CALIFORNIA STATE UNIVERISTY, NORTHRIDGE

Modeling in R and Weka for Course Enrollment Prediction

A thesis submitted in partial fulfillment of the requirements

For the degree of Master of Science in Computer Science

By

Amanda Watkins

August 2016

The thesis of Amanda Watkins is approved:

Dr. Jeff Wiegley	Date
Dr. John Noga	Date
Dr. Adam Kaplan, Chair	Date

California State University, Northridge

Dedication

Dedicated to Taylor Watkins, my amazing co-captain in softball and in life.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Modeling in R and Weka for Course Enrollment Prediction

By

Amanda Watkins

Master of Science in Computer Science

This thesis presents a tool developed for the comparison of R and Weka time series models for predicting undergraduate Computer Science course enrollments at CSUN. Current methodologies used at other universities along with related work on course enrollment prediction are examined to guide model selection as well as interpret the modeling test results. The models implemented in Weka are Gaussian Processes, Linear Regression, Multilayer Perceptron, and SMOreg. The models implemented using R's forecast package are ARIMA, ETS, and RWF. Predictions on holdout data are compared both in modified form, with numbers rounded up and negative values zeroed out, and unmodified form. The most accurate models when comparing three semesters of both modified and unmodified predictions against three semesters of holdout data were Gaussian Processes and SMOreg. All models were most accurate when predicting three semesters of holdout data using the maximum available enrollment data from Spring term of 2010 to Spring term of 2015 for training. Results at best predicted enrollment within 25 students for 93.5% of courses, and at worst for 77.4% of courses. Details on project maintenance as well as future enhancements are also included.

INTRODUCTION

California State University – Northridge (CSUN)'s Computer Science department needs to determine a schedule of undergraduate courses to offer each semester. In order to know how many sections of each course to offer, the student demand must be predicted. Failure to accurately predict the number of students wanting to enroll in particular courses has consequences. If the prediction of student enrollment is too high, sections may end up being cancelled due to having insufficient numbers to warrant offering the course. Consequently, students in cancelled sections would need to revise their schedules at the last minute, when other course options may no longer be available. Professors for the cancelled sections would be an under utilized resource and may not meet their required course load. Conversely, if the prediction of student enrollment were too low, then additional faculty would need to be found on short notice. This could result in inexperienced or under-prepared course instruction for above average class sizes. The overall schedule and physical classroom assignments might also need to change in order to accommodate the additional sections, feasibly causing conflicts with student and professor schedules. Additionally, planning for course tutors and other learning support, including career counseling, can be negatively affected by inaccurate course enrollment predictions. Ultimately, if the Computer Science department decides not to accommodate unforeseen course demand, then students could be set back in their academic schedule which could cause them to drop out of the program.

The enrollment situation at CSUN is unique due to a few factors. The California State University Statistical Reports show that out of the 23 California State

University campuses, CSUN consistently has one of the largest overall student bodies as well as the largest numbers of first-time freshmen and new undergraduate transfers every Fall.[1] When it comes to enrollment numbers, CSUN is among the top five universities in all of California, with its enrollment numbers consistently increasing.[2]

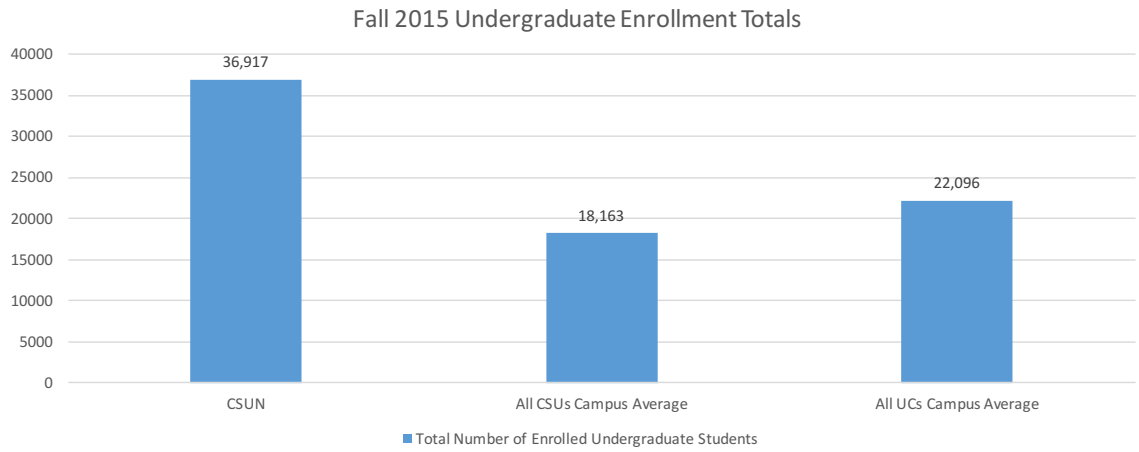

Figure 1 - Undergraduate Student Enrollment

CSUN's overall enrollment from Fall semester of 1993 to Fall semester of 2014 increased roughly 49%, with the College of Engineering and Computer Science experiencing an 82% increase in full-time enrollment.[3] As of 2015, the College of Engineering and Computer Science has the fifth-largest headcount among CSUN's eight colleges.[4] The composition of CSUN's new student enrollment for the Fall semester of 2015 was over 51% transfer students, with CSUN's overall undergraduate enrollment consisting of over 18% part-time students. For all 23 California State Universities for the Fall semester of 2015, the student body comprised of a little over 44% transfer students

[1] (California State University 2016)
[2] (California State University, Northridge 2013)
[3] (Rickes Associates Inc. 2015)
[4] (Rickes Associates Inc. 2015)

and just over 14% part-time students.[5] Both at CSUN and for all 23 CSUs approximately 4% of all Fall 2015 undergraduate students enrolled as computer science, computer engineering, or computer information technology majors.[6] While CSUN has the same ratio of undergraduate computer science and technology majors that CSUs overall have, it has an overall larger student body with more transfer and part-time students enrolled.

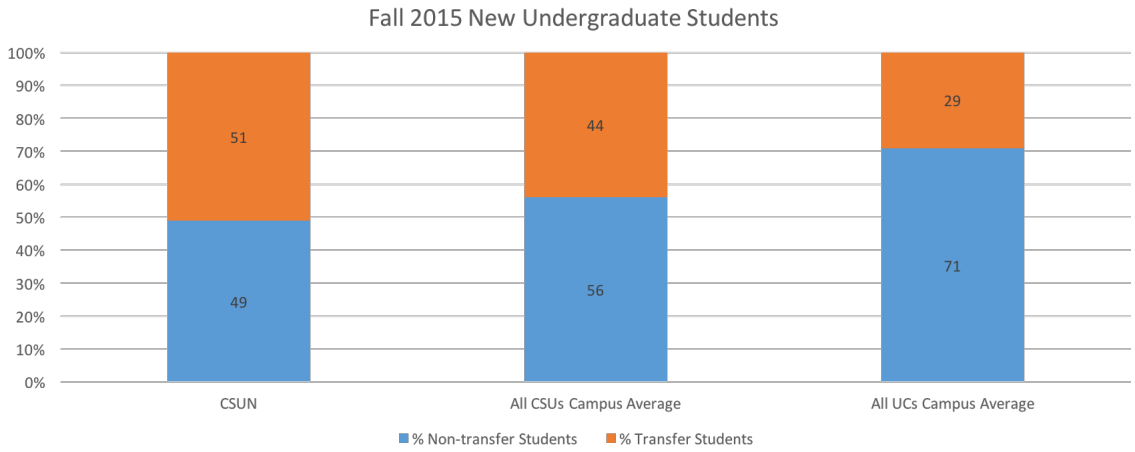

Figure 2 - Undergraduate Transfer Students

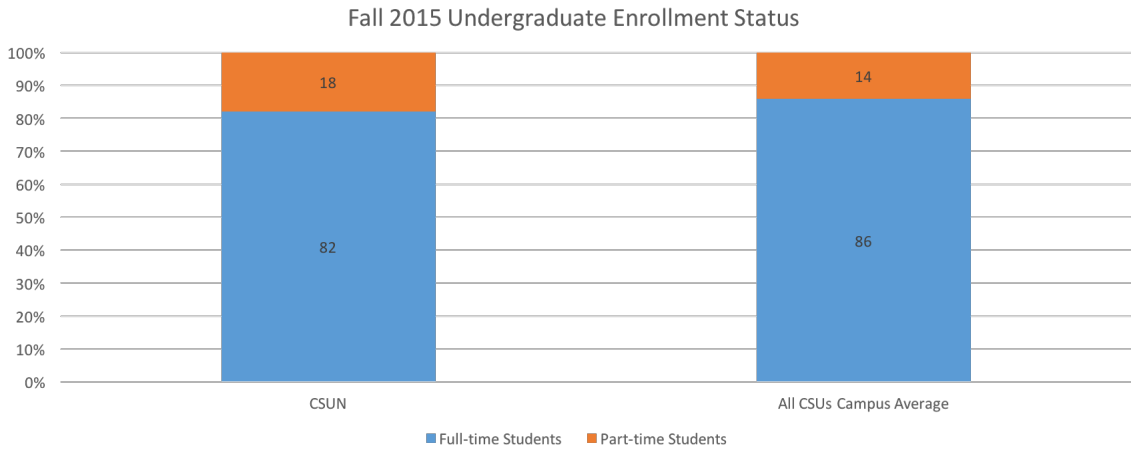

Figure 3 - Undergraduate Enrollment Status

[5] (California State University 2016)
[6] (California State University 2016)

When deciding how best to forecast enrollment in undergraduate Computer Science courses, factors such as this can help determine the suitability of different methods and configurations. In the best practices for enrollment modeling, there is the notion that "there is no 'one size fits all' approach to projecting postsecondary enrollment" due to "different cultural, financial, and political contexts" as well as "different challenges with increasing or decreasing enrollment trends", which supports the need for CSUN to have a custom course enrollment prediction system that is the most appropriate based on CSUN's particular contexts and trends.[7]

The enrollment planning situation at CSUN is unique, since at the time courses are scheduled it is unknown how many students will be classified (or trying to become classified) as majors or minors in Computer Science, Computer Engineering, Software Engineering, or Computer Information Technology, how many students are interested in which courses, how many students will be repeating courses due to failing grades, how many new transfer students will be enrolling, and how many continuing students will be returning. Currently, planning is done for two semesters into the future, with total student enrollment being estimated from past enrollment, and then estimating the number of course sections to offer by dividing the total course enrollment estimate by 25. While pre-semester surveys or filing enrollment plans could be instituted to help gauge course interest, student course interest could change for many reasons, rendering such endeavors likely high effort with low accuracy. Possible reasons that student reported future course enrollment plans could prove inaccurate are: students not considering course prerequisites, students not understanding and/or changing their

[7] (Reiss 2012)

priorities and goals, students relying on incomplete or invalid sources of information resulting in less than optimal course selection, students selecting courses due to one characteristic of the course – such as a humorous instructor – causing them to ignore other course or program requirements, students not being able to take desired courses due to time conflicts or overlap after the actual schedule is made, students not participating in the future course selection reporting process, or students not being able to take desired courses due to the final exam time after the actual schedule is made.[8] Unfortunately, there is no solution to knowing concretely before course scheduling how many students will fail courses the previous semester, how many new transfer students will enroll in the upcoming semester, or how many students will choose not to continue their education at CSUN. This means there is no way to gather concrete numbers for enrollment, so enrollment numbers must be predicted. To be useful the predicted numbers must be more accurate than the current educated guess method, with an acceptable bound being an error equal to or less than 25 students. The bound of 25 students is chosen since in 2015, 73.6% of classrooms at CSUN had enrollment between 20 and 49 students, making it an acceptable value for being off by one average class enrollment size.[9] Jacaranda Hall, where all Computer Science and Computer Information Technology classes are held, has 29 classrooms, where 22 have the capacity to accommodate 25 or more students.[10] This means that in the case that the predicted enrollment is underestimated by 25 students, the majority of classrooms available could handle the student overflow as a new course section. Contrarily, if the predicted enrollment is overestimated by 25 students, it would

[8] (Kardan, et al. 2013)
[9] (U.S. News 2016)
[10] (Rickes Associates Inc. 2015)

mean culling one course section.

Course planning is also complicated due to the large percentage of transfer and part-time students. CSUN is not a typical cohort style university where most students start as full-time, first-time freshmen then continue in lockstep with their peers until graduation. In the Fall of 2013, 81% of undergraduate students had an average age of 24 or younger and 19% had an average age of 25 or older.[11]

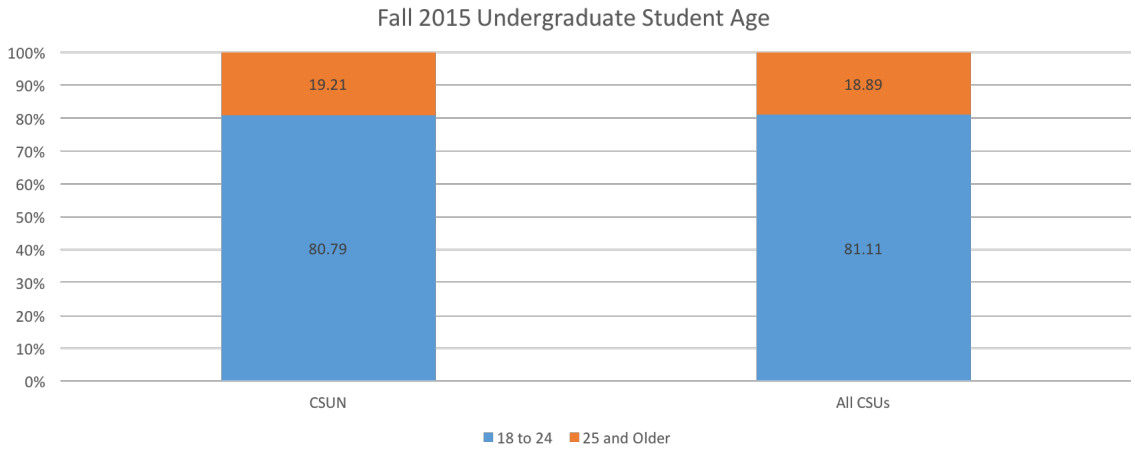

Figure 4 - Undergraduate Student Age

The percentage of first-time students who began a bachelor's degree program at CSUN in Fall of 2013 and returned in Fall of 2014 was 77% for full-time students and 40% for part-time students.[12]

[11] (National Center for Education Statistics 2016)
[12] (National Center for Education Statistics 2016)

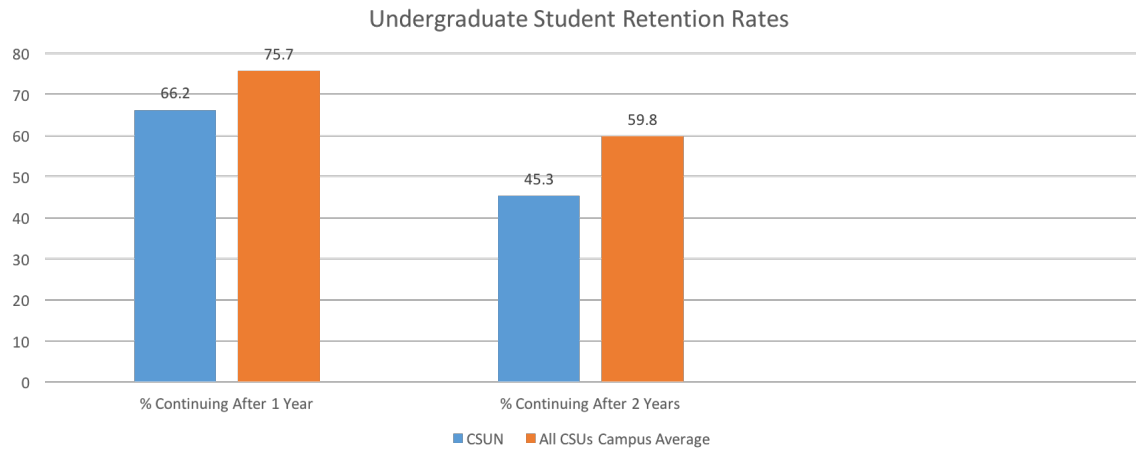

Figure 5 - Undergraduate Student Retention

This retention rate is much lower than that of University of California – Los Angeles (UCLA), which was 97% of full-time bachelor degree students returning and 62% of part-time students returning.[13] Graduation rates for undergraduate students at CSUN who began in the Fall of 2006 are 14% for students graduating after four years, 48% for students graduating after six years, and 55% for students graduating after eight years.[4]

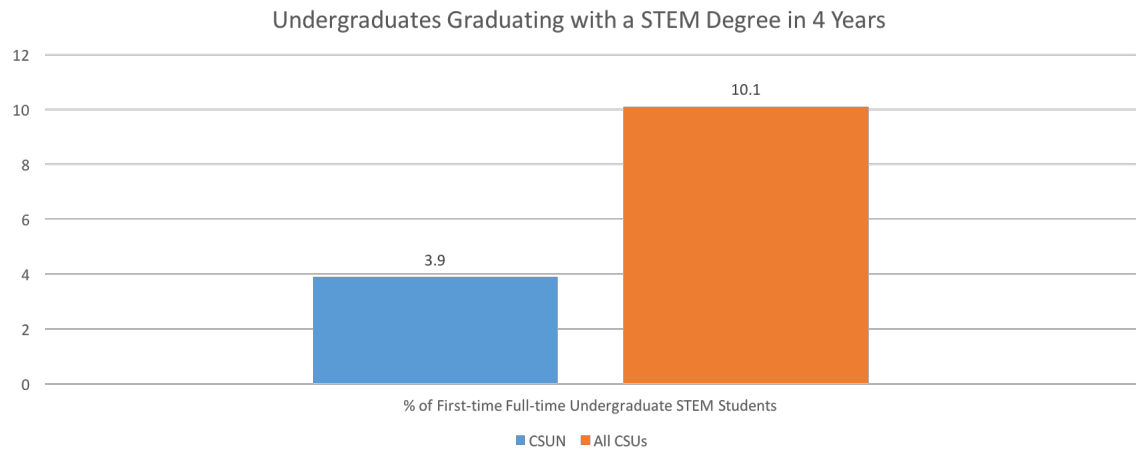

Figure 6 - STEM Undergraduate Student Graduation Rates

[13] (National Center for Education Statistics 2016)

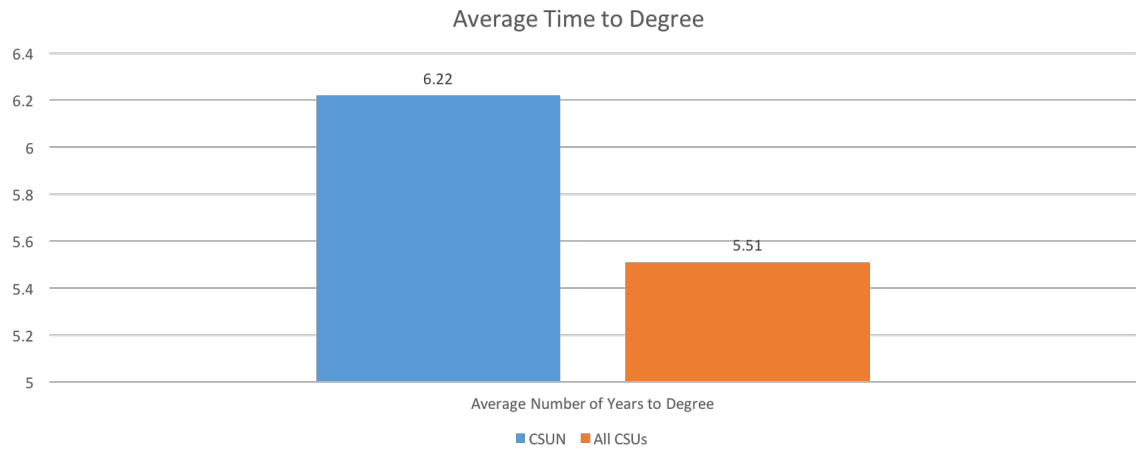

Figure 7 - Average Time to Degree

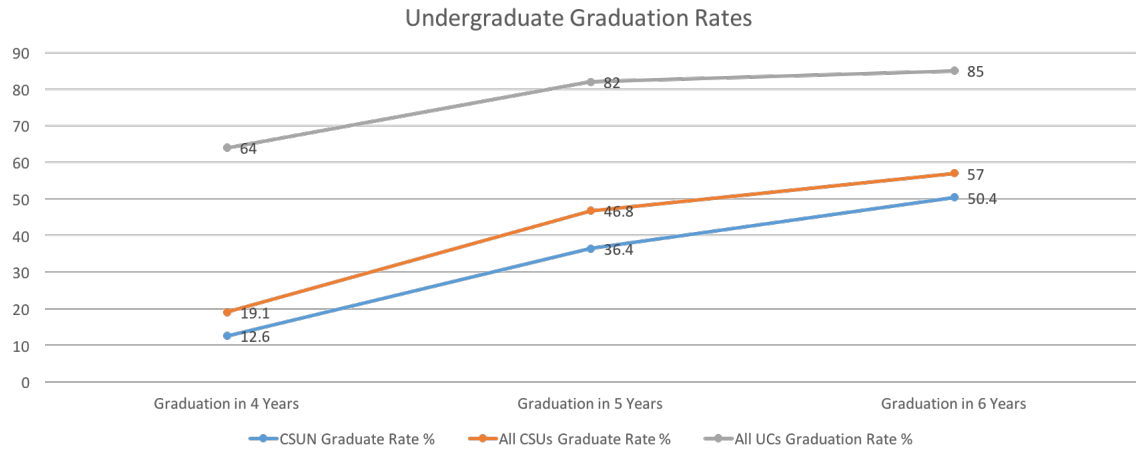

Figure 8 - Undergraduate Student Graduation Rates

The data supports the supposition that CSUN undergraduate students, when compared to other California State University (CSU) or University of California (UC) students, are less likely to continue their studies after their first year and more likely to be transfer students that are older, attend part-time, and take longer to graduate, making the CSUN student body less likely to be organized into groups of students that progress through educational programs at the same rate. For comparison, UCLA has a more traditional cohort program. In the Fall of 2015, UCLA's new undergraduate student enrollment was only 35% transfer students, with only 2% of the overall undergraduate student population

enrolled part-time.[14] In the Fall of 2013, 95% of undergraduate students had an average age of 24 or younger and only 5% had an average age of 25 or older.[4] Graduation rates for undergraduate students at UCLA who began in the Fall of 2006 are 71% for students graduating after four years, 92% for students graduating after 6 years, and 93% for students graduating after eight years.[4] Interpreting the data, UCLA students tend to enroll directly after high school and continue in their educational program full-time as a cohort until graduation. Due to this, UCLA's enrollment predictions likely employ forecasting models that would work well with a cohort oriented student body and that therefore would not be well suited for predicting enrollment at CSUN.

The problem of predicting course enrollment numbers is not unique to CSUN, yet there is a lack of information on how other universities deal with the issue and little research on how to deal with predicting course enrollment in general. While CSUN cannot rely on course enrollment approximation techniques that may be adequate for other universities, it is still useful to see how other institutions predict enrollment and to investigate whether those techniques could be adapted for CSUN. Information that would be useful is to know is the types of models being used by different institutions, how the models were chosen and why they were deemed appropriate for the institution's characteristics and needs, and how accurate the models have been in predicting enrollment. A search on the Internet for postsecondary enrollment projections in hopes of discovering how other colleges deal with enrollment predictions turns up varied information. The University of Colorado at Boulder uses “educated guesses surrounded

[14] (University of California, Los Angeles 2016)

by considerable error" to predict total university enrollment.[15] Their estimated margin of error for large, predictable groups such as undergraduate cohorts is between 2-3%. The inaccuracy of their predictions skyrockets to over 10% when projecting enrollment for smaller groups, such as out of state students or new international graduate studies students, which are more prone to fluctuation. The University of Wyoming conducted an Enrollment Project in order to predict university enrollment one to five years ahead.[16] They found that using four different models to predict the four different populations that comprise their overall enrollment worked well in that it only underestimated the actual enrollment count by 1%, which was likely due to the introduction of their Hathaway Scholarship Program.[17] The models used were linear regression for resident undergraduate enrollment, semi-log regression for regional undergraduate enrollment, linear regression for graduate student enrollment, and linear trend regression for all other undergraduate students.[18] The excel-spreadsheet based matrix-ratio model developed for course enrollment prediction at the University of Missouri-Columbia uses three years of historical student course enrollment data by major and student level to predict how many students in each major at each level (freshman, sophomore, junior, senior) will enroll in a specific course in the upcoming academic year. The model must be updated manually with the latest data to stay current, and its calculations require a ratio of students who will switch majors and the forecasted total university enrollment. The output of the model is 99% correlated with the actual course enrollment results of the university's population of

[15] (University of Colorado, Boulder 2015)
[16] (Aadland, Godby and Weichman 2007)
[17] (Aadland, Godby and Weichman 2007)
[18] (Aadland, Godby and Weichman 2007)

less than 6,000 students.[19] The University of Central Florida updated their enrollment projection model in 2015. The previous model in Excel was cohort based, relied on manual model updates for factors based on the judgment of the Institutional Research staff, and was based on student retention rates from the past ten years.[20,21] The mean absolute percentage error, a measure of how much the actual enrollment numbers varied from the model predictions, of their old method was approximately 0.5% for short term projections and 2% for long term projections.[22] The new University of Central Florida enrollment projection system uses the Java-based machine learning software suite Weka - the Waikato Environment for Knowledge Analysis - for modeling and R, a language and environment for statistical computing and graphics, for forecasting. This new system does not rely on manual updates.[23] The model uses two methods – neural network and regression with different variable selection methods – on the previous fall headcount, graduate students, first-time-in-college students, new students, and more. The forecast uses the non-seasonal Holt's linear method with damping and exponential smoothing. The error for the new projection framework is a little worse than the old method with a mean absolute percentage error of 2.2% when predicting enrollment for the following year. Oklahoma State University utilizes an Autoregressive Integrated Moving Average (ARIMA) methodology to predict enrollment with a mean absolute percentage error of 2.11%.[24] The University of Hawaii system also uses ARIMA methodology to predict enrollment with different models for each campus, with a mean absolute percentage error

[19] (Felts and Ehlert 2009)
[20] (Aadland, Godby and Weichman 2007)
[21] (Ramsey, Watts and Sklar 2015)
[22] (Aadland, Godby and Weichman 2007)
[23] (Ramsey, Watts and Sklar 2015)
[24] (Chen 2008)

of the models ranging from 2.391% to 6.771%.[25] It is useful to see which methods employed are more accurate, however the groupings used (such as new students, first time attending college students, etc.) may not be helpful for predicting specific course enrollment. It does, however, imply that maybe groupings such as new students and transfer students may add value to the model. Factors that were considered, such as high school graduation rates and unemployment rates are also enlightening. While high school graduation rates may not apply to course prediction, unemployment rates for the city of Los Angeles might and it is another option to consider.

The overall approach in predicting future course enrollment for CSUN undergraduate Computer Science courses is to first select an appropriate method. There are nine major methods of forecasting techniques: Subjective Judgment, Ratio Method, Cohort Survival Study, Markov Transition Model, Neural Network Model, Simulation Model, Time Series Analysis, Fuzzy Time Series Analysis, and Regression Analysis.[26] Each method was explored to determine which would be most suitable, either singularly or as part of a hybrid method, for the student enrollment prediction project.

The most basic method is Subjective Judgment, which is a qualitative method that makes predictions based on opinion and intuitive estimates of influential factors. The benefit of this method is that it does not require past data for analysis or a knowledge of statistics. It can be useful for short-term forecasts, since it tends to be the least accurate method and all forecast methods' prediction accuracy decreases as the time period being forecast increases. Since Subjective Judgment is not a quantitative method, it is not a candidate for predicting CSUN Computer Science undergraduate course

[25] (Institutional Research and Analysis Office for the University of Hawai'i System 2013)
[26] (Chen 2008)

enrollment programmatically.

A slightly more sophisticated method that is also useful for short-term forecasts is the Ratio Method. It is also referred to as the Grade Progression Ratio (GPR) method, and it expresses school enrollment patterns of a cohort of children over time as they advance in grade progression. It predicts a year in advance how many students will be promoted from one grade to the next by dividing the number of students in a particular grade by the number of students in the previous grade in the prior school year.[27] Due to CSUN's students not usually progressing through their curriculum as core groups, which cohort modeling assumes, the Ratio Method is not a good candidate for course enrollment predictions at CSUN.

In a Cohort Survival Study, the enrollment of cohort members is estimated by multiplying the number of students in the cohort during the previous time period by the cohort's survival rate. The assumption made in this technique when predicting more than one time-step into the future is that the survival ratio for a cohort does not change. Since CSUN's undergraduate students typically do not progress through the Computer Science or Computer Information Technology programs as a cohort, the Cohort Survival Study method is not a viable method for course enrollment predictions at CSUN.

A method that does not rely on cohorts is the Markov Transition Model, where matrices are used to represent the promotion rate from one grade to the next. Each matrix element a_{ij} represents the proportion of students enrolled in grade i in one time

[27] (University of Virginia Demographics Research Group 2014)

period who move to grade j in the next time period.[28] Since the matrix represents the overall promotion rate, it implicitly includes the grade repetition rate, dropout rate, and graduation rates. This model is memory-less, in that the next time period's enrollment numbers only depend upon the current state's enrollment numbers. This method would be useful if we could apply it to all students who have enrolled in an undergraduate Computer Science course. The Computer Science course taken would be the students' current state and then Markov transition modeling would determine the likelihood of each student moving to the other possible states, where moving to a possible state means enrolling in another undergraduate Computer Science course. This would allow each student's likely enrollment in each course to be predicted, with all of the student's combined forecasts giving the total enrollment prediction for each course. Unfortunately, data on each student who has taken an undergraduate Computer Science course is not available, so a Markov transition model cannot be practically applied to predicting undergraduate Computer Science course enrollment.

Similarly, Neural Network Models can be applied to use artificial intelligence to predict whether or not students will enroll in a particular course.[29] Neural Network Models simulate biological neural networks, where the artificial neurons decide which inputs are used and how they are translated into outputs. Many neurons combine to form a network, which consists of an input layer, intervening hidden layers, and an output layer. [30] The input to the neural network would be a student's past undergraduate Computer Science courses taken along with any other pertinent data such as their major,

[28] (Johnstone 1974)
[29] (Walczak 1998)
[30] (Hill, O'Connor and Remus 1996)

their class standing, and their grade point average. The output of the neural network would be whether or not the student would enroll in a particular course. Combining the likely course enrollment results for each student would yield the predicted enrollment totals for each undergraduate Computer Science course. As with the Markov Transition Model, the Neural Network Model approach on its own cannot be applied to predicting undergraduate Computer Science course enrollments due to a lack of available data on each individual student's past and current enrollment.

The Simulation Method, where a mathematical analog of student enrollment would be created to run what-if scenarios, would be of more use if there were a definite goal, such as offering a specific number of courses. Then, the Simulation Method would use the goal to work backwards to find the input scenario that leads to the goal being achieved. Typically, the Simulation Method is used for predictions in the physical sciences and not the social sciences, since the extreme complexity of social systems, such as a student body, makes it difficult to include all of the relevant factors in the mathematical model.[31] Since the goal of this thesis is predicting course enrollment, which is not controllable by the CSUN Computer Science department, and not reaching an actual enrollment goal, the Simulation Method is not appropriate.

Time Series Analysis assumes that the future depends on the present, and that the present depends on the past. This is certainly true of student course enrollment, since courses have prerequisites that must be met before a student can enroll. These prerequisites can include classification in the appropriate major or minor program of study, completing required coursework with a particular grade, not completing the course

[31] (Walonick 1993)

previously more than a certain number of times, and so on. A time series is a collection of observations made sequentially in time, for instance, the total course enrollment for undergraduate Computer Science classes from January 2010 to January 2016. There are many different statistical models that can be used to analyze a time series – from the simplest mean model where the predicted next value is equal to the historical sample mean to a more complex ARIMA model where the predicted next value is equal to a constant and/or a weighted sum of one or more of the recent values and/or a weighted sum of one or more recent error values. Combining Time Series Analysis with other methods, such as neural networks, is also a possibility, where the value at a time t is output as a forecast using the values at specific lags as input, where a lag is a period of time – for instance, 12 if the data is monthly. The neural network has at least one hidden layer with at least one node, where the inputs connect to each of the nodes in the hidden layer, with each connection having its strength denoted by a quantified weight. The prediction equation for computing a forecast at time t is then calculated as a function of weighted past observations. Neural network models are known to be effective for time series that are discontinuous or whose data cannot be approximated by a linear function.[32] Due to student enrollment having a temporal quality, the range of Time Series Analysis methods are appropriate for predicting future undergraduate Computer Science course enrollment.

Fuzzy Time Series Analysis is similar to classic Time Series Analysis, but it is more suitable for medium and long-range forecasts. Whereas classic Time Series Analysis constructs mathematical models from known past data in order to forecast the

[32] (Faraway and Chatfield 1998)

time series in the near future, Fuzzy Time Series Analysis is based on creating models that represent a well defined period of time using a long learning period in which the models are trained with various time series data.[33] Since CSUN only needs to predict undergraduate Computer Science course enrollment for the near future of three semesters ahead (Fall, Spring, and Summer semesters), Fuzzy Time Series Analysis is not an appropriate model.

The last method, Regression Analysis, predicts values for a chosen dependent variable, such as COMP 100 enrollment totals, from one or more chosen independent variables, such as tuition cost and graduation rates of local high schools. The Regression Model is developed to obtain insight into the behavior of both the dependent and independent variables over time. Using the Regression Model, a regression equation can be calculated mathematically, where values for the independent variables can be substituted into the equation to predict possible future values for the dependent variable.[34] Although the information for independent variables that could affect enrollment are limited, Regression Analysis is still an appropriate method for predicting undergraduate Computer Science course enrollment due to its simplicity and its inherent temporal nature.

When choosing a technology in order to create an undergraduate Computer Science course enrollment prediction tool, it should be capable of utilizing the modeling methods deemed appropriate, such as statistical models for Time Series Analysis and Regression Models. To generate predictions using appropriate data models, there exist many technologies that can be leveraged that are used both in industry and

[33] (Bocklisch and Päßler 2000)
[34] (Johnstone 1974)

academic research. Rexer Analytics conducts an annual data science survey of software tools used, which reveals R to be the tool used by most people. Out of 1,220 data scientists surveyed in 2015, approximately 76% use R and 21% use Weka.[35] Other products that are less popular than R but more than or equally popular as Weka are: IBM SPSS Statistics, SAS, Microsoft Excel Data Mining, Tableau, and IBM SPSS Modeler. The issue comes down to cost, and R and Weka are free while the other products listed are not free for academic institutions to use for non-learning purposes. Since the undergraduate course enrollment prediction tool project part of this thesis should be a viable tool for the Computer Science department to use and develop further, free software is preferable. Possible drawbacks of using free technology is a possible lack of current and future support which could hinder further development and maintenance of a tool that implements it. To minimize these impediments, the level of current support and likelihood of future development and maintenance of the free technologies are also considered. Any technologies chosen will also need to be able to read from a database, since the CSUN Computer Science department stores enrollment information in a SQL database. The predictions output from the different models should be in an easily comparable format, such as a spreadsheet, along with metrics that indicate their accuracy. Ideally, a portable, flexible, and well-known language should also be able to be be used to interface the different parts of the project so that the final tool is more easily extensible and maintainable. For example, the course enrollment prediction tool should be able to be developed further into a web-based and/or mobile application for ease-of-use. It should also be easily maintainable by undergraduate student interns whose programming

[35] (Rexer 2015)

language proficiency may be limited to C, C++, and Java. Since Java is popular, portable, web-and-mobile friendly, and lends itself to seeing statistical models as objects, it is the preferred language for codifying and interfacing the different components of the prediction tool. An investigation of R and Weka reveals that both are largely supported with extensive documentation available, are frequently used in academic research, and that they both meet all of the requirements for developing the course enrollment prediction tool. Since R and Weka are both popular in industry, government, and academia, are free of charge, can read data in from a SQL database, have forecasting statistical models available that can predict future values, can output model data in various formats such as CSV and Excel, and can interface with a variety of languages such as Java, they are appropriate technologies to utilize and compare for this thesis. R and Weka, their capabilities, and their chosen configurations are explored in the technical chapters of this thesis.

Development of the course enrollment prediction tool is iterative and componentized and follows the standard CRISP-DM framework. Testing is performed on each individual component, and then again on the overall tool. GitHub is utilized for free source code control management, so that different versions of committed code can be saved, retrieved, and compared. MySQLWorkbench is used to develop, save, retrieve, and test SQL queries that select previous course enrollment counts from the appropriate database tables in the correct format for creating statistical models before codification. VI is a free text editing tool used for editing source code. Weka's Explorer GUI and R's RStudio GUI are used for code development and testing of the individual Weka and R code components. In this way, sanity test cases for specific components are carried out to

ensure correctness before combining them. While there are automated test frameworks for Java, such as JUnit, how to use them to test statistical models is unclear. For R, the available debugging tools which will be utilized in RStudio are running the code line by line, outputting variables, and using built-in debugging commands such as traceback() to output the call stack.[36] Weka has an abstract unit test base class for classifiers that can be extended and used for unit testing and regression testing in JUnit, but it is not within a time-series forecasting context.[37] Instead, testing is done via sanity checking within the Weka Explorer GUI and through error handling in the Java code. The end product is a course enrollment prediction tool that is launched from an executable Java .jar file. The tool performs a database query to gather the latest enrollment data with which to create forecasting models with. The different models are then run to output predictions for two semesters into the future. Predictions made by each model along with the model's accuracy metrics are output in Microsoft Excel Workbook (.xlsx) format to the user's desktop. The user must have Java, R, and a program to read .xlsx files installed.

The following chapters review existing literature on enrollment prediction, discuss the technical implementation of the undergraduate Computer Science course enrollment prediction project, detail future work and maintenance of the project, and draw conclusions from the results of the project. There is little research on solving the specific problem of course enrollment, but there is literature on related research such as overall enrollment prediction, kindergarten through 12th grade enrollment prediction, predicting population sizes, and predictions for demand and usage. The existing research and related work on course enrollment prediction is examined, with different approaches

[36] (Feuerriegel 2016)
[37] (Bouckaert, et al. 2016)

being compared to determine a viable approach for the CSUN course enrollment project. The process of implementing the project utilizing the findings of past research is then explored. Details of the process from research to proof-of-concept to product through planning, design, development, and testing is laid out. Which technologies and models were chosen and why is explored, with overviews and definitions provided for completeness. How to maintain and enhance the course prediction tool is subsequently presented as a transfer of information for Computer Science department staff, including how to update the tool and its dependencies. Finally, the results of using the tool to predict future enrollment using available datasets is analyzed. The accuracy of the different models and how their results can be interpreted, explained, and improved is discussed.

REVIEW OF LITERATURE

There has been much research into predicting overall enrollment at postsecondary institutions, predicting kindergarten through 12th grade enrollment, and the different variables that affect those enrollments. For overall postsecondary enrollment prediction, time series models may possess a higher ability to capture the effects of influential variables. They may also, however, obfuscate the influence of different variables whose movements are correlated over time. It has been consistently found that the following variables have a statistically significant positive effect on postsecondary enrollment: family income, parents' educational attainment, student aid levels, and student's academic aptitude. The rate of return, or expected salary increase, for completing higher education is trickier to measure but is generally also found to have a positive effect on enrollment. Higher unemployment has also been noted to positively influence community college enrollment. Variables that have a statistically negative effect on postsecondary enrollment are: tuition levels – where the effect of rising tuition lowers as family income rises, and marriage rates – where the college enrollment rate for women decreases as the percentage of married women increases.[38] Unfortunately, it is difficult to get the data for these variables and translate the data into a form that can be incorporated into short-term enrollment prediction models. When predicting enrollment for kindergarten through 12th grade (K-12), the models typically used are the Cohort Survival (also referred to as Grade Progression Ratio) Method, the Ratio Method, and Regression Analysis Methods. Variables that affect K-12 enrollment are: internal policies

[38] (Ahlburg, McPherson and Schapiro 1994)

such as how old a child must be before enrolling in kindergarten and policies on scholastic retention and acceleration, external factors such as a significant increase or decrease in new home building, and other considerations such as the live birth rate.[39] There is research that claims that the Grade Progression Ratio Method, Markov Chain Models, and Cohort Flow Models can predict course enrollment for postsecondary institutions, but no models were developed or tested as part of the research to see if they could actually work.[40] The amount of research specifically to address postsecondary course enrollment predictions, as opposed to K-12 short-term enrollment or university level long-term enrollment, is sparse. Only four studies were found that specifically aim to address predicting student course selection at the postsecondary level. The approaches vary, and include Variable-Work Models, Neural Networks, the Analytical Hierarchy Process, and Adaptive Models.

Variable-Work Models forecast specific course enrollment where "work" is defined as the number of students who have yet to take a given course in an academic program. Its mathematical formulas can be adjusted to include students outside the program who could also enroll in the course. Balachandran and Gerwin explore three variations of the Variable-Work Model in their research: the Work Model, the Eligible Work Model Accounting for Prerequisites, and the Eligible Work Model with Program Requirements. The Variable-Work Models were developed in the context of graduate level studies, but can easily be adjusted for undergraduate programs by allowing for courses to be repeated. The Work Model predicts the number of students who will take a course based on how many students have not taken the course yet, the number of students

[39] (Pettibone and Bushan 1990)
[40] (Hopkins and Massy 1981)

who will likely drop out of the program before taking the course, the influx of new students who are not exempt from taking the course, and whether the course is required. In essence, the work model uses the conditional probability that a student will enroll in a particular, not-previously-enrolled-in course in order to forecast the total number of students who will enroll in that course. The Eligible Work Model Accounting for Prerequisites is similar to the work model, except that it takes into account whether students have completed prerequisites for the course first, and if they have not, those students are considered ineligible to enroll. Then, the conditional probability that an eligible student will enroll in a particular course is used to forecast the total number of students who will enroll in that course. The Eligible Work Model with Program Requirements makes forecasts based on the components of eligible and ineligible students. Students are determined to be eligible or ineligible based on their previously completed coursework towards a degree. Factors that are taken into account are: how many courses must be taken for a student to graduate, how many courses a student has taken without taking the required course including the direct and indirect prerequisites for the required course, how many courses a student has taken without taking the required course not including the prerequisites for the required course, and how many students are predicted to drop the required course. Then, the conditional probability that an eligible student will enroll in a particular course is calculated along with the conditional probability that an ineligible student will enroll in a particular course, in order to forecast the total number of students who will enroll in that course. For all models, there is an upper bound on the total number of courses a student can take that is set by the university. The Work Model and Eligible Work Model were tested against naïve

forecasts using data from 1971 for the School of Business Administration MBA program at the University of Wisconsin-Milwaukee and it was found that the Eligible Work Model showed some improvement over the Work Model, but not in all cases. Instances where the Eligible Work did not perform better than the Work Model were attributed to students not following the prerequisite structure for the program. Instances where neither the Eligible Work Model nor the Work Model performed well against the naïve forecast was when predicting enrollment for courses offered only during the day. The cause was deemed to be due to the day courses having lower student enrollment. Potential issues with the data used for these predictions is that they are trying to predict demand using enrollment numbers, which assumes that the enrollment numbers accurately reflect the courses' demand. For example, it assumes that when demand is higher than expected, more sections of the course are opened to accommodate the students, which may not necessarily be true. Also, it assumes that students follow the prerequisite structure and always take less than the maximum number of prescribed courses, which may not be the case. The authors surmise that despite data collection problems, the Variable-Work Models are still more accurate than alternative methods, and are most effective as a complement to using intuition instead of being a complete substitute for it.[41]

Neural Networks are inherently nonlinear mathematical models inspired by the functioning of biological neurons.[42] Multilayer Perceptrons (MPs) are a type of feed forward neural network consisting of an input layer, two or more hidden layers, and an output layer, with outputs of neurons in one layer being the inputs for the next layer. Kardan, Sadeghi, Ghidary, and Sani research how to utilize an MP to predict student

[41] (Balachandran and Gerwin 1971)
[42] (Hill, O'Connor and Remus 1996)

course enrollment by viewing course selection modeling as a multivariate, nonlinear problem that depends on many factors. The goal of their research is to: identify potential factors that affect student satisfaction of online courses, model the student course selection problem and fit a function to training data using neural networks, then use the discovered function to predict the total number of student enrollments for each online course. They note that no previous research on using neural networks to predict student course enrollment could be found. The data used in their research is from 714 online graduate courses in Information Technology and Management Engineering and Computer Networks Engineering programs for the online graduate college E-Learning Center of Amirkabir University of Technology from 2005 to 2012. The factors taken into account and used as inputs to the MP model for predicting student enrollment are: whether the course is considered interesting or useful, whether the instructor is considered good, how heavy the workload is, how difficult it is to earn a high grade, the type of course – required or elective, the day and time the course is offered, the number of courses with time conflicts, when the final exam will be administered, and initial student course selections before the course schedule is created. Fitting the function to the training data was performed through experimentation to find the optimal parameters such as how many hidden layers to use and how many neurons (or nodes) to use in each layer. Results from testing the trained MP revealed that including the collected student demand from surveys in the model led to greater precision. The accuracy of the MP model was compared to two traditional approaches and three different regression – or supervised learning – methods. The traditional naïve approaches used for comparison were: using intuition based off of previous enrollment numbers, and using intuition based off of

student course demand from surveys. The regression methods used for comparison were the following machine learning methods: Support Vector Regression, K-Nearest Neighborhood, and Decision Tree. For all tests, the MP model with student course demand added as input performed better than all of the naïve and machine learning methods. Even though the MP model did not perform as well without using the student course demands, it was also more accurate than the naïve and machine learning methods and is considered to be a viable alternative in the case that student course demand information is not available. The next accurate method after neural networks was Decision Tree, followed by K-Nearest Neighborhood, then Support Vector Regression. The conclusion is that although the MP model was the most accurate for the data examined, it may not be the most accurate for other institutes. It is advised that other machine learning techniques, including hybrid techniques, be tested and compared with neural network models using different inputs and configurations in order to find the most appropriate model for any university.[43]

The Analytic Hierarchy Process (AHP) was developed in the 1970s by Thomas Saaty. It is a structured decision making technique based on mathematics and psychology for handling complex, multi-dimensional and often conflicting preferences of individuals. While decision support techniques such as AHP have been utilized in medical, agricultural, and business management fields there existed no prior research on applying them to education for the express purpose of course enrollment prediction for given academic terms and years. Ognjanovic, Dragan, and Dawson in their research aim to develop and test an AHP framework that accurately predicts course enrollment for

[43] (Kardan, et al. 2013)

specific academic terms and years without requiring knowledge that is directly associated with a student's identity. This is notable since many forms of information that could aid in predicting course enrollment, such as student instructor reviews, must be kept confidential and are otherwise inaccessible due to government regulations of personally identifiable information. A benefit of the AHP process is that it takes into account that certain data may not be available nor accessible due to privacy concerns. The AHP framework itself is used to model different preference structures, and rank and assign weights to those preferences in order of their importance in student course selection. Preferences represented as layers in the AHP framework were: course characteristics, instructor characteristics, GPA value of the course, the time and day the course is offered, the demographic characteristics of the student, and the student demands. Course combinations that violate institutional rules concerning majors in programs and defined prerequisites, the number of required credits to maintain part-time or full-time status, the maximum number of students that can be enrolled in a given course, and overlap of course offerings, are not allowed. The AHP framework extracts the preferences which factor into student course selection from the university's data, represents the factors in a form suitable for processing by AHP, then assigns the values of the extracted factors to courses. Each individual student's course selection preferences are gauged based on the extracted factors. Then predictions about the student's course selections can be made as an AHP-based ranking of available courses. A benefit of AHP is being able to gain insight into factors that affect student enrollment. Ognjanovic, Dragan, and Dawson discovered that a student's grade point average relative to the courses they were considering taking was the greatest factor in determining which courses they would

ultimately enroll in. Test results using enrollment data for 1,061 students pursuing a Bachelor of Arts degree in Psychology that could enroll in any of 47 Psychology courses and 921 non-Psychology courses at a Canadian research university from 2007 to 2011 showed that despite available data being limited and complex, the predictions were relatively accurate and outperformed the neural network techniques used by Kardan, Sadeghi, Ghidary, and Sani when predicting overall student enrollment. However, the AHP-based approach was less accurate than the neural network techniques when predicting course selections for each individual student per academic term. The AHP and neural network models had approximately the same performance when predicting course selection for each individual student per academic year. Overall, the results from using AHP-based approaches to predict student enrollment overall was highly accurate, while for specific academic terms or years was relatively poor and not yet ready to be adopted to support university decision making processes. It was determined that student's interests and preferences are more important contributors to the accuracy of the predictive model than other data types, with course characteristics and GPA value for a course being the most important to students overall. Future research in using an AHP framework for course enrollment predictions is to consider more and different variables, such as contextual and environmental factors, that may make student course selection prediction more accurate.[44]

The Adaptive Model for course enrollment proposed by Kraft and Jarvis minimizes variability in student behavior by aggregating students into significant groups using broad categories instead of predicting on an individual basis. An appropriate

[44] (Ognjanovic, Gasevic and Dawson 2016)

grouping will demonstrate correlation between the grouping variable and enrollment in the course being modeled, with course enrollment rates for each group being significantly different from the rest of the student population. Conditional probabilities of the students in each group enrolling in specific courses is calculated using historical data. These group probabilities are used to estimate the group conditional probabilities for enrollment in the upcoming academic year. The estimated group conditional probabilities are multiplied by the predicted university enrollment for each group in the upcoming year to forecast future total course enrollment. The Adaptive Model relies on three different sets of data as input: the estimated new student populations, the current course enrollments, and historical enrollment data. The estimated new student population data is a targeted number of first-time and transfer students set by the admissions office and is not derived from historical data. Current course enrollments data provides the number and characteristics of currently enrolled students in each course along with the course's prerequisites. Possible student characteristics that can be used are declared major, SAT and ACT scores, class, gender, race, age, course load, number of semester hours for the current term, and the total number of credits earned so far. Pass and fail rates for the course are assumed to be unavailable and must be estimated. Historical enrollment data, combined with current enrollment data, is used to determine which student characteristics and course enrollment histories are significant predictors for modeling course enrollment. Any characteristics that increase variability of the model's predictions is not used. The Adaptive Model focuses on predicting Fall enrollment, but can be generalized to also include Winter, Spring, and Summer semesters by including the significant groupings for each semester to predict. Data used to test the Adaptive Model is from 4 undergraduate

Math courses at Clemson University from 2002 to 2004 that consistently had the highest enrollment numbers. Results of testing the model and comparing it against the actual historical enrollment numbers gives a 4% error rate, which is under the 5.3% error rate considered acceptable based on the weighted sum of the average first-time and average transfer error rates for Clemson University. The conclusion is that the Adaptive Model is sufficiently detailed and robust enough to predict courses with low enrollment or to predict for specific groups of students.[45,46]

[45] (Kraft 2007)
[46] (Kraft and Jarvis, An Adaptive Model for Predicting Course Enrollment 2005)

TECHNICAL IMPLEMENTATION

CRISP-DM is the Cross Industry Standard Practice for Data Mining that was developed in 1996 by a coalition of companies. Its purpose is to aid in converting business problems into data mining tasks, recommend appropriate data transformations and data mining techniques, provide a way to to evaluate the effectiveness of the results, and document projects. As of 2014, it remained the most used methodology with 43% of professionals surveyed utilizing CRISP-DM for their analytics, data mining, and data science projects.[47] The CRISP-DM framework breaks down the life cycle of a data mining project into 6 phases: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. In the business understanding phase, a project's objectives and requirements from a business perspective are fleshed out and then converted into a data mining problem definition with some basic understanding of what data is available. A preliminary project plan is drafted that achieves the stated objectives. Next, the data understanding phase starts with collecting and playing with available data to gain familiarity and understanding of the data, to identify data quality problems, and to detect subsets of the data that may be useful in uncovering hidden information. The four V's of data – volume, variety, velocity, and veracity should be considered. The data preparation phase follows, and it covers all tasks to construct the final dataset that will be used as input into the modeling tools. Tasks can be table, record, and attribute selection, data cleaning, construction of new attributes, as well as transformation of data to get it in the proper format for the modeling tools. Afterwards,

[47] (KDnuggets 2014)

the modeling phase consists of selecting and applying various modeling techniques and calibrating the model parameters to optimal values. Subsequently, the evaluation phase is about thoroughly evaluating the models and the steps that were taken to construct them, in order to confirm that they properly achieve the stated business objectives. Any business issues that have not been sufficiently considered should be found. Finally, in the deployment phase the knowledge gained from creating and running the models is organized and presented in a user-friendly way to the customer, for example, through reports and/or a repeatable data mining process.[48] Since creating models to predict undergraduate computer science enrollment is essentially data mining, the CRISP-DM framework was a logical framework to use.

The business understanding for this project is that the Computer Science department at CSUN needs an accurate, easy to use, easy to easy to interpret, easy to maintain, easy to enhance, free tool to aid in planning undergraduate course resources for one year in advance. The tool should create different types of predictive models to forecast total student enrollment for undergraduate Computer Science courses at CSUN. At least one model's predictions for each course should be accurate to within 25 students, or one class size, for the tool to be useful. Results of the tool should be the predictions generated by each model along with the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE) in an easily readable Excel workbook format, where the errors for each model of each course indicates which model will be likely to best forecast the total enrollment for that particular course. The tool should be easy to maintain and to enhance. This means requiring only an undergraduate Computer Science student level of

[48] (Wirth and Hipp 2000)

knowledge as well as only utilizing free technologies that are popular and well-documented. The preliminary project plan that was created is the proposal for this thesis.

The data understanding for this project is that the Computer Science department will maintain a separate database, scrubbed of all sensitive data, that contains historical enrollment information for CSUN. Considering the scale of the data, the volume can change, but will likely slowly increase over time since the Computer Science department probably would not put policies into place that would drastically lower or raise student enrollment in the short term. The velocity of the data should be low, with changes during the add/drop period for every new term and year with no changes to historical enrollment data for previous terms and years. Variety of the data should also be minimal, with there only being one form of data which resides in the prediction database. The veracity, or accuracy, of the data will be high since CSUN maintains accurate enrollment records. When constructing the prediction database, the number of years of historical data to import into and keep in the database can vary. This project will be developed using 19 semesters worth of data, from the spring semester which began on January 19th, 2010 to the spring semester which began on January 25th, 2016. Note that undergraduate Computer Science courses are only offered during the Fall, Spring, and Summer semesters and not during the Winter semester at CSUN. The database is named "prediction" and contains three tables: SA_CLASS_TBL, SA_STDNT_ENRLS, and SA_TERM_TBL. The queries to create the tables along with the tables' descriptions are shown in the following figures.

```
CREATE TABLE `SA_CLASS_TBL` (
  `STRM` int(11) NOT NULL DEFAULT '0' COMMENT 'the ERP/SIS assigned number',
  `CLASS_NBR` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `CLASS_STS` varchar(1) COLLATE utf8_unicode_ci DEFAULT NULL,
  `SUBJECT` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `CATALOG_NBR` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `DESCR` varchar(2048) COLLATE utf8_unicode_ci DEFAULT NULL,
  `CLASS_SECTION` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `DEPT_NAME` varchar(255) COLLATE utf8_unicode_ci DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`STRM`,`CLASS_NBR`),
  KEY `SUBJECT` (`SUBJECT`,`CATALOG_NBR`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
```

Figure 9 - prediction.SA_CLASS_TBL creation statement

Field	Type	Null	Key	Default	Extra
STRM	int(11)	NO	PRI	0	
CLASS_NBR	varchar(255)	NO	PRI		
CLASS_STS	varchar(1)	YES		NULL	
SUBJECT	varchar(255)	YES	MUL	NULL	
CATALOG_NBR	varchar(255)	YES		NULL	
DESCR	varchar(2048)	YES		NULL	
CLASS_SECTION	varchar(255)	YES		NULL	
DEPT_NAME	varchar(255)	YES		NULL	
created_at	timestamp	NO		CURRENT_TIMESTAMP	
updated_at	timestamp	YES		NULL	

Figure 10 - prediction.SA_CLASS_TBL description

```
CREATE TABLE `SA_STDNT_ENRLS` (
  `EMPLID` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `STRM` int(11) NOT NULL DEFAULT '0' COMMENT 'the ERP/SIS assigned number',
  `CLASS_NBR` varchar(255) COLLATE utf8_unicode_ci NOT NULL DEFAULT '',
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`EMPLID`,`STRM`,`CLASS_NBR`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
```

Figure 11 – prediction.SA_STDNT_ENRLS creation statement

Field	Type	Null	Key	Default	Extra
EMPLID	varchar(255)	NO	PRI	NULL	
STRM	int(11)	NO	PRI	0	
CLASS_NBR	varchar(255)	NO	PRI		
created_at	timestamp	NO		CURRENT_TIMESTAMP	
updated_at	timestamp	YES		NULL	

Figure 12 - prediction.SA_STDNT_ENRLS description

```
CREATE TABLE `SA_TERM_TBL` (
  `STRM` int(11) NOT NULL DEFAULT '0' COMMENT 'the ERP/SIS assigned number',
  `term` varchar(16) COLLATE utf8_unicode_ci DEFAULT NULL COMMENT 'the human friendly ID',
  `description` varchar(2048) COLLATE utf8_unicode_ci DEFAULT NULL,
  `begin_date` datetime DEFAULT NULL,
  `end_date` datetime DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`STRM`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci COMMENT='This table is derived from the ERP/SIS system'
```

Figure 13 - prediction.SA_TERM_TBL creation statement

Field	Type	Null	Key	Default	Extra
STRM	int(11)	NO	PRI	0	
term	varchar(16)	YES		NULL	
description	varchar(2048)	YES		NULL	
begin_date	datetime	YES		NULL	
end_date	datetime	YES		NULL	
created_at	timestamp	NO		CURRENT_TIMESTAMP	
updated_at	timestamp	YES		NULL	

Figure 14 - prediction.SA_TERM_TBL description

Some fields require more explanation than what is available through the table creation and table description results. First, fields in the SA_CLASS_TBL are explored, where each record in the table corresponds to a particular course in a specific academic term and year. The STRM field is a number generated by CSUN's enterprise resource planning (ERP) and student information system (SIS). It is unique for a particular academic term and year. For example, an STRM value of 2103 indicates a course was offered in the spring term of 2010. As a primary key in all three tables, it is the only field that can join all three tables. The field CLASS_NBR is a primary key in both the SA_CLASS_TBL and SA_STDNT_ENRLS tables, and combined with the STRM field can join the two tables on specific courses, sections, academic term, and year. It is a random number that is unique for each course and section offered in each academic term and year. For example, in Spring term of 2010, for course COMP 100, class section 04 has a CLASS_NBR value of 14381 while class section 05 has a CLASS_NBR value of 18871. No other courses in the Spring term of 2010 have those CLASS_NBR values. CLASS_STS is class status which can be X, A, T, or S. We are only concerned with classes with status A, which means the courses are/were active and not cancelled. SUBJECT is a multiple column index on the SUBJECT and CATLOG_NBR fields that may not be unique. The SUBJECT field values we are interested in are COMP and CIT, which correspond to Computer Science and Computer Information Technology courses

respectively. CATALOG_NBR, contrary to its name, is a string and can contain letters. Examples of CATALOG_NBR values for COMP courses are 110 and 110L, where 110 is the catalog number for the Introduction to Algorithms and Programming course and 110L is the Introduction to Algorithms and Programming course's required lab. Since we are only interested in undergraduate student computer science and computer information technology courses, the first character in the string must be extracted and checked that it is less than five. This is since 500 and 600 level courses are considered graduate level courses. DESCR is a description of the course, and for the initial project is not being used. However, it may provide extra data for enhancements, which is discussed in the future work chapter. CLASS_SECTION is a number assigned to a course to distinguish it from multiple offerings of the same course in the same academic term and year. Course sections are numbered sequentially beginning at one, however, sometimes there are missing CLASS_SECTION numbers and the reason for this is unknown. For example, for all COMP 100 courses in the fall of 2010 the database shows course sections 01 through 32, yet there are random section numbers missing such as 02, 05, 14, 16, and more. Since number of sections appears to supposed to indicate the number of courses offered, and we are not concerned with predicting the number of sections this information is not used in the initial project, but could be revisited as aiding in future enhancements. The DEPT_NAME field is the academic department name. For our purposes, we are only concerned with data where DEPT_NAME has value "Computer Science". This is because DEPT_NAME equal to "Electrical and Computer Engr" corresponds to SUBJECT equal to ECE and DEPT_NAME equal to "Engineering Computer Science" corresponds to SUBJECT equal to CECS. The fields created_at and updated at are

timestamps that aid in telling how fresh the data is. For this project, we know when the database was created and it was not going to be updated, so these fields were not used. They may have applications for future enhancements to take into account the freshness of the data. The only field in table SA_STDNT_ENRLS that has not been explained yet is the primary key EMPLID, which is an obfuscated student ID. Each record in the SA_STDNT_ENRLS table represents a course taken by a particular student in a specific academic term and year. Exploring the SA_TERM_TBL, each record corresponds to an academic term and year. Not every record in this table is historical, since there are records in this table for two years (or 6 semesters) in advance. The field term is a descriptive string containing the academic term and year, for example "Fall-2016". It can be used to group courses of the same academic term and year together. The field description is a longer description than the term field, for example "Fall Semester 2016". It is not useful for predicting course enrollment since essentially the same information is available via the term field. The begin_date and end_date fields represent when the academic term began and ended. They are datetime types in 'YYYY-MM-DD HH:MM:SS' format which can also be null.

The data preparation phase uses the results of exploring the three tables in the prediction database. Invalid values, such as null or future values, should be discarded, while values that will be used should be in the format required for processing by Weka and R. Only grabbing valid data is the responsibility of the database query, so that the Weka and R java code can focus on processing the data and not debugging it. Since Weka and R libraries can read data from CSV files, and the MySQL prediction database can output data into CSV files, data formatting is not an issue. The format chosen just has

to be communicated when it is processed. For instance, since JDBC will be used in the Java code to connect to the database, and JDBC changes the datetime field format to 'DD-MMM-YYY HH:MM:SS', this must be communicated to the Weka and R libraries. The database query is built and tested on small datasets first, for example, Summer semester data, where the results can be calculated by hand and then checked against the programmatic result. Since the typical course name consisting of the subject concatenated with the catalog number is not an existing database field, it must be constructed on the fly using a prepared statement. Also, since there is no total enrollment field that exists, the number of students who took each course each academic term and year must be calculated in the database query. Since it is unknown whether including the summer term data will aid in prediction accuracy or cause higher variance and less accuracy, both scenarios are tested and the more accurate one will be used in the final code.

```
/* QUERY FOR ALL COURSES NO SUMMER TERM */
USE prediction;
SET group_concat_max_len=990000;
SET @sql = NULL;
SELECT
  GROUP_CONCAT(DISTINCT
    CONCAT(
      'COUNT(DISTINCT(CASE WHEN sct.SUBJECT="', sct.SUBJECT, '" AND sct.CATALOG_NBR="', sct.CATALOG_NBR,
      '" THEN sse.EMPLID END)) AS `', sct.SUBJECT, sct.CATALOG_NBR, '_Enrollment_Total`'
    )
  ) INTO @sql
FROM SA_CLASS_TBL sct
WHERE
sct.DEPT_NAME="Computer Science" AND
sct.CLASS_STS="A" AND
CAST(LEFT(sct.CATALOG_NBR, 1)AS UNSIGNED)<5;

SET @sql = CONCAT('SELECT stt.BEGIN_DATE AS StartDate, ', @sql,
' FROM SA_CLASS_TBL sct LEFT JOIN SA_STDNT_ENRLS sse ON sct.STRM=sse.STRM AND sct.CLASS_NBR=sse.CLASS_NBR
LEFT JOIN SA_TERM_TBL stt ON stt.STRM=sct.STRM
WHERE
stt.BEGIN_DATE<NOW() AND
sct.DEPT_NAME="Computer Science" AND
sct.CLASS_STS="A" AND
stt.TERM NOT LIKE ''Summer%'' AND
CAST(LEFT(sct.CATALOG_NBR, 1)AS UNSIGNED)<5
GROUP BY stt.BEGIN_DATE, stt.TERM');

PREPARE stmt FROM @sql;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
```

Figure 15 - Database query with summer term data removed

```
/* QUERY FOR ALL COURSES FOR ALL SEMESTERS */
USE prediction;
SET group_concat_max_len=990000;
SET @sql = NULL;

SELECT
  GROUP_CONCAT(DISTINCT
    CONCAT(
      'COUNT(DISTINCT(CASE WHEN sct.SUBJECT="', sct.SUBJECT, '" AND sct.CATALOG_NBR="', sct.CATALOG_NBR,
      '" THEN sse.EMPLID END)) AS `', sct.SUBJECT, sct.CATALOG_NBR, '_Enrollment_Total`'
    )
  ) INTO @sql
FROM SA_CLASS_TBL sct
WHERE
sct.DEPT_NAME="Computer Science" AND
sct.CLASS_STS="A" AND
CAST(LEFT(sct.CATALOG_NBR, 1)AS UNSIGNED)<5;

SET @sql = CONCAT('SELECT stt.BEGIN_DATE AS StartDate, ', @sql,
' FROM SA_CLASS_TBL sct LEFT JOIN SA_STDNT_ENRLS sse ON sct.STRM=sse.STRM AND sct.CLASS_NBR=sse.CLASS_NBR
LEFT JOIN SA_TERM_TBL stt ON stt.STRM=sct.STRM
WHERE
stt.BEGIN_DATE<NOW() AND
sct.DEPT_NAME="Computer Science" AND
sct.CLASS_STS="A" AND
CAST(LEFT(sct.CATALOG_NBR, 1)AS UNSIGNED)<5
GROUP BY stt.BEGIN_DATE');

PREPARE stmt FROM @sql;
EXECUTE stmt;
DEALLOCATE PREPARE stmt;
```

Figure 16 - Database query with no term data removed

The modeling phase begins with comparing the available Time Series Models in Weka and the R forecast package. We use Time Series Models since the enrollment predictions and supporting data are innately temporal. The Time Series Models available in Weka are: Gaussian Processes, Linear Regression, Multilayer Perceptron, and SMOreg. The Time Series Models available in the R forecast package are: ARIMA, ETS, RWF, Meanf, Naïve, SNaïve, HoltWinters, DSHW, BATS/TBATS, LM/TSLM, StructTS, and NNetar.

The Gaussian Processes (GP) model in Weka for time series analysis implements GPs for regression without hyperparameter tuning, where hyperparameters are defined as the parameters of the covariance, or kernel, function. Any missing values in a dataset are replaced with global mean/mode values. A GP is a stochastic, or random, process that can perform probability distributions over functions. When used for regression, GPs infer the process, or function, f which maps inputs to outputs such that y_*

$= f(x_*) + \eta$ by learning from a dataset of input-output pairs $\{x_i, y_i\}_{\text{for } i=1:N}$ where output y_i is real-valued and η is a an additive noise process. Bayesian inference is used to calculate the probability distribution of y_* for some x_*, which is used to rank the likelihood of each possible function f. In the context of a time series, the set of possible inputs is the academic term and year a specific course is offered at. Since this data is precisely known there is no noise factor. The set of possible outputs is the total enrollment for the specific course for that academic term and year. Using the function f, a GP can predict the value of output y_* at any new test input x_*. This means a GP can predict the the total enrollment for a course given the term and year the course is offered. Benefits of using GPs are that: they can quantify uncertainty in predictions resulting from noise and errors in parameter estimation, they can model arbitrary non-linear functions of the input points, they use kernels that can take advantage of structure in the data, and they lead to simple and straightforward linear algebra implementations.[49,50,51,52,53] Since Gaussian Processes were chosen to as one of the models to compare for forecasting web site visits research, it is one of the models selected for this project.[54]

[49] (Roberts, et al. 2012)
[50] (Ghahramani 2011)
[51] (Frigola-Alcalde 2015)
[52] (Póczos 2009)
[53] (R. R. Bouckaert, et al. 2016)
[54] (Napagoda 2013)

Time Series Model	Description	Usage
Gaussian Processes (GPs)	• Given a dataset of input-output pairs $\{x_i, y_i\}$ for i=1:N and y real-valued, infers a function f such that $y^* = f(x^*)+\omega$ where ω is an additive noise process • Bayesian inference is used to calculate the probability distribution of y^* for some x^* which is used to rank the likelihood of each possible f • After the best f is found, the GP can predict the value of output y^* for any new test input x^*	• Forecast web site visits (Napagoda 2013)

Table 1 - Gaussian Processes Model

Weka's Linear Regression model for time series analysis attempts to model the relationship between two variables, in this instance total course enrollment Y and an academic semester and year X, by fitting a linear equation of the form Y = a + bX to the observed data of enrollment over time. In the linear equation the slope of the line is b, the x-intercept is a, the explanatory variable is X, and the dependent variable is Y. Weka automatically uses uses the Akaike criterion to measure the relative quality of the linear regression models so the best model is selected.[55] Since Linear Regression has been used for solving similar problems in research, such as predicting parking lot occupancy in Bolzano Italy, forecasting web site visits, forecasting energy consumption for Hvaler Norway, predicting the market value for residential buildings for real estate appraisers, and predicting overall enrollment at the University of Wyoming, it is also one of the models selected for this project.[56,57,58,59,60]

[55] (R. R. Bouckaert, et al. 2016)
[56] (Reinstadler, et al. 2013)
[57] (Napagoda 2013)
[58] (Gvaladze 2015)

Time Series Model	Description	Usage
Linear Regression	• Fits a linear equation Y = a+bX to the set of data that was observed over time • The slope of the line is b • The x-intercept is a • The explanatory variable is X • The dependent variable is Y	• Forecast web site visits (Napagoda 2013) • Predict parking lot occupancy (Reinstadler, et al. 2013) • Forecast energy consumption (Gvaladze 2015) • Predict market value of residential buildings (Graczyk, et al. 2009) • Forecast overall enrollment at the University of Wyoming (Aadland, et al. 2007)

Table 2 - Linear Regression Model

Time series analysis in Weka using Multilayer Perceptrons (MPs) uses the backpropagation supervised learning technique to train the neural network model. The MP has an input layer, hidden layers, and an output layer that are all fully connected. By default, the number of hidden layers is selected with the number of nodes total contained in the hidden layers equal to the mean value of the input and output layers. The input to the MP is the historic course enrollment data to predict on and the output is the predictions of future course enrollment. Since MPs along with other neural networks have been used previously in research to predict online graduate school course enrollments, forecast web site visits, predict Romanian stock exchange rates, predict the market value for residential buildings for real estate appraisers, and predict overall enrollment at the University of Central Florida, it is also a model selected for this project.[61,62,63,64,65]

[59] (Graczyk, Lasota and Trawiński 2009)
[60] (Aadland, Godby and Weichman 2007)
[61] (Kardan, et al. 2013)
[62] (Napagoda 2013)

Time Series Model	Description	Usage
Multilayer Perceptron / Neural Networks	• Is a group of interconnected nodes that form an input layer, hidden layer(s), and an output layer • Layers are fully connected, with the connection between neurons weighted to signify connection strength • Mimics the biology of the brain, using backpropagation to learn • Learning modifies the connection strength weights according to the input patterns and the backwards propagated error	• Forecast web site visits (Napagoda 2013) • Predict online grad school course enrollments (Kardan, et al. 2013) • Forecast stock exchange rates (Nemeş, et al. 2013) • Predict market value of residential buildings (Graczyk, et al. 2009) • Forecast overall enrollment at the University of Central Florida (Reiss 2012)

Table 3 - Multilayer Perceptron Model

The Sequential Minimal Optimization Algorithm for Support Vector Machine Regression (SMOreg) model for time series analysis in Weka implements a support vector machine for regression. The default optimization is performed using the RegSMOImproved algorithm. The goal of support vector machine regression is to use real-valued training data $\{(x_1,y_1),\ldots,(x_n,y_n)\}$ to find a function $f(x) = \langle\omega,x\rangle + b$ for ω in the space of input patterns and b in the space of real numbers, where $\langle .,. \rangle$ denotes the dot product. The function f(x) should be as flat as possible by minimizing $\langle\omega,\omega\rangle$ and should have at most ε deviation from the actual targets y_i for all of the training data. Sequential minimal optimization (SMO) is applied to iteratively select working sets of size two and analytically optimize f(x) with respect to them, until all of the training examples satisfy the optimality conditions.[66,67,68] For predicting course enrollments, x corresponds to the

[63] (Nemeş and Butoi 2013)
[64] (Graczyk, Lasota and Trawiński 2009)
[65] (Reiss 2012)
[66] (Bouckaert, et al. 2016)
[67] (Shevade, et al. 2000)

academic term and year, y corresponds to the course enrollment, and ω is in the space of input patterns $\Re^d$, where d is the number of distinct Computer Science (COMP) and Computer Information Technology (CIT) courses in the database. Since SMOreg was one of the models selected in previous research to predict web page visits, and SVM was selected to predict the market value for residential buildings for real estate appraisers and to predict online grad school course enrollments, it is also selected as one of the models to compare in this project.[69,70,71]

Time Series Model	Description	Usage
Sequential Minimal Optimization algorithm for support vector machine Regression (SMOreg)	• Use real-valued training data $\{(x_1,y_1),\ldots,(x_n,y_n)\}$ to find a function $f(x) = \langle\omega,x\rangle + b$ • ω is in the space of input patterns, b is in the space of real numbers, and $\langle\omega,x\rangle$ denotes the dot product between vectors ω and x • f(x) should be as flat as possible by minimizing $\langle\omega,\omega\rangle$ and should have at most ρ deviation from y_i for all training data • Sequential minimal optimization is used to iteratively select sets of training data of size 2 and analytically optimize f(x) with respect to the sets, until all training examples satisfy optimality conditions	• Forecast web site visits (Napagoda 2013) • Predict market value of residential buildings (Graczyk, et al. 2009) • Predict online grad school course enrollments (Kardan, et al. 2013)

Table 4 - SMOreg Model

The auto.arima (AutoRegressive Integrated Moving Average) model in R's forecast package uses the best ARIMA model for a univariate time series. An ARIMA model can be thought of as a filter that tries to separate a signal from noise, and then uses the signal to forecast future values. For predicting course enrollments, the predicted enrollment for a course is a sum of a constant, lagged values for the course

[68] (Smola and Schölkopf 2004)
[69] (Napagoda 2013)
[70] (Graczyk, Lasota and Trawiński 2009)
[71] (Kardan, et al. 2013)

enrollment (AR terms), and lagged errors for the course enrollment (MA terms).[72] Since the ARIMA model has been used in previous research to: determine whether unemployment rate and tuition predict total credits enrolled in at Monroe Community College, predict total campus enrollment at Oklahoma State University and University of Hawaii, and predict energy consumption for Hvaler Norway, it is also one of the models selected for comparison for this project.[73,74,75,76]

Time Series Model	Description	Usage
AutoRegressive Integrated Moving Average (ARIMA)	• Acts like a filter that tries to separate signal from noise • The signal is then used to forecast future values	• Determine whether tuition and unemployment affect community college enrollment (DeLeeuw 2012) • Forecast energy consumption (Gvaladze 2015) • Predict overall enrollment at Oklahoma State University (Chen 2008) • Predict overall enrollment at the University of Hawaii campuses (University of Hawaii 2013)

Table 5 - ARIMA Model

R's forecast package's Exponential Smoothing State Space (ETS) model automatically chooses the best ETS model for the given time series data. Models have a trend component (none, additive, multiplicative, or damped) and a seasonal component (none, additive, or multiplicative). The different models each have the property that forecasts are weighted averages of past observations, with recent observations weighted more heavily than older observations. The observation weights decrease exponentially as

[72] (Hyndman 2016)
[73] (DeLeeuw 2012)
[74] (Chen 2008)
[75] (Institutional Research and Analysis Office for the University of Hawai'i System 2013)
[76] (Gvaladze 2015)

the observations get older.[77] Each model deemed appropriate from the 24 possible models have their parameters optimized, are tested, and the optimized model that is determined to be the best predictor by the Akaike information criterion is selected. Since ETS in previous research was found to perform well on short-term forecasts (up to 6 periods ahead) on Makridakis forecasting competition data, it is also one of the models selected for comparison for this project.[78,79]

Time Series Model	Description	Usage
Exponential Smoothing State Space Model (ETS)	• ETS term encompasses a variety of models, each having trend and seasonal components • Trend components: none, additive, multiplicative, damped • Seasonal components: none, additive, multiplicative • Each model has the property that forecasts are weighted averages of past observations, with recent observations weighted more heavily • Observation weights decrease exponentially with age	• Predictions on Makridakis forecasting competition data (Hyndman 2016)

Table 6 - ETS Model

The Random Walk Forecast (RWF) model in R's forecast package calculates predictions using a random walk with drift applied to time series data. The model is described by the function $Y_t = c + Y_{t-1} + Z_t$ where Y_t would be the current course enrollment, Y_{t-1} would be the previous academic term's course enrollment, c is a drift constant, and Z_t is iid (independent and identically distributed) noise.[80] The RWF model is simple, where forecasts increase or decrease over time by the average change

[77] (R. J. Hyndman, Forecasting Based on State Space Models for Exponential Smoothing 2002)
[78] (Hyndman 2016)
[79] (Hyndman, et al. 2002)
[80] (Hyndman 2016)

seen in the historical data. Since this method is supposed to be good for short-term forecasting, and since it is a close approximation to the currently used method at CSUN, it will also be selected as a model to compare for this project.[81]

Time Series Model	Description	Usage
Random Walk Forecast (RWF)	• Calculates predictions using a random walk with drift • Forecasts increase or decrease over time by the average change seen in the historical data • Model is characterized by the function: $Y_t = c+Y_{t-1}+Z_t$ • Y_t is the current value, Y_{t-1} is the previous value in time, c is a drift constant, and Z_t is independent and identically distributed noise	• Good for short-term forecasts • Is a good approximation of CSUN's current course enrollment prediction scheme, so can be used for comparison purposes

Table 7 - RWF Model

Models in R's forecast package that will not be used are: Meanf, Naïve, SNaïve, HoltWinters, DSHW, BATS/TBATS, LM/TSLM, StructTS, and NNetar. In the Meanf model all forecasts are equal to the mean of the historical data. The Naïve model sets all forecasts to the value of the last observation, while SNaïve sets each forecast to the last observed value from the same season of the previous time period. Since course enrollment tends to increase and not remain stationary, these methods are not likely to be accurate and therefore were not chosen for this project. HoltWinters, DSHW (Double Seasonal Holt-Winters), BATS/TBATS (Exponential Smoothing State Space model with Box-Cox, for High Frequency Data, Transformation, ARMA errors, Trend, and Seasonal Components) are not selected since they are similar models to ets and ets is considered to be a better forecasting model. The LM/TSLM (Linear Model with possible Time Series

[81] (Dalrymple and King 1981)

components) is not chosen due to it being too similar to the already chosen LinearRegression model in Weka. StructTS (Structural Time Series model) is a Linear Gaussian State-Space model for univariate time series which is not selected due to the GaussianProcesses model in Weka already being selected. The NNetar (Neural Network Time Series Model) is a feed-forward neural network which is not selected due to the Multilayer Perceptron neural network in Weka already being chosen.[82]

In the evaluation phase, the chosen models of: Gaussian Processes, Linear Regression, Multilayer Perceptron, SMOreg, ARIMA, ETS, and RWF are created programmatically using a 95% confidence interval, frequency set to the correct seasonality (2 or 3), steps to predict set to the correct number per the test, and default settings for all other model parameters. Their total enrollment predictions for each undergraduate computer science course for the upcoming three semesters, along with Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) measurements, are output to an excel workbook. The measurements of MAE and RMSE were chosen since they are indicators of how close model predictions are to actual data and because they were available in both Weka and R's forecast package. MAE is the average of the absolute errors, where the error is the difference between the predicted value and the actual value. In the MAE measurement, all of the individual differences are weighted equally in the average. RMSE sums the squares of the difference between the predicted and actual values, averages them, then takes the square root of the average. Squaring the errors before they are averaged causes larger errors to have a greater effect on increasing the RMSE. This weighting effect is useful when large errors are especially undesirable,

[82] (Hyndman 2016)

as in the case of predicting course enrollments. Together, MAE and RMSE can find the variation in the errors of a set of forecasts. The RMSE will always be greater than or equal to the MAE, with a large difference between RMSE and MAE corresponding to a greater variance in the individual errors of the predictions. If RMSE is equal to the MAE, then all of the forecasting errors are of the same magnitude. The lower the RMSE and MAE values are, the smaller the error is for the forecasts, and therefore the better the model is at predicting future course enrollments.

Name	Formula	Description
Mean Absolute Error (MAE)	$\frac{1}{N}\sum_{i=1}^{N}\lvert f_i - o_i\rvert$	N = the number of forecasted/observed value pairs f = the forecasted value o = the observed (actual) value * Individual differences are weighted equally in the average
Root Mean Square Error (RMSE)	$\sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_i - o_i)^2}$	N = the number of forecasted/observed value pairs f = the forecasted value o = the observed (actual) value * Larger errors have higher weights

Table 8 - Performance Measures

Weka and the R forecast package are used to create the models as a Java program. Weka (Waikato Environment for Knowledge Analysis), is a free, non-commercial, open source suite of machine learning algorithms written in Java for data mining tasks. Weka can be used via a graphical user interface, a command line interface, or Java code. It also contains tools for data pre-processing, classification regression, clustering, association rules, and visualization. The Weka time series modeling environment must be installed separately using the package manager. R is a programming language and software environment for statistical computing and graphs that is free under the GNU general

public license. It is maintained and distributed by an international team of statisticians and computer scientists, with packages available from CRAN (the Comprehensive R Archive Network). The forecast package contains methods and tools for displaying and analyzing univariate time series forecasts. R packages tend to be mostly written in R with C/C++, but to have the project be more cohesive, understandable, and portable it was decided to call R from java using the rJava package and the JRI Java/R interface it includes. Then, the project code can all be written in java files. The project code logic is to first connect to the CSUN prediction database to get the course start date and course enrollment total for all undergraduate COMP and CIT courses offered in all previous academic terms and years. The data, along with the data column headers, is then output to a temporary CSV file for Weka and R to read from when generating models. Any courses that have total enrollment of zero for all semesters are purged, since creating a model to predict the enrollment for those courses will fail. The program environment is set including paths to the input data, paths to the output data, and what format the input data is in, including the format of the date field. The variables created from the data are utilized to predict the total enrollment of each course for a set number of steps into the future for each selected model, and then output those results along with the MAE and RMSE to a spreadsheet. Since Summer semester data is different from Fall and Spring semester data due to fewer courses being offered and fewer students enrolling, tests were run on all data and also with only Fall and Spring semester data to see if removing Summer data resulted in more accurate predictions. When running tests on all three semesters of data, the lag time and number of semesters in the future to predict is set to three. Without Summer semester data, the lag time and number of semesters in the future

to predict is set to two. The lag time sets the periodicity of the data, for example, for monthly data 12 lag steps would make sense and for hourly data 24 time steps would be logical. When creating models, the minimum amount of data required is anywhere from two to three times the lag time. The different tests that were conducted involved holding out different amounts of data from model creation to see which models predicted the held-out data the most accurately, and if that accuracy was increased by removing and not predicting on summer term data. Historical data that is used for model creation is referred to as training data, while historical data that is withheld to compare with the model's predictions is called holdout data. The standard amount of holdout data for testing is one-third, or 33%, of the available historical data. Available historical data in the prediction database is 19 semesters total, with 13 of them being fall or spring terms and six of them being summer terms.

Training Data: All Semesters	Semesters to Predict	Holdout Data
[Spring 2010, Spring 2015] = 16 Semesters	3 = 1 * lag	15.79%
[Spring 2010, Spring 2014] = 13 Semesters	6 = 2 * lag	31.58%
[Spring 2010, Spring 2013] = 10 Semesters	9 = 3 * lag	47.37%

Table 9 - Tests for three semesters lag (Fall, Spring, and Summer data)

Training Data: No Summer Semester	Semesters to Predict	Holdout Data
[Spring 2010, Fall 2014] = 11 Semesters	2 = 1 * lag	15.38%
[Spring 2010, Fall 2013] = 9 Semesters	4 = 2 * lag	30.77%
[Spring 2010, Fall 2012] = 7 Semesters	6 = 3 * lag	46.15%

Table 10 - Tests for two semesters lag (no Summer data)

Since there are 62 courses whose enrollments are being predicted, the focus of the results analysis is instead on the 10 most and 10 least predictable courses that have substantial

enrollment. While courses such as independent study or academic internship are highly predictable, this is mostly due to the lack of enrollment and as such are deemed not as relevant for this project. The predictability appears to be loosely related to the variance of the courses, with courses having higher variances tending to have a higher enrollment and be less predictable, but not always. Unpredictability may also be affected by administrative policies, where additional sections of required core classes will be opened up to accommodate demand. Comparatively, elective course demand is not similarly accommodated, with elective courses offered depending on instructor availability with students expected to enroll in any elective course with empty seats. New, infrequent, or courses whose content changes such as experimental electives, also tend to be less easy to predict simply due to the lack of historical data. Variance was calculated for each undergraduate COMP and CIT course over 19 semesters, with variance defined as the average of the squared differences from the mean. The variance for a particular course can be thought of as the amount that the course enrollment totals for the particular course vary from the average course enrollment for that course. All data in the most predictable and least predictable courses tables use 19 semesters of data for the calculations of variance and average number of students who enroll each semester.

Name	Formula	Description
Variance (σ)	$\frac{\sum(x - \mu)^2}{N}$	x = takes on each value in the sample set μ = the average of the values in the sample set N = the number of values in the sample set

Table 11 - Variability Measure

Course	Variance	Average Students /Semester	Restrictions
CIT 210: ***Deployment and Management of Operating Systems***	316.41	29.02	Prerequisites: CIT 101/L, COMP 122/L. Corequisite: CIT 210 L.
CIT 210 L: ***Deployment and Management of Operating Systems Lab***	316.41	29.02	Lab course for CIT 210.
CIT 360: ***CIT System Management***	184.34	18.57	Prerequisites: CIT 210/L, CIT 270/L.
COMP 100 HON: ***Computers: Their Impact and Use - Honors***	164.11	16.46	Not open to computer science majors. Must be in the honors program.
COMP 222: ***Computer Organization***	1,010.72	93	Prerequisites: COMP 122/L and COMP 182/L.
COMP 256: ***Discrete Structures for Computer Science***	194.24	23.16	Prerequisites: COMP 182/L, MATH 150 A, PHIL 230. Corequisite: COMP 256 L.
COMP 256 L: ***Discrete Structures for Computer Science Lab***	194.24	23.16	Lab course for COMP 256.
COMP 467: ***Multimedia Systems Design***	82.04	8.35	Prerequisites: COMP 380/L.
COMP 490 L: ***Senior Design Project Lab***	553.16	39.71	Lab course for COMP 490 with prerequisite COMP 380/L.
COMP 491 L: ***Senior Project Lab***	544.04	39.05	Lab course for COMP 491 with prerequisite COMP 490/L.

Table 12 - Most Predictable Courses

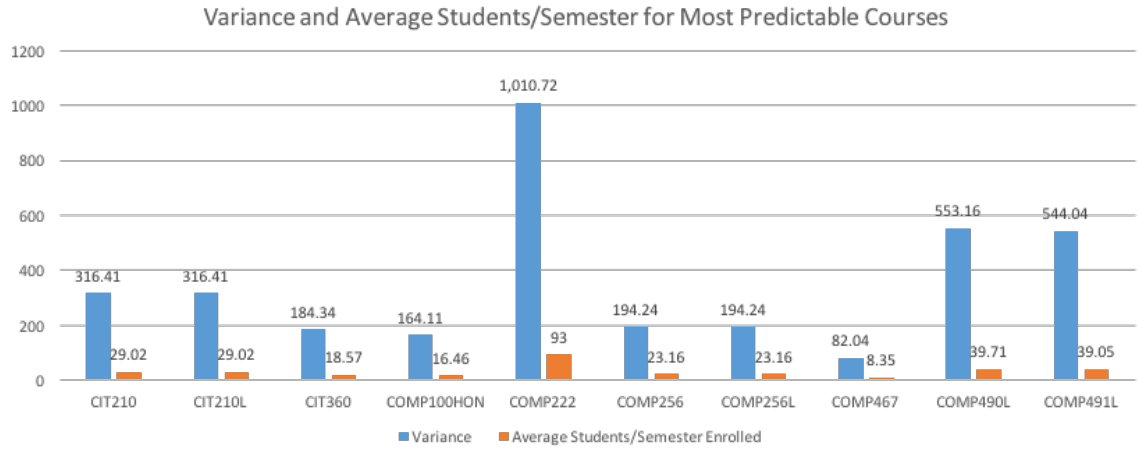

Figure 17 - Variance and Average Students/Semester for Most Predictable Courses

Course	Variance	Average Students /Semester	Restrictions
COMP 100: ***Computers: Their Impact and Use***	113,645.72	6,149.29	Not open to computer science majors.
COMP 122: ***Computer Architecture and Assembly Language***	2,401.58	178.98	Prerequisites: COMP 110/L, MATH 102 or 103 or 104 or 105 or 150A or 255A or passing score on math placement test. Corequisite: COMP 122 L.
COMP 122 L: ***Computer Architecture and Assembly Language Lab***	2,396.0	178.7	Lab course for COMP 122.
COMP 182: ***Data Structures and Program Design***	2,220.09	174.4	Prerequisites: COMP 110/L, MATH 102 or 103 or 104 or 105 or 150A or 255A or passing score on math placement test. Corequisite: COMP 182 L.
COMP 182 L: ***Data Structures and Program Design Lab***	2,205.9	173.65	Lab course for COMP 182.
COMP 310: ***Automata, Languages and Computation***	427.39	49.27	Prerequisites: COMP 256/L.
COMP 380: ***Introduction to Software Engineering***	391.75	56.44	Prerequisites: COMP 270/L or COMP 282, PHIL 230. Corequisite: COMP 380 L.
COMP 380 L: ***Introduction to Software Engineering Lab***	390.19	56.31	Lab course for COMP 380.
COMP 484: ***Web Engineering I***	316.98	39.75	Prerequisites: COMP 322/L or COMP 380/L or CIT 360, IS 451. Corequisite: COMP 484 L.
COMP 484 L: ***Web Engineering I Lab***	316.98	39.75	Prerequisites: COMP 322/L or COMP 380/L or CIT 360, IS 451. Corequisite: COMP 484 L.

Table 13 - Least Predictable Courses

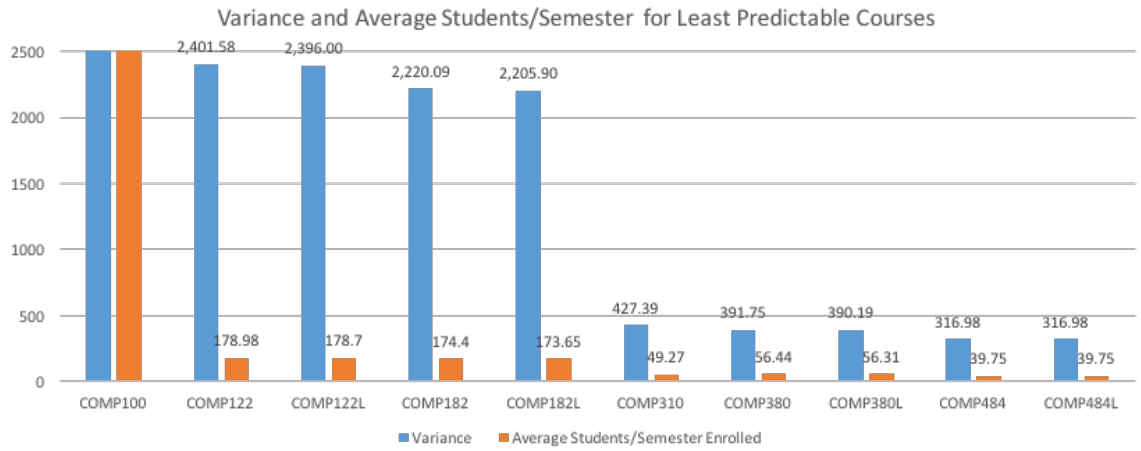

Figure 18 - Variance and Average Students/Semester for Least Predictable Courses

The final configuration of the project code is based off of the results of the testing, which is that the database query is faster when not removing the summer semester, the model predictions are better when not removing the summer semester, and the model predictions are better when using more data to build the model.

In the final phase of deployment, the project code can be run from a java .jar executable file. All of the java files, java dependencies, Weka dependencies, and R forecast dependencies are included. It is expected that the user has R, Java, and a program installed, such as Microsoft Excel, that can read .xlsx formatted files. It is also assumed that the user is assigned a role that has file read, write, and execute permissions. Knowing how much data is required to create a model is a tricky question that depends on the type of statistical model being used and the amount of random variation in the data. Because of this using as much data as possible is typically a good idea, since more data means being able to better identify the structure and patterns used for forecasting.[83] With this in mind, the program will create models based on all of the available historical

[83] (Hyndman and Kostenko, Minimum Sample Size Requirements for Seasonal Forecasting Models 2007)

data and does not hold out any data for testing. This is to ensure that the models learn from the most amount of data possible, in order to give the most accurate predictions. This means that the MAE and RMSE performance measure values are calculated for the models on the data used to train them, so it is possible for the models to suffer from undetected over-fitting. Over-fitting is when a model learns training data, but learns the noise as well as the signal. When the model is then tested on data that has no or different noise, then it does a poor job of predicting, since it is not predicting for the actual signal. An over-fitted model can still predict well on the data it was trained on however, since that data has the same noise the model learned. The program should be run at the end of every semester to help plan for the next semester to ensure it is using the most up-to-date data in order to get the most accurate predictions. The time it takes for the program to calculate and output predictions is directly related to the amount of data in the database. Currently, for 19 semesters worth of data it takes the query just under 7 minutes to run. Since the program predicts for three semesters ahead, it can be run at the end of the summer term every year to predict the upcoming Fall, Spring, and Summer semester enrollments.

CONCLUSION

The models that were the most accurate when comparing modified predictions for Fall, Spring, and Summer semesters using holdout data were: Gaussian Processes, SMOreg, and Linear Regression. This did not align with the models generated using all of the data, where accuracy measurements were calculated using in-sample data and not hold-out data. This is likely due to overfitting, where the models memorize the in-sample data instead of actually predicting it. Since accuracy measurements against hold-out data is more accurate than against in-sample data, future work would be to have the program partition hold-out data and use that to calculate the model accuracy, before generating models using all data for calculating predictions.

Prediction Type	Models Tested Against Hold-Out Data (Partitioned Data)	Models Tested Against Training Data (Non-Partitioned Data)
No Summer Data (Fall & Spring only)	Best Models: GaussianProcesses, SMOreg, LinearRegression	Predicted Best Models: GaussianProcesses, SMOreg, ARIMA
All Semesters (Summer, Fall, Spring)	Best Models: GaussianProcesses, SMOreg, ETS	Predicted Best Models: ARIMA, ETS, GaussianProcesses

Table 14 – Modified Predictions Results

Not predicting on Summer semester data caused Fall and Spring semester enrollment predictions for most, but not all, models to become more accurate. ARIMA and ETS models both became less accurate when constructed without Summer semester data. This could be due to them predicting better when there is more data or data that exhibits more seasonality. The models that were the least accurate when comparing modified predictions for all three semesters using holdout data were: Multilayer Perceptron, RWF,

and ARIMA. Courses that were the hardest to predict with the largest prediction error did not always align with the courses exhibiting the greatest variance. This could be due to administrative policies on when to open additional course sections, and the blanket policy for this research of rounding up the predictions when logically modifying the data. Experienced CSUN planners can eschew the blanket policy, and instead use their experience and intuition to evaluate whether to round up or round down the enrollment predictions for each course. For example, an outsider may think that lower division courses would be more difficult to predict than courses taken further along in the program due to sophomore and senior students being more predictable, when in fact due to administrative policies of accommodating demand for lower division courses and not higher division courses causes the opposite effect. Examination of the most predictable and least predictable courses as forecasted by the best model, Gaussian Processes, shows that despite applying the blanket policy the worst case scenario is not bad, and applying better intuition can ameliorate the results.

Best Predicted Courses	Number of GP Predictions Not Within 25 Students Over 3 Steps	Worst Predicted Courses	Number of GP Predictions Not Within 25 Students Over 3 Steps
CIT210	0	COMP100	2
CIT210L	0	COMP122	2
CIT360	0	COMP122L	2
COMP100HON	0	COMP182	3
COMP222	0	COMP182L	3
COMP256	0	COMP310	1
COMP256L	0	COMP380	1
COMP467	0	COMP380L	1
COMP490L	0	COMP484	2
COMP491L	0	COMP484L	2

Table 15 - Gaussian Processes Best and Worst Predicted Courses

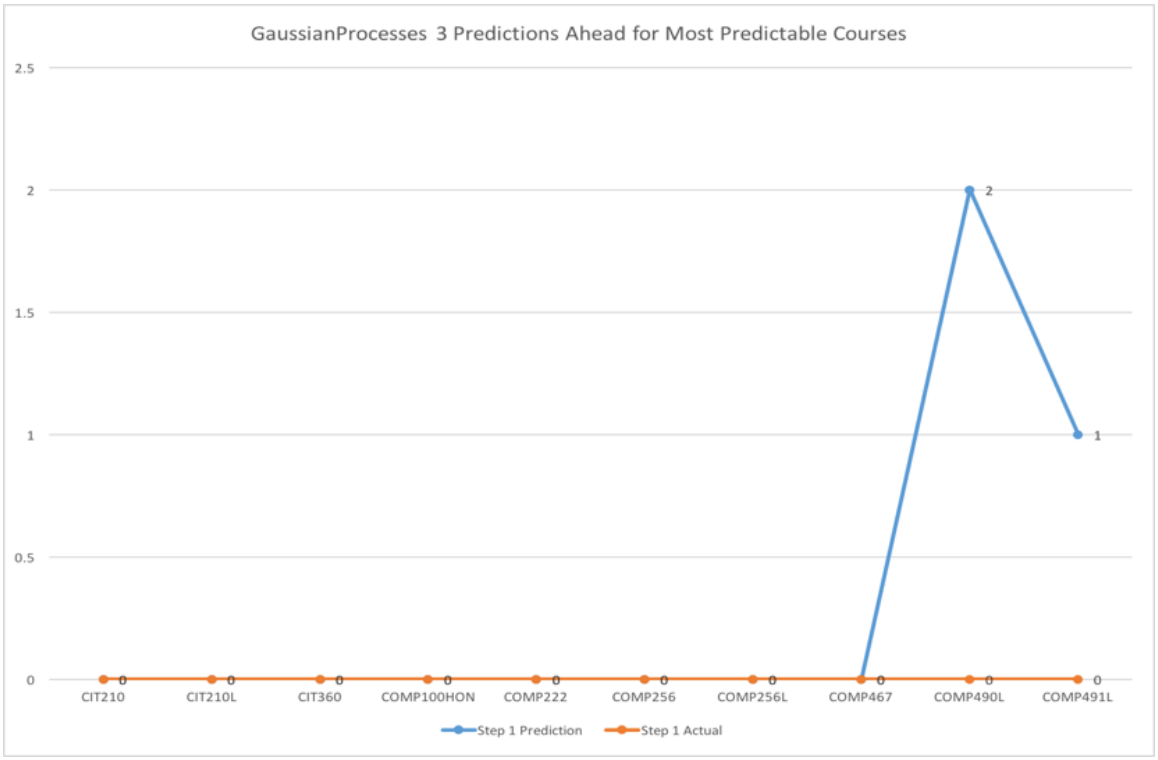

Figure 19 - Gaussian Processes 3 Predictions Ahead for Most Predictable Courses Step 1

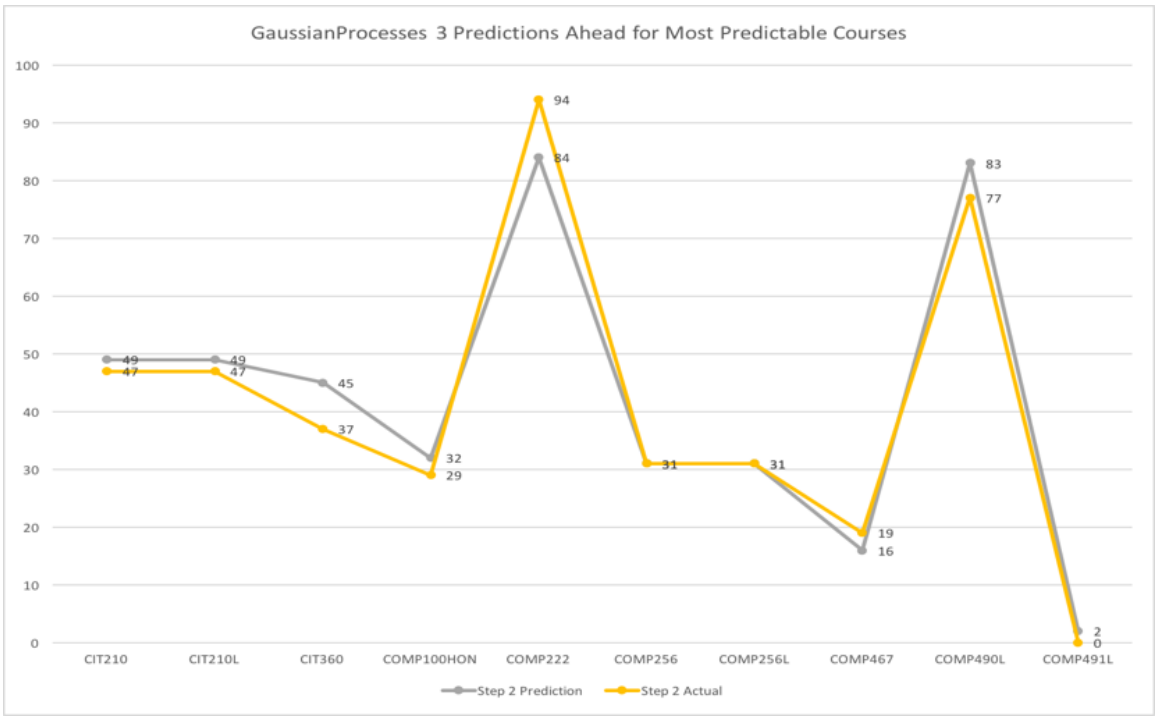

Figure 20 - Gaussian Processes 3 Predictions Ahead for Most Predictable Courses Step 2

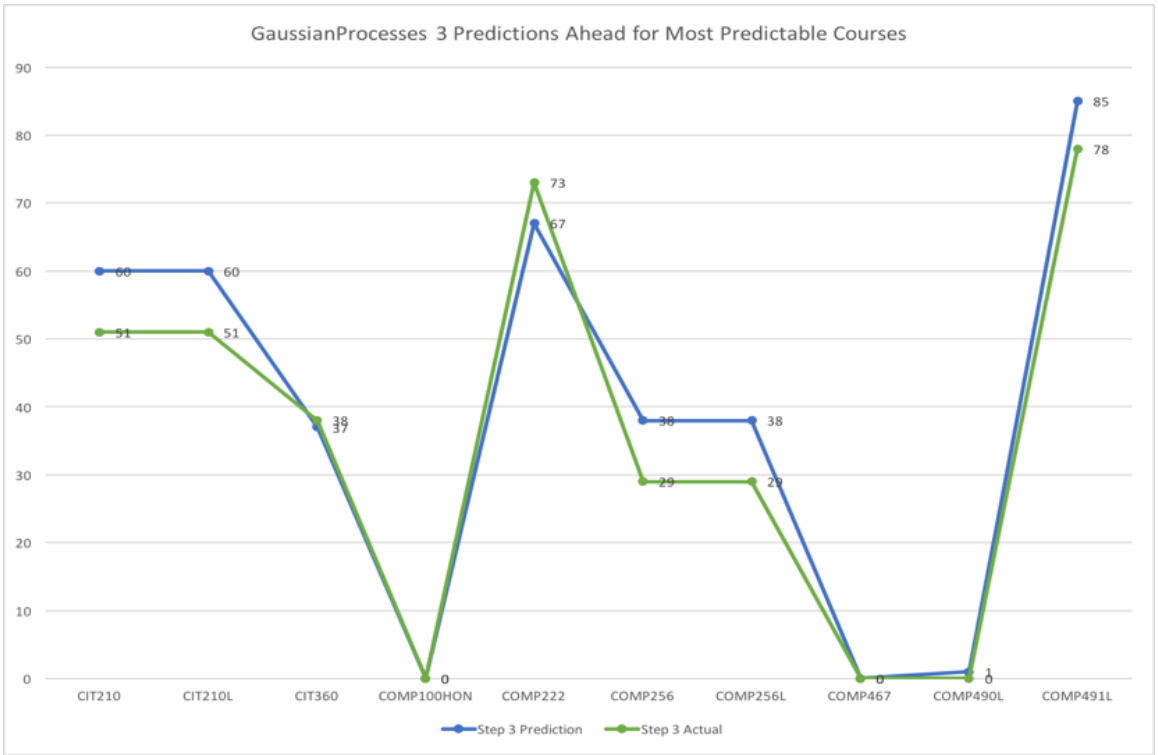

Figure 21 - Gaussian Processes 3 Predictions Ahead for Most Predictable Courses Step 3

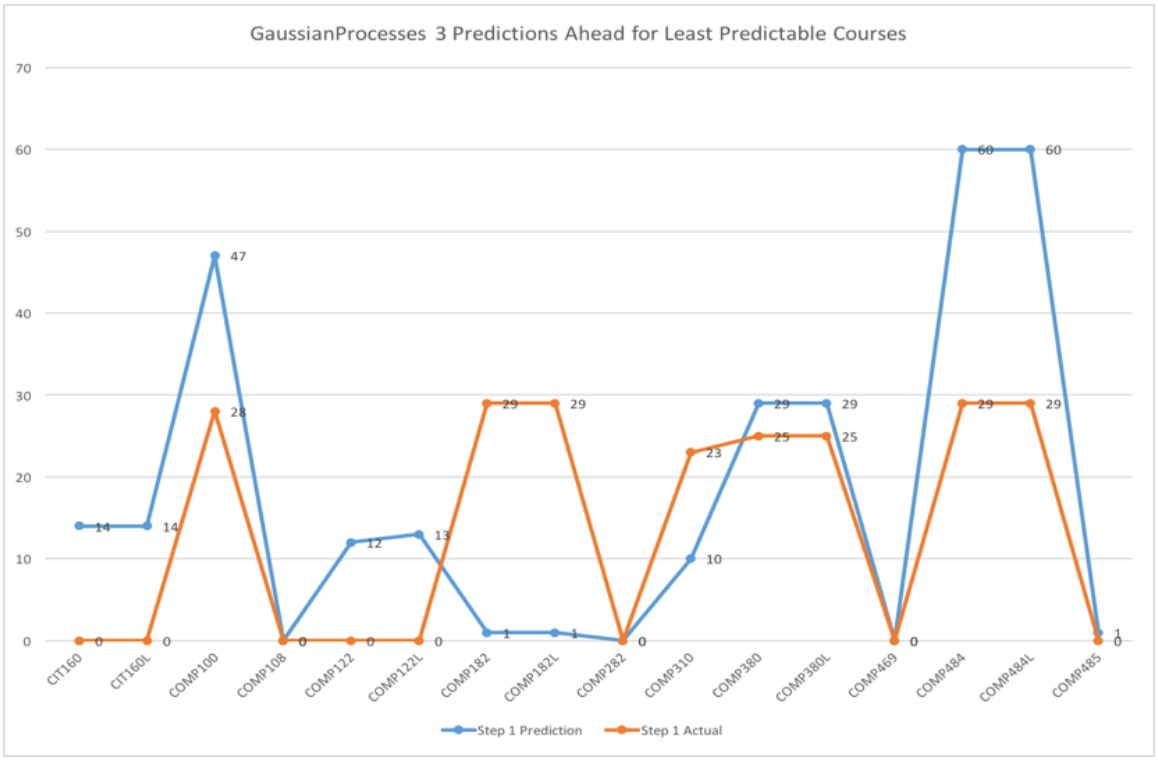

Figure 22 - Gaussian Processes 3 Predictions Ahead for Least Predictable Courses Step 1

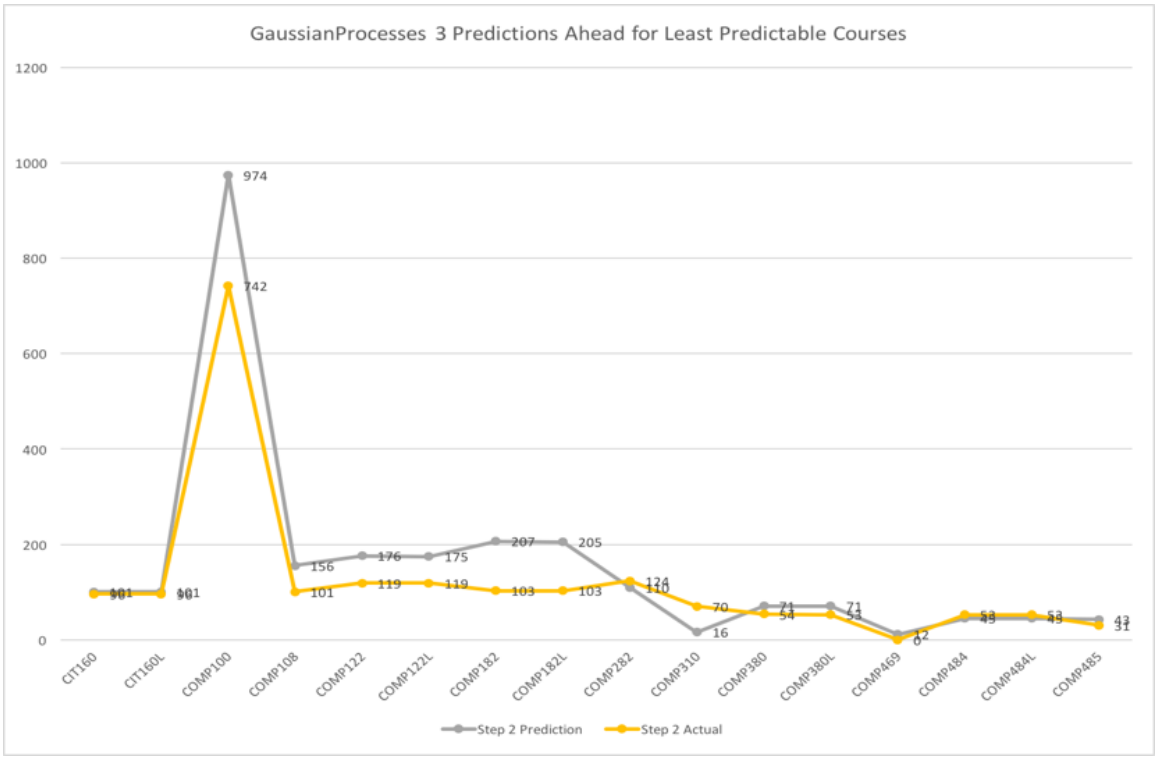

Figure 23 - Gaussian Processes 3 Predictions Ahead for Least Predictable Courses Step 2

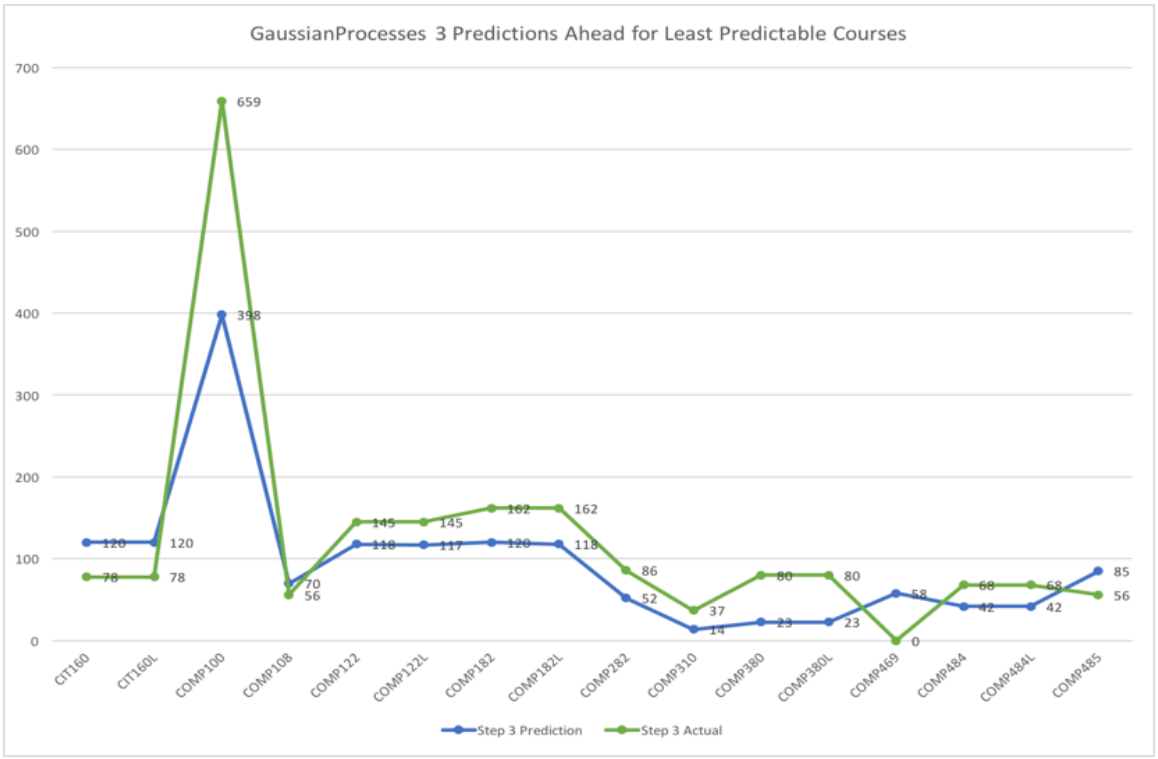

Figure 24 - Gaussian Processes 3 Predictions Ahead for Least Predictable Courses Step 3

Model prediction results will be the most accurate when using as much data as possible to create them, when predicting no further into the future than one season, or three semesters, ahead since the further out a prediction is the less accurate it becomes, and when interpreting the results using intuition instead of a blanket rounding policy. A future enhancement would be to evaluate which models best predict using Summer semester data, and use the results to explore a hybrid model method, with different models being used to predict using Summer, Fall, and Spring semester data.

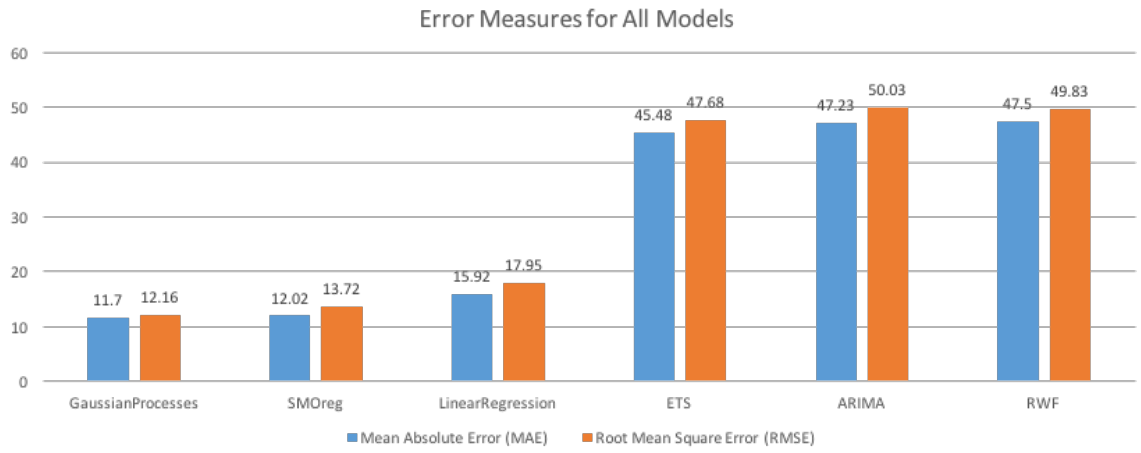

Figure 25 - 2 Predictions Ahead Unmodified Data

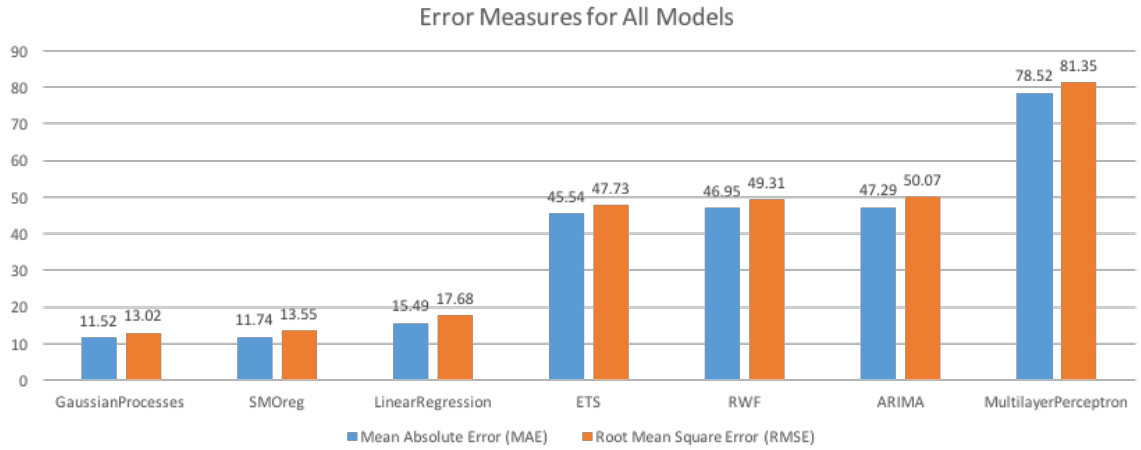

Figure 26 - 2 Predictions Ahead Modified Data

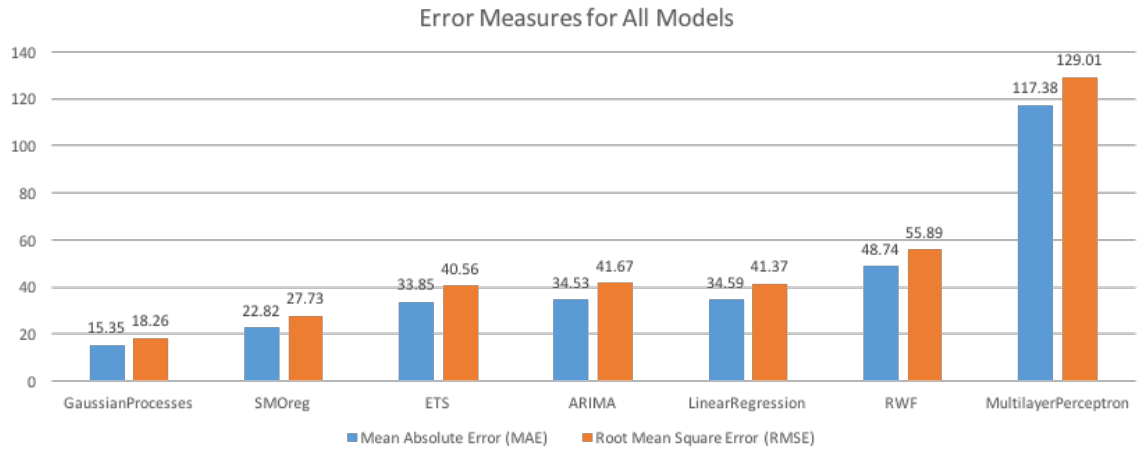

Figure 27 - 3 Predictions Ahead Unmodified Data

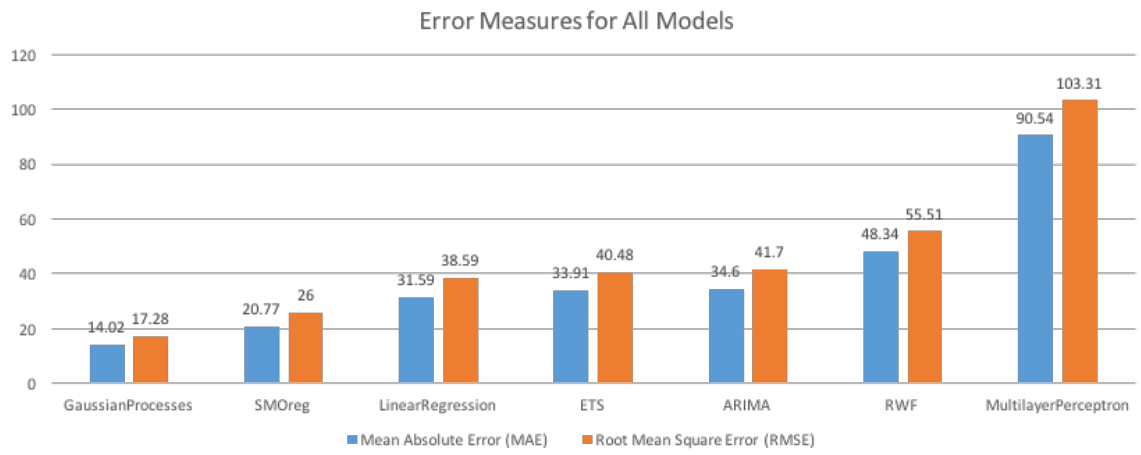

Figure 28 - 3 Predictions Ahead Modified Data

When calculating the number of students that the enrollment predictions are off by, the calculation is predicted enrollment minus actual enrollment. A resulting negative value indicates underestimation while a positive value indicates overestimation. Negative prediction values are treated as zero since there cannot be negative enrollment. Positive fractional values result in rounding up to the next whole number, since there cannot be fractions of students. Since student demand at CSUN is affected by administrative policies, this research decided to be conservative and round up fractional student enrollment prediction values, assuming that enrollment for all courses has not typically been accommodated in the past. Realistically, some course's demand is accommodated while other's is not per policies that a seasoned CSUN planner would be familiar with but that knowledge was not available for application to this project. The modified predictions using the blanket rounding up policy are considered acceptable as long as the absolute difference in the number of students is less than or equal to 25, which is a typical CSUN class size. For a Gaussian Processes model trained on 16 semesters of data beginning with spring 2010, the number of courses out of 62 that were predicted to within 25 students were: 58 for the first step prediction at summer 2015, 55 for the

second step prediction at fall 2015, and 48 for the third step prediction at spring 2016. Summer appears to be the easiest term to predict within 25 students, while spring is the most difficult term to predict within 25 students. This could be due to the courses that are offered in spring being more unpredictable. More likely it is that the accuracy of predictions declines the further out into the future a prediction is, regardless of which semester is being predicted. Over all three terms, the most difficult to predict course enrollments were consistently for COMP 100, COMP 182, COMP 182 L, COMP 122, and COMP 122 L.

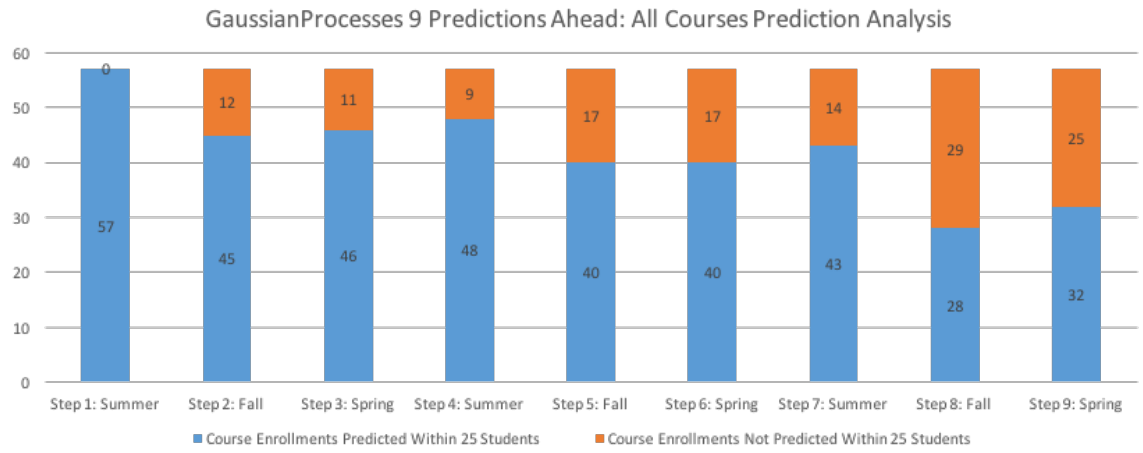

Figure 29 - Results of Training with 10 Semesters of Data

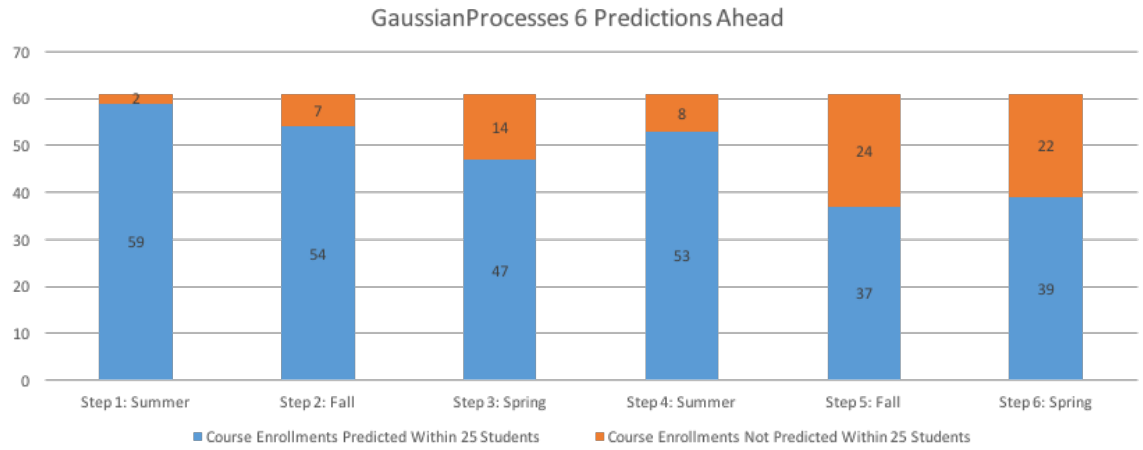

Figure 30 - Results of Training with 13 Semesters of Data

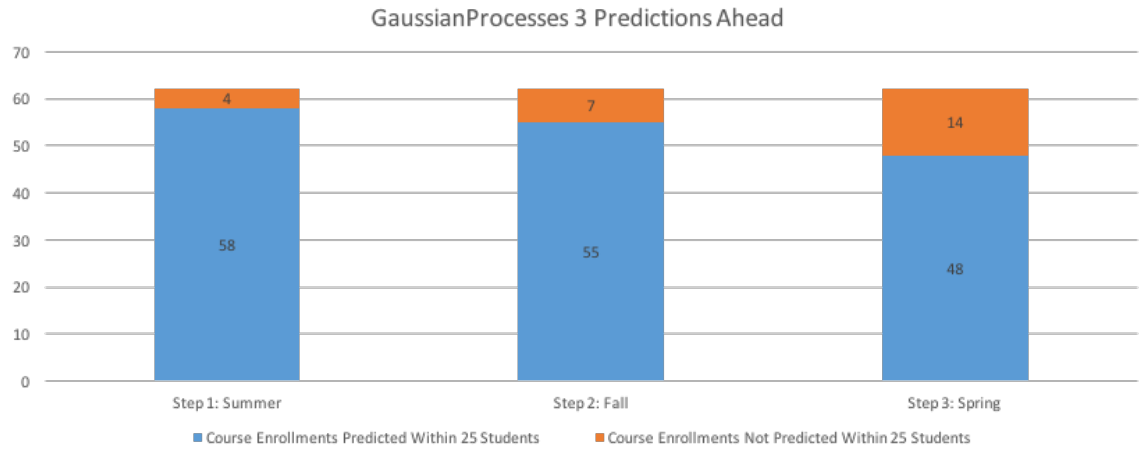

Figure 31 - Results of Training with 16 Semesters of Data

Course	Prediction Difference for Step 1	$\lvert(R_1-P_1)/R_1\rvert$	Prediction Difference for Step 2	$\lvert(R_2-P_2)/R_2\rvert$	Prediction Difference for Step 3	$\lvert(R_3-P_3)/R_3\rvert$
CIT 160	14	-	5	0.05	**42**	0.54
CIT 160 L	14	-	5	0.05	**42**	0.54
COMP 100	19	0.68	**232**	0.31	**-261**	0.4
COMP 108	0	-	**55**	0.54	14	0.25
COMP 122	12	-	**57**	0.48	**-27**	0.19
COMP 122 L	13	-	**56**	0.47	**-28**	0.19
COMP 182	**-28**	0.97	**104**	1	**-42**	0.26
COMP 182 L	**-28**	0.97	**102**	1	**-44**	0.27
COMP 282	0	-	-14	0.11	**-34**	0.4
COMP 310	-13	0.57	**-54**	0.77	-23	0.62
COMP 380	4	0.16	23	0.31	**-57**	0.71
COMP 380 L	4	0.16	23	0.34	**-57**	0.71
COMP 469	0	-	12	-	**58**	-
COMP 484	**31**	1	-8	0.15	**-26**	0.38
COMP 484 L	**31**	1	-8	0.15	**-26**	0.38

COMP 485	1	-	12	0.39	**29**	0.52

Table 16 – Courses with a prediction not within 25 students (R=real value, P=predicted value)

The Gaussian Processes model was overall the best predictor. This may be due to the fact that Gaussian Processes are traditionally used for regression on fixed data sets, and they use Bayesian inference which is good at dynamic analysis of a sequence of data. Additionally, since Gaussian Processes use probability distributions, they may accommodate the uncertainty of transfer and part-time student schedules better than the other models can. The accuracy of model predictions also has much to do with the data used to train the models and its characteristics. Only 19 semesters of data were available, leaving a maximum of 16 semesters for training in order to be able to test predictions for three semesters out. A three semester ahead forecast was used as the minimum future prediction size since the university typically plans one year in advance and because three semesters is considered one "season". In general, all models predict better when there is more data to learn from, but some models can be better at short-term predictions and learning on sparse training data, depending on the variance and seasonality of the data.

For the data set available in the prediction database, it is observed that particular courses tend to only be offered in specific academic terms, making the data more seasonal than linear. Since the historical enrollment data is also accurate, with no missing records, there is also no noise. The trend of the data is that course enrollments tend to go up over time, probably due to increased overall enrollment at the university and in the computer science program, and the models need to be able to detect it. Data issues that can affect the predictions could be experimental courses that are infrequently offered, courses that used to be offered but no longer are under the same course name, or

new courses that do not have adequate historical enrollment data yet. To know which variables affect the predictions, and to what degree, testing with more data, pre-processing the data to remove outliers, and including supplemental overlay data such as academic (tuition rates, high school graduation rates, community college transfer rates, average grades for a course, CSUN overall enrollment) or economic data (unemployment rates, interest rates) could provide further insight. Since only three courses, COMP 100, COMP 182, and COMP 182 L were consistently off in their predictions by two to four class sizes (approximately 50 plus students), the project is still successful. This is since the majority of COMP and CIT courses can be predicted consistently within an acceptable range of error, and the larger errors when predicting COMP 100 and COMP 182/L can likely be mitigated through training the models with more data and applying intuition when logically modifying the prediction results.

FUTURE WORK AND MAINTENANCE

The codebase for the project requires Weka, R, and Java. Optional development environment tools are RStudio – a free IDE for R, and MySQLWorkbench – a free visual tool for database querying. Routine maintenance only consists of updating the software versions. In the case of a deprecated library or package, updating to the new version and changing the project code accordingly.

Software	Current Version	Purpose	How to Update
Java JDK	1.8.0_77	Java Development Kit with runtime environment (JRE) for developing and running java software.	Download the latest version from http://www.oracle.com/technetwork/java/javase/downloads/index.html and follow the instructions. You may need to uninstall a preexisting version or change your environment configuration to point to the new version.
Weka	3.7.13	Data analysis and predictive modeling environment. Access programmatically via Java libraries.	Download the latest version from http://www.cs.waikato.ac.nz/ml/weka/downloading.html and follow the instructions. You may need to uninstall a preexisting version or change your environment configuration to point to the new version. You will need to re-install the timeseriesForecasting package.
Weka timeseriesForecasting Package	1.0.16	Add time series analysis environment to Weka.	Install the latest version following the instructions at https://weka.wikispaces.com/How+do+I+use+the+package+manager%3F#GUI package manager-Installing and removing packages
R	3.2.4	Language and environment for statistical computing. Access programmatically in R console or via rJava to call in Java code.	Download the latest version from https://cran.r-project.org/mirrors.html and follow the instructions. You may need to uninstall a preexisting version or change your environment configuration to point to the new version. You will need to reinstall all R packages such as forecast. Sometimes it also helps to run `sudo R CMD javareconf` to update the configurations after software updates.
R forecast package	7.1	Methods and tools for displaying and analyzing univariate time series.	Follow the instructions at https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Installing-packages or https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Updating-packages
R zoo package	1.7-12	Infrastructure for	Follow the instructions at https://cran.r-

		regular and irregular time series.	project.org/doc/manuals/r-release/R-admin.html#Installing-packages or https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Updating-packages
R timeDate package	3012.100	Chronological and calendar objects.	Follow the instructions at https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Installing-packages or https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Updating-packages
R rJava with JRI package	0.9-9	Low-level R to Java interface and Java to R interface.	Follow the instructions at https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Installing-packages or https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Updating-packages
R openxlsx package	3.1.16	Read, write, and exit .xlsx files.	Follow the instructions at https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Installing-packages or https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Updating-packages
R reshape2 package	1.4.1	Flexibly restructure and reshape aggregate data.	Follow the instructions at https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Installing-packages or https://cran.r-project.org/doc/manuals/r-release/R-admin.html#Updating-packages
mysql-connector-java	5.1.38	JDBC driver for MySQL.	Download the latest version from https://dev.mysql.com/downloads/connector/j/ and follow the instructions.
opencsv	3.7	CSV parser for Java.	Download the latest version from https://sourceforge.net/projects/opencsv/ and follow the instructions.
poi	3.14	Java library for manipulating Microsoft documents, such as Excel.	Download the latest version from https://poi.apache.org/download.html and follow the instructions.

Table 17 – Software

A difficult part in setting up the development environment is the path configuration. The project was developed in a Mac OS X version 10.10.5 environment using a bash profile script to set paths to necessary files. The project was also structured to store R support files under /Library/R/Extensions and store java support files under /Library/Java/Extensions. This meant copying over some R package files, such as the R JRI java files JRI.jar, JRIEngine.jar, REngine.jar, and libjri.jnilib, over to /Library/Java/Extensions to maintain a consistent structure. When developing it is easy to

test code piece-wise in the R console, RStudio IDE, or in the Weka Explorer GUI. This is especially true for debugging R code, since when using rJava it is difficult to debug the java wrapped R calls. Errors can be configured to go to a debug file, but to step through the code and examine variable values the R environment is better suited. Debugging the java code that implements Weka models is easy since the models are written in java and can also be tested in the Weka Explorer GUI.

```
export CLASSPATH=.:/Library/Java/Extensions/poi-3.14/*:/Library/Java/Extensions/poi-3.14/lib/*:/Library/Java/Extensions/poi-3.14/oox
ml-lib/*:/Library/Tomcat/lib/*:/usr/local/mysql/bin/*:/usr/local/mysql/lib/*:/Applications/weka-3-7-13-oracle-jvm.app/Contents/Java/
*:/Applications/weka-3-7-13-oracle-jvm.app/Contents/PlugIns/jdk1.7.0_71.jdk/Contents/Home/jre/lib/*:/Users/amandawatkins/wekafiles/p
ackages/timeSeriesFilters/*:/Users/amandawatkins/wekafiles/packages/timeseriesForecasting/*:/Library/Java/Extensions/*:/Library/Fram
eworks/R.framework/*:/Library/Frameworks/R.framework/Versions/Current/Resources/etc/*:/Library/Frameworks/R.framework/Resources/modu
les/R_X11.so:/Library/Frameworks/R.framework/Resources/modules/R_de.so:/Library/Frameworks/R.framework/Resources/modules/internet.so
:/Library/Frameworks/R.framework/Resources/modules/lapack.so:/Library/Java/Extensions/mysql-connector-java-5.1.38/*:/Library/Java/Ex
tensions/mysql-connector-java-5.1.38/src/lib/*:/Library/R/Extensions/xlsx/java/rexcel-0.5.1.jar:/Library/R/Extensions/xlsxjars/java/
commons-codec-1.6.jar:/Library/R/Extensions/xlsxjars/java/dom4j-1.6.1.jar:/Library/R/Extensions/xlsxjars/java/poi-3.10.1-20140818.ja
r:/Library/R/Extensions/xlsxjars/java/poi-ooxml-3.10.1-20140818.jar:/Library/R/Extensions/xlsxjars/java/poi-ooxml-schemas-3.10.1-201
40818.jar:/Library/R/Extensions/xlsxjars/java/xmlbeans-2.6.0.jar:/Library/Frameworks/R.framework/Resources/lib/*:/Library/R/Extensio
ns/rJava/*:/Library/R/Extensions/rJava/jri/JRI.jar:/Library/R/Extensions/rJava/jri/JRIEngine.jar:/Library/R/Extensions/rJava/jri/REn
gine.jar:/Library/Frameworks/R.framework/Resources/bin/*:/Library/Frameworks/R.framework/Libraries/*:/Library/Java/JavaVirtualMachin
es/jdk1.8.0_77.jdk/Contents/Home/jre/lib/*:/usr/lib/*:/usr/bin/*:/Library/R/Extensions/xlsx/*:/Library/R/Extensions/xlsxjars/*:/Libr
ary/R/Extensions/rJava/*:/Library/R/Extensions/rJava/jri/*:/opt/ImageMagick/lib/*:/Library/Java/Extensions/selenium-2.53.0/*:/Librar
y/Java/Extensions/selenium-2.53.0/libs/*:/Library/Java/Extensions/selenium-2.53-2.0/*:/Library/Java/Extensions/selenium-2.53-2.0/lib
s/*:/Library/Java/Extensions/selenium-safari-driver-2.53.0.jar:$CLASSPATH

export JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_77.jdk/Contents/Home

export LD_LIBRARY_PATH=/Library/Frameworks/R.framework/Resources/lib:/Library/Frameworks/R.framework/Resources/bin:$LD_LIBRARY_PATH

export R_HOME=/Library/Frameworks/R.framework/Resources
export R_LIBS=/Library/R/Extensions:/Library/Frameworks/R.framework/Resources/library
export R_INCLUDE_DIR=/Library/Frameworks/R.framework/Resources/include
export R_DOC_DIR=/Library/Frameworks/R.framework/Resources/doc
export JRI_HOME=/Library/Frameworks/R.framework/Resources/library/rJava/jri:/Library/R/Extensions/rJava/jri:/Library/Java/Extensions

export PATH=/Library/Java/JavaVirtualMachines/jdk1.8.0_77.jdk/Contents/Home/bin:/Library/Frameworks/R.framework/Resources/bin:/usr/b
in/:/Library/R/Extensions/rJava:/Library/R/Extensions/xlsx:.:/opt/ImageMagick/bin:$PATH

export DYLD_FALLBACK_LIBRARY_PATH=/Library/Java/JavaVirtualMachines/jdk1.8.0_77.jdk/Contents/Home/jre/lib:/Library/Frameworks/R.fram
ework/Resources/lib:$DYLD_FALLBACK_LIBRARY_PATH
```

Figure 32 - .bash_profile configuration

Enhancements to make in the future would be to add more historical data to the prediction database. This is because models perform better when they train on more data. It would also be helpful to code in a data split for training and testing data, output those results to the excel spreadsheet, and then perform what the code currently does by creating a model using all available data and outputting the MAE and RMSE for that model. By having the extra training results, it would indicate more accurately which model will perform better, since with testing on holdout data there is no overfitting to the holdout data. It would also be nice for the program to have a graphical user interface to

enable more customizable configurations, such as being able to change variables such as: which dates to use to partition the dataset, how many semesters ahead to predict, which courses to not predict (to eliminate possible outliers or courses that are no longer offered), whether to use actual values or rounded values - where negative values are zeroed out and any fractional values are rounded up – to apply human intuition, or fields to use as overlay data. Finding sources of overlay data would also be a useful enhancement, since economic data as well as local population data may help with enrollment predictions. The auto-generation of graphs would also be a good enhancement, to enable visualization of results. There are many existing technologies for R or Weka web applications, but not both. Shiny is a platform as a service (PaaS) for hosting R web applications, OpenCPU is an open source solution for embedded R computing, web applications for Weka can be developed using any web framework that is Java compatible, such as using Apache Tomcat and Servlets. It is likely possible to create a web application that handles both R and Weka, due to the popularity of both languages. If the required R packages are ported to Java as part of Renjin, then it will be easy to develop a webapp since everything will be Java-based. Advantages of the project using only Java is that it is easier to pre-process the data in Java than in R, and users wouldn't need to install R. Conversely, the project could migrate entirely to R, which has the advantage of running faster than Java code and being designed specifically for data analysis. There does exist an RWeka package for using Weka in R, but unfortunately it does not include the timeseriesForecasting package functionality yet, which is why this approach was not initially taken. Due to the advantages R offers, it may be worth the learning curve to migrate the project to only use R, assuming there is a model that

predicts as well or better than Weka's Gaussian Processes. As of this time, no R package could be found that has a GP model for time series prediction, which is why Weka was used in conjunction with R so that a greater number of models could be tested.

BIBLIOGRAPHY

Aadland, Dave, Rob Godby, and Jeremiah Weichman. 2007. *University of Wyonming Enrollment Project Final Report.* Department Project, Economics and Finance, Laramie, WY: University of Wyoming, Department of Economics and Finance, 1-29.

Ahlburg, Dennis, Michael McPherson, and Morton Owen Schapiro. 1994. *Predicting Higher Education Enrollment in the United States: An Evaluation of Different Modeling Approaches.* Discussion Paper, Williams College, Williamstown: Williams Project on the Economics of Higher Education.

Balachandran, K. R., and Donald Gerwin. 1971. "Variable-Work Models for Predicting Course Enrollments." *Operations Research* (INFORMS) 21 (3): 823-834.

Bocklisch, Steffen, and Michael Päßler. 2000. *Fuzzy Time Series Analysis.* Vol. 6, in *Fuzzy Control Theory and Practice*, by Steffen Bocklisch and Michael Päßler, 331-345. Chemnitz: Physica-Verlag HD.

Bouckaert, Remco R., Eibe Frank, Mark Hall, Richard Kirkby, Peter Reutemann, Alex Seewald, and David Scuse. 2016. *WEKA Manual for Version 3-9-0.* Hamilton, Waikato, April 14. http://weka.8497.n7.nabble.com/Unit-tests-td16122.html.

California State University. 2016. *The California State University Analytic Studies Statistical Reports.* January 20. Accessed March 2, 2016. http://www.calstate.edu/as/stats.shtml.

California State University, Northridge. 2013. *CSUN Outreach Publications View Book.* March 1. Accessed March 2, 2016. http://www.csun.edu/sites/default/files/viewbook.pdf.

Chen, Dr. Chau-Kuang. 2008. "An Integrated Enrollment Forecast Model." *IR Applications* (Association for Institutional Research) 15.

Dalrymple, Douglas J., and Barry E. King. 1981. "Selecting Parameters for Short-Term Forecasting Techniques." *Decision Sciences* 12 (4): 661-669.

DeLeeuw, Jamie. 2012. *Unemployment Rate and Tuition as Enrollment Predictors.* Report, Monroe County Community College, Monroe: Monroe County Community College, 1-13.

Faraway, Julian, and Chris Chatfield. 1998. "Time Series Forecasting with Neural Networks: A Comparative Study Using the Airline Data." *Applied Statistics* (47): 231-250.

Felts, Kathy Schmidtke, and Mark Ehlert. 2009. *Prediction Model for Course Demand at MU.* Presentation, Enrollment Management and Institutional Research, University of Missouri- Columbia, Columbia: University of Missouri- Columbia, 1-14.

Feuerriegel, Stefan. 2016. "Algorithm Design and Software Engineering (öffentlicher Zugriff)." *Information Systems Researc Albert-Ludwigs-Universität Freiburg.* February 10. Accessed June 8, 2016. https://www.is.uni-freiburg.de/ressourcen/algorithm-design-and-software-engineering-oeffentlicher-zugriff/11_softwaretesting.pdf.

Frigola-Alcalde, Roger. 2015. *Bayesian Time Series Learning with Gaussian Processes.* PhD Thesis, Engineering, University of Cambridge, University of Cambridge, 1-109.

Ghahramani, Zoubin. 2011. *A Tutorial on Gaussian Processes (or why I don't use SVMs).* Lecture 1 Slides, Engineering and Machine Learning, University of Cambridge, Cambridge: Zoubin Ghahramani, 1-31.

Graczyk, Magdalena, Tadeusz Lasota, and Bogdan Trawiński. 2009. "Comparative Analysis of Premises Valuation Models Using KEEL, RapidMiner, and WEKA." *First International Conference ICCCI.* Wrocław: Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems. 800-812.

Gvaladze, Sopiko. 2015. *Evaluating Methods for Time-Series Forecasting Applied to Energy Consumption Predictions for Hvaler (kommune).* Master's Thesis, Computer Science, Østfold University College, Halden: Østfold University College, 1-80.

Hill, Tim, Marcus O'Connor, and William Remus. 1996. "Neural Network Models for Time Series Forecasts." *Management Science* (INFORMS) 42 (7): 1082-1092.

Hopkins, David, and William Massy. 1981. *Planning Models for Colleges and Universities.* Stanford: Stanford University Press.

Hyndman, Rob J., and Andrey V. Kostenko. 2007. "Minimum Sample Size Requirements for Seasonal Forecasting Models." *Foresight: the International Journal of Applied Forecasting* (6): 12-15.

Hyndman, Rob J., Anne B. Koehler, Ralph D. Snyder, and Simone Grose. 2002. "A state space framework for automatic forecasting using exponential smoothing methods." *International Journal of Forecasting* (Elsevier) 18: 439-454.

Hyndman, Rob. 2016. *Package 'forecast'.* R Package Documentation, github: CRAN Repository.

Institutional Research and Analysis Office for the University of Hawai'i System. 2013. *Enrollment Projections for the University of Hawai'i System Fall 2013 to Fall 2018.* Department Report, Honolulu, HI: University of Hawai'i, 1-32.

Johnstone, James N. 1974. "Mathematical Models Developed for Use in Educational Planning: A Review." *Review of Educational Research* (American Educational Research Association) 44 (2): 177-201.

Kardan, Ahmad, Hamid Sadeghi, Saeed Shiry Ghidary, and Mohammad Reza Fani Sani. 2013. "Prediction of Student Course Selection in Online Higher Education Institutes Using Neural Network." *Computers & Education* (Elsevier Ltd.) 65: 1-11.

KDnuggets. 2014. *CRISP-DM, still the top methodology for analytics, data mining, or data science projects.* 10 28. Accessed 6 17, 2016. http://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html.

Kraft, Christine. 2007. *Planning, Scheduling, and Timetabling in a University Setting.* PhD Dissertation, Mathematical Sciences, Clemson University, Clemson: Clemson University TigerPrints, 41-58.

Kraft, Christine, and James Jarvis. 2005. *An Adaptive Model for Predicting Course Enrollment.* Thesis, Mathematical Sciences, Clemson University, Clemson: Clemson University, 1-10.

Napagoda, Chandana. 2013. "Web Site Visit Forecasting Using Data Mining Techniques." *International Journal of Scientific & Technology Research* (IJSTR) 2 (12): 170-174.

National Center for Education Statistics. 2016. *College Navigator.* March 2. Accessed March 2, 2016. http://nces.ed.gov/collegenavigator/.

Nemeş, Magdalena Daniela, and Alexandru Butoi. 2013. "Data Mining on Romanian Stock Market Using Neural Networks for Price Prediction." *Informatica Economică* 17 (3): 125-136.

Ognjanovic, Ivana, Dragan Gasevic, and Shane Dawson. 2016. "Using Institutional Data to Predict Student Course Selections in Higher Education." *Internet and Higher Education* (Elsevier) 29: 49-62.

Pettibone, Timothy J., and Latha Bushan. 1990. "School District Enrollment Projections: A Comparison of Three Methods." *Mid-South Educational Researcher.* New Orleans: Mid-South Educational Research Association.

Póczos, Barnabás. 2009. *Introduction to Gaussian Processes.* Lecture Notes, Machine Learning, University of Alberta, Alberta: University of Alberta.

Ramsey, Pat, Andre Watts, and Lisa Sklar. 2015. "Institutional Knowledge Management Enrollment Projection Model." *Southern Association for Institutional Research.* Savannah, GA: Southern Association for Institutional Research. 1-27.

Reinstadler, Martin, Matthias Braunhofer, Medhi Elahi, and Francesco Ricci. 2013. *Predicting Parking Lots Occupancy in Bolzano.* Academic Project, Computer Science, Free University of Bolzano Italy, Bolzano: SistemiProattividi Accessoall'Informazione, 1-22.

Reiss, Elayne. 2012. "Best Practices in Enrollment Modeling: Navigating Methodology and Processes." *Southern Association for Institutional Research.* Lake Buena Vista, FL: Southern Association for Institutional Research. 1-34.

Rexer, Karl. 2015. *2015 Data Science Survey.* Annual Survey, Boston: Rexer Analytics.

Rickes Associates Inc. 2015. "California State University Northridge: Teaching, Learning, Office, and Research Space Needs Assessment."

Roberts, S., M. Osborne, M. Ebden, S. Reece, N. Gibson, and S. Aigrain. 2012. *Gaussian Processes for Timeseries Modelling.* Academic Paper, Engineering Science and Astrophysics, University of Oxford, Oxford: The Royal Society Publishing, 1-27.

Shevade, S. K., S. S. Keerthi, C. Bhattacharyya, and K. R. K. Murthy. 2000. "Improvements to the SMO Algorithm for SVM Regression." *IEEE Transactions on Neural Networks* 11 (5): 1188-1193.

Smola, Alex J., and Bernhard Schölkopf. 2004. "A Tutorial on Support Vector Regression." *Statistics and Computing* (Kluwer Academic Publishers) 14: 199-222.

U.S. News. 2016. *U.S. News Colleges California State University Northridge 2016 Overview.* June 12. http://colleges.usnews.rankingsandreviews.com/best-colleges/csun-1153.

University of California. 2016. *The University of California at a Glance.* March 1. Accessed July 3, 2016. http://universityofcalifornia.edu/sites/default/files/uc-at-a-glance-mar-2016.pdf.

University of California, Los Angeles. 2016. *UCLA Academic Planning and Budget: Campus Statistics for Enrollment.* March 2. Accessed March 2, 2016. http://www.aim.ucla.edu/enrollment2.aspx.

University of Colorado, Boulder. 2015. *CU Boulder: Planning, Budget and Analysis - Enrollment Projections.* July 22. Accessed March 2, 2016.

http://www.colorado.edu/pba/enrlproj/.
University of Virginia Demographics Research Group. 2014. *School Enrollment Projections for Virginia Public Schools.* Department Report, Charlottesville, VA: University of Virginia, 1-5.
Walczak, Steven. 1998. "Neural Network Models for a Resource Allocation Problem." *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)* (IEEE Systems, Man, and Cybernetics Society) 28 (2): 276-284.
Walonick, David. 1993. *Survival Statistics.* St. Paul, Minnesota: StatPac.
Wirth, Rüdiger, and Jochen Hipp. 2000. "CRISP-DM: Towards a Standard Process Model for Data Mining." *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining* 29-39.

APPENDIX A: PROJECT CODE

```
import java.io.*;

import java.util.*;
import java.util.Set;
import java.util.Iterator;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

import java.text.DecimalFormat;

import java.sql.Connection;
import java.sql.Driver;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;

import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.attribute.PosixFileAttributes;
import java.nio.file.attribute.PosixFilePermission;
import java.nio.file.attribute.PosixFilePermissions;
import static java.nio.file.attribute.PosixFilePermission.*;

import com.opencsv.CSVParser;
import com.opencsv.CSVReader;
import com.opencsv.CSVWriter;

import weka.core.Instance;
import weka.core.Instances;
import weka.core.Attribute;
import weka.core.converters.CSVLoader;
import weka.core.converters.CSVSaver;
import weka.core.converters.ConverterUtils.DataSource;

import weka.classifiers.Classifier;

import weka.classifiers.functions.GaussianProcesses;
import weka.classifiers.functions.LinearRegression;
import weka.classifiers.functions.MultilayerPerceptron;
import weka.classifiers.functions.SMOreg;

import weka.classifiers.evaluation.NumericPrediction;

import weka.classifiers.timeseries.AbstractForecaster;
import weka.classifiers.timeseries.core.TSLagMaker;
import weka.classifiers.timeseries.eval.TSEvaluation;
import weka.classifiers.timeseries.WekaForecaster;

import org.rosuda.JRI.REXP;
import org.rosuda.JRI.Rengine;
```

```
import org.apache.poi.xssf.usermodel.XSSFRow;
import org.apache.poi.xssf.usermodel.XSSFCell;
import org.apache.poi.xssf.usermodel.XSSFFont;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;
import org.apache.poi.xssf.usermodel.XSSFCellStyle;

import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
import org.apache.poi.ss.util.CellUtil;
import org.apache.poi.ss.util.RegionUtil;
import org.apache.poi.ss.util.CellRangeAddress;

public class PredictEnrollment {

  public static void main(String[] args) {

      /************
       * System:  MySQL
          * HOST:    mysql-prod.ptg.csun.edu
          * Port: 50075
          * Accounts:
          *    Admin: 'a_prediction', ‘ZYnT3a4C'
          *    User:  'prediction', ‘ToViYiBG'
          *    ReadOnly: 'r_prediction', 'ToViYiBG'
       ***********/

    try {
      Connection mysqlConnection = DriverManager.getConnection("jdbc:mysql://mysql-
prod.ptg.csun.edu:50075/prediction","r_prediction","ToViYiBG");
      Statement stmt = mysqlConnection.createStatement();
      stmt.executeQuery("USE prediction;");
      stmt.executeQuery("SET group_concat_max_len=990000;");
      stmt.executeQuery("SET @sql = NULL;");
      /* Get enrollment data for all undergraduate computer science courses for all academic terms and years
that have begin dates before today's date */
      /* Do Not add any whitespace to the string arguments to executeQuery - it will result in errors */
      stmt.executeQuery("SELECT GROUP_CONCAT(DISTINCT CONCAT('COUNT(DISTINCT(CASE
WHEN sct.SUBJECT='", sct.SUBJECT, "' AND sct.CATALOG_NBR='", sct.CATALOG_NBR, "' THEN
sse.EMPLID END)) AS `', sct.SUBJECT, sct.CATALOG_NBR, '_Enrollment_Total`')) INTO @sql
FROM SA_CLASS_TBL sct WHERE sct.DEPT_NAME='Computer Science' AND sct.CLASS_STS='A'
AND CAST(LEFT(sct.CATALOG_NBR, 1)AS UNSIGNED)<5");
      stmt.executeQuery("SET @sql = CONCAT('SELECT stt.BEGIN_DATE AS StartDate, ', @sql, '
FROM SA_CLASS_TBL sct LEFT JOIN SA_STDNT_ENRLS sse ON sct.STRM=sse.STRM AND
sct.CLASS_NBR=sse.CLASS_NBR LEFT JOIN SA_TERM_TBL stt ON stt.STRM=sct.STRM WHERE
stt.BEGIN_DATE<NOW() AND sct.DEPT_NAME="Computer Science" AND sct.CLASS_STS="A"
AND CAST(LEFT(sct.CATALOG_NBR, 1) AS UNSIGNED)<5 GROUP BY stt.BEGIN_DATE,
stt.TERM')");
      stmt.executeQuery("PREPARE stmt FROM @sql;");
      ResultSet queryResult = stmt.executeQuery("EXECUTE stmt;");

      /* Write the mySQL query results including the header to a csv file */
      /* The csv file will be the data input to Weka and R for enrollment forecasting */
      CSVWriter writer = new CSVWriter(new FileWriter(new
```

```
File("/tmp/CoursePredictionSQLQueryOutput.csv")));
    writer.writeAll(queryResult,true);
    writer.flush();
    writer.close();

    stmt.executeQuery("DEALLOCATE PREPARE stmt;");
    stmt.close();

    String pathToData = "/tmp/CoursePredictionSQLQueryOutput.csv";
    String pathToCleanedData = "/tmp/CoursePredictionCleaned.csv";
    String pathToOutFile = "CourseEnrollmentPredictions.xlsx";

    /* Get all of the courses to predict enrollment for from the header of the
CoursePredictionSQLQueryOutput.csv starting at index 1 (indices start at 0) */
    CSVReader reader = new CSVReader(new FileReader(pathToData));
    String[] rowHeader = reader.readNext();
    reader.close();

    String allcourses[] = new String[rowHeader.length-1];
    for(int i=1; i<rowHeader.length; i++){
      allcourses[i-1] = rowHeader[i];
    }

    /* Load the csv enrollment data without any overlay fields */
    /* First column is the date, all subsequent columns are numeric total enrollment, memory buffer size is
increased */
    CSVLoader loader = new CSVLoader();
    loader.setSource(new File(pathToData));
    loader.setDateAttributes("1");

    /* JDBC changes the timestamp format to this format */
    loader.setDateFormat("dd-MMM-yyyy HH:mm:ss");
    loader.setBufferSize(100000);
    Instances enrollment = loader.getDataSet();

    /* Do not predict when a course has only enrollment of 0 by removing columns that only have 0 for
their value */
    /* SMOreg will not predict enrollment for a course when a course has the same enrollment number for
all of the data */
    /* Since it is unlikely for 5+ years of data for a course to have the same non-zero enrollment we only
look for the zero value */
    /* Since 0 would indicate the course is rarely offered and so is a likely occurrence */
    /* Note that the number of instances (aka rows in the csv) is correct because the header isn't counted but
the */
    /* number of attributes (aka columns in the csv) is 1 greater because it is counting the */
    /* first column which is the date which we want to ignore */
    int[] nonZeroEnrollment = new int[allcourses.length];
    for(int i=0; i<enrollment.numInstances(); i++){
      Instance semester = enrollment.instance(i);
      for(int j=1; j<semester.numAttributes(); j++){
        Attribute course = semester.attribute(j);
        if(semester.value(course)>0){
          nonZeroEnrollment[j-1] = 1;
        }
      }
    }
```

```
    /* Needed since when an attribute is removed, all subsequent attribute's indices decrease by 1 */
    int decrement = 0;
    for(int n=0; n<nonZeroEnrollment.length; n++){
      if(nonZeroEnrollment[n] == 0){
        /* Remove the attribute for each instance (ie remove the course values for each semester) */
        /* Index is +1 because the first attribute is the date */
        enrollment.deleteAttributeAt(n-decrement+1);
        decrement++;
      }
    }

    /* Write the cleaned data to a new csv file */
    /* Use Weka's util since the Instances obj is in .arff format and we want csv */
    CSVSaver saver = new CSVSaver();
    saver.setInstances(enrollment);
    saver.setFile(new File(pathToCleanedData));
    saver.writeBatch();

    /* Update the header and courses array to reflect the clean data */
    reader = new CSVReader(new FileReader(pathToCleanedData));
    rowHeader = reader.readNext();
    reader.close();

    String courses[] = new String[rowHeader.length-1];
    for(int i=1; i<rowHeader.length; i++){
      courses[i-1] = rowHeader[i];
    }

    /* Predict the enrollment for next two upcoming semesters where semesters are Fall, Spring, Summer */
    /* since no undergraduate computer science courses are offered over the Winter semester */
    int numSemestersToForecast = 3;

    /* Do not hold out any data when making the classifier for testing */
    /* The more data used to build the classifier = the more accurate the classifier will be */
    float percentHoldOut = 0.0f;

    /* Round predictions to 2 decimal places */
    DecimalFormat df = new DecimalFormat("#0.##");

    /* The row to start inserting prediction data at */
    /* Begin at 1 since the header is the first row and indices start at 0 */
    int rowP = 1;

    /* The row to start inserting accuracy data at */
    /* Begin at 2 since headers are the first two rows and indices start at 0 */
    int rowA = 2;

    /* The number of classifiers to process using R = Arima, ETS, and RWF models */
    int rModelCnt = 3;

    /* Classifiers to feed to Weka */
    Classifier[] classifiers = {new GaussianProcesses(), new LinearRegression(), new
MultilayerPerceptron(), new SMOreg()};

    String[] headerA = {"Courses", "GaussianProcesses", "LinearRegression", "MultilayerPerceptron",
```

```
"SMOreg", "Arima", "ETS", "RWF"};

    String[] headerP = new String[(headerA.length-1)*numSemestersToForecast+1];
    headerP[0] = "Courses";
    int step = 0;
    for(int i=1; i<headerA.length; i++){
      for(int j=0; j<numSemestersToForecast; j++){
        headerP[j+1+step] = headerA[i] + "-" + (j+1) + "_Sem_Ahead";
      }
      step += numSemestersToForecast;
    }


    /* There must be a minimum of two accuracy methods due to cell merges in formatting */
    /* Due to having to parse formatted output to get the accuracy values more code than the strings here
would need to be changed */
    String[] accuracy = {"MAE", "RMSE"};

    /* Create Excel Workbook (.xlsx) to write results to different spreadsheets */
    XSSFWorkbook workbook = new XSSFWorkbook();
    XSSFSheet sheetP = workbook.createSheet("CourseEnrollmentPredictions");
    XSSFSheet sheetA = workbook.createSheet("ForecastAccuracy");

    File outfile = new File(pathToOutFile);

    FileOutputStream fileout = new FileOutputStream(outfile);

    /* Set permissions on the xlsx workbook */
    outfile.setReadable(true, false);
    outfile.setWritable(true, false);
    outfile.setExecutable(true, false);

    /* Set permissions on the xlsx workbook using POSIX (won't work on Windows) */
    Set<PosixFilePermission> perms = new HashSet<>();
    perms.add(OWNER_READ);
    perms.add(OWNER_WRITE);
    perms.add(OWNER_EXECUTE);
    perms.add(GROUP_READ);
    perms.add(GROUP_WRITE);
    perms.add(GROUP_EXECUTE);
    perms.add(OTHERS_READ);
    perms.add(OTHERS_WRITE);
    perms.add(OTHERS_EXECUTE);
    Files.setPosixFilePermissions(Paths.get(pathToOutFile), perms);

    workbook.write(fileout);
    fileout.flush();
    fileout.close();

    /* BEGIN R FORECAST CODE */
    Rengine re = new Rengine (new String [] {"--vanilla"}, false, null);

     if (!re.waitForR())
     {
        System.out.println ("Unable to load R");
        return;
```

```
    }
    else {
        System.out.println ("Connected to R\n");

        /* Generate a log file for any R errors that are thrown */
        //re.eval("log<-file('R_Logfile.txt');");
        //re.eval("sink(log, append=TRUE);");

        re.eval("library(zoo);");
        re.eval("library(timeDate);");
        re.eval("library(rJava);");
        re.eval("library(openxlsx);");
        re.eval("library(forecast);");
        re.eval("library(reshape2);");

        re.eval("data<-read.csv(file='/tmp/CoursePredictionCleaned.csv');");

        /* Convert the data into a time series and predict on the non-date, course enrollment columns */
        /* The date is the first column in the data so remove it to predict on the subsequent columns */
        /* The frequency corresponds to the seasonality of the data, where 3 represents the 3 semesters
courses are offered = fall, spring, summer */
        re.eval("datats<-ts(data[,-1],frequency=3);");

        /* Number of columns representing courses to predict enrollment for */
        re.eval("ncc<-ncol(datats);");

        /* Number of columns for accuracy */
        /* R by default outputs 7 (ME, RMSE, MAE, MPE, MAPE, MASE, ACF1) */
        re.eval("nca<-7;");

        /* The accuracy measurements to write to the Excel spreadsheet */
        re.eval("accmeasures<-c('MAE','RMSE');");
        re.eval("laccm<-length(accmeasures);");

        /* Number of columns for forecast items output by default by R */
        /* PointForecast, Lo80, Hi80, Lo95, Hi95 (PointForecast = $mean) */
        re.eval("ncf<-5;");

        /* Number of predictions to make after the end of the existing data */
        /* Each prediction is for a semester, where semesters are Fall, Spring, and Summer */
        /* Due to Excel formatting changing this will require changing Excel formatting code */
        re.eval("h<-3;");

        /* Number of models (aka Classifiers) used = auto.arima, ets, rwf */
        re.eval("nmu<-3;");

        /* Store the PointForecasts ($mean) in matrices */
        re.eval("fcast_arima<-matrix(NA,nrow=h,ncol=ncc);");
        re.eval("fcast_ets<-matrix(NA,nrow=h,ncol=ncc);");
        re.eval("fcast_rwf<-matrix(NA,nrow=h,ncol=ncc);");

        /* accuracy() returns in-sample ME, RMSE, MAE, MPE, MAPE, MASE, ACF1 measurements */
        /* Will need to pick off the columns we want to put in the Excel spreadsheet (MAE and RMSE) */
        re.eval("facc_arima<-list();");
        re.eval("facc_ets<-list();");
        re.eval("facc_rwf<-list();");
```

```
        re.eval("for(i in 1:ncc){facc_arima[[i]]<-matrix(NA,nrow=1,ncol=nca)};");
        re.eval("for(i in 1:ncc){facc_ets[[i]]<-matrix(NA,nrow=1,ncol=nca)};");
        re.eval("for(i in 1:ncc){facc_rwf[[i]]<-matrix(NA,nrow=1,ncol=nca)};");

        /* Store the full forecast (ffc) as a list of 1x5 matrices = (in_sample_accuracy_results)x(forecast
items: PointForecast, Lo80, Hi80, Lo95, Hi95) */
        /* It can be expanded to 2x5 when there is a test data passed to accuracy, then the 2nd row would be
test accuracy results */
        re.eval("ffc_arima<-list();");
        re.eval("ffc_ets<-list();");
        re.eval("ffc_rwf<-list();");
        re.eval("for(i in 1:ncc){ffc_arima[[i]]<-matrix(NA,nrow=1,ncol=ncf)};");
        re.eval("for(i in 1:ncc){ffc_ets[[i]]<-matrix(NA,nrow=1,ncol=ncf)};");
        re.eval("for(i in 1:ncc){ffc_rwf[[i]]<-matrix(NA,nrow=1,ncol=ncf)};");

        /* forecast() returns PointForecast, Lo80, Hi80, Lo95, Hi95 (where PointForecast = $mean) */
        re.eval("for(i in 1:ncc){ffc_arima[[i]]<-
forecast(auto.arima(datats[,i],approximation=FALSE,trace=FALSE),h=h)};");
        re.eval("for(i in 1:ncc){ffc_ets[[i]]<-forecast(ets(datats[,i]),h=h)};");
        re.eval("for(i in 1:ncc){ffc_rwf[[i]]<-forecast(rwf(datats[,i],h=h,drift=TRUE),h=h)};");

        /* Extract the PointForecast ($mean) from the full forecast (ffc) and round to 2 decimal places */
        re.eval("for(i in 1:ncc){fcast_arima[,i]<-round(ffc_arima[[i]]$mean,2)};");
        re.eval("for(i in 1:ncc){fcast_ets[,i]<-round(ffc_ets[[i]]$mean,2)};");
        re.eval("for(i in 1:ncc){fcast_rwf[,i]<-round(ffc_rwf[[i]]$mean,2)};");

        /* Round the training set (aka in-sample) accuracy measurements to 2 decimal places */
        re.eval("for(i in 1:ncc){facc_arima[[i]]<-round(accuracy(ffc_arima[[i]]),2)};");
        re.eval("for(i in 1:ncc){facc_ets[[i]]<-round(accuracy(ffc_ets[[i]]),2)};");
        re.eval("for(i in 1:ncc){facc_rwf[[i]]<-round(accuracy(ffc_rwf[[i]]),2)};");

        /* Open the Excel Workbook to write to Predictions and Accuracy Measurements sheets */
        re.eval("file.exists('CourseEnrollmentPredictions.xlsx');");
        re.eval("outwb<-loadWorkbook('CourseEnrollmentPredictions.xlsx',xlsxFile=NULL);");

        /* Write the forecast prediction results to Predictions sheet */
        /* Output the prediction values */

re.eval("writeData(outwb,sheet=1,t(fcast_arima),startRow=1,startCol=1,colNames=FALSE,rowNames=FA
LSE);");

re.eval("writeData(outwb,sheet=1,t(fcast_ets),startRow=1,startCol=1+h,colNames=FALSE,rowNames=FA
LSE);");

re.eval("writeData(outwb,sheet=1,t(fcast_rwf),startRow=1,startCol=1+(2*h),colNames=FALSE,rowNames
=FALSE);");

        /* Add MAE data for each classifier to sheetAcc spreadsheet */
        /* Index to access RMSE from R output is 2, Index to access MAE from R output is 3, Indices start
at 1 */
        re.eval("for(i in
1:ncc){writeData(outwb,sheet=2,data.frame(facc_arima[[i]][3],facc_arima[[i]][2]),startRow=i,startCol=1,c
olNames=FALSE,rowNames=FALSE)};");
        re.eval("for(i in
1:ncc){writeData(outwb,sheet=2,data.frame(facc_ets[[i]][3],facc_ets[[i]][2]),startRow=i,startCol=1+laccm,
colNames=FALSE,rowNames=FALSE)};");
```

```
        re.eval("for(i in
1:ncc){writeData(outwb,sheet=2,data.frame(facc_rwf[[i]][3],facc_rwf[[i]][2]),startRow=i,startCol=1+(2*la
ccm),colNames=FALSE,rowNames=FALSE)};");

        /* Save the Excel Workbook */
        re.eval("saveWorkbook(outwb,'CourseEnrollmentPredictions.xlsx',overwrite=TRUE);");
      }
      re.end();
      /* END OF R CODE SECTION */

    /* Read R predictions and accuracies from Excel spreadsheet before adding headers or Weka data */
    /* R data must be appended to Weka data before writing add data to spreadsheet since writing
overwrites all existing data */
    FileInputStream inputStream = new FileInputStream(new File(pathToOutFile));
    XSSFWorkbook readWorkbook = new XSSFWorkbook(inputStream);

    /* Get prediction data from first sheet */
    XSSFSheet predSheet = readWorkbook.getSheetAt(0);

    Row[] rPredRows = new Row [courses.length];
    int rowIdx = 0;

    /* Go through prediction sheet row by row and store the R forecast data */
    Iterator<Row> predRowIterator = predSheet.iterator();
    while (predRowIterator.hasNext()) {
      rPredRows[rowIdx] = predRowIterator.next();
      rowIdx++;
    }

    /* Get accuracy data from second sheet */
    XSSFSheet accSheet = readWorkbook.getSheetAt(1);

    Row[] rAccRows = new Row [courses.length];
    rowIdx = 0;

    /* Go through accuracy sheet row by row and store the R forecast data */
    Iterator<Row> accRowIterator = accSheet.iterator();
    while (accRowIterator.hasNext()) {
      rAccRows[rowIdx] = accRowIterator.next();
      rowIdx++;
    }

    /* Close the connection to the Excel workbook since done reading */
    inputStream.close();

    /* Create Workbook sheet header style */
    XSSFCellStyle headerStyle = workbook.createCellStyle();
    headerStyle.setBorderLeft(XSSFCellStyle.BORDER_MEDIUM);
    headerStyle.setBorderRight(XSSFCellStyle.BORDER_MEDIUM);
    headerStyle.setBorderTop(XSSFCellStyle.BORDER_MEDIUM);
    headerStyle.setBorderBottom(XSSFCellStyle.BORDER_MEDIUM);

    /* Create Workbook sheet header font */
    XSSFFont headerFont = workbook.createFont();
    headerFont.setFontHeightInPoints((short) 12);
    headerFont.setBoldweight(XSSFFont.BOLDWEIGHT_BOLD);
```

```
    headerStyle.setFont(headerFont);

    /* Map to store the prediction and accuracy data to output to the Excel spreadsheets */
    Map<String, Object[]> dataP = new TreeMap<String, Object[]>();
    Map<String, Object[]> dataA = new TreeMap<String, Object[]>();

    /* Object[Rows][Columns] where */
    /* Rows = the courses for which to predict the total enrollment for numSemestersToForecast steps
ahead in time */
    /* Columns = the predictions for each step ahead in time for each classifier in classifiers[] */
    Object[][] predictions = new Object[courses.length][classifiers.length*numSemestersToForecast];
    Object[][] accuracies = new Object[courses.length][classifiers.length*accuracy.length];

    /* Add the CourseEnrollmentPredictions sheet column headers */
    XSSFRow hrowP1 = sheetP.createRow(0);
    for(int col=0; col<headerP.length; col++){
      XSSFCell hcell = hrowP1.createCell(col);
      hcell.setCellValue(headerP[col]);
      hcell.setCellStyle(headerStyle);
    }

    /* Add the ForecastAccuracy sheet column headers */
    XSSFRow hrowA1 = sheetA.createRow(0);
    XSSFRow hrowA2 = sheetA.createRow(1);
    for(int row=0; row<rowA; row++){
      /* Add 1 in the comparison check due to the first column header being 'Courses' and not a classifier */
      for(int col=0; col<(1+((headerA.length-1)*accuracy.length)); col++){
        /* 1st column is 'Courses' and not a classifier so it is not merged */
        if(col==0){
          if(row==0){
            XSSFCell hcell = hrowA1.createCell(col);
            hcell.setCellValue(headerA[col]);
            hcell.setCellStyle(headerStyle);
          }
          else{ /* Do nothing because row=1, col=0 is empty */ }
        }
        else{
          /* First header classifiers start at column index 1 */
          /* Each classifier name is in a merged cell of size accuracy.length */
          if(row==0){
            if(col==1 || (col%accuracy.length==1)){
              CellRangeAddress range = new CellRangeAddress(0,0,col,col+(accuracy.length-1));
              sheetA.addMergedRegion(range);
              XSSFCell hcell = hrowA1.createCell(col);
              int index = (col/accuracy.length) + 1;
              hcell.setCellValue(headerA[index]);
              hcell.setCellStyle(headerStyle);
              RegionUtil.setBorderRight(XSSFCellStyle.BORDER_MEDIUM, range, sheetA, workbook);
              RegionUtil.setBorderLeft(XSSFCellStyle.BORDER_MEDIUM, range, sheetA, workbook);
              RegionUtil.setBorderTop(XSSFCellStyle.BORDER_MEDIUM, range, sheetA, workbook);
              RegionUtil.setBorderBottom(XSSFCellStyle.BORDER_MEDIUM, range, sheetA,
workbook);
            }
          }
          /* Second header of accuracy measurements starts at column index 1 */
          else {
```

```
            XSSFCell hcell = hrowA2.createCell(col);
            int index = (col+1)%accuracy.length;
            hcell.setCellValue(accuracy[index]);
            hcell.setCellStyle(headerStyle);
          }
        }
      }
    }

    /* Forecaster to get predicted values */
    WekaForecaster forecaster = new WekaForecaster();

    int minLag = 1;
    int maxLag = 3;

    int j = 0;
    int s = 0;

    /* Get predictions and accuracy for each classifier and generate an Excel spreadsheet with the results */
    for(int i=0; i<classifiers.length; i++){

      /* Set the targets we want to forecast */
      forecaster.setFieldsToForecast(String.join(",", courses));

      forecaster.setBaseForecaster(classifiers[i]);
      forecaster.setConfidenceLevel(0.95);

      /* StartDate corresponds to SA_TERM_TBL.BEGIN_DATE */
      forecaster.getTSLagMaker().setTimeStampField("StartDate");
      forecaster.getTSLagMaker().setMinLag(minLag);
      forecaster.getTSLagMaker().setMaxLag(maxLag);

      /* Add month and quarter of the year indicator field */
      forecaster.getTSLagMaker().setAddMonthOfYear(true);
      forecaster.getTSLagMaker().setAddQuarterOfYear(true);

      /* Build the model */
      forecaster.buildForecaster(enrollment, System.out);

      /* Run the model with recent historical data */
      /* There must be at least maxLag amount of data */
      forecaster.primeForecaster(enrollment);

      /* Forecast for numSemestersToForecast units beyond the end of the training data without overlay
data */
      List<List<NumericPrediction>> forecast = forecaster.forecast(numSemestersToForecast,
System.out);

      /* Output the predictions */
      /* Outer j loop is over the steps forward in time & is the Excel spreadsheet columns */
      /* Inner k loop is over the target courses which are the different courses for which to predict
enrollment & is the Excel spreadsheet rows */
      for(j=j; j<((numSemestersToForecast*i)+numSemestersToForecast); j++) {
        List<NumericPrediction> predsAtStep = forecast.get(j%numSemestersToForecast);

        for (int k=0; k<courses.length; k++) {
```

```
            NumericPrediction predForTarget = predsAtStep.get(k);

            /* Fill predictions column by column, storing with 2 decimal points of accuracy as Float objects */
            predictions[k][j] = new Float(df.format(predForTarget.predicted()));
          }
        }

        /* Compute the evaluation metrics */
        TSEvaluation eval = new TSEvaluation(enrollment, percentHoldOut);
        eval.setHorizon(numSemestersToForecast);

        /* Get the Time Series Evaluation Summaries */
        eval.evaluateForecaster(forecaster, System.out);
        String accuracyMeasures = eval.toSummaryString();

        Pattern patternMAE = Pattern.compile("Mean absolute error.+");
        Pattern patternRMSE = Pattern.compile("Root mean squared error.+");

        Matcher matcherMAE = patternMAE.matcher(accuracyMeasures);
        Matcher matcherRMSE = patternRMSE.matcher(accuracyMeasures);

        /* To keep track of which find (aka course) the accuracy measure is for */
        int k = 0;

        /* Each match in String accuracyMeasures is for a forecasted field (a course) for the current classifier
*/
        while(matcherMAE.find()){
          /* Match will look like: "Mean absolute error   330.1107   320.5035" */
          String groupMAE = matcherMAE.group();

          /* Extract the MAE numbers for each forecasted step (total of numSemestersToForecast steps) from
the string */
          Pattern extractMAE = Pattern.compile("\\d+(\\.\\d+)?");
          Matcher matchMAE = extractMAE.matcher(groupMAE);

          float maeTotal = 0.0f;
          while(matchMAE.find()){
            String mae = matchMAE.group();
            maeTotal += Float.parseFloat(mae);
          }

          /* Store the average MAE across all forecasted steps with 2 decimal points of accuracy as Float
objects */
          accuracies[k][s] = new Float(df.format(maeTotal/numSemestersToForecast));
          k++;
        }

        /* Reset for the next find loop */
        k = 0;

        while(matcherRMSE.find()){
          /* Match will look like: "Root mean squared error   415.7785   399.6755" */
          String groupRMSE = matcherRMSE.group();

          /* Extract the RMSE numbers for each forecasted step (total of numSemestersToForecast steps)
from the string */
```

```
          Pattern extractRMSE = Pattern.compile("\\d+(\\.\\d+)?");
          Matcher matchRMSE = extractRMSE.matcher(groupRMSE);

          float rmseTotal = 0.0f;
          while(matchRMSE.find()){
            String rmse = matchRMSE.group();
            rmseTotal += Float.parseFloat(rmse);
          }

          /* Store the RMSE in the column after the MAE column with 2 decimal points of accuracy as Float
objects */
          accuracies[k][s+1] = new Float(df.format(rmseTotal/numSemestersToForecast));
          k++;
        }

        /* Increment the column index by the number of accuracy measures - MAE & RMSE - so by 2 */
        s += accuracy.length;

        /* Reset the forecaster */
        forecaster.reset();

      } /* END for loop that calculates predictions and accuracy of the classifiers */


      /* Write predictions row by row, where each row is predictions for an undergraduate COMP or CIT
course */
      /* The index into the row[] can be thought of as the column in that row */
      for(int m=0; m<courses.length; m++) {
        /* Prepend the Course column data to each prediction row */
        /* Use classifiers.length+rModelCnt to get the total count of all models (Weka and R) */
        Object[] row = new Object [1+((classifiers.length+rModelCnt)*numSemestersToForecast)];

        /* row[0] is a String containing the course name, for example "COMP 100" */
        row[0] = courses[m];

        /* row[1] to row[classfiers.length*numSemestersToForecast] contain the Weka model predictions */
        for(int n=0; n<(classifiers.length*numSemestersToForecast); n++){
          row[n+1] = predictions[m][n];
        }

        /* row[classifiers.length*numSemestersToForecast + 1] to
row[1+((classifiers.length+rModelCnt)*numSemestersToForecast)] */
        /* contain the R forecast model predictions */
        Iterator<Cell> predCellIterator = rPredRows[m].cellIterator();
        rowIdx = (classifiers.length*numSemestersToForecast) + 1;
        while (predCellIterator.hasNext()) {
          Cell rPredCell = predCellIterator.next();
          row[rowIdx++] = new Float(rPredCell.getNumericCellValue());
        }

        /* Insert the row after the header in the Excel spreadsheet */
        dataP.put(Integer.toString(rowP+m), row);
      }

      /* Write accuracies row by row, where each row is the average accuracy measurements of MAE and
RMSE */
```

```
    /* for all predicted steps for each undergraduate COMP or CIT course */
    /* The index into the row[] can be thought of as the column in that row */
    for(int m=0; m<courses.length; m++) {
      /* Prepend the Course column data to each accuracy row */
      /* Use classifiers.length+rModelCnt to get the total count of all models (Weka and R) */
      Object[] row = new Object [1+((classifiers.length+rModelCnt)*accuracy.length)];

      /* row[0] is a String containing the course name, for example "COMP 100" */
      row[0] = courses[m];

      /* row[1] to row[classifiers.length*accuracy.length] contain accuracy data for Weka models */
      for(int n=0; n<(classifiers.length*accuracy.length); n++){
        row[n+1] = accuracies[m][n];
      }

      /* row[classifiers.length*accuracy.length] to
row[1+((classifiers.length+rModelCnt)*accuracy.length)] */
      /* contain accuracy data for R forecast models */
      Iterator<Cell> accCellIterator = rAccRows[m].cellIterator();
      rowIdx = (classifiers.length*accuracy.length) + 1;
      while (accCellIterator.hasNext()) {
        Cell rAccCell = accCellIterator.next();
        row[rowIdx++] = new Float(rAccCell.getNumericCellValue());
      }

      /* Insert the row after the header in the Excel spreadsheet */
      dataA.put(Integer.toString(rowA+m), row);
    }

    /* Write the prediction data to the first Excel spreadsheet CourseEnrollmentPredictions */
    /* Data starts at Row 1, Column 0 (indices start at 0) */
    Set<String> keysetP = dataP.keySet();
    int rownumP = rowP;
    for(String key : keysetP){
      XSSFRow row = sheetP.createRow(rownumP++);
      Object[] objArr = dataP.get(key);

      /* Add cells to the row column by column */
      int cellnumP = 0;
      for(Object obj : objArr){
        XSSFCell cell = row.createCell(cellnumP++);
        /* The predictions are of type Float */
        if(obj instanceof Float){
          cell.setCellValue((Float)obj);
        }
        /* The course names are of type String */
        else {
          /* Treat cell contents as type String by default */
          cell.setCellValue((String)obj);
        }
      }
    }

    /* Write the accuracy data to the second Excel spreadsheet named: ForecastAccuracy.xlsx */
    /* Data starts at Row 2, Column 1 (indices start at 0) */
    Set<String> keysetA = dataA.keySet();
```

```
      int rownumA = rowA;
      for(String key : keysetA){
        XSSFRow row = sheetA.createRow(rownumA++);
        Object[] objArr = dataA.get(key);
        int cellnumA = 0;
        for(Object obj : objArr){
          XSSFCell cell = row.createCell(cellnumA++);
          /* The accuracies are of type Float */
          if(obj instanceof Float){
            cell.setCellValue((Float)obj);
          }
          /* The course names are of type String */
          else {
            /* Treat cell contents as type String by default */
            cell.setCellValue((String)obj);
          }
        }
      }

      /* Write the Headers, Weka, and R output to the xlsx workbook */
      /* The cells that R data will be written to are initialized to an empty string as required by openxlsx's
loadWorkbook */
      outfile = new File(pathToOutFile);
      fileout = new FileOutputStream(outfile);
      workbook.write(fileout);
      fileout.flush();
      fileout.close();
    }
    catch (Exception ex) {
      ex.printStackTrace();
    }
  }
}
```

APPENDIX B: GENERATED COURSE ENROLLMENT PREDICTIONS SPREADSHEET

Courses	GaussianProcesses-1_Sem_Ahead	GaussianProcesses-2_Sem_Ahead	GaussianProcesses-3_Sem_Ahead
CIT101_Enrollment_Total	2.70	135.88	28.04
CIT101L_Enrollment_Total	2.95	134.53	29.62
CIT160_Enrollment_Total	3.94	113.88	54.58
CIT160L_Enrollment_Total	3.94	113.88	54.58
CIT210_Enrollment_Total	4.77	60.78	50.59
CIT210L_Enrollment_Total	4.77	60.78	50.59
CIT270_Enrollment_Total	7.41	65.61	48.61
CIT270L_Enrollment_Total	8.70	64.65	53.20
CIT360_Enrollment_Total	0.48	52.33	32.69
CIT480_Enrollment_Total	2.31	57.18	-9.83
CIT480L_Enrollment_Total	2.31	57.18	-9.83
CIT481_Enrollment_Total	3.16	21.54	67.18
CIT481L_Enrollment_Total	3.16	21.54	67.18
COMP100_Enrollment_Total	21.63	707.24	385.11
COMP100HON_Enrollment_Total	-0.58	27.65	-7.71
COMP105BAS_Enrollment_Total	-3.74	16.73	0.37
COMP108_Enrollment_Total	-9.99	104.52	27.66
COMP110_Enrollment_Total	54.47	206.20	199.90
COMP110L_Enrollment_Total	54.76	206.62	198.72
COMP122_Enrollment_Total	0.10	159.92	109.31
COMP122L_Enrollment_Total	0.24	159.86	108.30
COMP182_Enrollment_Total	13.72	158.31	120.56
COMP182L_Enrollment_Total	14.13	158.31	120.62
COMP196CF_Enrollment_Total	0.31	2.98	1.27
COMP196CFL_Enrollment_Total	0.31	2.98	1.27
COMP196CT_Enrollment_Total	0.24	2.35	1.00
COMP196CTL_Enrollment_Total	0.24	2.35	1.00
COMP222_Enrollment_Total	2.44	89.69	46.50
COMP232_Enrollment_Total	-0.99	6.14	4.93
COMP256_Enrollment_Total	-0.80	28.65	16.95

Courses	GaussianProcesses-1_Sem_Ahead	GaussianProcesses-2_Sem_Ahead	GaussianProcesses-3_Sem_Ahead
COMP256L_Enrollment_Total	-0.80	28.65	16.95
COMP282_Enrollment_Total	5.58	142.13	56.38
COMP296FCL_Enrollment_Total	-0.41	1.40	1.30
COMP296FCS_Enrollment_Total	-0.41	1.40	1.30
COMP300_Enrollment_Total	-0.22	30.97	19.72
COMP310_Enrollment_Total	32.95	66.41	9.01
COMP322_Enrollment_Total	2.78	62.94	92.29
COMP322L_Enrollment_Total	1.52	61.13	99.89
COMP333_Enrollment_Total	3.37	61.22	55.79
COMP380_Enrollment_Total	27.96	62.33	57.20
COMP380L_Enrollment_Total	27.69	61.31	57.19
COMP410_Enrollment_Total	0.90	-6.80	-11.24
COMP424_Enrollment_Total	29.05	72.47	61.23
COMP426_Enrollment_Total	-3.09	-8.00	-0.64
COMP429_Enrollment_Total	-1.74	28.82	14.55
COMP440_Enrollment_Total	-2.97	-9.46	84.38
COMP450_Enrollment_Total	-2.75	-16.75	-14.42
COMP465_Enrollment_Total	-1.57	24.37	-8.69
COMP465L_Enrollment_Total	-1.57	24.37	-8.69
COMP467_Enrollment_Total	1.57	12.49	-3.44
COMP469_Enrollment_Total	0.28	-15.08	26.50
COMP480_Enrollment_Total	0.45	4.40	1.87
COMP480L_Enrollment_Total	0.45	4.40	1.87
COMP484_Enrollment_Total	51.34	67.18	61.18
COMP484L_Enrollment_Total	51.34	67.18	61.18
COMP485_Enrollment_Total	-3.39	47.27	55.36
COMP490_Enrollment_Total	10.72	82.62	-10.24
COMP490L_Enrollment_Total	8.29	90.23	-23.37
COMP491L_Enrollment_Total	-2.68	8.73	92.24
COMP494A_Enrollment_Total	-0.06	0.14	1.17
COMP496ALG_Enrollment_Total	8.45	32.63	44.26
COMP499A_Enrollment_Total	-0.07	-0.04	-0.25

Courses	LinearRegression-1_Sem_Ahead	LinearRegression-2_Sem_Ahead	LinearRegression-3_Sem_Ahead
CIT101_Enrollment_Total	10.11	175.65	-32.08
CIT101L_Enrollment_Total	13.27	174.28	-31.02
CIT160_Enrollment_Total	5.15	153.32	-1.63
CIT160L_Enrollment_Total	5.15	153.32	-1.63
CIT210_Enrollment_Total	7.22	79.91	29.74
CIT210L_Enrollment_Total	7.22	79.91	29.74
CIT270_Enrollment_Total	9.70	85.81	24.83
CIT270L_Enrollment_Total	10.81	81.64	29.23
CIT360_Enrollment_Total	-0.26	66.54	17.36
CIT480_Enrollment_Total	3.93	59.92	-18.04
CIT480L_Enrollment_Total	3.93	59.92	-18.04
CIT481_Enrollment_Total	3.86	24.81	66.26
CIT481L_Enrollment_Total	3.86	24.81	66.26
COMP100_Enrollment_Total	33.43	727.74	225.43
COMP100HON_Enrollment_Total	0.22	23.12	-1.85
COMP105BAS_Enrollment_Total	-3.25	4.67	5.04
COMP108_Enrollment_Total	-13.30	128.14	-12.73
COMP110_Enrollment_Total	60.24	202.23	214.26
COMP110L_Enrollment_Total	61.64	203.25	213.27
COMP122_Enrollment_Total	2.08	184.63	87.60
COMP122L_Enrollment_Total	2.13	184.29	86.04
COMP182_Enrollment_Total	11.73	188.23	70.21
COMP182L_Enrollment_Total	12.75	186.82	71.97
COMP196CF_Enrollment_Total	-0.45	4.63	15.59
COMP196CFL_Enrollment_Total	-0.45	4.63	15.59
COMP196CT_Enrollment_Total	-0.36	3.66	12.31
COMP196CTL_Enrollment_Total	-0.36	3.66	12.31
COMP222_Enrollment_Total	4.36	86.26	32.41
COMP232_Enrollment_Total	3.51	-35.89	37.89
COMP256_Enrollment_Total	0.52	45.45	9.66

Courses	LinearRegression-1_Sem_Ahead	LinearRegression-2_Sem_Ahead	LinearRegression-3_Sem_Ahead
COMP256L_Enrollment_Total	0.52	45.45	9.66
COMP282_Enrollment_Total	6.84	138.82	46.51
COMP296FCL_Enrollment_Total	0.64	-10.14	8.46
COMP296FCS_Enrollment_Total	0.64	-10.14	8.46
COMP300_Enrollment_Total	-0.24	5.01	24.54
COMP310_Enrollment_Total	37.71	54.11	10.77
COMP322_Enrollment_Total	4.42	60.26	101.65
COMP322L_Enrollment_Total	4.37	56.10	113.87
COMP333_Enrollment_Total	10.71	88.57	48.42
COMP380_Enrollment_Total	27.36	56.74	70.07
COMP380L_Enrollment_Total	27.42	55.93	70.12
COMP410_Enrollment_Total	1.38	-13.81	7.22
COMP424_Enrollment_Total	30.72	74.66	54.56
COMP426_Enrollment_Total	-5.24	-8.38	-2.48
COMP429_Enrollment_Total	-2.57	16.71	30.65
COMP440_Enrollment_Total	-4.01	-11.88	118.12
COMP450_Enrollment_Total	-6.04	-42.02	-10.92
COMP465_Enrollment_Total	-1.72	19.99	-6.45
COMP465L_Enrollment_Total	-1.72	19.99	-6.45
COMP467_Enrollment_Total	-0.52	9.47	-5.45
COMP469_Enrollment_Total	3.08	-7.59	24.57
COMP480_Enrollment_Total	-0.67	6.83	22.98
COMP480L_Enrollment_Total	-0.67	6.83	22.98
COMP484_Enrollment_Total	58.16	76.81	51.28
COMP484L_Enrollment_Total	58.16	76.81	51.28
COMP485_Enrollment_Total	-2.03	55.45	45.96
COMP490_Enrollment_Total	12.46	76.32	-9.20
COMP490L_Enrollment_Total	8.50	89.18	-38.97
COMP491L_Enrollment_Total	-0.48	11.28	101.38
COMP494A_Enrollment_Total	-0.03	-0.28	1.92
COMP496ALG_Enrollment_Total	9.28	38.06	41.23
COMP499A_Enrollment_Total	-0.08	0.06	-1.51

Courses	MultilayerPerceptron-1_Sem_Ahead	MultilayerPerceptron-2_Sem_Ahead	MultilayerPerceptron-3_Sem_Ahead
CIT101_Enrollment_Total	31.25	31.25	31.25
CIT101L_Enrollment_Total	31.17	31.17	31.17
CIT160_Enrollment_Total	23.74	89.64	21.17
CIT160L_Enrollment_Total	23.74	89.64	21.17
CIT210_Enrollment_Total	7.18	6.49	18.17
CIT210L_Enrollment_Total	7.18	6.49	18.17
CIT270_Enrollment_Total	9.73	1.56	16.08
CIT270L_Enrollment_Total	171.62	27.57	78.83
CIT360_Enrollment_Total	2.79	0.53	31.14
CIT480_Enrollment_Total	3.97	-2.15	3.96
CIT480L_Enrollment_Total	3.97	-2.15	3.96
CIT481_Enrollment_Total	-2.64	-0.05	40.39
CIT481L_Enrollment_Total	-2.64	-0.05	40.39
COMP100_Enrollment_Total	-118588.18	921.33	1186.88
COMP100HON_Enrollment_Total	-0.79	30.91	-0.24
COMP105BAS_Enrollment_Total	11.85	11.85	11.85
COMP108_Enrollment_Total	3172.20	50.56	34.67
COMP110_Enrollment_Total	66.86	94.82	214.05
COMP110L_Enrollment_Total	66.87	93.75	215.38
COMP122_Enrollment_Total	55.99	55.99	55.99
COMP122L_Enrollment_Total	56.02	56.02	56.02
COMP182_Enrollment_Total	4.48	-37.99	131.39
COMP182L_Enrollment_Total	-56.94	-8.08	104.78
COMP196CF_Enrollment_Total	-12.78	-0.11	69.38
COMP196CFL_Enrollment_Total	-12.78	-0.11	69.38
COMP196CT_Enrollment_Total	-10.09	-0.09	54.78
COMP196CTL_Enrollment_Total	-10.09	-0.09	54.78
COMP222_Enrollment_Total	82.69	80.12	91.23
COMP232_Enrollment_Total	-80.19	3.87	7.18
COMP256_Enrollment_Total	12.42	12.42	12.42

Courses	MultilayerPerceptron-1_Sem_Ahead	MultilayerPerceptron-2_Sem_Ahead	MultilayerPerceptron-3_Sem_Ahead
COMP256L_Enrollment_Total	12.42	12.42	12.42
COMP282_Enrollment_Total	-1007.03	48.03	171.83
COMP296FCL_Enrollment_Total	54.03	-5.66	3.34
COMP296FCS_Enrollment_Total	54.03	-5.66	3.34
COMP300_Enrollment_Total	-0.91	36.90	35.67
COMP310_Enrollment_Total	29.33	29.33	29.33
COMP322_Enrollment_Total	35.76	35.76	35.76
COMP322L_Enrollment_Total	38.00	38.00	38.00
COMP333_Enrollment_Total	-602.20	1.06	55.48
COMP380_Enrollment_Total	-880.21	8.08	65.64
COMP380L_Enrollment_Total	-605.91	18.71	69.56
COMP410_Enrollment_Total	-233.09	-59.84	111.97
COMP424_Enrollment_Total	3.69	17.98	17.98
COMP426_Enrollment_Total	-5.56	12.79	-12.02
COMP429_Enrollment_Total	-1.91	37.49	-7.45
COMP440_Enrollment_Total	-111.17	11.17	11.17
COMP450_Enrollment_Total	157.71	2.08	21.05
COMP465_Enrollment_Total	-2.15	33.41	-1.84
COMP465L_Enrollment_Total	-2.15	33.41	-1.84
COMP467_Enrollment_Total	0.98	28.27	-30.80
COMP469_Enrollment_Total	29.91	-52.83	-66.21
COMP480_Enrollment_Total	-18.83	-0.16	102.25
COMP480L_Enrollment_Total	-18.83	-0.16	102.25
COMP484_Enrollment_Total	61.04	14.44	30.77
COMP484L_Enrollment_Total	61.04	14.44	30.77
COMP485_Enrollment_Total	3.43	31.04	105.96
COMP490_Enrollment_Total	-3828.89	41.69	6412.04
COMP490L_Enrollment_Total	-590.57	-0.84	12.12
COMP491L_Enrollment_Total	12.20	12.21	12.21
COMP494A_Enrollment_Total	0.00	1.00	-0.10
COMP496ALG_Enrollment_Total	2.73	13.48	2.72
COMP499A_Enrollment_Total	-19.68	-0.49	-4.63

Courses	SMOreg-1_Sem_Ahead	SMOreg-2_Sem_Ahead	SMOreg-3_Sem_Ahead
CIT101_Enrollment_Total	8.01	145.84	8.35
CIT101L_Enrollment_Total	8.28	144.31	9.99
CIT160_Enrollment_Total	9.81	121.45	39.79
CIT160L_Enrollment_Total	9.81	121.45	39.79
CIT210_Enrollment_Total	6.91	64.86	46.66
CIT210L_Enrollment_Total	6.91	64.86	46.66
CIT270_Enrollment_Total	10.22	69.62	45.65
CIT270L_Enrollment_Total	11.67	68.63	51.00
CIT360_Enrollment_Total	1.36	56.61	28.76
CIT480_Enrollment_Total	2.84	59.09	-16.93
CIT480L_Enrollment_Total	2.84	59.09	-16.93
CIT481_Enrollment_Total	4.53	23.15	69.67
CIT481L_Enrollment_Total	4.53	23.15	69.67
COMP100_Enrollment_Total	44.44	770.62	188.25
COMP100HON_Enrollment_Total	-0.17	27.95	-13.79
COMP105BAS_Enrollment_Total	-4.34	19.19	-6.02
COMP108_Enrollment_Total	-9.89	109.58	5.32
COMP110_Enrollment_Total	62.86	221.99	178.88
COMP110L_Enrollment_Total	63.10	222.43	177.59
COMP122_Enrollment_Total	4.29	177.92	87.54
COMP122L_Enrollment_Total	4.58	177.96	86.37
COMP182_Enrollment_Total	15.56	179.14	86.38
COMP182L_Enrollment_Total	15.96	179.18	86.76
COMP196CF_Enrollment_Total	-0.17	4.02	3.80
COMP196CFL_Enrollment_Total	-0.17	4.02	3.80
COMP196CT_Enrollment_Total	-0.14	3.18	3.00
COMP196CTL_Enrollment_Total	-0.14	3.18	3.00
COMP222_Enrollment_Total	3.79	95.26	27.81
COMP232_Enrollment_Total	-2.40	9.93	1.18
COMP256_Enrollment_Total	0.03	29.19	12.80

Courses	SMOreg-1_Sem_Ahead	SMOreg-2_Sem_Ahead	SMOreg-3_Sem_Ahead
COMP256L_Enrollment_Total	0.03	29.19	12.80
COMP282_Enrollment_Total	7.52	153.13	38.62
COMP296FCL_Enrollment_Total	-0.85	2.67	-1.07
COMP296FCS_Enrollment_Total	-0.85	2.67	-1.07
COMP300_Enrollment_Total	-0.88	35.51	14.60
COMP310_Enrollment_Total	36.58	71.90	-2.13
COMP322_Enrollment_Total	5.93	66.97	88.84
COMP322L_Enrollment_Total	4.76	64.00	96.58
COMP333_Enrollment_Total	6.12	61.52	49.67
COMP380_Enrollment_Total	28.18	70.83	44.09
COMP380L_Enrollment_Total	27.97	69.86	44.17
COMP410_Enrollment_Total	1.09	-9.68	-9.85
COMP424_Enrollment_Total	31.53	75.77	61.70
COMP426_Enrollment_Total	-4.55	-9.53	-1.60
COMP429_Enrollment_Total	-1.31	24.81	19.10
COMP440_Enrollment_Total	-4.39	-14.57	90.42
COMP450_Enrollment_Total	-4.60	-17.61	-20.20
COMP465_Enrollment_Total	-1.50	24.60	-14.76
COMP465L_Enrollment_Total	-1.50	24.60	-14.76
COMP467_Enrollment_Total	1.60	11.27	-3.84
COMP469_Enrollment_Total	1.94	-18.65	24.01
COMP480_Enrollment_Total	-0.26	5.93	5.60
COMP480L_Enrollment_Total	-0.26	5.93	5.60
COMP484_Enrollment_Total	57.27	74.27	56.27
COMP484L_Enrollment_Total	57.27	74.27	56.27
COMP485_Enrollment_Total	-2.94	51.39	51.31
COMP490_Enrollment_Total	13.77	84.35	-16.30
COMP490L_Enrollment_Total	10.49	93.60	-32.22
COMP491L_Enrollment_Total	-1.26	12.13	92.65
COMP494A_Enrollment_Total	-0.05	-0.03	1.44
COMP496ALG_Enrollment_Total	10.21	32.80	45.32
COMP499A_Enrollment_Total	-0.07	-0.01	-0.50

Courses	Arima-1_Sem_Ahead	Arima-2_Sem_Ahead	Arima-3_Sem_Ahead
CIT101_Enrollment_Total	9.69	119.69	78.69
CIT101L_Enrollment_Total	9.69	118.69	79.69
CIT160_Enrollment_Total	9.88	105.88	87.88
CIT160L_Enrollment_Total	9.88	105.88	87.88
CIT210_Enrollment_Total	6.12	53.12	57.12
CIT210L_Enrollment_Total	6.12	53.12	57.12
CIT270_Enrollment_Total	6.31	60.31	53.31
CIT270L_Enrollment_Total	6.50	59.50	57.50
CIT360_Enrollment_Total	4.69	41.69	42.69
CIT480_Enrollment_Total	0.00	45.00	0.00
CIT480L_Enrollment_Total	0.00	45.00	0.00
CIT481_Enrollment_Total	6.11	27.60	53.73
CIT481L_Enrollment_Total	6.11	27.60	53.73
COMP100_Enrollment_Total	28.00	742.00	659.00
COMP100HON_Enrollment_Total	0.00	29.00	0.00
COMP105BAS_Enrollment_Total	0.00	14.00	15.00
COMP108_Enrollment_Total	0.00	101.00	56.00
COMP110_Enrollment_Total	44.02	207.79	234.04
COMP110L_Enrollment_Total	44.27	207.88	234.22
COMP122_Enrollment_Total	9.31	128.31	154.31
COMP122L_Enrollment_Total	9.31	128.31	154.31
COMP182_Enrollment_Total	29.00	103.00	162.00
COMP182L_Enrollment_Total	29.00	103.00	162.00
COMP196CF_Enrollment_Total	0.00	0.00	0.00
COMP196CFL_Enrollment_Total	0.00	0.00	0.00
COMP196CT_Enrollment_Total	0.00	0.00	0.00
COMP196CTL_Enrollment_Total	0.00	0.00	0.00
COMP222_Enrollment_Total	4.00	98.00	77.00
COMP232_Enrollment_Total	0.00	0.00	0.00
COMP256_Enrollment_Total	3.75	34.75	32.75

Courses	Arima-1_Sem_Ahead	Arima-2_Sem_Ahead	Arima-3_Sem_Ahead
COMP256L_Enrollment_Total	3.75	34.75	32.75
COMP282_Enrollment_Total	16.10	140.10	102.10
COMP296FCL_Enrollment_Total	0.00	0.00	0.00
COMP296FCS_Enrollment_Total	0.00	0.00	0.00
COMP300_Enrollment_Total	1.40	29.30	34.10
COMP310_Enrollment_Total	23.00	70.00	37.00
COMP322_Enrollment_Total	0.00	58.00	89.00
COMP322L_Enrollment_Total	0.00	58.00	92.00
COMP333_Enrollment_Total	11.69	69.73	78.69
COMP380_Enrollment_Total	30.16	58.54	69.48
COMP380L_Enrollment_Total	30.10	58.10	69.31
COMP410_Enrollment_Total	3.79	3.79	3.79
COMP424_Enrollment_Total	48.05	77.39	89.75
COMP426_Enrollment_Total	0.00	0.00	0.00
COMP429_Enrollment_Total	0.00	30.00	0.00
COMP440_Enrollment_Total	0.00	0.00	61.00
COMP450_Enrollment_Total	0.57	0.00	0.00
COMP465_Enrollment_Total	0.00	25.00	0.00
COMP465L_Enrollment_Total	0.00	25.00	0.00
COMP467_Enrollment_Total	0.00	19.00	0.00
COMP469_Enrollment_Total	10.37	7.06	7.06
COMP480_Enrollment_Total	0.00	0.00	0.00
COMP480L_Enrollment_Total	0.00	0.00	0.00
COMP484_Enrollment_Total	35.94	59.94	74.94
COMP484L_Enrollment_Total	35.94	59.94	74.94
COMP485_Enrollment_Total	0.00	31.00	56.00
COMP490_Enrollment_Total	14.59	77.60	7.74
COMP490L_Enrollment_Total	0.00	87.15	0.00
COMP491L_Enrollment_Total	0.00	0.00	89.78
COMP494A_Enrollment_Total	0.08	0.10	0.10
COMP496ALG_Enrollment_Total	0.00	31.00	29.00
COMP499A_Enrollment_Total	0.00	0.00	0.00

Courses	ETS-1_Sem_Ahead	ETS-2_Sem_Ahead	ETS-3_Sem_Ahead
CIT101_Enrollment_Total	1.56	109.97	69.00
CIT101L_Enrollment_Total	2.50	109.16	70.00
CIT160_Enrollment_Total	-0.43	93.94	78.00
CIT160L_Enrollment_Total	-0.43	93.94	78.00
CIT210_Enrollment_Total	3.28	47.93	51.00
CIT210L_Enrollment_Total	3.28	47.93	51.00
CIT270_Enrollment_Total	4.98	54.15	47.00
CIT270L_Enrollment_Total	7.66	54.37	51.00
CIT360_Enrollment_Total	3.69	39.21	38.00
CIT480_Enrollment_Total	-1.51	42.69	0.00
CIT480L_Enrollment_Total	-1.51	42.69	0.00
CIT481_Enrollment_Total	2.43	15.87	47.00
CIT481L_Enrollment_Total	2.43	15.87	47.00
COMP100_Enrollment_Total	23.01	802.67	592.85
COMP100HON_Enrollment_Total	-0.08	27.33	0.10
COMP105BAS_Enrollment_Total	-0.20	17.83	19.46
COMP108_Enrollment_Total	-4.11	98.99	56.00
COMP110_Enrollment_Total	75.51	206.35	193.75
COMP110L_Enrollment_Total	72.98	205.23	195.47
COMP122_Enrollment_Total	7.28	128.32	142.67
COMP122L_Enrollment_Total	7.53	128.42	142.43
COMP182_Enrollment_Total	53.41	128.67	130.94
COMP182L_Enrollment_Total	53.05	128.50	130.83
COMP196CF_Enrollment_Total	1.00	1.00	1.00
COMP196CFL_Enrollment_Total	1.00	1.00	1.00
COMP196CT_Enrollment_Total	0.79	0.79	0.79
COMP196CTL_Enrollment_Total	0.79	0.79	0.79
COMP222_Enrollment_Total	10.26	75.40	74.61
COMP232_Enrollment_Total	-3.27	-3.46	-3.62
COMP256_Enrollment_Total	-0.31	30.59	29.00

Courses	ETS-1_Sem_Ahead	ETS-2_Sem_Ahead	ETS-3_Sem_Ahead
COMP256L_Enrollment_Total	-0.31	30.59	29.00
COMP282_Enrollment_Total	6.78	126.32	86.00
COMP296FCL_Enrollment_Total	0.02	0.02	0.02
COMP296FCS_Enrollment_Total	0.02	0.02	0.02
COMP300_Enrollment_Total	-2.21	27.55	32.16
COMP310_Enrollment_Total	3.84	38.19	43.68
COMP322_Enrollment_Total	4.04	60.04	88.68
COMP322L_Enrollment_Total	4.52	60.50	92.00
COMP333_Enrollment_Total	30.76	70.89	66.28
COMP380_Enrollment_Total	30.27	63.93	63.72
COMP380L_Enrollment_Total	27.75	61.41	63.66
COMP410_Enrollment_Total	3.79	3.79	3.79
COMP424_Enrollment_Total	34.79	54.25	59.00
COMP426_Enrollment_Total	1.63	1.63	1.63
COMP429_Enrollment_Total	6.80	26.64	-0.23
COMP440_Enrollment_Total	10.84	10.84	10.84
COMP450_Enrollment_Total	-1.31	-0.69	0.25
COMP465_Enrollment_Total	0.15	26.06	-0.15
COMP465L_Enrollment_Total	0.15	26.06	-0.15
COMP467_Enrollment_Total	-0.01	14.10	0.02
COMP469_Enrollment_Total	6.63	6.63	6.63
COMP480_Enrollment_Total	1.47	1.47	1.47
COMP480L_Enrollment_Total	1.47	1.47	1.47
COMP484_Enrollment_Total	51.74	60.94	67.01
COMP484L_Enrollment_Total	51.74	60.94	67.01
COMP485_Enrollment_Total	0.69	31.27	56.00
COMP490_Enrollment_Total	0.00	78.00	0.00
COMP490L_Enrollment_Total	0.02	77.00	0.00
COMP491L_Enrollment_Total	0.56	0.58	77.96
COMP494A_Enrollment_Total	0.00	0.00	0.00
COMP496ALG_Enrollment_Total	22.84	22.84	22.84
COMP499A_Enrollment_Total	0.11	0.11	0.11

Courses	RWF-1_Sem_Ahead	RWF-2_Sem_Ahead	RWF-3_Sem_Ahead
CIT101_Enrollment_Total	72.83	76.67	80.50
CIT101L_Enrollment_Total	73.89	77.78	81.67
CIT160_Enrollment_Total	82.33	86.67	91.00
CIT160L_Enrollment_Total	82.33	86.67	91.00
CIT210_Enrollment_Total	53.83	56.67	59.50
CIT210L_Enrollment_Total	53.83	56.67	59.50
CIT270_Enrollment_Total	49.61	52.22	54.83
CIT270L_Enrollment_Total	53.83	56.67	59.50
CIT360_Enrollment_Total	40.11	42.22	44.33
CIT480_Enrollment_Total	0.00	0.00	0.00
CIT480L_Enrollment_Total	0.00	0.00	0.00
CIT481_Enrollment_Total	49.61	52.22	54.83
CIT481L_Enrollment_Total	49.61	52.22	54.83
COMP100_Enrollment_Total	667.22	675.44	683.67
COMP100HON_Enrollment_Total	0.00	0.00	0.00
COMP105BAS_Enrollment_Total	14.39	13.78	13.17
COMP108_Enrollment_Total	57.61	59.22	60.83
COMP110_Enrollment_Total	228.72	234.44	240.17
COMP110L_Enrollment_Total	228.67	234.33	240.00
COMP122_Enrollment_Total	149.94	154.89	159.83
COMP122L_Enrollment_Total	149.94	154.89	159.83
COMP182_Enrollment_Total	167.44	172.89	178.33
COMP182L_Enrollment_Total	167.39	172.78	178.17
COMP196CF_Enrollment_Total	-1.06	-2.11	-3.17
COMP196CFL_Enrollment_Total	-1.06	-2.11	-3.17
COMP196CT_Enrollment_Total	-0.83	-1.67	-2.50
COMP196CTL_Enrollment_Total	-0.83	-1.67	-2.50
COMP222_Enrollment_Total	74.28	75.56	76.83
COMP232_Enrollment_Total	-2.94	-5.89	-8.83
COMP256_Enrollment_Total	30.61	32.22	33.83

Courses	RWF-1_Sem_Ahead	RWF-2_Sem_Ahead	RWF-3_Sem_Ahead
COMP256L_Enrollment_Total	30.61	32.22	33.83
COMP282_Enrollment_Total	87.67	89.33	91.00
COMP296FCL_Enrollment_Total	-0.89	-1.78	-2.67
COMP296FCS_Enrollment_Total	-0.89	-1.78	-2.67
COMP300_Enrollment_Total	30.67	29.33	28.00
COMP310_Enrollment_Total	36.83	36.67	36.50
COMP322_Enrollment_Total	91.67	94.33	97.00
COMP322L_Enrollment_Total	94.72	97.44	100.17
COMP333_Enrollment_Total	70.72	74.44	78.17
COMP380_Enrollment_Total	81.61	83.22	84.83
COMP380L_Enrollment_Total	81.61	83.22	84.83
COMP410_Enrollment_Total	-1.22	-2.44	-3.67
COMP424_Enrollment_Total	61.06	63.11	65.17
COMP426_Enrollment_Total	0.00	0.00	0.00
COMP429_Enrollment_Total	0.00	0.00	0.00
COMP440_Enrollment_Total	63.00	65.00	67.00
COMP450_Enrollment_Total	-1.94	-3.89	-5.83
COMP465_Enrollment_Total	0.00	0.00	0.00
COMP465L_Enrollment_Total	0.00	0.00	0.00
COMP467_Enrollment_Total	0.00	0.00	0.00
COMP469_Enrollment_Total	0.00	0.00	0.00
COMP480_Enrollment_Total	-1.56	-3.11	-4.67
COMP480L_Enrollment_Total	-1.56	-3.11	-4.67
COMP484_Enrollment_Total	70.61	73.22	75.83
COMP484L_Enrollment_Total	70.61	73.22	75.83
COMP485_Enrollment_Total	57.89	59.78	61.67
COMP490_Enrollment_Total	0.00	0.00	0.00
COMP490L_Enrollment_Total	0.00	0.00	0.00
COMP491L_Enrollment_Total	82.33	86.67	91.00
COMP494A_Enrollment_Total	0.00	0.00	0.00
COMP496ALG_Enrollment_Total	30.61	32.22	33.83
COMP499A_Enrollment_Total	0.00	0.00	0.00

APPENDIX C: GENERATED FORECAST ACCURACY SPREADSHEET

Courses	GaussianProcesses		LinearRegression		MultilayerPerceptron		SMOreg		Arima		ETS		RWF	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
CIT101_Enrollment_Total	18.43	22.82	42.76	54.11	2208.41	2879.93	28.76	38.46	10.72	14.34	12.34	17.05	42.35	53.30
CIT101L_Enrollment_Total	18.30	22.63	42.27	53.41	2413.46	3150.12	28.61	38.26	10.51	14.10	12.30	16.73	42.09	52.87
CIT160_Enrollment_Total	18.51	22.49	42.60	55.06	3833.06	5058.18	29.51	39.61	9.54	13.81	12.06	17.00	36.81	49.84
CIT160L_Enrollment_Total	18.51	22.49	42.60	55.06	3833.06	5058.18	29.51	39.61	9.54	13.81	12.06	17.00	36.81	49.84
CIT210_Enrollment_Total	8.75	11.04	20.30	26.47	14.40	18.68	14.74	19.09	5.05	5.88	6.29	8.02	13.93	20.11
CIT210L_Enrollment_Total	8.75	11.04	20.30	26.47	14.40	18.68	14.74	19.09	5.05	5.88	6.29	8.02	13.93	20.11
CIT270_Enrollment_Total	8.65	10.98	21.04	27.25	15.17	20.01	14.24	18.69	5.28	6.93	5.80	8.64	14.08	20.49
CIT270L_Enrollment_Total	8.52	10.81	20.95	27.59	206.49	290.83	13.80	18.27	5.53	7.17	5.84	8.66	13.35	19.60
CIT360_Enrollment_Total	6.91	8.45	14.02	18.01	2041.90	2601.02	10.36	13.61	4.93	6.08	4.78	7.26	10.38	15.00
CIT480_Enrollment_Total	6.91	9.18	8.60	11.77	2617.09	3358.46	9.10	13.02	3.95	8.29	4.97	8.41	10.89	19.38
CIT480L_Enrollment_Total	6.91	9.18	8.60	11.77	2617.09	3358.46	9.10	13.02	3.95	8.29	4.97	8.41	10.89	19.38
CIT481_Enrollment_Total	5.90	8.02	7.39	10.18	15.36	23.04	7.88	10.61	2.64	5.42	3.74	7.50	9.95	15.92
CIT481L_Enrollment_Total	5.90	8.02	7.39	10.18	15.36	23.04	7.88	10.61	2.64	5.42	3.74	7.50	9.95	15.92
COMP100_Enrollment_Total	207.62	246.27	344.67	421.30	28146.52	45486.58	262.77	323.38	59.23	97.55	41.51	67.22	528.63	588.84
COMP100HON_Enrollment_Total	6.52	7.82	8.07	9.60	8.31	10.22	8.26	9.93	0.32	0.78	0.50	0.85	18.33	22.49
COMP105BAS_Enrollment_Total	5.32	6.51	5.96	7.03	497.81	646.43	7.09	8.87	2.42	3.97	2.30	3.04	14.06	16.03
COMP108_Enrollment_Total	19.64	25.15	46.23	59.68	6297.50	9822.62	29.21	37.75	7.58	12.39	8.17	12.19	58.04	63.94
COMP110_Enrollment_Total	31.47	39.23	67.03	85.40	48.98	60.91	49.21	62.43	16.60	21.87	18.74	23.83	93.39	106.87
COMP110L_Enrollment_Total	31.62	39.45	67.48	86.13	48.61	60.75	49.87	63.35	16.67	22.14	18.63	23.87	93.44	106.89
COMP122_Enrollment_Total	23.15	29.11	50.76	65.76	5789.92	7462.34	36.24	47.34	14.23	18.23	14.02	18.94	61.05	73.88
COMP122L_Enrollment_Total	23.11	29.08	50.59	65.64	5789.97	7462.42	36.22	47.34	14.23	18.12	13.86	18.86	61.06	73.85
COMP182_Enrollment_Total	22.79	29.08	47.71	60.20	9143.89	11596.70	35.13	43.93	18.11	24.65	18.03	23.22	57.27	67.69
COMP182L_Enrollment_Total	22.70	28.96	47.10	59.35	9153.07	11609.19	34.92	43.67	17.95	24.51	17.80	22.99	57.21	67.55
COMP196CF_Enrollment_Total	2.32	2.76	5.58	7.36	256.00	321.66	3.88	4.66	1.00	4.36	1.90	4.24	1.99	4.35
COMP196CFL_Enrollment_Total	2.32	2.76	5.58	7.36	256.00	321.66	3.88	4.66	1.00	4.36	1.90	4.24	1.99	4.35
COMP196CT_Enrollment_Total	1.83	2.18	4.41	5.81	202.11	253.94	3.06	3.68	0.79	3.44	1.50	3.35	1.57	3.44
COMP196CTL_Enrollment_Total	1.83	2.18	4.41	5.81	202.11	253.94	3.06	3.68	0.79	3.44	1.50	3.35	1.57	3.44
COMP222_Enrollment_Total	17.71	21.30	35.54	45.61	666.42	934.81	25.38	32.20	5.90	7.97	6.17	8.50	46.36	54.11
COMP232_Enrollment_Total	12.14	15.80	43.14	61.37	725.32	930.08	20.11	26.90	6.75	13.41	7.70	11.32	9.91	18.12

Courses	GaussianProcesses		LinearRegression		MultilayerPerceptron		SMOreg		Arima		ETS		RWF	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
COMP256_Enrollment_Total	9.13	11.22	20.56	29.53	105.27	129.85	14.80	19.97	4.13	5.74	3.86	6.24	14.48	19.23
COMP256L_Enrollment_Total	9.13	11.22	20.56	29.53	105.27	129.85	14.80	19.97	4.13	5.74	3.86	6.24	14.48	19.23
COMP282_Enrollment_Total	17.22	22.96	34.25	44.57	1507.40	2165.02	25.29	32.74	9.54	13.10	10.30	14.22	49.48	56.57
COMP296FCL_Enrollment_Total	4.76	6.46	10.95	15.44	162.07	212.26	7.66	10.31	2.74	6.77	3.79	5.75	3.58	6.90
COMP296FCS_Enrollment_Total	4.76	6.46	10.95	15.44	162.07	212.26	7.66	10.31	2.74	6.77	3.79	5.75	3.58	6.90
COMP300_Enrollment_Total	9.41	11.50	15.25	18.24	17.06	22.16	13.07	16.12	3.33	4.58	5.22	6.59	22.81	27.36
COMP310_Enrollment_Total	11.21	14.20	16.85	23.86	1598.30	2071.91	14.11	19.85	7.53	13.46	8.15	11.01	30.87	34.60
COMP322_Enrollment_Total	11.64	14.76	15.16	20.06	4202.58	5402.18	13.01	16.22	5.58	8.52	6.35	8.42	37.59	43.66
COMP322L_Enrollment_Total	12.72	16.25	13.43	16.59	4441.26	5712.96	13.20	16.54	6.16	9.00	6.95	9.17	39.98	45.96
COMP333_Enrollment_Total	15.49	19.39	38.55	55.18	544.51	767.35	26.78	35.55	9.03	12.26	10.44	12.24	26.66	34.18
COMP380_Enrollment_Total	11.72	14.33	25.40	34.24	644.56	944.37	17.09	21.77	5.19	8.21	7.56	9.64	26.17	29.73
COMP380L_Enrollment_Total	11.71	14.30	25.26	34.03	560.17	811.30	17.02	21.70	5.13	8.19	7.60	9.54	26.17	29.72
COMP410_Enrollment_Total	4.38	5.96	6.53	7.57	222.87	278.25	6.16	7.61	6.38	8.82	6.38	8.82	7.59	12.90
COMP424_Enrollment_Total	9.66	11.92	10.85	14.62	3920.96	4942.90	11.02	14.39	4.44	7.26	8.22	10.55	19.06	22.11
COMP426_Enrollment_Total	3.13	4.22	5.64	7.89	9.15	11.29	4.48	5.60	1.63	5.16	2.92	4.89	3.44	7.49
COMP429_Enrollment_Total	6.90	8.93	19.94	27.88	14.87	17.99	9.75	12.71	2.42	4.78	3.80	5.74	18.11	21.70
COMP440_Enrollment_Total	10.49	13.68	15.01	19.51	8464.22	10894.94	14.41	18.87	9.05	16.74	14.83	18.47	17.44	25.98
COMP450_Enrollment_Total	8.87	11.25	9.71	12.16	3035.82	3890.36	12.41	15.95	4.85	7.42	6.43	8.30	14.37	19.11
COMP465_Enrollment_Total	6.13	7.37	7.26	8.61	7.99	10.22	7.62	9.30	0.69	1.64	0.95	1.76	17.11	21.11
COMP465L_Enrollment_Total	6.13	7.37	7.26	8.61	7.99	10.22	7.62	9.30	0.69	1.64	0.95	1.76	17.11	21.11
COMP467_Enrollment_Total	5.25	7.02	8.58	12.10	12.11	14.68	7.42	10.10	2.58	7.16	3.12	6.22	9.44	14.68
COMP469_Enrollment_Total	9.01	12.27	11.37	13.73	420.45	541.37	11.04	14.13	9.77	12.64	10.47	14.01	14.00	22.51
COMP480_Enrollment_Total	3.42	4.06	8.23	10.84	377.26	474.02	5.72	6.87	1.47	6.42	2.79	6.25	2.94	6.41
COMP480L_Enrollment_Total	3.42	4.06	8.23	10.84	377.26	474.02	5.72	6.87	1.47	6.42	2.79	6.25	2.94	6.41
COMP484_Enrollment_Total	7.32	9.75	15.52	20.95	26.55	31.95	11.38	15.06	8.57	11.80	7.59	9.24	11.19	14.12
COMP484L_Enrollment_Total	7.32	9.75	15.52	20.95	26.55	31.95	11.38	15.06	8.57	11.80	7.59	9.24	11.19	14.12
COMP485_Enrollment_Total	9.71	11.33	11.44	14.18	468.91	627.98	13.00	16.11	5.74	11.03	7.05	10.80	20.75	26.21
COMP490_Enrollment_Total	10.70	14.43	13.35	17.12	20866.73	25900.70	14.75	20.96	5.72	7.99	5.77	10.31	25.44	36.17
COMP490L_Enrollment_Total	11.49	15.33	15.94	20.80	4264.10	5668.49	15.55	21.54	2.25	5.03	5.75	10.10	26.78	38.83
COMP491L_Enrollment_Total	10.46	13.92	10.74	14.83	6691.33	8462.90	13.60	18.16	2.32	5.25	4.86	8.38	23.44	33.47
COMP494A_Enrollment_Total	0.40	0.55	0.94	1.35	1.24	1.54	0.76	1.06	0.14	0.22	0.20	0.30	0.11	0.33
COMP496ALG_Enrollment_Total	5.11	6.45	9.29	11.81	1237.43	1588.41	6.81	8.96	3.16	9.59	4.00	9.31	6.20	11.75
COMP499A_Enrollment_Total	0.21	0.30	0.40	0.52	4.12	6.53	0.33	0.44	0.11	0.32	0.19	0.31	0.22	0.47