IIT ADMISSION PREDICTION USING MACHINE LEARNING

A Dissertation Submitted to
Devi Ahilya Vishwavidyalaya, Indore (M.P.)
towards the partial fulfillment of the degree of

Master of Science
In
Applied Mathematics
(Specialization in Computer Science)

Supervised by:
Dr. Smita verma
Co-Supervised by:
Dr. Ranu Sharma

Submitted by:
Riya Wagh

DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

SHRI G.S. INSTITUTE OF TECHNOLOGY AND SCIENCE, INDORE (M.P.)

2023

SHRI G.S. INSTITUTE OF TECHNOLOGY AND SCIENCE, INDORE (M.P.)

DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

RECOMMENDATION

We are pleased to recommend that the dissertation work entitled, **IIT ADMISSION PREDICTION USING MACHINE LEARNING** submitted by Riya Wagh partial fulfillment for Degree of **Master of Science in Applied Mathematics (specialization in Computer Science)**, of Devi Ahilya Vishwavidyalaya, Indore during the year 2023.

Dr. Smita Verma
(Supervisor)
Professor and Head,
Department of Applied Mathematics and Computational Science

Dr. Anjulata Yadav
(TEPREC Member) Professor,
Department of Electronics and TC Engineering

Dr. Ranu Sharma
(Co-Supervisor)
Professor and Head,
Department of Applied Mathematics and Computational Science

Head of Department

Forwarded by
Dean Academics

SHRI G.S. INSTITUTE OF TECHNOLOGY AND SCIENCE, INDORE (M.P.)

DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCE

CERTIFICATE

This is to certify that the Dissertation work entitled **IIT ADMISSION PREDICTION USING MACHINE LEARNING** is bona fide work carried out by **Riya Wagh** towards partial fulfillment for Degree of **Master of Science in Applied Mathematics (specialization in Computer Science)**, of Devi Ahilya Vishwavidyalaya, Indore during the year 2023.

Internal Examiner	**External Examiner**
Date:	**Date:**

DECLARATION

I, **(Riya Wagh, M.Sc. Applied Mathematics, Department of Applied Mathematics and Computational Science)** declare that the dissertation **IIT Admission Prediction Using Machine Learning** is my own work conducted under the supervision of Dr. Smita Verma, Professor and Head, **Department of Applied Mathematics and Computational Science, S.G.S.I.T.S., Indore (M.P.).**

I further declare that to the best of my knowledge the dissertation work does not contain any part of any work which has been submitted for the award of any degree either in this university or in any other university/ deemed university without proper citation.

Signature of the candidate:

Name of the candidate: **Riya Wagh**

Enrollment No.: **0801MA21MS05**

Date: **31-08-2023**

ACKNOWLEDGEMENT

Completion of dissertation and writing report is satisfying event and the pleasant part is the opportunity to thank those who inspired, contributed and cooperated to it. Research in any field fetches knowledge of various people. I would acknowledge all those who have helped me by providing a fertile ground for much of development process that has followed.

I am thankful to my Supervisor **Dr. Smita Verma** Professor & Head and Co-supervisor **Dr. Ranu Sharma**, Assistant Professor, Department of Applied Mathematics and Computational Science, S.G.S.I.T.S. Indore for giving me her valuable inspiration, able guidance and untiring help which enabled me to carry out and complete this work. I am also thankful for giving support and permission to use resources of S.G.S.I.T.S. Indore.

I am grateful to **Prof. R.K. Saxena,** Director, S.G.S.I.T.S. for providing all the facilities and academic environment during the course of study.

Without the kind help and valuable suggestions of all the faculty members of the department this assignment would not have taken the shape, I thank them all. I also take this opportunity to thank my parents, friends and everyone who helped me directly or indirectly during the entire span of this project.

Riya Wagh

Enrollment No. 0801MA21MS05

Table of Contents

CHAPTER 1

INTRODUCTION

1.1 Preamble

A student's decision to enroll in a specific educational institution holds significant Importance and it is a pivot choice as it greatly impacts their career development and progression. The Indian Institute of technology – Joint Entrance Examination (IIT-JEE) is an annual academic test conducted in India. It is organized by one of the seven zonal IIT's (IIT Roorkee, IIT Kharagpur, IIT Delhi, IIT Kanpur, IIT Bombay, IIT Madras and IIT Guwahati) under the guidance of the Join Admission Board (JAB). The JEE Exam is crucial for student aspiring to gain admission into top engineering college in India. The IITs, NITs and IIITs are esteemed engineering Institute in our country. This college Prediction system suggests IIT (Indian Institute of Technology) based on a student's IIT JEE ranking.

1.2 Need of the Project

A Prediction system can be valuable for students who are seeking admission to prestigious institute like IIT. A system can help students to make decision about which college and branch will be available for their IIT-JEE ranking.

Personalized Guidance: The recommendation system employs sophisticated algorithm to provide tailored guidance to student. It leverages advanced data input and analytics

to offer customized suggestions based on a student's individual preferences like IIT-JEE Ranking, Gender, Quota, Round, Branch, Duration and Category. By meticulously considering these factors, the system can predict best course and college for student.

Time and Efforts Saving: Applying to multiple IIT's can be an arduous and time consuming process. A Prediction System can simplify the process by analyzing the student's profile and it will provide a relevant college.

Increase Chance of Admission: To take admission IITs is highly competitive, and the cutoff scores can vary across branches and colleges. A recommendation system can analysis a rank in entrance exam, and compare it with the previous year's cutoffs.

1.3 Problem Statement

Design and develop a College Prediction System that assist student in making judicious decision regarding their selection of educational institutions after appearing for the Indian Institute of Technology-Joint Entrance Examination (IIT-JEE).the system should consider the student's IIT-JEE ranking and predict suitable institute from IITs. The goal is to help student for identify and select the most appropriate engineering college in India.

For the purpose of predicting college in college prediction system, various machine learning algorithms are utilized. These algorithms tackle the challenge of selecting the most effective method to analyze their performance in terms of accuracy.

Machine Learning algorithms such as Decision Trees, Random Forest, Support Vector Machine (SVM), CatBoost, Logistic Regression and Neural Network are commonly employed in college prediction system. Each algorithm has its strengths and weakness and their suitability depends on the specific requirements and characteristics of the dataset.

To determine the most suitable algorithm for analyzing, several evaluation metrics are considered. Accuracy, Precision, Recall, F1 Score and area under the receiver operating characteristics curve (AUC-ROC) are some common metrics used to access the performance of Machine Learning Algorithm.

The choice of algorithm often involves a trade-off between accuracy and computational efficiency. Some algorithm, such as Deep Learning, Neural Network may offer high accuracy but require significant computational resources and training time. On other hand simpler algorithm like Decision Tree or Logistic Regression may have lower accuracy but offer faster computations and interpretability.

To select the best algorithm, researcher and data scientist employ techniques like cross-validation, hyper parameter tuning and comparison of performance metrics across different algorithms. This method helps to identify the algorithm that provides the optimal balance of accuracy, computational efficiency and interpretability for the specific college prediction system.

1.4 Objective

The main objective of a college prediction system is to assist students in making informed decision about their choice of educational institutions. The system in making provide personalized prediction system based on student s IIT-JEE Ranking, Quota, and other pertinent factors.

The primary goal is to help student identify and select the most suitable colleges or university that align with their individual needs and goals. The Prediction System aims to provide leverages Data Analysis, Machine Learning Algorithm and use input to generate suggestion that match the student profile criteria and preferences.

1.5 Overview of Thesis

The thesis is divided into seven chapters. **Chapter 2** deals with the proposed approach for college prediction system and also discuss various techniques of prediction system. In **Chapter 3**, Machine Learning and its types are presented. **Chapter 4** deals with Methodology. In **Chapter 5**, Data Collection and implementation of various techniques are elaborated. Testing of the procedure is done in **Chapter 6**, it also chows the final results. Conclusion and future scope are discussed in **Chapter 7**.

CHAPTER 2

PROPOSED APPROACH AND BACKGROUND STUDY

2.1. The Proposed approach of college prediction system

Data Collection: Data collection in Machine Learning (ML) is the process of gathering and organizing relevant data from various sources to train a machine learning model. This data includes input features and output labels and is essential for the model to learn patterns and make accurate predictions or decisions. The quality and quantity of the collected data significantly impact the performance and effectiveness of the machine learning model.

Data Preprocessing: Data Preprocessing involves cleaning, preparing the collected data and make it Suitable for Analysis and Modeling. We have to Handling Missing Values, Encoding Categorical Variable, Normalizing Numerical Feature and Data Formatting.

Featured Engineering: Feature Engineering includes analysis the data and adding additional features that can potentially improve the predictive performance of model. For example in this project we will include Average ranking.

2.2 College Prediction System

College Prediction systems aim to provide personalized recommendation to students regarding colleges or universities that best align with their preferences, goals, and

academic profiles. These systems utilize various techniques and algorithms to analyze student data and generate relevant suggestion. College Prediction System is a progressive area of research in Education.

2.3. Techniques used in College Prediction System

There are many techniques used in college Prediction system. Such as:

2.3.1 Regression Analysis

Regression analysis is a statistical technique used to predict a continuous outcome variable based on one or more predictor variables. In a college prediction system, regression models can be used. Linear regression, polynomial regression, or other regression variants can be used depending on the nature of the data and the relationships between variables.

2.3.2 Classification Algorithms

Classification algorithms are used when the outcome variable is categorical or discrete, such as predicting Institute, branch or Stream. Various classification algorithms can be used in college recommendation system such as: logistic regression, decision trees, random forests, support vector machines (SVM), or naive Bayes classifiers.

2.3.3 Time-Series Analysis

Time-series analysis techniques are employed when historical data is available, and predictions need to be made based on temporal patterns and trends. This can be relevant in predicting future enrollment trends, graduation rates, or academic performance over time. Time-series models like autoregressive integrated moving average (ARIMA) or exponential smoothing methods can be utilized.

2.3.4 Ensemble Methods

Ensemble methods combine the predictions of multiple models to improve prediction accuracy and robustness. Techniques like bagging, boosting, or random forests can be applied to aggregate predictions from multiple regression or classification models. Ensemble methods help to reduce bias, variance, and overfitting, leading to more reliable predictions.

2.3.5 Feature Selection Techniques

Feature selection techniques are used to identify the most relevant and informative features from the available data. By selecting the most influential features, prediction models can be simplified, improved, and made more interpretable. Feature selection techniques include methods like forward selection, backward elimination, or regularization techniques like Lasso or Ridge regression.

2.3.6 Dimensionality Reduction

Feature selection techniques are used to identify the most relevant and informative features from the available data. By selecting the most influential features, prediction models can be simplified, improved, and made more interpretable. Feature selection techniques include methods like forward selection, backward elimination, or regularization techniques like Lasso or Ridge regression.

2.3.7 Neural Network

Neural Network, a type of deep learning algorithm, are increasingly used in prediction system due to their ability to learn complex patterns and relationships in the data. They consist of interconnected nodes or neurons organized in layers, capable of capturing intricate nonlinear relationships.

2.4 Tools and Technology

In the context of building a prediction system we use several tools and technology. Here is some commonly used one:

Language Used: Python is a popular programming language known for its simplicity and versatility. It offers a wide range of libraries and frameworks for data analysis, machine learning, and predictive modeling, making it a preferred choice for many data science projects.

Interactive coding environments commonly used for ML projects:

- **Google Colab:** Google Colab is a cloud-based interactive coding environment that supports ML libraries like Scikit-learn, TensorFlow, and PyTorch. It offers free access to GPUs and TPUs, making it suitable for training deep learning models and computationally intensive tasks.
- **Visual Studio Code:** Visual Studio Code (VS Code) is a versatile code editor with extensions and plugins for ML libraries. It provides features like code completion, debugging, and integration with popular ML frameworks.
- **PyCharm:** PyCharm is a dedicated Python IDE that offers an interactive coding environment for ML projects. It provides support for ML libraries, code analysis, debugging tools, and version control integration.
- **RStudio:** RStudio is an integrated development environment for R programming language. It supports ML libraries in R, data visualization tools, and interactive coding features for statistical analysis and modeling.
- **MATLAB Live Editor:** MATLAB Live Editor combines code, visualizations, and narrative text in an interactive environment. It supports MATLAB's ML libraries and offers live execution, debugging, and interactive exploration of data and results.
- **Spyder:** Spyder is a scientific IDE for Python that supports ML libraries like Scikit-learn and TensorFlow. It provides an interactive coding environment with features such as a variable explorer, integrated plotting, and debugging capabilities.

Library used

Scikit-learn: Scikit-learn is a comprehensive machine learning library in Python. It provides a wide range of algorithms, tools, and utilities for tasks such as data preprocessing, feature selection, model training, evaluation, and more. For random forest and multi-class classification, you can utilize the following modules:

a. **sklearn.ensemble.RandomForestClassifier:** This module provides the implementation of the random forest algorithm for classification tasks.
b. **sklearn.multioutput.MultiOutputClassifier:** This module enables multi-output classification, allowing you to handle multi-class classification tasks.

NumPy: NumPy is a fundamental library for numerical computations in Python. It provides powerful N-dimensional array objects and functions for array manipulation, mathematical operations, linear algebra, and random number generation. NumPy is commonly used for handling data arrays and matrices in machine learning tasks.

Pandas: Pandas is a versatile library for data manipulation and analysis. It offers data structures like DataFrames, which allow for easy handling and preprocessing of structured data. Pandas can be used to load, clean, and transform datasets before training random forest models or performing multi-class classification.

Matplotlib: Matplotlib is a powerful library for creating visualizations in Python. It provides functions for creating various types of plots, charts, and graphs, which are

useful for analyzing data, understanding model performance, and presenting results.

Seaborn: Seaborn is a statistical data visualization library built on top of Matplotlib. It provides a high-level interface for creating attractive and informative statistical graphics. Seaborn simplifies the process of creating visualizations for exploratory data analysis, pattern discovery, and model evaluation.

CHAPTER 3

INTRODUCTION TO MACHINE LEARNING

3.1 Machine Learning

Machine Learning is a subset of Artificial Intelligence (AI). Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed. It involves the study of pattern recognition, statistical analysis, and computational learning theory.

In traditional programming, developers write explicit instructions that dictate how a computer system should perform a specific task. In contrast, in machine learning, algorithms are designed to learn from data and improve their performance through experience.

In traditional programming, developers write explicit instructions that dictate how a computer system should perform a specific task. In contrast, in machine learning, algorithms are designed to learn from data and improve their performance through experience. This process is often iterative and adaptive, allowing the algorithm to automatically adjust its parameters or rules based on the data it encounters.

Types of Machine Learning

1. Supervised learning

2. Unsupervised learning

3. Supervised Machine Learning

4. Reinforcement Learning

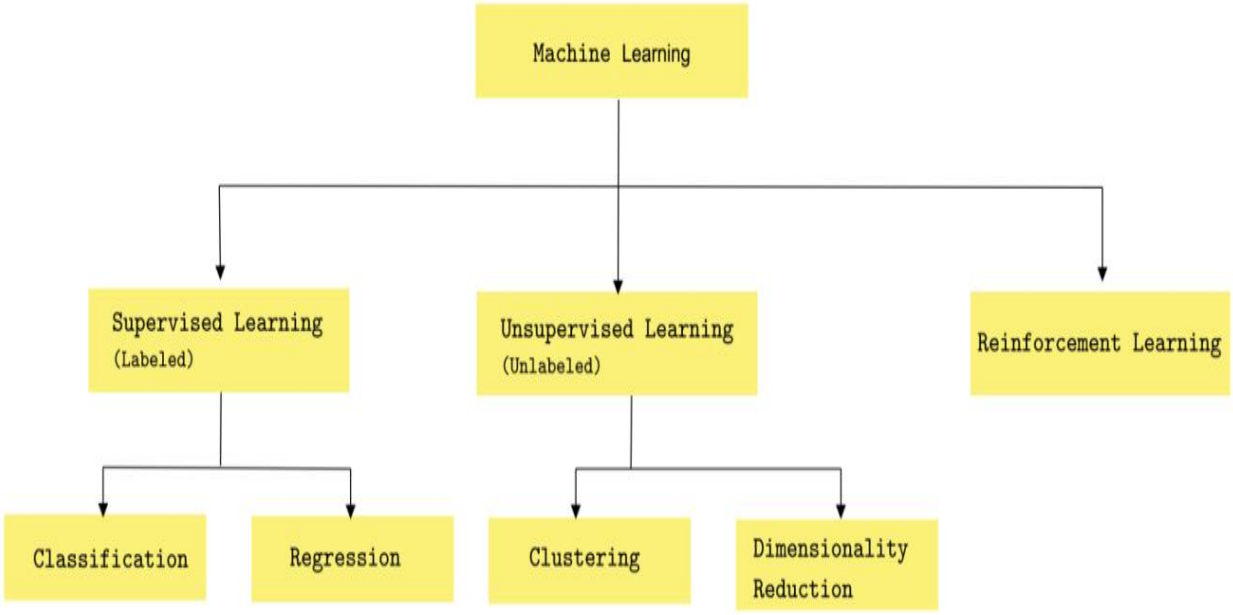

Figure: 3.1 Type of Machine Learning

3.2 Supervised Machine Learning

Supervised learning involves training a machine learning algorithm using labeled data, where each data point is associated with a known output or target value. The algorithm aim is to learn a mapping between the input features and the correct output based on the provided labels.

There are two type of supervised learning

1. Regression

2. Classification

Regression: Regression is another type of supervised learning where the goal is to predict a continuous output or target variable. The output variable in regression can take any numeric value, such as a real number or a range of values. The algorithm learns from labeled examples to predict a value based on the input features. Some popular regression algorithms include:

- **Linear Regression:** Linear regression models the relationship between the input features and the output variable as a linear equation. It finds the best-fit line or hyper plane that minimizes the sum of squared differences between the predicted and actual values.
- **Polynomial Regression:** Polynomial regression extends linear regression by including higher-order polynomial terms to capture nonlinear relationships between the features and the output variable.

Classification: Classification is a type of supervised learning where the goal is to assign input data to predefined categories or classes. The output or target variable in classification is discrete and categorical. The algorithm learns from labels. Some common algorithms used for classification include:

3.2.1 Logistic Regression

Logistic regression is a statistical model used for binary classification problems in supervised learning. The logistic regression model assumes a linear relationship between the input features and the log-odds of the input belonging to a specific class.

The log-odds, also known as the logits, are then transformed using the sigmoid function to obtain the predicted probabilities.

3.2.2 Support Vector Machines (SVM)

Support Vector Machines (SVM) is a powerful and versatile machine learning algorithm commonly used for both classification and regression tasks. SVM is particularly effective in solving complex problems with high-dimensional data. It works by constructing hyper planes or decision boundaries that best separate different classes or approximate the target variable in regression.

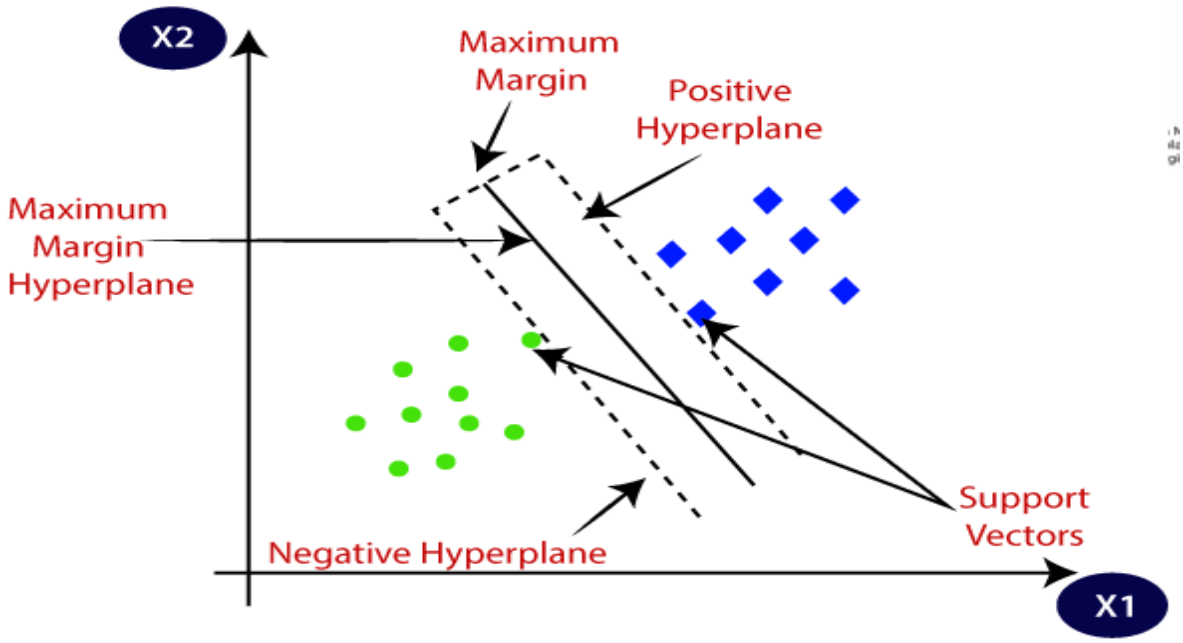

Figure 3.2. Support Vector Machine

3.2.3 Decision Trees

A decision tree is a widely used supervised learning algorithm that is particularly effective for both classification and regression tasks. It is a simple yet powerful model that can handle both categorical and continuous input features. In a decision tree, the data is split based on different features using a tree-like structure of nodes and branches. Each internal node represents a decision based on a specific feature, and each leaf node represents a class label (in classification) or a predicted value (in regression).

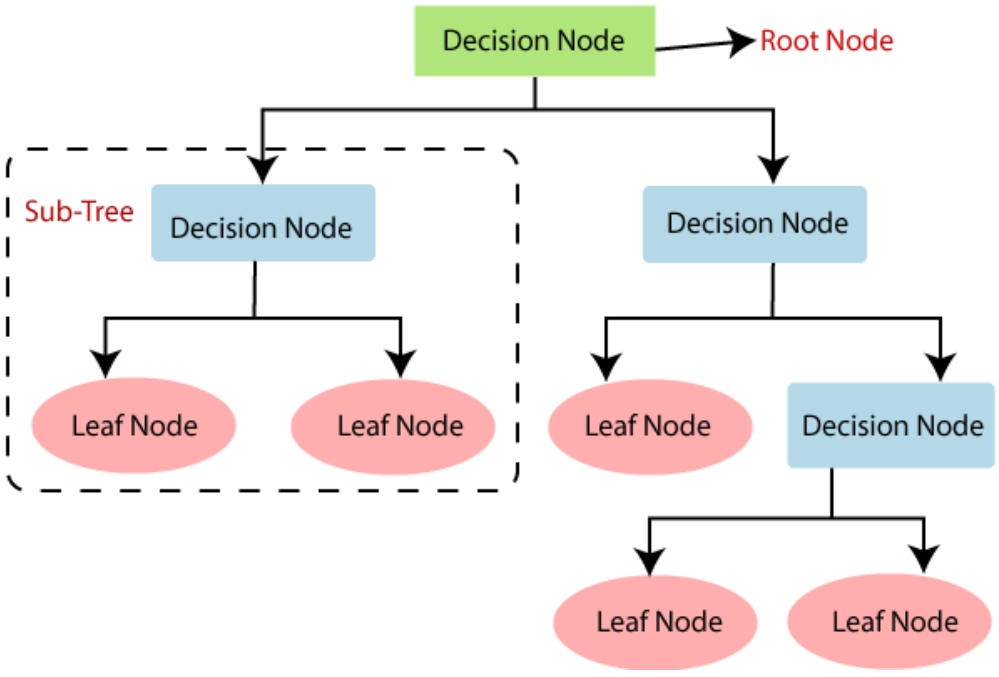

Figure: 3.3 Decision Trees

3.2.4 CatBoosting Regressor

CatBoost Regressor is a machine learning algorithm for regression tasks developed by Yandex. It is based on the gradient boosting framework and is particularly known for its ability to handle categorical features efficiently. The algorithm was designed to achieve high performance with minimal hyper parameter tuning and is popular in various data science competitions and real-world applications.

3.2.5 XGBRegressor

XGBRegressor, also known as XGBoost Regressor, is a machine learning algorithm for regression tasks developed by Tianqi Chen. It is based on the gradient boosting framework, similar to CatBoostRegressor, and is known for its high performance, scalability, and efficiency. XGBoost stands for "Extreme Gradient Boosting."

3.2.6 Adaboost Regressor

AdaBoostRegressor is a machine learning algorithm for regression tasks that belongs to the AdaBoost (Adaptive Boosting) family of algorithms. AdaBoost is an ensemble learning technique that combines the predictions of multiple weak learners (usually decision trees) to create a strong predictive model. In the context of regression, AdaBoostRegressor aims to predict continuous numeric values.

3.2.7 Random Forests

Random Forests is an ensemble learning method that combines multiple decision trees to create a robust and accurate predictive model. It is a popular and powerful machine learning algorithm that is widely used for both classification and regression tasks. In a Random Forest, an ensemble of decision trees is built, where each tree is trained on a random subset of the data and a random subset of the features. The key idea behind Random Forests is to introduce randomness in the training process to reduce over fitting and improve the model's generalization ability.

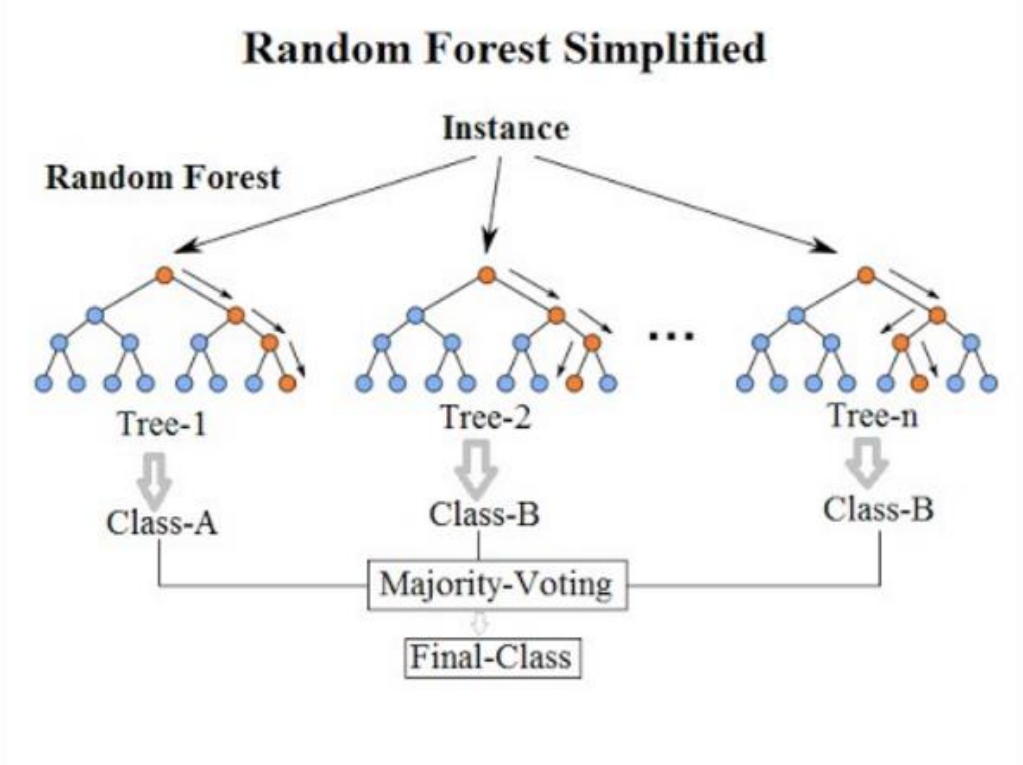

Figure 3.4 Random Forests

3.3 Unsupervised Machine Learning

Unsupervised learning is a type of machine learning paradigm where the algorithm is trained on a dataset without explicit supervision or labeled data. Unlike supervised learning, where the algorithm learns from input-output pairs, unsupervised learning focuses on finding patterns, relationships, and structures within the data without any pre-existing guidance.

The main goal of unsupervised learning is to uncover hidden structures and gain insights from the data. It is particularly useful when dealing with large, complex datasets where manually labeling each data point would be impractical or expensive. Unsupervised learning is often used for tasks such as clustering, dimensionality reduction, and anomaly detection.

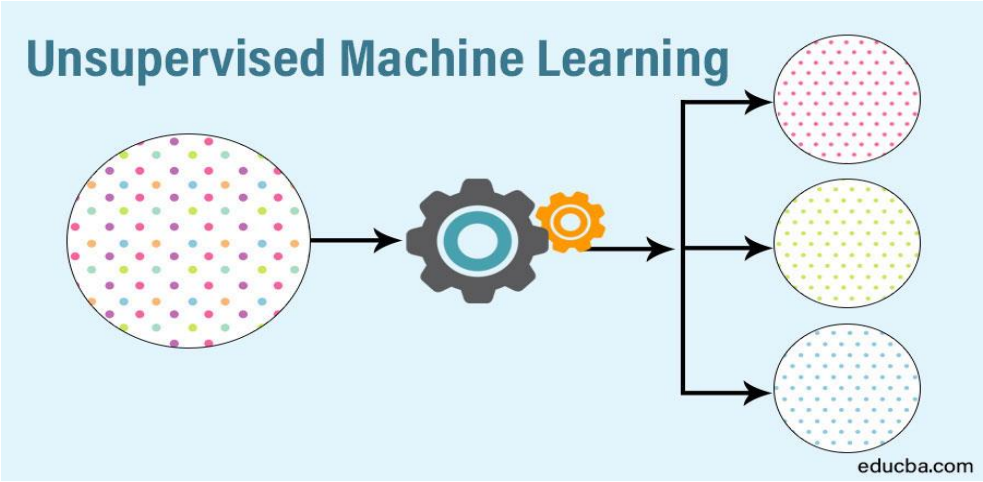

Figure 3.5 Unsupervised Machine Learning

3.4 Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning paradigm where an agent learns to make decisions and take actions in an environment to achieve a specific goal or maximize a cumulative reward over time. Unlike supervised learning, where the algorithm learns from labeled input-output pairs, and unsupervised learning, where the algorithm learns from unlabeled data, reinforcement learning involves learning from interaction with the environment.

In reinforcement learning, the agent observes the current state of the environment and takes actions based on a policy. The policy is a strategy that maps states to actions, and the goal of the agent is to learn the optimal policy that leads to the highest cumulative reward. The agent then receives feedback in the form of rewards or penalties from the environment based on the actions it takes.

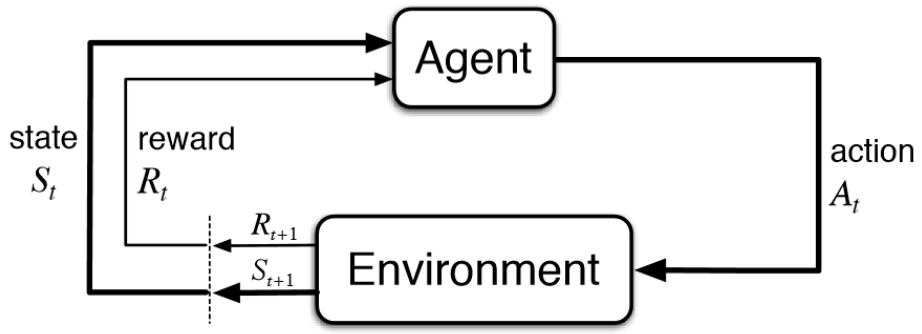

Figure 3.6 Reinforcement Learning

CHAPTER 4

METHODOLOGY

4.1 Detailed Problem Statement

In the college admission process, students often face challenges in determining their likelihood of getting accepted into their desired colleges or universities. The objective of this project is to develop a College Admission Prediction System using the Random Forest algorithm. This system will utilize historical admission data and various applicant attributes to predict the probability of admission for prospective students.

The specific problem to be addressed includes the following key aspects:

Data Collection: Gather historical data from multiple colleges and universities, including information on admitted students and their corresponding attributes. The data should include features such as Institute name, Program, Duration, Degree, Category, Pool, Opening, Closing, and Quota.

Data Preprocessing: Cleanse and preprocess the collected data to handle missing values, outliers, and inconsistencies. This may involve imputing missing data, scaling numerical features, encoding categorical variables, and normalizing data if required.

Feature Selection: Identify the most important features that contribute significantly to college admissions. Conduct exploratory data analysis, statistical tests, or employ techniques like feature importance from the model to select the relevant features.

Model Development: Train a model on the preprocessed data using the selected features as input. Optimize the model parameters, such as the number of trees, maximum depth, and minimum samples per leaf, to achieve the best performance.

Model Evaluation: Evaluate the performance of the trained model using appropriate evaluation metrics such as accuracy, precision, recall, and F1-score. Employ techniques like cross-validation or holdout validation to assess the model's ability to generalize to unseen data.

4.2 Requirement Analysis

Requirement Analysis is an essential step in the software development process. It involves identifying, analyzing the needs objectives. Here is an example of requirement analysis:

4.2.1 Functional Requirements

Functional requirements specify the functionality of a system. It depends upon the type of software, expected users and the type of system where the software used.

The major functional requirements are:

- The approach should work properly while prediction.
- The system should be able to select the better accuracy classifier for classification.
- Pre-processing of input data should be done prior to fitting data into the model.

4.2.2 Non Functional Requirements

The non-functional requirements describe how the system should behave. The non-functional requirements are as follow:

Usability: The system should be user friendly and it can be used by user without any extra efforts.

Performance: Performance requirements ensure that the system meets the expected level of efficiency and responsiveness. The system should take less time response time for every search given to it.

Reliability: Reliability requirements focus on the system's ability to function consistently and predictable over time. The system should operate without any failure.

Data Integrity: The system should be able to perform accordingly as per changes in dataset.

Accuracy: The system should able be to maximize classification accuracy.

4.3 System Requirements

System requirements specify the hardware, software, and network infrastructure needed to support the functioning of a particular system or software application. These requirements define the technical specifications and capabilities necessary for the system to operate effectively.

Following are System requirements:

4.3.1 Hardware Requirements

This includes the specifications for the physical hardware components required to run the system. It may involve details such as processor speed, memory (RAM) requirements, storage capacity, and graphics capabilities. The hardware requirements ensure that the system has the necessary resources to support its operations efficiently.

Minimum hardware requirements for the implementation for these projects are:

- RAM 8 GB or above
- Disk space 2 GB or above
- Processor Intel core i3 or above

4.3.2 Software Requirements

Software requirements for the implementation of the project are:

- Operating system Window 7 or above
- Software used: Anaconda
- Language used: python

4.4 Feasibility analysis

- All hardware and software are available.
- Domain of the project has been analysis properly.

CHAPTER 5

DATA COLLECTION AND IMPLEMENTATION

5.1. Language Used

Python is a popular programming language known for its simplicity and versatility. It is a high level programming language. It is very easy to learn and it allows writing code in very few lines. This makes python language very popular.

It offers a wide range of libraries and frameworks for data analysis, machine learning, and predictive modeling, making it a preferred choice for many data science projects.

5.2 IDE Used

Jupyter Notebook provides an interactive environment for ML projects. They support Python and various ML libraries like Scikit-learn, TensorFlow, and PyTorch. Jupyter Notebooks allow you to combine code, visualizations, and explanations, making them ideal for data exploration, model development, and documentation.

5.3 Library Used

Pandas: Pandas is an open-source Python library for data manipulation and analysis, providing powerful data structures and functions to work with structured data efficiently. It offers two primary data structures: Series, a one-dimensional labeled

array, and Data Frame, a two-dimensional labeled data structure resembling a table. Pandas are widely used in data science and data analysis tasks due to its versatility and ease of use.

Numpy: NumPy (Numerical Python) is an open-source Python library that provides support for large, multi-dimensional arrays and matrices, along with an extensive collection of mathematical functions to perform operations on these arrays efficiently. NumPy is a fundamental tool in scientific computing and data analysis in Python, and it serves as the foundation for various other libraries in the data science ecosystem. Its primary data structure is the ndarray (N-dimensional array), which allows for high-performance numerical computations and is significantly faster than native Python lists for numerical operations.

Matplotlib: Matplotlib is a popular open-source Python library used for creating static, interactive, and animated visualizations. It provides a wide range of functionalities for generating 2D plots and graphs, making it a powerful tool for data visualization and exploration.

Seaborn: Seaborn is an open-source Python data visualization library built on top of Matplotlib. It provides a high-level interface for creating informative and visually appealing statistical graphics. Seaborn is particularly useful for visualizing complex datasets and making it easier to explore and understand the relationships between variables.

Sklearn: Scikit-learn, often abbreviated as sklearn, is an open-source machine learning library for Python. It is one of the most widely used and well-regarded libraries for machine learning tasks, offering a wide range of tools for data mining, data preprocessing, model selection, evaluation, and deployment.

5.4 Implementation Detail of Proposed diagram

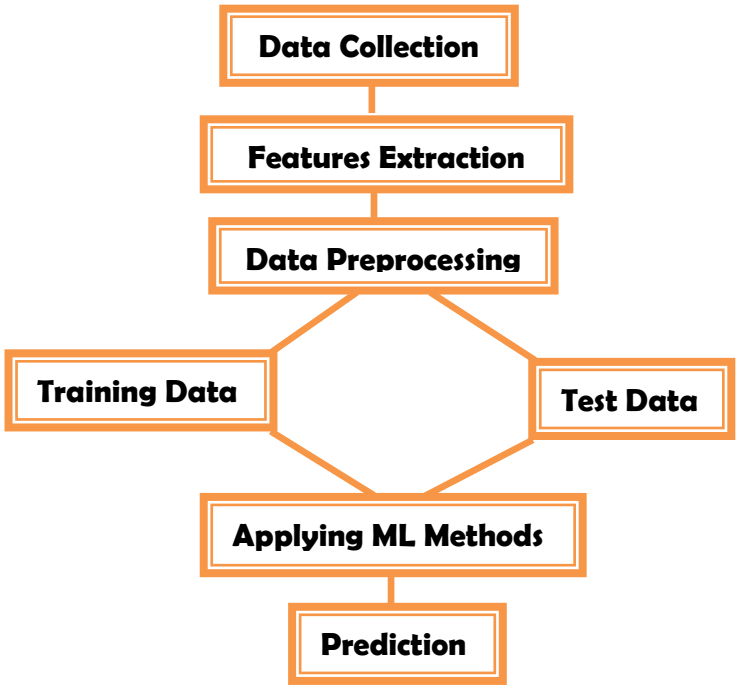

Figure 5.1 Implementation Detail of Proposed diagram

5.5 Data Collection

IITs data for the past Six years (2018-2023) is collected from Joint Seat Allocation Authority Admission website https://josaa.nic.in/ . The chosen college data is divided

into two groups training data and testing data where data from year 2018 to 2022 is selected as a training data set where year 2023 data is selected for testing.

```
In [17]: IIT.head(5)
```

Out[17]:

	Institute	Program	Quota	Seat Type	Gender	Opening	Closing	Round	Year
0	Indian Institute of Technology Bhilai	Computer Science and Engineering (4 Years, Bac...	AI	OPEN	Gender-Neutral	3702	4752	1.0	2018
1	Indian Institute of Technology Bhilai	Computer Science and Engineering (4 Years, Bac...	AI	OPEN	Female-only (including Supernumerary)	6179	7652	1.0	2018
2	Indian Institute of Technology Bhilai	Computer Science and Engineering (4 Years, Bac...	AI	OBC-NCL	Gender-Neutral	1018	1832	1.0	2018
3	Indian Institute of Technology Bhilai	Computer Science and Engineering (4 Years, Bac...	AI	OBC-NCL	Female-only (including Supernumerary)	3272	3649	1.0	2018
4	Indian Institute of Technology Bhilai	Computer Science and Engineering (4 Years, Bac...	AI	SC	Gender-Neutral	807	1051	1.0	2018

Figure 5.2 Data Collection

Our data has Nine Attributes. These named as Year, Institute, Program, Seat Type, Gender, Round, Opening, Closing and Quota.

Data set contain 82149 rows and 9 columns. In our data set 22 IITs data are present.

- 'Indian Institute of Technology Bombay',
- 'Indian Institute of Technology Kanpur',
- 'Indian Institute of Technology Kharagpur',
- 'Indian Institute of Technology Madras',
- 'Indian Institute of Technology (ISM) Dhanbad',
- 'Indian Institute of Technology Roorkee',
- 'Indian Institute of Technology (BHU) Varanasi',

- 'Indian Institute of Technology Gandhinagar',
- 'Indian Institute of Technology Hyderabad',
- 'Indian Institute of Technology Jodhpur',
- 'Indian Institute of Technology Mandi',
- 'Indian Institute of Technology Guwahati',
- 'Indian Institute of Technology Dharwad',
- 'Indian Institute of Technology Delhi',
- 'Indian Institute of Technology Bhubaneswar',
- 'Indian Institute of Technology Indore',
- 'Indian Institute of Technology Jammu',
- 'Indian Institute of Technology Palakkad',
- 'Indian Institute of Technology Ropar',
- 'Indian Institute of Technology Tirupati',
- 'Indian Institute of Technology Bhilai',
- 'Indian Institute of Technology Goa'

In Seat Type ten unique value such as:

- 'OPEN',
- 'OPEN (PwD)'
- 'EWS'
- 'OBC-NCL'
- 'OBC-NCL (PwD)',

- 'SC',
- 'ST', 'SC (PwD)'
- 'EWS (PwD)'
- 'ST (PwD)'

PWD: Person with Disability

Economically Weaker Section

OBC: Other Backward Classed

OBC-NCL: Other Backward Classes-Non Creamy Layer

SC: Scheduled Castes,

ST: Scheduled Tribes

5.6 Feature Extraction

After data collection is completed we will perform Feature Extraction. Feature extraction is a critical step in machine learning, where relevant and informative features are extracted from raw data to facilitate effective modeling and analysis. It involves transforming the original data into a reduced and more meaningful representation that captures the essential characteristics or patterns necessary for the task at hand.

In our data set we will add more features name Degree, Duration, Branch, in our data set and we take average of opening and closing rank and add new feature Avg_ranking.

```
In [18]: IIT.head(5)
```

Out[18]:

	Institute	Quota	Seat Type	Gender	Opening	Closing	Round	Year	Duration	Degree	Branch	Avg_ranking
0	Indian Institute of Technology Bombay	AI	OPEN	Gender-Neutral	1040	1974	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	4227.0
1	Indian Institute of Technology Bombay	AI	OPEN	Female-only (including Supernumerary)	1425	3605	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	6915.5
2	Indian Institute of Technology Bombay	AI	OPEN (PwD)	Gender-Neutral	20	20	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	1425.0
3	Indian Institute of Technology Bombay	AI	OBC-NCL	Gender-Neutral	703	1037	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	3460.5
4	Indian Institute of Technology Bombay	AI	OBC-NCL	Female-only (including Supernumerary)	1161	1185	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	929.0

Figure: 5.3 Data after Feature Extraction:

5.7 Data Preprocessing

After feature Extraction is complete we will perform data preprocessing. In data preprocessing we will fill missing values, removing duplicates and perform data encoding. Data set is taken as input and then removes missing and duplicate values from the dataset and make dataset more accurate and easy.

5.7.1 Null Values: Dealing with missing data is crucial in machine learning because most algorithms require complete data for training. If missing data is not handled properly, it can lead to biased or inaccurate models. There are various techniques to handle missing data in machine learning, some of which include:

- Dropping Rows: One simple approach is to remove rows that contain missing values. However, this method may not be suitable if the amount of missing data is significant, as it could result in a loss of valuable information.

- Mean/Median/Mode Imputation: Replace missing values with the mean, median, or mode of the feature. This approach is straightforward but may not be appropriate if the missing data is related to specific patterns.

df.isnull().sum() this code is used to find the total number of null values in each feature in the data set and df.isnull().sum().sum() returns the number of missing values is to skip rows with missing values in data set.

```
In [83]: df.isnull().sum().sum()
Out[83]: 666
```

```
In [82]: df.isnull().sum()
Out[82]: Institute       0
         Quota           0
         Seat Type       0
         Gender          0
         Opening         0
         Closing         0
         Round           0
         Year            0
         Duration      666
         Degree          0
         Branch          0
         Avg_ranking     0
         dtype: int64
```

Figure: 5.4 Checking Null Values in Data Set

Here we use df.fillna(df["Duration"].mode()[0], inplace=True) method to fill null values.

```
In [84]: df.fillna(df["Duration"].mode()[0], inplace=True)
```

Figure: 5.5 Filling Null Values with Mode

5.7.2 Duplicate Values: df.duplicated().sum() this code is used to find the total number of duplicate values in data set.

Here df.drop_duplicates(inplace=True) code we will use to drop dublicte in our data set.

```
In [35]: df.duplicated().sum()
Out[35]: 42

In [36]: df.drop_duplicates(inplace=True)
```

Figure 5.6 Checking Duplicate Value in Data Set

5.7.3 Feature Encoding

Feature encoding is the process of transforming categorical variables into numerical representations that can be understood by machine learning algorithms. This may involve techniques such as one-hot encoding, label encoding, or ordinal encoding, depending on the nature and requirements of the categorical data.

The code snippet df.dtypes is typically used in Python with the Pandas library to display the data types of each column in a DataFrame, where df is the variable representing the DataFrame. The dtypes attribute provides information about the data types of the columns in the DataFrame.

```
In [7]: df.dtypes

Out[7]: Institute       object
        Quota           object
        Seat Type       object
        Gender          object
        Opening          int64
        Closing          int64
        Round          float64
        Year             int64
        Duration        object
        Degree          object
        Branch          object
        Avg_ranking    float64
        dtype: object
```

Figure 5.7 Checking Data Type of Columns

As we see Institute, Quota, Seat Type, Gender, Duration, Degree, Branch are categorical value and we have to convert categorical value into integer. We use LabelEncoding to convert categorical data into integer. First of all we have to import library from sklearn.preprocessing import LabelEncoder.

```
In [1]: import pandas as pd
        from sklearn.preprocessing import LabelEncoder
```

Figure 5.8 Import library Label Encoder and Pandas

After importing library we will use command le.fit_transform(df['Quota']) to convert our categorical column into integer.

```
In [8]: le = LabelEncoder()
        df['Quota'] = le.fit_transform(df['Quota'])
        df['Seat Type'] = le.fit_transform(df['Seat Type'])
        df['Gender'] = le.fit_transform(df['Gender'])
        df['Duration'] = le.fit_transform(df['Duration'])
        df['Degree'] = le.fit_transform(df['Degree'])
        df['Branch'] = le.fit_transform(df['Branch'])
```

Figure 5.9 Data Encoding

5.8 X_TRAIN and Y_TRAIN

After data preprocessing and feature engineering is performed we divided data set into two groups training data and testing data where data from year 2016 to 2022 is selected as a training data. Our data has twelve attributes Opening, Closing, Avg_ranking, Quota, Seat Type, Institute, Gender, Year, Duration, Round, Degree, Durat

ion.

```
X_TRAIN and Y_TRAIN

[50]: X_train = train[[ 'Quota', 'Seat Type','Institute' , 'Gender','Year','Duration','Round','Opening','Closing', "Avg_ranking"]]
      X_test = test[[ 'Quota', 'Seat Type','Institute' , 'Gender','Year','Duration','Round','Opening','Closing' ,"Avg_ranking"]]
```

Figure 5.10 X_TRAIN and Y_TRAIN

Quota, Seat Type, Institute, Gender, Year, Duration, Round, Opening, Closing, Avg_ranking columns are selected as in put data.

5.9 X_TEST and Y_TEST

After data preprocessing and feature engineering is performed we will data is divided into two groups training data and testing data where year 2023 data is selected for testing. Our data has twelve attributes Opening, Closing, Avg_ranking, Quota, Seat Type, Institute, Gender, Year, Duration, Round, Degree, Duration.

```
In [51]: Y_train = train[["Institute" , 'Degree' , 'Branch']]
         Y_test = test[["Institute" , 'Degree', 'Branch']]
```

Institute, Degree, Branch are select for output.

Figure 5.11 X_TRAIN and Y_TRAIN

5.10 Model Building

After divide data into training and testing and selecting our input and output we will fit X_train and X_test data into multiple models. For model fitting we will have to follow some steps such as:

5.10.1 Import library

Step 1: firstly we have to import essential library.

Importing Pandas, Numpy, Matplotlib, Seaborn and Warings Library.

```
In [1]: # Basic Import
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        # Modelling
        from sklearn.multioutput import MultiOutputClassifier
        from sklearn.metrics import mean_squared_error, r2_score
        from sklearn.neighbors import KNeighborsRegressor
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.ensemble import RandomForestRegressor,AdaBoostRegressor,RandomForestClassifier,AdaBoostClassifier
        from sklearn.svm import SVR
        from sklearn.linear_model import LinearRegression, Ridge,Lasso
        from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
        from sklearn.model_selection import RandomizedSearchCV
        from catboost import CatBoostRegressor
        from xgboost import XGBRegressor
        from sklearn.metrics import accuracy_score
        import warnings
```

Figure 5.12 Import Library

5.5.2 Import data

Step 2: Second step is to load data.

Import the CSV Data as Pandas DataFrame

```
In [2]: train = pd.read_csv("train")
        test = pd.read_csv("test")
```

Figure 5.13 Import Training and Testing Data Set

5.10.3 Split Input and Output Data

Step 3: third step is to split input and output data. Split training and testing data into X_train, Y_train and testing data as X_test , Y_test.

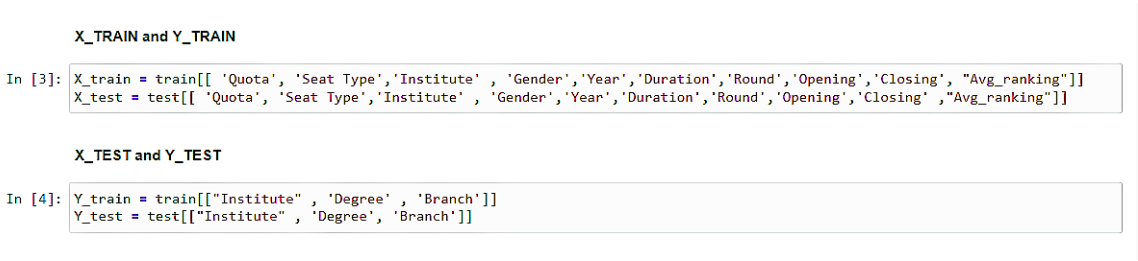

Figure 5.14 X_Train, Y_TRAIN, X_TEST, Y_TEST

5.11 Model Training

Step 4: fourth step is model training. Now we will train model by applying different machine learning algorithms.

5.11.1 Linear model

Firstly we will fit our training data X_train and Y_train data in Linear Regression model for our model training.

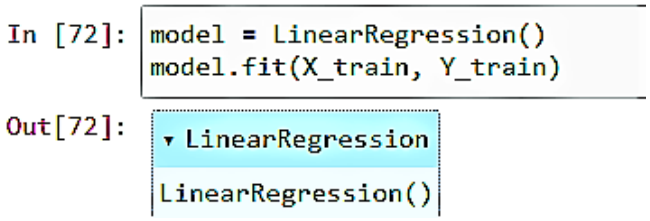

Figure 5.15 linear model

5.11.2 Lasso

Now we will fit our training data X_train and Y_train data in Lasso for our model training.

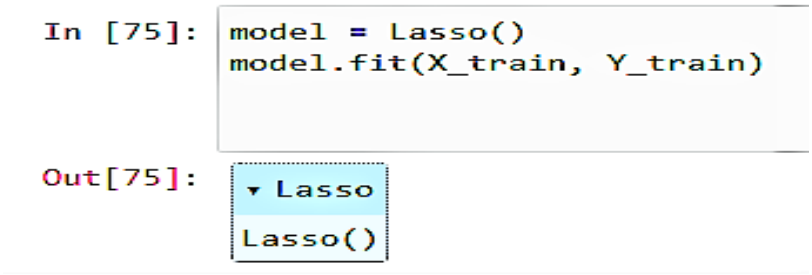

Figure **5.16 Lasso**

5.11.3 Ridge

Now we will fit our training data X_train and Y_train data in Ridge for our model training.

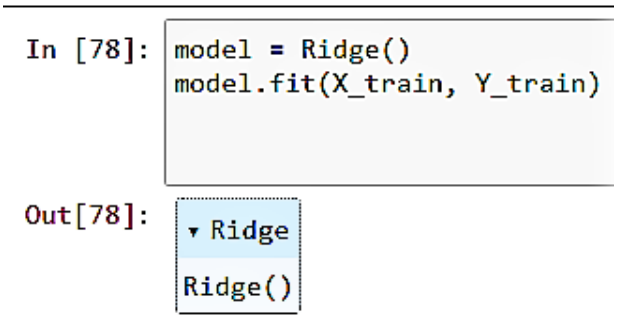

Figure 5.17 Ridge

5.11.4 K-Neighbors Regressor

Now we will fit our training data X_train and Y_train data in K-neighbour Regressor for our model training.

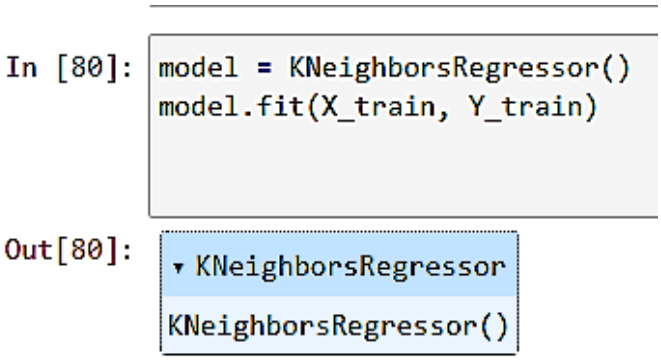

Figure 5.18 K-Neighbors Regressor

5.11.5 Decision Tree

Now we will fit our training data X_train and Y_train data in Decision Tree Regressor for our model training.

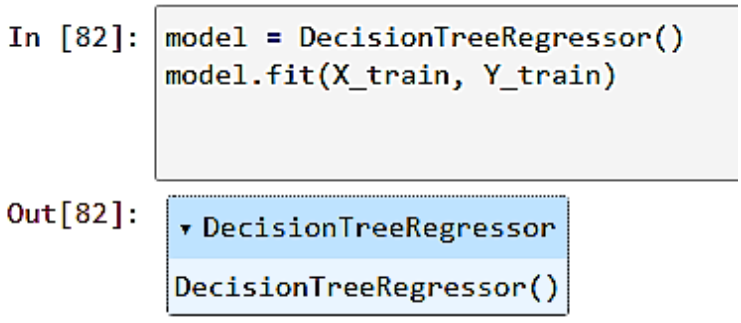

Figure 5.19 Decision Tree

5.11.6 Random Forest

Now we will fit our training data X_train and Y_train data in Random Forest for our model training.

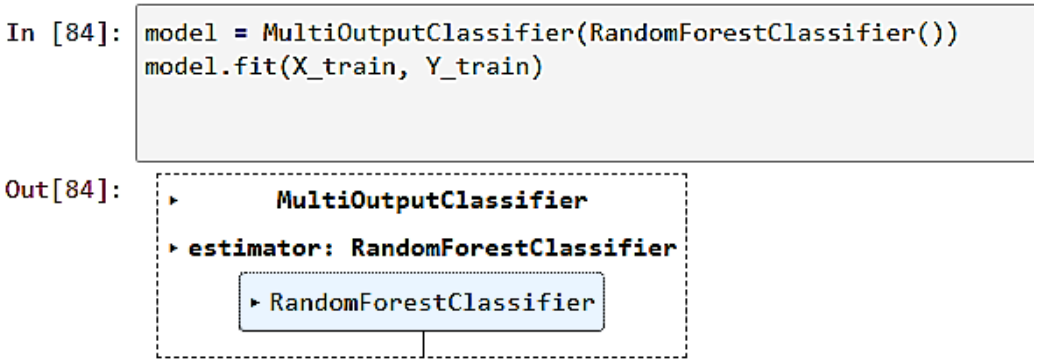

Figure 5.20 Random Forest

5.11.7 CatBoost Regressor

Now we will fit our training data X_train and Y_train data in CatBoost Regressor for our model training.

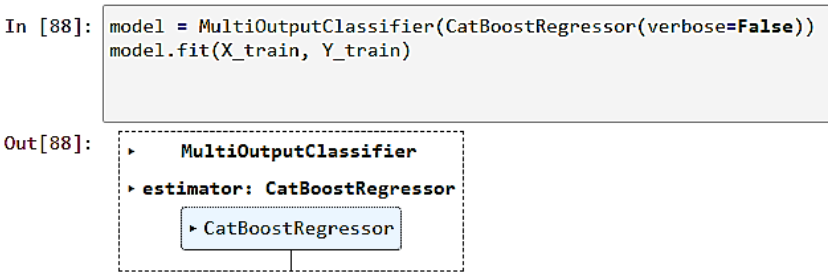

Figure 5.21 CatBoost Regressor

5.11.8 AdaBoost Classifier

Now we will fit our training data X_train and Y_train data in AdaBoost Classifier for our model training.

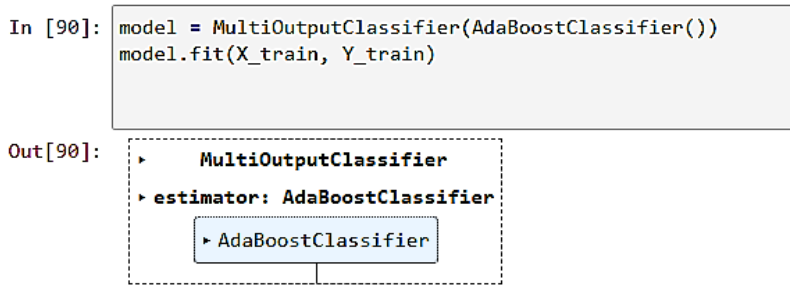

Figure 5.22 AdaBoost Classifier

CHAPTER 6

TESTING AND RESULT

This chapter presents testing of proposed approach and experimental result which are used for college prediction system and analyze the performance of used methods.

6.1 Data Set Used

IITs data for the past Six years (2018-2023) is collected from Joint Seat Allocation Authority Admission website https://josaa.nic.in/. Our data has nine attributes, names as Year, Institute, Program, Seat Type, Gender, Round, Opening, Closing and Quota. After Feature Engineering and adding new features in our data it has twelve attributes named as Year, Institute, Quota, Seat Type, Gender Duration, Degree, Avg_ranking, Round , Opening, and Closing.

5]:

	Institute	Quota	Seat Type	Gender	Opening	Closing	Round	Year	Duration	Degree	Branch	Avg_ranking
0	Indian Institute of Technology Bombay	AI	OPEN	Gender-Neutral	1040	1974	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	4227.0
1	Indian Institute of Technology Bombay	AI	OPEN	Female-only (including Supernumerary)	1425	3605	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	6915.5
2	Indian Institute of Technology Bombay	AI	OPEN (PwD)	Gender-Neutral	20	20	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	1425.0
3	Indian Institute of Technology Bombay	AI	OBC-NCL	Gender-Neutral	703	1037	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	3460.5
4	Indian Institute of Technology Bombay	AI	OBC-NCL	Female-only (including Supernumerary)	1161	1185	1.0	2018	4 Years	Bachelor of Technology	Aerospace Engineering	929.0
...	...	...	...	...	...	...	...	...	...	...	...	...
2673	Indian Institute of Technology Delhi	AI	OBC-NCL	Female-only (including Supernumerary)	5742	6280	1.0	2023	4 Years	Bachelor of Technology	Textile Technology	7761.0
2674	Indian Institute of Technology Delhi	AI	SC	Gender-Neutral	1460	1771	1.0	2023	4 Years	Bachelor of Technology	Textile Technology	1659.5
2675	Indian Institute of Technology Delhi	AI	SC	Female-only (including Supernumerary)	2722	2769	1.0	2023	4 Years	Bachelor of Technology	Textile Technology	2578.0
2676	Indian Institute of Technology Delhi	AI	ST	Gender-Neutral	840	984	1.0	2023	4 Years	Bachelor of Technology	Textile Technology	1220.0
2677	Indian Institute of Technology Delhi	AI	ST	Female-only (including Supernumerary)	1578	1578	1.0	2023	4 Years	Bachelor of Technology	Textile Technology	4645.5

Figure 6.1 Data Set

6.2 Testing of Proposed Approach

Proposed approach is implemented using Jupyter Notebook. It is evaluating on a IITs dataset which are collected from Joint Seat Allocation Authority Admission website https://josaa.nic.in/. "Test data" is a set of data that is used to evaluate the performance of a machine learning model or any other predictive model. It is separate from the data used for training the model and is used to assess how well the model generalizes to new, unseen. Data set of year 2023 is our testing data.

We will create a function name evaluate model. Our function will take two parameters name as true and predicted where true in our true output and predicted is output predicted by our machine learning model. In this function we will calculate mean square error, mean absolute error, r2 square and accuracy of our model.

```
def evaluate_model(true, predicted):
    mae = mean_absolute_error(true, predicted)
    mse = mean_squared_error(true, predicted)
    rmse = np.sqrt(mean_squared_error(true, predicted))
    r2_square = r2_score(true, predicted)
    acuracy = r2_square*100
    return mae, rmse, r2_square,acuracy
```

Figure 6.2. Evaluation Model Function

We will predict output of our train data and test data and save results In Y_train_pred and Y_test_pred. after predicting output we will pass our ture vale and predicted value in our function evaluate_model and calculate mean absolute error, mean square error, root mean square error and acuracy.

Make predictions

```
In [100]: Y_train_pred = model.predict(X_train)
          Y_test_pred = model.predict(X_test)
```

Evaluate Train and Test dataset

```
In [101]: model_train_mae , model_train_rmse, model_train_r2 ,model_train_acuracy = evaluate_model(Y_train, Y_train_pred)
          model_test_mae , model_test_rmse, model_test_r2, model_test_acuracy = evaluate_model(Y_test, Y_test_pred)
          print('Model performance for Training set')
          print("- Root Mean Squared Error: {:.4f}".format(model_train_rmse))
          print("- Mean Absolute Error: {:.4f}".format(model_train_mae))
          print("- R2 Score: {:.4f}".format(model_train_r2))
          print("- Acuracy Score: {:.4f}".format(model_train_acuracy))
          print('**********')
          print('Model performance for Test set')
          print("- Root Mean Squared Error: {:.4f}".format(model_test_rmse))
          print("- Mean Absolute Error: {:.4f}".format(model_test_mae))
          print("- R2 Score: {:.4f}".format(model_test_r2))
          print("- Acuracy Score: {:.4f}".format(model_test_acuracy))
```

Figure 6.3 Make Prediction

6.3 Model Performance

6.3.1. Linear Regressor

```
Model performance for Training set
- Root Mean Squared Error: 14.9319
- Mean Absolute Error: 7.4500
- R2 Score: 0.5811
- Acuracy Score: 58.1077
**********
Model performance for Test set
- Root Mean Squared Error: 13.2287
- Mean Absolute Error: 6.7228
- R2 Score: 0.5458
- Acuracy Score: 54.5794
```

Figure 6.4 Linear Regression Model Performance

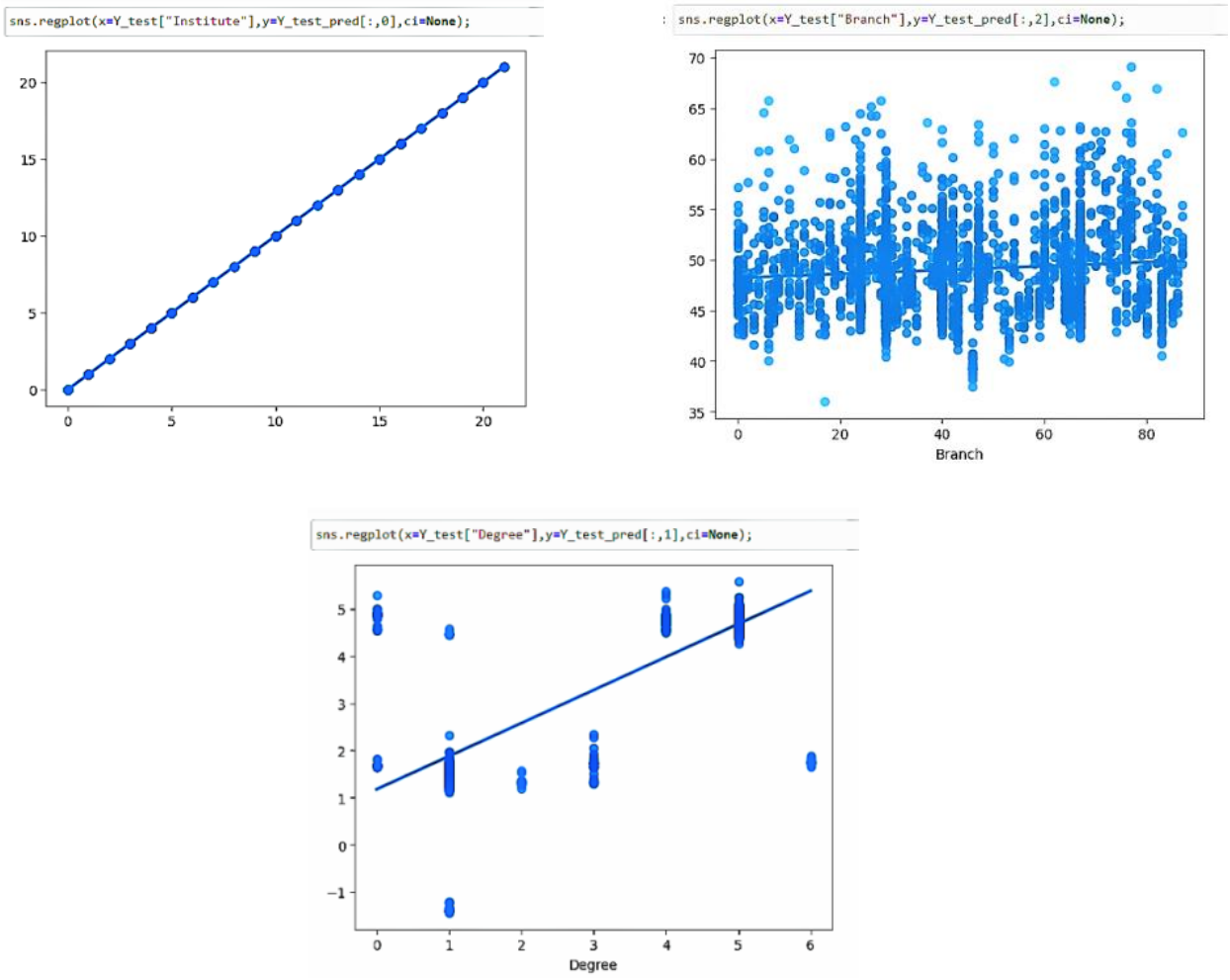

Figure 6.5 Actual Vs Predicted Graph in Linear Model

6.3.2. Lasso

```
Model performance for Training set
- Root Mean Squared Error: 14.9792
- Mean Absolute Error: 7.7930
- R2 Score: 0.3486
- Acuracy Score: 34.8603
**********************************
Model performance for Test set
- Root Mean Squared Error: 13.3790
- Mean Absolute Error: 7.0851
- R2 Score: 0.3082
- Acuracy Score: 30.8179
```

Figure 6.6 Lasso Model Performance

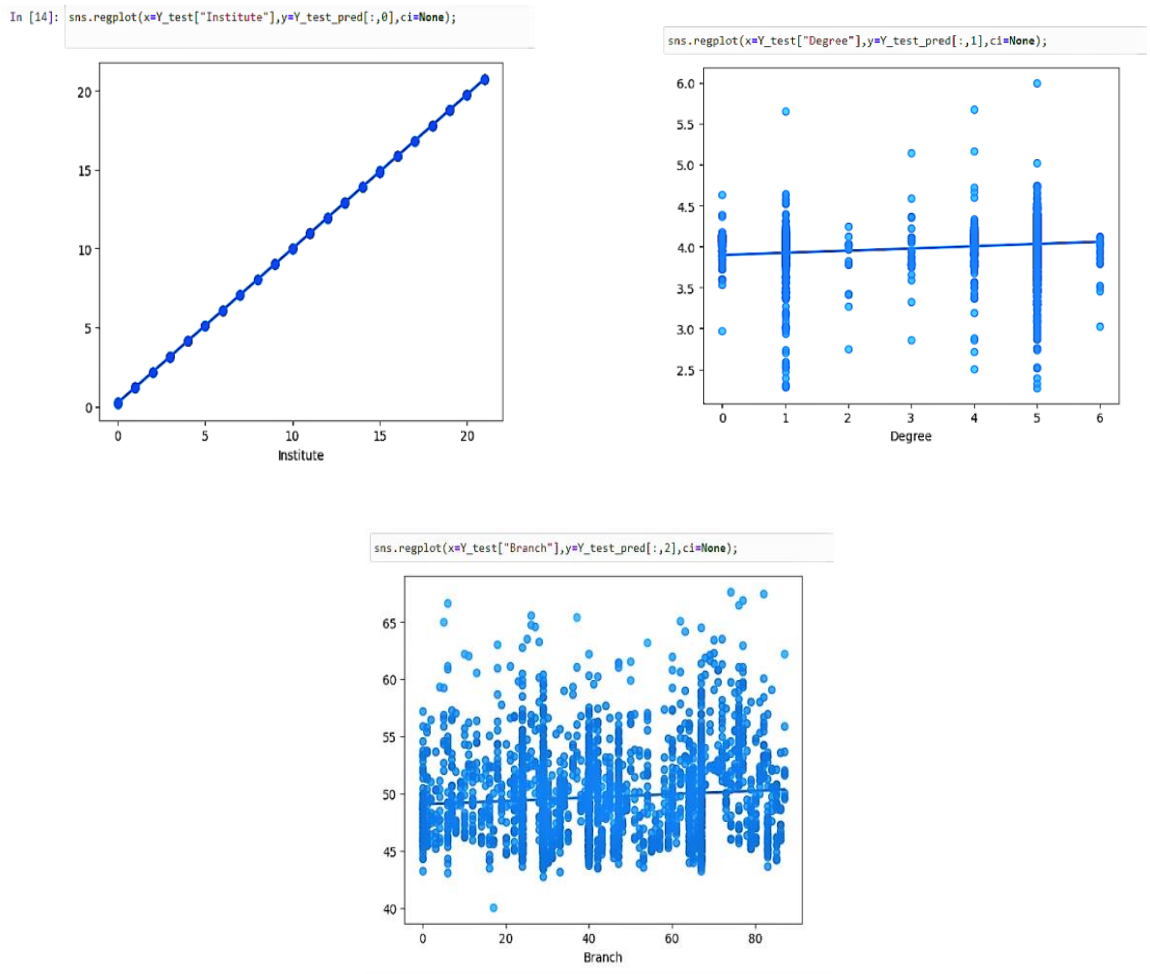

Figure 6.7 Actual Vs Predicted Graph in Lasso

6.3.3. Ridge:

```
Model performance for Training set
- Root Mean Squared Error: 14.9319
- Mean Absolute Error: 7.4500
- R2 Score: 0.5811
- Acuracy Score: 58.1077
**********************************
Model performance for Test set
- Root Mean Squared Error: 13.2287
- Mean Absolute Error: 6.7228
- R2 Score: 0.5458
- Acuracy Score: 54.5793
```

Figure 6.8 Ridge Model Performance

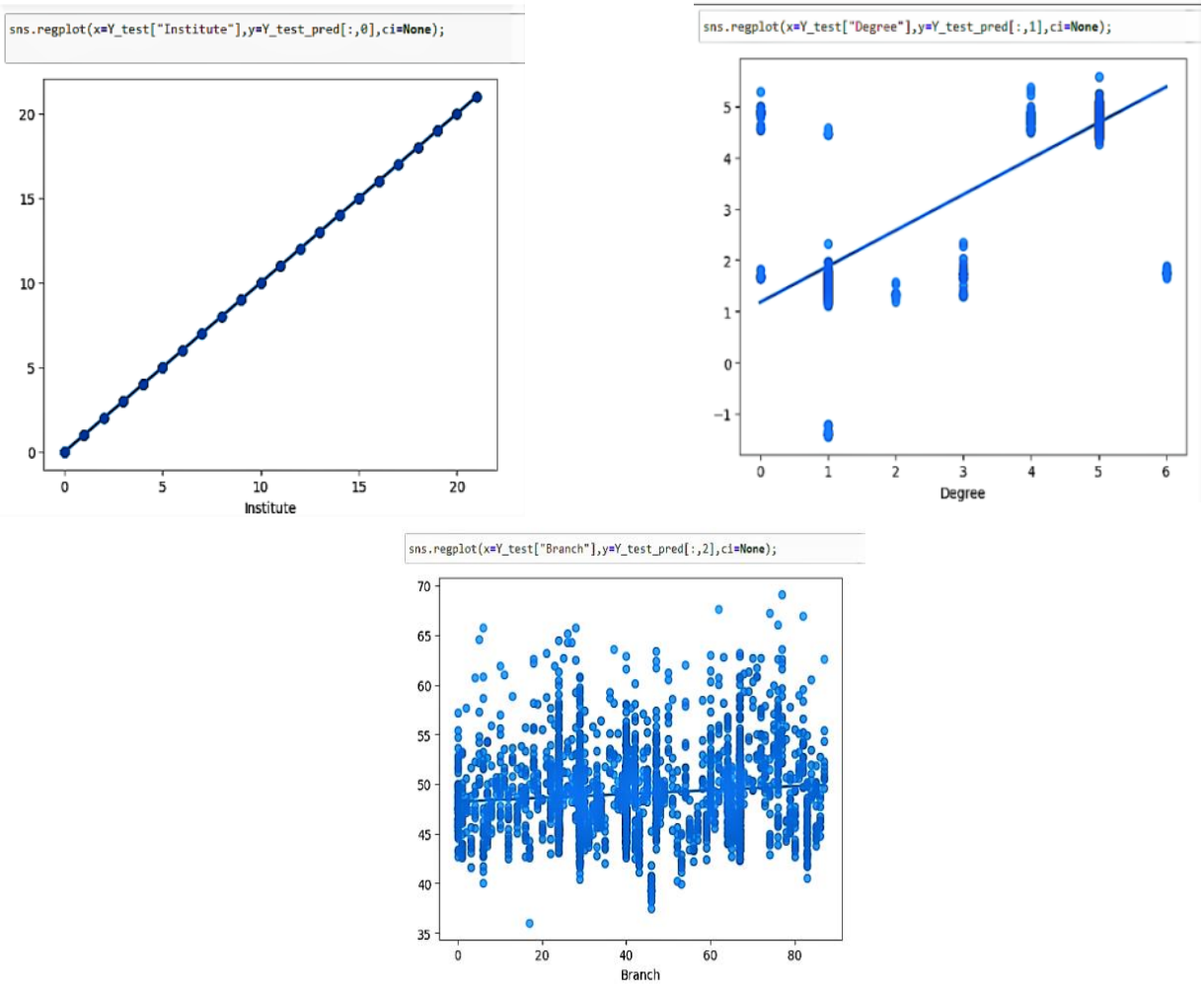

Figure 6.9 Actual Vs Predicted Graph in Ridge

6.3.4. K-Neighbor Regressor

```
Model performance for Training set
- Root Mean Squared Error: 13.1580
- Mean Absolute Error: 7.7280
- R2 Score: 0.3084
- Acuracy Score: 30.8397
**********************************
Model performance for Test set
- Root Mean Squared Error: 15.9106
- Mean Absolute Error: 9.5319
- R2 Score: -0.2609
- Acuracy Score: -26.0945
```

Figure 6.10 Model Performance in K-Neighbor Regressor

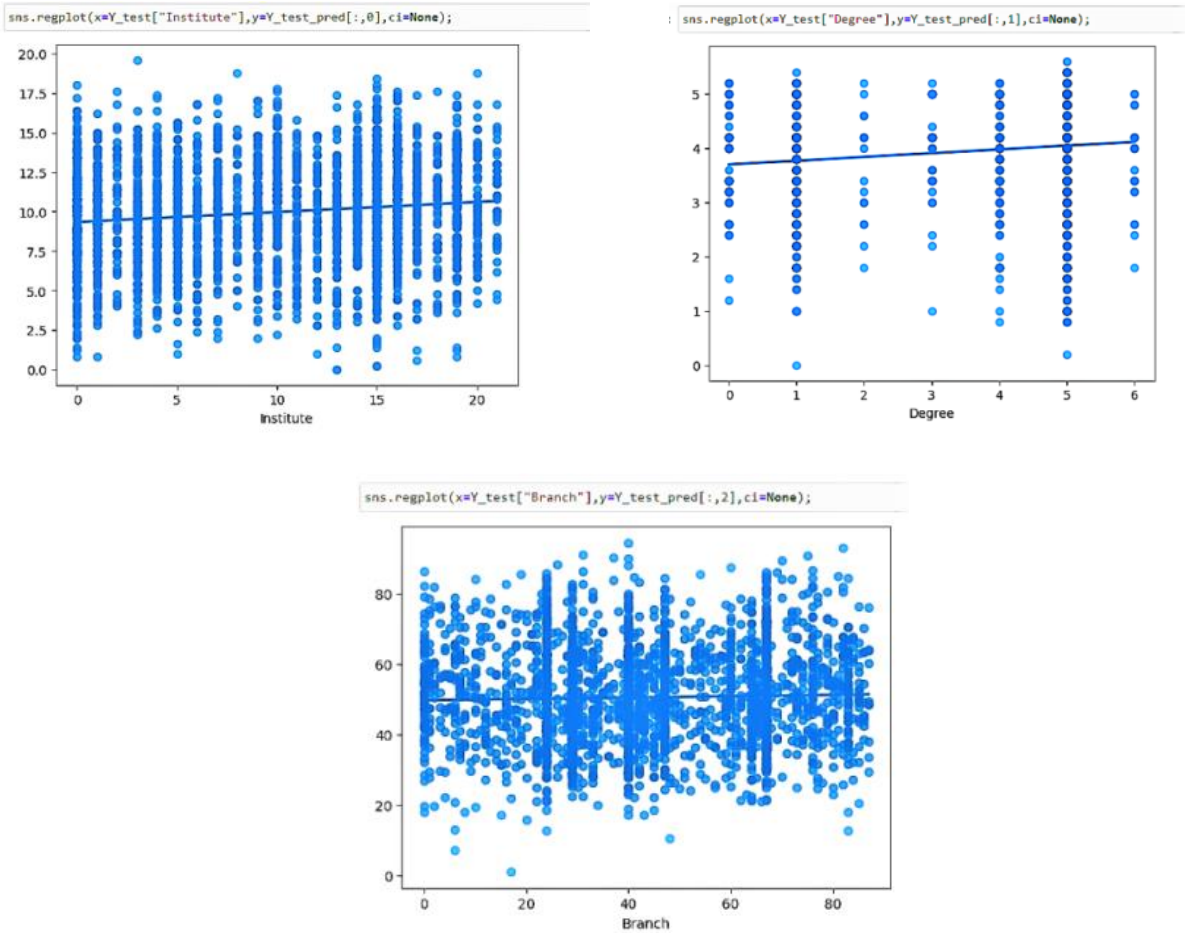

Figure 6.11 Actual vs. Predicted Graph in K-Neighbor Regressor

6.3.5. Decision Tree Regressor

```
Model performance for Training set
- Root Mean Squared Error: 0.0000
- Mean Absolute Error: 0.0000
- R2 Score: 1.0000
- Acuracy Score: 100.0000
**********************************
Model performance for Test set
- Root Mean Squared Error: 17.0630
- Mean Absolute Error: 7.3079
- R2 Score: 0.2322
- Acuracy Score: 23.2247
```

Figure 6.12 Model Performance in Decision Tree Regressor

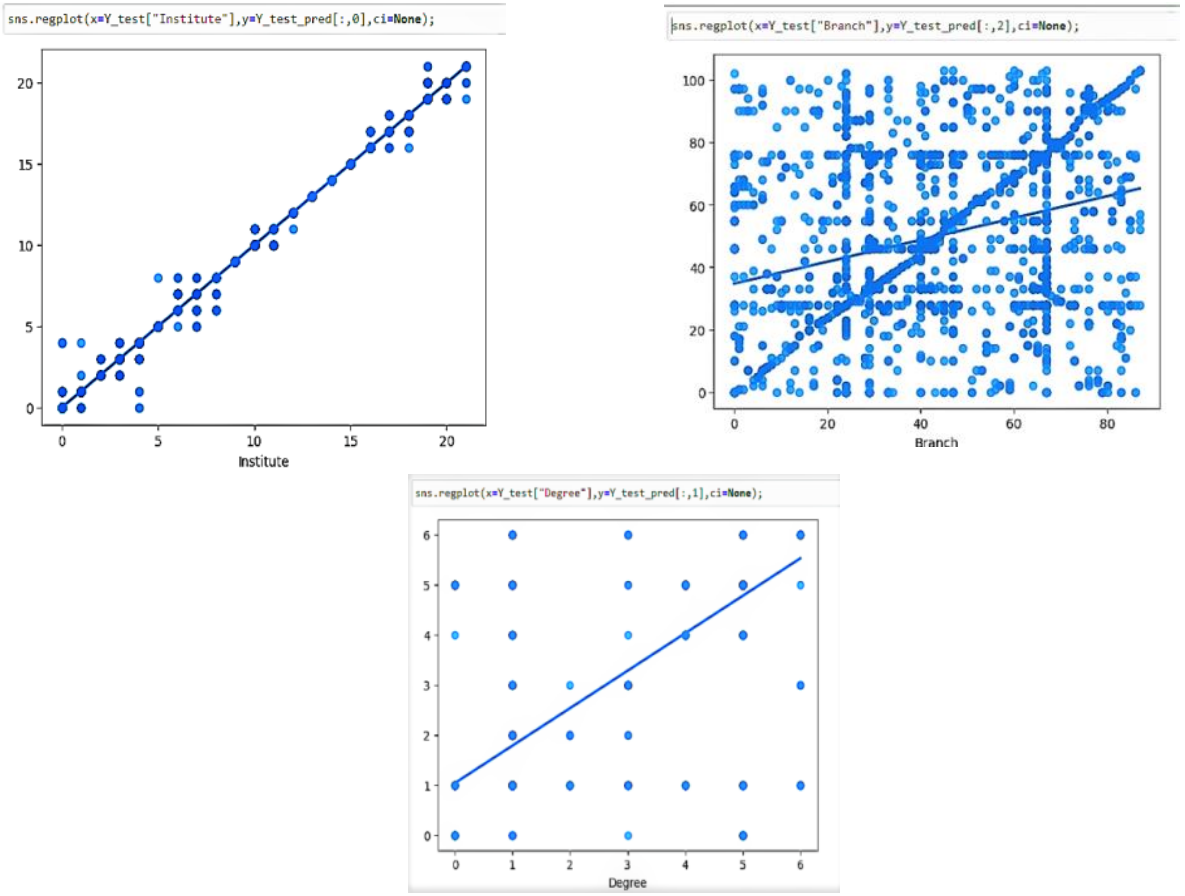

Figure 6.13 Actual vs. Predicted Graph in Decision Tree Regressor

6.3.6. Random Forest Classifier:

```
Model performance for Training set
- Root Mean Squared Error: 0.0000
- Mean Absolute Error: 0.0000
- R2 Score: 1.0000
- Acuracy Score: 100.0000
**********************************
Model performance for Test set
- Root Mean Squared Error: 15.6519
- Mean Absolute Error: 6.2799
- R2 Score: 0.4240
- Acuracy Score: 42.3964
```

Figure 6.14 Model Performance in Random Forest Classifier

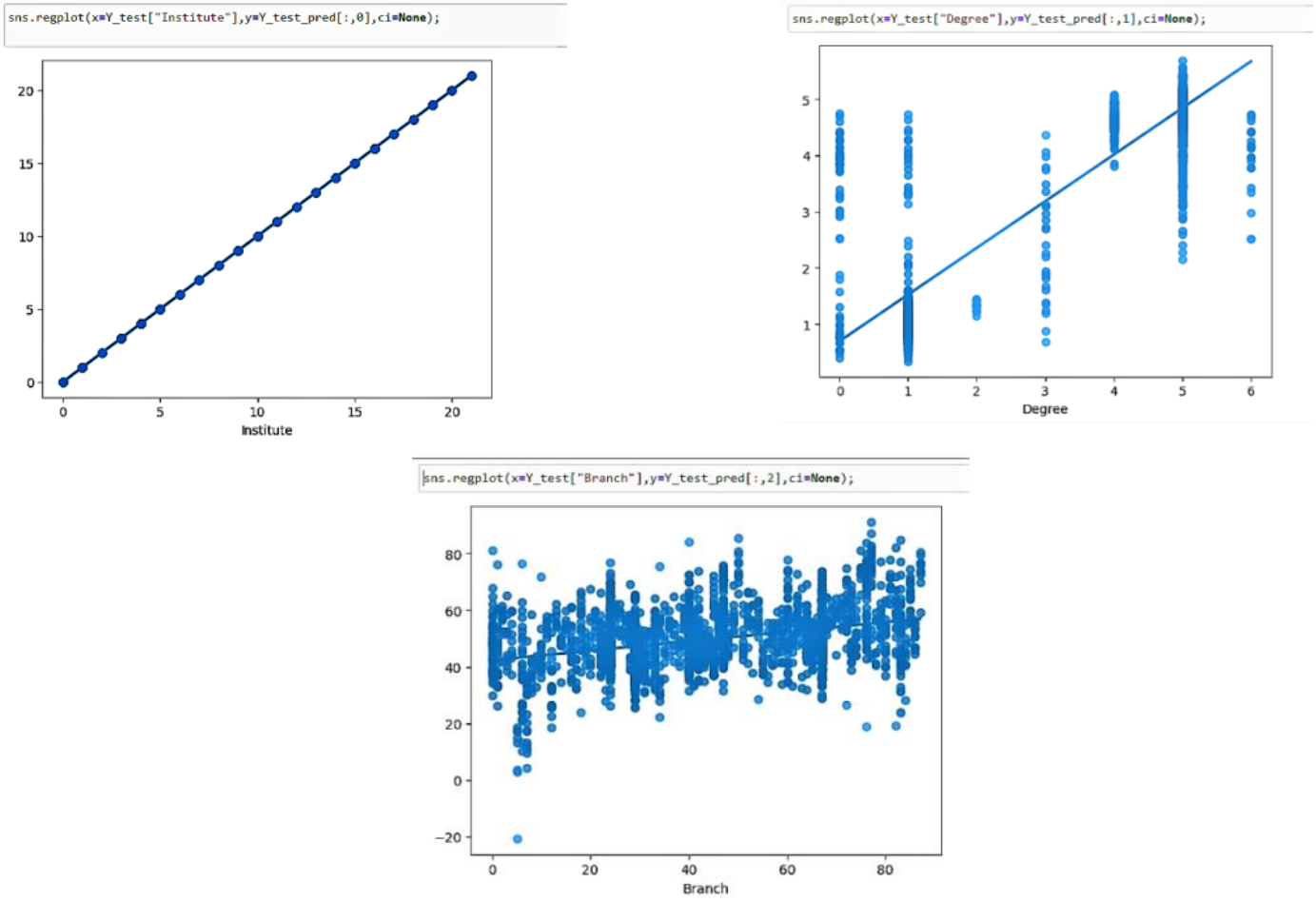

Figure 6.15 Actual vs. Predicted Graph in Random Forest Classifier

6.3.7. CatBooast:

```
Model performance for Training set
- Root Mean Squared Error: 11.8865
- Mean Absolute Error: 5.6524
- R2 Score: 0.7677
- Acuracy Score: 76.7659
**********************************
Model performance for Test set
- Root Mean Squared Error: 12.6052
- Mean Absolute Error: 6.0651
- R2 Score: 0.6170
- Acuracy Score: 61.7024
```

Figure 6.16 Model Performance in CatBooast

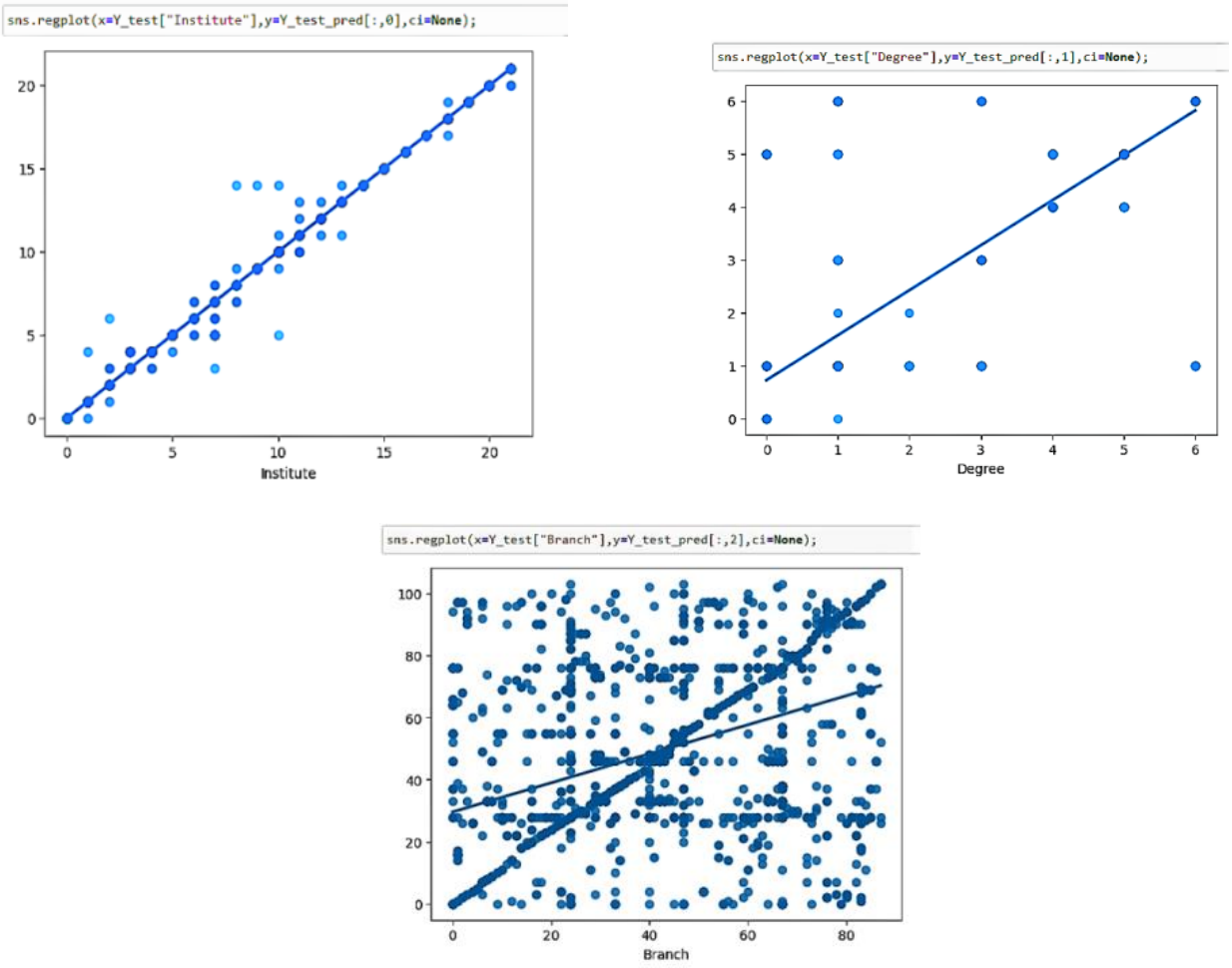

Figure 6.17 Actual vs. Predicted Graph in CatBooast

6.3.8. XGB Regressor

```
Model performance for Training set
- Root Mean Squared Error: 11.0561
- Mean Absolute Error: 5.1857
- R2 Score: 0.8047
- Acuracy Score: 80.4731
**********************************
Model performance for Test set
- Root Mean Squared Error: 12.9150
- Mean Absolute Error: 6.1865
- R2 Score: 0.5938
- Acuracy Score: 59.3833
```

Figure 6.18 Model Performance in XGB Regressor

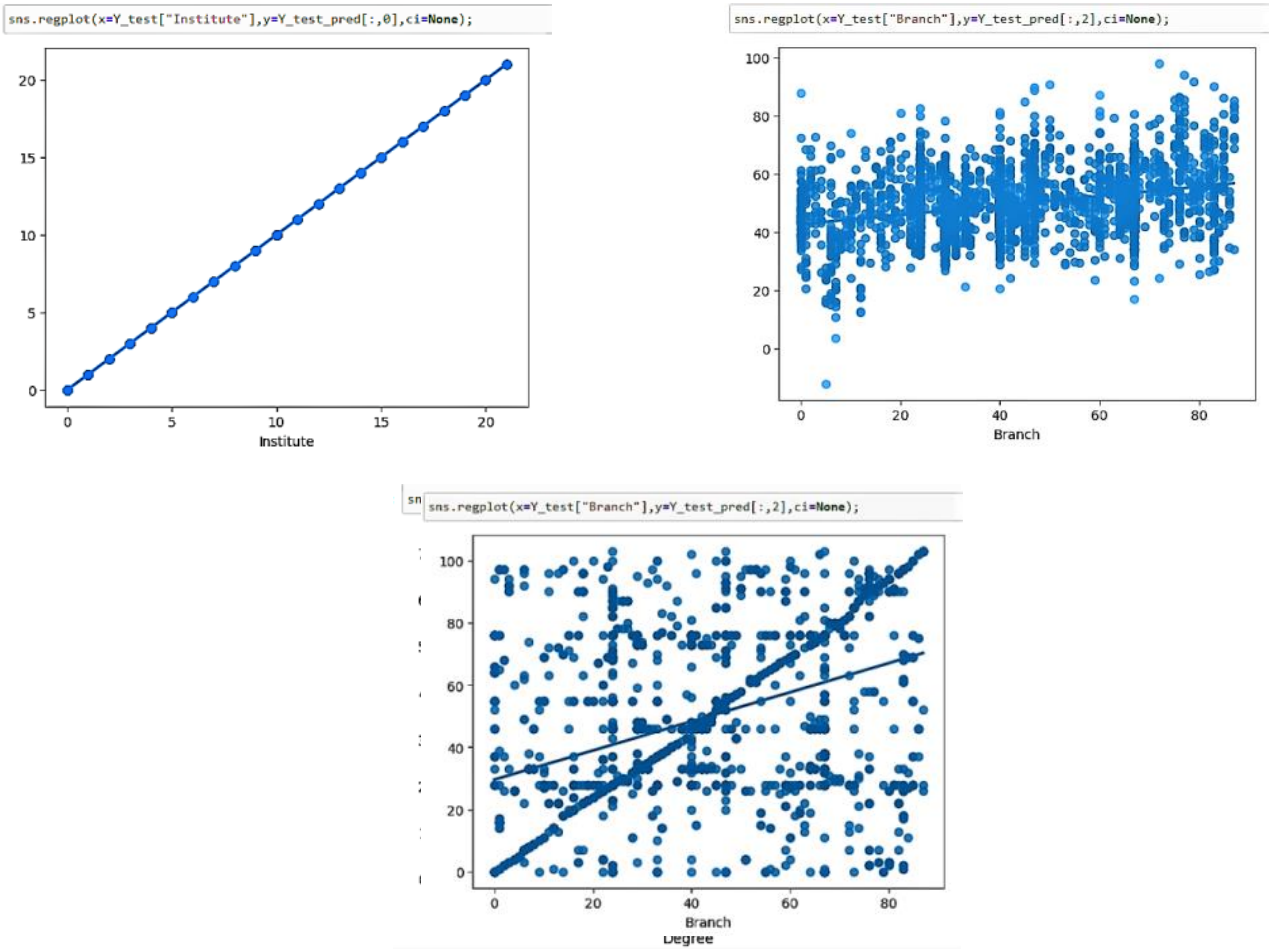

Figure 6.19 Actual vs. Predicted Graph in XGB Regressor

6.4 Results

Proposed approach is implemented using python. It is evolved on IIT college data set which contain attributes like Institute, Branch, Degree, Duration, Round, Gender, Seat Type, Opening, Closing, Avg_ranking, Year. The data set is dividing into two parts training and testing where data of year 2018, 2019, 2020, 2021, 2022 are taken as training data and data for year 2023 is taken as testing data. In order to measure performance of our model we will calculate mean square error, mean absolute error, r2 square.

```
pd.DataFrame(list(zip(model_list, accuracy_list)), columns=['Model Name', 'Accuracy']).sort_values(by=["Accuracy"],
                                                                                                   ascending=False)
```

	Model Name	Accuracy
7	CatBoosting Regressor	0.617024
6	XGBRegressor	0.593833
0	Linear Regression	0.545794
2	Ridge	0.545793
5	Random Forest Regressor	0.412385
1	Lasso	0.308179
4	Decision Tree	0.236623
3	K-Neighbors Regressor	-0.260945
8	AdaBoost Regressor	-2.453126

Figure 6.20 Accuracy of Model

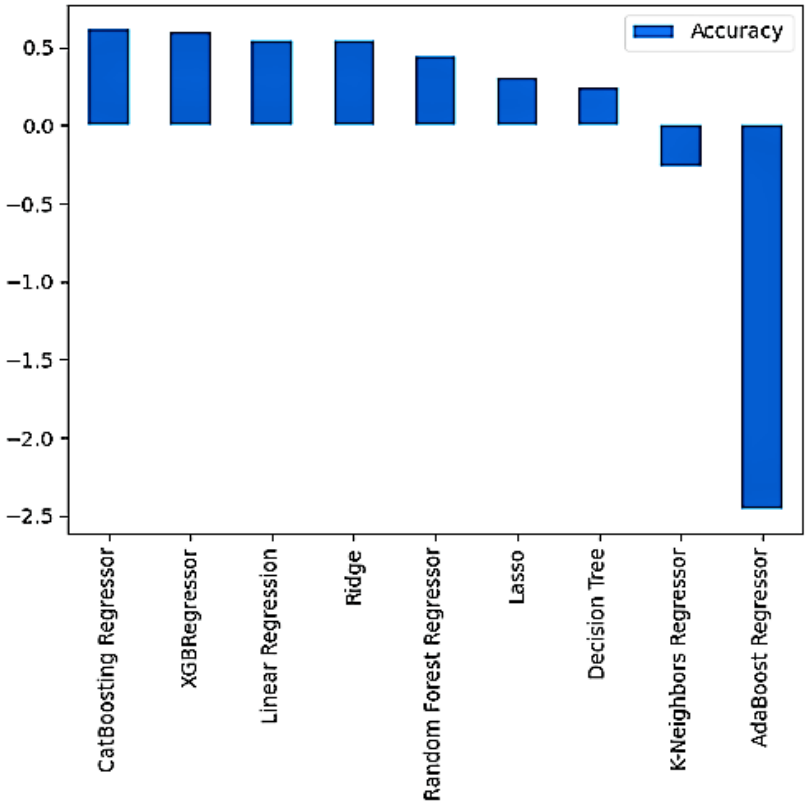

Figure 6.21 Graphical Representation of Model Accuracy

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 Conclusion

The main aim of this project is to design and implement a college prediction system using machine learning. The purposed approach uses various machine learning algorithms like Logistic regression, Ridge, Lasso, K-neighbor Regressor, SVM, Random Forest, AdaBoost, CatBoost are used and CatBoost machine learning algorithm reached maximum accuracy 61.7%.so the above calculation we conclude that CatBoost is ideal algorithm for College prediction system having 61.7% Accuracy higher than other algorithms .

7.2 Limitation of Purposed Approach

The Purposed Approach has following limitation:

- The proposed approach works only with limited data.
- The proposed approach emphasizes more on time efficiency, and less on other parameters.
- The proposed approach can predict the only IIT Colleges.

7.3 Future Enhancement

Enhancements which can be done in future are as follows:

- The proposed approach works only with limited data of restricted areas, so further research can be done with vast data.
- Working on some more attributes live Placements, Alumni Review, Extra Curriculum activities etc.
- Work can be done on improving time efficiency.
- Algorithms can be improvised for efficient prediction and for enhancement of working.
- Future work can be done on implementing techniques on high dimensional data.
- Further research can be done for multi-class classification.

Book References

1. Introduction to Machine Learning with Python, Andreas C. Müller and Sarah Guido, O'Reilly Media, Inc Publication, USA, 2016.
2. Python Machine Learning, Sebastian Raschka and Vahid Mirjallil, Packt Publishing Ltd., 2019.
3. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, Aurélien Géron 2nd Edition, O'Reilly Media, Inc, Publication, USA, 2019.

Website References

- https://www.kaggle.com/
- https://www.youtube.com/
- https://www.google.com/
- https://www.wikipedia.org/
- https://www.geeksforgeeks.org/
- https://www.tutorialspoint.com/
- https://www.javatpoint.com/
- https://pwskills.com/

Appendix A

User's Manual

This User Manual contains all essential information for this project to make full use of the information system. This project is implemented on Jupyter Notebook.

Following is the procedure for installation and execution of this project:

Installation Procedure

1. To code python packages, powerful IDE is or web application is also required e.g. Jupyter Notebook.

2. Choose Windows Operating System like Windows 7 or above and install setup for "Jupyter Notebook".

3. Jupyter Notebook installation can be done by using Anaconda Distribution or by using pip command.

4. Download Anaconda's latest python3 version.

5. Installed the downloaded version of anaconda by following instruction on official documentation of anaconda's page.

6. Now Congratulations you have install your Jupyter Notebook.

7. Instead of using Anaconda you can install Jupyter by using pip.By typing "pip install jupyter" on command prompt.

8. Jupyter required python version3 or greater version for better support.

Process Execution

1. Open Anaconda and select the Jupyter Notebook.

2. Select the project you want to run and debug.

3. Now user can know about the project by clicking in to the folders appearing on the screen in project folder.

4. User can start running the project by right clicking on screen and visualize result and working of the program.