values of μ (centroids) randomly

2. Assign each point from the dataset into a relevant centroid using the following formula:

$$S_i^{(t)} = \left\{ x_p : \left\| x_p - \mu_i^{(t)} \right\|^2 \leq \left\| x_p - \mu_j^{(t)} \right\|^2 \ \forall j, 1 \leq j \leq k \right\}$$

, where each x_p has to be assigned to exactly one $S^{(t)}$.

3. Update centroids of each cluster using points assigned to it in the previous step so that

$$\mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

4. Repeat steps 2. a 3. until the μ values do not change.

Note that in the step 2 Euclidean distance is used, although different distance metrics can be applied as well.

5.1.2 Choosing number of clusters

The algorithm needs to be specified number of clusters beforehand. Choosing the right number of clusters - k is an important part as it is tightly coupled with the accuracy of the algorithm. The k can be selected either by prior knowledge of the input dataset or using a different method. In both cases, choice of the k should balance the data compression (select only one cluster) and maximum accuracy (cluster for each data sample) (Xu and Wunsch 2005).

Many methods to select the k exist and elbow method is one of them (Kodinariya and Makwana 2013). The quality of the model, can be described by calculating the within-cluster sum of squares (WCSS), which is defined by formula $\sum_{j=1}^{k} \sum_{x_j \to c_j} D(c_j x_j)^2$, where j represents a number of a cluster, k total number of clusters, $x_j \to c_j$ means sample x from a cluster j and $D(c_j x_j)^2$ is calculated as distance between center of a cluster j and the x_j point. WCSS cannot be minimized as it would result in k equal to a number of samples in the dataset and so the method tries to select k so that adding more clusters does not help to model the data significantly better. It can be done either visually or mathematically. In both cases, WCSS for multiple ks needs to be obtained. For the visual computing, the WCSS is displayed as a function of the number of clusters (Figure 5.1). The number of clusters is selected at a point where slope in a change of the variance between clusters drops. The mathematical approach maximizes the second derivative of the variance formula.

5.2 Supervised learning

Unsupervised learning divides data into several groups based on the similarity between elements in those groups. Supervised learning seeks for a decision function f, which generalizes the input data so that the function may be used for a classification of unseen samples. The function is inferred from N training samples. Each sample is defined by a vector of features x and label (class) c of the sample (Xu and Wunsch 2005). For the decision function we can define expected loss of the function:

$$R(f) = \frac{1}{N} \sum_{i=1}^{N} W(c_i, f(x_i))$$

, where $W(c_i, x_i)$ is loss function for classification of a sample x to class d.

The supervised learning algorithms aim at minimization of the expected loss function R. As in case of the clustering, there are various algorithms and the choice depends on the input dataset. The following lines demonstrate the classification algorithms used within the thesis.

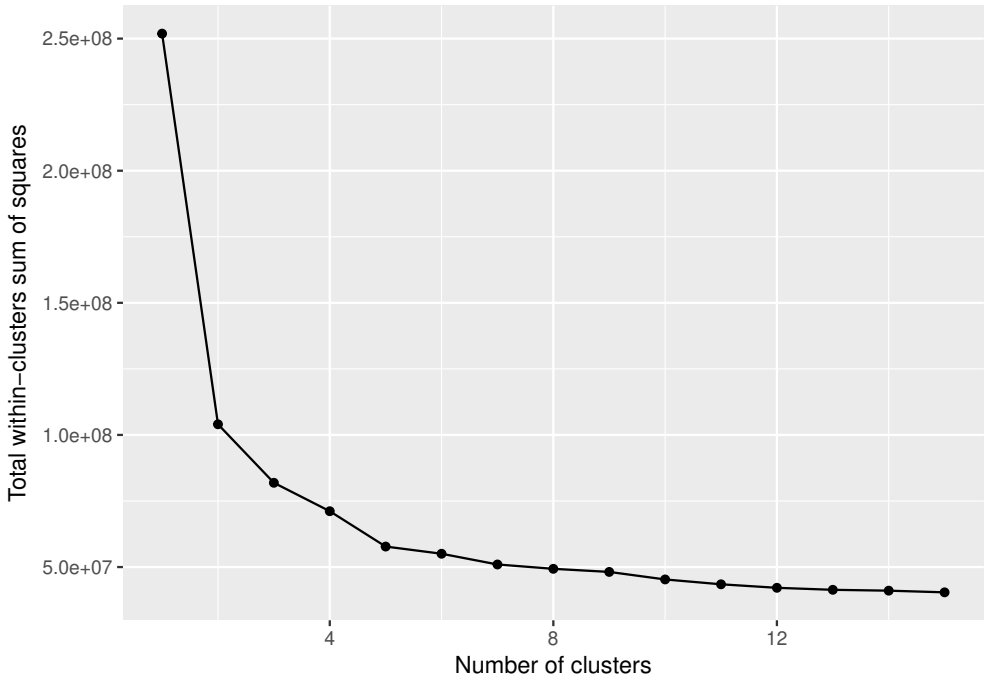


Figure 5.1: Clustering Elbow method graph

5.2.1 *k*-nearest neighbors (*k*NN)

One of the simplest supervised learning strategies is called *k*NN (Altman 1992). It does not need a training phase, training data with known classes are only stored in the memory (Cunningham and Delany 2007). Classification of a new sample is based on k nearest samples (neighbours) from the training set. The nearest samples are typically chosen by distance metric, for example, Euclidian distance (Cunningham and Delany 2007). Since the classes of the selected samples are known, they create a final class of the testing sample by selecting the most frequent one by majority voting. The majority voting has its drawbacks in situations where samples of one class are way more frequent than the other. Such class typically dominates the voting and the result does not have to be correct (Cunningham and Delany 2007). The problem can be solved by many techniques, for example weighing the classes with regards to the distance from the testing point. The following example shows how does the method work.

In the example, we use $k = 3$. It means that three nearest neighbours are to be found for each test sample. The training dataset is depicted in the left part of the Figure 5.2. The graph shows results from Test 1 on the X-axis and on the Y-axis results from Test 2. Based on the results, students are assigned either class *Pass* or *Fail*. When a new student (testing sample) needs to be assigned a class, the *k*NN algorithm finds three nearest samples from training dataset. The result is shown in the right part of the Figure 5.2, where selected samples are encapsulated by the black circle with centre at the position of the new sample. Since there are two *Fail* samples and one *Pass* sample, the student is assigned *Fail* class by the algorithm.

The *k*NN algorithm has high memory requirements as it stores all the data into the memory. Voronoi diagrams (Kolahdouzan and Shahabi 2004) or condensation (Sutton 2012) of the data may be used. To improve the performance of the algorithm K-D trees (Bentley 1975) may be applied.

5.2.2 Naive Bayes classifier

Another used classifier is Naive Bayes classifier (Murphy 2006). The algorithm combines the Bayes' formula with an assumption of independence of predictors. In other words, the presence of a feature in a class is not

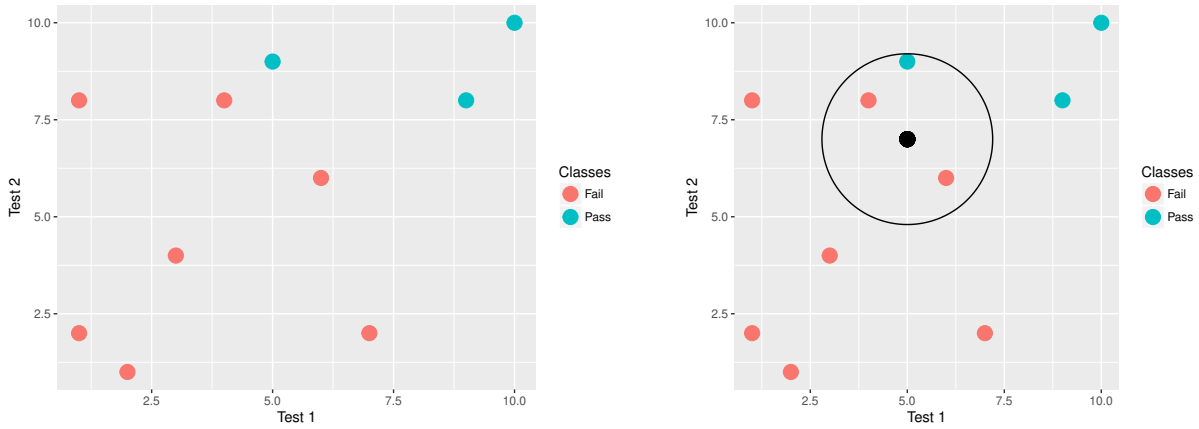


Figure 5.2: Example of kNN algorithm. Training dataset (left) and test sample with its three nearest neighbors (right)

related to any other feature.

The Bayes' formula describes the probability of an event, knowing the prior probability of conditions that might be related to the event. It is defined as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{5.1}$$

where A and B are events and $P(B) \neq 0$.

The formula states how to calculate posterior probability $P(A|B)$ when prior probabilities $P(A)$ and $P(B)$ (both needs to be independent of each other) and conditional probability $P(B|A)$ are known. The formula can be used in many fields. For example in medicine to calculate a probability that a randomly selected individual with a positive test is a drug user, knowing the proportion of drug users in a population and parameters of the test, or in a case of the Naive Bayes classifier.

Let us demonstrate the use of the theorem with an example. The left part of Table 5.1 represents the training dataset of student results in a course. The dataset consists of students from three study years and provides their results in the course. In this case, years of studies make a set of observations - *First*, *Second* and *Third* and set of classes contains *Pass* and *Fail* cases. From the frequency table (Table 5.1) we can calculate the prior probabilities and likelihood probabilities. For example, the prior probability of class *Pass* can be calculated as a sum of all *Pass* students divided by the total number of students (central and right part of the Table 5.1):

$$P(Pass) = \frac{100}{200} = 0.5$$

Likelihood probabilities state a probability of an evidence based on given class. For example probability that student is from second study year given that comes from passing group is

$$P(Second|Pass) = \frac{30}{100} = 0.3$$

All likelihood probabilities are shown in left part of the Table 5.2. Last part of the computation is a calculation of the posterior probabilities using the Bayes' formula. In this case, the formula has the following form

$$P(Class|Evidence) = \frac{P(Evidence|Class)P(Class)}{P(Evidence)}$$

and its application creates posterior probabilities presented in right part of the Table 5.2.

Table 5.1: Frequencty table of input dataset (left) and prior probabilites (center and right)

Classes	First	Second	Third
Pass	10	30	60
Fail	70	30	0

Class	Probability
Pass	0.5
Fail	0.5

Observation	Probability
First	0.4
Second	0.3
Third	0.3

Table 5.2: Likelihood frequency table and posterior probabilities table (right)

Classes	First	Second	Third
Pass	0.1	0.3	0.6
Fail	0.7	0.3	0.0

	First	Second	Third
Pass	0.125	0.5	1
Fail	0.875	0.5	0

5.2.3 Classification and regression tree (CART)

Classification and regression tree is a method for building classification models from data (Breiman 2017). The model partitions data space into multiple parts and fits the model within each part. The parts can be displayed in a form of decision trees.

Recall the dataset from kNN example, there were two tests based on which the class *Pass* or *Fail* has been assigned. Figure 5.3 demonstrates a decision tree constructed from those data. Each node of the tree creates a rule and based on the rule the samples are divided into classes. If a sample satisfies the rule, it follows the *yes* path, otherwise, it goes to the *no* path of the tree. For example the root node, creates a rule $Test1 < 8$. Eight data samples fulfil the rule and so they follow the *yes* path. Similarly for the rest of the nodes. Leaves of the tree represent the final class of a sample. There are seven samples with *Fail* class and three *Pass* samples. It is consistent with the data in the dataset. New samples are classified by applying rules from the root of the tree. In case of the new student with results five and seven from Test 1 and Test 2 respectively, we apply the rule $Test1 < 8$, which is met and then rule $Test2 < 8.5$ which is also met. Therefore the test sample is again classified by *Fail* class.

5.3 Technologies

In this section, we introduce R programming language and its packages providing additional features.

5.3.1 R statistical tool

R is a programming language for statistical computing, graphics and interactive environment for a wide spectrum of statistical and graphical techniques (Chambers 1976). It has been developed under GNU General Public License as an open source project in 1993. The language is an implementation of another statistical language S (R Core Team 2017) developed in Bell Labs.[1] R can be used on a majority of UNIX-like operating systems, Windows and MacOS. By default, R provides basic statistical and graphical features that are added to the R in a form of packages. The features can be further extended by adding other packages, which are available at Comprehensive R Archive Network[2] (CRAN). CRAN is a collection of sites gathering these packages. The ecosystem of the packages is vast[3] and provides features from a diverse number of fields. One of the R's strengths is the way how data can be easily manipulated, transformed and analyzed using a variable number of analytical techniques and visualized by using one of the available packages for data

[1] http://www.bell-labs.com/
[2] https://cran.r-project.org/
[3] At the time of writing there were 12507 different packages

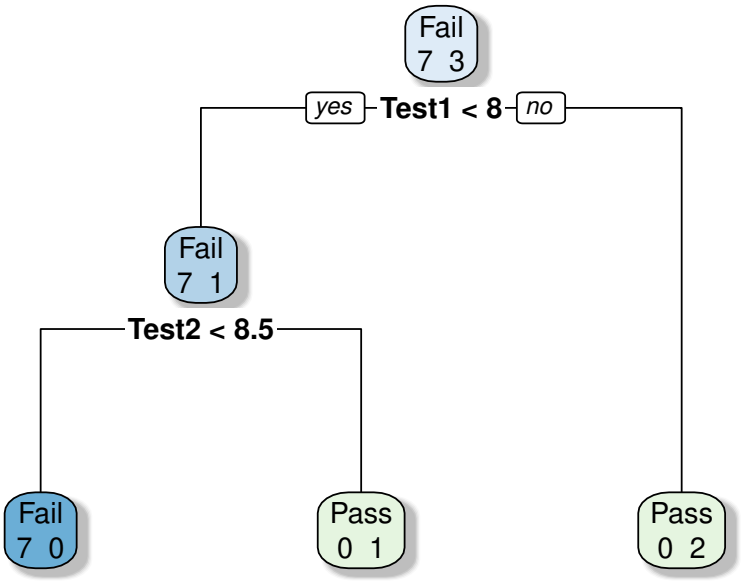


Figure 5.3: Decition tree based on dataset from kNN example

visualisation. R comes with a default plotting package, although oftentimes other packages are used as they offer an easier way to create more complex graphs.

5.3.2 ggplot2

An alternative plotting package to the base package provided by R is ggplot2 (Wickham 2009), which is a system of plotting functions based on the grammar of graphics (Wilkinson 2006) written by Leland Wilkinson in 1999. The idea behind the grammar of graphics is that the graph is divided into multiple layers (Table 5.3 lists the layers and describes the meaning of each layer) and each of them can be defined separately. Parameters of the layers are defined using a pipeline, which is shown in the following snippet of R code:

```
ggplot(data, aes(x,y)) +   # definition of data and data on x and y-axis
  geom_point() +           # display data as points
  geom_smooth() +          # turn on smoother
  theme_classic() +        # use clasic theme
  ggtitle("") +            # define title of the graph
  xlab("Description of X axis") + # define label of X axis
  ylab("Description of Y axis")   # define label of Y axis
```

The pipeline syntax results in easily editable code. Any line of the pipeline can be removed or new line can be added and the code remains valid/functional. Such operations may require a lot more work using the base package. Moreover, the layer approach forces developers to think about the graphs more from programmatic rather than visual perspective. Performance of the ggplot2 graphs is a little bit worse compared to the base package. On the other hand, the final graph typically looks better using the default settings (Figure 5.4).

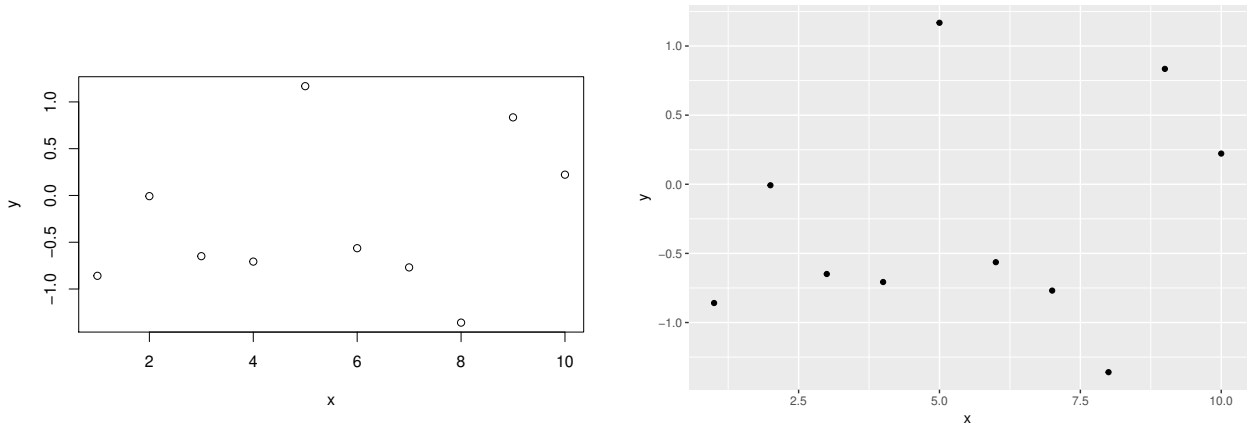


Figure 5.4: Example of graphs created using base (left) and ggplot2 (right) packages

Table 5.3: Layers defined by grammar of graphics

Layers	Description	Mandatory
Data	The dataset being plot	Yes
Aesthetics	The scales onto which the data is plotted	Yes
Geometries	Visual elements used to plot the data	Yes
Facets	Small multiples	No
Statistics	Visual elements used to create a summary of the data	No
Coordinates	The space on which the data will be plotted	No
Themes	Other parameters not related to data	No

5.3.3 R Shiny

Shiny is a powerful R package (Chang et al. 2017) which makes the development of web applications directly from R language very simple.

The code of the Shiny application is divided into two sections. UI section where structure and components of the web pages are defined. As a result of the UI sections, HTML code is generated. The HTML components are actually only placeholders for a content which is defined in server section. The section can contain any R function to create a web page content, for example, text, graph, table or more complex combination of the previous. Finally, the outcomes are converted by special Shiny functions and the results are inserted into HTML structure instead of the placeholders.

Shiny comes with a built-in web server hence a developer only needs to specify the two sections, runs the server and Shiny takes care of converting the two sections into necessary form, deploying it at the web server and makes the application available from a browser. The created application is called Shiny application.

The Shiny applications are created under the idea of reactive applications (Bernhardt 2016). Its underlying idea is to create applications which are scalable, responsive and fault-tolerant. It is achieved by message-driven architecture. In the world of the Shiny applications, it means that whenever the user interacts with a component of the application a message is sent to a server. The server checks what component generated the message recalculates functions defined in the server section relevant to the component and sends new data to the browser. In case that other components are affected by the change, the Shiny application updates those components as well. This behaviour also reduces the amount of code needed by a developer to write.

5.3.4 plotly

Plotly is another data visualization tool, which has its implementation for R (Sievert et al. 2017). Its main advantage over the base and ggplot2 packages is that the final graphs are interactive by default. The user can zoom in/out, extra labels can be added to data points and finally, the graphs can be stored as a PNG file for future use. ggplot2 graphs can be converted into the Plotly format using a single function. Also, the Plotly graphs can be used with Shiny, which gives the Shiny web application another feature.

Chapter 6

Analysis

In the previous chapter, we have introduced methods which we will use for identification of at-risk students. Firstly, we reproduce methods from work of Zdrahal Zdrahal et al. (2016). Then, k means clustering on the AC datasets is applied and analysis of the clusters is performed. Finally, classification is performed on both AC and MECH2016 datasets.

6.1 Performance classes

For the needs of the analysis, students are divided into performance groups using the methodology described in Zdrahal et al. (2016). The performance is measured with respect to the number of European Credit Transfer and Accumulation System (ECTS) ("European Credit Transfer and Accumulation System (Ects)" 2003) credits earned in both winter and summer semesters. Together with the FME staff, we have created criteria for the division of students into the groups based on their teaching expertise. Criteria of the categories are defined by Table 6.1, where the first column provides short description of a class, the second defines abbreviation used later in the analysis, the third defines a condition on number of credits earned in winter semester and the last defines a condition on number of credits earned by the end of an academic year. x variable stands for a number of credits earned by a student in a respective period.

Table 6.1: Definition of performance classes.

Category	**Abbreviation**	**Winter semester**	**End of the year**
Without problems	SOK	$30 \leq x$	$60 \leq x$
Passing with problems in winter or summer semester	OK	$15 \leq x$	$30 \leq x < 60$
Passing with problems only in winter semester	PS	$15 \leq x \leq 30$	$60 \leq x$
Failing in summer semester	NF	$15 \geq x$	$x < 30$
Failing in winter semester	FF	$x < 15$	NA

6.2 Timeline of credits earned by performance class

Performance classes are computed from data for the whole academic year. However, by analyzing the data leading to the assignment of a student to a performance class, we can reveal important patterns (Zdrahal et al. 2016). We have created a graph consisting of five lines. Each line represents one performance group

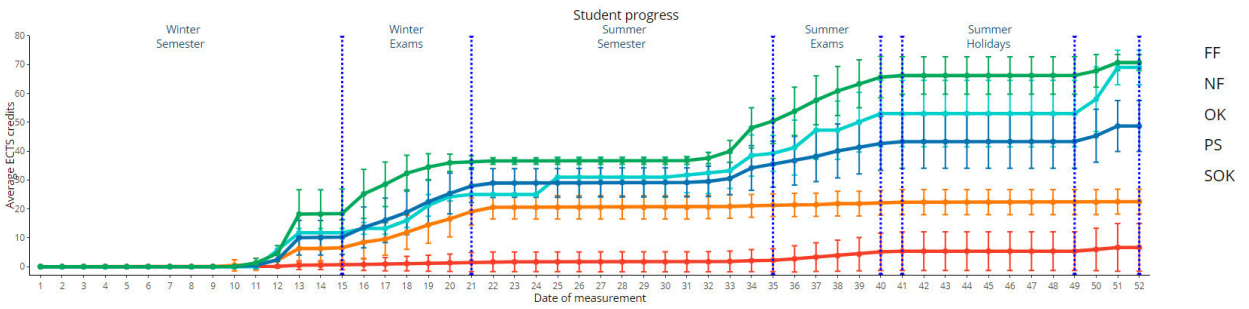


Figure 6.1: Average credits earned by each performance class

and shows the average amount of ECTS credits achieved by students of each group. To construct the graph, students are divided based on the performance classes. Next, a number of credits earned by the students is calculated for each week of an academic year. The sums of credits are averaged and plotted to the graph so that on the x axis are weeks of an academic year and on the y axis the averages are shown. For easier orientation in the graph, dotted lines are added to separate individual parts of the academic year. The parts of an academic year are set according to the white book:

- winter semester
- winter exam period
- summer semester
- summer exam period before summer holidays
- summer holidays
- summer exam period after summer holidays

The graph also contains information about the standard deviation in the data, which is depicted by vertical lines coming from each data point (week) of each line. The Figure 6.1 shows the graph, which has been created using the AC2013 dataset. A similar graph can be made for all the AC datasets.

An important pattern is observed by having a look at the graph. The differentiation into performance classes happens even before the start of the winter period and the observation holds for all the AC datasets. Credits achieved before the start of the winter exam period are obtained for both courses without exam and courses with a possibility of an early exam. A similar pattern occurs in summer semester between PS and OK classes, but not in all the datasets.

6.3 Study probabilities

Another view supports analysis of the first pattern explained in the previous section. We compute the probability of assigning a performance class to a student based on the number of credits up until the start of the winter exam period. The probabilities are calculated using the Bayes' formula (Formula (5.1)). The goal is to calculate the posterior probability of classification into a performance class based on a number of credits earned up until a specific moment. Results are depicted in Figure 6.2, where x axis shows the numbers of credits and y axis shows the probabilities of final performance class.

The data show that students without any credit at the beginning of the winter exam period fail in the academic year. These students are assigned to either *FF* or *NF* performance class and the pattern holds for all the AC datasets.

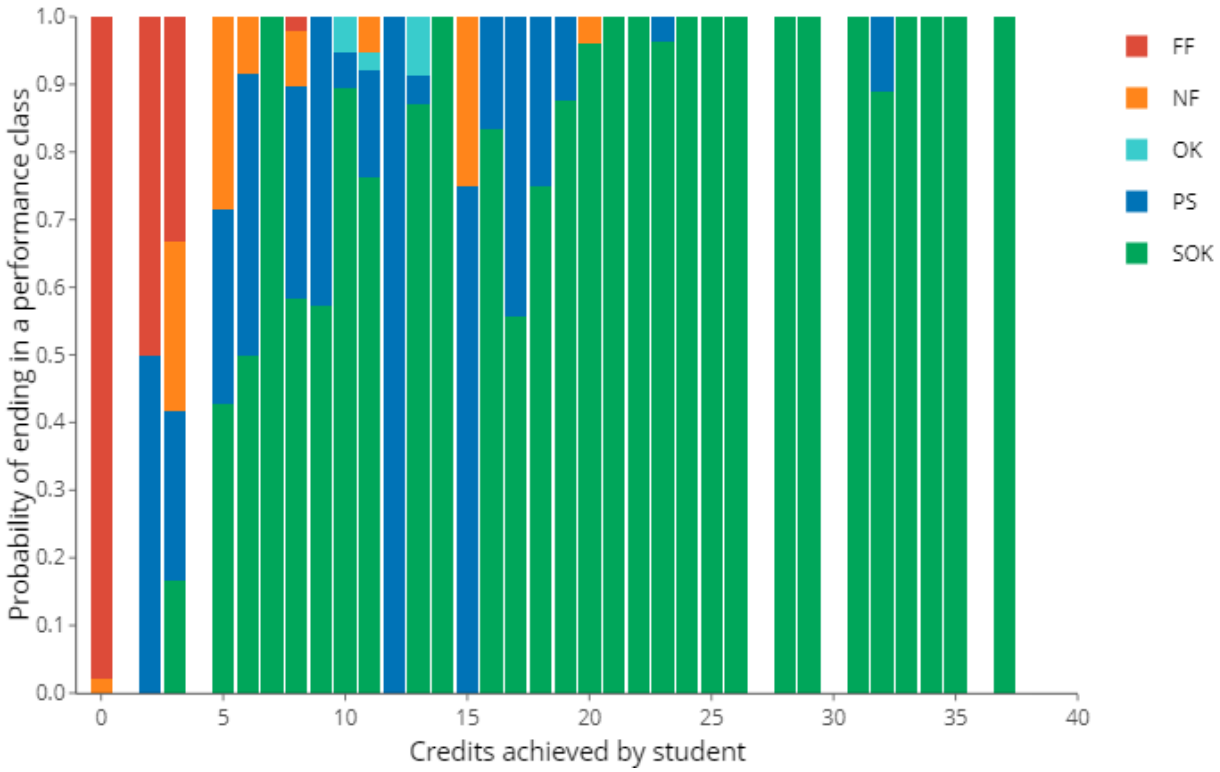


Figure 6.2: Study probabilities graph

6.4 Student clustering

The previous methods work with the predefined performance classes. In this section, we apply the clustering method to divide the students into clusters and analyse the clusters.

Two types of features have been selected for the clustering. The first type is a numeric representation of result in a course. Marks A - F are ordinal values and so we used the values 1 - 6. A second type is a number of days a student needed to successfully finish a course. In case of course taught in a winter semester it is a number of days from the start of the winter exam period. In case of a course taught in a summer semester, it is a number of days from the start of the summer exam period. If a student has not finished a course at all, a number of days between the start of the winter exam period and end of the academic year is used. A negative number of days is given to courses finished in early term periods, i.e. before the start of the winter period. Since each AC dataset contains fifteen courses, the clustering works with thirty features in total - two for each of the fifteen courses.

The elbow method has been used to select a number of clusters. We ran the k-means algorithm with values of k varying from 1 to 15. Figure 5.1 in Chapter 5 was created from the WCSS values obtained from the clustering. By looking at it, we can see that the optimal number of clusters is somewhere between 3 and 6. We decided to follow more than one number of clusters because of two intuitive reasons. If we used only three clusters, the students would be probably split into clusters of *failing*, *passing* and *"somewhere in between"* students. On the other hand, students are given grades on a scale with six possible outcomes. The division into six clusters might reflect the scale and provide interesting insights. We have run the clustering algorithm and performed the analysis of results.

6.4.1 Feature analysis with respect to a course

k-means divides the students into k clusters and we present the analysis of results for $k = 5$. The Figure 6.3 shows a graph where two box plots are created from values of features. The upper boxplot shows days required to finish the course and the lower shows the grades. Clusters are displayed on the x axis and grades/days are shown on the y axis. Numbers of the clusters are assigned randomly by the algorithm. Each run of the algorithm uses constant seed in order to make the results reproducible.

The first thing we can notice is that the bottom plot shows values under zero. It is caused by the fact that some students acquired credits both for early exams but also for assessments carried out during the winter semester.

We can sort the clusters by the performance. When we analyse the Figure 6.3, it is obvious that students from cluster 4 have the best results compared to the other clusters as they have the lowest median. Then, the rest of the clusters may be sorted in order 5, 1, 2, 3. The graph can be also used for comparison of courses between each other (Appendix B). The comparison shows, for example, courses in which students tend to get better grades or have fewer problems with successful finishing.

Last note regarding the Figure 6.3 concerns the worst cluster (cluster 3). The cluster has both medians at the maximum value for both feature types. Such cluster occurs in all runs of the *kmeans* function with $k > 3$. By further analysis of the cluster, we have found that students of the cluster are deregistered from the degree in the winter period because they fail to complete the requirement of a minimum number of earned ECTS credits. Moreover, the majority of them do not attend even a single exam. Since the cluster appears for almost all choices of k, we decided to filter out from all AC datasets students, which do not continue in studies in the second semester. This group of students requires different analytical methods and approaches such as quantitative research. Those students are probably from a group of *passive withdrawals* - students, who are registered in a degree but do not intend to study or group of students, who absolutely struggle with the studies.

We ran the clustering again on the reduced datasets. The removal of the students made the box plots more readable (Figure 6.4). In the rest of the cluster analysis, results from the reduced datasets are presented.

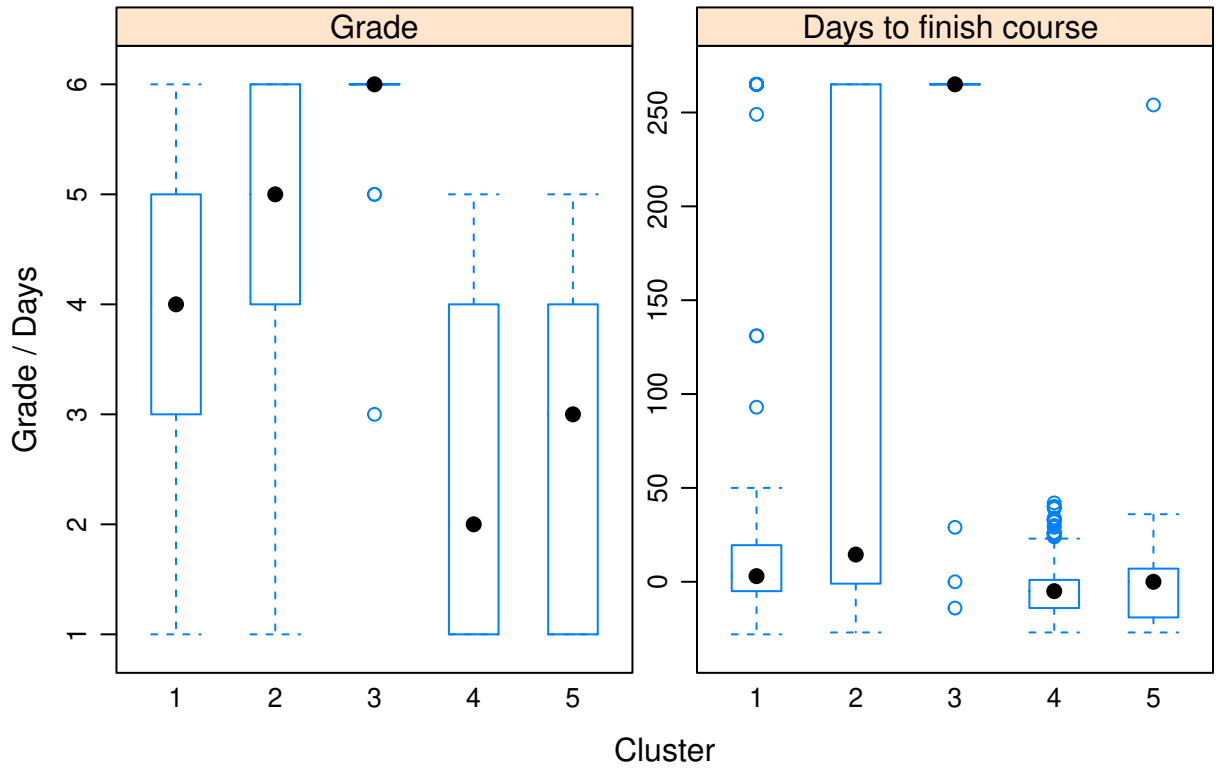


Figure 6.3: Box plot of features of a random course

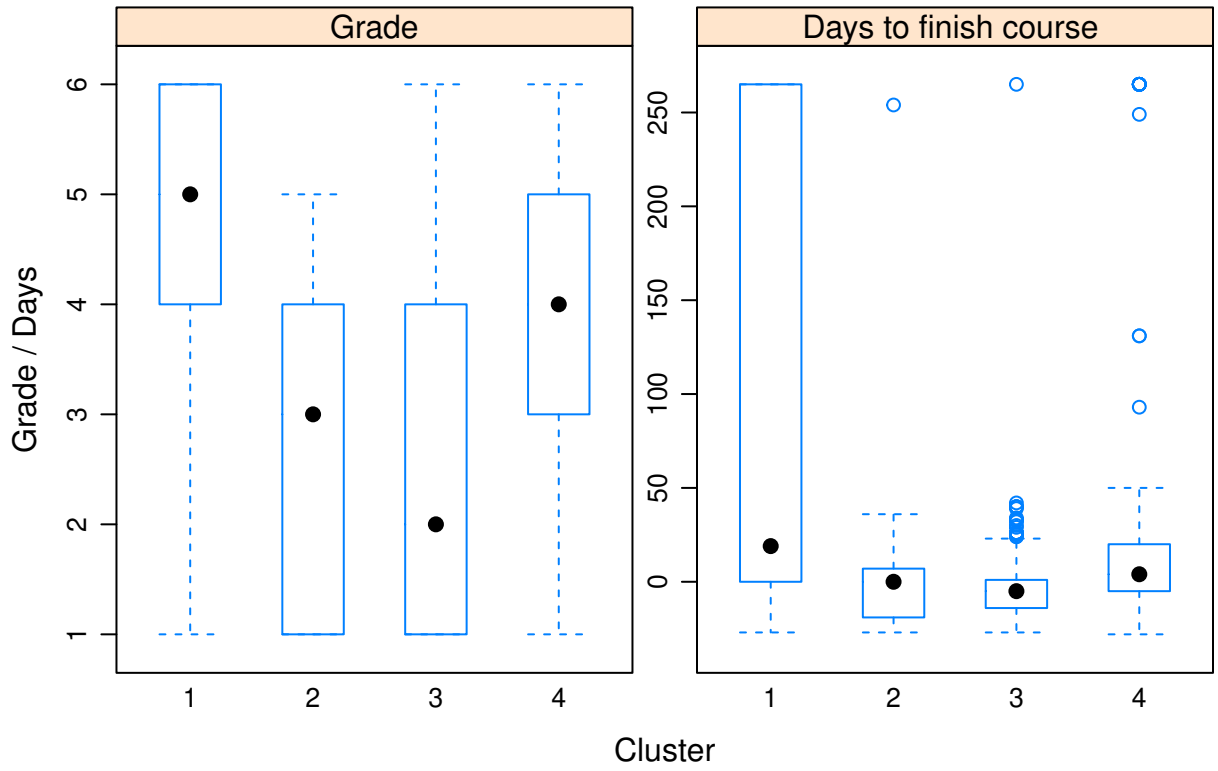


Figure 6.4: Box plot of features of a random course without passive withdrawal students

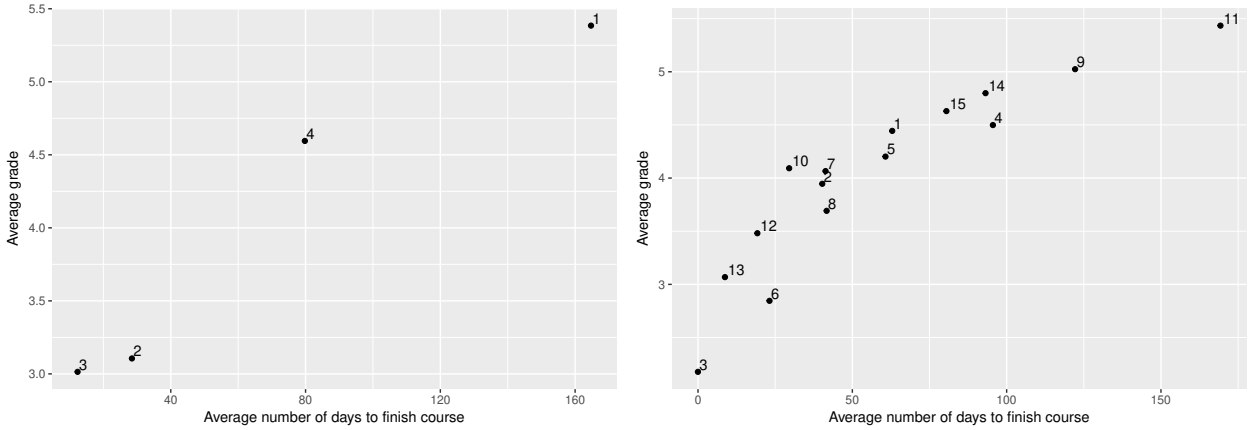


Figure 6.5: Graph of averaged centroids for 4 clusters (left) and 15 clusters(right) (taken from dataset ACFULL)

6.4.2 Cluster centroids

In the k-means theory section, we explained that the algorithm finds a set of centroids. In our case, we use thirty features and so the data space has thirty dimensions. To visualise the high number of dimensions, we have taken the advantage of the fact, that we use only two types of features. We grouped values of the same type and made an average of them. As a result, we obtained only two values for each cluster and these values can be easily displayed in a form of a 2D graph. An example is shown in Figure @ref(fig: centres), where on the x axis is average grade and on the y axis is an average number of days needed to finish a course. Both averages are calculated from data of all students in a cluster. It is not directly visible for the graph with few clusters (left part of the figure), but the relation between the average grade and the average number of days to finish a course is exponential (right part of the figure). The longer it takes to finish the courses, the worse grades the students obtain.

6.4.3 Using postcode information

Since each student is assigned a postcode in the dataset, we are able to map the postcodes to the regions of Czech Republic. The first view divides the students into regions and then draws a proportion of students coming from the cluster to the number of all students from the region. The plot has on x axis regions of Czech Republic and on y axis probabilities of being from a cluster while being from a region (Figure 6.6). The plot contains *NOT DEFINED* class, which concentrates foreign students and students, who cannot be assigned a region due to missing data.

6.4.4 Using gender of students

For another analysis of the clusters, we have used the gender of the students and checked, how gender influences the division of students into clusters. Students in clusters are grouped by the gender and probability of ending up in a cluster based on a gender is calculated. The probabilities are shown using the graph in Figure 6.7. Both genders are represented by one column, one stripe of the column represents one cluster and its size demonstrates the proportion to the total number of students of the gender. Text in a stripe contains the number of students in a cluster and proportion to the size of a cluster.

Visual analysis of the graph suggests that there is no statistical difference between clusters of males and females. To support the conclusion, we applied χ^2 test, which is typically used to determine whether there is a significant difference between expected and observed frequencies in data (Pearson 1900). The two key assumptions of the test are that the data is independent and there are more than five samples in each category.

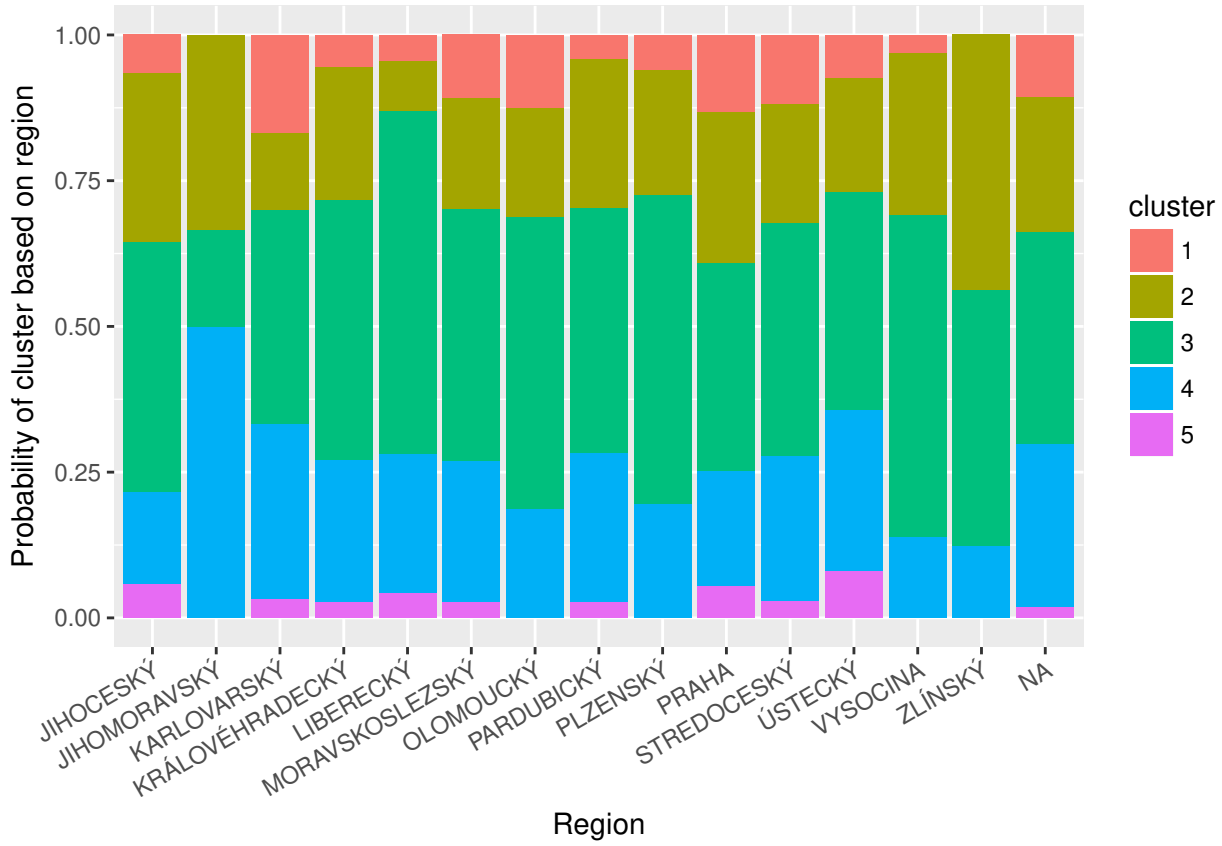


Figure 6.6: Probability of assigning a student into a cluster based on the region

Table 6.2: Results of Chi squared test for various number of clusters.

Cluster	2	3	4	5	6	7	8	9	10	11	12	13	14	15
X-squared	1.42	3.02	2.97	7.28	7.94	7.63	8.60	8.09	9.16	9.19	10.13	8.27	11.78	13.83
p-value	0.23	0.22	0.40	0.12	0.16	0.27	0.28	0.42	0.42	0.51	0.52	0.76	0.55	0.46

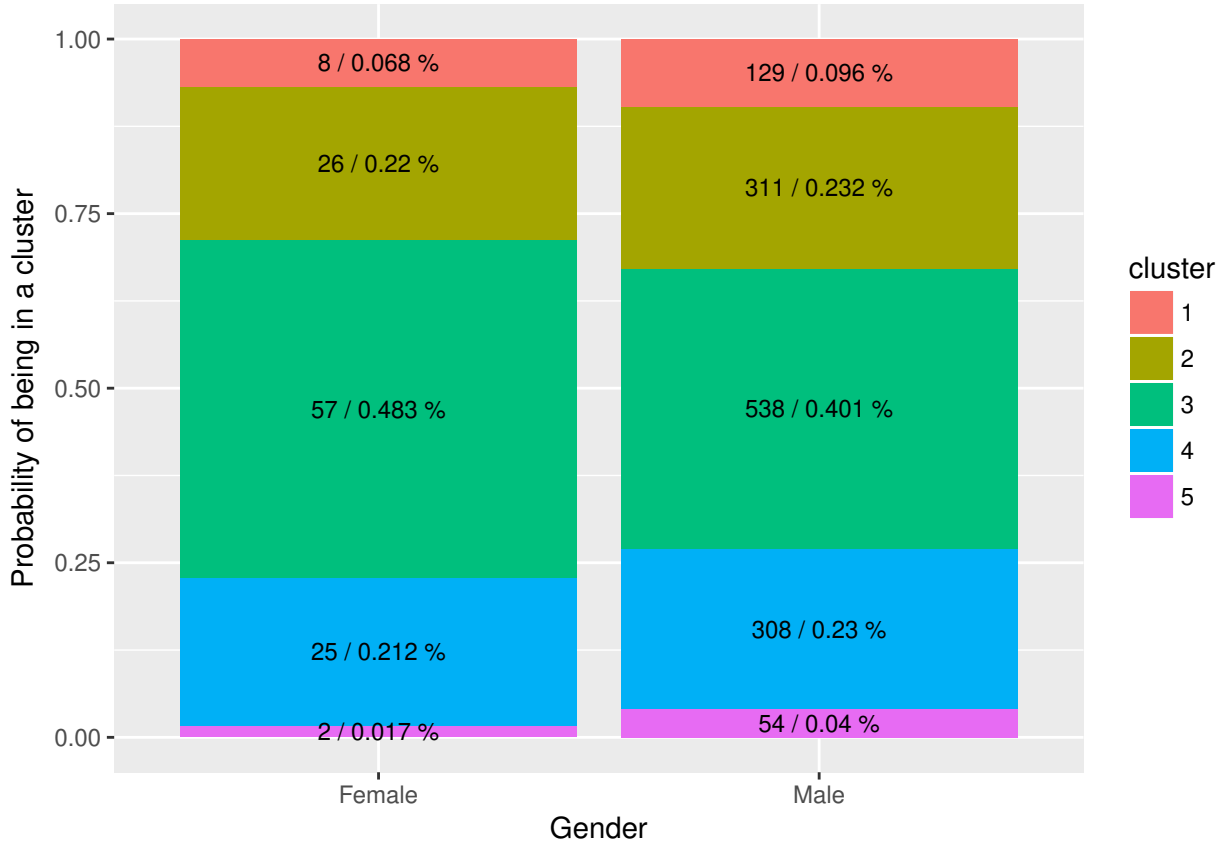


Figure 6.7: Relative proportion of students in clusters based on gender (taken from ACFULL dataset)

We are aware of the fact that the number of females in the fifth cluster is lower than the minimum. On the other hand, it never happens in more than 20% of clusters, which is another criteria for the assumption. Thus the test can be performed.

We test the null hypothesis that the division of students into clusters is independent of their gender against the alternative hypothesis that the division depends on the gender. We used the χ^2 test on results of clustering for $k = 2..15$ and results are shown in Table 6.2. Since all the p-values are greater than 0.05, we cannot reject the null hypothesis at significance level 5%.

We work with data coming from the FME, which is technical faculty hence the conclusion is surprising. We think that the comparison may be highly motivating for females as it shows that they can perform as well as males (but also as poor). Nevertheless, the fact is that the number of females is still ten times lower than the number of males in our data.

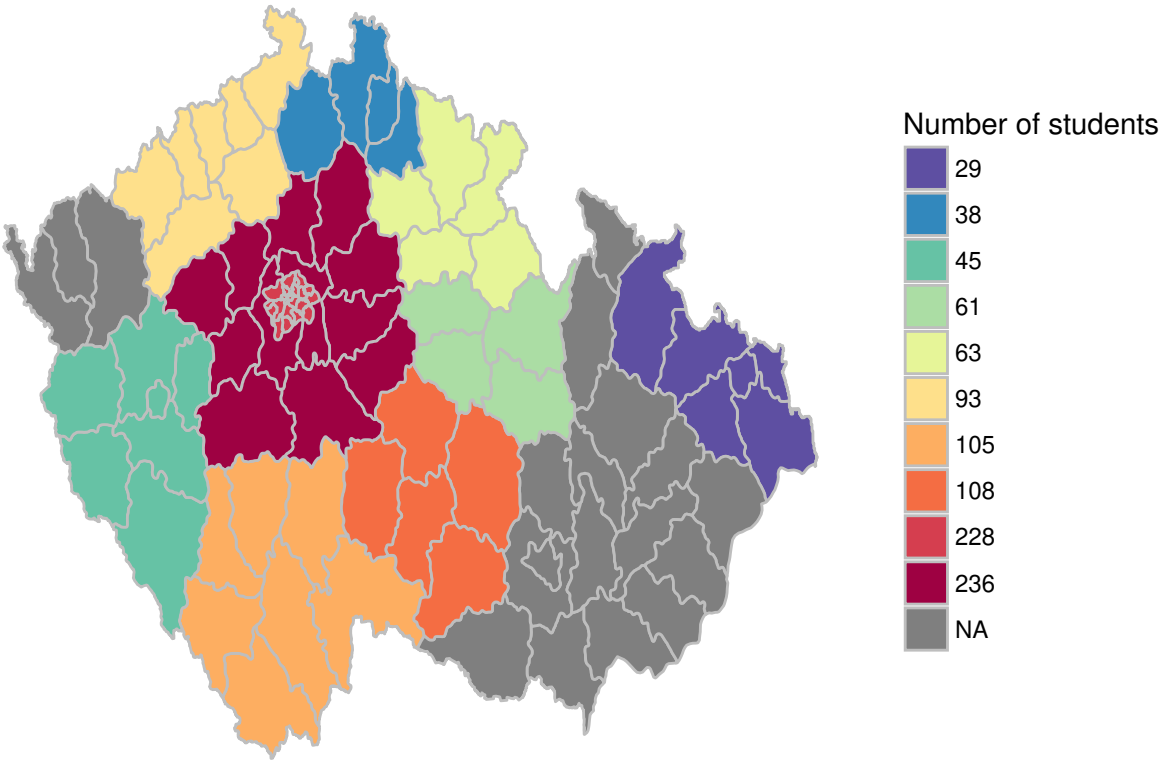


Figure 6.8: Number of SOK students divided into regions of Czech republic

6.5 Division of performance classes in regions

To provide a more visually interesting view on the data, we also created a view, which maps students based on their performance class to the regions (6.8). Each region is filled with colours which correspond to a number of students coming from the region and having the selected performance class. Grey colour (*NA* class) represents regions with less than five students.

6.6 Classification

Methods in the previous section show that patterns in the students' performance can be found. In this section, we apply classification algorithms on the datasets to confirm, that such patterns can be used for prediction of results on unseen data. For this purpose, we chose to apply Naive Bayes classifier, k-nearest neighbours algorithm and CART. The algorithms have been applied to predict results in an academic year - on AC datasets and results in a single course - MECH2016 dataset.

6.6.1 Classification of AC datasets

Classification on data from whole academic year aims at a prediction of final performance class based on a number of credits. We made an simplification of performance classes. Students from FF and NF groups are considered as *fail* students and the rest (OK, PS and SOK) of the students are considered *pass*. As the only feature, we have selected a number of credits earned up to a specific week.

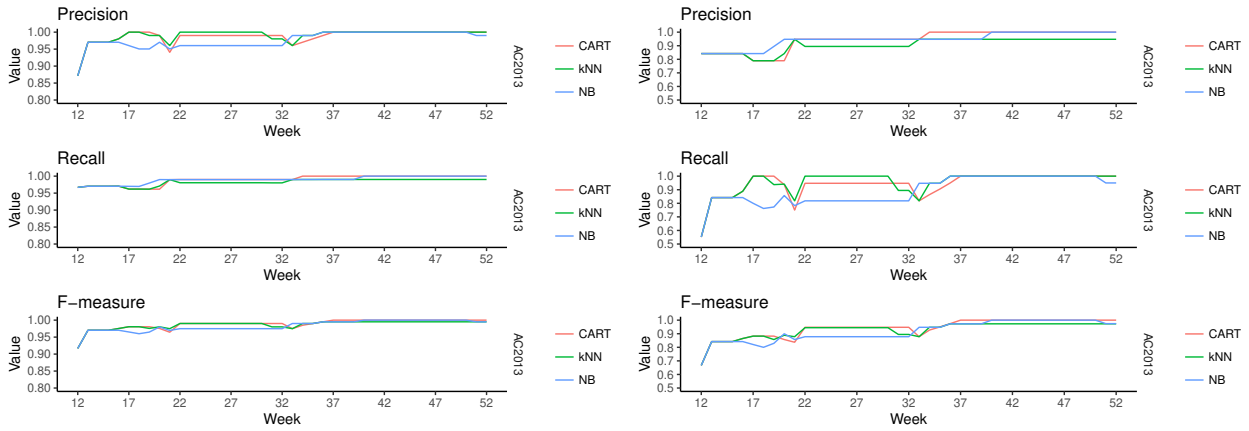


Figure 6.9: F-measure for pass class (left) and fail class (right)

On the simplified datasets, we applied all the chosen algorithm for each week of an academic year and classified the students into the *fail* and *pass* classes. 70% of the data in a dataset are taken as training data for the training of the classification model and the rest of the data is used for validation purposes. 10-fold cross-validation has been used to select parameters of model learning. The classification model is validated using the testing dataset and precision, recall and F-measure properties of validation of the models are calculated.

Firstly, we analysed the performance of the classificators in order to choose the one with the best performance. Figure 6.9 shows graphs with the progress of precision, recall and F-measure on AC2013. Left graphs display the measures for the *pass* class and right graphs the *fail* class. All graphs start from a twelveth week (x axis) as from that moment students started to get credits hence the classification can be used, values of the metrics are depicted on the y axis. Appendix C contains values of the precision, recall and F-measure for all AC dataset and the rest of the graphs with the progress of the values.

Look at the Figure 6.9 suggests that Naive Bayes performs slightly worse than the other two methods especially due to lower recall values on *fail* class in weeks 16 to 32. On the other hand, results of classification using other datasets has different outcomes. For example, it seems that Naive Bayes applied on the AC2015 has a little bit better results compared to kNN and CART. In any case, even if a classificator has worse performance compared to the others they all perform well. F-measure in any of the datasets does not drop under 94% for *pass* class and 79% for *fail* class after week 17, which is typically the first or second week of the winter exam period (Table 6.3). An important finding is that classification on ACFULL dataset (combination of data from four consecutive academic years) performs with similar results and in some cases better results than classification on a single academic year. Patterns in student results remain across academic years.

In order to support the last conclusion, we have performed another set of tests. The first run of the classifications used one dataset to create training and validation sets of data. At this moment, we have used one dataset for the training phase of the model and validated the model on another AC dataset. Four cases we created:

- AC2013 for training, AC2014 for validation
- AC2014 for training, AC2015 for validation
- AC2015 for training, AC2016 for validation
- AC2013, AC2014, AC2015 for training, AC2016 for validation

Results of the second set of classifications are demonstrated in Appendix D and Table 6.4. Similarly, as in the case of the first run, we cannot choose only one classificator based on the performance. Additionally, the differences between them are even smaller. It is also important that the performance measures have similar

Table 6.3: Minimum values of F-measure of classification on AC datasets after week 17.

Method	Dataset	Min pass	Min fail
CART	AC2016	94.3 %	79.25 %
kNN	AC2016	94.3 %	79.25 %
CART	AC2014	96.23 %	78.95 %
NB	AC2014	96.23 %	80 %
NB	AC2016	96.97 %	87.5 %
NB	ACFULL	97.09 %	85.54 %
kNN	AC2014	97.2 %	82.35 %
CART	AC2013	97.49 %	87.8 %
kNN	AC2013	97.49 %	87.8 %
NB	AC2013	97.49 %	87.8 %
NB	AC2015	97.61 %	89.36 %
CART	ACFULL	97.85 %	88.16 %
kNN	ACFULL	97.86 %	87.84 %
CART	AC2015	98.11 %	90.91 %
kNN	AC2015	98.58 %	93.33 %

Table 6.4: Minimum values of F-measure of classification on AC datasets after week 17.

Method	Dataset	Min pass	Min fail
CART	AC2016 on AC2015	95.4 %	82.76 %
kNN	AC2016 on AC2015	96.08 %	83.95 %
NB	AC2016 on AC2015	96.34 %	82.52 %
NB	AC2016 on AC2013-AC2015	96.42 %	83.33 %
CART	AC2016 on AC2013-AC2015	96.49 %	82.96 %
kNN	AC2016 on AC2013-AC2015	96.53 %	82.09 %
NB	AC2014 on AC2013	96.7 %	84.35 %
CART	AC2014 on AC2013	97.87 %	89.21 %
kNN	AC2014 on AC2013	97.87 %	89.21 %
NB	AC2015 on AC2014	98.08 %	89.06 %
CART	AC2015 on AC2014	98.22 %	89.92 %
kNN	AC2015 on AC2014	98.22 %	89.92 %

values as in the first run. Above that, the classification performance seems to be more constant across weeks as the graphs of precision, recall and F-measure are smoother.

Both types of tests show, that it is possible to predict results of students in the academic year early in the academic year. Already in first weeks of the winter exam period, the models perform with very high F-measure values - from 94% in case of classification to *pass* class and from 79% in case of classification to *fail* class. The fact, that the classification has been made in recent phase of the academic year, opens a space for the FME staff to intervene with the students and help them improve their performance. Further analysis of the differences between the classification in single week and differences in performance between different weeks may discover another pattern.

6.6.2 Classification of MECH2016 dataset

Classification of the final result in a single course has been made on the MECH2016 dataset. We have worked with the attendance of the students on seminars and their test results. For the classification of the MECH2016 dataset, we reduced the final results into only two classes, similarly as in case of the AC datasets. Students, who finished the course with a grade better than E are considered to be from *pass* group and the rest from the *fail* group. Next, the selected classification algorithms have been applied on the simplified MECH2016 dataset.

The classification has been performed for each week of the semester. Data for a week has been created so that attendance and results available up until the selected week have been considered. It means that in each week the number of features has changed. We have taken 80% of the data to train the classification model and the rest as validation of the resulting model. The increased proportion of training data (compared to the classification of AC datasets) comes from the fact that there is less data.

Figure 6.10 shows two graphs, which sum up the performance of the classifications. The top part shows classification of *pass* class and in the bottom classification of *fail* class. Both show accuracy, F-measure, recall and accuracy each in a single part of the graph. On the x axis are weeks of the semester, on the y axis are values of each of the measures and dashed lines demonstrate weeks in which tests were written.

Performance of classification into *pass* class is satisfying for all the classificators. F-measure has the lowest value 78% in week 6 and the highest value 88% in week 9. On the other hand, classification of *fail* class shows much worse results. Especially in case of the kNN classifier, which has F-measure at very low values even in later parts of the semester (approximately 25% from week 10 and later). The CART and Naive Bayes classifiers perform more or less same (between 50% and 70%).

The low value of F-measure is often caused by a low value of precision. However, in case of the classification of *fail* class, it does not have to be necessary as a big problem as in case of the *pass* classification. If an intervention is performed on a student incorrectly classified as failing, the result should be that the student passes the course, which is the expected result of the intervention. The fact is, that the student would pass even without the intervention. The conclusion supposes that the intervention is made in a way, that the student is not demotivated by the intervention.

6.6.2.1 CART model

One advantage of the CART classifier over the others is that the final model can be visualized using the decision tree, which can be further analysed. Figure 6.11 shows models used for the classification of the MECH2016 dataset in weeks 6 and 13. We can see that the model classifies based on the result of the first test in week 6 (label *tst1*) and attendance at the seminar in the sixth week (label *cv6*). The model in week 13 considers all tests and no presence at seminars. It is correct as the final grade depends on the sum of point obtained from the tests.

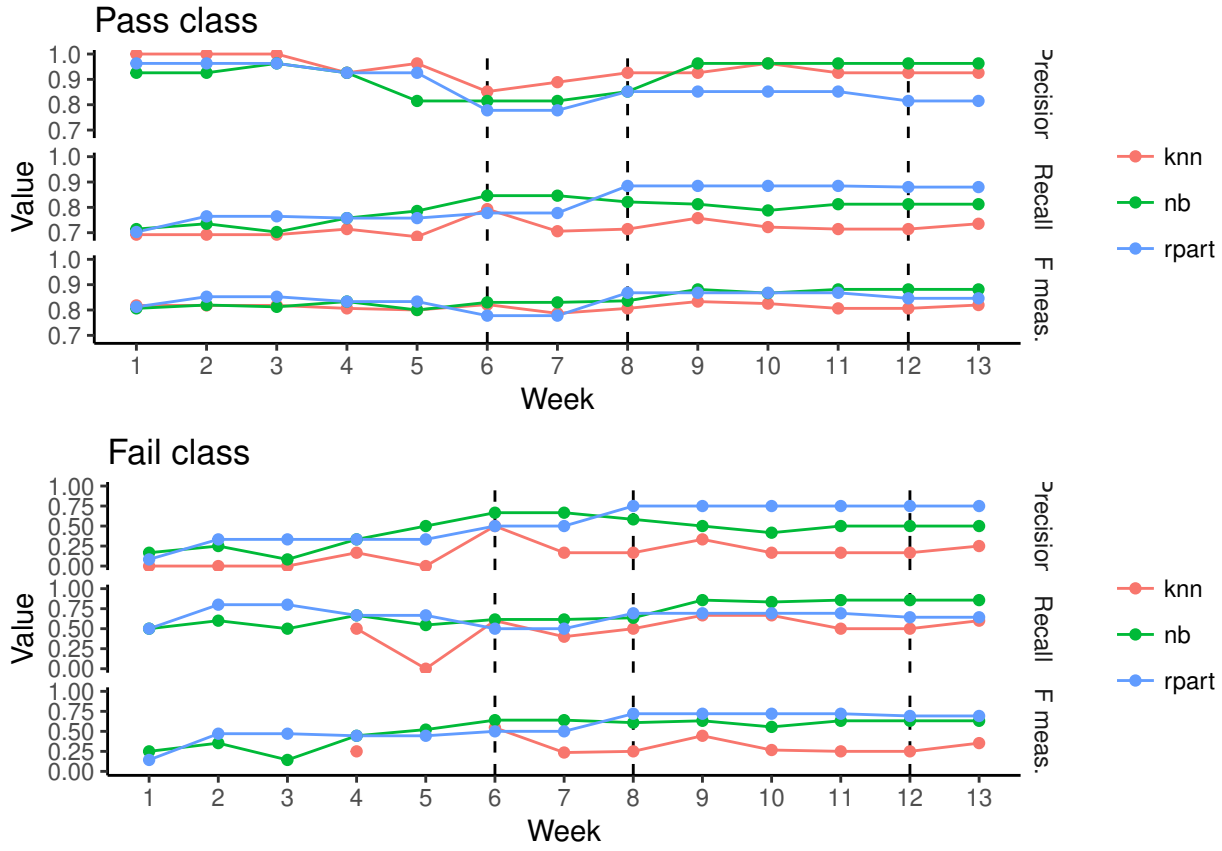


Figure 6.10: Classification of pass class (top part) and fail class (bottom part).

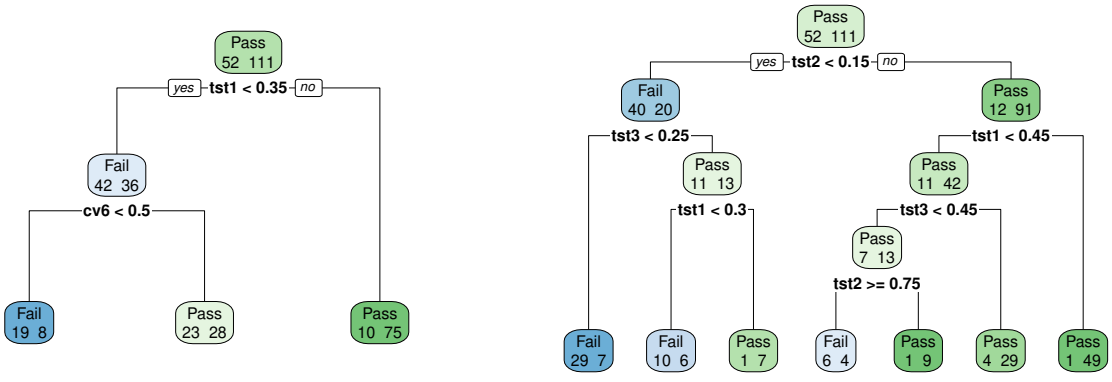


Figure 6.11: Decision trees created from CART classification models.

Chapter 7

Analyst

In the previous chapter, we have used various analytical methods. As a part of these methods, there is a lot of graphs through which the outcomes may be presented. In this chapter, we introduce Learning Analytics dashboard - *Analyst*, which has been implemented to provide the results to the FME staff in a more convenient way. The goal of the dashboard is to enable the users to apply the analytical methods on selected data and discuss the results. The dashboard is still in its development phase and so not all the methods from the previous chapter are available. However, we plan to provide the user with all the methods we have shown in the previous chapter. In the following sections, we introduce the application's architecture and functionalities.

7.1 Architecture overview

The Analyst has been implemented as a web application using the Shiny technology and runs on the same infrastructure as the Seminarist application. It means that the application is hidden behind the reverse proxy and cannot be accessed without successful login. The Analyst uses MySQL database, where all the available datasets are stored. User roles are not distinguished (as in case of the Seminarist), which simplifies the database model (Appendix E). Additionally, we have aggregated the available data so that we can work with them more easily. Since we use the Shiny technology, we can implement both front-end and back-end directly from R (Section 5.3.3).

All graph components of the application are created using the ggplot2 and converted using the plotly to make all the graphs in the application interactive.

7.2 The application

The user interface of the application is divided into three parts (Figure 7.1):

- Upper part provides a filter, which can be used to select a dataset, which is used as input to the analytical methods. It can be used to further filter the data by gender, a form of study, course type or first study year. In case the first study year filter is turned off, students repeating a course from the set of first-year courses are included in the analysis.
- Left part contains a navigation menu, which provides the user with an option to switch between analytical and administration tools presented later.
- Central part shows content based on the selection in the left part navigation menu.

Figure 7.1: Page layout of the Analyst application

Until this moment, two analytical methods from the previous chapter, summary statistics and grade histogram have been added to the dashboard.

The first method is from the section 6.2, where the average number of ECTS credits of students divided by performance class has been calculated for each week of an academic year. The graph is shown in the central part of Figure 7.1.

The second method provides results of the analysis presented in section 6.3. The method takes the input data and based on a selected moment of an academic year calculates the probability of a different performance class depending on the number of credits earned up until the moment. In the section 6.3 the moment was set to the start of the winter exam period. However, the same can be done for any moment of the academic year. In the application, the method is extended so that the user can select a week of an academic year, probabilities are calculated based on the selected week and graph (Figure 6.2) is created. Moreover, the application can automatically iterate over the weeks, which makes from the set of graph an animation.

Another view is available in order to provide a summary of the data, which the user has filtered. The view (Figure 7.2) is made of two parts. The first component provides a division of students into performance classes. Each class is represented by one box, which shows a number of students in the class and percentage with respect to the whole dataset. The second component shows a graph of proportions of grades in all courses of the filtered dataset. The courses are displayed in the x axis and probability of a grade is shown on the y axis.

Besides the analytical tools, the navigation menu contains links to administration views. These views can be used to upload files containing data about students and their performance in courses. It is also possible to modify these data and set few parameters of the application, for example, define important dates of an academic year as the dates differ from year to year or define the set of courses, which define a first-year students (Figure 7.3).

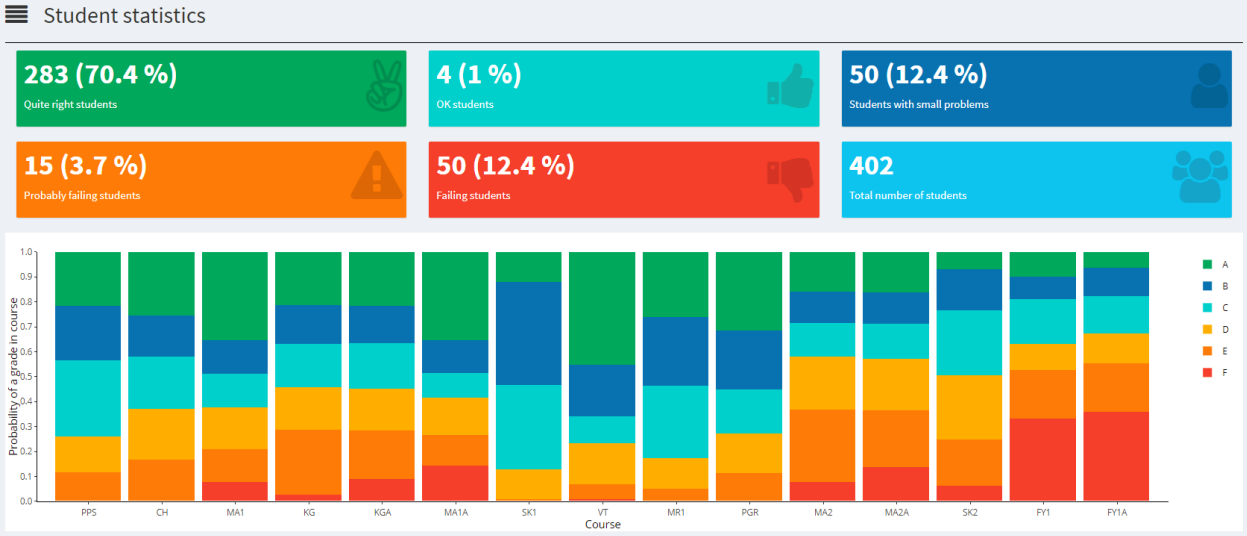


Figure 7.2: Summaries view

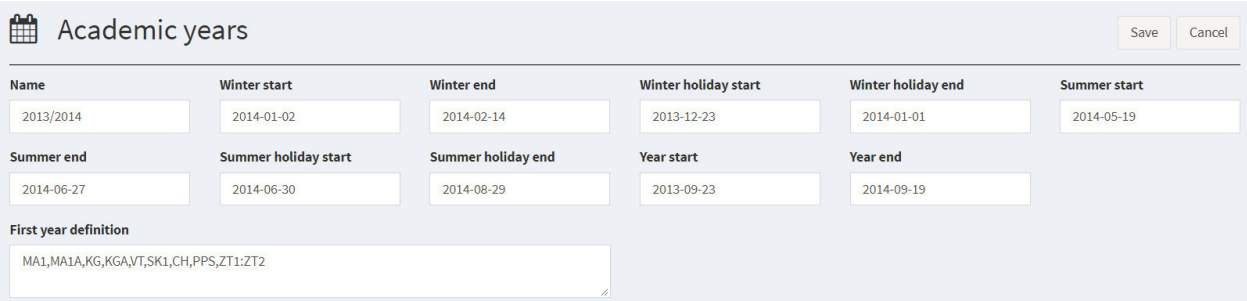


Figure 7.3: Form used to change parameteres of an academic year

Chapter 8

Conclusion

In past years, student retention is a problem of the universities. Methods from a field of LA address this problem and help to decrease the student drop-out. We have cooperated with FME, which faces the problem of low retention especially in the first-year of the degree. We collected student-related data from various sources, analysed them and provided multiple tools, which can be used for the improvement of student retention.

CTU university system KOS was used as the first source of data. It provides information about students, courses and results. The second source of the data was a web application *Seminarist*. We have implemented the application to provide a teaching staff with a comfortable way of collecting student attendance and results. The application uses responsive design in order to make the application available for various types of devices.

For the purpose of the analysis, a database was created. It gathers data from the KOS system and the *Seminarist* application. Using the data from the database, we applied several machine learning methods to reveal patterns in student data and created predictive models, which can be used for identification of students at risk of failing studies. Predictive models were created for available datasets using Naive Bayes classifier, kNN classifier and CART. Three testing scenarios were tested. Firstly a subset of data from one academic year was used for the training of a model and the rest of the data from the same dataset validated the model. Secondly, the whole dataset was used for the training phase and another dataset was used for validation of the model. Lastly, classification models were trained on data for a single course.

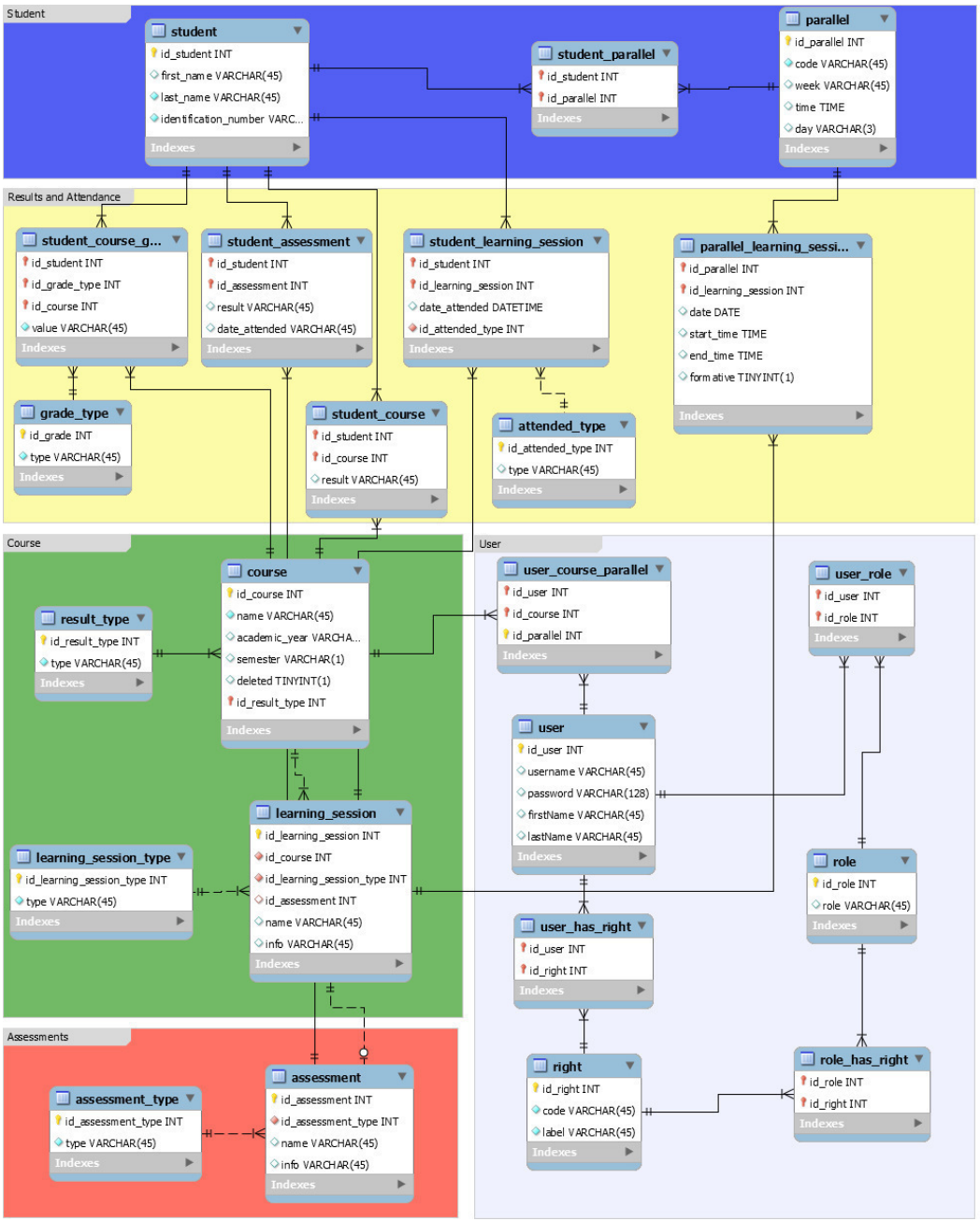
The analysis shows that students with no credits at the beginning of the winter exam period fails at the end of an academic year. We performed clustering of the students by k-means. Results of the clustering were analysed and discussed. An important outcome is that gender of the students does not influence the division into clusters. The most important finding is that we can predict results of students in the whole academic year in an early stage of the year. Already in the first week of winter exam period, the F-measure for successful students is around 95% and around 80% for failing students. Prediction of results in a single course is possible, however, F-measure reaches maximally 70% for *fail* class and 88% for *pass* class.

Furthermore, web application *Analyst* was created to provide outcomes of the analysis in a comfortable way. An infrastructure for the web applications *Seminarist* and *Analyst* was designed and constructed, including the databases, web servers and security configuration.

At this moment the *Analyst* does not provide all the methods, we have used during the analysis. However, it is planned to add all the features. Moreover, we would like to examine the possibilities of applying Markov chains on the available data and also investigate the influence of selecting different features for the predictive modelling. During the clustering, the group of students with no credits at the end of the winter exam period was identified. Further analysis of these students may uncover patterns from the data, which may be used to increase the retention.

Appendix A

Seminarist database model



Appendix B

Feature analysis box plots

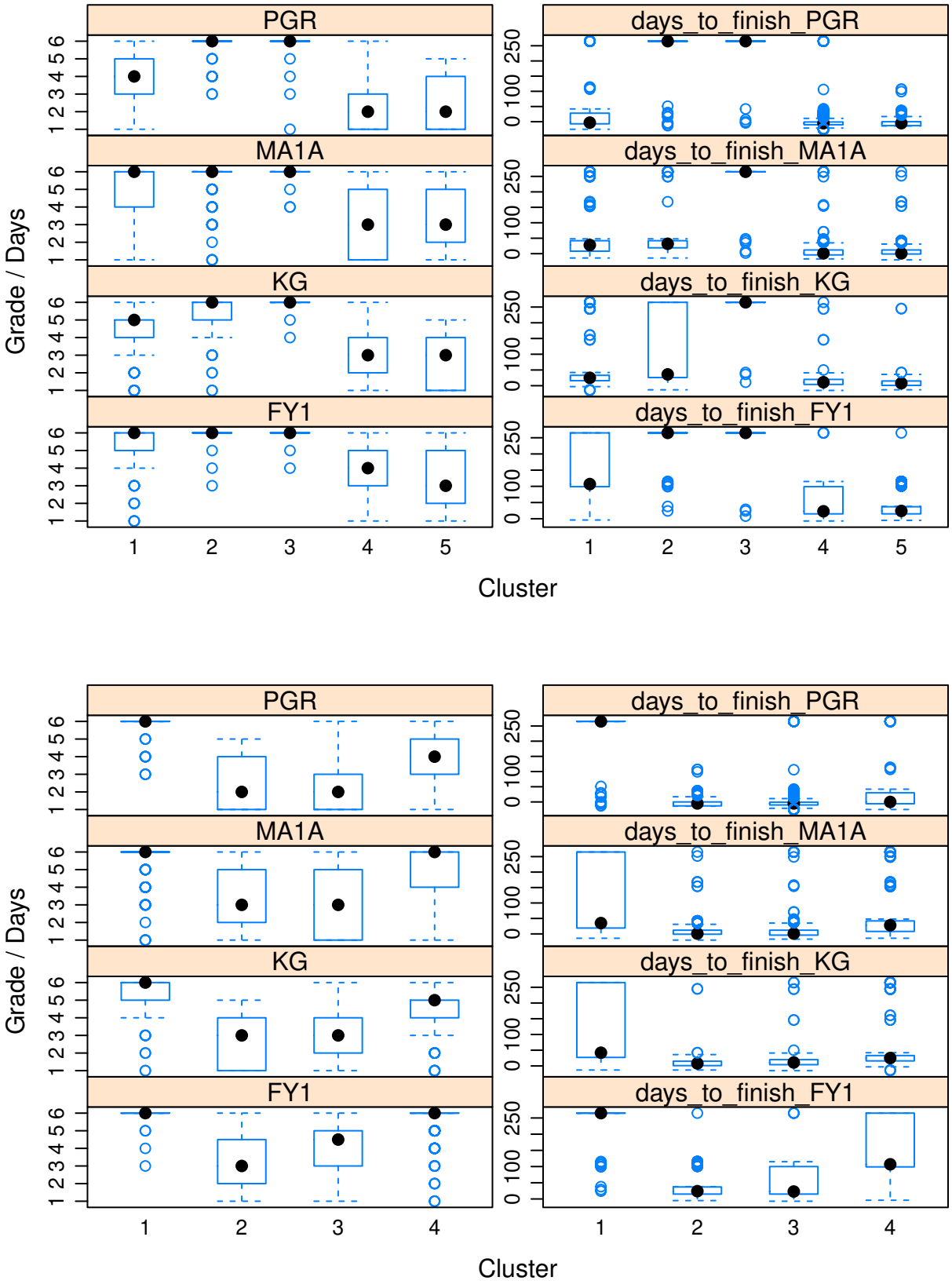


Figure B.1: Box plot of features. Datasets including passive withdrawals in the top. Dataset without them in the bottom.

Appendix C

Quality measures of classification on AC datasets

Table C.1: Precision values from classification on AC2013 dataset for pass and fail classes in the first six columns. (excluding week). Recall values are in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.87	0.87	0.87	0.84	0.84	0.84	0.97	0.97	0.97	0.55	0.55	0.55
13	0.97	0.97	0.97	0.84	0.84	0.84	0.97	0.97	0.97	0.84	0.84	0.84
14	0.97	0.97	0.97	0.84	0.84	0.84	0.97	0.97	0.97	0.84	0.84	0.84
15	0.97	0.97	0.97	0.84	0.84	0.84	0.97	0.97	0.97	0.84	0.84	0.84
16	0.98	0.98	0.97	0.84	0.84	0.84	0.97	0.97	0.97	0.89	0.89	0.84
17	1	1	0.96	0.79	0.79	0.84	0.96	0.96	0.97	1	1	0.8
18	1	1	0.95	0.79	0.79	0.84	0.96	0.96	0.97	1	1	0.76
19	1	0.99	0.95	0.79	0.79	0.89	0.96	0.96	0.98	1	0.94	0.77
20	0.99	0.99	0.97	0.79	0.84	0.95	0.96	0.97	0.99	0.94	0.94	0.86
21	0.94	0.96	0.95	0.95	0.95	0.95	0.99	0.99	0.99	0.75	0.82	0.78
22	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
23	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
24	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
25	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
26	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
27	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
28	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
29	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
30	0.99	1	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	1	0.82
31	0.99	0.98	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	0.89	0.82
32	0.99	0.98	0.96	0.95	0.89	0.95	0.99	0.98	0.99	0.95	0.89	0.82
33	0.96	0.96	0.99	0.95	0.95	0.95	0.99	0.99	0.99	0.82	0.82	0.95
34	0.97	0.99	0.99	1	0.95	0.95	1	0.99	0.99	0.86	0.95	0.95
35	0.98	0.99	0.99	1	0.95	0.95	1	0.99	0.99	0.9	0.95	0.95
36	0.99	1	1	1	0.95	0.95	1	0.99	0.99	0.95	1	1
37	1	1	1	1	0.95	0.95	1	0.99	0.99	1	1	1
38	1	1	1	1	0.95	0.95	1	0.99	0.99	1	1	1
39	1	1	1	1	0.95	0.95	1	0.99	0.99	1	1	1
40	1	1	1	1	0.95	1	1	0.99	1	1	1	1
41	1	1	1	1	0.95	1	1	0.99	1	1	1	1
42	1	1	1	1	0.95	1	1	0.99	1	1	1	1
43	1	1	1	1	0.95	1	1	0.99	1	1	1	1
44	1	1	1	1	0.95	1	1	0.99	1	1	1	1
45	1	1	1	1	0.95	1	1	0.99	1	1	1	1
46	1	1	1	1	0.95	1	1	0.99	1	1	1	1
47	1	1	1	1	0.95	1	1	0.99	1	1	1	1
48	1	1	1	1	0.95	1	1	0.99	1	1	1	1
49	1	1	1	1	0.95	1	1	0.99	1	1	1	1
50	1	1	1	1	0.95	1	1	0.99	1	1	1	1
51	1	1	0.99	1	0.95	1	1	0.99	1	1	1	0.95
52	1	1	0.99	1	0.95	1	1	0.99	1	1	1	0.95

Table C.2: F-measure values from classification on the AC2013 dataset for pass and fail classes.

week	CART	kNN	NB	CART	kNN	NB
12	0.92	0.92	0.92	0.67	0.67	0.67
13	0.97	0.97	0.97	0.84	0.84	0.84
14	0.97	0.97	0.97	0.84	0.84	0.84
15	0.97	0.97	0.97	0.84	0.84	0.84
16	0.98	0.98	0.97	0.86	0.86	0.84
17	0.98	0.98	0.97	0.88	0.88	0.82
18	0.98	0.98	0.96	0.88	0.88	0.8
19	0.98	0.98	0.96	0.88	0.86	0.83
20	0.98	0.98	0.98	0.86	0.89	0.9
21	0.96	0.97	0.97	0.84	0.88	0.86
22	0.99	0.99	0.97	0.95	0.94	0.88
23	0.99	0.99	0.97	0.95	0.94	0.88
24	0.99	0.99	0.97	0.95	0.94	0.88
25	0.99	0.99	0.97	0.95	0.94	0.88
26	0.99	0.99	0.97	0.95	0.94	0.88
27	0.99	0.99	0.97	0.95	0.94	0.88
28	0.99	0.99	0.97	0.95	0.94	0.88
29	0.99	0.99	0.97	0.95	0.94	0.88
30	0.99	0.99	0.97	0.95	0.94	0.88
31	0.99	0.98	0.97	0.95	0.89	0.88
32	0.99	0.98	0.97	0.95	0.89	0.88
33	0.97	0.97	0.99	0.88	0.88	0.95
34	0.98	0.99	0.99	0.93	0.95	0.95
35	0.99	0.99	0.99	0.95	0.95	0.95
36	1	1	1	0.97	0.97	0.97
37	1	1	1	1	0.97	0.97
38	1	1	1	1	0.97	0.97
39	1	1	1	1	0.97	0.97
40	1	1	1	1	0.97	1
41	1	1	1	1	0.97	1
42	1	1	1	1	0.97	1
43	1	1	1	1	0.97	1
44	1	1	1	1	0.97	1
45	1	1	1	1	0.97	1
46	1	1	1	1	0.97	1
47	1	1	1	1	0.97	1
48	1	1	1	1	0.97	1
49	1	1	1	1	0.97	1
50	1	1	1	1	0.97	1
51	1	1	1	1	0.97	0.97
52	1	1	1	1	0.97	0.97

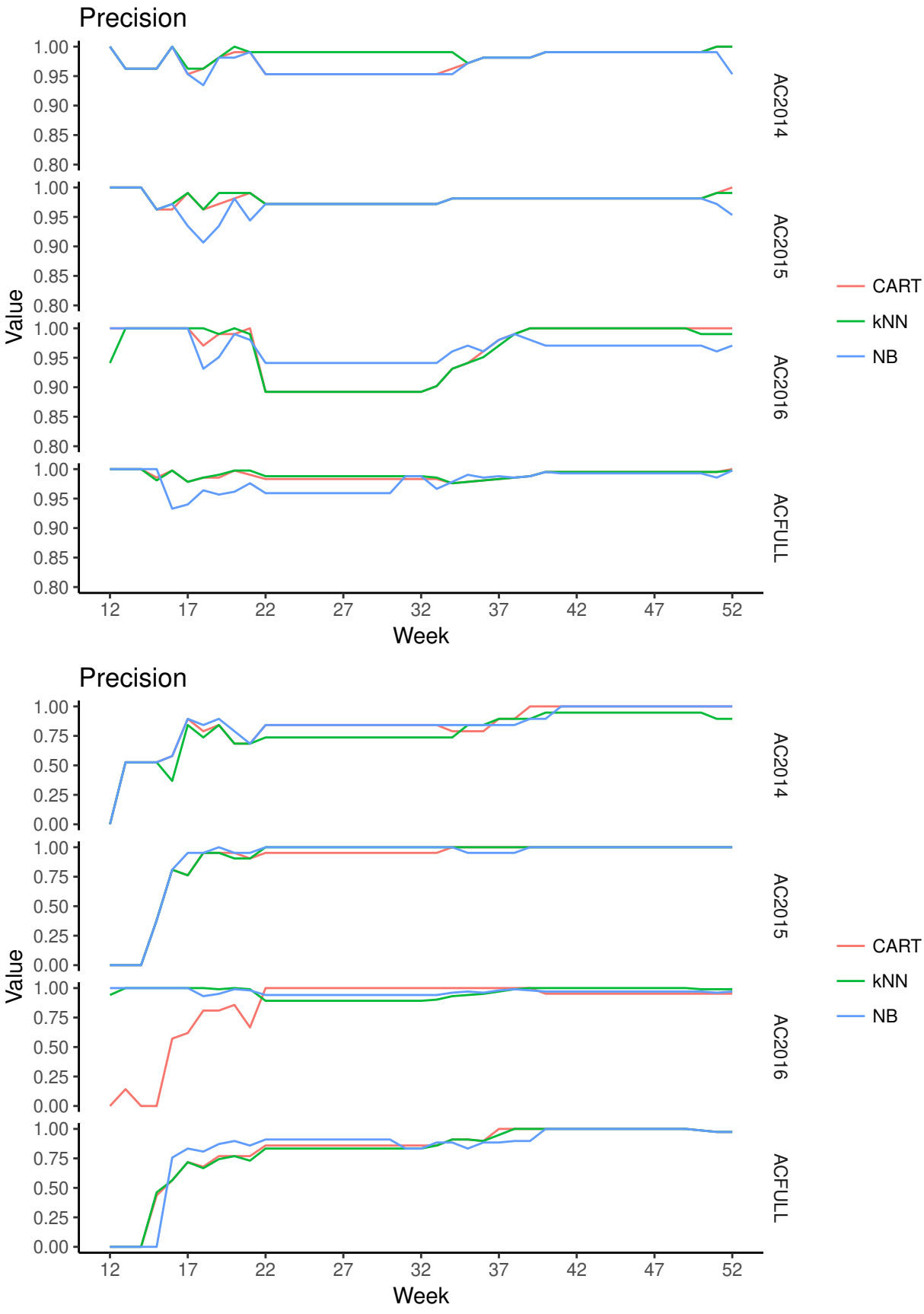


Figure C.1: Precision for pass class (top) and fail class (bottom)

Table C.3: Precision values from classification on the AC2014 dataset for pass and fail classes are in the first six columns (excluding week). Similarly, values obtained from classification on AC2015 dataset are in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	1	1	1	0	0	0	1	1	1	0	0	0
13	0.96	0.96	0.96	0.53	0.53	0.53	1	1	1	0	0	0
14	0.96	0.96	0.96	0.53	0.53	0.53	1	1	1	0	0	0
15	0.96	0.96	0.96	0.53	0.53	0.53	0.96	0.96	0.96	0.38	0.38	0.38
16	1	1	1	0.58	0.37	0.58	0.96	0.97	0.97	0.81	0.81	0.81
17	0.95	0.96	0.95	0.89	0.84	0.89	0.99	0.99	0.93	0.76	0.76	0.95
18	0.96	0.96	0.93	0.79	0.74	0.84	0.96	0.96	0.91	0.95	0.95	0.95
19	0.98	0.98	0.98	0.84	0.84	0.89	0.97	0.99	0.93	0.95	0.95	1
20	0.99	1	0.98	0.68	0.68	0.79	0.98	0.99	0.98	0.95	0.9	0.95
21	0.99	0.99	0.99	0.68	0.68	0.68	0.99	0.99	0.94	0.9	0.9	0.95
22	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
23	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
24	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
25	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
26	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
27	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
28	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
29	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
30	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
31	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
32	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
33	0.95	0.99	0.95	0.84	0.74	0.84	0.97	0.97	0.97	0.95	1	1
34	0.96	0.99	0.95	0.79	0.74	0.84	0.98	0.98	0.98	1	1	1
35	0.97	0.97	0.97	0.79	0.84	0.84	0.98	0.98	0.98	1	1	0.95
36	0.98	0.98	0.98	0.79	0.84	0.84	0.98	0.98	0.98	1	1	0.95
37	0.98	0.98	0.98	0.89	0.89	0.84	0.98	0.98	0.98	1	1	0.95
38	0.98	0.98	0.98	0.89	0.89	0.84	0.98	0.98	0.98	1	1	0.95
39	0.98	0.98	0.98	1	0.89	0.89	0.98	0.98	0.98	1	1	1
40	0.99	0.99	0.99	1	0.95	0.89	0.98	0.98	0.98	1	1	1
41	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
42	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
43	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
44	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
45	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
46	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
47	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
48	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
49	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
50	0.99	0.99	0.99	1	0.95	1	0.98	0.98	0.98	1	1	1
51	1	1	0.99	1	0.89	1	0.99	0.99	0.97	1	1	1
52	1	1	0.95	1	0.89	1	1	0.99	0.95	1	1	1

Table C.4: Precision values from classification on the AC2016 dataset for pass and fail classes are in the first six columns (excluding week). Similarly, values obtained from classification on ACFULL dataset are in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	1	0.94	1	0	0.94	1	1	1	1	0	0	0
13	1	1	1	0.14	1	1	1	1	1	0	0	0
14	1	1	1	0	1	1	1	1	1	0	0	0
15	1	1	1	0	1	1	0.99	0.98	1	0.44	0.46	0
16	1	1	1	0.57	1	1	1	1	0.93	0.56	0.56	0.76
17	1	1	1	0.62	1	1	0.98	0.98	0.94	0.72	0.72	0.83
18	0.97	1	0.93	0.81	1	0.93	0.99	0.99	0.96	0.68	0.67	0.81
19	0.99	0.99	0.95	0.81	0.99	0.95	0.99	0.99	0.96	0.77	0.74	0.87
20	0.99	1	0.99	0.86	1	0.99	1	1	0.96	0.77	0.77	0.9
21	1	0.99	0.98	0.67	0.99	0.98	0.99	1	0.98	0.77	0.73	0.86
22	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
23	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
24	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
25	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
26	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
27	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
28	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
29	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
30	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.96	0.86	0.83	0.91
31	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.99	0.86	0.83	0.83
32	0.89	0.89	0.94	1	0.89	0.94	0.98	0.99	0.99	0.86	0.83	0.83
33	0.9	0.9	0.94	1	0.9	0.94	0.98	0.99	0.97	0.86	0.86	0.88
34	0.93	0.93	0.96	1	0.93	0.96	0.98	0.98	0.98	0.91	0.91	0.88
35	0.94	0.94	0.97	1	0.94	0.97	0.98	0.98	0.99	0.91	0.91	0.83
36	0.96	0.95	0.96	1	0.95	0.96	0.98	0.98	0.99	0.9	0.9	0.88
37	0.98	0.97	0.98	1	0.97	0.98	0.98	0.98	0.99	1	0.95	0.88
38	0.99	0.99	0.99	1	0.99	0.99	0.99	0.99	0.99	1	1	0.9
39	1	1	0.98	1	1	0.98	0.99	0.99	0.99	1	1	0.9
40	1	1	0.97	0.95	1	0.97	1	1	1	1	1	1
41	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
42	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
43	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
44	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
45	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
46	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
47	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
48	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
49	1	1	0.97	0.95	1	0.97	1	1	0.99	1	1	1
50	1	0.99	0.97	0.95	0.99	0.97	1	1	0.99	0.99	0.99	0.99
51	1	0.99	0.96	0.95	0.99	0.96	1	1	0.99	0.97	0.97	0.97
52	1	0.99	0.97	0.95	0.99	0.97	1	1	1	0.97	0.97	0.97

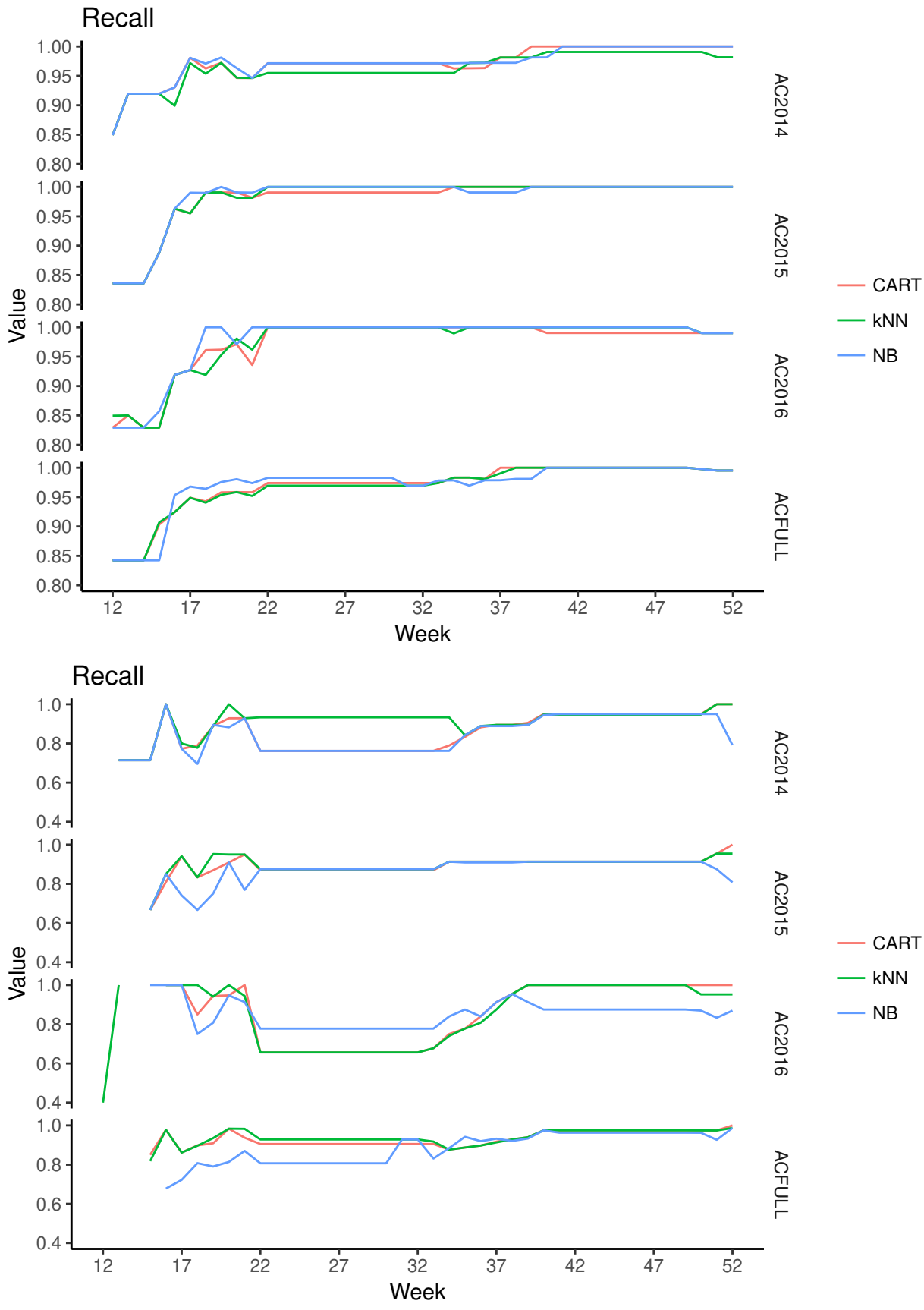


Figure C.2: Recall statictics on Pass class (top) and Fail class (bottom)

Table C.5: Recall values from classification on the AC2014 dataset for pass and fail classes are in the first six columns (excluding week). Similarly, values obtained from classification on AC2014 dataset are in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.85	0.85	0.85	NA	NA	NA	0.84	0.84	0.84	NA	NA	NA
13	0.92	0.92	0.92	0.71	0.71	0.71	0.84	0.84	0.84	NA	NA	NA
14	0.92	0.92	0.92	0.71	0.71	0.71	0.84	0.84	0.84	NA	NA	NA
15	0.92	0.92	0.92	0.71	0.71	0.71	0.89	0.89	0.89	0.67	0.67	0.67
16	0.93	0.9	0.93	1	1	1	0.96	0.96	0.96	0.81	0.85	0.85
17	0.98	0.97	0.98	0.77	0.8	0.77	0.95	0.95	0.99	0.94	0.94	0.74
18	0.96	0.95	0.97	0.79	0.78	0.7	0.99	0.99	0.99	0.83	0.83	0.67
19	0.97	0.97	0.98	0.89	0.89	0.89	0.99	0.99	1	0.87	0.95	0.75
20	0.95	0.95	0.96	0.93	1	0.88	0.99	0.98	0.99	0.91	0.95	0.91
21	0.95	0.95	0.95	0.93	0.93	0.93	0.98	0.98	0.99	0.95	0.95	0.77
22	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
23	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
24	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
25	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
26	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
27	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
28	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
29	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
30	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
31	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
32	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
33	0.97	0.95	0.97	0.76	0.93	0.76	0.99	1	1	0.87	0.88	0.88
34	0.96	0.95	0.97	0.79	0.93	0.76	1	1	1	0.91	0.91	0.91
35	0.96	0.97	0.97	0.83	0.84	0.84	1	1	0.99	0.91	0.91	0.91
36	0.96	0.97	0.97	0.88	0.89	0.89	1	1	0.99	0.91	0.91	0.91
37	0.98	0.98	0.97	0.89	0.89	0.89	1	1	0.99	0.91	0.91	0.91
38	0.98	0.98	0.97	0.89	0.89	0.89	1	1	0.99	0.91	0.91	0.91
39	1	0.98	0.98	0.9	0.89	0.89	1	1	1	0.91	0.91	0.91
40	1	0.99	0.98	0.95	0.95	0.94	1	1	1	0.91	0.91	0.91
41	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
42	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
43	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
44	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
45	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
46	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
47	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
48	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
49	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
50	1	0.99	1	0.95	0.95	0.95	1	1	1	0.91	0.91	0.91
51	1	0.98	1	1	1	0.95	1	1	1	0.95	0.95	0.88
52	1	0.98	1	1	1	0.79	1	1	1	1	0.95	0.81

Table C.6: Recall values from classification on the AC2016 dataset for pass and fail classes are in the first six columns (excluding week). Similarly, values obtained from classification on ACFULL dataset are in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.83	0.85	0.83	NA	0.4	NA	0.84	0.84	0.84	NA	NA	NA
13	0.85	0.85	0.83	1	1	NA	0.84	0.84	0.84	NA	NA	NA
14	0.83	0.83	0.83	NA	NA	NA	0.84	0.84	0.84	NA	NA	NA
15	0.83	0.83	0.86	NA	NA	1	0.9	0.91	0.84	0.85	0.82	NA
16	0.92	0.92	0.92	1	1	1	0.92	0.92	0.95	0.98	0.98	0.68
17	0.93	0.93	0.93	1	1	1	0.95	0.95	0.97	0.86	0.86	0.72
18	0.96	0.92	1	0.85	1	0.75	0.94	0.94	0.96	0.9	0.9	0.81
19	0.96	0.95	1	0.94	0.94	0.81	0.96	0.95	0.98	0.91	0.94	0.79
20	0.97	0.98	0.97	0.95	1	0.95	0.96	0.96	0.98	0.98	0.98	0.81
21	0.94	0.96	1	1	0.94	0.91	0.96	0.95	0.97	0.94	0.98	0.87
22	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
23	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
24	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
25	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
26	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
27	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
28	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
29	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
30	1	1	1	0.66	0.66	0.78	0.97	0.97	0.98	0.91	0.93	0.81
31	1	1	1	0.66	0.66	0.78	0.97	0.97	0.97	0.91	0.93	0.93
32	1	1	1	0.66	0.66	0.78	0.97	0.97	0.97	0.91	0.93	0.93
33	1	1	1	0.68	0.68	0.78	0.97	0.97	0.98	0.91	0.92	0.83
34	1	0.99	1	0.75	0.74	0.84	0.98	0.98	0.98	0.88	0.88	0.88
35	1	1	1	0.78	0.78	0.88	0.98	0.98	0.97	0.89	0.89	0.94
36	1	1	1	0.84	0.81	0.84	0.98	0.98	0.98	0.9	0.9	0.92
37	1	1	1	0.91	0.88	0.91	1	0.99	0.98	0.92	0.91	0.93
38	1	1	1	0.95	0.95	0.95	1	1	0.98	0.93	0.93	0.92
39	1	1	1	1	1	0.91	1	1	0.98	0.94	0.94	0.93
40	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.98
41	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
42	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
43	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
44	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
45	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
46	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
47	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
48	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
49	0.99	1	1	1	1	0.88	1	1	1	0.98	0.98	0.96
50	0.99	0.99	0.99	1	0.95	0.87	1	1	1	0.97	0.97	0.96
51	0.99	0.99	0.99	1	0.95	0.83	1	1	1	0.97	0.97	0.93
52	0.99	0.99	0.99	1	0.95	0.87	1	1	1	1	0.99	0.99

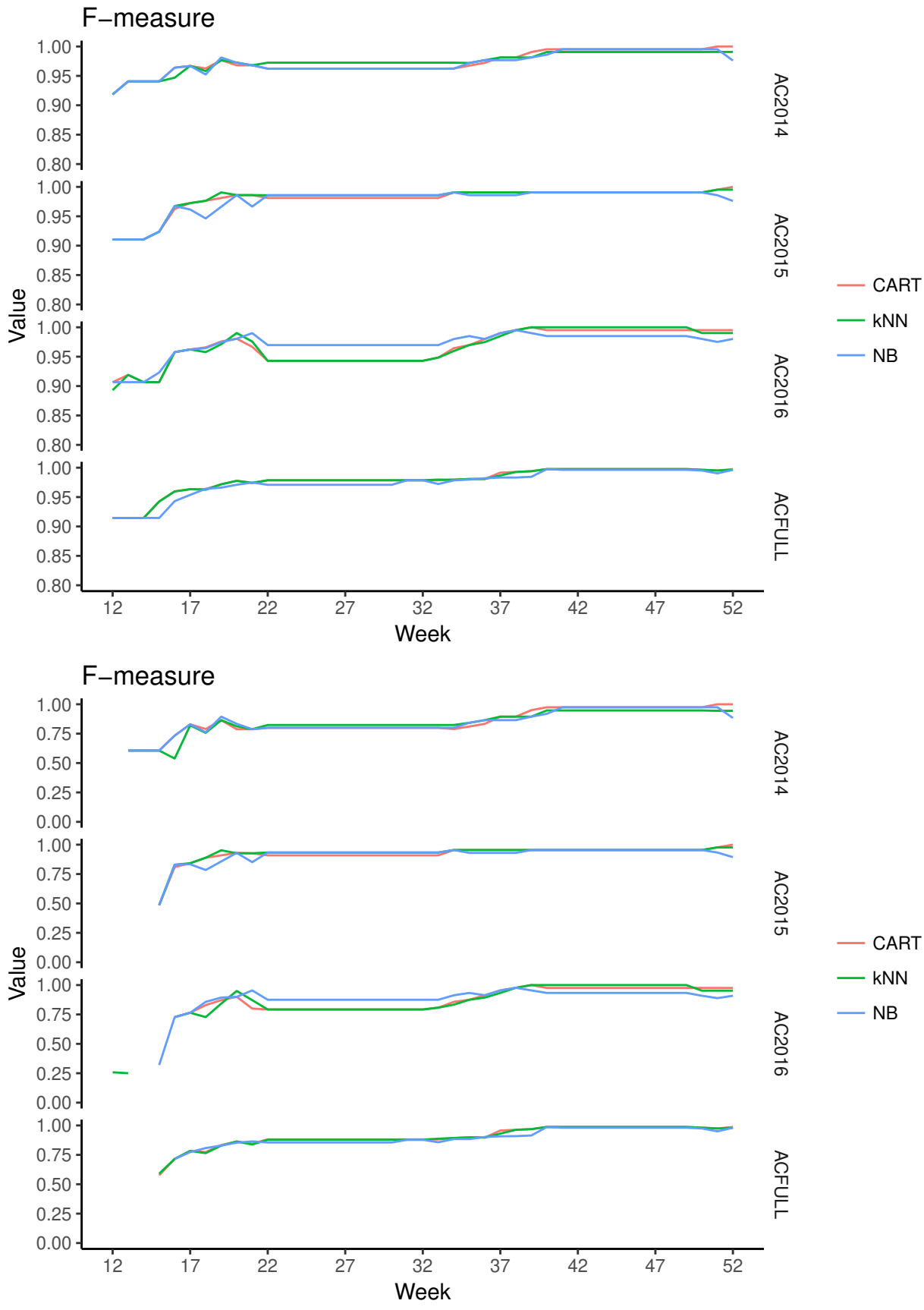


Figure C.3: F-measure statictics on Pass class (top) and Fail class (bottom)

Table C.7: F-measure values from classification on the AC2014 dataset for pass and fail classes are in the first six columns (excluding week). Similarly, values obtained from classification on AC2015 dataset are in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.92	0.92	0.92	NA	NA	NA	0.91	0.91	0.91	NA	NA	NA
13	0.94	0.94	0.94	0.61	0.61	0.61	0.91	0.91	0.91	NA	NA	NA
14	0.94	0.94	0.94	0.61	0.61	0.61	0.91	0.91	0.91	NA	NA	NA
15	0.94	0.94	0.94	0.61	0.61	0.61	0.92	0.92	0.92	0.48	0.48	0.48
16	0.96	0.95	0.96	0.73	0.54	0.73	0.96	0.97	0.97	0.81	0.83	0.83
17	0.97	0.97	0.97	0.83	0.82	0.83	0.97	0.97	0.96	0.84	0.84	0.83
18	0.96	0.96	0.95	0.79	0.76	0.76	0.98	0.98	0.95	0.89	0.89	0.78
19	0.98	0.98	0.98	0.86	0.86	0.89	0.98	0.99	0.97	0.91	0.95	0.86
20	0.97	0.97	0.97	0.79	0.81	0.83	0.99	0.99	0.99	0.93	0.93	0.93
21	0.97	0.97	0.97	0.79	0.79	0.79	0.99	0.99	0.97	0.93	0.93	0.85
22	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
23	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
24	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
25	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
26	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
27	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
28	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
29	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
30	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
31	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
32	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
33	0.96	0.97	0.96	0.8	0.82	0.8	0.98	0.99	0.99	0.91	0.93	0.93
34	0.96	0.97	0.96	0.79	0.82	0.8	0.99	0.99	0.99	0.95	0.95	0.95
35	0.97	0.97	0.97	0.81	0.84	0.84	0.99	0.99	0.99	0.95	0.95	0.93
36	0.97	0.98	0.98	0.83	0.86	0.86	0.99	0.99	0.99	0.95	0.95	0.93
37	0.98	0.98	0.98	0.89	0.89	0.86	0.99	0.99	0.99	0.95	0.95	0.93
38	0.98	0.98	0.98	0.89	0.89	0.86	0.99	0.99	0.99	0.95	0.95	0.93
39	0.99	0.98	0.98	0.95	0.89	0.89	0.99	0.99	0.99	0.95	0.95	0.95
40	1	0.99	0.99	0.97	0.95	0.92	0.99	0.99	0.99	0.95	0.95	0.95
41	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
42	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
43	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
44	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
45	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
46	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
47	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
48	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
49	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
50	1	0.99	1	0.97	0.95	0.97	0.99	0.99	0.99	0.95	0.95	0.95
51	1	0.99	1	1	0.94	0.97	1	1	0.99	0.98	0.98	0.93
52	1	0.99	0.98	1	0.94	0.88	1	1	0.98	1	0.98	0.89

Table C.8: F-measure values from classification on the AC2016 dataset for pass and fail classes are in the first six columns (excluding week). Similarly, values obtained from classification on ACFULL dataset are in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.91	0.89	0.91	NA	0.26	NA	0.91	0.91	0.91	NA	NA	NA
13	0.92	0.92	0.91	0.25	0.25	NA	0.91	0.91	0.91	NA	NA	NA
14	0.91	0.91	0.91	NA	NA	NA	0.91	0.91	0.91	NA	NA	NA
15	0.91	0.91	0.92	NA	NA	0.32	0.94	0.94	0.91	0.58	0.59	NA
16	0.96	0.96	0.96	0.73	0.73	0.73	0.96	0.96	0.94	0.72	0.72	0.72
17	0.96	0.96	0.96	0.76	0.76	0.76	0.96	0.96	0.95	0.78	0.78	0.77
18	0.97	0.96	0.96	0.83	0.73	0.86	0.96	0.96	0.96	0.77	0.76	0.81
19	0.98	0.97	0.97	0.87	0.84	0.89	0.97	0.97	0.97	0.83	0.83	0.83
20	0.98	0.99	0.98	0.9	0.95	0.9	0.98	0.98	0.97	0.86	0.86	0.85
21	0.97	0.98	0.99	0.8	0.87	0.95	0.97	0.97	0.97	0.85	0.84	0.86
22	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
23	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
24	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
25	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
26	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
27	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
28	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
29	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
30	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.97	0.88	0.88	0.86
31	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.98	0.88	0.88	0.88
32	0.94	0.94	0.97	0.79	0.79	0.88	0.98	0.98	0.98	0.88	0.88	0.88
33	0.95	0.95	0.97	0.81	0.81	0.88	0.98	0.98	0.97	0.88	0.89	0.86
34	0.96	0.96	0.98	0.86	0.83	0.91	0.98	0.98	0.98	0.89	0.89	0.88
35	0.97	0.97	0.99	0.88	0.88	0.93	0.98	0.98	0.98	0.9	0.9	0.88
36	0.98	0.97	0.98	0.91	0.89	0.91	0.98	0.98	0.98	0.9	0.9	0.9
37	0.99	0.99	0.99	0.95	0.93	0.95	0.99	0.99	0.98	0.96	0.93	0.91
38	1	1	1	0.98	0.98	0.98	0.99	0.99	0.98	0.96	0.96	0.91
39	1	1	0.99	1	1	0.95	0.99	0.99	0.98	0.97	0.97	0.92
40	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.99
41	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
42	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
43	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
44	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
45	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
46	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
47	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
48	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
49	1	1	0.99	0.98	1	0.93	1	1	1	0.99	0.99	0.98
50	1	0.99	0.98	0.98	0.95	0.91	1	1	1	0.98	0.98	0.97
51	1	0.99	0.98	0.98	0.95	0.89	1	1	0.99	0.97	0.97	0.95
52	1	0.99	0.98	0.98	0.95	0.91	1	1	1	0.99	0.98	0.98

Appendix D

Quality measures of AC datasets trained on another dataset

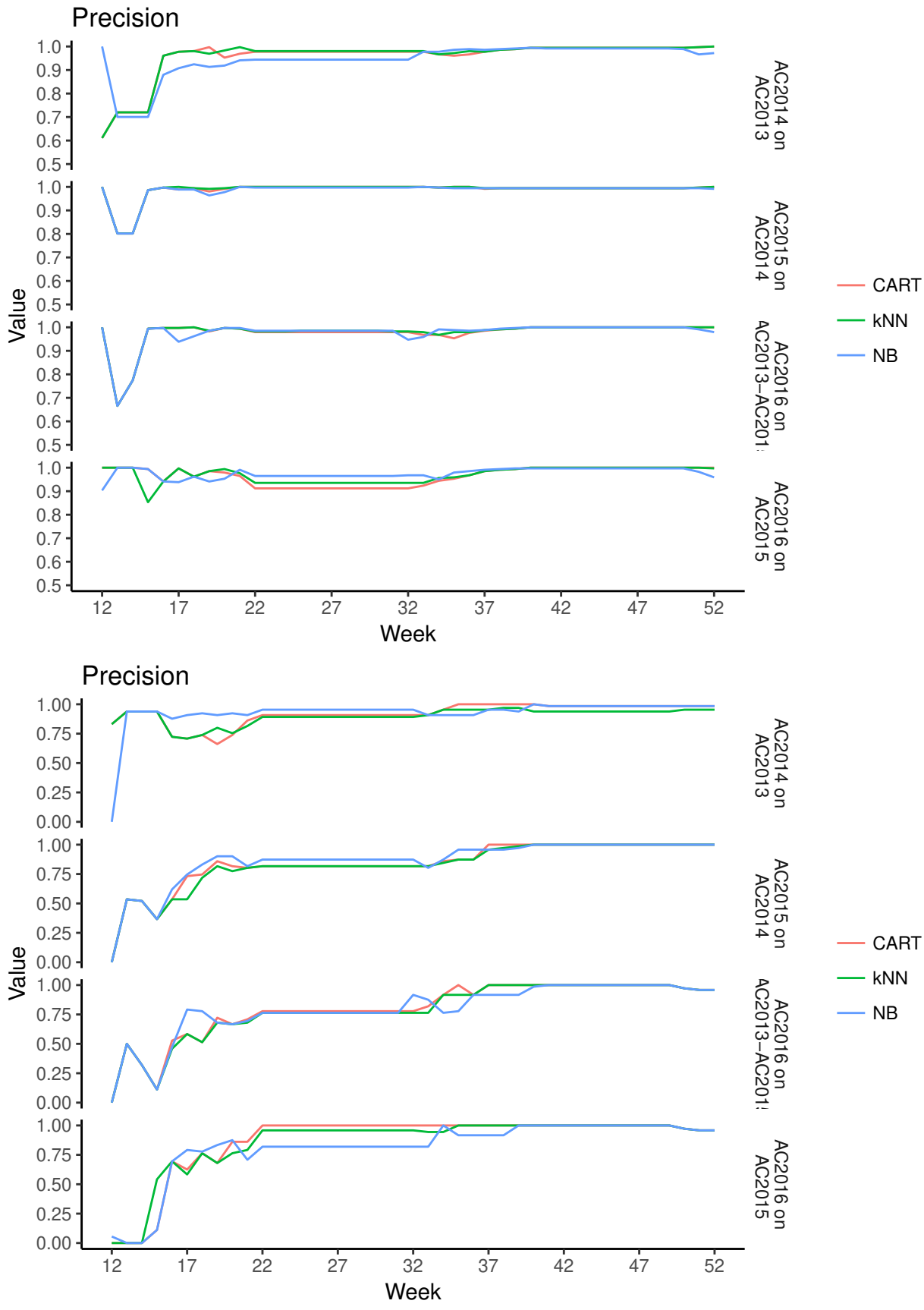


Figure D.1: Precision statictics on Pass class (top) and Fail class (bottom)

Table D.1: Precision values for pass and fail classes for AC2014 dataset trained on the AC2013 dataset in the first six columns and AC2015 dataset trained on the AC2014 dataset in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.61	0.61	1	0.83	0.83	0	1	1	1	0	0	0
13	0.72	0.72	0.7	0.94	0.94	0.94	0.8	0.8	0.8	0.54	0.54	0.54
14	0.72	0.72	0.7	0.94	0.94	0.94	0.8	0.8	0.8	0.52	0.52	0.52
15	0.72	0.72	0.7	0.94	0.94	0.94	0.99	0.99	0.99	0.37	0.37	0.37
16	0.96	0.96	0.88	0.72	0.72	0.88	1	1	1	0.54	0.54	0.62
17	0.98	0.98	0.91	0.71	0.71	0.91	0.99	1	0.99	0.73	0.54	0.75
18	0.98	0.98	0.92	0.74	0.74	0.92	0.99	0.99	0.99	0.75	0.72	0.83
19	1	0.97	0.91	0.66	0.8	0.91	0.98	0.99	0.96	0.86	0.82	0.9
20	0.95	0.98	0.92	0.74	0.75	0.92	0.99	0.99	0.98	0.82	0.77	0.9
21	0.97	1	0.94	0.86	0.82	0.91	1	1	1	0.8	0.8	0.82
22	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
23	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
24	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
25	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
26	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
27	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
28	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
29	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
30	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
31	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
32	0.98	0.98	0.94	0.91	0.89	0.95	1	1	1	0.82	0.82	0.87
33	0.98	0.98	0.98	0.91	0.91	0.91	1	1	1	0.82	0.82	0.8
34	0.97	0.97	0.98	0.95	0.95	0.91	1	1	1	0.86	0.85	0.87
35	0.96	0.97	0.99	1	0.95	0.91	1	1	0.99	0.87	0.87	0.96
36	0.97	0.98	0.99	1	0.95	0.91	1	1	0.99	0.87	0.87	0.96
37	0.98	0.98	0.99	1	0.95	0.95	0.99	0.99	0.99	1	0.96	0.96
38	0.99	0.99	0.99	1	0.97	0.95	0.99	0.99	0.99	1	0.97	0.96
39	0.99	0.99	0.99	1	0.97	0.94	0.99	0.99	0.99	1	0.99	0.97
40	0.99	0.99	0.99	1	0.94	1	0.99	0.99	0.99	1	1	1
41	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
42	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
43	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
44	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
45	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
46	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
47	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
48	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
49	0.99	0.99	0.99	0.98	0.94	0.98	0.99	0.99	0.99	1	1	1
50	0.99	0.99	0.99	0.98	0.95	0.98	0.99	0.99	0.99	1	1	1
51	1	1	0.97	0.98	0.95	0.98	1	1	0.99	1	1	1
52	1	1	0.97	0.98	0.95	0.98	1	1	0.99	1	1	1

Table D.2: Precision values for pass and fail classes for AC2016 dataset trained on the AC2015 dataset in the first six columns and AC2016 dataset trained on AC2013, AC2014 and AC2015 datasets in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	1	1	0.9	0	0	0.06	1	1	1	0	0	0
13	1	1	1	0	0	0	0.67	0.67	0.67	0.5	0.5	0.5
14	1	1	1	0	0	0	0.77	0.77	0.77	0.32	0.32	0.32
15	0.99	0.85	0.99	0.11	0.54	0.11	0.99	0.99	0.99	0.11	0.11	0.11
16	0.94	0.94	0.94	0.69	0.69	0.69	1	1	1	0.53	0.46	0.47
17	1	1	0.94	0.62	0.58	0.79	1	1	0.94	0.58	0.58	0.79
18	0.96	0.96	0.96	0.76	0.76	0.78	1	1	0.96	0.51	0.51	0.78
19	0.99	0.99	0.94	0.68	0.68	0.83	0.98	0.99	0.99	0.72	0.68	0.68
20	0.98	0.99	0.95	0.86	0.76	0.88	1	1	1	0.67	0.67	0.67
21	0.96	0.98	0.99	0.86	0.79	0.71	0.99	0.99	1	0.71	0.68	0.69
22	0.91	0.94	0.96	1	0.96	0.82	0.98	0.98	0.99	0.78	0.76	0.76
23	0.91	0.94	0.96	1	0.96	0.82	0.98	0.98	0.99	0.78	0.76	0.76
24	0.91	0.94	0.96	1	0.96	0.82	0.98	0.98	0.99	0.78	0.76	0.76
25	0.91	0.94	0.96	1	0.96	0.82	0.98	0.99	0.99	0.78	0.76	0.76
26	0.91	0.94	0.96	1	0.96	0.82	0.98	0.99	0.99	0.78	0.76	0.76
27	0.91	0.94	0.96	1	0.96	0.82	0.98	0.99	0.99	0.78	0.76	0.76
28	0.91	0.94	0.96	1	0.96	0.82	0.98	0.99	0.99	0.78	0.76	0.76
29	0.91	0.94	0.96	1	0.96	0.82	0.98	0.99	0.99	0.78	0.76	0.76
30	0.91	0.94	0.96	1	0.96	0.82	0.98	0.99	0.99	0.78	0.76	0.76
31	0.91	0.94	0.96	1	0.96	0.82	0.98	0.98	0.99	0.78	0.76	0.76
32	0.91	0.94	0.97	1	0.96	0.82	0.98	0.98	0.95	0.78	0.76	0.92
33	0.92	0.94	0.97	1	0.94	0.82	0.97	0.98	0.96	0.82	0.76	0.88
34	0.94	0.96	0.95	1	0.94	1	0.97	0.97	0.99	0.92	0.92	0.76
35	0.95	0.96	0.98	1	1	0.92	0.95	0.98	0.99	1	0.92	0.78
36	0.97	0.97	0.99	1	1	0.92	0.98	0.98	0.99	0.92	0.92	0.92
37	0.99	0.99	0.99	1	1	0.92	0.99	0.99	0.99	1	1	0.92
38	0.99	0.99	0.99	1	1	0.92	0.99	0.99	0.99	1	1	0.92
39	0.99	0.99	1	1	1	1	0.99	0.99	1	1	1	0.92
40	1	1	1	1	1	1	1	1	1	1	1	0.99
41	1	1	1	1	1	1	1	1	1	1	1	1
42	1	1	1	1	1	1	1	1	1	1	1	1
43	1	1	1	1	1	1	1	1	1	1	1	1
44	1	1	1	1	1	1	1	1	1	1	1	1
45	1	1	1	1	1	1	1	1	1	1	1	1
46	1	1	1	1	1	1	1	1	1	1	1	1
47	1	1	1	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1	1	1	1
49	1	1	1	1	1	1	1	1	1	1	1	1
50	1	1	1	0.97	0.97	0.97	1	1	1	0.97	0.97	0.97
51	1	1	0.98	0.96	0.96	0.96	1	1	0.99	0.96	0.96	0.96
52	1	1	0.96	0.96	0.96	0.96	1	1	0.98	0.96	0.96	0.96

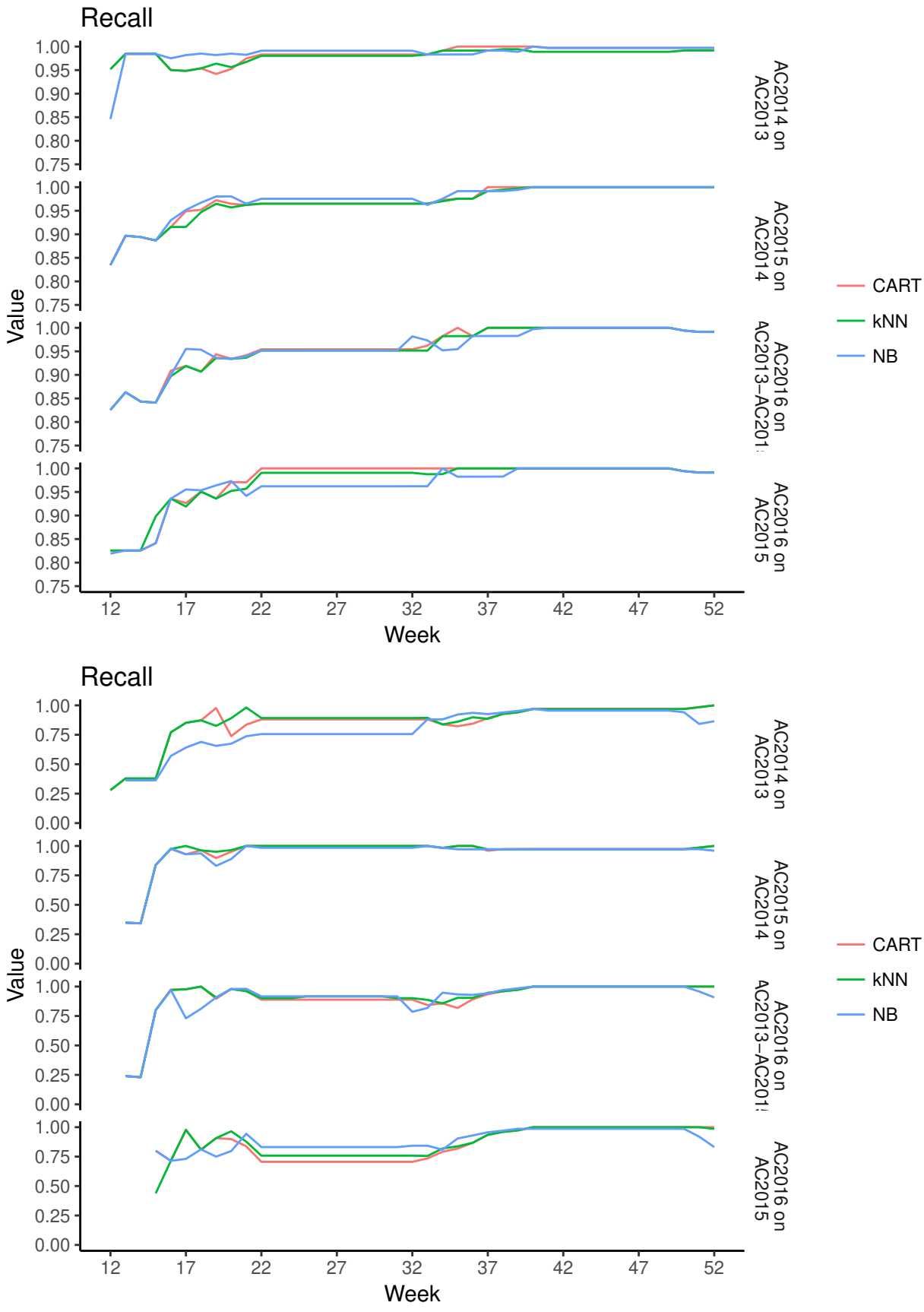


Figure D.2: Recall statictics on Pass class (top) and Fail class (bottom)

Table D.3: Recall values for pass and fail classes for AC2014 dataset trained on the AC2013 dataset in the first six columns and AC2015 dataset trained on AC2014 dataset in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.95	0.95	0.85	0.28	0.28	NA	0.83	0.83	0.83	NA	NA	NA
13	0.98	0.98	0.98	0.38	0.38	0.36	0.9	0.9	0.9	0.35	0.35	0.35
14	0.98	0.98	0.98	0.38	0.38	0.36	0.89	0.89	0.89	0.34	0.34	0.34
15	0.98	0.98	0.98	0.38	0.38	0.36	0.89	0.89	0.89	0.84	0.84	0.84
16	0.95	0.95	0.98	0.77	0.77	0.57	0.92	0.92	0.93	0.97	0.97	0.98
17	0.95	0.95	0.98	0.85	0.85	0.64	0.95	0.92	0.95	0.93	1	0.93
18	0.95	0.95	0.99	0.87	0.87	0.69	0.95	0.95	0.97	0.96	0.96	0.94
19	0.94	0.96	0.98	0.98	0.83	0.66	0.97	0.96	0.98	0.9	0.95	0.83
20	0.95	0.96	0.98	0.74	0.89	0.67	0.96	0.96	0.98	0.95	0.96	0.89
21	0.97	0.97	0.98	0.84	0.98	0.74	0.96	0.96	0.96	1	1	1
22	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
23	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
24	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
25	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
26	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
27	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
28	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
29	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
30	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
31	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
32	0.98	0.98	0.99	0.88	0.89	0.76	0.96	0.96	0.98	1	1	0.98
33	0.98	0.98	0.98	0.88	0.89	0.88	0.96	0.96	0.96	1	1	1
34	0.99	0.99	0.98	0.84	0.84	0.88	0.97	0.97	0.98	0.98	0.98	0.98
35	1	0.99	0.98	0.82	0.86	0.92	0.98	0.98	0.99	1	1	0.97
36	1	0.99	0.98	0.84	0.9	0.94	0.98	0.98	0.99	1	1	0.97
37	1	0.99	0.99	0.89	0.89	0.93	1	0.99	0.99	0.96	0.97	0.97
38	1	0.99	0.99	0.93	0.93	0.94	1	0.99	0.99	0.97	0.97	0.97
39	1	0.99	0.99	0.94	0.94	0.95	1	1	0.99	0.97	0.97	0.97
40	1	0.99	1	0.97	0.97	0.97	1	1	1	0.97	0.97	0.97
41	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
42	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
43	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
44	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
45	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
46	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
47	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
48	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
49	1	0.99	1	0.97	0.97	0.96	1	1	1	0.97	0.97	0.97
50	1	0.99	1	0.97	0.97	0.94	1	1	1	0.97	0.97	0.97
51	1	0.99	1	0.98	0.98	0.84	1	1	1	0.99	0.99	0.97
52	1	0.99	1	1	1	0.86	1	1	1	1	1	0.96

Table D.4: Recall values for pass and fail classes for AC2016 dataset trained on the AC2015 dataset in the first six columns and AC2016 dataset trained on AC2013, AC2014 and AC2015 datasets in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.83	0.83	0.82	NA	NA	0.11	0.83	0.83	0.83	NA	NA	NA
13	0.83	0.83	0.83	NA	NA	NA	0.86	0.86	0.86	0.24	0.24	0.24
14	0.83	0.83	0.83	NA	NA	NA	0.84	0.84	0.84	0.23	0.23	0.23
15	0.84	0.9	0.84	0.8	0.44	0.8	0.84	0.84	0.84	0.8	0.8	0.8
16	0.94	0.94	0.94	0.71	0.71	0.71	0.91	0.9	0.9	0.97	0.97	0.97
17	0.93	0.92	0.96	0.98	0.98	0.73	0.92	0.92	0.96	0.98	0.98	0.73
18	0.95	0.95	0.95	0.81	0.81	0.81	0.91	0.91	0.95	1	1	0.81
19	0.94	0.94	0.96	0.91	0.91	0.75	0.94	0.94	0.94	0.9	0.91	0.91
20	0.97	0.95	0.97	0.9	0.96	0.8	0.93	0.93	0.93	0.98	0.98	0.98
21	0.97	0.96	0.94	0.84	0.88	0.94	0.94	0.94	0.94	0.96	0.96	0.98
22	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.9	0.92
23	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.9	0.92
24	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.9	0.92
25	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.92	0.92
26	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.92	0.92
27	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.92	0.92
28	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.92	0.92
29	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.92	0.92
30	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.92	0.92
31	1	0.99	0.96	0.71	0.76	0.83	0.95	0.95	0.95	0.89	0.9	0.92
32	1	0.99	0.96	0.71	0.76	0.84	0.95	0.95	0.98	0.89	0.9	0.79
33	1	0.99	0.96	0.73	0.76	0.84	0.96	0.95	0.97	0.84	0.89	0.82
34	1	0.99	1	0.79	0.82	0.81	0.98	0.98	0.95	0.86	0.86	0.95
35	1	1	0.98	0.82	0.84	0.9	1	0.98	0.95	0.82	0.9	0.93
36	1	1	0.98	0.87	0.87	0.93	0.98	0.98	0.98	0.89	0.9	0.93
37	1	1	0.98	0.94	0.94	0.96	1	1	0.98	0.94	0.95	0.94
38	1	1	0.98	0.96	0.96	0.97	1	1	0.98	0.96	0.96	0.97
39	1	1	1	0.97	0.97	0.99	1	1	0.98	0.97	0.97	0.99
40	1	1	1	1	1	0.99	1	1	1	1	1	1
41	1	1	1	1	1	0.99	1	1	1	1	1	1
42	1	1	1	1	1	0.99	1	1	1	1	1	1
43	1	1	1	1	1	0.99	1	1	1	1	1	1
44	1	1	1	1	1	0.99	1	1	1	1	1	1
45	1	1	1	1	1	0.99	1	1	1	1	1	1
46	1	1	1	1	1	0.99	1	1	1	1	1	1
47	1	1	1	1	1	0.99	1	1	1	1	1	1
48	1	1	1	1	1	0.99	1	1	1	1	1	1
49	1	1	1	1	1	0.99	1	1	1	1	1	1
50	0.99	0.99	0.99	1	1	0.99	0.99	0.99	0.99	1	1	1
51	0.99	0.99	0.99	1	1	0.92	0.99	0.99	0.99	1	1	0.96
52	0.99	0.99	0.99	1	0.99	0.83	0.99	0.99	0.99	1	1	0.91

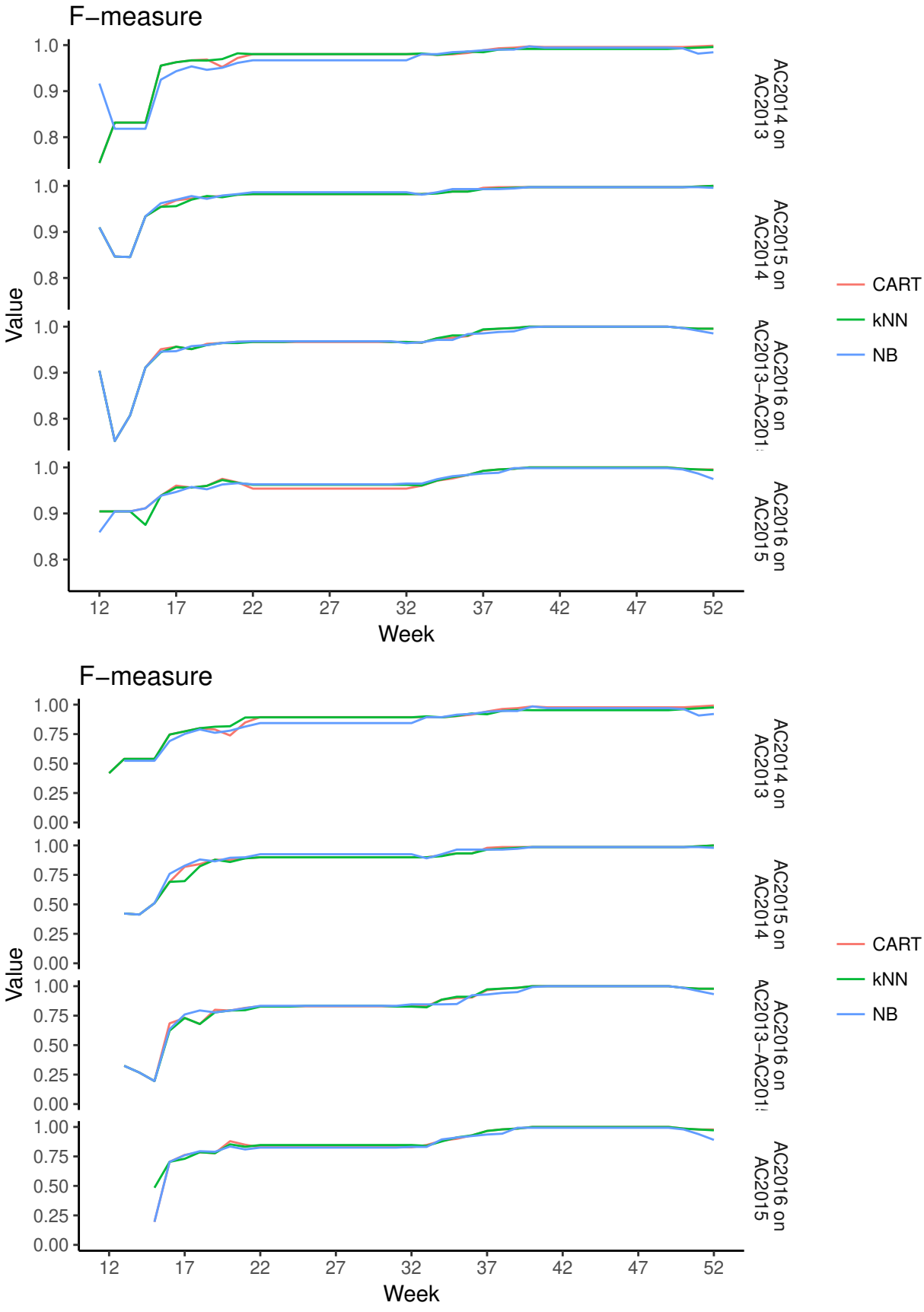


Figure D.3: F-measure statictics on Pass class (top) and Fail class (bottom)

Table D.5: F-measure values for pass and fail classes for AC2014 dataset trained on the AC2013 dataset in the first six columns and AC2015 dataset trained on AC2014 dataset in the rest of the columns.

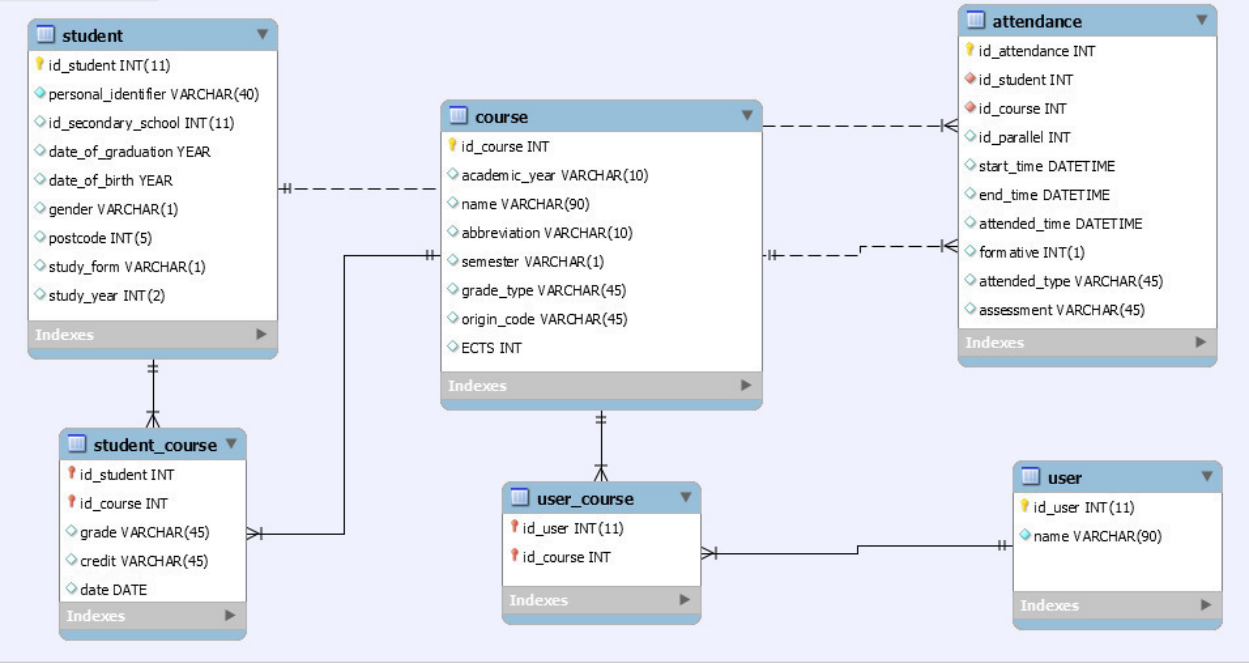
week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.74	0.74	0.92	0.42	0.42	NA	0.91	0.91	0.91	NA	NA	NA
13	0.83	0.83	0.82	0.54	0.54	0.52	0.85	0.85	0.85	0.42	0.42	0.42
14	0.83	0.83	0.82	0.54	0.54	0.52	0.85	0.85	0.85	0.41	0.41	0.41
15	0.83	0.83	0.82	0.54	0.54	0.52	0.93	0.93	0.93	0.51	0.51	0.51
16	0.96	0.96	0.92	0.75	0.75	0.69	0.95	0.95	0.96	0.69	0.69	0.76
17	0.96	0.96	0.94	0.77	0.77	0.75	0.97	0.96	0.97	0.82	0.7	0.83
18	0.97	0.97	0.95	0.8	0.8	0.79	0.97	0.97	0.98	0.84	0.82	0.88
19	0.97	0.97	0.95	0.79	0.81	0.76	0.98	0.98	0.97	0.88	0.88	0.86
20	0.95	0.97	0.95	0.74	0.82	0.78	0.98	0.98	0.98	0.88	0.86	0.9
21	0.97	0.98	0.96	0.85	0.89	0.81	0.98	0.98	0.98	0.89	0.89	0.9
22	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
23	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
24	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
25	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
26	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
27	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
28	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
29	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
30	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
31	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
32	0.98	0.98	0.97	0.89	0.89	0.84	0.98	0.98	0.99	0.9	0.9	0.93
33	0.98	0.98	0.98	0.89	0.9	0.89	0.98	0.98	0.98	0.9	0.9	0.89
34	0.98	0.98	0.98	0.89	0.89	0.89	0.98	0.98	0.99	0.92	0.91	0.93
35	0.98	0.98	0.98	0.9	0.91	0.91	0.99	0.99	0.99	0.93	0.93	0.96
36	0.98	0.99	0.99	0.92	0.93	0.92	0.99	0.99	0.99	0.93	0.93	0.96
37	0.99	0.98	0.99	0.94	0.92	0.94	1	0.99	0.99	0.98	0.96	0.96
38	0.99	0.99	0.99	0.96	0.95	0.95	1	0.99	0.99	0.99	0.97	0.96
39	0.99	0.99	0.99	0.97	0.95	0.95	1	1	0.99	0.99	0.98	0.97
40	1	0.99	1	0.98	0.95	0.98	1	1	1	0.99	0.99	0.99
41	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
42	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
43	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
44	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
45	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
46	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
47	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
48	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
49	1	0.99	0.99	0.98	0.95	0.97	1	1	1	0.99	0.99	0.99
50	1	0.99	0.99	0.98	0.96	0.96	1	1	1	0.99	0.99	0.99
51	1	0.99	0.98	0.98	0.97	0.91	1	1	1	0.99	0.99	0.99
52	1	1	0.98	0.99	0.98	0.92	1	1	1	1	1	0.98

Table D.6: F-measure values for pass and fail classes for AC2016 dataset trained on the AC2015 dataset in the first six columns and AC2016 dataset trained on AC2013, AC2014 and AC2015 datasets in the rest of the columns.

week	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB	CART	kNN	NB
12	0.9	0.9	0.86	NA	NA	0.07	0.9	0.9	0.9	NA	NA	NA
13	0.9	0.9	0.9	NA	NA	NA	0.75	0.75	0.75	0.32	0.32	0.32
14	0.9	0.9	0.9	NA	NA	NA	0.81	0.81	0.81	0.27	0.27	0.27
15	0.91	0.88	0.91	0.2	0.48	0.2	0.91	0.91	0.91	0.2	0.2	0.2
16	0.94	0.94	0.94	0.7	0.7	0.7	0.95	0.94	0.95	0.68	0.62	0.64
17	0.96	0.96	0.95	0.76	0.73	0.76	0.96	0.96	0.95	0.73	0.73	0.76
18	0.96	0.96	0.96	0.79	0.79	0.79	0.95	0.95	0.96	0.68	0.68	0.79
19	0.96	0.96	0.95	0.78	0.78	0.79	0.96	0.96	0.96	0.8	0.78	0.78
20	0.98	0.97	0.96	0.88	0.85	0.83	0.96	0.96	0.96	0.79	0.79	0.79
21	0.97	0.97	0.97	0.85	0.83	0.81	0.97	0.96	0.97	0.82	0.8	0.81
22	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
23	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
24	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
25	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
26	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
27	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
28	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
29	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
30	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
31	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.97	0.83	0.83	0.83
32	0.95	0.96	0.96	0.83	0.85	0.83	0.97	0.97	0.96	0.83	0.83	0.85
33	0.96	0.96	0.96	0.85	0.84	0.83	0.96	0.97	0.97	0.83	0.82	0.85
34	0.97	0.97	0.97	0.88	0.88	0.89	0.97	0.97	0.97	0.89	0.89	0.85
35	0.98	0.98	0.98	0.9	0.91	0.91	0.98	0.98	0.97	0.9	0.91	0.85
36	0.98	0.98	0.98	0.93	0.93	0.92	0.98	0.98	0.98	0.9	0.91	0.92
37	0.99	0.99	0.99	0.97	0.97	0.94	0.99	0.99	0.99	0.97	0.97	0.93
38	1	1	0.99	0.98	0.98	0.94	1	1	0.99	0.98	0.98	0.94
39	1	1	1	0.99	0.99	0.99	1	1	0.99	0.99	0.99	0.95
40	1	1	1	1	1	0.99	1	1	1	1	1	0.99
41	1	1	1	1	1	0.99	1	1	1	1	1	1
42	1	1	1	1	1	0.99	1	1	1	1	1	1
43	1	1	1	1	1	0.99	1	1	1	1	1	1
44	1	1	1	1	1	0.99	1	1	1	1	1	1
45	1	1	1	1	1	0.99	1	1	1	1	1	1
46	1	1	1	1	1	0.99	1	1	1	1	1	1
47	1	1	1	1	1	0.99	1	1	1	1	1	1
48	1	1	1	1	1	0.99	1	1	1	1	1	1
49	1	1	1	1	1	0.99	1	1	1	1	1	1
50	1	1	1	0.99	0.99	0.98	1	1	1	0.99	0.99	0.99
51	1	1	0.99	0.98	0.98	0.94	1	1	0.99	0.98	0.98	0.96
52	1	0.99	0.97	0.98	0.97	0.89	1	1	0.99	0.98	0.98	0.93

Appendix E

Analyst database model



Appendix F

Content of CD

- readme.txt - file with CD contents description
- text - directory with the text in pdf

References

"9 Best Databases to Use for Node.js Applications as of 2018." 2018. https://www.slant.co/topics/179/~best-databases-to-use-for-node-js-applications.

Altman, Naomi S. 1992. "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression." *The American Statistician* 46 (3). Taylor & Francis:175–85.

Andras, Samuel. 2017. "JavaScript Frameworks, Why and When to Use Them." https://blog.hellojs.org/javascript-frameworks-why-and-when-to-use-them-43af33d0608d.

"AngularJS — Superheroic Javascript Mvw Framework." 2010. https://angularjs.org/.

Bentley, Jon Louis. 1975. "Multidimensional Binary Search Trees Used for Associative Searching." *Communications of the ACM* 18 (9). ACM:509–17.

Bernhardt, Manuel. 2016. *Reactive Web Applications: Covers Play, Akka, and Reactive Streams*. Manning Publications Co.

Breiman, Leo. 2017. *Classification and Regression Trees*. Routledge.

Brian Behlendorf, Rob Hartill, Roy Fielding. 1995. "The Apache Http Server Project." https://httpd.apache.org/.

Chambers, John. 1976. *The S System*. Bell Labs. https://www.R-project.org/.

Chang, Winston, Joe Cheng, JJ Allaire, Yihui Xie, and Jonathan McPherson. 2017. *Shiny: Web Application Framework for R*. https://CRAN.R-project.org/package=shiny.

Cunningham, Padraig, and Sarah Jane Delany. 2007. "K-Nearest Neighbour Classifiers." *Multiple Classifier Systems* 34. Springer:1–17.

"European Credit Transfer and Accumulation System (Ects)." 2003. http://ec.europa.eu/education/resources/european-credit-transfer-accumulation-system_en.

"Express - Node.js Web Application Framework." 2010. https://expressjs.com/.

Ferguson, Rebecca. 2012. "Learning Analytics: Drivers, Developments and Challenges." *International Journal of Technology Enhanced Learning* 4 (5-6). Inderscience Publishers:304–17.

Ferguson, Rebecca, Andrew Brasher, Doug Clow, Adam Cooper, Garron Hillaire, Jenna Mittelmeier, Bart Rienties, Thomas Ullmann, and Riina Vuorikari. 2016. "Research Evidence on the Use of Learning Analytics: Implications for Education Policy." Joint Research Centre.

Jivet, Ioana, Maren Scheffel, Marcus Specht, and Hendrik Drachsler. 2018. "License to Evaluate: Preparing Learning Analytics Dashboards for Educational Practice."

Kodinariya, Trupti M, and Prashant R Makwana. 2013. "Review on Determining Number of Cluster in K-Means Clustering." *International Journal* 1 (6):90–95.

Kolahdouzan, Mohammad, and Cyrus Shahabi. 2004. "Voronoi-Based K Nearest Neighbor Search for Spatial Network Databases." In *Proceedings of the Thirtieth International Conference on Very Large Data Bases-Volume 30*, 840–51. VLDB Endowment.

Kuzilek, Jakub, Martin Hlosta, Drahomira Herrmannova, Zdenek Zdrahal, and Annika Wolff. 2015. "OU Analyse: Analysing at-Risk Students at the Open University." *Learning Analytics Review*, 1–16.

Lloyd, Stuart. 1982. "Least Squares Quantization in Pcm." *IEEE Transactions on Information Theory* 28 (2). IEEE:129–37.

Maly, Martin. 2009. "REST: Architektura Pro Webové Api." https://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/.

Moodle. 2018. *Moodle Hq, Perth*. https://moodle.com/.

Murphy, Kevin P. 2006. "Naive Bayes Classifiers." *University of British Columbia* 18.

"MySQL." 1995. https://www.mysql.com/.

"Node.js." 2010. https://nodejs.org/en/.

"Npm Package Manager." 2010. https://www.npmjs.com/.

Papamitsiou, Zacharoula, and Anastasios A Economides. 2014. "Learning Analytics and Educational Data Mining in Practice: A Systematic Literature Review of Empirical Evidence." *Journal of Educational Technology & Society* 17 (4). JSTOR:49.

Pearson, Karl. 1900. "X. On the Criterion That a Given System of Deviations from the Probable in the Case of a Correlated System of Variables Is Such That It Can Be Reasonably Supposed to Have Arisen from Random Sampling." *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 50 (302). Taylor & Francis:157–75.

Quinn, Jocey. 2013. "Drop-Out and Completion in Higher Education in Europe." *European Union*.

R Core Team. 2017. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. http://ect.bell-labs.com/sl/S/.

"React - a Javascript Library for Building User Interfaces." 2013. https://reactjs.org/.

Shacklock, Xanthe. 2016. *From Bricks to Clicks: The Potential of Data and Analytics in Higher Education*. Higher Education Commission London.

Sharma, Siddharth. 2014. "Performance Comparison of Angularjs 1.x and Angularjs 2 Through Contacts Manager Application." https://bit.ly/2FbZ08N.

Sievert, Carson, Chris Parmer, Toby Hocking, Scott Chamberlain, Karthik Ram, Marianne Corvellec, and Pedro Despouy. 2017. *Plotly: Create Interactive Web Graphics via 'Plotly.js'*. https://CRAN.R-project.org/package=plotly.

Sutton, Oliver. 2012. "Introduction to kNN Classification and Cnn Data Reduction." http://www.math.le.ac.uk/people/ag153/homepage/KNN/OliverKNN_Presentation.pdf.

Vaclavek, Jonas. 2015. "Web Front-End for Student Data Analysis Application."

Wickham, Hadley. 2009. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. http://ggplot2.org.

Wilkinson, Leland. 2006. *The Grammar of Graphics*. Springer Science & Business Media.

Xu, Rui, and Donald Wunsch. 2005. "Survey of Clustering Algorithms." *IEEE Transactions on Neural Networks* 16 (3). Ieee:645–78.

Zdrahal, Zdenek, UK CIIRC, CZ Prague, Martin Hlosta, and Jakub Kuzilek. 2016. "Analysing Performace of First Year Engineering Students."