

**COMPREHENSIVE ANALYSIS  
AND FEATURE TRANSFORMATION OF  
HETEROGENOUS DATA FOR MODEL ANALYSIS**

Teja Rose Jijo

Student Id : 2250289

MSc CS648 Project

MSc Data Science and Analytics



Department of Computer Science

Maynooth University, Co. Kildare, Ireland

A thesis submitted in fulfilment of the requirements for the MSc Data Science and Analytics

Supervisor

Dr Muhammad Salman Pathan

Department of Computer Science

Maynooth University

August 07 2023

## ACKNOWLEDGEMENT

I would like to extend the sincere thanks to my supervisor Dr Muhammad Salman Pathaan who has been incredibly helpful during this project by dedicating his time and efforts and helping me move in right track. I also thank my friends and family for supporting me and cheering me on.

## TABLE OF CONTENTS

ACKNOWLEDGEMENT .....	2
ABSTRACT.....	5
<b>CHAPTER 1</b>	
INTRODUCTION AND MOTIVATION .....	6
1. 1 INTRODUCTION.....	6
1. 2 PROBLEM STATEMENT .....	6
1. 3 SOLUTIONS IMPLEMENTED .....	7
1.4 KAGGLE .....	7
1.5 OBJECTIVE.....	8
<b>CHAPTER 2</b>	
BACKGROUND MATERIALS .....	10
2.1 LITERATURE REVIEW .....	10
2.2 RELATED WORK.....	11
2.3 TOOLS USED.....	12
2.3.1 TABULA .....	12
2.3.2 PYTHON .....	13
2.3.3 GOOGLE COLABORATORY .....	14
<b>CHAPTER 3</b>	
CONCEPTUAL WORK DESIGN .....	16
3.1 PROCESS FLOW .....	16
3.2 METHODOLOGY .....	17
3.2.1 DATA COLLECTION .....	17
3.2.2 DATA PREPROCESSING .....	18
3.2.3 DATA ANALYSIS .....	20
3.2.4 DATA ENCODING .....	23
3.2.5 MACHINE LEARNING (ML) MODELING .....	23

3.2.6 ACCURACY TESTING .....	24
<b>CHAPTER 4</b>	
ACTUAL IMPLEMENTATION .....	27
4.1 DATA EXPLORATION .....	27
4.2 DATA MERGING .....	28
4.3 DATA CLEANING .....	30
4.3.1 HANDLING MISSING VALUES .....	30
4.3.2 CHECKING FOR DUPLICATE VALUES .....	32
4.3.3 DEALING WITH OUTLIERS .....	32
4.4 DATA ANALYSIS.....	33
4.5 DATA ENCODING .....	38
4.5.1 ONE_HOT ENCODING .....	39
4.5.2 LABEL ENCODING.....	39
4.6 ML MODELING AND ACCURACY TESTING.....	40
4.6.1 SPLITTING DATA.....	40
4.6.2 MODEL CREATION AND TRAINING.....	42
<b>CHAPTER 5</b>	
RESULTS.....	46
<b>CHAPTER 6</b>	
CONCLUSION AND FUTURE WORK.....	51
FUTURE WORK.....	52
REFERENCES .....	54
APPENDIX.....	56

## ABSTRACT

The openly accessible datasets today encompass a variety of data types, creating a heterogeneous mixture. These diverse data types are frequently used for data analysis methodologies, including the development of machine learning models. These datasets encompass a combination of different data types that necessitate careful management and analysis. Establishing effective strategies to manage categorical data becomes imperative to empower machine learning models to efficiently process and learn from these variables. This approach is crucial not only to prevent information loss or bias but also to ensure precise predictions and meaningful insights are derived from the data. This thesis explores data analysis techniques for mixed datasets, segregating by type. Numerical data underwent statistical analysis, while categorical data underwent frequency counts and visualization. This synthesis yielded comprehensive insights into data distributions. The study then delves into transforming categorical variables via encoding techniques, converting them into numerical representations. A range of encoding methods addresses categorical data challenges, enhancing model training and analysis. Encoding retains vital categorical information, seamlessly integrating into statistical models and machine learning. This integration enhances robustness and accuracy. The study highlights the pivotal role of data analysis and transformation in boosting insights and performance in analytics and modeling.

# CHAPTER 1

## INTRODUCTION AND MOTIVATION

### 1. 1 INTRODUCTION

Machine Learning is an interdisciplinary field which has concealed more or less every scientific domain [1]. It is the development of algorithms and statistical models that enable computers to learn and improve from experience without being explicitly programmed. The primary goal of machine learning is to enable computers to automatically learn and make predictions or decisions based on patterns and data. They have become indispensable tools in the age of data-driven decision-making for sifting through massive and varied datasets in search of useful patterns and insights. They work on data by processing it through a series of mathematical and statistical operations to learn patterns, relationships, and trends within the data. A variety of traditional data models and different query languages are not sufficient to process these, since data is often irregular [2]. Real-world datasets frequently include dataset with different datatypes, giving rise to what are known as "mixed data types". The dataset of mixed data types will have categorical variables which are type of data that represent different groups or categories and can take on a limited, fixed set of values. These variables do not have a numerical meaning but are used to label or classify data into distinct groups. To fully utilize the potential of such datasets, this difference presents specific challenges and calls for creative solutions.

The thesis tackles the issue of heterogeneous data by combining information from various sources, mitigating the problem of data scarcity. The goal is to enable more effective analyses of the problem by leveraging the diverse dataset, leading to a more comprehensive understanding and informed decision-making. Through this approach, the study aims to contribute valuable insights and findings to the field of research. The thesis also focuses on techniques that can be adopted analysing patterns in heterogenous data using ML models.

### 1. 2 PROBLEM STATEMENT

Data that are currently available as opensource are heterogenous datasets which contain more than one type of data types. These data types are generally used by students in different universities across world to conduct experiments on data analysis techniques such as creating ML models. All these data sets are of mixed data types which has to be maintained and analysed. Finding out coping mechanism to handle categorical data is essential to enable

machine learning models to process and learn from these variables effectively, avoid information loss or bias, and generate accurate predictions and meaningful insights from the data.

Choosing appropriate encoding methods and preprocessing techniques is vital to harness the full potential of categorical data in machine learning applications. Analysing datasets with mixed data types and non-uniformity poses challenges due to their complexity. Gaining a comprehensive understanding of such datasets remains difficult in the field of data analysis. Addressing categorical data and finding effective methods to handle it have become crucial aspects of research. As a result, numerous studies have been conducted to develop solutions and approaches to tackle these challenges effectively.

### 1.3 SOLUTIONS IMPLEMENTED

The dataset was analysed by segregating it based on the available categorical data. The numerical data underwent statistical analysis, while the categorical data was studied using frequency counts and visualization techniques. By combining these methods, a comprehensive analysis of the dataset was obtained, revealing insights into data distributions and patterns.

After analysis, categorical variables in data are transformed using encoding techniques, also known as feature transformation. These methods evaluate the features of each categorical value and convert them into numerical representations suitable for analysis. By employing various encoding techniques, researchers can effectively address the challenges associated with handling categorical data, enabling better model training and analysis. Encoding helps in preserving the valuable information within the categorical variables and facilitates their integration into statistical models and machine learning algorithms, leading to more robust and accurate results.

### 1.4 KAGGLE

The thesis opts for Kaggle as a data source, recognizing it as a prominent platform for data science and machine learning. Kaggle's extensive dataset collection, interactive coding environments (Kaggle Notebooks), and sponsored competitions make it ideal. Kernels facilitate code and analysis sharing, while forums encourage collaboration among data scientists. Learning resources aid skill enhancement, and Kaggle's job board offers career opportunities. The platform's vibrant and competitive community continues to drive innovation and knowledge sharing, playing a crucial role in advancing the field of data science.

## 1.5 OBJECTIVE

The thesis encompasses methods that are aimed at effectively analysing the heterogeneous datasets containing categorical data. These objectives are as follows:

- Data Collection: The primary objective of the thesis is to gather data from various sources, establishing the foundation for subsequent analyses. This process includes selecting the relevant domain, ensuring an adequate data collection, and addressing potential challenges related to incomplete or inconclusive data. In such cases, identifying key features within the existing dataset and seeking additional data from diverse sources becomes crucial to ensure a comprehensive and meaningful analysis.
- Data Preprocessing: This involves data preprocessing to handle various data anomalies such as missing values, duplicates, and outliers. By cleaning and preparing the data, the study ensures a smooth and reliable processing pipeline and enables the data for data analysis.
- Data Analysis: In this phase, we will examine the cleaned and merged data to identify outliers, patterns, and other relevant insights. This analysis will help us decide on appropriate encoding techniques for preparing the dataset and making it suitable for training our machine learning model. Once the encoding is applied, we will proceed to fit the dataset to the model and assess its accuracy to determine how well it performs in making predictions or classifications.
- Categorical Variable Handling: The thesis emphasizes addressing categorical variables, specifically the attributes "status" and "gender," which require special treatment due to their non-numeric nature. Encoding also known as feature transformation will be employed to effectively convert these variables into numerical representations.
- Seamless Integration with Machine Learning Model: A critical aspect of the research is to ensure the seamless integration of the pre-processed data, including the transformed categorical variables, into the machine learning model and check the model accuracy to check the effectiveness of encoding in dataset.

The thesis utilizes open-source datasets containing academic performance records of BE Computer Science students from the 7th semester at Mumbai University. The data includes marks for ten subjects, encompassing internal assessments, external exams, total marks, SGPI scores, gender, and academic status. Two categorical variables, namely "status" and "gender,"



need special handling for predictive machine learning models. Handling these categorical variables can make analysis process smooth in an ML model.

In Chapter 2, an examination will be conducted on categorical variables, encompassing their definitions, classifications, illustrated with examples. The historical context of prior research in this domain and the tools employed to attain outcomes will also be discussed. Moving on to Chapter 3, the focus shifts to the procedural sequence, methodologies, and fundamental terminologies essential for a comprehensive understanding before embarking on the implementation phase. Chapter 4 will be dedicated to the practical execution of the outlined process, and the ensuing Chapter 5 will present the derived outcomes. The conclusive Chapter 6 encapsulates findings, implications, and potential future directions stemming from this study.

## CHAPTER 2

### BACKGROUND MATERIALS

#### 2.1 LITERATURE REVIEW

Homogeneity and heterogeneity are the two main concepts with respect to the uniformity in a dataset frequently used in all sciences and statistics. A homogeneous dataset is made up of things that are similar to each other, that mean it is uniform in character or composition (i.e. colour, size, weight, height, shape, distribution, texture, architectural design, etc.); the other one which is heterogeneous is manifestly no uniform in any one of these qualities [3].

Heterogenous data constitutes 95% of Big Data [4]. They can have categorical, numerical, textual, audio, videos and different type of data as the dataset contains data from real world.

Machine learning algorithms accept only numerical inputs; hence it is necessary to encode other data types in the heterogenous dataset like categorical values. These values can be converted into numerical values using encoding techniques [5]. In this thesis, we focus on categorical values present in the heterogenous datasets.

Two main types of categorical variables are [6]:

- Nominal variables: Nominal variables have categories that have no natural order or ranking. They are merely labels used to group data. Examples of nominal variables include:
  - Colours (e.g., red, blue, green)
  - Gender (e.g., male, female, non-binary)
  - Marital status (e.g., married, single, divorced)
  - Types of animals (e.g., dog, cat, bird)
- Ordinal variables: Ordinal variables also represent categories, but they have a meaningful order or ranking between the categories. However, the difference between the categories is not quantifiable. Examples of ordinal variables include:
  - Education level (e.g., high school, bachelor's, master's, Ph.D.)
  - Economic status (e.g., low-income, middle-income, high-income)
  - Likert scale responses (e.g., strongly agree, agree, neutral, disagree, strongly disagree)

Here's an example of a dataset containing categorical variables for a group of people:

Name	Gender	Marital Status	Education Level	Employment Status
Alice	Female	Married	Master's	Employed
Bob	Male	Single	Bachelor's	Unemployed

## 2.2 RELATED WORK

The historical development of statistical analysis and modeling techniques for categorical variables can be traced back to the groundbreaking works of renowned statisticians and researchers. Ronald A. Fisher's work in 1922, titled "On the Mathematical Foundations of Theoretical Statistics," laid the groundwork for statistical analysis, including the treatment of categorical variables. Karl Pearson's contribution in 1900, with his paper "On the Criterion That a Given System of Deviations from the Probable in the Case of a Correlated System of Variables Is Such That It Can Be Reasonably Supposed to Have Arisen from Random Sampling," introduced the chi-squared test, which remains widely used for testing associations between categorical variables. In 1958, the book "The Advanced Theory of Statistics, Volume 1" by Kendall, M., and Stuart, A., became an influential reference for various statistical methods for analysing categorical data, including contingency tables and tests for association. Subsequently, in 1990, Agresti, A., published "Categorical Data Analysis," a seminal book extensively covering the analysis of categorical data, introducing many essential statistical techniques for modeling and interpreting such data.

Moving towards the era of machine learning and data analysis, the book "The Elements of Statistical Learning" by Hastie, T., Tibshirani, R., & Friedman, J. in 2009, introduced several machine learning algorithms widely used for analysing mixed datasets containing both categorical and numerical variables. Recent advancements have also focused on modeling categorical data in specific domains. In 2019, Pedersen, E.J., Miller, D.L., Simpson, G.L., & Ross, N., presented "Hierarchical generalized additive models in ecology: an introduction with mgcv," exploring modeling categorical data in the context of ecological research using generalized additive models. Additionally, in 2021, Petropoulos, G., & Kourentzes, N., introduced "The Theta Model: A decomposition approach to forecasting," which tackles categorical time series data through the Theta method.

These works collectively reflect the continuous evolution and significance of research and methodologies related to categorical variables, shaping the landscape of statistical analysis and data modeling in various scientific fields.

The thesis investigates the impact of explored one-hot and label encoding on categorical data, in the heterogenous dataset, converting them into numerical representations for machine learning. Evaluating the resulting datasets on three models allows to assess the effectiveness of these techniques, measuring model accuracy. This research contributes valuable insights, aiding researchers and practitioners in making informed decisions when dealing with categorical data in machine learning, optimizing data preprocessing for improved model performance.

## 2.3 TOOLS USED

### 2.3.1 TABULA

Tabula is an open-source software tool specifically designed for extracting tables from PDF documents. The dataset we had was incomplete and didn't provide a complete information on the students. So, the information including total points and status were extracted from the related pdf provided. Reasons to choose tabula are:

- **User-Friendly Interface:** Tabula offers a simple and intuitive user interface, making it easy for users to upload PDF files, select tables, and export data without requiring advanced technical skills.
- **Accuracy:** Tabula utilizes advanced algorithms to accurately extract tables from PDF documents. It handles complex table structures, including merged cells and nested tables, with a high degree of precision.
- **Flexibility in Table Selection:** Users can manually draw selection areas around tables in the PDF, enabling precise control over the data extraction process. This feature is especially useful for documents with multiple tables or irregular table layouts.
- **Batch Processing:** Tabula supports batch processing, allowing users to extract tables from multiple PDF files simultaneously. This saves time and streamlines the data extraction workflow for large-scale projects.

- **Open-Source and Free:** Tabula is open-source software, which means it is freely available to use and modify. This makes it an accessible and cost-effective choice for users who need to extract data from PDFs regularly.
- **Platform Independence:** Tabula is cross-platform, compatible with Windows, macOS, and Linux operating systems, ensuring accessibility and convenience for users on different platforms.
- **Community Support:** Being open-source, Tabula benefits from a vibrant community of users and developers who actively contribute to its improvement and provide support through forums and documentation.
- **Privacy and Security:** Since Tabula is a desktop-based tool, all data extraction happens locally on the user's machine, ensuring the privacy and security of sensitive data.

### 2.3.2 PYTHON

Python makes it easy for data analysis as it provides different set of libraries for data exploration, data manipulation, building machine learning models using different algorithms. This is a widely used programming language because of its flexibility, availability and easy use. Libraries used in this thesis are:

#### 1.1 NumPy

NumPy is fundamental library and these are used for doing mathematical calculations needed to fill in null values and check the patterns in dataset to identify outliers and other manipulation tasks

#### 1.2 Pandas

Pandas is popular library built on top of NumPy that provides high performance data structures and data analysis tools. It is used to understand the data set, create data frames, which are powerful data structure for working with structured data, and data merging function.

#### 1.3 Scikit-learn

Scikit-learn is a popular machine learning library in Python. This is a most used library after Pandas. Use of Scikit-learn is mentioned as follows:

- Encoding categorical variables: The module “sklearn.preprocessing” in scikit-learn is used to apply one-hot encoding and label encoding techniques in categorical data. They are used such that they can be

converted to numerical values which are readable by machine learning models.

- Divide Data set: The module “sklearn.model\_selection” is used to dividing dataset into training set and testing set. Training set is used to train the machine learning model build and testing set is used to check model accuracy.
- Building Machine Learning Models: The module “sklearn.ensemble” is used to build machine learning models Gradient Boosting Model (GBM) and Random forest classifier and “sklearn.tree” was used to build decisiontree classifier and the tree to determine how the encoding techniques works on. module
- Calculating Model Accuracy: The module “sklearn.metrics” is used to calculate accuracy of models created to check efficiency of encoding techniques.

### 2.3.3 GOOGLE COLABORATORY

Google Colaboratory, commonly known as Google Colab, is a cloud-based, interactive development environment provided by Google. It allows users to write and execute Python code in a web browser, eliminating the need for setting up local development environments. Colab is particularly popular among data scientists and machine learning researchers as it provides free access to GPUs and TPUs (Tensor Processing Units) for accelerating computations.

Reasons to choose Google Colab over other development environments are:

- Jupyter Notebooks: Colab is built on Jupyter Notebooks, enabling users to create interactive documents that combine code, text, and visualizations.
- Free GPU/TPU: Colab provides free access to GPUs and TPUs, allowing users to perform computationally intensive tasks like training deep learning models more efficiently.
- Collaborative Editing: Multiple users can collaborate in real-time on the same Colab notebook, making it suitable for teamwork and code sharing.
- Pre-installed Libraries: Colab comes pre-installed with popular data science libraries like NumPy, Pandas, Scikit-learn which are used in thesis.

- Cloud Storage Integration: Colab allows users to import data directly from Google Drive and save their notebooks to Google Drive, facilitating data sharing and persistence.
- Code Snippet and Output Sharing: Colab notebooks can be shared publicly, allowing others to view the code and output or copy and run the code snippets.
- Markdown Support: Users can add formatted text, images, and equations using Markdown cells in Colab notebooks.

## CHAPTER 3

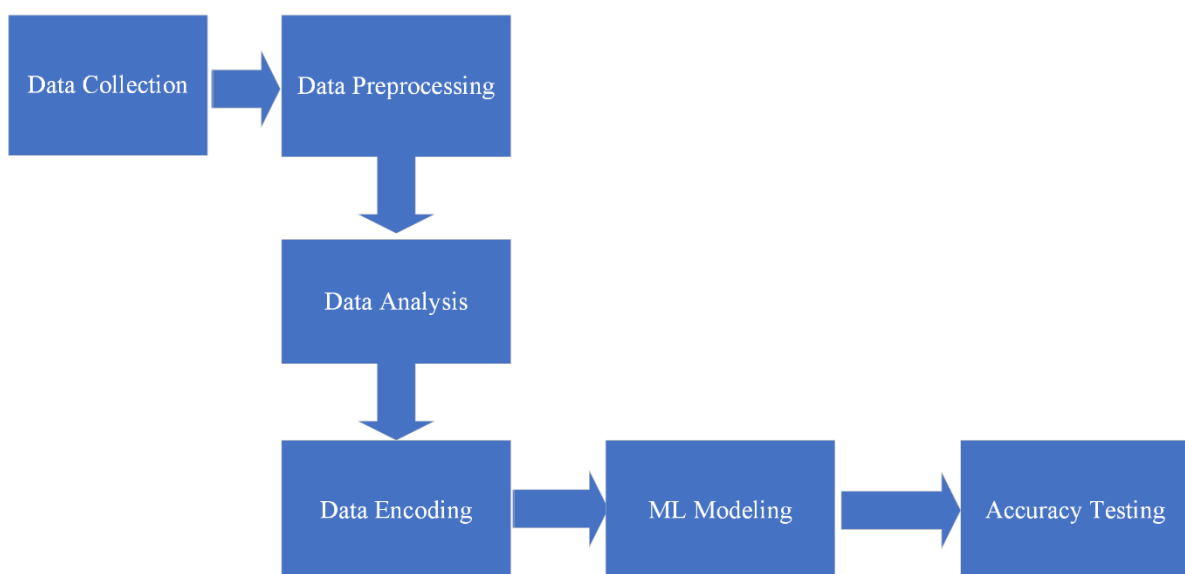
### CONCEPTUAL WORK DESIGN

Creating a framework on how to work on the problem and how the work should be organised is important. It involves defining the overall structure of tasks, and workflows in a way that aligns with the organization's goals and objectives. This sets the foundation for more detailed operational and procedural design, helping to optimize efficiency, productivity, and employee satisfaction. It encompasses the strategic decision-making aspects of work organization and serves as a blueprint for the implementation of work processes and systems within the organization.

#### 3.1 PROCESS FLOW

Creating a process flow is vital as it provides a visual representation of a workflow, enhancing understanding and reducing confusion among stakeholders. It helps identify inefficiencies, standardize procedures, and supports decision-making for process optimization. Additionally, process flows aid in training, compliance, and risk management, fostering continuous improvement and efficient collaboration within organizations.

The process flow of the thesis is as follows:



*Fig 3-3-1 Process flow of the thesis*



As the flow chart above, the work design of the thesis is divided into 6 parts. They are

- **Data Collection:** Collecting relevant and heterogenous data from various sources for analysis and model creation.
- **Data Preprocessing:** Preprocessing the data by cleaning, handling missing values, removing duplicates, and transforming it into a consistent format to ensure high-quality data.
- **Data Analysis:** Analysing the pre-processed data by exploring, calculating summary statistics, and visualizing it to identify patterns, trends, and anomalies for decision-making.
- **Data Encoding:** Converting categorical data into numerical form using techniques like one-hot encoding and label encoding to make it suitable for machine learning models.
- **ML Modeling:** Creating a machine learning model by selecting an appropriate algorithm based on the problem and training it on the encoded data to learn patterns.
- **Accuracy Testing:** Evaluating the model's performance using a test dataset and metrics like accuracy, precision, recall, and F1 score to assess how well it generalizes to new data.
- **Efficiency of Encoding Techniques:** Assessing the efficiency of data encoding techniques by tabulating accuracy scores received on each classification model, measuring efficiency of one-hot and label encoding.

## 3.2 METHODOLOGY

### 3.2.1 DATA COLLECTION

The process of data collection for this study involved sourcing datasets from online forums, such as Kaggle, which provide open-source datasets accessible to students and researchers. Initially, the dataset obtained contained marks for 10 subjects from Mumbai University's seventh semester. However, it lacked essential information, such as whether the student passed or failed, as well as the total marks obtained by each student. Recognizing the need for additional data to make the analysis more meaningful, an additional dataset that included total marks and other crucial features was created from the files that was available with the dataset. The file was in pdf format and data from this pdf is collected according to the requirements

and another dataset was created based on common fields in the earlier dataset. These two datasets are considered for next step, i.e., data preprocessing.

### 3.2.2 DATA PREPROCESSING

Data Pre-processing is the First step in Machine learning in which the data gets transformed/Encoded so that it can be brought in such a state that now the machine can quickly go through or parse that Data [7]. Data pre-processing steps are designed in such a way that data receiving as final result will be much easy to analyse and encode. The whole process is divided into two parts. They are:

- Data Merging
- Data Cleaning

Initially, the collected dataset may contain incomplete information, making it insufficient to draw meaningful conclusions. This led to decision of creating a meaningful dataset and merge with first one. Merging datasets is a key operation for data analytics. A frequent requirement for merging is joining across columns that have different surface forms for the same entity [8].

The above limitation was solved by making additional related datasets were created using the pdf existed related to the students' marks. The data in the pdf was mapped and collected to excel sheet using Tabula and new dataset was created focusing on common columns to obtain more comprehensive insights. Leveraging the Pandas library in Python, the datasets are merged, creating a cohesive and enriched dataset. Subsequently, the merged dataset undergoes data cleaning to handle missing values, outliers, and ensure data consistency. The result is a high-quality dataset, ready for more accurate and effective analysis and machine learning modeling.

Once the dataset is merged, it undergoes the data cleaning process, which involves various techniques to identify and address outliers and null values. Once the dataset is merged, it undergoes the data cleaning process, which involves various techniques to identify and address outliers and null values. It is a critical step in ensuring that the dataset is devoid of incorrect or erroneous data. It can be done manually with data wrangling tools, or it can be completed automatically with a computer program [9].

Techniques used to check and remove outliers and null values are:

1. Imputation (for numerical data):

Imputation for numerical data is a technique used to handle missing values in a dataset. When dealing with numerical data, missing values can arise due to various reasons such as data collection errors or data unavailability. Imputation is the process of replacing these missing values with estimated values based on the available data, ensuring that the dataset remains complete and usable for analysis and modeling. The choice of imputation method depends on the nature of the data and the underlying assumptions. Here we choose custom imputation.

#### Custom Imputation

It is a specific data imputation technique that you designed based on the structure and characteristics of your dataset. Missing columns were marks of the students. Columns included internal assessment, external exams and total marks of 10 subjects. One of internal assessment or external assessment columns were missing. Since they have relationship with total grades column, custom imputation method is best suitable methods to handle null values.

Custom imputation methods like this can be effective in certain scenarios where domain knowledge suggests that the missing values can be estimated using information from nearby columns.

2. Frequency Checks (for categorical data)

Frequency checks for categorical data involve examining the distribution of categories within each categorical variable to ensure that the data is valid, complete, and representative. These checks help identify any inconsistencies, missing values, or unexpected patterns in the data. By conducting frequency checks for categorical data, data analysts can gain a better understanding of the data's characteristics, identify potential data quality issues, and make informed decisions regarding data preprocessing and modeling strategies. These checks contribute to ensuring the reliability and accuracy of the analysis results and will enable dataset to go to further steps of data analysis.

### 3.2.3 DATA ANALYSIS

Data analysis encompasses examining, cleansing, converting, and interpreting data to derive valuable insights, patterns, and trends. The dataset at hand comprises two types of values: numerical and categorical. Analysing such a dataset requires employing a mix of methods to extract meaningful information and draw significant conclusions. This process involves handling both numerical and categorical data appropriately to unveil valuable insights and support informed interpretations. Since we have numerical and categorical data, the analysis is divided into statistical analysis and visualizations.

#### STATISTICAL ANALYSIS

Statistical analysis of numeric data is fundamental in understanding data patterns, drawing meaningful insights, and making data-driven decisions across various fields, including scientific research, finance, engineering, social sciences, and many more. It helps researchers and analysts make evidence-based conclusions and supports the development of predictive models to aid in forecasting and decision-making.

Statistics analysis techniques used are:

- **Correlation Matrix**

A correlation matrix is a square matrix that displays the pairwise correlations between variables in a dataset. It is a valuable tool in data analysis as it helps identify the strength and direction of relationships between numerical variables. Choosing this method will help to identify specific patterns in the data. This can also bring out relationship between variables.

In a correlation matrix:

- The diagonal elements contain the correlations of each variable with itself, which are always equal to 1, as a variable is perfectly correlated with itself.
- The off-diagonal elements show the correlations between pairs of variables. The value can range from -1 to +1, where -1 indicates a perfect negative correlation, +1 indicates a perfect positive correlation, and 0 indicates no correlation.
- The values generated can be visualised using heatmap.

- Heatmap

A heatmap is a graphical representation of a correlation matrix, where the strength and direction of relationships between variables are visualized using colors. Heatmaps are commonly used to display the correlations between numerical variables in a dataset. They are used in thesis to bring out patterns and can show how these datapoints are distributed.

In a heatmap:

- Each cell represents the correlation between two variables.
- The color of each cell indicates the strength and direction of the correlation. Warm colors (e.g., red or orange) represent positive correlations, while cool colors (e.g., blue or green) represent negative correlations.
- The intensity of the color reflects the magnitude of the correlation. Darker colors indicate stronger correlations, while lighter colors indicate weaker correlations.

- Boxplot

Boxplots, also known as box-and-whisker plots, are graphical representations that display the distribution and spread of numerical data. They provide a concise summary of the data's central tendency, variability, and presence of outliers. Boxplots are valuable tools for comparing distributions, detecting skewness, identifying potential outliers, and providing a visual summary of the data's variability. They are commonly used in exploratory data analysis (EDA) to gain insights into numerical data and make data-driven decisions in fields like statistics, data science, and data visualization.

In a boxplot:

- The box represents the interquartile range (IQR), which contains the middle 50% of the data. The lower and upper edges of the box correspond to the first quartile (Q1) and third quartile (Q3), respectively. The length of the box illustrates the spread of the middle 50% of the data.
- Inside the box, a line or dot denotes the median, which is the central value of the dataset.

- The whiskers extend from the edges of the box to indicate the range of the data. They are often calculated using a specific formula or can be defined based on statistical rules. Points outside the whiskers are considered outliers.
- Outliers, if present, are individual data points that fall significantly far from the rest of the dataset. They are displayed as individual points beyond the whiskers.

They are used in thesis to check the spread of data points and understand how students scored marks generally in each year.

## VISUALIZATIONS

Visualizations of categorical data analysis is about understanding the characteristics of different categories, detecting patterns, and uncovering relationships. Appropriate visualization techniques and statistical tests help in making informed decisions, drawing meaningful insights, and gaining a deeper understanding of the dataset.

Visualization techniques used to analyse categorical variables are:

- Pie Chart

A pie chart is a circular graphical representation that displays data as slices of a circle, with each slice representing a portion or percentage of the whole. Pie charts are used to visualize the distribution of categorical data and show the relationship between different categories.

In a pie chart:

- Each slice corresponds to a specific category or group in the dataset.
- The size of each slice is proportional to the percentage or frequency of that category relative to the whole dataset.
- The entire pie represents 100% of the data, representing the entire set of categories.

In the thesis, categorical data are visually analysed by pie chart.

These analysis plays a crucial role in data encoding by providing insights and information that help make informed decisions about how to encode categorical data effectively

### 3.2.4 DATA ENCODING

Data encoding is the process of converting categorical data into a numerical format that can be used by machine learning algorithms. Many machine learning models and algorithms require input data in numerical form to perform calculations and make predictions effectively. Data encoding is a critical step in preparing the data for modeling.

Encoding techniques used for thesis are:

- **One-Hot Encoding:** In this method, each unique category in a categorical variable is converted into a binary vector. For each category, a new binary column is created, where the presence of the category is represented by "1," and the absence by "0."
- **Label Encoding:** Label encoding assigns a unique integer value to each category in the variable. It transforms categorical labels into numerical values, allowing the model to understand the ordinal relationship between categories.

### 3.2.5 MACHINE LEARNING (ML) MODELING

After encoding the categorical data, they can be introduced to machine learning models. This is the process of using algorithms and statistical techniques to build predictive or descriptive models from data. These models can be trained on historical data to make predictions or identify patterns in new, unseen data.

Before choosing the model for training, we have to understand the type of ML model we are using. There are two types of ML models. They are

- **Supervised Learning Models:** In supervised learning, the model is trained on a labelled dataset, where both the input features and the corresponding target labels are provided. The goal is to learn the mapping between the input and output to make predictions on new, unseen data.
- **Unsupervised Learning Models:** In unsupervised learning, the model is trained on an unlabelled dataset, where only the input features are provided. The model aims to find patterns, relationships, or structure within the data without any explicit target labels

Since we are using labelled dataset, we are going to conduct supervised learning. After deciding the type of machine learning, we have to decide which type of supervised learning

technique is going to be applied on the dataset. There are two type of supervised learning techniques. They are:

- **Regression:** In regression, the output variable is continuous, meaning it can take on any real value within a range. The goal of regression is to predict a numerical value, such as predicting the price of a house or the temperature.
- **Classification:** In classification, the output variable is categorical, meaning it falls into specific classes or categories. The goal of classification is to assign an input data point to a specific class or category, such as classifying emails as spam or non-spam or identifying whether an image contains a cat or a dog.

Since we have to determine whether the student passes for semester or not and they are categorical variable, this is a classification problem.

Some of the classification algorithms used in the thesis are:

- **Decision Trees:**  
Decision trees are a popular and intuitive machine learning model used for both regression and classification tasks. They are a type of supervised learning algorithm that learns a series of if-else decision rules from the data to make predictions.
- **Random Forests**  
Random Forest is an ensemble learning method that combines multiple decision trees to create a more powerful and robust model. It is a popular machine learning algorithm used for both classification and regression tasks. Random Forest is designed to address some of the limitations of individual decision trees, such as overfitting and sensitivity to small changes in the data.
- **Gradient Boosting Machines (GBMs):**  
GBMs, such as XGBoost and LightGBM, are powerful ensemble models that can handle heterogeneous data effectively. They can handle both numerical and categorical variables and are known for their high predictive accuracy.

### 3.2.6 ACCURACY TESTING

Accuracy testing, also known as model evaluation or performance evaluation, is assessing the effectiveness of a machine learning model. It aims to measure how well the model performs



in making predictions on new, unseen data. The primary goal is to determine the model's accuracy in correctly classifying or predicting outcomes.

Accuracy testing methods used thesis are:

- **Confusion Matrix:** A table that summarizes the model's performance by comparing predicted class labels with the actual class labels, showing true positives, true negatives, false positives, and false negatives.
- **Classification report:** A classification report is a standard evaluation metric used to assess the performance of a classification model. It provides a comprehensive summary of key classification metrics for each class in the dataset. The classification report includes metrics such as precision, recall, F1-score, and support. It is particularly useful when dealing with imbalanced datasets, where the number of samples in different classes varies significantly.

Here is a breakdown of the metrics included in a classification report:

- **Precision:** Precision is the ratio of true positive predictions to the total number of positive predictions made by the model. It measures the model's ability to correctly identify positive samples and avoid false positives.
- **Recall:** Recall is the ratio of true positive predictions to the total number of actual positive samples in the dataset. It measures the model's ability to correctly identify all positive samples, including the ones that were missed i.e., false negatives.
- **F1-score:** The F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall and is useful when the classes are imbalanced.
- **Support:** Support represents the number of samples in each class in the dataset. It helps to understand the distribution of the data and is useful in interpreting the significance of the metrics for each class.

By using these methods, we can work with the mixed data efficiently. We will explore, combine, and clean the data to better understand it. Then, we will use encoding techniques to handle categorical data properly, ensuring we keep the valuable information intact. With all

these steps in place, we can build machine learning models and test them accurately, leading to valuable insights and better performance in handling the diverse data.

## CHAPTER 4

### ACTUAL IMPLEMENTATION

The implementation part, methods to analyse heterogenous data, were taken by following steps:

- Data Exploration
- Data Merging
- Data Cleaning
- Data Analysis
- Data Encoding
- ML Modeling and Accuracy Testing

#### 4.1 DATA EXPLORATION

The dataset was released by Mumbai University containing the result of semester 7 for B.E Computer Science. It contains details of students from 2014 - 2022 in which most of the students belong to year 2019.

```
results.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5211 entries, 0 to 5210
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype  
---  --
 0   seat_no             5211 non-null  int64  
 1   total_gradepoints    5211 non-null  float64
 2   sgpi                 5211 non-null  float64
 3   status               5211 non-null  object  
 4   year_of_admission    5211 non-null  int64  
 5   gender               5211 non-null  object  
dtypes: float64(2), int64(2), object(2)
memory usage: 244.4+ KB
```

File 1 (results.csv)

The dataset was created by extracting data from a pdf file that was with the other dataset using Tabula. After the extraction, necessary values were taken and dataset was created.

The dataset, results.csv, has details about the grade points, sgpi, year of admission and status of each student, whether the student

passed or not are mentioned. It contains 9 columns and 5211 rows. Below table shows fields, description and data type of each column



marks.info()



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5211 entries, 0 to 5210
Data columns (total 31 columns):
#   Column                Non-Null Count  Dtype
---  -
0   seat_no                5211 non-null   int64
1   paper_1_IA             4836 non-null   float64
2   paper_2_IA             4837 non-null   float64
3   paper_3_IA             4838 non-null   float64
4   paper_4_IA             4838 non-null   float64
5   paper_5_IA             4838 non-null   float64
6   paper_6_IA             4834 non-null   float64
7   paper_7_IA             4830 non-null   float64
8   paper_8_IA             4835 non-null   float64
9   paper_9_IA             4837 non-null   float64
10  paper_10_IA            4832 non-null   float64
11  paper_1_TOT            4807 non-null   float64
12  paper_2_TOT            4805 non-null   float64
13  paper_3_TOT            4838 non-null   float64
14  paper_4_TOT            4838 non-null   float64
15  paper_5_TOT            4838 non-null   float64
16  paper_6_TOT            4804 non-null   float64
17  paper_7_TOT            4720 non-null   float64
18  paper_8_TOT            4801 non-null   float64
19  paper_9_TOT            4837 non-null   float64
20  paper_10_TOT           4832 non-null   float64
21  paper_1_EX             5085 non-null   float64
22  paper_2_EX             5086 non-null   float64
23  paper_3_EX             4803 non-null   float64
24  paper_4_EX             4803 non-null   float64
25  paper_5_EX             4838 non-null   float64
26  paper_6_EX             5084 non-null   float64
27  paper_7_EX             5003 non-null   float64
28  paper_8_EX             5081 non-null   float64
29  paper_9_EX             4838 non-null   float64
30  paper_10_EX            4838 non-null   float64
dtypes: float64(30), int64(1)
memory usage: 1.2 MB
```

File -2 (marks.csv)

The file contains 31 columns and 5211 rows. The columns include seat number of student, internal assessment, exam marks and total marks ten subjects of each student. Below table shows the columns and fig 3.2 shows data type of each column.

## 4.2 DATA MERGING

From the above data frame information, we can see that

- File 2, marks.csv does provide marks of every student in all the subject for every exam. But it does not provide any information on student status like did they pass or failed, total grades etc, which is incomplete.

- File 1, results show final results of every student but does not show how the marks are obtained by each student, which is also incomplete.

The above reasons motivated the option of merging the two files to one, so that all the information regarding exams will be available in one dataset which will be much helpful in future analysis. On looking at the data frames, we can see the column, seat no, is common in both of them. So, it is easy and better option to merge two of the datasets.

```
# merging marks and results dataset
final_marksheet = pd.merge(marks, results, on='seat_no')
final_marksheet.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5211 entries, 0 to 5210
Data columns (total 36 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   seat_no                               5211 non-null   int64
1   paper_1_IA                           4836 non-null   float64
2   paper_2_IA                           4837 non-null   float64
3   paper_3_IA                           4838 non-null   float64
4   paper_4_IA                           4838 non-null   float64
5   paper_5_IA                           4838 non-null   float64
6   paper_6_IA                           4834 non-null   float64
7   paper_7_IA                           4830 non-null   float64
8   paper_8_IA                           4835 non-null   float64
9   paper_9_IA                           4837 non-null   float64
10  paper_10_IA                          4832 non-null   float64
11  paper_1_TOT                          4807 non-null   float64
12  paper_2_TOT                          4805 non-null   float64
13  paper_3_TOT                          4838 non-null   float64
14  paper_4_TOT                          4838 non-null   float64
15  paper_5_TOT                          4838 non-null   float64
16  paper_6_TOT                          4804 non-null   float64
17  paper_7_TOT                          4720 non-null   float64
18  paper_8_TOT                          4801 non-null   float64
19  paper_9_TOT                          4837 non-null   float64
20  paper_10_TOT                        4832 non-null   float64
21  paper_1_EX                           5085 non-null   float64
22  paper_2_EX                           5086 non-null   float64
23  paper_3_EX                           4803 non-null   float64
24  paper_4_EX                           4803 non-null   float64
25  paper_5_EX                           4838 non-null   float64
26  paper_6_EX                           5084 non-null   float64
27  paper_7_EX                           5003 non-null   float64
28  paper_8_EX                           5081 non-null   float64
29  paper_9_EX                           4838 non-null   float64
30  paper_10_EX                         4838 non-null   float64
31  total_gradepoints                    5211 non-null   float64
32  sgpi                                 5211 non-null   float64
33  status                              5211 non-null   object
34  year_of_admission                    5211 non-null   int64
35  gender                              5211 non-null   object
dtypes: float64(32), int64(2), object(2)
memory usage: 1.5+ MB
```

The two datasets are merged to form a new dataset, named final\_marksheet.csv, having 35 columns and 5211 rows. This dataset will be used in future for Data analysis after data cleaning.

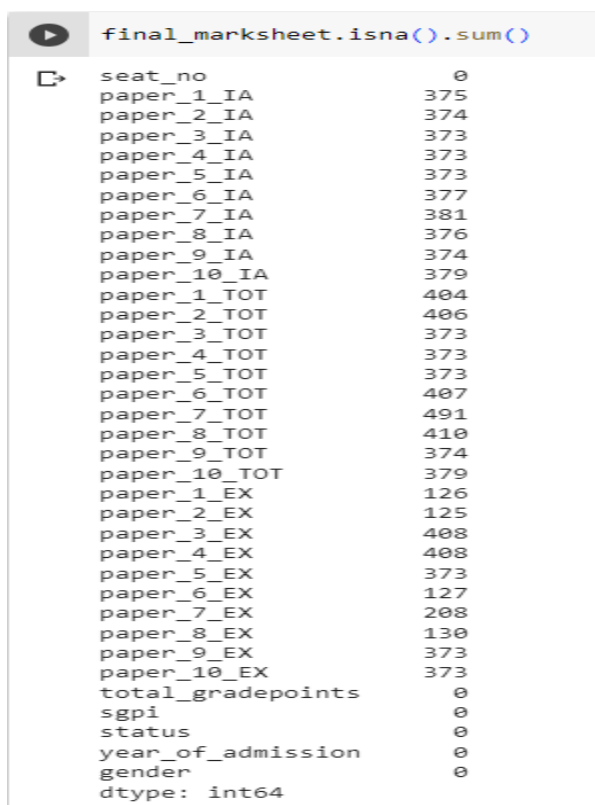
### 4.3 DATA CLEANING

Data Cleaning is important part as it can give dataset without any duplicates or errors.

The data cleaning was performed in three steps.

- Handling missing values
- Checking for duplicate values
- Dealing with outliers

#### 4.3.1 HANDLING MISSING VALUES



```
final_marksheet.isna().sum()
```

seat_no	0
paper_1_IA	375
paper_2_IA	374
paper_3_IA	373
paper_4_IA	373
paper_5_IA	373
paper_6_IA	377
paper_7_IA	381
paper_8_IA	376
paper_9_IA	374
paper_10_IA	379
paper_1_TOT	404
paper_2_TOT	406
paper_3_TOT	373
paper_4_TOT	373
paper_5_TOT	373
paper_6_TOT	407
paper_7_TOT	491
paper_8_TOT	410
paper_9_TOT	374
paper_10_TOT	379
paper_1_EX	126
paper_2_EX	125
paper_3_EX	408
paper_4_EX	408
paper_5_EX	373
paper_6_EX	127
paper_7_EX	208
paper_8_EX	130
paper_9_EX	373
paper_10_EX	373
total_gradepoints	0
sgpi	0
status	0
year_of_admission	0
gender	0
dtype: int64	

From the image, we can see that the dataset has a considerable number of null values in marks field. In the data description, it is clearly mentioned that, if the columns are null value, that means the students either failed in external assessment or internal assessment and the status will be considered as RR (Reserved).

Taking this into consideration, all the null values in the fields of internal assessments and external exams were filled with 0.0.


Using Consistency Checks, it was found that, the column labelled with "\_TOT" has random rather than having sum of the respective \_IA and \_EX columns and the values that do not appear to be typical statistical measures like the median or mean.

Using custom imputation technique, we created pattern “paper\_1\_TOT = paper\_1\_IA + paper\_1\_EX”, python code was created to fill all the \_TOT columns with their sum of respective \_IA and EX columns. Algorithm to complete the task is as follows:


### Algorithm

- Initialize i as 1 to start the loop, and the final\_marksheet
- Loop through i from 1 to 10 (inclusive):
  - Create strings ia\_column, exam\_column, and tot\_column to represent the column names for internal assessment, external examination, and total marks for the current paper i.
  - Calculate the total marks for the current paper i by adding the values from the ia\_column and exam\_column.
  - Assign the calculated total marks to the tot\_column in the final\_marksheet.
- After the loop completes, the final\_marksheet will have new columns for the total marks of each paper.

After replacing all the \_TOT columns with sum of their respective fields, all the null values were handled the image is as follows.



```
final_marksheet.isna().sum()
```



seat_no	0	paper_7_IA	0
paper_1_IA	0	paper_7_EX	0
paper_1_EX	0	paper_7_TOT	0
paper_1_TOT	0	paper_8_IA	0
paper_2_IA	0	paper_8_EX	0
paper_2_EX	0	paper_8_TOT	0
paper_2_TOT	0	paper_9_IA	0
paper_3_IA	0	paper_9_EX	0
paper_3_EX	0	paper_9_TOT	0
paper_3_TOT	0	paper_10_IA	0
paper_4_IA	0	paper_10_EX	0
paper_4_EX	0	paper_10_TOT	0
paper_4_TOT	0	total_gradepoints	0
paper_5_IA	0	sgpi	0
paper_5_EX	0	status	0
paper_5_TOT	0	year_of_admission	0
paper_6_IA	0	gender	0
paper_6_EX	0	dtype: int64	
paper_6_TOT	0		

#### 4.3.2 CHECKING FOR DUPLICATE VALUES

The column seat no was considered to the column which separates each student details and there were no duplicate values found on both of the files.

```
final_marksheet.duplicated().sum()
```

```
0
```

#### 4.3.3 DEALING WITH OUTLIERS

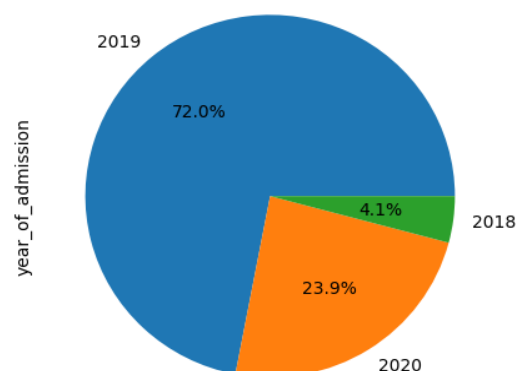
Using Frequency Counts, it was found that, frequency of students in year 2014, 2015, 2016, 2017, 2021 and 2022 are very low and not enough to make valid analysis of particular year.

```
# checking number of students in each year
```

```
final_marksheet['year_of_admission'].value_counts()
```

```
2019    3708
2020    1230
2018     213
2017     42
2022     12
2016      3
2014      1
2021      1
2015      1
Name: year_of_admission, dtype: int64
```

This outlier was removed by calculating the proportion of students in each year. Number of students who got into colleges in the university in 2019 constitutes 71.2% of the data and students whose admission year is 2020 constitutes 23.6% of the data. 2018 holds 4.1% of the data and rest of the years, 2014 – 2017 and 2021 – 2022 holds only 1.1%. So, data belonging to years 2014 – 2017 and 2021 – 2022 was dropped in order to make accurate analysis. The result formed dataset with 72% of students from 2019, 23.9% from 2020 and 4.1% from 2018.





## 4.4 DATA ANALYSIS

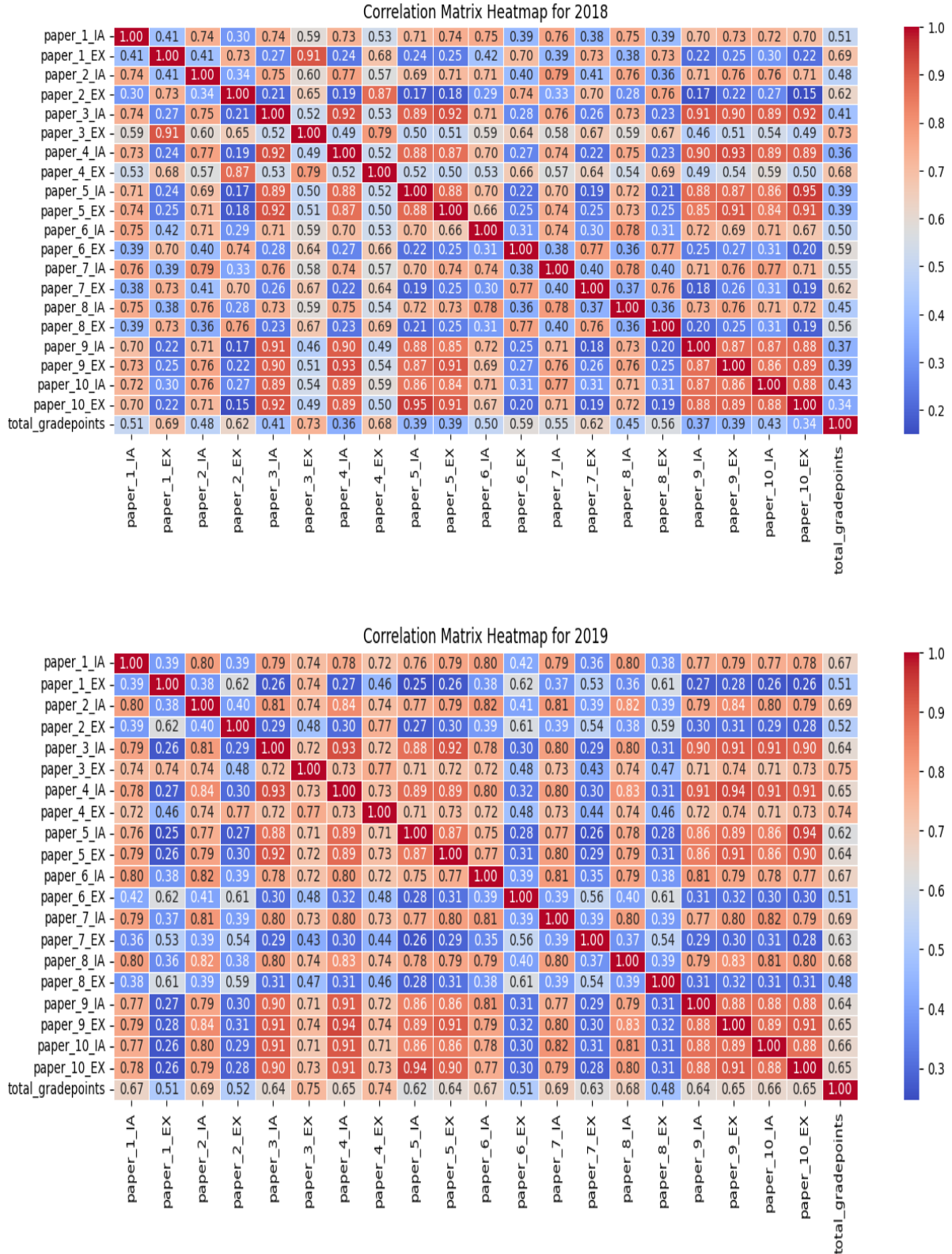
### STATISTICAL ANALYSIS ON NUMERICAL DATA

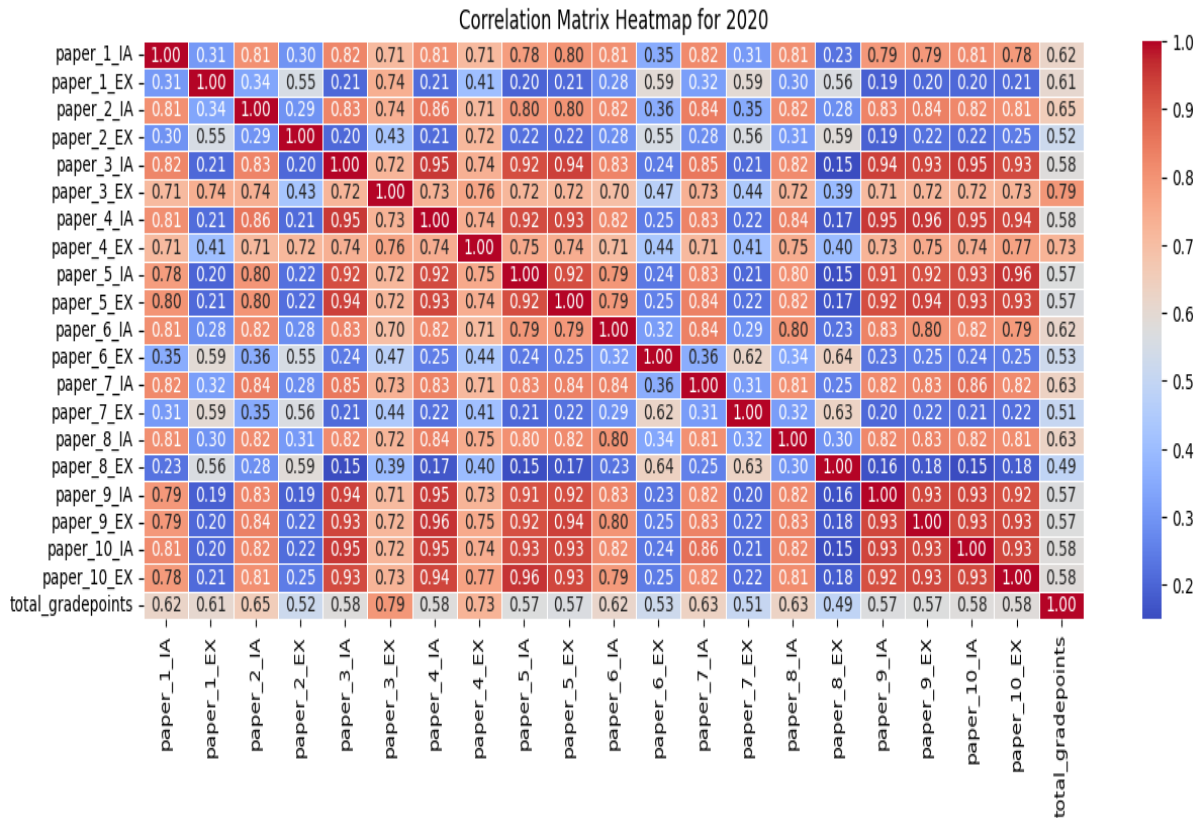
#### 1. **Correlation matrix and Heat Map**

To create a correlation matrix for numerical data, we followed these organized steps to simplify and gain meaningful insights from the process:

- a. **Data Segmentation:** Entire dataset was divided into three subsets based on the categorical column "year\_of\_admission," which contained three distinct values: 2018, 2019, and 2020. This allowed us to analyse the correlations within each admission year separately, potentially revealing any year-specific trends or patterns.
- b. **Identifying Numerical Columns:** From each subset, all the numerical columns, which contain quantitative data such as marks in various subjects and the total grade point of each student was identified. By isolating these columns, we focused solely on the relevant variables for the correlation analysis.
- c. **Creating the Correlation Matrix:** Correlation matrix was created using the numerical data arrays. The correlation matrix provided us with a comprehensive view of how the marks in different subjects and the total grade point relate to each other.
- d. **Visualizing with Heatmap:** Due to the potentially large number of values in the correlation matrix, understanding the patterns directly from the matrix could be challenging. To overcome this, we opted to visualize the correlation matrix using a heatmap. The heatmap effectively represented the correlations with colours, making it easier to identify strong and weak relationships between variables. Darker colours indicated stronger correlations, while lighter colours represented weaker or negligible correlations.

The maps created for each year is as follows:





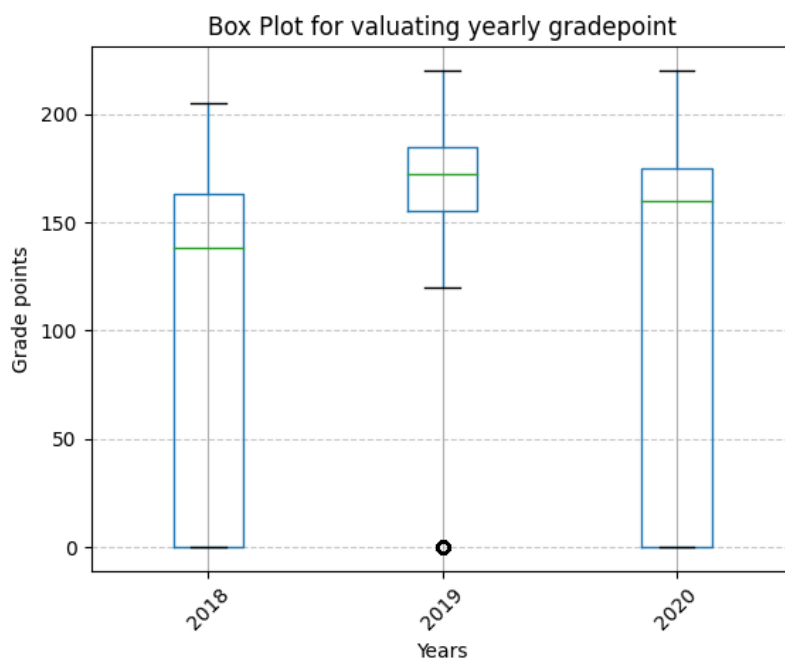
Based on the correlation matrices represented in the maps for each year, several meaningful observations can be made:

- In 2018, the majority of the correlation values lie between 0.2 to 0.5, indicating a relatively weak overall performance of students during that year. However, in both 2019 and 2020, the correlation values increase and fall within the range of 0.7 to 0.9. This suggests a consistent improvement in the academic performance of students each year, with 2020 showing the highest correlation values.
- The maps reveal that orange clusters consistently appear for papers 3, 9, and 10 across all three years. This indicates that students performed well in these subjects, and they contributed significantly to their overall high performance. Additionally, for the years 2019 and 2020, clusters also emerge for papers 4 and 5, suggesting a notable growth in students' proficiency in these specific subjects.
- Blue clusters are noticeable for external assessment papers, namely papers 1, 2, 6, 7, and 8. This indicates that a substantial number of students scored lower marks in these external assessment subjects. These clusters highlight areas where students may need further support or improvement.

Visual analysis of the correlation matrices provides valuable insights into the academic progress of students over the years, identifying areas of improvement and subjects where students excel or face challenges. The trends observed in the maps underscore the overall positive trajectory in students' academic performance while pointing out specific subjects that warrant attention for better outcomes.

## 2. **Box Plot**

The same subset used for the correlation matrix was employed to generate box plots, offering visual insights into the data's distribution and variations. Box plots aid in comparing the performance of students in different years and subjects, highlighting patterns, trends, and potential outliers.



The box plots offer comprehensive insights into the trends and variations in students' academic performance over the three years, emphasizing areas of improvement, challenges, and the impact of outliers on the data distribution.

The analysis of the box plots provides detailed observations for each academic year:

- In 2018, the majority of marks are concentrated between 0 and 170, with approximately 50% of students scoring below 150, indicating a high proportion of academic struggles. The highest mark in this year is only slightly above 200, suggesting limited exceptional performance.

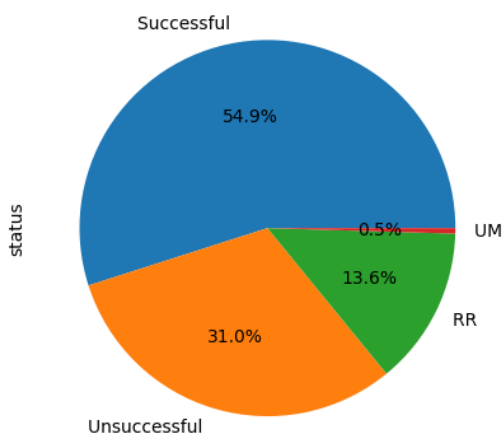
- In 2019, a noticeable improvement is evident, with marks clustering between 160 and 180, reflecting an overall enhancement in students' academic performance. However, a black dot as an outlier indicates a group of students who scored significantly poorer marks than the majority, warranting further investigation into their performance. The top mark in 2019 surpasses the best score achieved in 2018.
- In 2020, the top mark decreases compared to 2019, signifying a slight decline in academic achievement compared to the previous year. Furthermore, a direct comparison of individual marks indicates an overall dip in students' academic performance compared to the peak observed in 2019.

## VISUALIZATIONS ON CATEGORICAL DATA

Visualizations were basically done on the basis of frequency counts taken for the categorical data. This means that frequency of each value in categorical data is taken and compared to get idea how these values are distributed. Here categorical variables status and gender and these variables are used to create visualisations

### 1. Pie Chart

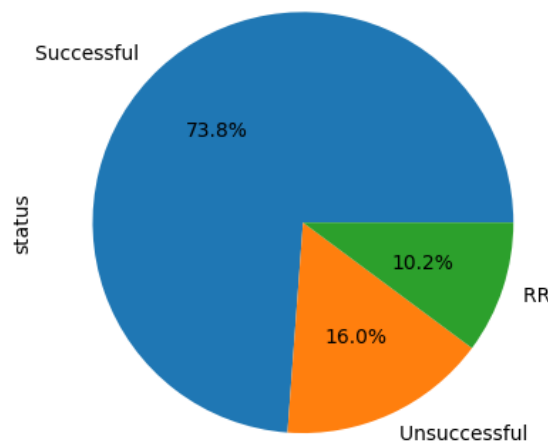
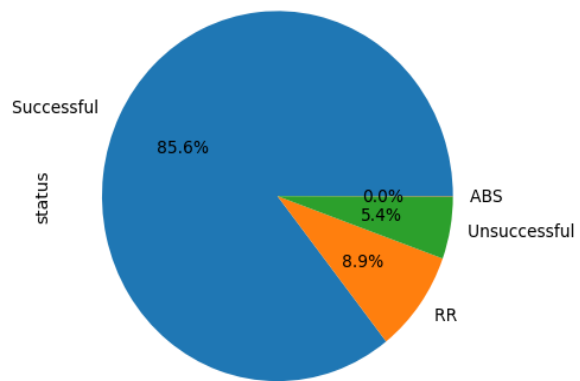
To create pie chart, we have taken column status to determine number of students passed or failed in each year.



According to pie chart illustrating the success rate in the academic year 2018, a mere 54.9% of students managed to pass, signifying a relatively low success rate. Surprisingly, almost half of the students (approximately 50%) experienced complete failure during the year. Among the students who were not successful, a notable 13.6% belonged to the reserved category (status RR), indicating they faced challenges in either theory or practical exams. Overall, the performance of students in the year 2018 appears to be significantly below expectations.

The pie chart illustrates the success rate in the academic year 2019, indicating a substantial improvement compared to 2018. Impressively, 85.6% of students successfully passed their exams, demonstrating significant academic achievement.

Furthermore, the number of students in status RR, representing those who faced challenges in either theory or practical exams, notably decreased to 8.9%. This decline suggests an overall enhancement in students' performance and academic outcomes in the year 2019.



2020, there was a noticeable decline in the success rate, which dropped to 73.8%, indicating a reduction compared to the previous year, 2019. Upon examining the group of unsuccessful students, it was observed that the number of students with status RR, meaning they failed either the theory or practical paper, increased. This

observation implies that one of the papers, presumably one of the toughest, remains challenging for students to pass.

#### 4.5 DATA ENCODING

Categorical conversion includes converting the categorical data into numerical data. This involves two different steps: Two techniques used for encoding are

1. One-hot encoding
2. Label encoding

The data set contain two sets of categorical data which are 'status' and 'gender'. Two categories hold different features so both one-hot encoding and label encoding was used in the dataset

#### 4.5.1 ONE\_HOT ENCODING

The gender column underwent one-hot encoding because this method provides binary outputs, specifically 0 and 1. Given that the gender column contains only two unique values - male and female - one-hot encoding effectively transformed this categorical data into a binary representation. This technique is well-suited for cases where there are only two possible categories, making it easier for machine learning algorithms to process the data in a numerical format.

	F	M
seat_no		
5201541	0	1
5201542	0	1
5201543	0	1
5201544	0	1
5201545	0	1

The encoding process involved analysing the 'gender' column, which contained values 'M' and 'F'. Two new columns were created with the names 'F' and 'M'. For each row in the dataset, if the 'gender' value was 'M', a '1' was marked in the 'M' column, and if the value was 'F', a '1' was marked in the 'F' column. If the 'gender' value was neither 'M' nor 'F', '0' was marked in both new columns. This encoding technique efficiently converted the categorical 'gender' data into a binary representation.

#### 4.5.2 LABEL ENCODING

Label encoding was chosen for the 'status' column as it contained multiple distinct values. By using label encoding, each unique status was assigned an integer value, making it a suitable choice for transforming this categorical data into a numerical representation. Unlike one-hot encoding, which creates binary columns for each category, label encoding simplifies the data by representing various status categories with integer labels.



Successful	4200	2	4200
RR	484	1	484
Unsuccessful	465	4	465
UM	1	3	1
ABS	1	0	1
Name: status, dtype: int64		Name: Encoded_status, dtype: int64	

The encoding process involved analysing the 'status' column, which contained five distinct values. Each of these values was encoded with a unique integer, resulting in the outcome described above. The encoding assigned specific integer labels to represent each distinct status value. This approach allows for efficient numerical representation of the categorical data, facilitating the use of machine learning algorithms that require numerical inputs.

Distinct values	Encoded Values
ABS	0
RR	1
Successful	2
UM	3
Unsuccessful	4

Note: Both status column and gender column were dropped after encoding was performed.

The encoding above converted the categorical data into corresponding numerical data and now these datasets can be used for model creation and accuracy testing.

## 4.6 ML MODELING AND ACCURACY TESTING

The dataset encoded can be used for testing the Machine learning model. The steps involved in this process are:

- Splitting data
- Model Creation and Training

### 4.6.1 SPLITTING DATA

Splitting data is one of the important steps before we directly introduce dataset to the model. This includes selecting independent variables and dependent variables to create model. We have chosen Encoded\_status as independent or target variable because ultimate aim is to know the status of students in their exam.



```
[12] # splitting data
      X = marksheet.drop('Encoded_status',axis=1)
      y = marksheet['Encoded_status']
```

After splitting the data, the dataset should be converted into arrays before we give dataset to the model. So, the dataset was converted into arrays as follows:

```
[13] X = X.values
      y = y.values
```

After converting dataset to array, they are divided into following ways:

1. Training Dataset (60%)
2. Validation Dataset (20%)
3. Testing Dataset (20%)

```
[14] # Split the data into training (60%), validation (20%), and test (20%) sets
      X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.4, random_state=42)
      X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)
```

Function of each line are:

1. `train_test_split (X, y, test_size=0.4, random_state=42):`  
 This line splits the original dataset X (features) and y (labels/targets) into two sets: X\_train and X\_temp for features, and y\_train and y\_temp for labels. The test\_size parameter is set to 0.4, meaning that 40% of the data will be used for testing, and the remaining 60% will be used for training. The random\_state parameter ensures reproducibility in the data split, so the same random split can be obtained each time the code is run.
2. `train_test_split (X_temp, y_temp, test_size=0.5, random_state=42):`  
 This line further splits the temporary dataset X\_temp and y\_temp (created from the first split) into two sets: X\_val and X\_test for features, and y\_val and y\_test for labels. The test\_size parameter is set to 0.5, meaning that 50% of the temporary data (which is 20% of the original data) will be used for validation, and the remaining 50% will be used for final testing.

At the end of the code execution, you will have the following datasets:

- X\_train: Features for training the machine learning model (60% of the original data).

- `y_train`: Corresponding labels/targets for training (60% of the original data).
- `X_val`: Features for validation (20% of the original data).
- `y_val`: Corresponding labels/targets for validation (20% of the original data).
- `X_test`: Features for final testing (20% of the original data).
- `y_test`: Corresponding labels/targets for final testing (20% of the original data).

The resulting dataset is used in machine learning models to train and test data.

#### 4.6.2 MODEL CREATION AND TRAINING

After splitting the data, they are introduced to the model. The following are the model experiments performed. Basic steps involved in creating, training and testing a model are:

1. Creating model classifier with hyper parameters
2. Training the model by finding optimal data splits to create a decision tree for predictions using `fit()`.
3. The trained model is then used to predict labels on the testing data. To get accuracy score, we use `accuracy_score()`.
4. Evaluate model performance by creating confusion matrix and classification report using `confusion_matrix()` and `classification_report()` respectively.

#### 1. DECISSION TREE

##### Implementation

```
# Train Decision Tree Model

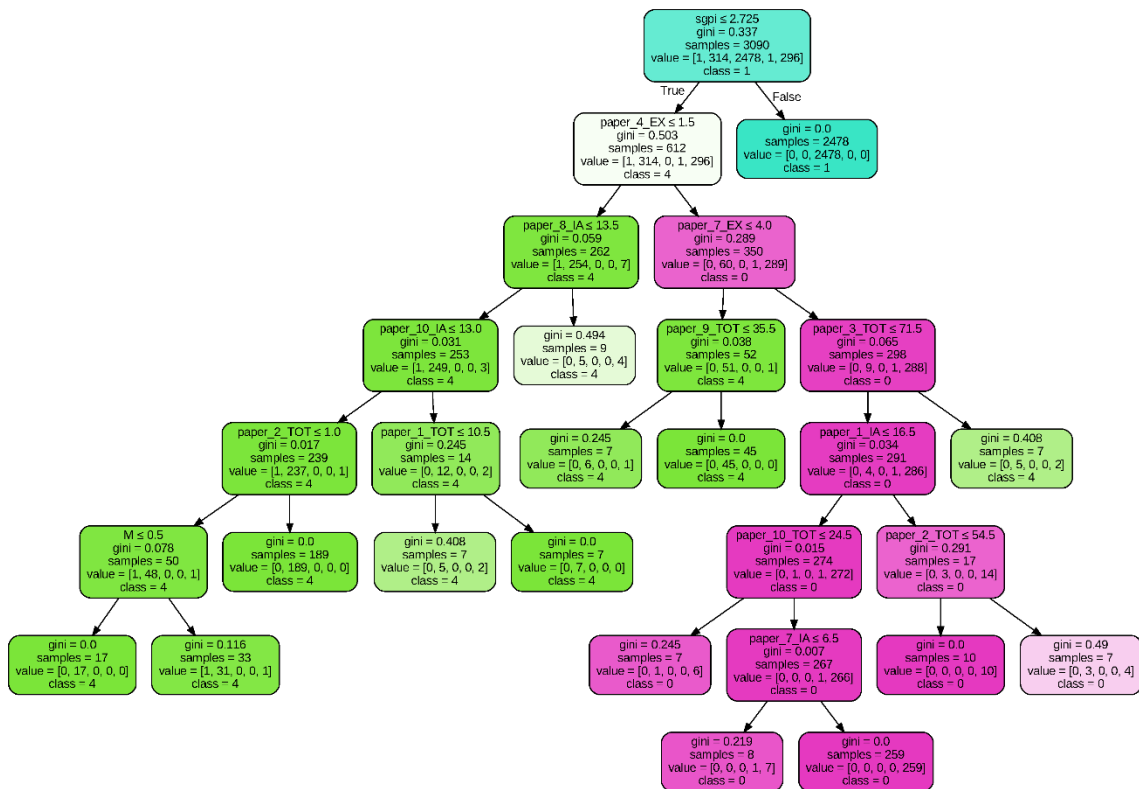
# Create the decision tree model
model = DecisionTreeClassifier(min_samples_split=12, min_samples_leaf=7)

# Train the model on the training data
model.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
confusion_mat = confusion_matrix(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)
```

Visualization of the decision tree created is as follows:




The hyper parameters used in decision tree classifier are `min_samples_split=12` and `min_sample_leaf=7` which controls the minimum number of samples required to split an internal node and the minimum number of samples required to be at a leaf node, respectively.

Reasons to choose Decision tree as one of the models is because it is easy to understand and interpret due to their visual nature and simple decision rules and require little data preparation compared to other models. They can also capture non-parametric model, meaning they can capture complex relationships between features.

## 2. RANDOM FOREST

### Implementation

```
 # Create the Random Forest model
rf_model = RandomForestClassifier(n_estimators=100, max_depth=5)

# Train the model on the training data
rf_model.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = rf_model.predict(X_test)


# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
confusion_mat_rf = confusion_matrix(y_test, y_pred)
classification_rep_rf = classification_report(y_test, y_pred)
```

The hyper parameters used in random forest classifier are `n_estimators=100`, `max_depth=5` giving a random forest consists of 100 individual decision trees, and each of these trees has a maximum depth of 5 levels. By combining 100 decision trees with a maximum depth of 5, the Random Forest aims to strike a balance between capturing complex relationships in the data and avoiding overfitting, resulting in a model that generalizes well to new, unseen data.

Reason to choose random forest is that it has high accuracy in classification tasks. It reduces overfitting by combining multiple decision trees and handles high-dimensional data well. Feature importance helps identify relevant features. It requires no assumptions on data distribution, making it flexible. It resists overfitting and can be parallelized for faster training. As an ensemble method, it leverages multiple models for improved performance. Despite being easy to implement, interpreting the complete Random Forest may be challenging. Nevertheless, it remains a popular and powerful choice for various real-world applications in machine learning.

### 3. GRADIENT BOOSTING MACHINES (GBM)

#### Implementation

```
 # Train GBM Model

gbm_model = GradientBoostingClassifier(n_estimators=100, max_depth=5)

# Train the model on the training data
gbm_model.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = gbm_model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
confusion_mat_gbm = confusion_matrix(y_test, y_pred)
classification_gbm = classification_report(y_test, y_pred)
```

The hyper parameters used in decision tree classifier are `n_estimators=100`, `max_depth=5` as same as in random forest model as they strike a balance between capturing complex patterns in the data while avoiding overfitting. These hyperparameter values can be adjusted based on the problem and dataset characteristics to optimize the model's performance. Hyperparameter tuning is crucial to finding the best combination of settings for the specific task and achieving the best possible model performance.

Reason to choose GBM is for its high predictive accuracy, ability to handle complex relationships in data, feature importance estimation, and robustness to outliers. It can effectively model both regression and classification tasks, making it suitable for a wide range of real-world applications.

## CHAPTER 5

### RESULTS

The visual analysis of the correlation matrices for each year yields valuable insights into students' academic performance trends. The correlation values in 2018 fall between 0.2 to 0.5, indicating weak overall performance. However, in 2019 and 2020, the values increase to 0.7 to 0.9, suggesting consistent improvement. Notably, orange clusters are observed across all years for papers 3, 9, and 10, indicating strong performance. In 2019 and 2020, clusters also emerge for papers 4 and 5, suggesting growth in proficiency. Conversely, blue clusters appear for external assessment papers 1, 2, 6, 7, and 8, indicating lower marks. The analysis provides valuable insights, identifying areas of improvement and subjects needing attention for better outcomes. These findings inform evidence-based decisions for educators and policymakers to enhance students' academic experience and performance continually.

The box plots offer valuable insights into students' academic performance trends over three years. In 2018, the majority of marks clustered between 0 and 170, with around 50% of students scoring below 150, indicating significant academic struggles. Exceptional performance was limited, with the highest mark just slightly above 200. In 2019, there was a notable improvement, with marks concentrated between 160 and 180, reflecting an overall enhancement in academic performance. However, a black dot outlier highlighted a subgroup of students who scored much lower marks. Despite this, the top mark in 2019 exceeded the previous year's best score, indicating progress. In 2020, the top mark decreased compared to 2019, hinting at a slight decline in academic achievement. An overall dip in students' performance compared to the peak in 2019 was also evident. The analysis underscores the need to address challenges and sustain improvements in students' academic journey.

The pie charts present a comprehensive overview of students' success rates for the academic years 2018, 2019, and 2020. In 2018, a mere 54.9% of students passed, with nearly half experiencing complete failure. Among the unsuccessful students, 13.6% belonged to the reserved category (status RR), indicating challenges in theory or practical exams. In contrast, the year 2019 showed a significant improvement, with an impressive 85.6% of students passing, and the number of students with status RR decreased to 8.9%. However, in 2020, the success rate dropped to 73.8%, and the number of students with status RR increased, suggesting ongoing challenges in passing a challenging paper. These findings highlight

fluctuations in students' academic performance over the years, emphasizing the importance of continuous efforts to enhance outcomes and address obstacles for a successful educational journey. This analysis helped in understanding patterns in data helped in making right decisions in choosing ML model type, problem type, encoding methods and ML models.

The accuracy values for each model were computed during the model creation and training process. As shown in the code snippets, the accuracy for each model was obtained using evaluation metrics like `accuracy_score()` from `scikit-learn`.

After calculating the accuracy for each model, the results were tabulated to provide a clear comparison of their performance. The table below shows us the comparison of the accuracy of the Decision Tree, Random Forest, and Gradient Boosting Machine (GBM) models.

MODEL NAME	TRAINING ACCURACY	VALIDATION ACCURACY	MODEL ACCURACY
Decision Tree	0.9948220064724919	0.9922330097087378	0.9922405431619786
Random Forest	0.9967637540453075	0.9941747572815534	0.9922405431619786
Gradient Boosting Machine	0.9996763754045307	0.9970873786407767	0.9941804073714839

We can see that all the models where the encoded data was used for training and testing gives model accuracies of 0.99, which means model achieved an accuracy of approximately 99.22% on the dataset used for testing.

An accuracy of 99.22% indicates that these models correctly predicted the target variable for about 99.22% of the instances in the test dataset. The higher the accuracy, the better the model's performance in making correct predictions.

But we can see that, numerical values accuracy values are similar for all three models. This can be because of following reasons:

- The three models are trained and tested on the same dataset, and the data distribution is such that all models perform equally well and achieve the same accuracy.

- The hyperparameter settings for the three models are the similar, same for decision tree and random forest, leading to similar model behaviour and performance.
- The complexity of the three models is similar, and they have similar capacity to learn from the data, resulting in comparable accuracy.
- Random Forest and GBM, use randomness in their training process. If the random seeds used are the same, the models might produce the same results.

Having the same accuracy values for multiple models is not necessarily a problem. It simply means that all three models are performing equally well on the given dataset.

In addition to accuracy testing, additional accuracy tests were performed on each model to evaluate the efficiency of model after fitting encoded data. The tests conducted on dataset was confusion matrix and classification report using `confusion_matrix()` and `classification_report()` respectively.

The table below shows the accuracy test conducted and results

MODEL NAMES	CONFUSION MATRIX	CLASSIFICATION REPORT				
		CLASS	PRECISION	RECALL	F1-SCORE	SUPPORT
Decision Tree	[[ 98 0 2]	1	0.94	0.98	0.96	100
	[ 0 850 0]	2	1.00	1.00	1.00	850
	[ 6 0 75]]	3	0.97	0.93	0.95	81
Random Forest	[ [ 98 0 2]	1	0.95	0.99	0.97	100
	[ 0 850 0]	2	1.00	1.00	01.00	850
	[ 4 0 77]]	3	0.99	0.94	0.96	81
Gradient Boosting Machines	[[ 97 0 3]	1	0.97	0.97	0.97	100
	[ 0 850 0]	2	1.00	1.00	1.00	850
	[ 5 0 76]]	3	0.96	0.96	0.96	81



The table presents the evaluation results for three different machine learning models—Decision Tree, Random Forest, and Gradient Boosting Machines—used for a classification task. The models are evaluated using a confusion matrix and various performance metrics like precision, recall, F1-score, and support for each class (1, 2, and 3).

Analysis the metrics for each model is as follows:

- Decision Tree:
  - Precision: For class 1, the model has a precision of 0.94, which means 94% of the instances predicted as class 1 are correct. For class 2, the precision is 1.00, indicating a perfect prediction. For class 3, the precision is 0.97, meaning 97% of the instances predicted as class 3 are correct.
  - Recall: Also known as the true positive rate, recall measures the percentage of actual instances of a class that the model correctly predicts. The values for all classes are very high, indicating that the model can identify instances of each class well.
  - F1-score: The F1-score is the harmonic mean of precision and recall. It balances both metrics and provides an overall evaluation of the model's performance. The F1-scores for all classes are high, showing a good balance between precision and recall.
  - Support: The support is the number of instances in each class. There are 100 instances of class 1, 850 instances of class 2, and 81 instances of class 3 in the dataset.
- Random Forest:
  - The performance metrics for Random Forest are quite similar to those of the Decision Tree. The precision, recall, and F1-scores for all classes are consistently high, indicating that the Random Forest model is performing well in predicting all three classes.
  - The confusion matrix shows a few differences in the number of misclassifications compared to the Decision Tree, but overall, it's performing well.
- Gradient Boosting Machines:
  - The Gradient Boosting Machines model also shows impressive performance metrics. The precision, recall, and F1-scores are all very high for all classes.

- The confusion matrix indicates slightly different numbers of misclassifications compared to the previous models, but the overall performance is strong.

The high accuracy and strong performance of Decision Tree, Random Forest, and Gradient Boosting Machines on model accuracy tests, confusion matrices, and classification reports demonstrate the power of encoding methods in handling categorical data. Categorical variables pose a challenge in machine learning, but encoding techniques transform them into numerical representations, enabling algorithms to process and learn from the data effectively.

The data analysis conducted provided valuable insights into students' academic performance trends and subject correlations. By understanding the patterns and variations in the data, we made informed decisions on data preprocessing and encoding methods. The identification of subjects with consistent high performance guided the selection of appropriate encoding strategies. Addressing imbalances and challenges revealed by the analysis helped mitigate bias and improve model generalization. Additionally, the correlation analysis provided valuable feature selection cues, contributing to a more focused and effective model. These data-driven decisions enhanced the accuracy of the machine learning models, resulting in higher prediction performance and better overall accuracy scores.

## CHAPTER 6

### CONCLUSION AND FUTURE WORK

The analysis of heterogeneous data in this study posed significant challenges due to the presence of mixed data types. However, through meticulous preprocessing, including null value checks, duplicate handling, and outlier identification using custom imputation and frequency checks, the dataset was prepared for in-depth analysis.

For numerical data analysis, the dataset was segmented based on the categorical variable "year\_of\_admission." Subsequently, a correlation matrix was constructed to explore the relationships between variables. To enhance the interpretation of the matrix, heatmaps were employed, enabling the identification of subject areas where students excelled or faced difficulties in achieving high marks.

Furthermore, box plots were utilized to assess data distribution and variation in overall student results across different years. These analyses provided valuable insights into the academic performance trends, identifying years of notable progress, instances of academic challenges, and the percentage of students scoring above or below average in each year.

Addressing categorical data analysis, pie charts were utilized to visualize the success rates in each year, facilitating a comprehensive comparison of success rates among the three years.

Following the thorough data analysis, appropriate encoding methods, including one-hot encoding and label encoding, were selected based on the distribution of categorical data. These techniques effectively transformed the data for integration with three machine learning models: Decision Tree, Random Forest, and Gradient Boosting Machines. The outcomes demonstrated remarkably high accuracy levels, reaching 99%.

Moreover, additional accuracy testing, such as confusion matrix and classification report evaluations, further affirmed the effectiveness of encoding techniques in successfully handling categorical data.

In conclusion, this study provides compelling evidence of the prowess of encoding methods in managing heterogeneous data, bridging the gap between numerical and categorical data types. The comprehensive analysis and meticulous preprocessing have significantly contributed to the robustness of the results. The remarkable accuracy achieved by the machine learning models substantiates the practical significance of this research and offers

valuable implications for similar studies dealing with mixed data types. Further accuracy testing like confusion matrix and classification report was conducted on the model giving high accurate results. These results interpret the positive effect of encoding techniques in handling categorical data.

## FUTURE WORK

In this thesis, we have focused on exploring heterogeneous data, which refers to data that contains various types of variables, such as categorical and numerical. The main objective was to analyse and understand the challenges and complexities associated with handling such diverse data types within a unified framework. During the research, we investigated methods to effectively analyse heterogeneous data, including techniques for data preprocessing, feature engineering, and dimensionality reduction. We addressed the significance of handling categorical data appropriately, as it requires special treatment in machine learning models due to its non-numeric nature.

Furthermore, we evaluated various encoding techniques for categorical data, such as one-hot encoding and label encoding, and tested their effectiveness in machine learning models. The goal was to determine which approach best preserves the information contained in the categorical features and leads to improved model performance.

As we conclude this study, several potential areas for future work arise. First, exploring more advanced encoding methods for categorical data, such as target encoding or entity embedding, could offer further insights into enhancing model accuracy and generalization.

Secondly, investigating ensemble techniques that combine the strengths of multiple machine learning models could lead to improved predictive performance on heterogeneous datasets, potentially mitigating overfitting and handling outliers effectively.

Moreover, conducting experiments on larger and more diverse datasets from different domains would strengthen the generalizability of the findings and provide a more comprehensive understanding of how to handle heterogeneous data effectively in practical scenarios.

Lastly, integrating deep learning approaches into the analysis of heterogeneous data could unlock new possibilities for extracting complex patterns and relationships, potentially

revolutionizing the way we handle and leverage diverse data types in machine learning applications.

In conclusion, this thesis has laid the groundwork for understanding and managing heterogeneous data with a focus on categorical variables. However, exploring the above-mentioned future research directions would further advance the field of data analysis and contribute to the development of more robust and accurate machine learning models for heterogeneous datasets.

## REFERENCES

- [1] Krzysztof J. Cios, G. William Moore, Uniqueness of medical data mining, *Artificial Intelligence in Medicine* 26, 1–24, 2002.
- [2] Processing” *EURASIP Journal on Advances in Signal Processing* (2016) 2016:67
- [3] Dr. Poornima Nataraja, Bharathi Ramesh “machine learning algorithms for heterogeneous data: a comparative study”
- [4] Junfei Qiu, Qihui Wu, Guoru Ding\*, Yuhua Xu and Shuo Feng “A survey of machine learning for big data
- [5] Kedar Potdar, Taher S. Pardawala,, Chinmay D. Pai , “A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers”
- [6] Vijay Kotu, Bala Deshpande “Chapter 3, Data Exploration, Categorical Data”
- [7] Kiran Maharana, Surajit Mondal, Bhushankumar Nemade “A review: Data pre-processing and data augmentation techniques”.
- [8] Kavitha Srinivas, Abraham Gale, Julian Dolby, “Merging datasets through deep learning”.
- [9] Ga Young Lee, Lubna Alzamil, Bakhtiyar Doskenov, Arash Termehchy, “A Survey on Data Cleaning Methods for Improved Machine Learning Model Performance”
- [10] Lili Zhang; Yuxiang Xie; Luan Xidao; Xin Zhang “Multi-source heterogeneous data fusion”
- [11] Wenli Zhang and Sudha Ram “A Comprehensive Analysis of Triggers and Risk Factors for Asthma Based on Machine Learning and Large Heterogeneous Data Sources”.
- [12] P. Zadrozny and R. Kodali, “Big Data Analytics using Splunk, Berkeley, CA, USA: Apress,
- [13] Ricardo G. Mantovani and André C. P. L. F. de Carvalho., "Data Preprocessing Techniques for Classification without Discrimination".
- [14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, Gradient-Based Learning Applied to Document Recognition.

- [15] J. R. Quinlan, "Induction of Decision Trees".
- [16] Souhaib Ben Taieb and Rob J. Hyndman, "Elementary Applications of Random Forests".
- [17] Shalini Gupta, K. P. S. Rana, and Mukesh Saraswa, "A comparison of twelve algorithms for the imputation of missing values in univariate time series"
- [18] Erdem Varol and Joachim M. Buhmann, "Categorical Feature Encoding for Ordinal Regression"
- [19] Guozhu Dong and Huan Liu, "Feature Engineering for Machine Learning: A Review".
- [20] Ruslan Baiburin and Vasil Khalidov, "Encoding Categorical Variables: Label vs. One-Hot vs. Ordinal Encoding".

## APPENDIX

The code corresponding to the thesis titled "comprehensive analysis and feature transformation of heterogenous data for model analysis" has been uploaded to the GitHub repository. All elements discussed in the thesis can be reviewed within the codebase. You can access the code through the following link:

[https://github.com/tejajijo/Thesis\\_CS648\\_2023](https://github.com/tejajijo/Thesis_CS648_2023)