

**ACLC COLLEGE OF BUTUAN**

**Franchised and Operated by Butuan Information Technology Service, Inc. (BSIT)**

HDS Bldg, 999 J.C. Aquino Avenue,Butuan City, Agusan Del Norte, Philippines 8600

**Machine Learning-Based Time Series Forecasting for** **Enrollment Predictions of All Courses in ACLC College of** **Butuan**

A Thesis Project Presented to the faculty of

COMPUTER EDUCATION DEPARTMENT

In Partial Fulfillment of

The Requirements for the Degree in

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

Submitted by:

**ELFIE ME S. LAMPINAQUE**

**HAROLD JEY N. MADJOS**

**REANIER JAMES P. VERTUDEZ**

**SEPTEMBER 2025**

# Table of Content

1.1 Background of the Study…

1.2 Statement of the Problem

1.3 Objectives of the Study 1.3.1 General Objectives

1.3.2 Specific Objectives

1.4 Significance of the Study

1.5 Scope and Limitations 1.5.1 Scope

1.5.2 Limitations .

1.6 Definition of Terms

.

# **CHAPTER 1**

**INTRODUCTION**

## 1.1 Background of the Study

With the rapid advancement of technology and artificial intelligence, forecasting through machine learning has become an essential tool across various fields because it enables the prediction of future outcomes from historical data (Bousnguar et al., 2022). In the education sector, one of the most important applications of forecasting is predicting student enrollment. Accurate enrollment forecasts allow schools to prepare ahead of time, avoiding problems such as overstaffing or shortages of teachers, classrooms, and learning materials (Singh & Pardhi, 2024; James & Weese, 2022). However, enrollment forecasting is not always accurate. Sudden fluctuations in student numbers can still occur, which may result in overcrowded classrooms or underutilized resources (Singh & Pardhi, 2024). To address these challenges, many institutions have begun adopting smart forecasting systems that use machine learning to analyze past enrollment records and generate more reliable predictions. These systems are designed to support school administrators in planning and making informed decisions (Abbasimehr & Paki, 2022).

Several forecasting models have been applied in student enrollment prediction. Traditional methods such as the Autoregressive Integrated Moving Average (ARIMA) model focus on identifying linear patterns and trends in historical data (Siami-Namini et al., 2019). While effective in earlier studies with smaller datasets, ARIMA is limited when handling complex and dynamic enrollment patterns. With the growing availability of larger datasets, researchers have increasingly turned to machine learning and deep learning approaches, which can capture both linear and nonlinear behaviors in the data (Mehtab & Sen, 2020). One of the most powerful deep learning models in this context is the Long

Short-Term Memory (LSTM) network. LSTM is particularly suited for time-series data, such as semester-based enrollment records, because of its ability to retain long-term dependencies while addressing the vanishing gradient problem in recurrent neural networks. Studies have consistently shown that LSTM outperforms ARIMA in forecasting accuracy, especially when dealing with large or highly variable datasets (Siami-Namini et al., 2019; Elsaraiti & Merabet, 2021). Advanced approaches such as CNN-LSTM and attention-based LSTM architectures have also been proposed to further enhance accuracy (Abbasimehr & Paki, 2022; Mehtab & Sen, 2020; Ghanbari & Borna, 2021). In relevance to this, Bousnguar et al. (2022a) developed a web-based enrollment forecasting system using LSTM to address the limitations of traditional methods. Their study demonstrated that deep learning models significantly reduce forecasting errors compared to manual estimation or simple statistical techniques, which often led to resource misallocation such as over-hiring or insufficient staffing. The results confirmed the practical benefits of using LSTM-based systems in supporting institutional planning.

Building upon this foundation, the present study aims to design a **web application for time-series forecasting of student enrollment at ACLC College of Butuan**. Unlike prior works that rely on a single model, the proponents will evaluate at least four machine learning models, with a focus on LSTM due to its proven accuracy advantage over traditional approaches like ARIMA. By comparing models, the study seeks to identify the most effective approach for predicting future enrollment. The forecasts will provide valuable insights for institutional decision-making in resource allocation, faculty hiring, academic planning, and marketing strategies. The dataset will consist of enrollment records from **School Year 2018–2019 (1st semester) to School Year 2025–2026 (2nd semester)**, gathered from the registrar’s office of ACLC College of Butuan. Currently, the institution lacks an automated enrollment forecasting system; decisions are made based on historical numbers without reliable predictive analytics. This study addresses that gap by developing a data-driven solution tailored to the institution’s needs, ensuring more informed and proactive planning.

## ****1.2 Statement of the Problem****

The ACLC College of Butuan currently does not have an automated system to forecast student enrollment. Institutional decisions, planning, and resource allocation rely heavily on past trends, manual estimation, or simple statistics, which may not always produce accurate results. This study seeks to address the following issues:

1. The absence of an automated forecasting system for student enrollment at ACLC College of Butuan.
2. Reliance on manual estimation and basic statistical methods, which may lead to inaccurate enrollment predictions.
3. The lack of a reliable, data-driven tool to support decision-making in resource allocation and academic planning.
4. The need for a forecasting model that improves the accuracy of enrollment predictions through the application of machine learning.

## 1.3 Objectives of the Study

### 1.3.1 General Objectives

The proponents aim to design and implement a machine learning-based time- series forecasting model for student enrollment prediction at ACLC College of Butuan using Long-Short Term Memory (LSTM).

### 1.3.2 Specific Objectives

The study specifically aims to;

* Collect and preprocess historical enrollment datasets, including the number of enrollees per course, year, department, semester, and year level.
* Design and develop a web application capable of forecasting student enrollment based on historical data.
* Analyze historical enrollment trends and patterns to support institutional planning.
* Provide a predictive model that assists the institution in preparing for future enrollment demands.
* Develop a prototype dashboard that visualizes both historical and projected enrollment data for decision-making purposes.
* Evaluate and compare the accuracy of different time series forecasting models, including ARIMA, Prophet, and LSTM.
* Focus primarily on the LSTM model as the proposed forecasting approach; however, if comparative results show higher accuracy from another model, that model may be temporarily used in the web application.
* Address potential inconsistencies in the study by integrating a continuous monitoring mechanism within the web application, ensuring the model is updated whenever new datasets are added.

## Significance of the Study

* ACLC College of Butuan Administrators – Aids decision-making in allocating resources, faculty recruitment, and course planning through enrollment forecasting.
* Faculty and Staff – Aids in forecasting class sizes and distribution of workloads, to improve planning for teaching and management.
* Researchers – Reference for future machine learning-based forecasting studies in education and advancing the practice of predictive analytics.
* Education Institutions – Depicts how enrollment management could be enhanced by AI and machine learning, and can serve as an example for other institutions to embrace data- driven decision-making.
* Policy Makers and Government Agencies – Facilitates educational planning and policymaking by giving insight into enrollment trends and growth in student population.

## Scope and Limitations

### 1.5.1 Scope

* Development of a system that predicts student enrollment at ACLC College of Butuan using LSTM as the primary model.
* The system will display historical and forecasted enrollment data through a web-based application.
* Predictions will extend up to at least three academic years in advance.
* The dataset will include past enrollment records from S.Y. 2018 to S.Y. 2025 (1st and 2nd semesters).
* Additional models (e.g., ARIMA, Prophet) will be explored for performance comparison.
* The system will assist administrators in planning, hiring, and resource management based on forecasted enrollment trends.

### 1.5.2 Limitations

* Predictions depend on the accuracy and completeness of the historical dataset.
* The model does not account for external, real-time events such as sudden economic changes, policy shifts, or unexpected circumstances affecting enrollment.
* The system is designed specifically for ACLC College of Butuan and is not generalized to other institutions.
* Personal and individual factors affecting student enrollment decisions are not included.
* The system only forecasts enrollment by course and department within the institution.

## 1.6 Definition of Terms

* **Machine Learning (ML):** A branch of artificial intelligence that enables computers to learn from data and make predictions or decisions without explicit programming.
* **Time Series:** A sequence of data points collected at regular time intervals, such as enrollment figures by semester or year.
* **Forecasting:** The process of using historical data to predict future values or trends.
* **LSTM (Long Short-Term Memory):** A type of recurrent neural network (RNN) designed to learn long-term dependencies in sequential data, particularly useful for time series forecasting.
* **Enrollment Prediction:** The process of estimating future student enrollment numbers based on historical trends and patterns.
* **Neural Network:** A computational model inspired by the human brain, consisting of interconnected layers of "neurons" that process and learn data patterns.
* **Epoch:** One complete pass of the training dataset during the model training process.
* **Overfitting:** A condition where a model learns the training data too well, including noise and outliers, resulting in poor performance on new data.
* **Underfitting:** A condition where a model is too simplistic and fails to capture the underlying patterns of the dataset.
* **Mean Squared Error (MSE):** A measure of model accuracy that calculates the average squared difference between predicted and actual values.
* **Root Mean Squared Error (RMSE):** A metric derived from MSE that represents the average magnitude of error in predictions.
* **R-squared (R²):** A statistical measure indicating how well a model explains the variance in the observed data. Values closer to 1 imply better performance.
* **Training Data:** The portion of the dataset used to teach the model patterns and relationships.
* **Testing Data:** The portion of the dataset used to evaluate model performance on unseen data.
* **Recurrent Neural Network (RNN):** A class of neural networks designed for sequential data, with LSTM being one of its improved variants.
* **Multivariate Time Series:** A dataset containing multiple variables recorded over time, such as enrollment by course, semester, and department.
* **Trend Analysis:** The study of historical data to identify consistent patterns or trends that may continue in the future.
* **Feature Engineering:** The process of selecting, creating, and transforming variables to improve model performance.
* **Dataset:** A structured collection of data, in this study referring to historical enrollment records.
* **Predictive Analytics:** The use of statistical and machine learning techniques to analyze historical data and predict future outcomes.

# **CHAPTER 2**

**REVIEW OF RELATED LITERATURE**

This chapter contains a literature review and studies that are relevant to enrollment forecasting and machine learning models that are relevant to this study. Investigating existing research gives a foundation understanding and identifies effective methods to forecast the student enrollment trends.

## 2.1 Related Literature

### 2.1.1 Enrollment Forecasting.

Enrollment forecasting has become increasingly important for educational institutions as they seek to predict student enrollment trends and make informed decisions regarding staffing, resource allocation, and facilities management. Accurate enrollment forecasts enable schools to better prepare for future needs and prevent resource shortages or underutilization (Singh & Pardhi, 2024). Several studies have searched different forecasting methods, with machine learning techniques offering promising results in comparison to traditional methods. (Siami-Namini et al., 2019a) demonstrated the effectiveness of Long Short-Term Memory (LSTM) networks in handling the complexities of time-series data for enrollment prediction. LSTM networks are particularly useful in situations where historical data contains complex and nonlinear trends, as they can capture long-term dependencies in the data. LSTM models, when trained on historical enrollment

data, have been shown to provide more accurate forecasts compared to methods like ARIMA, especially in environments with fluctuating patterns or unexpected factors.

Traditional methods such as ARIMA and exponential smoothing have been used extensively in enrollment forecasting, However, these models often struggle to adapt to sudden changes in enrollment trends, such as those caused by demographic shifts or economic factors. In contrast, machine learning methods, particularly LSTM networks, offer a more robust solution. According to (Abbasimehr & Paki, 2022), LSTM models excel at predicting future trends even when the data is noisy or incomplete. Their ability to adapt to changing patterns over time makes them ideal for enrollment forecasting, where sudden changes in factors like population demographics or school policies can lead to shifts in enrollment numbers.

Furthermore, (Mehtab & Sen, 2019) explored the use of hybrid models, combining CNN (Convolutional Neural Networks) with LSTM networks to improve the accuracy of forecasts. This hybrid approach enables the model to extract important features from the data, such as patterns in the enrollment trends, and then use LSTM to make accurate predictions, such hybrid models have shown improved performance, especially when dealing with large datasets or complex, multi-variable trends in student enrollment.

The integration of machine learning techniques into enrollment forecasting offers several benefits, including increased accuracy, adaptability to new data, and the ability to handle complex, multi-dimensional datasets. However, challenges remain, such as the need for large datasets and specialized knowledge in implementing machine learning models. Despite these challenges, the advantages of machine learning-based forecasting, particularly in terms of accuracy and adaptability, make it a promising approach for predicting student enrollment trends (Siami-Namini et al., 2019; Abbasimehr & Paki, 2022).

### 2.1.2 Machine Learning in Enrollment Prediction

In recent years, predicting student enrollment has become increasingly important for educational institutions. Accurate enrollment forecasting allows schools to prepare better for the future, ensuring they have enough resources like teachers, classrooms, and materials to meet the needs of their students. Traditional forecasting methods have had some success, but with the rise of machine learning (ML) and deep learning (DL), enrollment prediction has seen significant improvements. This section reviews how machine learning models, particularly Long Short-Term Memory (LSTM) networks, have been used to forecast student enrollment. One of the most popular machine learning models used in enrollment forecasting is theLong Short-Term Memory (LSTM) network, which is a type of recurrent neural network (RNN). LSTM is designed to work with time-series data, such as past enrollment records, to predict future values. This makes it an ideal tool for forecasting student numbers, which often show seasonal or yearly patterns.

(Siami-Namini et al., 2018) conducted an in-depth comparison between traditional time series forecasting methods like ARIMA (AutoRegressive Integrated Moving Average) and modern deep learning techniques such as LSTM (Long Short-Term Memory) and BiLSTM (Bidirectional Long Short-Term Memory). ARIMA has long been a staple for time series forecasting because of its simplicity and ability to model linear relationships in data. However, one of the main drawbacks of ARIMA is its limitation when dealing with non-linear relationships, which often occur in real-world data that exhibits complex patterns over time. For instance, in educational settings, student enrollment data is influenced by numerous dynamic factors, such as socioeconomic trends, policy changes, and population movements, which ARIMA struggles to capture effectively.

In contrast, LSTM and BiLSTM models, both of which are types of Recurrent Neural Networks (RNNs), are well-suited to modeling sequential data with long-term dependencies, making them ideal for time series analysis. LSTM is designed to overcome the vanishing gradient problem that traditional RNNs face, allowing it to retain information over longer periods. This ability to capture long-term dependencies makes LSTM particularly powerful when applied to enrollment forecasting, where trends may change over time due to various factors, such as shifts in local demographics or government education policies. The key advantage of BiLSTM over standard LSTM is that it processes data in both forward and backward directions, enabling it to capture patterns not only from past data but also from future data. This bidirectional nature can significantly improve prediction accuracy by allowing the model to leverage both the historical context and the future trends that influence student enrollment patterns.

(Singh & Pardhi, 2024)further emphasized the limitations of traditional methods, pointing out that while ARIMA has been widely used in the past, it fails to deliver reliable results when there are significant changes in trends or patterns. In contrast, machine learning models like LSTM do not require the same assumptions about data patterns. These models learn directly from the data, meaning they can continuously improve as new enrollment information is added. Singh and Pardhi explained that by using historical enrollment data, machine learning algorithms can recognize underlying patterns that may not be obvious at first glance, making predictions more reliable. They also noted that machine learning models can easily incorporate multiple factors, such as academic year, semester, and course offerings, which traditional methods struggle to manage.

(Abbasimehr & Paki, 2022)focused on improving time series forecasting using LSTM and attention mechanisms. Their study demonstrated that combining LSTM with attention mechanisms can further enhance the accuracy of predictions by allowing the model to focus on important parts of the data, such as specific enrollment periods that show significant changes. This can be particularly useful when there are sudden spikes or drops in enrollment, which can often lead to resource management problems. Their approach showed that by adding an attention mechanism to LSTM, the model becomes more sensitive to critical changes in the data, leading to better decision-making for school administrators.

In addition, LSTM models are capable of handling noisy or incomplete data, which is a common challenge when dealing with real-world enrollment data. In many cases, data may be missing due to various reasons, such as changes in school administration or incomplete records. LSTM's ability to handle such imperfections and still make reliable predictions makes it a strong choice for institutions seeking accurate enrollment forecasts. Another significant advantage of LSTM is its ability to predict long-term trends, which is significant for institutions that need to plan for several years ahead, (Kong et al., 2024)examined how LSTM can be used for long-term forecasting of enrollment trends. They found that LSTM is particularly effective for institutions that need to predict future enrollment over multiple years. For example, a university might want to know how the number of incoming students will change over the next 5 or 10 years. This kind of long-term prediction is important for planning infrastructure, staffing, and resource allocation. (Kong et al., 2024) demonstrated that LSTM’s ability to learn from long-term patterns and adapt to new data allows it to make accurate predictions, even for complex datasets with long timeframes. Their study also noted that LSTM models could capture cyclical trends, such as a rise in enrollment during certain years (e.g., following the launch of new programs) or declines due to external factors (e.g., economic downturns). This ability to predict long-term trends is a major advantage over traditional methods, which may only be suitable for short-term forecasts. (Mehtab & Sen, 2019)explored the potential of combining LSTM with other machine learning techniques to improve the accuracy of enrollment predictions.

Specifically, they used a hybrid model that combined LSTM with Convolutional Neural Networks (CNN). CNN is typically used in image processing, but when paired with LSTM, it can help extract spatial features from the data that might be difficult for LSTM to capture on its own. This hybrid model, called CNN-LSTM, can process both the spatial and temporal components of data, leading to better predictions. Their research showed that the CNN-LSTM hybrid model could extract complex patterns from historical enrollment data and make more accurate forecasts. This is particularly useful for schools offering a wide variety of courses or programs, where the enrollment numbers may vary greatly between departments. By using CNN to extract spatial features (such as trends in specific courses) and LSTM to handle time-series data, the hybrid model provided more robust and precise predictions, even for complex enrollment datasets.

The integration of machine learning into enrollment forecasting has a profound impact on decision-making. By using LSTM and other machine learning techniques, schools can predict enrollment trends with greater confidence. (Singh & Pardhi, 2024)argued that these predictive insights help schools make better decisions about resource allocation, staffing, and facility management. For instance, if a school can predict a significant increase in enrollment for the next semester, it can prepare by hiring more teachers or expanding classroom space, similarly, if enrollment is expected to decline, schools can avoid wasting resources by scaling down operations, additionally, machine learning models allow schools to adjust their strategies based on new data, as more data becomes available, the machine learning model can re-train itself and update its predictions, ensuring that the forecasts remain accurate. This is a significant advantage over traditional methods, which often require manual adjustments and updates.

This study is related to my research because it talks about how LSTM models can be used to predict student enrollment, which is also the main goal of our study. The proponent’s study is about forecasting the number of students enrolling in different courses at ACLC College of Butuan. The studies reviewed here explain that LSTM is a good choice for this because it works well with time-series data like enrollment numbers that change every semester and every year. It is also related because, like the proponent’s study, it focuses on how to make predictions more accurate and useful for schools. The papers reviewed show that LSTM is better than old methods like ARIMA, especially when the data has sudden changes or missing records — problems that can also happen with the enrollment data in ACLC College of Butuan.

Some studies also explain that LSTM can be combined with other tools, like CNN or attention mechanisms, to improve the predictions. This idea is useful for my study because I might need to improve my model later if the data is too complicated. In short, this study supports my research by showing that using LSTM for predicting enrollment is a good and effective way to help schools plan ahead and manage resources better, which is the main purpose of this thesis.

### 2.1.3 LSTM Forecasting

In recent years, forecasting has become more advanced because of new computer models that can study and predict future events. One of the most popular models today is the Long Short-Term Memory (LSTM) model. LSTM is a type of deep learning model that is good at remembering information from the past and using it to predict what will happen next. Many researchers use LSTM for forecasting because it can handle data that changes over time, like student enrollments, weather, and stock prices.

A study by (Siami-Namini & Namin, 2018) tested different forecasting models, including traditional models like ARIMA and deep learning models like LSTM. The researchers found that LSTM models gave better and more accurate results, especially when the data had patterns that changed over time. The study showed that LSTM was good at learning from past data and making predictions about the future.

Another study by (Raut, 2024) used LSTM to predict stock market prices. The researcher compared the LSTM model with other forecasting methods. The results showed that LSTM models gave lower errors and more accurate forecasts. This study proved that LSTM could be used not only in finance but also in other fields where data changes every day, like education.

A research by (Abbasimehr & Paki, 2022) also explained how LSTM models can improve predictions when combined with attention models. They tested this technique on time series data and found that it gave more accurate results than using LSTM alone. The study suggested that combining LSTM with other models can help in getting better forecasts, especially when the data has sudden changes. These studies show that LSTM is a powerful tool for forecasting. It is better than traditional models when dealing with data that changes over time. Because of its ability to remember past data, LSTM can help predict future events more accurately. This makes it very useful in education, business, and other areas where people need to make decisions based on future predictions.

This review is important to the proponent’s study because it highlights how LSTM models are effective in forecasting data that changes over time. The proponent’s study aims to predict student enrollment numbers at ACLC College of Butuan, which naturally vary each semester and year. The reviewed studies show that LSTM models perform better than traditional forecasting methods, especially when the data has patterns that shift over time.

These studies also prove that LSTM can produce more accurate predictions with lower errors in different fields like finance, education, and time series forecasting. This supports the use of LSTM in my research as a reliable tool for predicting future student enrollments. Additionally, findings suggest that LSTM models can be combined with other techniques, such as attention mechanisms, to further improve accuracy when the data has sudden or unexpected changes. Overall, the related literature confirms that LSTM is a powerful and flexible model for forecasting, making it suitable for my study, where accurate predictions are needed to support decision-making in school enrollment planning.

### 2.1.4 Time Series Forecasting Using Prophet

Apart from the machine learning based methods, Facebook's Prophet model has also been used to forecast student enrollments (Patayon & Crisostomo, 2022). This model accommodates and deals with seasonal changes and shifts in the trend, thus it is appropriate for educational enrollment data, which tend to have both prolonged shifts and seasonalities focused around semesters.

(Patayon & Crisostomo, 2022). used all of the student enrollment data that was available at a state university in Zamboanga del Norte between the years 2000 and 2022. Significant increases in enrollment from A.Y. were detected by the analysis. from 2013–2014 to 2015–2016 and from 2018–2019 to 2021–2022, in addition to the second semester's periodic declines in enrollment. RMSE and R2 were used to evaluate the forecasting models. Because the individual course datasets showed more accuracy than models constructed from aggregated data, the Prophet models of the individual courses (BSBA, BSA, and BSCrim) obtained the lowest RMSE and greatest R².

The efficiency of Prophet in identifying long-term and seasonal trends in enrollment data is demonstrated by this study. Its conclusions offer a standard by which to evaluate the effectiveness of forecasting models based on LSTM, especially with regard to accuracy, flexibility, and resource allocation. The models that best meet the forecasting requirements of educational institutions can be chosen by combining Prophet results with LSTM studies.

### 2.1.5 Time Series Forecasting using ARIMA

(Qin et al., 2019) conducted both simulation and empirical studies to evaluate how the **length of historical data affects the accuracy of ARIMA forecasting** in enrollment prediction. The simulation study, based on the average undergraduate enrollment across the top 10 HBCUs, showed that **short time series (5 years)** produced the **largest bias** between forecasted and true values. In contrast**, medium-length series (10–20 years)** provided significantly higher forecasting accuracy, and with the **20-year series achieving the most precise predictions**.

The practical study utilizing 35 years of undergraduate enrollment data from Howard University confirmed these findings. The prediction using the 20-year historical series revealed the least discrepancy (~370 students) from real enrollment figures, suggesting that ARIMA can successfully identify historical trends when adequate data is available.

The research also questioned the belief that extended time series consistently enhance forecasting precision. Including excessively long historical data (e.g., 30 years) did not improve precision and may increase residual errors due to higher autocorrelation in the ARIMA model.

These findings suggest that ARIMA is a **robust statistical tool** for enrollment forecasting, but its accuracy depends critically on the **appropriate length of historical data**, ideally around **20 years**. This insight provides a useful reference for enrollment planning, budgeting, and policy-making in higher education and serves as a **baseline for comparing ARIMA with machine learning-based models** such as LSTM.

(Parvez et al., 2021) conducted a study to forecast undergraduate enrollment at a U.S. higher education institution from 1999 to 2020. The study compared **ARIMA** with **Recurrent Neural Networks (RNNs)** to evaluate prediction accuracy. The dataset included 65 observations from the second quarter of 1999 to the second quarter of 2020. Since the enrollment data was non-stationary, differencing was applied, resulting in a stationary series suitable for ARIMA modeling. Using the Box-Jenkins methodology, the **ARIMA (3,1,3)** model was selected.

The ARIMA model produced an autoregressive coefficient of AR (3) = 1.00, which was highly significant (t-stat = 240.52, p < 0.001), and a moving average coefficient of MA (3) = -0.99, also highly significant (t-stat = -36,588.17, p < 0.001). The constant term was 16.50 but not statistically significant (t-stat = 0.005, p = 0.996), while the error variance (SigmaSQ) was 5,862,156. The model evaluation metrics indicated a good fit to the historical enrollment data, with an R-squared of 0.718 and an adjusted R-squared of 0.704, suggesting that the model explained a substantial portion of the variation. The standard error of regression was 2,499.31, while information criteria values were AIC = 18.73, SC = 18.86, and HQ = 18.78. The Durbin-Watson statistic of 2.57 indicated no significant autocorrelation in the residuals, and the overall F-statistic of 51.814 (p < 0.001) confirmed that the model was statistically significant in explaining the changes in enrollment over time.

These results indicate that the ARIMA model fit the historical enrollment data well, capturing the main trends and seasonality. Forecasts for 2021 using ARIMA were: **Spring – 7,869; Summer – 3,435; Fall – 7,898,** showing that the model performed better for **summer enrollment**, which had less variation compared to spring and fall.

### 2.1.6 Hybrid ARIMA-LSTM Forecasting

(Mun et al., 2025) conducted a study on forecasting daily gold prices during the COVID-19 pandemic using ARIMA, LSTM, and a hybrid ARIMA–LSTM model. The dataset covered two years of gold price data, from January 2020 to December 2021, and was divided into training and testing sets. The researchers evaluated the models using Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). The results showed that the ARIMA model achieved an RMSE of 952.27 and a MAPE of 1.62%, while the LSTM model performed better with an RMSE of 560.71 and a very low MAPE of 0.01%. In any case, the hybrid ARIMA–LSTM model recorded the lowest RMSE at 204.01, but its MAPE reached 33.29%, indicating unstable forecasts with frequent overestimation and underestimation.

The study by (Jain et al., 2024) tested different forecasting models such as ARIMA, Prophet, LSTM, GRU, and hybrid approaches (ARIMA+ANN and ARIMA+LSTM) to predict COVID-19 cases. The proposed ARIMA–LSTM hybrid model was designed to combine the strengths of both statistical and deep learning techniques. ARIMA was used to capture the linear and seasonal trends, while LSTM modeled the nonlinear residual errors, and their results were combined to make final predictions.

The performance of the models was measured using three metrics: Mean Absolute Percentage Error (MAPE), Symmetric MAPE (SMAPE), and Median Absolute Percentage Error (MDAPE). The results showed that in India, ARIMA–LSTM achieved the lowest errors with MAPE **=** 2.46%, SMAPE = 2.47%, and MDAPE = 2.55%, outperforming GRU (MAPE = 2.81%) and ARIMA+ANN (MAPE = 2.66%). In Brazil, ARIMA–LSTM recorded the best results with MAPE = 0.96%, SMAPE = 0.96%, and MDAPE = 0.80%, better than both LSTM (MAPE = 7.72%) and ARIMA+ANN (MAPE = 1.64%). In Russia, ARIMA–LSTM again performed better, with MAPE **=** 2.89%, SMAPE = 2.92%, and MDAPE = 2.98%, lower than ARIMA+ANN (MAPE = 3.10%) and GRU (MAPE = 4.03%). In the United States, ARIMA–LSTM was also superior, reaching MAPE **=** 2.75%, SMAPE = 2.77%, and MDAPE = 2.85%, which was lower than GRU (MAPE = 3.24%) and ARIMA+ANN (MAPE = 5.05%).

Across all four countries, the ARIMA–LSTM model achieved the lowest MAPE values overall, showing that it was the most accurate forecasting method compared to both baseline models and other hybrid models.

### 2.1.7 LSTM vs Different model

(Shobayo et al., 2025)conducted a study comparingthe forecasting performance of three models. These models were Long Short-Term Memory (LSTM), Support Vector Regression (SVR), and another type of Recurrent Neural Network (RNN). The study used the NGX All-Share Index dataset to predict stock prices. The LSTM model showed the best results in this comparison. It achieved a very low Mean Squared Error (MSE) of 0.000343 and a Root Mean Squared Error (RMSE) of 0.0185. Its R-squared value was 0.993, which means it was very accurate in predicting stock prices. The RNN model had an MSE of 0.000388, RMSE of 0.0197, and an R-squared of 0.992, which was also good but slightly lower than LSTM. The SVR model performed the lowest, with an MSE of 0.00459, RMSE of 0.0214, and R-squared of 0.99. Based on these results, the researchers concluded that the LSTM model provided the highest accuracy for stock price prediction in this study.

A similar study is conducted by (Zhou, 2024) with a tittle "Enhancing Stock Price Prediction: A Comparative Study of KNN and LSTM Models for Apple Inc.", the study was carried out to compare the performance of two machine learning models in forecasting stock prices. These models are the K-Nearest Neighbors (KNN) and the Long Short-Term Memory (LSTM) neural network. The primary objective of this research was to determine which model is more accurate and reliable in predicting the stock prices of Apple Inc. The models were trained with historical Apple stock data, which was obtained and utilized for the study. The performance of the models was measured through significant evaluation parameters such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). These parameters assist in indicating how close the forecasted stock prices are to the actual prices.

The outcome indicated that the LSTM model provided more accurate and stable predictions compared to the KNN model. The LSTM model had lower error rates in all three metrics of evaluation, which indicates that it was more accurate in stock price prediction. The research clarified that LSTM performed well because it excels at recalling past patterns in data and processing time-series data such as stock prices. This research is significant since it demonstrates that LSTM is not only beneficial in applications such as education forecasting but also performs well in financial forecasting. The results validate the concept that LSTM models are good predictors of data that evolves over time, such as stock prices.

The study of (Shobayo et al., 2025)is important to this research because it showed that the LSTM model is better than other models like SVR and RNN in predicting stock prices. Stock prices, like student enrollment, also change over time. This means if LSTM works well for stock price prediction, it can also work well for predicting how many students will enroll in the school. Also, the study conducted by (Zhou, 2024) is also important because it compared LSTM and KNN models in predicting stock prices for Apple Inc. The result showed that LSTM gave more accurate predictions than KNN. This is helpful for this study because enrollment numbers also follow patterns through time. If LSTM can predict stock prices accurately, it can also be used to predict future enrollment in ACLC College of Butuan.

### 2.1.7 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) used for forecasting and time series prediction. It was introduced by (Hochreiter & Schmidhuber, 1997) to solve the vanishing gradient problem found in traditional RNNs. This problem happens when the network forgets old data as it processes new data, making it hard to learn patterns in long sequences. LSTM was designed with a special memory cell and different gates that help the model remember important information for a longer time while removing unnecessary data.

The LSTM model uses three main gates: the input gate, forget gate, and output gate. These gates control the flow of information inside the memory cell. The input gate decides what new information will be stored, while the forget gate removes old or unimportant data. The output gate controls what information will be sent out of the memory cell. This process allows LSTM to learn and remember patterns in sequential data, making it very useful for tasks like weather forecasting, speech recognition, stock market prediction, and student enrollment forecasting.

Several studies have shown the effectiveness of LSTM in different forecasting problem (Siami-Namini et al., 2019a) proved that LSTM performs better than traditional models like ARIMA in predicting financial data. Another study by (Abbasimehr & Paki, 2022) combined LSTM with attention mechanisms to improve prediction accuracy in complex time series data. In the field of education, (James & Weese, 2022) successfully used LSTM to predict student enrollment numbers, showing that the model can accurately forecast future trends based on past records. These studies show how powerful and reliable LSTM is for forecasting and prediction tasks.

## 2.2 Theoretical Background

Forecasting enrollment trends requires models that can capture both **linear** and **nonlinear** patterns, as well as seasonal effects that occur across semesters. This section presents the theoretical foundations of the models employed in this study: **Autoregressive Integrated Moving Average (ARIMA), Long Short-Term Memory (LSTM), Hybrid ARIMA–LSTM, and Prophet.**

### 2.2.1 Autoregressive Integrated Moving Average (ARIMA)

The ARIMA model is a traditional time series forecasting method designed to capture **linear dependencies** in sequential data. It combines three components *p, d, q.*

*Equation 1: ARIMA model components*

In the ARIMA model, three key parameters define its structure: ***p*, *d*, and *q* [1]**. The parameter ***p*** refers to the order of the autoregressive (AR) part, which determines how many past values of the series are used to predict the current value. The parameter ***d*** represents the order of differencing needed to make the time series stationary, meaning that the statistical properties such as mean and variance remain constant over time. Finally, ***q*** is the order of the moving average (MA) part, which incorporates the dependency between the forecasted value and past forecast errors.

*Equation 2: ARIMA equation*

In the ARIMA model [2], , which is influenced by both past values and past errors. The parameter , capturing how strongly previous enrollments affect the current value. Meanwhile, , which adjust the forecast based on past error terms. Finally, , accounting for unexpected variations.

### 2.2.2 Long Short-Term Memory (LSTM)

LSTM, a type of recurrent neural network (RNN), is capable of learning **nonlinear, long-term dependencies** in sequential data. It uses gating mechanisms to control information flow.

#### **2.2.2.1 Forget Gate**

*Equation 3: Forget gate.*

The forget gate, is responsible for deciding which information from the past should be discarded from the memory cell. It takes two inputs, the previous hidden state and the current input , multiplies them by the weight matrix , adds the bias and passes the result through a sigmoid activation function . Because the sigmoid function outputs values between 0 and 1, the forget gate produces a “filtering factor” for each piece of information in the memory cell. If , the LSTM forgets that information completely. If , the LSTM retains it entirely.

#### **2.2.2.2 Input Gate**

*Equation 4: Input Gate*

The **input gate** decides **what new information will be added** to the LSTM’s memory cell at the current time step. It plays a crucial role in **learning new patterns** from the incoming data. Considering two main inputs, the previous hidden state , which carries information from past sequences, and the current input , which represents the new data at the present time. These inputs are first multiplied by a weight matrix and adjusted with a bias term to refine the signal. The result is then passed through a sigmoid activation function ,which outputs values between 0 and 1. When the output of the gate is close to 0, the LSTM ignores the new information, treating it as irrelevant. However, when the output is close to 1, the LSTM fully accepts the new information and stores it in its memory.

#### **2.2.2.3 Candidate Cell**

*Equation 5: Candidate Cell*

The **candidate memory cell** is responsible for generating new potential content that could be added to the LSTM’s memory. It takes the previous hidden state and the current input processes them through a weight matrix and a bias term , and then applies the hyperbolic tangent activation function ). The result is a set of candidate values that range between -1 and 1, representing possible information that may be incorporated into the memory cell. This step ensures that the model has new information ready, but whether it is stored or not depends on the input gate’s control.

#### **2.2.2.4 Cell State Update**

*Equation 6: Cell State Update*

The **cell state update** [6] (​) is the heart of LSTM memory management. It combines both the past and the present information to maintain long-term knowledge. Specifically, the forget gate decides how much of the old memory should be retained, while the input gate decides how much of the newly generated candidate memory should be added. By blending these two sources of information, the LSTM updates its cell , ensuring that relevant knowledge is preserved across time while also allowing the model to learn from new data.

#### **2.2.2.5 Output Gate**

*Equation 7; Output Gate*

The output gate (​) regulates how much information from the memory cell is exposed to the hidden state at the current time step. It works by first combining the previous hidden state (​) and the current input (​), then applying a weighted transformation through the weight matrix (​) and adding a bias term (​). The result is passed through a sigmoid activation function (), which scales the values between 0 and 1. When the output is closer to 1, more information is allowed to pass through, while values closer to 0 restrict the flow. The output gate acts as a filter, determining which parts of the stored memory are relevant and should be revealed for forecasting tasks such as predicting student enrollment trends.

#### **2.2.2.6 Hidden State**

*Equation 8; Hidden State*

The **hidden state** (​) represents the actual output of the LSTM at the current time step. It is computed by applying the tanh to the updated cell state (), which squashes the values into a manageable range between -1 and 1. The result is then multiplied by the output gate’s signal , effectively filtering which parts of the memory are released. This hidden state serves both as the prediction at time () and as the input for the next step in the sequence, making it crucial for forecasting tasks such as predicting future enrollment patterns.

### **2.2.3 Prophet Model Prophet**

Is an additive time series model developed by Facebook, designed to capture trend, seasonality, and holiday effects.

*Equation 9; Prophet model*

The Prophet model decomposes the enrollment time series into four key components, trend, seasonality, holiday/event effects, and error. The trend component, captures the long-term direction of student enrollment, whether it is steadily increasing, decreasing, or leveling off. The seasonal component, ,represents regular, repeating patterns such as the fluctuations in enrollment during June and December semesters. The holiday or event component , accounts for irregular but significant influences such as academic breaks, national holidays, or government policy changes that affect enrollment behavior. Finally, the error term models, random variations that cannot be explained by the other components, typically assumed to follow a normal distribution with mean zero and variance ().

#### **2.2.3.1 Trend**

Trend component, Prophet allows different forms of growth.

*Equation 10; Trend Component*

where is the growth rate (slope), and is the offset or baseline level. This assumes a constant rate of increase or decrease in enrollment over time. When comes the point that the growth is expected to saturate, Prophet uses a logistic growth model.

*Equation 11; Logistic growth*

where represents the carrying capacity () controls the rate of growth, and () shifts the curve along the time axis. This logistic form is particularly useful in academic institutions where enrollment cannot grow indefinitely due to physical or policy constraints.

#### **2.2.3.2 Seasonality**

Seasonality models repeating patterns across a fixed period, such as semester-based cycles.

*Equation 12; Seasonality Component*

Where is, designed to capture repeating or periodic patterns within the data, such as semester-based enrollment cycles. The summation symbol indicates that multiple sine and cosine terms are combined to model different levels of seasonal variation. The coefficients and control the strength and shape of these seasonal effects, while the functions and allow the model to capture cyclical behavior that rises and falls over time. The parameter (*P*) represents the length of the seasonal period, and (t) refers to the specific point in time. By combining these terms, the model can flexibly approximate both simple and complex seasonal patterns. In the context of enrollment forecasting, this allows the system to account for recurring fluctuations during academic cycles.

**2.2.3.3 Holiday/Event**

*Equation 13; Holiday/Event Component*

This equation models the effects of special events, holidays, or policy changes on the forecast. The summation symbol indicates that multiple events, indexed by (j), may influence the enrollment at time (t). Each is an indicator function that equals 1 if event (j) occurs at time (t), and 0 otherwise. The parameter (\kappa\_j) measures the magnitude of the impact of each event, either increasing or decreasing enrollment depending on whether the effect is positive or negative.

#### **2.2.3.4 Error**

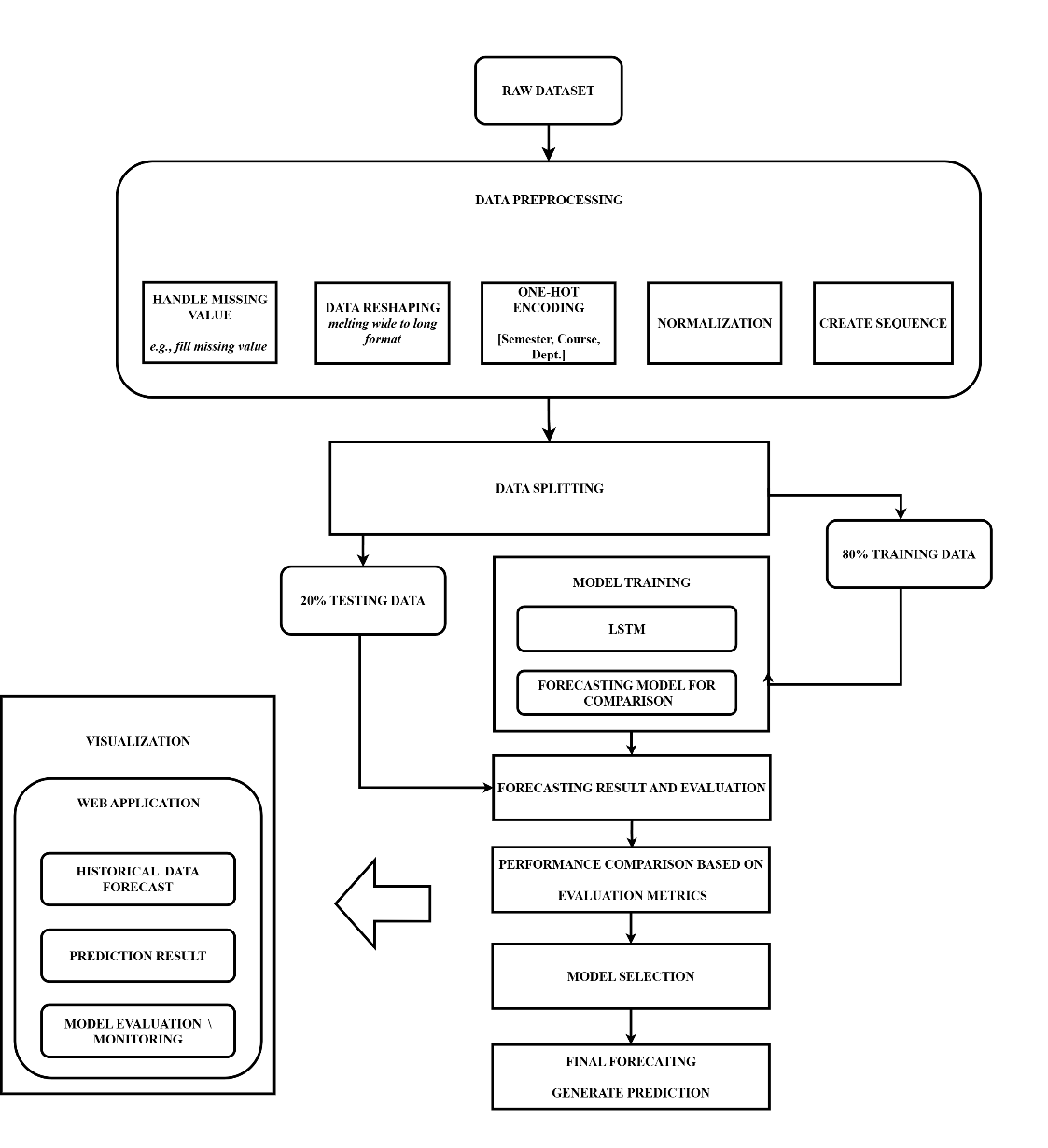
*Equation 14; Error Component*

Error term represents the unpredictable noise or variability in the enrollment data. where the mean is zero and is the variance. This component ensures that Prophet does not overfit the data by trying to explain every irregularity. Instead, it acknowledges that some variations in enrollment are due to random, unforeseen factors such as sudden student decisions, external events, or reporting inconsistencies.

# **CHAPTER 3**

**METHODOLOGY**

This chapter contains methods, discussions, tools and illustrations that will be used in the development of the proposed study.

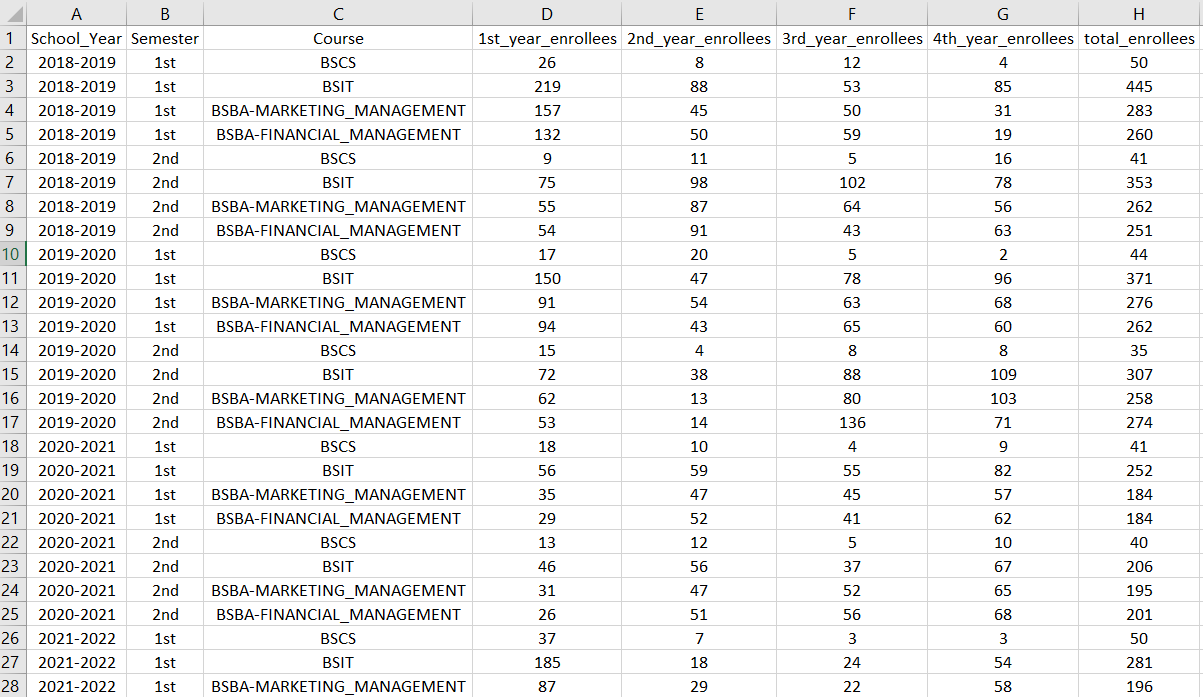
****

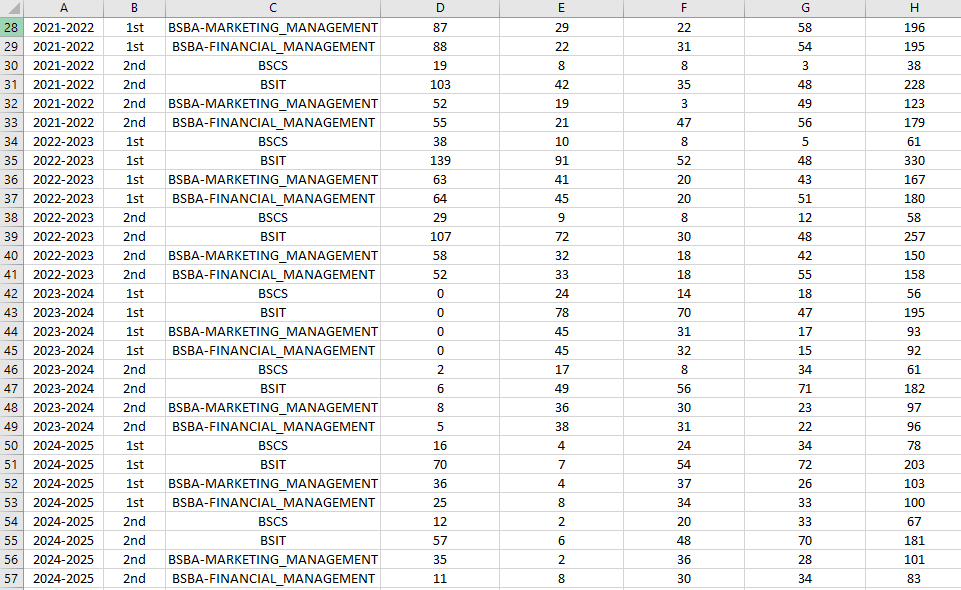
*Figure 1; Theoretical framework*

Figure 1 shows our general approach for the study. The specific steps are as follows:

## 3.1 Datasets

### 3.1.1 Actual data format

**** *Figure 2; Actual Data table*

*  
 Figure 3; Actual Data table (2)*

### 3.1.2 Data Collection

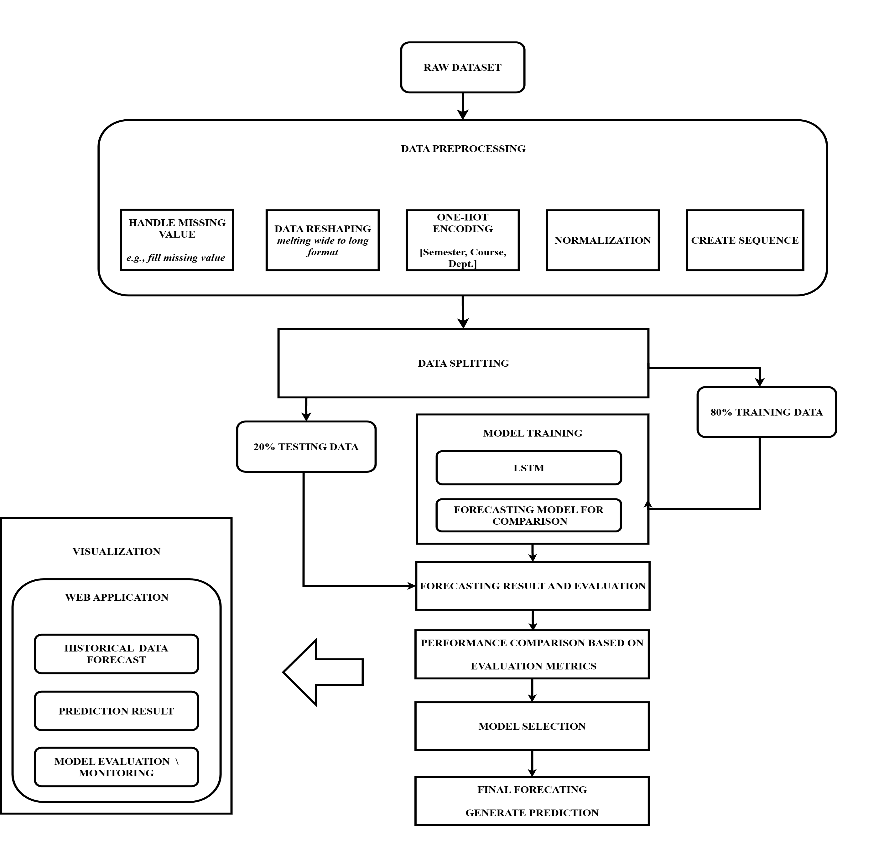
This study fucoses exclusively on data collected from ACLC College of Butuan, covered the span of school year 2018 – 2025 to provide a comprehensive analysis of enrollment trends and patterns. The primary objective of this data collection is to gather significant data for analysis on forecasting of enrollees and provide insights of possible numbers of enrollees on a specified school year and semester. By understanding these patterns, the study aims to provide visualization that can help improve management and strategies, such as resource allocation, marketing approach, and faculty hiring.

The enrollment records which are updated per semester, will be systematically gather for a period of 7 years specifically school year 2018 – 2025, In compliance of data privacy regulations, we will not include personal information on the datasets to provide identity protection of the involved individuals. The data will be provided by the school registrar of ACLC College of Butuan solely for research purposes only, with the assurance of, that the gathered data will serve only its purpose by which to help the institution without compromising privacy and security.

The collected dataset forms the foundation of this research. It will be used to analyzed trends to forecast enrollees and will serve as input for predictive modeling. The dataset includes records of numbers of enrollees on every schoolyear. The key details in the dataset include number of enrolled students, date of enrollment, semester (1st semester or 2nd semester), course (IT, Business, or ComSci.), and department. The data is stored in excel format for further processing.

## 3.2 Methods

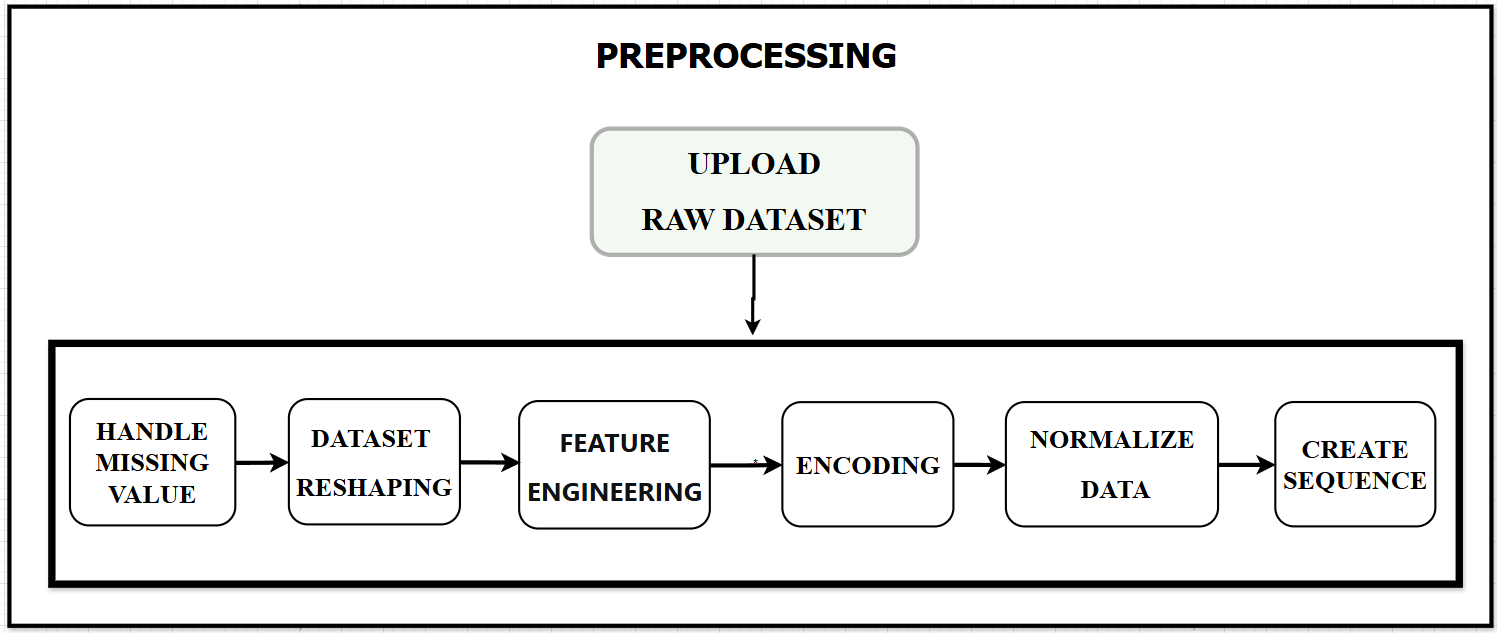
### 3.2.1 Experiment design

****

*Figure 4; Experiment design*

In this shown figure, illustrated the flow of enrollment forecasting for prediction using Long Short-Term Memory forecasting model. The dataset will be collected, prepared and split into 70% for training, and 20% for testing. The LSTM model will be trained using the training data. After training, the model will predict future values, the predictions will be compared with the actual testing data, and the result will be evaluated using accuracy measurements. Lastly, the results will be shown through a simple web application for visualization. This study employed a **quantitative research design** using a **time series forecasting approach**. Specifically, the proponents utilized machine learning techniques, particularly to forecast student enrollment trends across different semesters and school years.

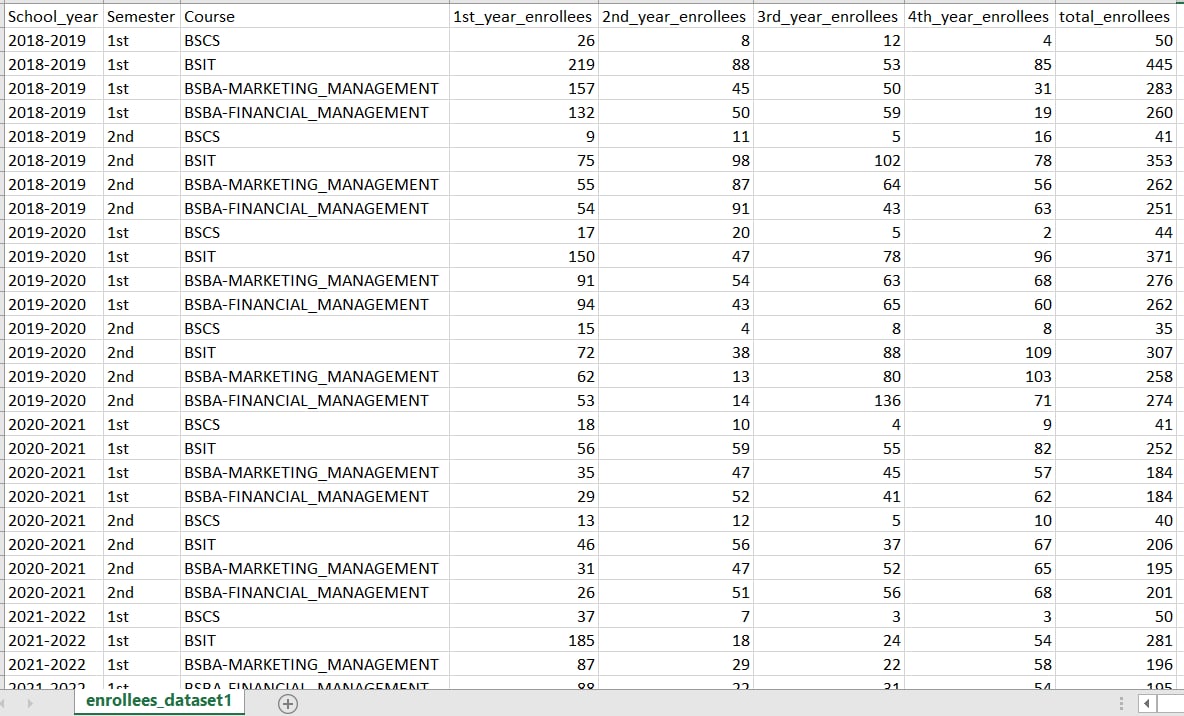
### **3.2.2 Data preprocessing**



*Figure 5; Representation of data preprocessing*

Before training the model, the dataset has undergone additional preprocessing steps for accuracy and consistency. It removed missing values so that missing data will not influence the learning, and managed outliers so as to mitigate their impact on model predictions. Feature engineering created new features based on generated meaning, while categorical variables were cast into numerical 0/1 form through one-hot encoding. All the numerical data were normalized onto a similar scale, so the model could learn effectively and make accurate predictions.

#### **Raw datasets**



*Figure 6; Actual datasets gathered*

The dataset for this study consists of historical enrollment records from ACLC Butuan, covering the period from 2018 to 2025. It includes key variables such as Date, Number of Enrollees, Semester, and School Year. The dataset contains 9514 entries and serves as the foundation for enrollment forecasting.

#### **3.2.2.1 Upload Raw Dataset**



*Figure 7; Loaded the dataset*

The data was loaded from a CSV file into a Pandas DataFrame, uncovering a structure consisting of 56 rows and 8 columns. The first analysis, as seen in Figure X, gave a general overview of the composition of the dataset, i.e., its shape, data type distribution, there being no missing values, and total enrollment counts distribution.

#### **3.2.2.2 Handle missing value**

Handling missing values is the process by which to handle missing values on the dataset, in this step the system will fill or remove the missing values.

##### **Fill missing values**

*Equation 15; fill missing vale equation*

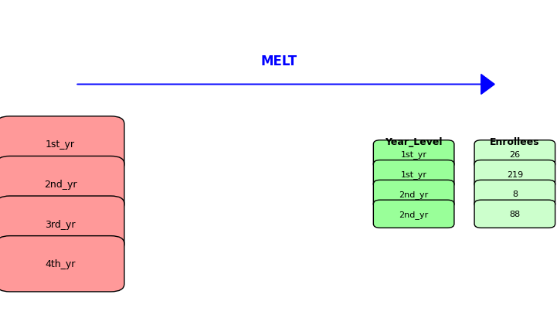
Where ​ is the value used to replace the missing data, () is the number of available data points, and ()​ are the known values. This process makes sure the missing values are filled properly, keeping the dataset complete and balanced.

#### **3.2.2.3 Data reshaping**

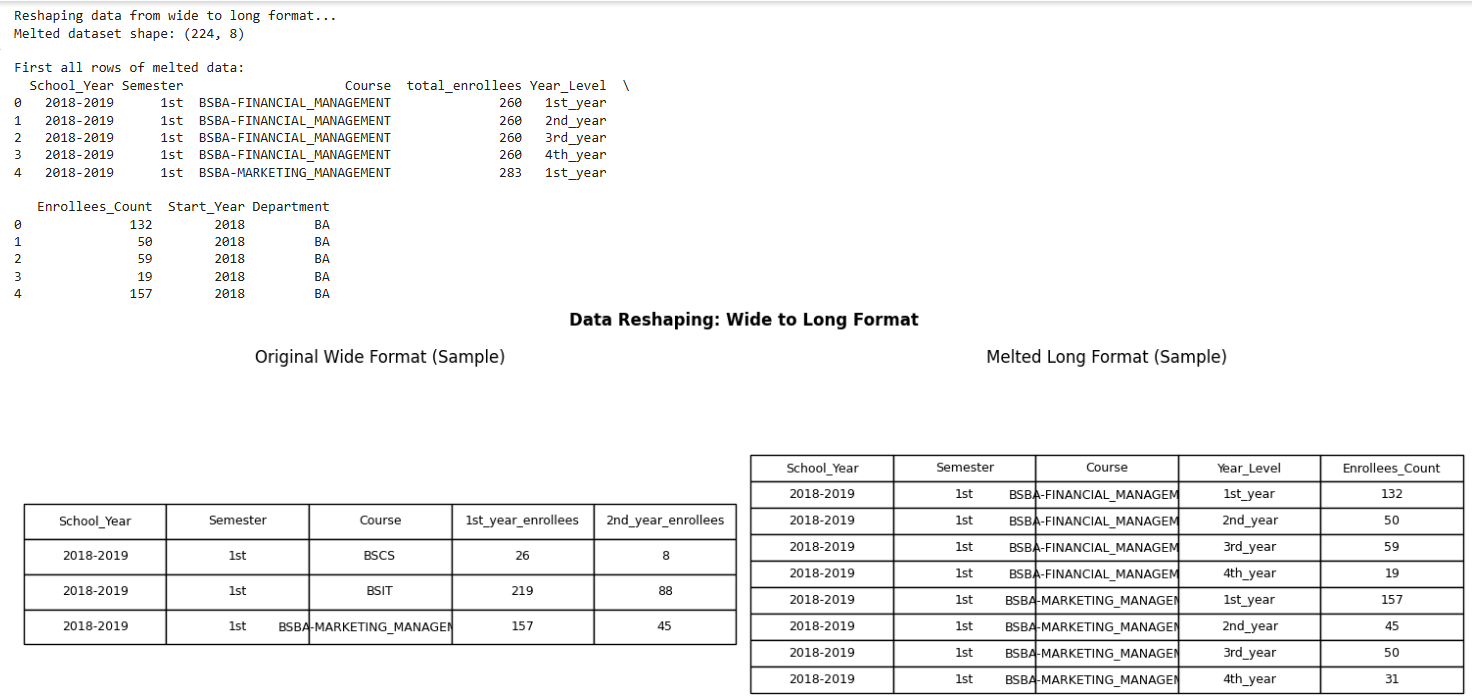
  
  
 *Figure 8; Data reshaping*

The data reshaping procedure used in this study is shown in Figure 3. A long-format dataset with a single Year\_Level column and a matching Enrollees\_Count column is created by reshaping the original wide-format dataset, which had distinct columns for each year level. By doing this, the dataset structure is made simpler and the model is able to read sequential time steps for forecasting. The dataset is sorted chronologically to maintain the proper order of enrollment records, and additional features like Department, Year\_Semester, and Start\_Year are developed to provide categorical and temporal information. Data reshaping guarantees that the dataset is organized so that LSTM models can effectively discover temporal patterns and make precise predictions about future enrollments.

##### **Melt datasets**



*Figure 9; Melting process*

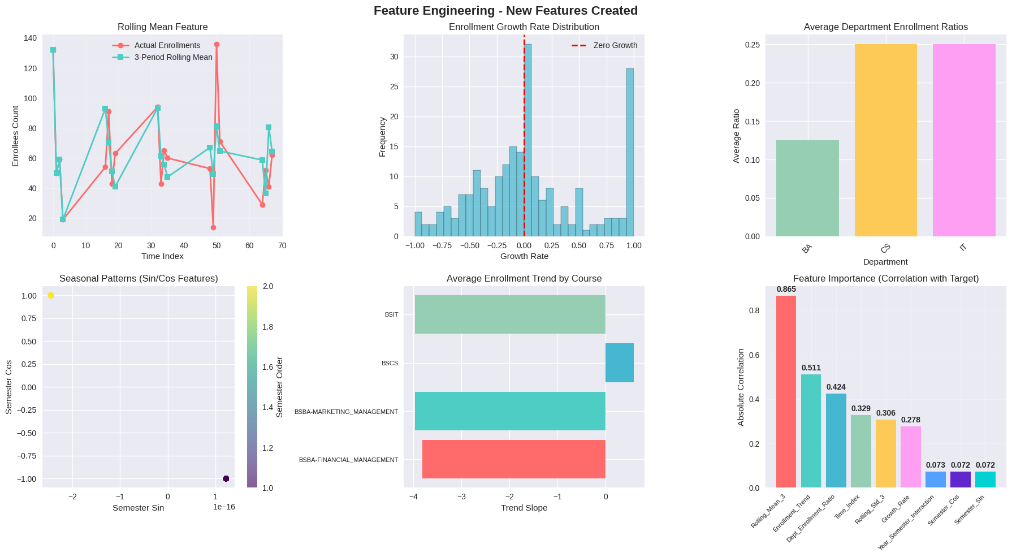
**

*Figure 10; Before and After melting long format*

The original wide format data was then converted to a long format through Pandas' melt() method. This was the key step that broke down the aggregated year-level columns (1st\_year\_enrollees, 2nd\_year\_enrollees, etc.) into one Year\_Level column and its corresponding Enrollees\_Count column. This operation enhanced the number of observable sequences from 56 to 224, and this gave a more detailed observation of enrollment per course, semester, and year level, which is vital for time-series analysis.

#### **3.2.2.4 Feature Engineering**

To enhance the predictive capability of the model, a list of new features was designed from the given data. This involves generating extra input signals that help the LSTM model pick up on sophisticated temporal patterns and relationships not evident in the raw data itself.



*Figure 11; Engineered feature*

The following features were built systematically:

##### **Temporal Features:**

* Year\_Semester: An identifier generated by combining the School\_Year and Semester fields (e.g., "2023-1").
* Start\_Year: The initial year was pulled from the School\_Year string and parsed as an integer (2023, for example).
* Semester\_Order: Categorical semester fields ('1st', '2nd') were remapped to ordinal integers (1, 2) to supply a numeric value.

##### **Lag Feature:**

* Lag\_1\_Enrollees: Previous semester enrollment figure for the same course and year level. This offers the model the latest historical value, with a solid basis for forecasting the next time step.

##### **Rolling Statistics:**

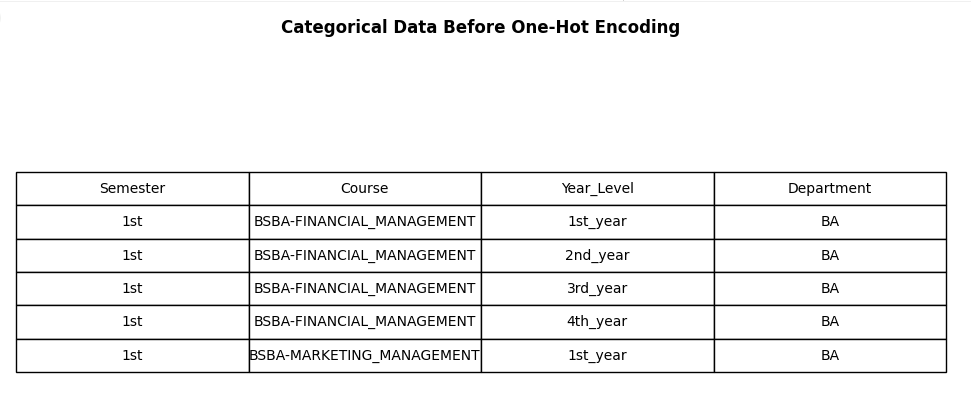
* Rolling\_Mean\_3: Simple moving average of enrollment over the last three semesters. This reduces short-term spikes and emphasizes underlying trends.
* Rolling\_Std\_3: Rolling standard deviation of enrollment for the same window. This is a measure of volatility or stability of enrollment counts.

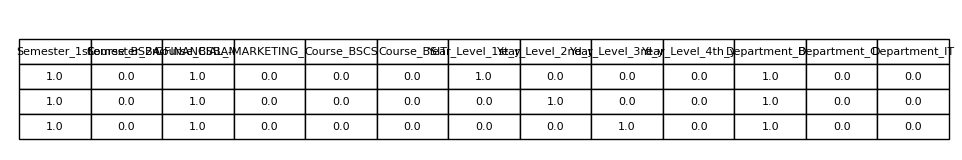
1. **Growth Rate:**

* Growth\_Rate: Percent growth in enrollment between the last semester and the current semester.
* Semester\_Sin and Semester\_Cos: Semester\_Order was converted to two features with sine and cosine functions being used to encode its cyclical nature correctly.

This encoding avoids the model incorrectly viewing the semester as a linearly incrementing value and instead preserves its constant, repeating pattern. The generation of these features is an important part of the preprocessing data pipeline, ensuring the model sees a dense and informative dataset during training.

#### **3.2.2.5 Data Encoding**

*Figure 11; Categorical Data Before One-hot Encoding*

******

*Figure 12; Categorical Data After One-hot Encoding*

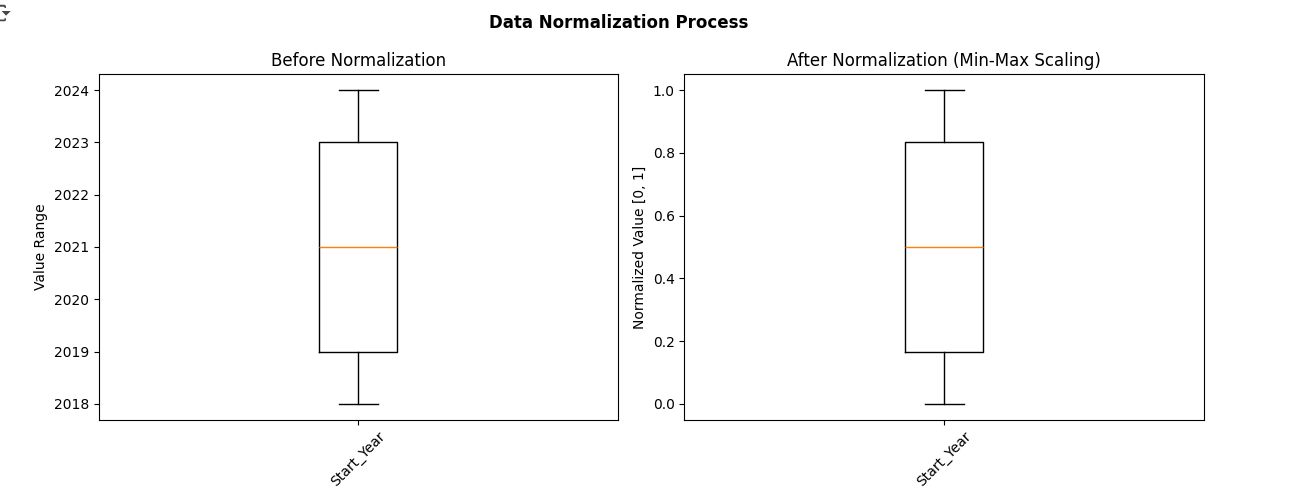
The categorical variables (Semester, Course, Year\_Level, Department) were then encoded in a numerical form using one-hot encoding through Scikit-learn's OneHotEncoder. This process built binary columns per category so that the model could handle these features without any assumption of ordinal relationship.

#### **3.2.2.6 Normalize Data**

All numeric features were normalized via Scikit-learn's StandardScaler, which scales data to have a mean of 0 and standard deviation of 1. This keeps no one feature overwhelming the learning process of the model with its size, important for good functioning of the LSTM model.

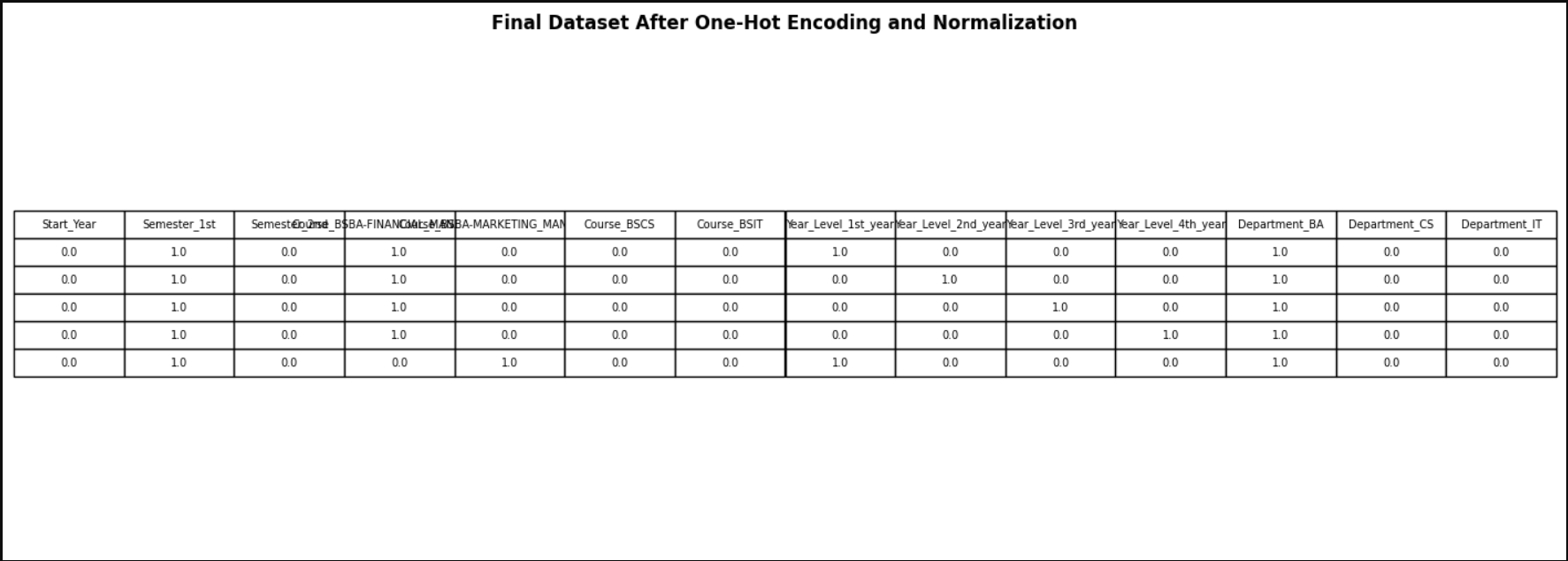
*Equation 16;**Data normalization equation*

The shown equation above is the representation of Normalize Data, where the is normalized value, and is original value which will be subtracted by the minimum value in the column . The maximum value () is subtracted to the minimum value, the difference from the minimum (numerator) and range (denominator) is then divided to get the Normalize data ().



*Figure 13;**Before and After Min-Max scaling*

The Start\_Year feature's values before and after normalization are displayed in the boxplot. Large values may dominate smaller ones, which could have an impact on the LSTM model's learning process. Prior to normalization, the values in their original scale ranged from 2018 to 2024. The values were rescaled between 0 and 1 after Min-Max Scaling was applied, making 2018 0.0, 2024 1.0, and the years in between proportionately adjusted (e.g., 2021 ≈ 0.5). By ensuring that every feature is on the same scale, this procedure improves the model's learning capabilities.

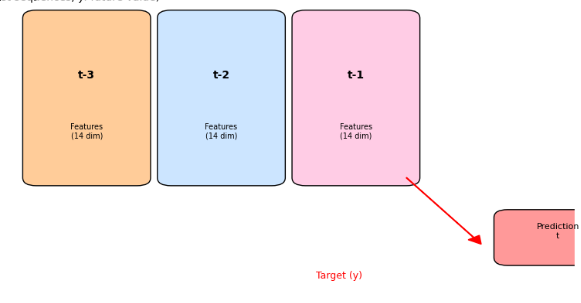


*Figure 14; after One-hot encoding and Normalize data*LSTM is sensitive to the scale of input data. Therefore, numerical features are scaled to a range between **0 and 1** using **Min-Max Scaling.**

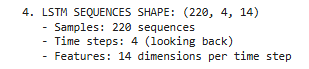
#### **3.2.2.7 Sequencing**

The normalized time-series data was transformed into a supervised learning structure suitable for the LSTM model. A function was used to produce input-output pairs in which each input sequence is composed of data from a specified number of previous time steps (TIME\_STEPS), and the output accordingly is the value to be predicted at the next time step. This produced the final 3D array of shape (samples, time\_steps, features) suitable for model input.

The preprocessed data and the fitted preprocessors (encoder, scalers) were stored to disk through Joblib to maintain training and deployment consistency.



*Figure 15; Creating a sequence for LSTM*

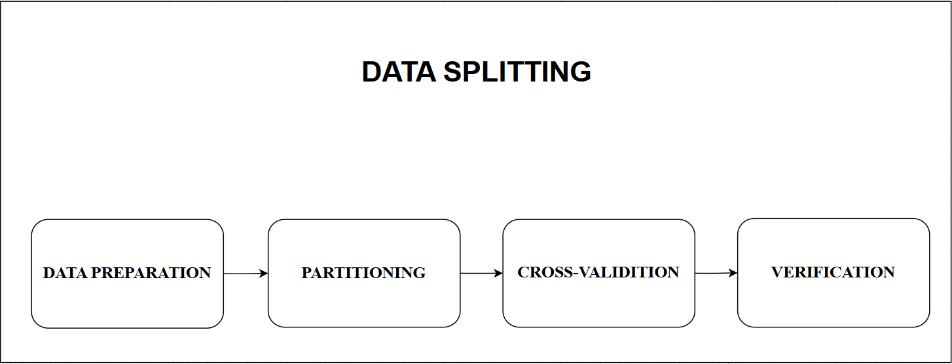
**

*Figure 16; Sequence for LSTM*

*Equation 14; Sequence creation equation*

The long-format enrollment dataset was organized into sequences appropriate for learning temporal patterns prior to training the Long Short-Term Memory (LSTM) model. The model used enrollment data from the preceding four periods (e.g., semesters) to predict the enrollment in the subsequent period because each sequence had four past time steps. In order to represent various aspects of the dataset, including school year, semester, course information, year level, department, enrollment count, and additional engineered features like Year\_Semester and Start\_Year, fourteen features were included at each time step. Formally, if represents the feature vector at time step then a single sequence can be expressed as . By sliding this sequence window across the entire dataset, **220 sequences** were generated, each serving as an independent sample for the LSTM model. Consequently, the final input shape of the LSTM was defined as where 220 represents the total sequences, 4 indicates the number of past time steps considered, and 14 denotes the features at each time step.

### 3.2.3 Data Splitting

****

*Figure 17;**Data splitting*

The data was split with a hold-out approach in order to maintain the temporal order of the data. The last 20% of the sequential data was reserved as an ultimate test set, and it was employed once only for the final assessment of the model's generalization performance. The other 80% of data was considered as the development set. To guarantee a strong and unbiased selection of models, Nested Cross-Validation was utilized here in this development set. The inner loop of the process used TimeSeriesSplit cross-validation for hyperparameter selection, and the outer loop generated the data to train the final model from after tuning.

###### **Data Preparation**

In this step, the dataset is cleaned to remove any missing values or duplicate entries. Time-related features, such as school year or semester, are also changed into a format that the model can understand. This makes the dataset clean and ready for the next steps.

###### **Partitioning**

For guaranteeing a strict testing of the model's performance and validity, the data was divided by a hold-out technique with a strict preservation of temporal order. The data was divided into two separated subsets. Training Set (80%) was utilized for the learning process of the model, including parameter estimation and tuning of hyperparameters using cross-validation and Test Set (20%) of completely retained subset in the development period. It was utilized once, in the final testing, to give a fair evaluation of how the model would perform on unseen future data.

Importantly, to avoid data leakage and achieve an authentic imitation of forecasting, the division was done chronologically instead of being random. The oldest 80% of data was assigned to the training set, while the latest 20% was held out for testing.

Rather than using a fixed validation split, a TimeSeriesSplit cross-validation method was used in the training dataset. This technique generates several train-validation folds in a time-ordered way such that the model is never validated on data that occurs chronologically before its training data. This is utilized for hyperparameter tuning, overfitting detection, and giving a stable internal performance estimate during the model building stage.

###### **TimeSeries Split Cross-Validation**

In TimeSeriesSplit, splitting the dataset 𝐷 in to 𝑁 samples into folds are performed in a sequential manner. At split, for () the training and testing sets may be written as equation [15].

***Equation 15; Equation for Timeseries Split Cross-Validation***

This method was applied to further evaluate the model. It allowed the training set to be divided into smaller folds, helping in fine-tuning the model, monitoring its performance, and reducing the risk of overfitting. **To interpret the equation, () are the data points utilized for the training in the( ) split and () is/are data points utilized for the testing purposes in the ()split. refers to the number of samples included in the training set up to that split. () is pertaining to the number of samples used in the test set for the same split. When using three folds of cross-validation, the amount of training data points grows as subsequent train folds become larger (), while the testing set always follows immediately after the training set, thereby ensuring the preservation of the time order of the data.**

###### **Verification**

After splitting, the sizes of the training and testing sets were verified to make sure they contained 80% and 20% of the data, respectively, without any overlap.

*Equation 16; Verify split data Avoid overlapping*

*Equation 17; Avoid overlapping*

The equation represents the total number of splits, where is equal to 80% and is equal to 20% of the total number of samples (). For clarification, if the total number of samples is , then the process will be: .This shows the proportion used in the train-test split. To avoid overlapping of data, the condition is applied. Here, the symbol “∩” (intersection) ensures that the training and testing sets do not share the same records. If an intersection is detected, meaning that the training and testing sets contain identical data, it will be considered overlapping, which is not allowed in proper data splitting, this process helps verify that the data is ready for the LSTM model.

### 3.2.4 Model ****Architecture: Long Short-Term Memory (LSTM)****

#### **3**.2.4.1 **Long Short-Term Memory (LSTM) Architecture**

There are Long Short-Term Memory networks (LSTM) among the RNNs, which are tailored to learn temporal sequences and lengthy dependencies better than other RNNs. Such networks conquer the commonly known vanishing gradient issue, which occurs in simple RNNs by means of a unique cell state structure and three mechanisms known as "gates." The following equations define the forward pass of an LSTM cell at every time step .

**Core Components;**

The LSTM cell state, denoted as acts as the network's "memory," transporting relevant information through the sequence.

###### **Forget gate ()**

The Forget Gate decides what information from the past should be removed. It checks the previous output and the current input and uses a sigmoid function to produce a value between 0 and 1. A value close to 0 means “forget this,” and a value close to 1 means “keep this.”

*Equation 17*;*Forget gate equation*

where:

* = sigmoid function
* = weight for forget gate
* = Previous output
* = current input
* = bias for forget gate

###### **Input gate** ()

This gate determines what new information will be stored in the cell state. It consists of two components: a sigmoid layer that decides which values to update, and a layer that creates a vector of new candidate values.

*Equation 18;**Input gate equation (Gate)*

The actual "gate." It produces a vector of values between 0 and 1, It looks at the current input () and the previous hidden state ( the sigmoid function (σ) squashes the result to a range between 0 and 1, each number in the  vector acts like a **switch**, that if a value is close to 1 it will **update the corresponding part of the cell state** with the new candidate value, however, if the value is close to 0, then it will **ignore the new candidate value** for this part of the cell state. Which that **decide on what to update, and where**  is the output of the input gate's sigmoid layer, that represents the weight for input gate, bias for input gate

**b.1 Candidate value ()**

Then, a candidate value is created for adding to the memory

*Equation 19; Input gate (Candidate value)*

Simultaneously, a separate, parallel layer creates the **candidate values**  that could be added to the cell state. This layer uses a hyperbolic tangent (tanh) activation function, which produces values between -1 and 1. The final update to the cell state is the element-wise product of the gate vector and the candidate vector ensuring only the filtered, relevant information is incorporated into the long-term memory.

###### **Cell state update ()**

After the Forget and Input gates, the cell state gets updated. It keeps useful information and removes what is no longer needed.

*Equation 20;**Cell state update**formula*

Where, () is new cell state, then old cell state. The old cell state is now updated to the new cell state  The previous state is multiplied by the forget gate  discarding the information deemed unnecessary. Then, we add the new candidate values  scaled by how much we decided to update each value .

###### **Output gate ()**

 Finally, the output gate decides what the next hidden state ​ will be. The hidden state contains information about previous inputs and is used for predictions. The cell state is first passed through a tanh function to push the values to be between -1 and 1. This is then multiplied by the output of the sigmoid gate  which filters the output based on the current input and previous hidden state.

*Equation 21;**Output gate equation*

The output gate value, denoted as is computed via a sigmoid activation function which applies a learned weight matrix and an additive bias term to the concatenated input vector thereby regulating the extent to which the current cell state contributes to the new hidden state output.

###### **Final output ()**

The final output is the result of the Output gate multiplied by the updated cell state passed through a tanh () activation function. This final value moves to the next step in the sequence

*Equation 22;**Final output equation*

The final hidden state output is computed as the element-wise product of the output gate vector and the hyperbolic tangent of the updated cell state , formally expressed as thereby filtering the long-term memory through the gate's regulatory mechanism to produce the short-term memory for the subsequent time step.

###### **Rationale for Use in This Study:**

The gated architecture offers great promise for using these models for trend forecasting because enrollment practices become time series data with complex long-term dependencies (e.g., how the popularity of a program has changed over the years) and short-term variations (e.g., semester-to-semester shifts). The ability of LSTMs to remember, forget, and output information will help capture even the most intricate characteristics of these temporal dynamics, such as seasonality trends and long-term trends that simpler models may be unable to capture. The model will be trained to connect historical enrollment sequence data (as described in Section 3.2.1.1.6) to future enrollment values.

### 3.2.5 Model Training

The model was created and optimized with a Nested Cross-Validation method to be robust and avoid overfitting. The development dataset prepared (80% of the data) was used for that purpose. The training procedure entails the repeated feeding of input sequences to the model, and in the process, its internal parameters (weights and biases) are updated to reduce the discrepancy between its predictions and true target values. The optimization is carried out utilizing the Adam algorithm (Kingma & Ba, 2014) to calculate adaptive learning rates, and the Mean Squared Error (MSE) as the loss function to minimize. Hyperparameter optimization was done through an inner loop of TimeSeriesSplit Cross-Validation (with n\_splits=5). This process forms several train-validation folds in a temporally ordered manner such that no future data is ever utilized to validate a model that was trained on previous data, thereby avoiding data leakage. A randomized search (RandomizedSearchCV) was executed across 20 iterations over a specified hyperparameter space, which contained:

Number of LSTM units: [32, 64, 128, 256]

Learning rate: [0.0001, 0.0005, 0.001, 0.005, 0.01]

Batch size: [16, 32, 64]

Dropout rate: [0.2, 0.3, 0.5]

Each combination of hyperparameters, the model was trained and tested on all TimeSeriesSplit folds. The hyperparameter combination that provided the best average performance (minimum Mean Absolute Error) over all folds was chosen. The training schedule of each model was set up with the following guiding principles:

#### **Epochs**

The total number of passes made over the training set. Early stopping was used with patience set to 15 epochs for stopping training if the validation loss on the given fold failed to reduce, avoiding overfitting and maximizing computational efficiency.

#### **Batch Size**

The batch size refers to the number of samples executed before the internal parameters of the model are updated.

The best-performing model was subsequently retrained on the full development set with the optimal hyperparameters found during this process. This model was tested on the held-out test set (20% of data) to give a final, unbiased measure of its performance.

### 3.2.6 ****Forecasting****

After the training and hyperparameter optimization process, the performance of the final model was strictly assessed on the held-out test set (20% of the data). The set included the latest, unseen temporal sequences, giving the model a realistic simulation of its forecasting capability with future data. The forecasting procedure entailed making multi-step predictions of the enrollment values in the test set using the trained LSTM model. The predictions from the model were compared to the actual, observed enrollment values.

These forecasts' predictive performance and accuracy were strictly measured using a collection of four standard regression metrics that provided a holistic assessment of the model's errors and goodness-of-fit.

### 3.2.7 Evaluation Metrics

In this study, four evaluation metrics are used: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE) and score. These metrics are commonly used in time series forecasting to measure the difference between the predicted values and the actual values. By using these evaluation tools, the study can clearly identify how close or far the predictions are from the real enrollment data. This is important to prove the usefulness and accuracy of the LSTM model for enrollment forecasting.

#### **Mean Absolute Error (MAE)**

MAE shows the average difference between the predicted values and the actual values.

*Equation 23; mathematical equation for MAE*

It adds all the absolute errors and takes the average. It tells us, on average, how many students the model was wrong by.

#### **Root Mean Squared Error (RMSE)**

RMSE shows the square root of the average squared differences between actual and predicted values.

*Equation 24; mathematical equation for RMSE*

It adds all the absolute errors and takes the average. It tells us, on average, how many students the model was wrong by.

#### **Mean Absolute Percentage Error (MAPE)**

MAPE shows the average error as a percentage of the actual value.

*Equation 25; mathematical equation for MAPE*

It tells us how far the predictions were, in percentage. It is easy to understand and compare across different models or datasets.

#### **R² score (coefficient of determination)**

How closely a model's predicted values match the actual data is indicated by the R2 score. It indicates the percentage of the dependent variable's variance that can be predicted based on the independent variables.

*Equation 26; Mathematical equation for R² score.*

For this equation, is the representation of the actual value, represents the predicted value, is the mean of the actual values, and is the total number of samples. The total number of samples is *n*. The range of the R2 value is -1 to 1. A perfect prediction, in which every predicted value precisely matches the actual values, is represented by a value of 1. A value of 0 indicates that the model's predictions are no more accurate than those made by merely taking the data mean. The model performs worse than using the mean, indicating poor predictive accuracy, if the R2 score is less than 0. Alongside other metrics like Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE), this metric is frequently used to assess regression models.

## ****System Implementation & Design****

### 3.3.1 Materials

#### **3.3.1.1 Software**

##### **Front-end Tools**

***Table 1;*** *Front-end Tools*

|  |  |
| --- | --- |
| **Tool** | **Description** |
| *Python 3.10* | *will be used as the primary programming language on development and model training* |
| *HTML5* | *Used to structure the content and layout of the web interface.* |
| *CSS3* | *Provides the styling and design for the web interface, including custom animations and transitions.* |
| *JavaScript* | *Handles dynamic content and interactions on the front-end, ensuring the interface is interactive and responsive.* |
| *Chart.js* | *A JavaScript library for creating simple and interactive charts. It will help in visualizing key metrics like predicted vs actual enrollment or trend overtime.* |
| *Plotly.js* | *A graphing library that allows you to create more advanced interactive charts and graphs mostly use in data visualizations* |
| *Bootstrap 5.3* | *A front-end framework that provides responsive design components, allowing the application be easily on different screen sizes* |
| *Jinja (Flask templating engine)* | *Injects variables such as forecast results, enrollment data, and chart labels into HTML templates ({{course}}, {{prediction}}), allowing each user to see personalized result.* |
| *Matplotlib* | *Produces server-side plots that are saved as images and displayed in the web interface* for *additional performance analysis.* |

##### **Back-end Tools**

***Table 2;*** *Back-end Tools*

|  |  |
| --- | --- |
| **Tool** | **Description** |
| *Pandas 2.0.3* | *for data manipulation and analysis* |
| *NumPy 1.24.3:* | *for numerical operations* |
| *Scikit-learn 1.3.0:* | *for machine learning implementation* |
| *Flask 2.3.3* | *for API development and web service integration* |
| *Joblib 1.2.0* | *for model serialization and persistence* |
| *Werkzeug 2.3.7* | *Flask’s underlying utility library (routing, debugging, WSGI).* |
| *Flask-WTF 1.1.1* | *helps build and validate forms easily.* |
| *python-dotenv 1.0.0:* | *loads environment variables from a .env file.* |
| *TensorFlow 2.13.0/Keras* | *deep learning framework intended for LSTM.* |
| ***Matplotlib 3.7.1 &*** *Seaborn 0.12.2* | *in tandem to generate high-quality static visualizations on the server for data analysis.* |
| *Plotly 5.15.0:* | *interactive charts for frontend/backend (your enrollment trend graphs).* |

#### **3.3.1.2 Hardware**

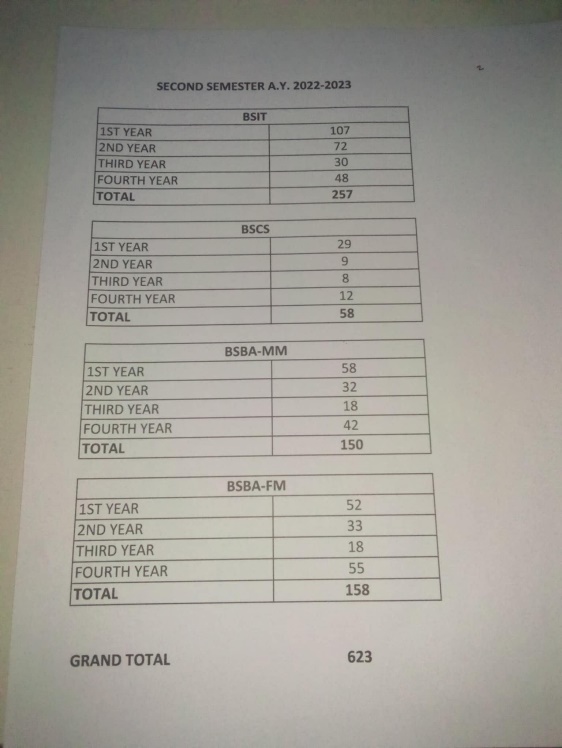
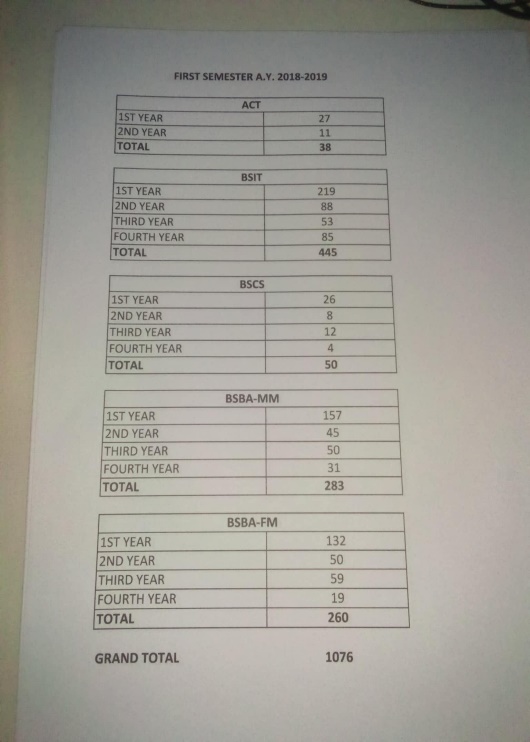
|  |  |
| --- | --- |
| Component | Specification |
| Processor (CPU) | Intel Core i5 (11th gen) |
| Memory (RAM) | 16 GB DDR4 |
| Graphics Card (GPU) | NVIDIA GeForce RTX 3050 (6GB VRAM) |
| Storage | 440 SSD |
| Operating System | Windows 11 |

***Table 3;*** *Hardware Tools*

# **CHAPTER 4**

**RESULT AND DISCUSSION**

This study aims to forecast student enrollment at ACLC College of Butuan using machine learning-based time series models. The collected enrollment records and model evaluations provided insights into the performance of different algorithms, with the Long Short-Term Memory (LSTM) model emerging as the most effective approach for generating accurate predictions and supporting decision-making through a web-based application.

**4.1 Data Collection**

*Figure 18; Actual dataset obtained from the Registrar*

The dataset consisted of historical enrollment records obtained from the registrar of ACLC College of Butuan, covering the period from School Year 2018–2019 (1st semester) up to 2025–2026 (2nd semester). These records included total enrollment counts categorized by course, department, year level, and semester. A total of 9,514 entries were collected, representing a comprehensive view of the institution’s enrollment patterns over seven academic years.

**4.1.1 Data Scraping**

Figure 18. Data Scraping

The raw data underwent extensive preprocessing to ensure quality and consistency before being used in model training. Missing values were handled by interpolation or removal, outliers were mitigated, and categorical features such as semester, course, and year level were encoded numerically. Numerical features were normalized to a scale between 0 and 1, making them suitable for the LSTM model. The final dataset was reshaped into a long format and sequenced to capture temporal dependencies.

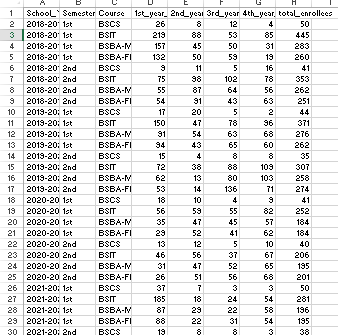
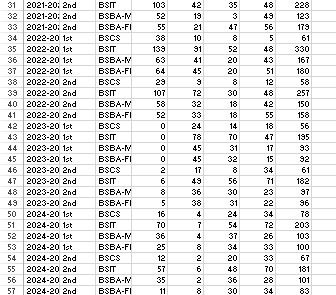
4.1.2 Raw Data

The raw data underwent extensive preprocessing to ensure quality and consistency before being used in model training. Missing values were handled by interpolation or removal, outliers were mitigated, and categorical features such as semester, course, and year level were encoded numerically. Numerical features were normalized to a scale between 0 and 1, making them suitable for the LSTM model. The final dataset was reshaped into a long format and sequenced to capture temporal dependencies.

Figure 19. Raw Dataset

The initial raw dataset contained enrollment counts from different courses and year levels across multiple semesters. However, some entries included incomplete or inconsistent formatting, such as missing semester identifiers, irregular spacing, and mismatched course codes. At this stage, the dataset was not yet standardized, but it provided a comprehensive overview of the enrollment patterns at ACLC College of Butuan.

**4.1.3 Data Labeling**

For analytical purposes, the raw data were labeled and categorized. Each record was assigned identifiers such as:

**Figure 20. Data Labeling**

* **Semester** (e.g., 1st Semester, 2nd Semester, Summer)
* **Academic Year** (e.g., 2018–2019, 2019–2020)
* **Course** (e.g., BSIT, BSCS, BSBA)
* **Year Level** (e.g., 1st Year, 2nd Year)
* **Enrollment Count** (numeric value)

**4.2 Data Preprocessing**

Data preprocessing was a crucial step in preparing the enrollment dataset for machine learning models. Since raw institutional records often contain inconsistencies, missing values, and categorical variables, several transformations were applied to ensure the dataset was accurate, structured, and compatible with forecasting algorithms.

Image 21. Handle Missing Value

4.2.1 Handle Missing Value

(Explanation)

Image 22. Data Reshaping

4.2.2 Data Reshaping

(Explanation)

Image 23. One Hot Encoding

4.2.3 One Hot Encoding

(Explanation)

Image 24. Normalization

4.2.4 Normalization

(Explanation)

Image 25. Create Sequence

4.2.5 Create Sequence

(Explanation)

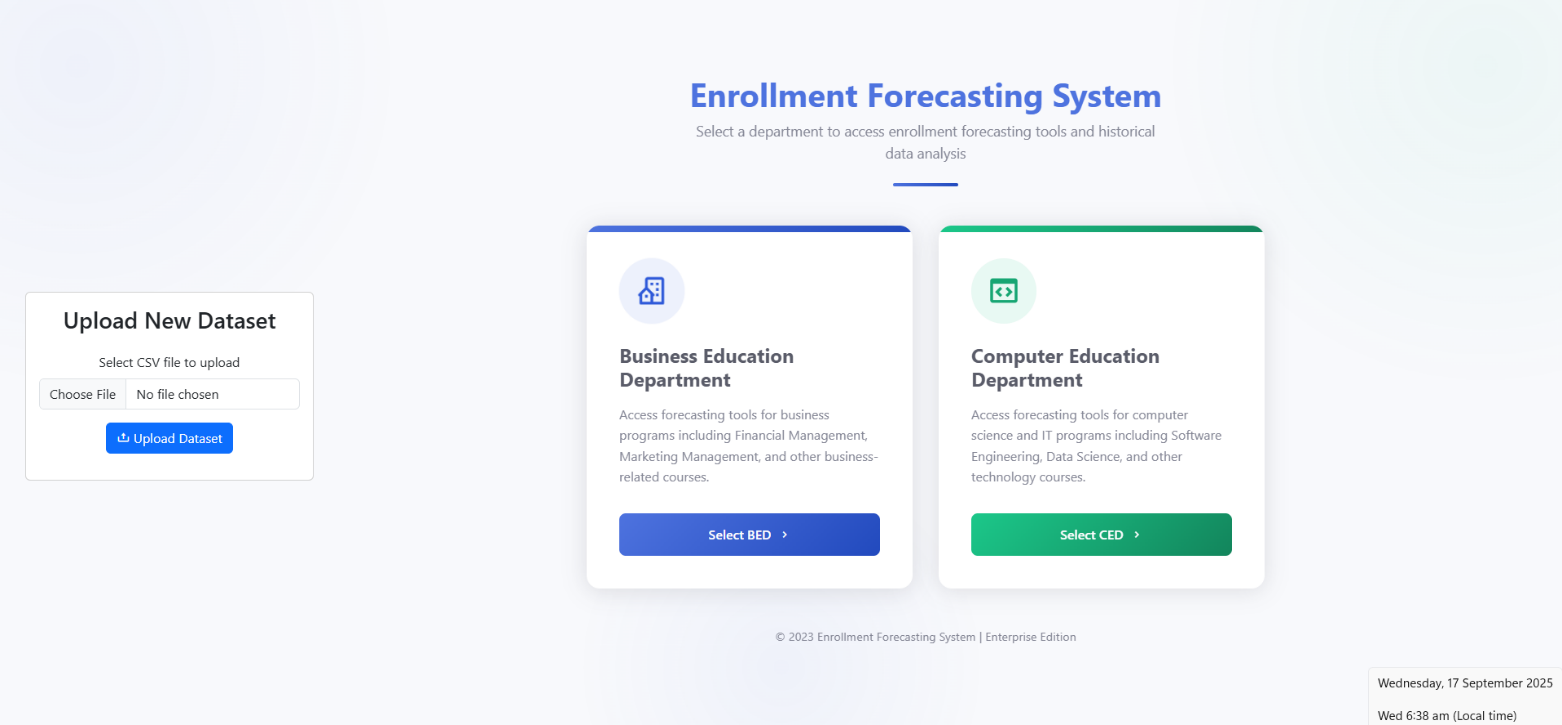
**4.3 Data Splitting**

After preprocessing, the dataset was divided into **training and testing sets** to properly evaluate the forecasting models. Following a time-series approach, 80% of the data (from 2018 up to 2023) was allocated for training, while the remaining 20% (2024–2025) was reserved for testing. This ensured that the models were trained on past records and validated on unseen future semesters, providing a fair assessment of their predictive accuracy. Unlike random splitting, chronological splitting was used to preserve the temporal order of enrollment data, which is essential for reliable time-series forecasting.

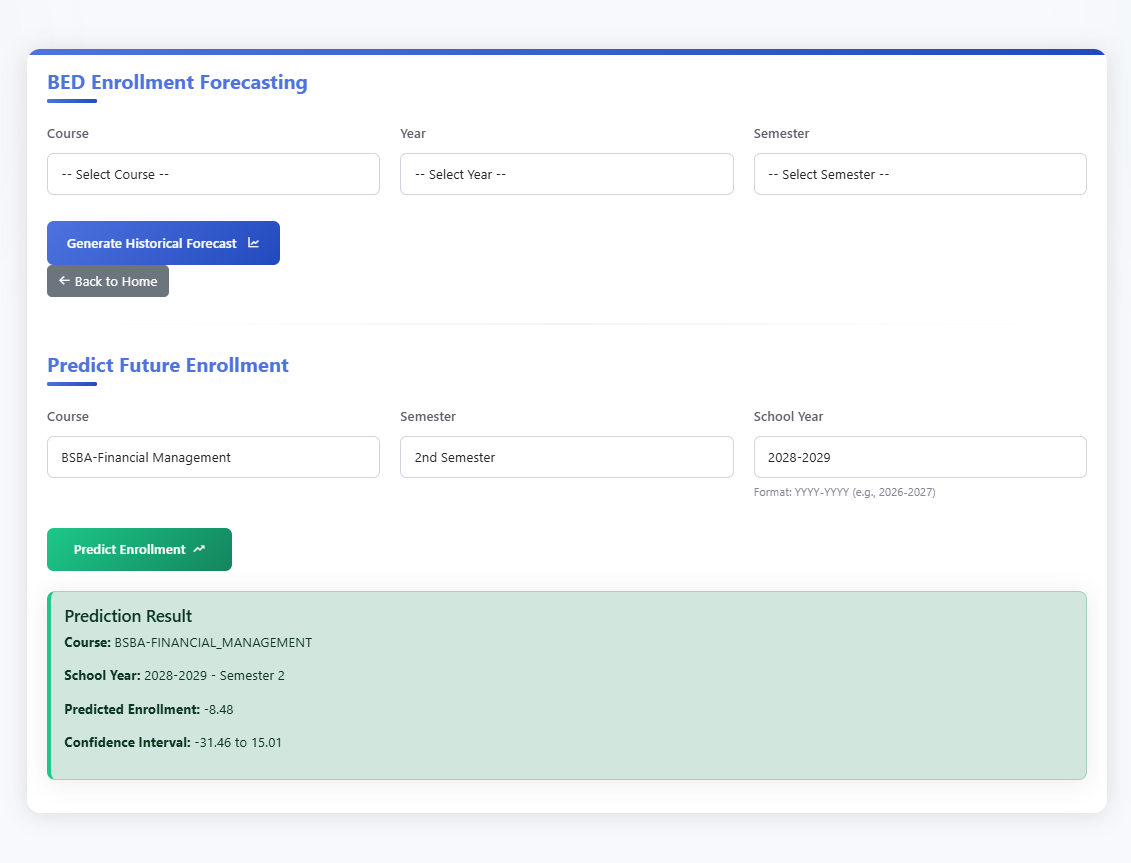
(Training, Testing, and Comparison Images)

**4.4 Visualization**

To better understand the data distribution, the enrollment records were **visualized through line graphs and trend plots** in the web application. The application provides an interactive dashboard where users can explore historical enrollment patterns by semester, course, and department. The visualization highlights fluctuations and seasonal cycles, such as the recurring increase in first-semester enrollment. Once the forecasting models are trained, the web application also overlays the **predicted values** alongside actual enrollment counts, allowing administrators to clearly compare historical trends with projected results.



In this image, the user can upload the dataset, which is the first user interface. After that, they can choose between forecasting from the Business Education Department and the Computer Education Department.



In this image, the user can choose courses, year, and semester in this graphic to predict past enrolment as well as future enrolment. This also applies to CED Enrollment Forecasting.



The image above displays the enrolment distribution, enrolment trends, annual enrolment, and enrolment data summary.

**REFERENCES**

Jain, S., Agrawal, S., Mohapatra, E., & 0000-0002-9352-0237, S. K. A.-O. (2024). *A novel ensemble ARIMA-LSTM approach for evaluating COVID-19 cases and future outbreak preparedness. PG - 409-425 LID - 10.1002/hcs2.123 [doi]* (Issues 2771-1757 (Electronic)). Health Care Science. https://doi.org/doi.org/10.1002/hcs2.123

Mun, W. K., Sufahani, S. F., Kamil, A. A., & Nawawi, M. K. M. (2025). Prediction of Gold Prices Using Hybrid Model Arima-Lstm. *Advances and Applications in Statistics*, *92*(5), 749–766. https://doi.org/10.17654/0972361725031

Parvez, R., Ali Meerza, S. I., & Hasan Khan Chowdhury, N. (2021). *Forecasting student enrollment using time series models and recurrent neural networks*. 14.

Patayon, U. B., & Crisostomo, R. V. (2022). Time Series Analysis on Enrolment Data: A case in a State University in Zamboanga del Norte, Philippines. *2022 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 13–18. https://doi.org/10.1109/ICACSIS56558.2022.9923436

Qin, L., Shanks, K., Phillips, G. A., & Bernard, D. (2019). The impact of lengths of time series on the accuracy of the ARIMA forecasting. *International Research in Higher Education*, *4*(3), 58–68.

Shobayo, O., Adeyemi-Longe, S., Popoola, O., & Okoyeigbo, O. (2025). A Comparative Analysis of Machine Learning and Deep Learning Techniques for Accurate Market Price Forecasting. In *Analytics* (Vol. 4, Issue 1). https://doi.org/10.3390/analytics4010005

Williams, R. J., & Zipser, D. (1989). A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, *1*(2), 270–280. https://doi.org/10.1162/neco.1989.1.2.270

Jain, S., Agrawal, S., Mohapatra, E., & 0000-0002-9352-0237, S. K. A.-O. (2024). *A novel ensemble ARIMA-LSTM approach for evaluating COVID-19 cases and future outbreak preparedness. PG - 409-425 LID - 10.1002/hcs2.123 [doi]* (Issues 2771-1757 (Electronic)). Health Care Science. https://doi.org/doi.org/10.1002/hcs2.123

Mun, W. K., Sufahani, S. F., Kamil, A. A., & Nawawi, M. K. M. (2025). Prediction of Gold Prices Using Hybrid Model Arima-Lstm. *Advances and Applications in Statistics*, *92*(5), 749–766. https://doi.org/10.17654/0972361725031

Parvez, R., Ali Meerza, S. I., & Hasan Khan Chowdhury, N. (2021). *Forecasting student enrollment using time series models and recurrent neural networks*. 14.

Patayon, U. B., & Crisostomo, R. V. (2022). Time Series Analysis on Enrolment Data: A case in a State University in Zamboanga del Norte, Philippines. *2022 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, 13–18. https://doi.org/10.1109/ICACSIS56558.2022.9923436

Qin, L., Shanks, K., Phillips, G. A., & Bernard, D. (2019). The impact of lengths of time series on the accuracy of the ARIMA forecasting. *International Research in Higher Education*, *4*(3), 58–68.

Shobayo, O., Adeyemi-Longe, S., Popoola, O., & Okoyeigbo, O. (2025). A Comparative Analysis of Machine Learning and Deep Learning Techniques for Accurate Market Price Forecasting. In *Analytics* (Vol. 4, Issue 1). https://doi.org/10.3390/analytics4010005

Williams, R. J., & Zipser, D. (1989). A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, *1*(2), 270–280. https://doi.org/10.1162/neco.1989.1.2.270