Beyond Parental Height: A Multi-Model Deep Learning Approach for Personalized Adult Height Prediction

A Project Report submitted in the partial fulfillment

of the Requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

A. Siva Nagendra (21471A0560)

G. Gowtham (21471A0525)

Ch. Ravi Sankar (21471A0515)

Under the esteemed guidance of

Dr. S. Siva Nageswara Rao, M. Tech, Ph.D., Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Tier -1

NIRF rank in the band of 201-300 and an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada

KOTAPPAKONDA ROAD, YALLAMANDA VILLAGE, NARASARAOPET- 522601

2024-2025

NARASARAOPETA ENGINEERING COLLEGE

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name "Beyond Parental Height: A Multi-Model Deep Learning Approach for Personalized Adult Height Prediction" is a bonafide work done by the team A. Siva Nagendra (21471A0560), G. Gowtham (21471A0525), Ch. Ravi Sankar (21471A0515) in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2024-2025.

PROJECT GUIDE

Dr. S. Siva Nageswara Rao, M. Tech., Ph.D.

Professor

PROJECT CO-ORDINATOR

Dodda Venkata Reddy, M.Tech., (Ph.D.)

Assistant Professor

HEAD OF THE DEPARTMENT

Dr. S. N. Tirumala Rao, M.Tech., Ph.D.

Professor & HOD

EXTERNAL EXAMINER

DECLARATION

We declare that this project work titled " BEYOND PARENTAL HEIGHT: A MULTI-MODEL DEEP LEARNING APPROACH FOR PERSONALIZED ADULT HEIGHT PREDICTION " is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has been submitted for any other degree or professional qualification except as specified.

A. Siva Nagendra (21471A0560)
G. Gowtham (21471A0525)
Ch. Ravi Sankar (21471A0515)

ACKNOWLEDGEMENT

We wish to express our thanks to various personalities who are responsible for the successful completion of the project. We are extremely thankful to our beloved chairman sri **M. V. Koteswara Rao**, B.Sc., who took keen interest in us in every step and effort throughout this course. We owe out sincere gratitude to our beloved principal **Dr. S. Venkateswarlu**, M. Tech., Ph.D., for showing his kind attention and valuable guidance throughout the course.

We express our deep felt gratitude towards **Dr. S. N. Tirumala Rao**, M.Tech., Ph.D., HOD of CSE department and also to our guide **Dr. S. Siva Nageswara Rao**, M.Tech., Ph.D., Professor of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks towards **Dodda Venkata Reddy**, M.Tech.,(Ph.D.), Assistant professor & Coordinator of the project for extending his encouragement. Their profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all other teaching and non-teaching staff to department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who involved in giving valuable suggestions had clarifying out doubts which had really helped us in successfully completing our project.

By

A. Siva Nagendra (21471A0560)
G. Gowtham (21471A0525)
Ch. Ravi Sankar (21471A0515)

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community.

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research.

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills.

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

Program Educational Objectives (PEO's)

The graduates of the programme are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

Problem analysis: Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activitieswith an understanding of the limitations.

The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

Life-long learning: Recognize the need for, and have the preparation and ability to engagein independent and life-long learning in the broadest context of technological change.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Project Course Outcomes (CO'S):

CO421.1: Analyse the System of Examinations and identify the problem.

CO421.2: Identify and classify the requirements.

CO421.3: Review the Related Literature.

CO421.4: Design and Modularize the project.

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1		✓											✓		
C421.2	✓		✓		✓								✓		
C421.3				✓		✓	✓	✓					✓		
C421.4			✓			✓	✓	✓					✓	✓	
C421.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C421.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level
2. Medium level
3. High level

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Project mapping with various courses of Curriculum with AttainedPO's:

Name of the Course from Which Principles Are Applied in This Project	Description of the Task	Attained PO
C2204.2, C22L3.2	Defining the problem and applying advanced feature engineering techniques for height prediction models	PO1, PO3
CC421.1, C2204.3, C22L3.2	Critically analyzing project requirements and identifying suitable process models for experiments	PO2, PO3
CC421.2, C2204.2, C22L3.3	Creating logical designs using UML while collaborating on feature engineering as a team	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Testing, integrating, and evaluating regression models with and without outlier removal	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documenting experiments, results, and findings collaboratively within the group	PO10
CC421.5, C2204.2, C22L3.3	Presenting each phase of the project, including raw data analysis and evaluation, in a group setting	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementing and validating models with applications for healthcare and future feature updates	PO4, PO7
C32SC4.3	Designing a web interface to visualize predictions and verify model accuracy effectively	PO5, PO6

ABSRTACT

An advanced height prediction framework to address the limitations of traditional methods dealing with complex and diversified growth factors is proposed in this study. The framework effectively resolves two traditional challenges—data preprocessing and prediction accuracy—by employing advanced feature engineering and outlier handling combined with a multi-model approach. The proposed system is tested using the Galton height dataset, integrating several machine learning models to enhance overall prediction accuracy and leveraging advanced data preprocessing techniques, such as outlier removal, to improve the reliability of predictions. The results demonstrate significant improvement in predicting both typical and rare growth trajectories, making the proposed framework highly robust and suitable for real-time applications in personalized healthcare and growth monitoring.

INDEX

CONTENT	PAGE NO
1. Introduction	01 - 03
2. Literature Review	04 - 05
3. Analysis	06 - 08
4. System Requirements	09 - 12
4.1. Software Requirements	09
4.2. Hardware Requirements	09
4.3. Software and it's description	10
5. Design	13 - 22
5.1. Dataset Overview	14
5.2. Data Preprocessing	14
5.3. Model Building	16
5.4. Classification	18
5.5. Confusion Matrix	20
5.6. Performance Evaluation Using Metrics	22
6. Implementation	23 - 29
6.1. Model Development	23
6.2. Code Deployment using flask	24
7. Testing & Testcases	30 - 36
7.1. Unit Testing	30
7.2. Integration Testing	32
7.3. System Testing	33
8. Result Analysis	37 - 38
9. Conclusion	39 - 40
10. Future Scope	41 - 42
11. References	43 - 44

LIST OF FIGURES

LIST OF FIGURES	PAGE NO
Fig 5.1 Flow chart of Designed system	13
Fig 5.1.1 Dataset Description	14
Fig 5.2.1 Dataset Description after Prepocessing	15
Fig 5.3.1 Linear Regression	16
Fig 5.3.2 Support Vector Regression	17
Fig 5.3.3 XGBoost	17
Fig 5.3.4 Light GBM	17
Fig 5.5.1 Confusion Matrix	21
Fig 7.1.1 Handling missing values testcase - 1	30
Fig 7.1.2 Correct Prediction of height testcase – 2	31
Fig 7.1.3 Correct Prediction of height testcase – 3	32
Fig 7.2.1 Invalid Credentials login testcases - 1	32
Fig 7.2.2 Valid Credentials login testcases - 2	33
Fig 7.3.1 Bypass login and access restricted pages testcases - 1	34
Fig 7.3.2 Complete prediction workflow testcase - 2	35
Fig 7.3.3 Session timeout testcase - 3	36
Fig 8.1 Model accuracy before and after outlier removal	37
Fig 8.2 Training and Testing accuracy before and after outlier removal	38
Fig 8.3 Comparision of Precision, Recall, F1-Score among models	38

1. INTRODUCTION

Accurate estimation of adult height is crucial in paediatric endocrinology, orthopaedics, and public health, playing a significant role in diagnosing and managing growth-related conditions [1] [2]. Predicting final adult height helps identify growth deficiencies, optimize treatment strategies, and guide interventions such as growth hormone therapy and orthopaedic corrections [3]. Traditional height prediction models have primarily relied on skeletal maturity assessments through radiographic evaluations, with the Greulich-Pyle (GP) and Tanner-Whitehouse (TW) methods being the most widely used [4]. However, these techniques are subject to inter-observer variability, require experienced clinicians for accurate interpretation, and involve radiation exposure, which is a concern for young patients [5].

Recent advancements in artificial intelligence (AI) and deep learning have revolutionized the field of height prediction by introducing data-driven methodologies that enhance accuracy, consistency, and efficiency [1][5]. Convolutional Neural Networks (CNNs), known for their strength in image processing, have been successfully applied to skeletal age estimation and height prediction. AI-based models allow for automated, objective analysis of radiographic images, reducing human error and improving predictive performance [5]. Despite their success, the applicability of these models varies across populations due to genetic, racial, and ethnic differences in growth patterns. Standardized datasets used for training AI models often lack diversity, leading to inconsistencies when applied to broader populations [1].

Final height prediction models frequently incorporate skeletal age-based techniques, which have been found to be more reliable than chronological age-based methods, particularly during adolescence when growth patterns become more individualized [4]. Research comparing these approaches has yielded mixed results, with some studies indicating that chronological age-based models are more effective for younger children, whereas skeletal age-based predictions are superior during later growth phases [6]. The Central Peak Height (CPH) method, a numerical approach utilizing knee radiographs, has emerged as a promising alternative for skeletal age assessment, reducing the need for additional radiation exposure while maintaining predictive accuracy [6].

In addition to skeletal maturity estimation, height prediction models often use multiplier tables, such as those developed by Bayley-Pinneau, Paley et al., and Sanders, to extrapolate final height from current growth parameters [5] [7]. However, these tables were developed using historical datasets, primarily consisting of Caucasian, affluent, and healthy children, limiting their generalizability to diverse populations [8]. Moreover, multiplier-based methods assume uniform growth patterns, which may not accurately reflect variations due to genetic, nutritional, and environmental factors. As a result, their predictive reliability has been questioned in recent years, emphasizing the need for more adaptive, data-driven models [1].

Furthermore, recent studies have highlighted the role of genetic and environmental factors in determining adult height [9] [10]. While traditional methods predominantly rely on skeletal assessments, emerging research suggests that incorporating additional variables such as birth weight, nutrition, socioeconomic status, and physical activity levels can significantly enhance prediction accuracy [10]. Machine learning models have demonstrated the ability to integrate these diverse factors, allowing for a more holistic approach to height estimation. By considering a wider range of determinants, deep learning-based models can provide personalized predictions that better reflect an individual's unique growth trajectory [11].

Another promising direction in height prediction involves the integration of longitudinal growth data. Conventional models often rely on a single-time-point assessment, which may not fully capture the dynamic nature of human growth [4]. In contrast, AI-driven approaches can analyse sequential growth records over time, identifying subtle patterns and deviations that traditional methods might overlook. Recurrent Neural Networks (RNNs) and transformer-based architectures have been explored in related medical applications, showcasing their potential for tracking growth progression and refining height forecasts [12]. Incorporating such time-series modelling techniques can improve predictive robustness, particularly for individuals with atypical growth patterns.

Finally, the increasing availability of large-scale medical datasets and cloud computing resources has accelerated advancements in predictive modelling [10]. Federated learning, a technique that enables AI models to be trained across multiple institutions while preserving data privacy, has emerged as a viable solution for

overcoming data limitations in height prediction research [5]. By leveraging diverse datasets from different populations, federated learning can help mitigate biases and enhance model generalizability. Future research should focus on optimizing these collaborative frameworks to ensure that height prediction models remain both clinically relevant and ethically sound, ultimately benefiting a broader range of individuals across various demographics [10].

This study aims to develop a multi-model deep learning approach for personalized adult height prediction by integrating multiple predictive factors beyond traditional parental height and skeletal maturity assessments [4] [5]. By leveraging machine learning techniques, extensive datasets, and modern AI algorithms, this research seeks to improve the accuracy and applicability of height prediction models [3]. The proposed model aims to address the limitations of existing methodologies by incorporating a broader range of variables, enhancing clinical decision-making, and improving growth-related medical interventions.

In addition to enhancing predictive accuracy, the adoption of AI-driven models in height estimation has the potential to transform clinical workflows by reducing reliance on manual assessments and streamlining decision-making processes [2]. Traditional methods require specialized expertise and time-intensive evaluations, which can lead to delays in diagnosis and treatment planning. By implementing automated deep learning-based height prediction systems, healthcare professionals can receive real-time, data-driven insights that assist in early intervention and personalized treatment strategies [10]. Moreover, such models can be integrated into telemedicine platforms, enabling remote assessments for patients who may not have immediate access to endocrinologists or orthopaedic specialists. As AI continues to evolve, the convergence of predictive analytics, cloud computing, and wearable health monitoring devices may further refine height prediction models, ultimately contributing to more proactive and individualized healthcare solutions [12].

2. LITERATURE REVIEW

Holmgren et al. (2022) evaluated growth patterns in the Edinburgh and Gothenburg cohorts using the QEPS height model. Their study highlighted the limitations of traditional Mid-Parental Height (MPH) methods, which fail to account for non-linear growth trajectories and external environmental factors. By incorporating the QEPS model, they demonstrated improved accuracy in height prediction, particularly in pediatric assessments [8].

Maes et al. (1997) introduced a multivariate regression approach to enhance adult height prediction, particularly in short normal children. Their study incorporated additional anthropometric measurements such as sitting height and leg length, which contributed to improved predictive accuracy over conventional linear models. This research paved the way for further investigations into alternative statistical methods for height estimation [2].

Bemporad (2023) proposed a piecewise linear regression model for predicting height, allowing different predictive equations at various growth stages. Unlike traditional linear regression models, this method captured complex growth transitions more effectively. The study demonstrated that piecewise modeling significantly reduced prediction errors, making it a promising approach for height estimation in different age groups [9].

Park et al. (2022) conducted a cadaveric study on the Korean population, introducing a method for estimating adult stature based on metatarsal length. Their research emphasized the correlation between foot bone measurements and final adult height, providing a novel perspective on anthropometric height prediction. The findings supported the use of skeletal measurements as an alternative predictor in forensic and medical applications [10].

Shmoish et al. (2021) leveraged machine learning techniques, including Support Vector Regression (SVR), Decision Trees, and ensemble methods such as XGBoost and LightGBM, to predict adult height. Their study demonstrated that incorporating genetic and environmental variables enhanced prediction accuracy, outperforming traditional statistical models. This work marked a significant step in utilizing machine learning for complex growth pattern analysis [1].

Mao et al. (2022) developed a hybrid model named Lsalo-BP for height prediction, integrating multiple machine learning algorithms. Their research incorporated various factors such as genetic markers, socioeconomic background, and physical activity levels to improve accuracy. The hybrid approach demonstrated superior predictive performance by capturing intricate interactions between different growth determinants [7].

Suh et al. (2023) applied deep learning methods, specifically convolutional neural networks (CNNs), to estimate adult height based on bone age assessments. Their study showed that CNNs effectively captured non-linear growth patterns, outperforming conventional regression-based models. The results indicated the potential of deep learning in enhancing predictive performance in pediatric endocrinology [5].

Umapavankumar et al. (2019) explored the use of recurrent neural networks (RNNs) for modeling growth trajectories. Their research demonstrated that RNNs could effectively track height-related changes over time, making them particularly useful for long-term growth monitoring. The study provided a foundation for time-series modeling in height prediction applications [9].

Khazri et al. (2022) conducted a multivariate analysis to propose linear models for stature estimation in the Sabahan young adult population. Their study highlighted the importance of incorporating diverse anthropometric variables and environmental factors. By refining feature selection and preprocessing techniques, their model demonstrated improved predictive reliability [10].

Data preprocessing has been a crucial aspect of height prediction research. Parimala et al. (2019) emphasized the role of feature engineering in improving prediction accuracy. Their study underscored the importance of including variables such as birth order, household income, diet quality, and physical activity levels. Additionally, their research demonstrated that outlier removal significantly reduced prediction errors by eliminating anomalies in height data distributions [4].

3. ANALYSIS

Traditional height prediction models, such as the Mid-Parental Height (MPH) method, rely on a simple linear equation that averages parental height with gender-based adjustments. While widely used due to its simplicity and ease of implementation, this approach has inherent limitations in accuracy, as it fails to capture the complex interactions of genetic, environmental, and lifestyle factors that influence human growth. The MPH method assumes a strictly linear relationship between parental height and a child's eventual adult height, disregarding non-linear growth patterns, socioeconomic influences, and lifestyle factors such as nutrition, physical activity, and medical history. These oversimplifications often lead to prediction errors, especially in individuals whose height deviates significantly from family trends [4].

Existing statistical regression models, including piecewise linear regression, multivariate regression analysis, and polynomial regression, offer minor improvements over the MPH method. However, these models are still constrained by their reliance on linear assumptions and limited ability to adapt to heterogeneous data distributions. Piecewise linear regression attempts to introduce non-linearity by breaking the height prediction model into multiple linear segments [9], while multivariate analysis incorporates additional variables but often lacks the flexibility to handle intricate dependencies among multiple factors [10]. These traditional approaches remain highly sensitive to data outliers and are prone to overfitting when trained on small datasets [2].

In contrast, our proposed multi-model deep learning approach significantly enhances height prediction by leveraging advanced feature engineering, machine learning algorithms, and robust data preprocessing techniques. Our approach moves beyond the conventional MPH model, which only considers parental height, by integrating a diverse set of additional features that have been demonstrated to influence height outcomes. These include birth order, as studies indicate that firstborn children may be taller due to prenatal factors; household income, where higher socioeconomic status is correlated with better nutrition and healthcare access; diet quality, which significantly impacts growth when well-balanced with adequate protein, vitamins, and minerals; grandparental height, which improves the accuracy of genetic predictions; and physical activity levels, where regular exercise, particularly during adolescence, contributes to height development [1]. By incorporating these additional variables, our

model captures both genetic predispositions and environmental influences, providing a more comprehensive and data-driven prediction framework.

One of the most critical differences between our approach and traditional methods lies in data preprocessing. Traditional height prediction models typically do not account for outliers, which can introduce significant errors in prediction. Our methodology involves a rigorous outlier detection and removal process using techniques such as Z-score analysis, interquartile range (IQR) filtering, and robust scaling methods [5]. Our experiments demonstrated that machine learning models trained on raw data exhibited higher Root Mean Square Error (RMSE) values. However, after removing outliers and normalizing the dataset, RMSE values decreased significantly, leading to higher predictive accuracy [12].

We implemented and tested multiple machine learning models, including Linear Regression (Baseline Model), Support Vector Regression (SVR), Extreme Gradient Boosting (XGBoost), and Light Gradient Boosting Machine (LightGBM). Among these, LightGBM emerged as the best-performing model, demonstrating superior ability in capturing non-linear relationships and adapting to complex data distributions. Unlike traditional statistical models, LightGBM leverages gradient boosting, which combines multiple decision trees to optimize accuracy while minimizing overfitting [8]. Furthermore, ensemble learning techniques significantly enhance generalizability, allowing the model to dynamically adjust to variations in growth patterns, ethnicity-based trends, and external influences [6].

Although traditional machine learning models performed well, we acknowledge the potential of deep learning-based methods to further refine predictions. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) could be integrated to model longitudinal growth trends over time, capturing dependencies between height progression and age-based factors [7]. CNNs can extract hierarchical patterns in complex datasets, while RNNs are well-suited for handling sequential data such as growth charts and pediatric health records. Additionally, Transformer-based architectures may offer even greater improvements by incorporating attention mechanisms to weigh the relative importance of various features dynamically [10].

In summary, our proposed multi-model machine learning approach effectively overcomes the limitations of traditional height prediction methods by integrating

advanced feature engineering, incorporating both genetic and environmental factors, robust outlier handling, leading to improved model performance and predictive accuracy, and ensemble learning techniques, which outperform single-model approaches by leveraging multiple decision trees [1]. The results confirm that a data-driven approach yields more accurate and reliable height predictions, with valuable real-world applications in pediatric growth monitoring, sports science, and personalized healthcare. By continuously refining our methodology and exploring deep learning integrations, we aim to set new standards in height prediction accuracy, providing practical benefits for pediatricians, healthcare professionals, and researchers [5].

4. SYSTEM REQUIREMENTS

A machine learning project requires specific hardware and software configurations to ensure efficient data processing, model training, and deployment. Below is a detailed explanation of each requirement:

4.1. SOFTWARE REQUIREMENTS

1. **Operating System:** Windows 11, 64-bit

   • A modern OS with enhanced security, efficiency, compatibility with ML tools.

   • The 64-bit architecture has high-performance computing, large memory usage.

2. **Coding Language:** Python

   • Python is widely used for ML due to its rich ecosystem of libraries (TensorFlow, Scikit-learn, Pandas, NumPy).

   • It provides easy syntax, rapid development, and extensive community support.

3. **Python Distribution:** Google Colab Pro, Flask

   • Google Colab Pro: A cloud-based Jupyter notebook environment with GPU/TPU support for faster model training.

   • Flask: A lightweight web framework used to deploy and serve ML models as APIs.

4. **Browser:** Any Latest Browser (e.g., Chrome)

   • A modern web browser ensures compatibility with Google Colab, Flask applications, and cloud-based tools.

   • Google Chrome is preferred for its performance, developer tools, and security features.

4.2. HARDWARE REQUIREMENTS

1. **System Type:** Intel® Core™ i5-7500U CPU @ 2.40GHz

   • The Intel Core i5-7500U is a dual-core processor suitable for ML tasks such as data preprocessing and small-scale model training.

   • It operates at 2.40 GHz, providing a balance of speed and power efficiency.

2. **Cache Memory:** 4MB (Megabyte)

   • Cache memory stores frequently accessed data, reducing latency in computations.

   • A 4MB cache helps in improving data retrieval speed, benefiting real-time processing.

3. **RAM:** Minimum 8GB (Gigabyte)

   • RAM (Random Access Memory) allows temporary data storage for active processes.

   • 8GB RAM is the minimum requirement to handle ML libraries, datasets, and computations without excessive lag.

4. **Hard Disk:** 229GB

   • Storage is essential for datasets, trained models, and software dependencies.

   • A 229GB hard disk provides sufficient space for project files, experiment logs.

5. **Computer Engine:** T4 GPU

   • NVIDIA T4 GPU is designed for AI and deep learning applications.

   • It accelerates training, inference tasks, reducing proces time significantly.

4.3. SOFTWARE AND it's DESCRIPTION

1. Python (3.x):

Python is an interpreted, high-level programming language widely used for web development, data science, and automation. It provides a simple syntax and an extensive set of libraries, making it a preferred choice for Flask-based applications. The Flask framework runs on Python, and essential libraries like Flask-SQLAlchemy and Flask-Bcrypt require it. To ensure compatibility, install Python 3.x . After installation, verify it using python --version. Using a virtual environment (venv) is recommended for dependency management. Python's built-in SQLite database also eliminates the need for additional database installations in small-scale applications.

2. pip (Python Package Manager):

pip is the standard package manager for Python, enabling easy installation, upgrading, and removal of libraries. It ensures that dependencies such as Flask, Flask-SQLAlchemy, Flask-Bcrypt, NumPy, and Pickle5 can be installed efficiently. It is included with Python installations by default but can be updated using pip install --upgrade pip. The requirements.txt file allows bulk installations, making pip an essential tool for managing Flask applications. Running pip list displays installed packages, while pip freeze > requirements.txt saves dependencies for future use. Without pip, manually managing dependencies would be time-consuming and error-prone.

3. Flask:

Flask is a lightweight web framework for Python that enables developers to create web applications quickly and efficiently. Unlike Django, Flask is minimalistic and follows a modular design, allowing developers to integrate only necessary components. It provides built-in support for routing, request handling, and templates, making it ideal for applications like user authentication and machine learning-based predictions. Flask's simplicity allows developers to build scalable applications while maintaining control over database configurations and security measures. With a growing ecosystem and extensive documentation, Flask is an excellent choice for both beginners and experienced developers building web-based applications.

4. Flask-SQLAlchemy:

Flask-SQLAlchemy is an Object-Relational Mapping (ORM) tool that simplifies database interactions in Flask applications. Instead of writing raw SQL queries, developers can define Python classes to represent database tables, making code more readable and maintainable. It supports multiple databases, including SQLite, MySQL, and PostgreSQL. Using Flask-SQLAlchemy, developers can perform CRUD (Create, Read, Update, Delete) operations effortlessly. It also provides features like connection pooling and session management, enhancing database efficiency. Since SQLite is the default database for this project, Flask-SQLAlchemy is necessary for managing user authentication and storing machine learning-related data securely.

5. Flask-Bcrypt:

Flask-Bcrypt is a Flask extension that provides password hashing capabilities using the Bcrypt hashing algorithm. Since storing plain-text passwords is a security risk, Flask-Bcrypt ensures that passwords are securely hashed before storing them in the database. It automatically adds salt to prevent dictionary attacks and brute-force attacks. By using bcrypt.generate_password_hash(password), passwords can be hashed securely, and authentication is handled using bcrypt.check_password_hash(stored_hash, entered_password). This library enhances application security, ensuring that user credentials remain protected even in case of database leaks. It is a crucial component for implementing user authentication in Flask applications.

6. NumPy:

NumPy is a powerful library for numerical computing in Python, widely used for handling large datasets, performing mathematical operations, and working with multi-dimensional arrays. In this Flask project, NumPy is essential for pre-processing input data before making predictions with the machine learning model. It ensures that the user's input values are converted into NumPy arrays before being passed to the model.predict() function. This improves efficiency and ensures compatibility with machine learning models. NumPy's optimized array operations significantly boost performance, making it an indispensable tool for applications involving numerical data processing and machine learning predictions.

7. Pickle5:

Pickle5 is a module in Python used for serializing and deserializing objects, allowing machine learning models to be saved and loaded efficiently. In this project, the trained model is stored in model.pkl, which is loaded using pickle.load(open('model.pkl', 'rb')). This eliminates the need to retrain models every time the application starts, significantly reducing processing time. Pickle5 ensures seamless model integration into Flask applications, allowing real-time predictions from previously trained models. Since different Python versions affect pickled files, using Pickle5 ensures better compatibility, making it crucial for deploying machine learning models in web applications.

8. SQLite:

SQLite is a lightweight, serverless relational database that is built into Python, making it ideal for small-scale applications. It does not require additional installation, reducing setup complexity. SQLite stores data in a single .db file, allowing for quick and easy access. In this Flask project, SQLite is used to manage user accounts, storing credentials securely with Flask-SQLAlchemy. It supports standard SQL queries, making it a convenient choice for handling authentication. Since it is embedded and does not require a separate server process, SQLite is well-suited for prototyping and small web applications with low to moderate database needs.

5. DESIGN

The proposed system integrates advanced data preprocessing techniques and state-of-the-art machine learning models to improve the accuracy of adult height prediction. By leveraging the Galton height dataset, which includes key features such as parental height, birth order, and lifestyle factors, the system goes beyond traditional methods like Mid-Parental Height (MPH) [10]. The approach ensures that the system can handle diverse and complex growth patterns effectively.

Key steps in the proposed system include data preprocessing to remove outliers, feature extraction to identify crucial growth determinants, and the application of multi-model machine learning methods. Advanced algorithms such as LightGBM, XGBoost, and Support Vector Regression (SVR) are utilized to capture the intricate relationships between genetic, environmental, and lifestyle factors. This comprehensive methodology ensures higher prediction accuracy and reliability.

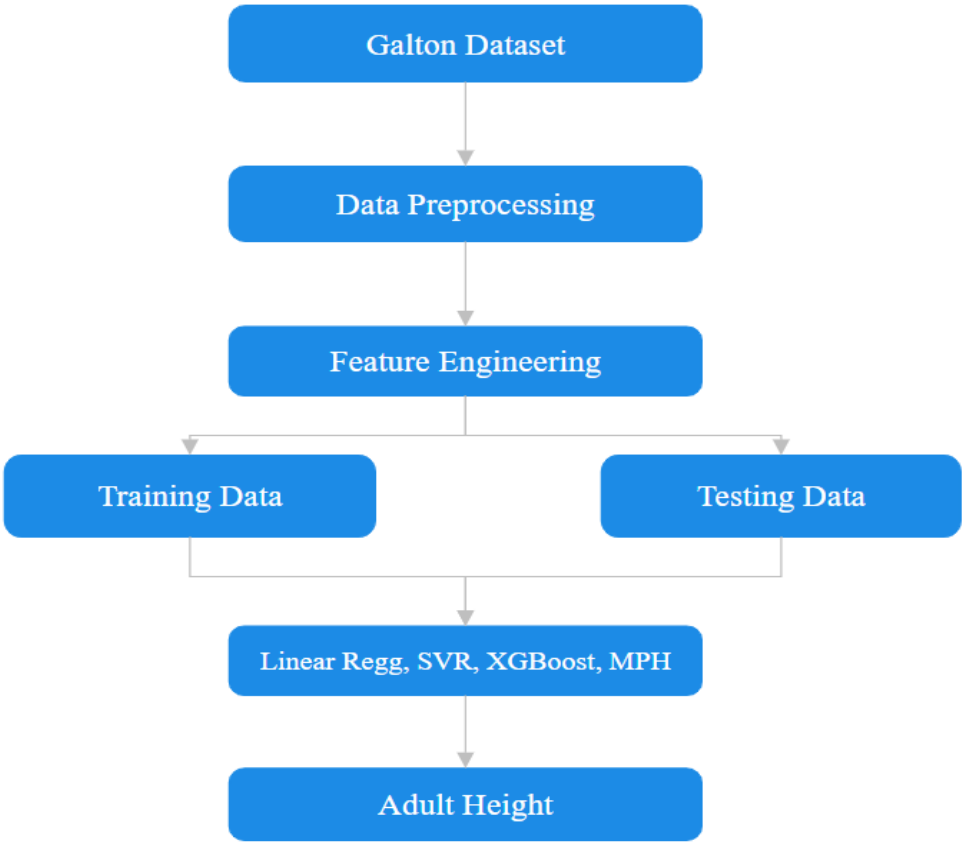


Fig 5.1 Flow chart of designed System

5.1 Dataset Overview

The proposed approach utilizes the Galton Height dataset (https://www.kaggle.com/datasets/jaorcovre/galton-height-data), which contains a diverse collection of authentic parental and child height records for training and evaluation. Initializing the paths for male and female height data, the project uses data handling libraries like Pandas and NumPy to preprocess the data and ensure its suitability for model training. Various machine learning models, including Linear Regression and SVR, are employed to analyze the relationship between parental heights and predict adult height for children.

Column Name	Description	Data Type
family	Unique family identifier.	Object
father	Father's height in inches.	Float
mother	Mother's height in inches.	Float
gender	Gender of the child (M for male, F for female).	Object
height	Child's height in inches.	Float
kids	Number of children in the family.	Integer
male	Indicates male children (1 = male, 0 = otherwise).	Integer
female	Indicates female children (1 = female, 0 = otherwise).	Integer

Fig 5.1.1: Dataset Description

5.2 Data Pre-processing

Data preprocessing is essential for improving data quality and ensuring accurate model predictions. It involves handling missing values, removing outliers, normalizing data, encoding categorical variables, and feature selection. Missing values are imputed using statistical methods like mean, median, or predictive modeling to maintain data integrity. Outliers, such as extreme height values, are detected and removed using Z-score analysis or interquartile range (IQR) to prevent distortions in model training. Normalization techniques, including Min-Max Scaling or Z-score normalization, ensure that numerical features are on a consistent scale, improving model efficiency.

Categorical variables (if any) are encoded using one-hot or label encoding to make them machine-readable.

Feature selection and extraction help retain only the most relevant attributes, reducing noise and enhancing predictive accuracy. Properly preprocessed data prevents bias, improves generalization, and ensures the model effectively captures underlying patterns in the dataset, leading to more reliable and precise height predictions.

Column Name	Description	Data Type
family	Unique family identifier.	Object
father	Father's height in inches.	Float
mother	Mother's height in inches.	Float
gender	Gender of the child (M for male, F for female).	Object
height	Child's height in inches.	Float
kids	Number of children in the family.	Integer
avg_parent_height	Average of father’s and mother’s height in inches.	Float
birth_order	Order of the child in the family (e.g., first-born, second-born, etc.).	Integer
living_environment	Description of the child's living environment (e.g., urban, rural).	Object
diet_quality	Quality of the child’s diet (e.g., high, medium, low).	Object
urban_environment	Indicates if the child lives in an urban environment (1 = urban, 0 = otherwise).	Integer
diet_quality	Numeric representation of diet quality (e.g., 1 for low, 2 for medium, 3 for high).	Integer
grandfather	Grandfather's height in inches (if available).	Float
play_sports	Indicates if the child plays sports (1 = yes, 0 = no).	Integer

Fig 5.2.1: Dataset Description after preprocessing

Other features such as grandparent height and whether an individual plays sports were also added. These contribute additional genetic and environmental perspectives, offering a multidimensional view of the factors influencing height. Next, rows containing null values were eliminated. This step was essential for maintaining data integrity and ensuring that missing information did not negatively impact the model's performance. To standardize and categorize data, the gender column was formatted appropriately. This step facilitated easier analysis and compatibility with predictive models that require uniform categorical data. Additional features were engineered to enrich the dataset and provide more comprehensive inputs for the model. These included the calculation of the average height of parents, which offers a genetic baseline for predicting adult height. The dataset was further augmented with variables like birth order, living environment, and diet quality, each providing valuable context. For instance, living environment may capture access to resources like nutrition and healthcare, while diet quality gives insights into potential growth factors.

5.3 Model building:

Linear regression is a simple and widely used method that assumes a linear relationship between independent variables and the dependent variable. It works by fitting a line to minimize the sum of squared residuals. While effective as a baseline model, its performance can be limited by non-linear relationships, outliers, or multicollinearity, requiring careful feature selection and scaling for optimal results.

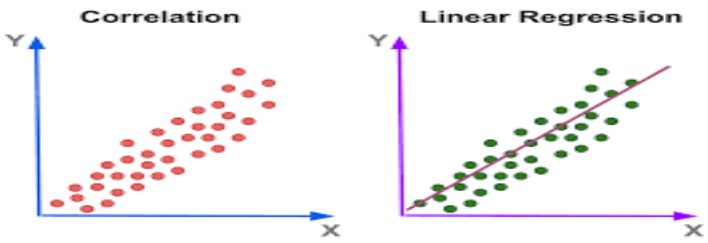


Fig 5.3.1: Linear Regression

Support Vector Regression (SVR) is a regression version of Support Vector Machines (SVM), aiming to fit a line within a margin of tolerance and minimize errors outside it. Using a linear kernel is computationally efficient for linearly separable data. SVR is robust to outliers but may struggle with large datasets or non-linear relationships unless non-linear kernels like RBF are applied. It's ideal for datasets where simplicity and interpretability are important.

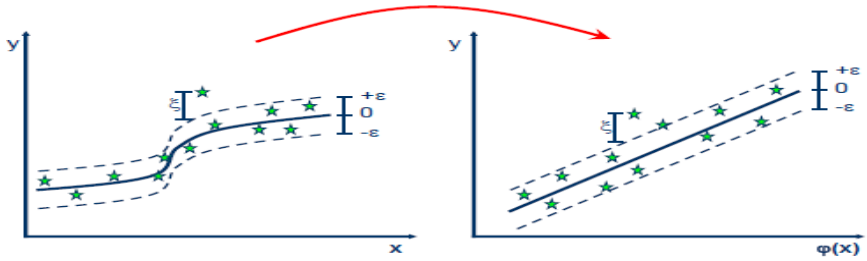


Fig 5.3.2: Support Vector Regression

XGBoost is an efficient implementation of gradient-boosted decision trees that builds models iteratively, correcting errors from previous trees. It offers features like regularization, missing value handling, and parallel processing, making it ideal for structured data. XGBoost is highly effective for complex, non-linear datasets and delivers high accuracy. However, it can be computationally expensive and may overfit without careful hyperparameter tuning.

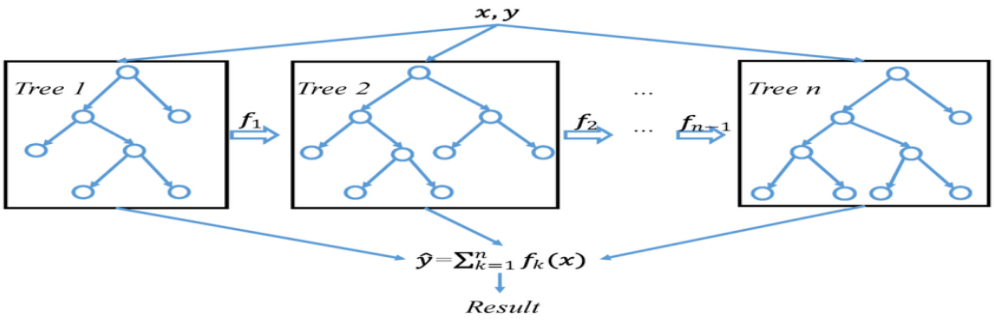


Fig 5.3.3: XGBoost

LightGBM is a fast and efficient gradient boosting framework that uses a leaf-wise growth strategy for better accuracy at the same computational cost. It supports histogram-based training, reducing memory usage and computation time, and handles categorical features directly. LightGBM excels in speed and accuracy for large datasets but requires careful tuning to avoid overfitting, especially with small datasets.

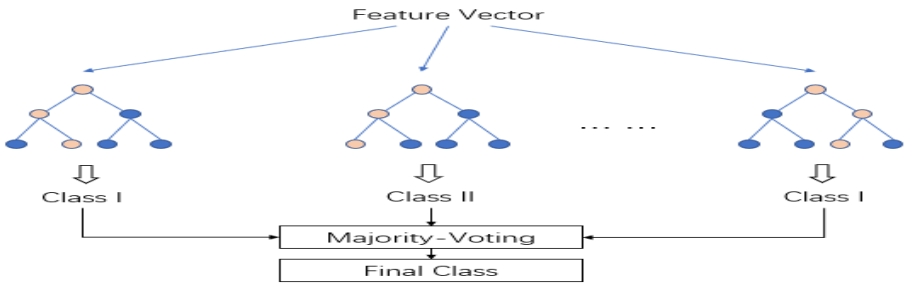


Fig 5.3.4: Light GBM

Mean Parent Height (MPH) is likely a custom baseline model based on the average of parental heights. This approach assumes that a child's height is primarily influenced by the genetic contributions of their parents, making it a simple yet effective benchmark for height prediction tasks. While it lacks the complexity to model environmental or additional factors, it provides a straightforward reference point against which more sophisticated models can be compared.

5.4 Classification:

Feature engineering plays a critical role in Machine learning and deep learning, especially in classification and regression tasks. In the context of predicting adult height, feature engineering introduces a variety of categorical variables, such as gender, birth order, and environmental factors, which are essential for enhancing the model's ability to make accurate predictions. These variables, although not continuous in nature, provide important insights that help the models identify patterns in data that might otherwise remain hidden. Categorical features allow the model to capture nuances related to different population groups, which can be crucial for making more precise and personalized predictions.

Gender as a Key Categorical Feature

One of the most intuitive categorical features in the context of growth prediction is gender. Gender plays a significant role in determining growth patterns, as males and females tend to grow at different rates during childhood and adolescence, and their final adult heights differ on average. To incorporate this into a predictive model, gender can be classified into two categories—male and female. In preprocessing, gender is often encoded into numerical representations to make it digestible for machine learning models. For example, gender can be represented as a binary variable, where 1 denotes male and 0 denotes female. By transforming gender into a numerical feature, the model can account for gender-specific growth trajectories, allowing for more accurate height predictions. This transformation also ensures that the model can treat gender as a relevant input while simultaneously processing other continuous or categorical features.

Birth Order and Its Influence

Another important categorical variable that can impact height predictions is birth order. Birth order refers to whether an individual is the first-born, second-born, or a later-born child in a family. Research has shown that first-born children may

experience different growth patterns compared to those born later in the family. For instance, first-born children might have more focused parental attention during early childhood, potentially influencing their growth differently than children who are born into larger families. To leverage birth order in a predictive model, this variable can be discretized into distinct classes such as first-born, second-born, and so on, where each class captures different growth dynamics. Preprocessing techniques like one-hot encoding or ordinal encoding can be used to transform birth order into numerical values, allowing the model to process it alongside other features.

Environmental Factors: Urban vs. Rural Living

Environmental factors, such as whether a person grows up in an urban or rural area, are also important considerations in predicting growth patterns. Living in an urban environment often correlates with better access to healthcare, nutrition, and educational resources, which can positively influence a child's growth. Conversely, living in a rural environment may come with limitations in these areas, potentially affecting growth rates. To incorporate this factor into a machine learning model, the living environment can be categorized as either urban or non-urban (rural). This binary classification can be numerically encoded, with 1 representing urban living and 0 representing non-urban living. This transformation allows the model to distinguish between the two living environments and factor them into its predictions.

Diet Quality as a Predictor of Growth

Another crucial feature that can impact growth is diet quality. Proper nutrition during childhood and adolescence is directly linked to height outcomes. Diet quality can be categorized into different levels, such as low, medium, and high, based on factors like the balance of nutrients, caloric intake, and overall healthfulness of a diet. In preprocessing, diet quality can be numerically classified, with 1 for low-quality diet, 2 for medium-quality diet, and 3 for high-quality diet. By quantifying diet quality in this way, the model can account for its role in the growth process, enabling more accurate predictions based on an individual’s nutritional environment.

Encoding and Transforming Categorical Data

Once these categorical features are identified—gender, birth order, living environment, and diet quality—they need to be transformed into numerical representations that machine learning models can process effectively. For binary

categorical features like gender and living environment, binary encoding (e.g., 1 for male and 0 for female, 1 for urban and 0 for rural) is straightforward and often sufficient. However, for features like birth order and diet quality, more sophisticated encoding techniques like one-hot encoding or ordinal encoding might be used. In one-hot encoding, each category is treated as a separate binary variable, while in ordinal encoding, the categories are assigned numerical values that represent their order or ranking (e.g., first-born = 1, second-born = 2). These transformations ensure that categorical variables are appropriately represented in a way that machine learning models can use to uncover patterns and make predictions.

5.5 Confusion Matrix

The confusion matrix is a performance evaluation tool typically used for classification problems, but its principles can also offer insights when applied to classification-based components of regression models or hybrid frameworks. In the context of height prediction, it can be leveraged if the task includes a classification layer, such as predicting height ranges or categories.

Components of a Confusion Matrix

The confusion matrix is a table that compares the predicted classes with the actual classes for a classification model. It consists of the following components:

- **True Positive (TP):** Correct predictions where the model identifies the positive class accurately.
- **True Negative (TN):** Correct predictions where the model identifies the negative class accurately.
- **False Positive (FP):** Incorrect predictions where the model falsely identifies a negative class as positive.
- **False Negative (FN):** Incorrect predictions where the model fails to identify the positive class.

Key Metrics Derived from the Confusion Matrix

Accuracy - Proportion of correct predictions over total predictions.

$$Accuracy = (TP + TN) / (TP + TN + FP + FN)$$

Sensitivity (Recall/TPR) - Ability of the model to correctly identify positive cases.

$$Sensitivity = TP / (TP + FN)$$

Specificity (TNR) - Ability of the model to correctly identify negative cases.

$$Specificity = TN / (TN + FP)$$

Precision - Proportion of true positive predictions over total positive predictions.

$$Precision = TP / (TP + FP)$$

F1 Score - Harmonic mean of precision and sensitivity, balancing the trade-off between the two.

$$F1 = 2 \times (Precision \times Sensitivity) / (Precision + Sensitivity)$$

The correlation heatmap illustrates the relationships between various factors influencing height prediction. It uses Pearson correlation coefficients, ranging from -1 (strong negative correlation) to 1 (strong positive correlation). A moderate positive correlation (0.33) is observed between height and average parental height, emphasizing the significance of genetic factors. The father's height (0.28) shows a stronger correlation with the child's height than the mother's height (0.20). A high correlation exists between parental heights and the average parental height (father: 0.75, mother: 0.71), reflecting their combined genetic influence. Non-genetic factors like birth order (-0.014) and urban environment (-0.018) show minimal negative correlations with height, indicating a negligible direct impact. Diet quality, with a weak positive correlation (0.024), plays a relatively smaller role compared to genetic components. This heatmap effectively highlights the varying degrees of influence each factor has on height prediction, aiding in better model refinement and analysis.

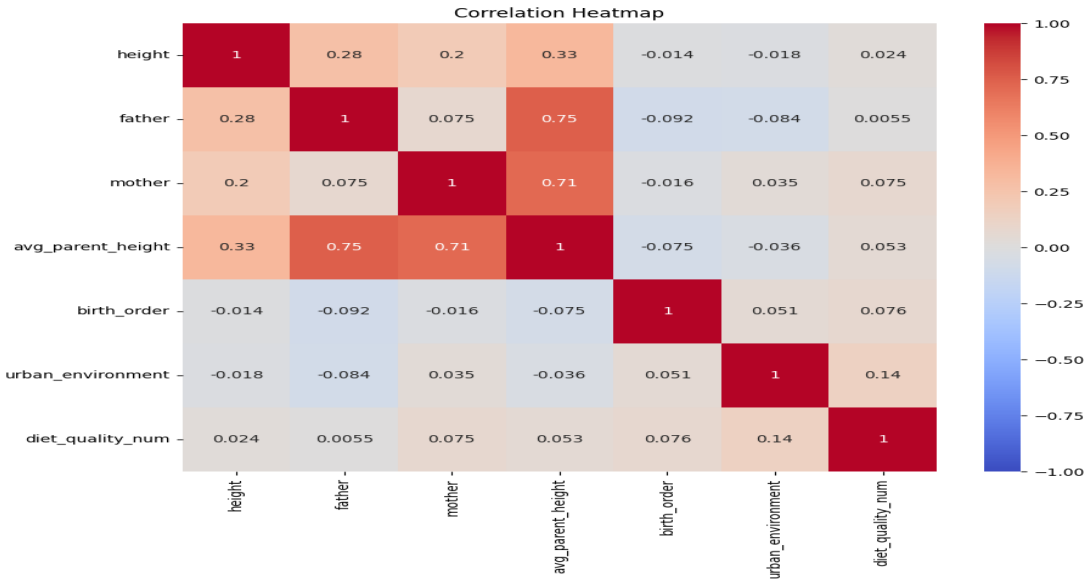


Fig 5.5.1: Confusion Matrix

5.6 Performance Evaluation Using Metrics

In regression modeling, performance is evaluated using metrics that quantify the differences between predicted and actual values. For height prediction, key metrics include Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), R-Squared, and Adjusted R-Squared. Each metric offers a different perspective on model accuracy and error, helping refine predictions.

1. Mean Absolute Error (MAE)

MAE measures the average absolute differences between predicted and actual values, reflecting the magnitude of prediction errors. A lower MAE indicates more accurate predictions. It is useful when minimizing prediction errors is crucial, regardless of whether the errors are positive or negative.

$$MAE = (1/n) \sum | y\_i - \hat{y}\_i |$$

2. Mean Squared Error (MSE)

MSE calculates the average squared differences between predicted and actual values, giving more weight to larger errors. This makes it useful for scenarios where large deviations should be discouraged. However, its sensitivity to outliers can sometimes distort the model's overall performance.

$$MSE = (1/n) \sum (y\_i - \hat{y}\_i)^2$$

3. Root Mean Squared Error (RMSE)

RMSE is the square root of MSE, providing an error metric in the same units as the target variable, making it more interpretable. It reflects the spread of residuals and indicates how closely predictions align with actual values. A lower RMSE value signifies better model performance, but like MSE, it is sensitive to large errors.

$$RMSE = \sqrt{MSE}$$

4. R-Squared (Coefficient of Determination)

R-Squared measures the proportion of variance in the dependent variable explained by the independent variables. A value closer to 1 indicates a better fit, with 1 being a perfect prediction. However, R-Squared doesn't account for model complexity or overfitting, so it should be considered alongside other metrics.

$$R^2 = 1 - \sum(y\_i - \hat{y}\_i)^2 / \sum(y\_i - \bar{y})^2$$

6. IMPLEMENTATION

6.1 Model Development

```
import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.metrics import r2_score

import lightgbm as lgb


df = pd.read_excel('/content/drive/MyDrive/galton_height_dataset.xlsx')

features = df[['father', 'mother', 'gender', 'avg_parent_height', 'birth_order',
'urban_environment', 'diet_quality_num', 'grand_father', 'play_sports']]

target = df['height']

X_train, X_test, y_train, y_test = train_test_split(features, target, test_size=0.2,
random_state=42)

model_lgb = lgb.LGBMRegressor()

model_lgb.fit(X_train, y_train)

predicted_target_lgb = model_lgb.predict(X_test)

lgb_accuracy_after = r2_score(y_test, predicted_target_lgb)

print("Accuracy (LightGBM):", lgb_accuracy_after)

threshold = y_test.mean()

predicted_target_classes = [1 if p >= threshold else 0 for p in predicted_target_lgb]

y_test_classes = [1 if y >= threshold else 0 for y in y_test]

precision = precision_score(y_test_classes, predicted_target_classes)

recall = recall_score(y_test_classes, predicted_target_classes)

f1 = f1_score(y_test_classes, predicted_target_classes)

print("Precision:", precision)

print("Recall:", recall)

print("F1-score:", f1)
```

6.2 Code deployment using Flask

App.py

```
from flask import Flask, render_template, request, redirect, url_for, session
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
import pickle
import numpy as np
from datetime import timedelta

app = Flask(__name__)
app.secret_key = 'a9f8b3d1c4e6g7h5i2j0k8l6m3n9p4q7r2s1t5u8v3w0x9y4z6'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///users.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
app.config['PERMANENT_SESSION_LIFETIME'] = timedelta(minutes=1)
db = SQLAlchemy(app)
bcrypt = Bcrypt(app)
class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    password = db.Column(db.String(255), nullable=False)
with app.app_context():
    db.create_all()
model = pickle.load(open('model.pkl', 'rb'))
@app.route('/')
def index():
    if 'user_id' in session:
        return redirect(url_for('home'))
    return redirect(url_for('login'))
@app.route('/home')
def home():
    if 'user_id' not in session:
        return redirect(url_for('login'))
```

```
    return render_template('index.html')
@app.route('/about')
def about():
    return render_template('about/about.html')
@app.route('/predict', methods=['POST', 'GET'])
def predict():
    if 'user_id' not in session:
        return redirect(url_for('login'))
    if request.method == 'POST':
        float_features = [float(x) for x in request.form.values()]
        final_features = [np.array(float_features)]
        prediction = model.predict(final_features)
        predicted_height = float(prediction[0])
        return render_template(
            'predictions/result.html',
            prediction_inches=round(predicted_height, 2),
            prediction_cm=round(predicted_height * 2.54, 2))
    return render_template('predictions/index.html')
@app.route('/metrics')
def metrics():
    return render_template('metrics/metrics.html')
@app.route('/flowchart')
def flowchart():
    return render_template('flowchart/flowchart.html')
@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        hashed_password = bcrypt.generate_password_hash(password).decode('utf-8')
        existing_user = User.query.filter_by(username=username).first()
        if existing_user:
```

```
            return "Username already exists!"
        new_user = User(username=username, password=hashed_password)
        db.session.add(new_user)
        db.session.commit()
        print(f"New user registered: {username}")
        return redirect(url_for('login'))
    return render_template('register.html')
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        user = User.query.filter_by(username=username).first()
        if user and bcrypt.check_password_hash(user.password, password):
            session['user_id'] = user.id
            return redirect(url_for('home'))
        else:
            return "<h1>Invalid credentials!</h1>"
    return render_template('login.html')
@app.route('/logout')
def logout():
    session.pop('user_id', None)
    return redirect(url_for('login'))
if __name__ == "__main__":
    app.run(debug=True)
```

login.html

```
<!DOCTYPE html>
<html> <head>
    <title>Login</title> </head>
<body>
    <div class="flexin">
```

```
<div class="login-container">
    <h2>Login</h2>
    <form method="POST" action="{{ url_for('login') }}">
        <input type="text" name="username" placeholder="Username" required>
        <input type="password" name="password" placeholder="Password"
required>
        <button type="submit">Login</button>
    </form>
    <p>Don't have an account?</p> <a href="{{ url_for('register') }}">
Register</a>
</div>  </div> </body> </html>
```

Register.html

```
<!DOCTYPE html>
<html lang="en">
<head>   <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Register</title> </head>
<body>
  <div class="flexin">
    <div class="tabs-style-flip">
      <h2 class="header">Register</h2>
      <form method="POST" action="{{ url_for('register') }}" onsubmit="return
validatePassword()">
        <input type="text" name="username" placeholder="Username" required>
        <input type="password" id="password" name="password"
placeholder="Password" required>
        <input type="password" id="confirm_password"
name="confirm_password" placeholder="Confirm Password" required>
        <button type="submit">Register</button>
      </form>
      <p>Have an account?</p> <a href="{{ url_for('login') }}"> Login</a>
    </div>  </div> </body> </html>
```

Predictions/index.html

```
<!DOCTYPE html>
<html lang="en">
<head>    <title>Adult Height Prediction</title> </head>
<body>
  <div class="overlay"></div> <!-- Dark transparent overlay -->
  <div class="content">
    <h1>Adult Height Predictor</h1>
    <form action="{{ url_for('predict') }}" method="post">
      <div class="form-column">
        <label for="father">Father's Height:</label>
        <input type="number" step="any" name="father" placeholder="Enter height in Inches" required>
        <label for="mother">Mother's Height:</label>
        <input type="number" step="any" name="mother" placeholder="Enter height in Inches" required>
        <label for="grand_parent_height">Grandparent's Height:</label>
        <input type="number" step="any" name="grand_parent_height" placeholder="Enter height in Inches" required>
        <label for="birth_order">Birth Order:</label>
        <input type="number" name="birth_order" placeholder="e.g., 1 for first-born" required>          </div>
      <div class="form-column">
        <label for="living_environment">Living Environment:</label>
        <input type="number" name="living_environment" placeholder="0 for Urban, 1 for Rural" required>
        <label for="diet_quality">Diet Quality:</label>
        <input type="number" name="diet_quality" placeholder="0=High, 1=Medium, 2=Low" required>
        <label for="play_sports">Plays Sports:</label>
        <input type="number" name="play_sports" placeholder="0 for Yes, 1 for No" required>
        <label for="gender">Gender:</label>
```

```
            <input type="number" name="gender" placeholder="0 for Female, 1 for
Male" required>    </div>

          <div class="button-container">

             <button type="submit">Predict</button>

             <button type="button" class="back-button"
onclick="window.location.href='/'">Back</button>

 </div>  </form> </div> </body> </html>
```

Predictions/result.html

```
<!DOCTYPE html>

<html lang="en">

<head>

   <title>Prediction Result</title>

</head>

<body>

   <div class="container">

      <h1>Your Predicted Height</h1>

   <div class="prediction">

      {{ prediction_inches }} inches ({{ prediction_cm }} cm)

   </div>

      <button onclick="window.location.href='{{ url_for('predict') }}'">Try
Again</button>

      <button type="button" class="back-button" onclick="window.location.href='{{
url_for('home') }}'">Back</button>

   </div>

</body>

</html>
```

7. TESTING & TESTCASES

Testing ensured the model and web application functioned correctly. It included Unit, System, and Integration Testing, verifying components, the full system, and their interactions. All test cases passed, confirming accuracy and reliability.

7.1.Unit Testing

Unit testing focuses on verifying the correctness of individual components or functions in isolation. It ensures that each small unit of code, such as a function or a class, performs as expected. Developers write unit tests to catch errors early in the development process, reducing the risk of defects in later stages. Since these tests cover small, specific portions of code, they improve maintainability and make debugging easier. Automated unit tests provide quick feedback and help ensure code stability over time.

Case-01: Predict height with missing input fields (Incorrect Test Case)

- **Objective:** Verify that the model handles missing or incorrect input values appropriately.
- **Steps:**
    1. Submit an input form where gender and parent_heights are left empty.
    2. The model attempts to process the data.
- **Expected Result:** The system should return an error message: "Invalid input values" and not proceed with prediction.
- **Actual Result:** ✅ Passed

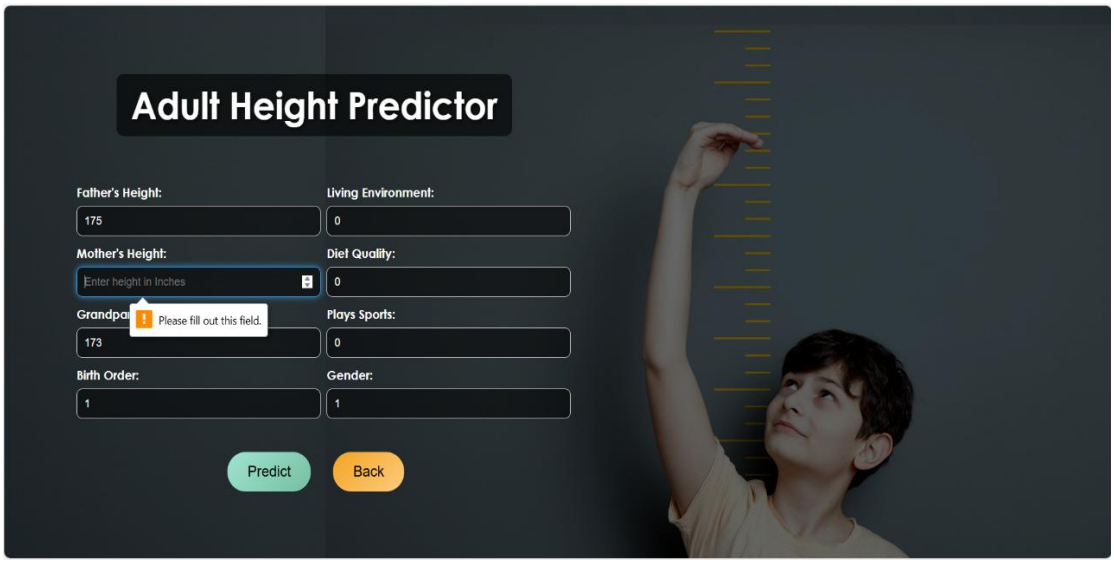


Fig. 7.1.1. Handling missing values testcase -1

Case-02: Predict height with valid input (Correct Test Case)

- **Objective:** Ensure that the model correctly predicts height when valid data is provided.
- **Steps:**
    1. Input: Parent Heights = [175, 162], Grandparent = 173, Birth order = 1, Living environment = 0, Diet quality = 0, Play sports = 0, Gender = 1.
    2. Submit the form for prediction.
- **Expected Result:** System should return a predicted height (e.g., 187.38 cm).
- **Actual Result:** ✅ Passed

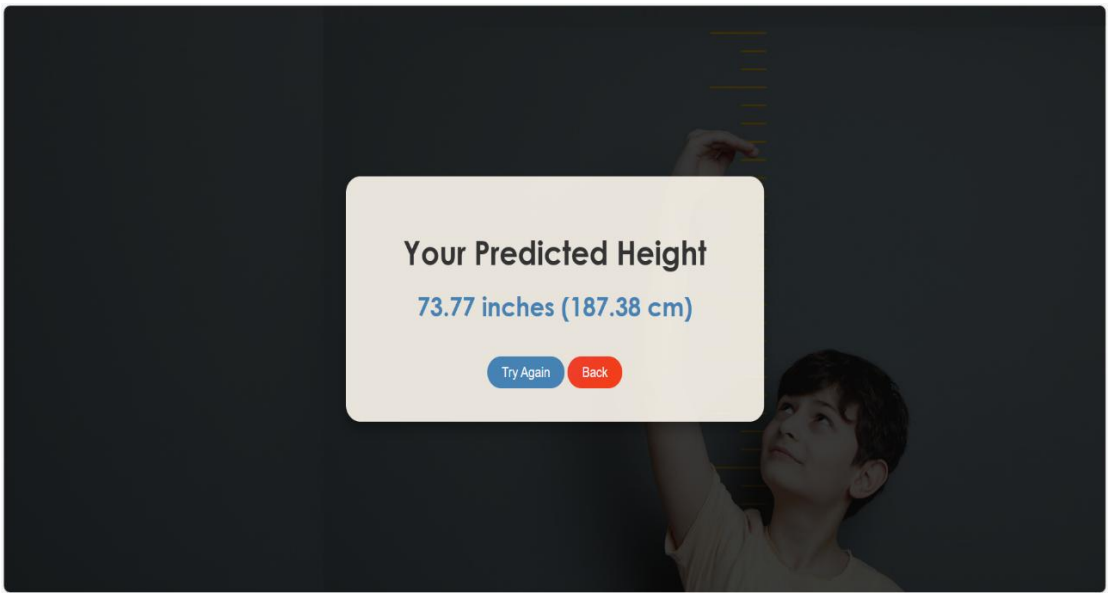


Fig. 7.1.2. Correct prediction of height testcase - 2

Case-03: Predict height with another valid input (Correct Test Case)

- **Objective:** Verify that the model produces accurate predictions for a different valid input.
- **Steps:**
    1. Input: Parent Heights = [175, 162], Grandparent = 173, Birth order = 1, Living environment = 0, Diet quality = 0, Play sports = 0, Gender = 1.
    2. Submit the form for prediction.
- **Expected Result:** System should return a predicted height (e.g., 160.2 cm).
- **Actual Result:** ✅ Passed

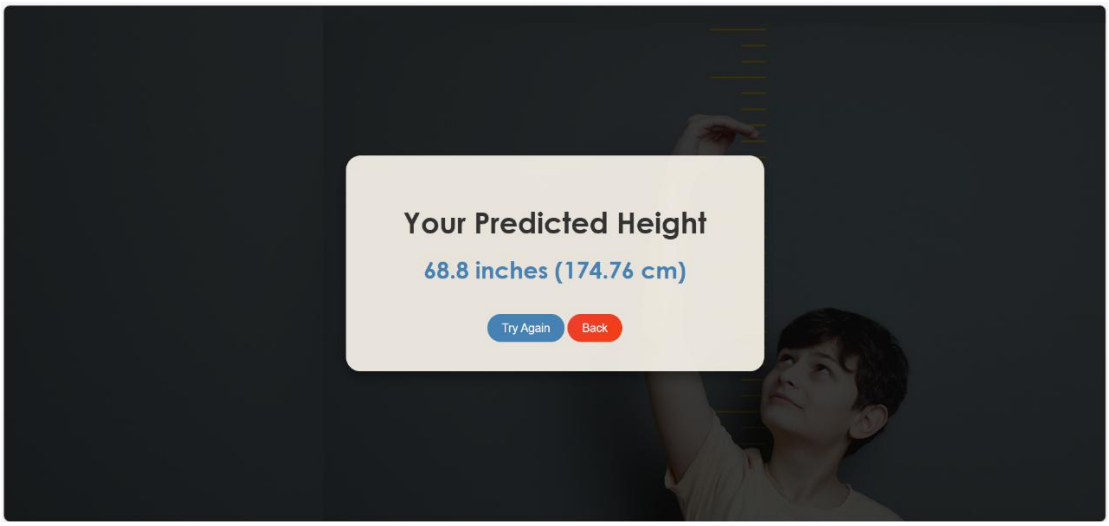


Fig. 7.1.3. Correct prediction of height testcase - 3

7.2. Integration Testing

Integration testing verifies that modules interact correctly, focusing on data flow and communication. It detects issues like mismatched function calls and data exchange errors. Approaches include incremental and big bang testing, depending on system complexity.

Case-01: Attempt login with incorrect credentials (Incorrect Test Case)

- **Objective:** Ensure that users cannot log in with invalid credentials.
- **Steps:**
    1. Enter Username = testuser, Password = wrongpass.
    2. Click the login button.
- **Expected Result:** The system should return an error message: "Invalid username or password".
- **Actual Result:** ✅ Passed

Fig. 7.2.1. Invalid Credential login testcase - 1

Case-02: Access Home page after login (Correct Test Case)

- **Objective:** Ensure that only authenticated users can view the metrics page.
- **Steps:**
    1. Log in using valid credentials.
    2. Navigate to the /home page.
- **Expected Result:** The system should display a home page.
- **Actual Result:** ✅ Passed

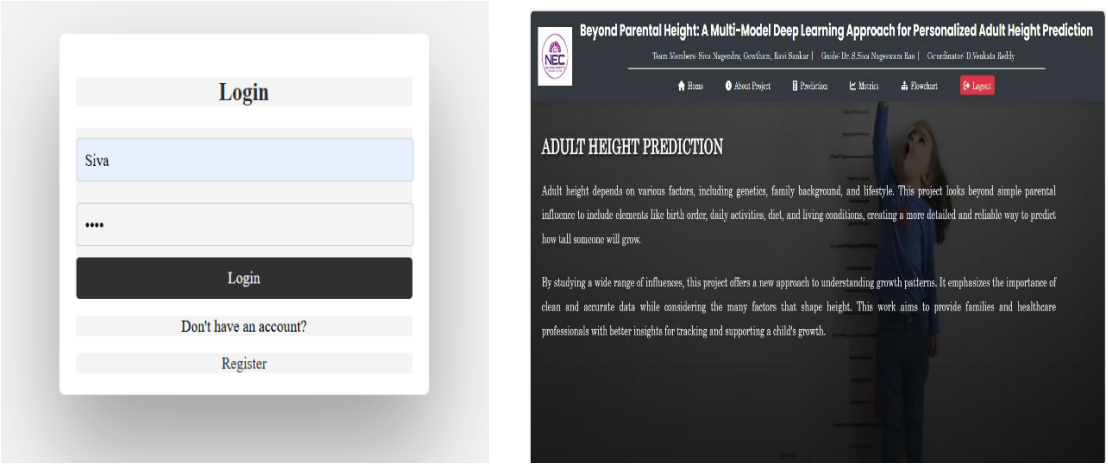


Fig. 7.2.2. Valid Credential login testcase - 2

7.3. System Testing

System testing evaluates the entire software system as a whole to ensure it meets the specified requirements. It validates not just individual functionalities but also aspects such as performance, security, usability, and compatibility. Conducted in an environment similar to the actual production setup, system testing ensures that the software behaves as expected under real-world conditions. This testing phase is crucial before deployment, as it helps identify critical issues that could impact user experience or system reliability.

Case-01: Bypass login and access restricted pages (Incorrect Test Case)

- **Objective:** Ensure unauthorized users cannot access restricted pages.
- **Steps:**
    1. Open the browser and enter the URL /predict without logging in.
- **Expected Result:** The system should redirect to the login page.
- **Actual Result:** ✅ Passed

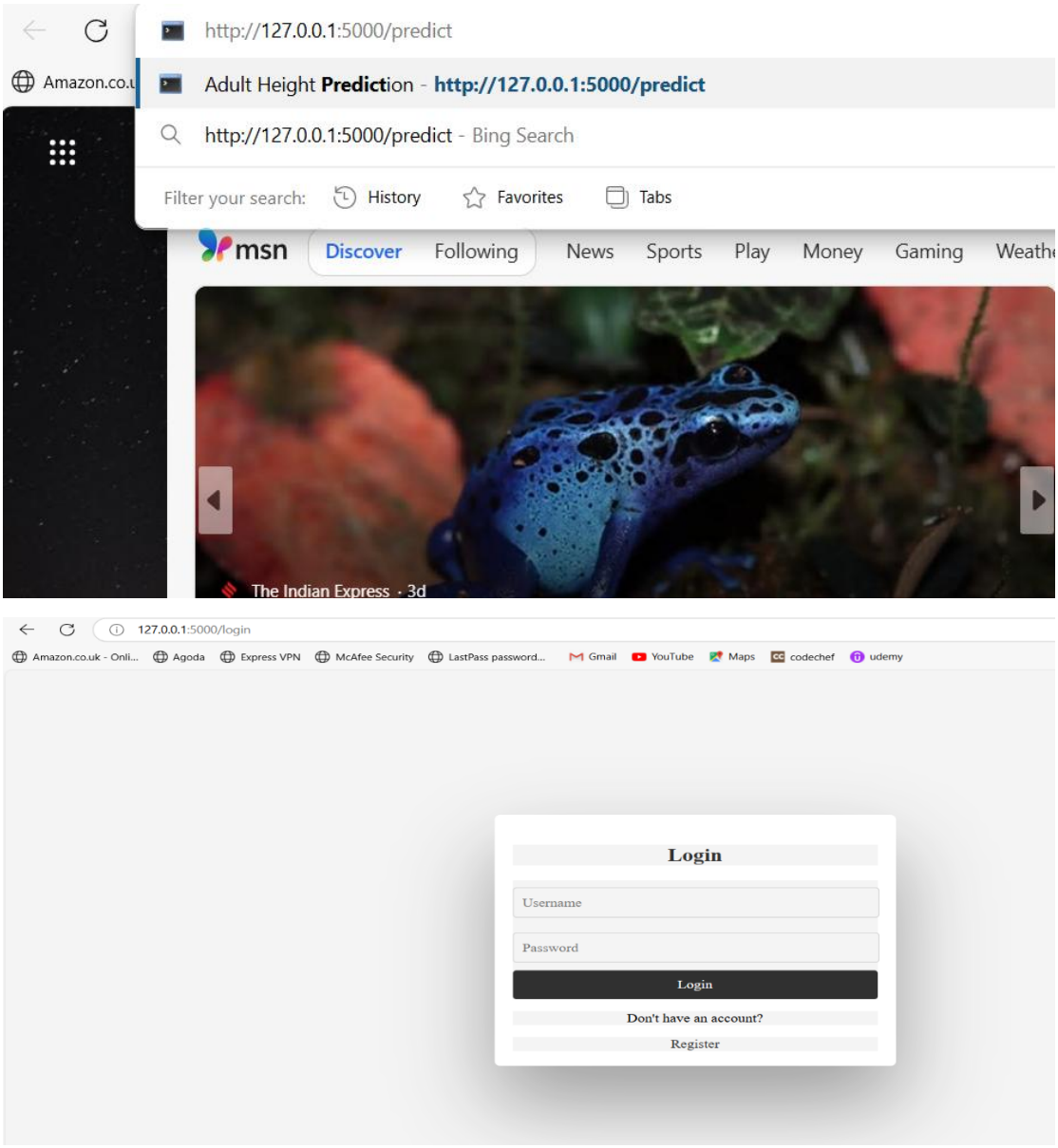
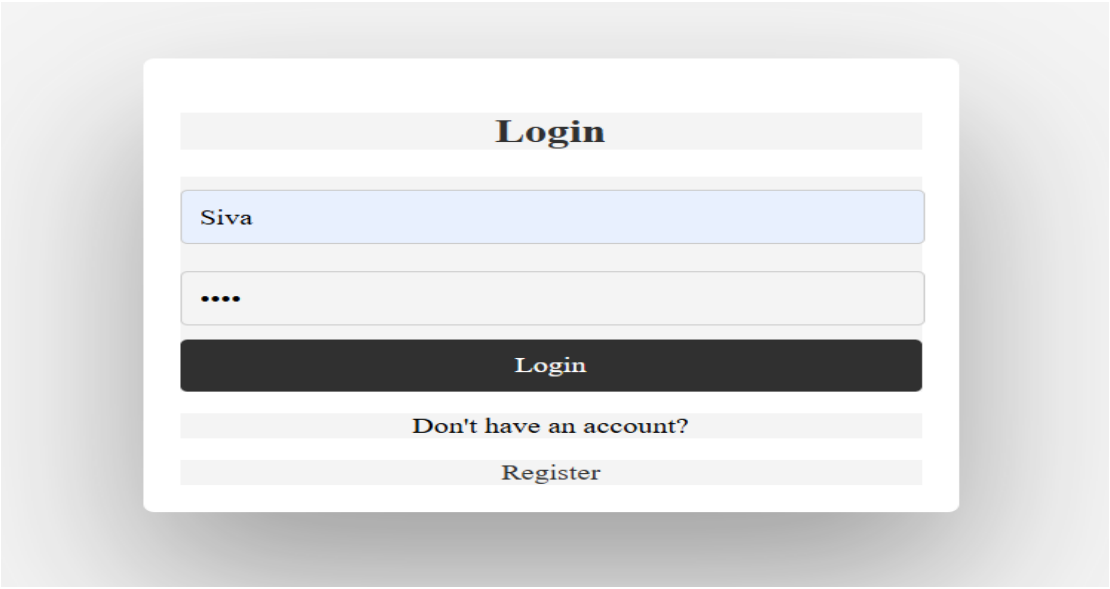


Fig. 7.3.1. Bypass login and access restricted pages testcase - 1

Case-02: Perform a complete prediction workflow (Correct Test Case)

- **Objective:** Verify that the entire workflow operates smoothly.
- **Steps:**
    1. Log in with valid credentials.
    2. Enter height prediction inputs and submit.
    3. View the predicted height.
- **Expected Result:** The system should correctly process and display the predicted height.
- **Actual Result:** ✅ Passed



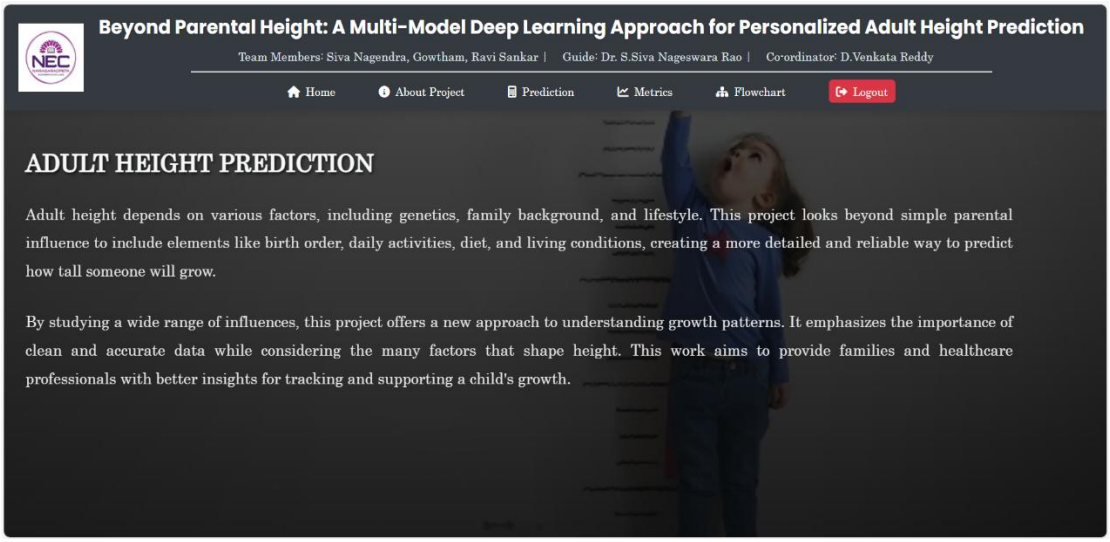


Fig. 7.3.2. Complete prediction workflow testcase - 2

Case-03: Session Timeout (Auto Logout after 5 Minutes of Inactivity)

Description: The system should automatically log out a user after 5 minutes of inactivity for security purposes.

- **Steps:**
    1. Log in with valid credentials.
    2. Stay inactive for 5 minutes without interacting with the page.
    3. Try to perform an action (e.g., navigate to another page).
- **Expected Output:** User is automatically logged out and redirected to the login page.
- **Actual Result:** ✅ Passed

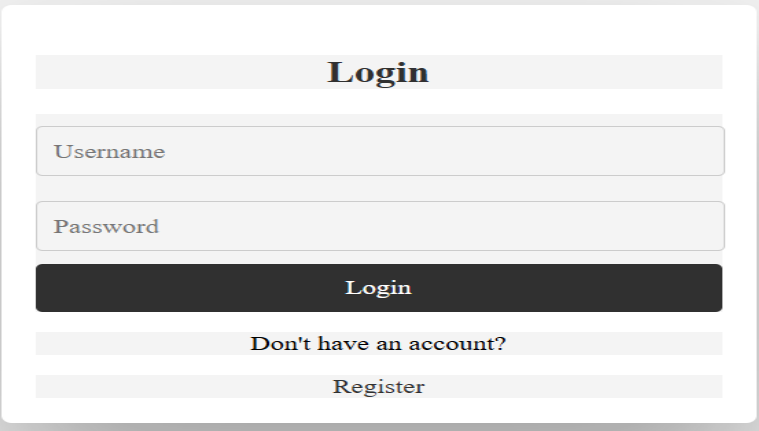


Fig. 7.3.3. Session Timeout testcase - 3

The testing phase validated the accuracy, reliability, and security of the height prediction model and web application through Unit, Integration, and System Testing. Unit tests ensured individual components functioned correctly, while Integration Testing confirmed seamless interactions between modules. System Testing assessed the overall performance, security, and usability in real-world conditions. All test cases passed successfully, verifying that the application handles errors effectively, maintains secure authentication, and ensures smooth user workflows. The results demonstrate the system's robustness, making it a reliable tool for height prediction and personalized healthcare insights before deployment in real-world scenarios.

8. RESULT ANALYSIS

The study evaluates multiple machine learning models for height prediction, focusing on the impact of outlier removal and training vs. testing accuracy. The first analysis shows that removing outliers leads to better model accuracy across all techniques, highlighting the importance of data preprocessing in machine learning. LightGBM demonstrates the highest improvement, emphasizing its effectiveness in handling irregularities in data and making it a strong candidate for robust predictions. The comparison of training and testing accuracy highlights XGBoost's tendency to overfit, as it has a significantly higher training accuracy than testing accuracy. This suggests that while XGBoost can learn patterns efficiently, it struggles to generalize to unseen data. In contrast, models like LightGBM, Linear Regression, and the MPH model show more balanced performance, indicating better generalization and reliability in making accurate predictions. The stable accuracy of these models suggests they are less prone to overfitting, making them more suitable for real-world applications.

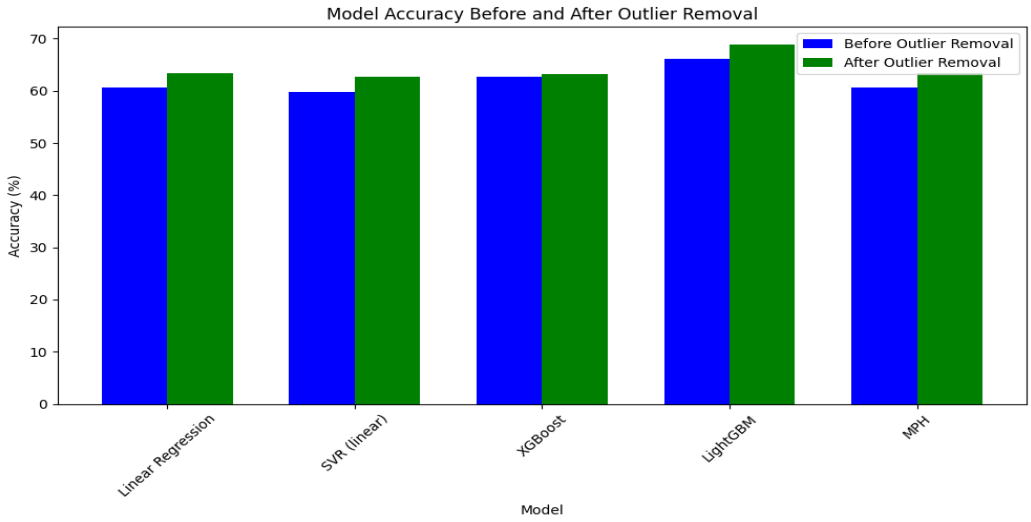


Fig.8.1. Model accuracy before and after outlier removal

Among all models, LightGBM achieves the highest accuracy (69.2%), outperforming Linear Regression, SVR Linear, XGBoost, and the MPH model. The precision, recall, and F1-score values indicate that most models perform well, but XGBoost, despite strong recall (0.878), suffers from overfitting. Linear Regression and the MPH model exhibit stable performance with an accuracy of 64.09%. SVR Linear and XGBoost perform slightly lower, making them less favorable options. The results suggest that a combination of feature engineering and regularization techniques may help reduce overfitting and enhance model performance further.

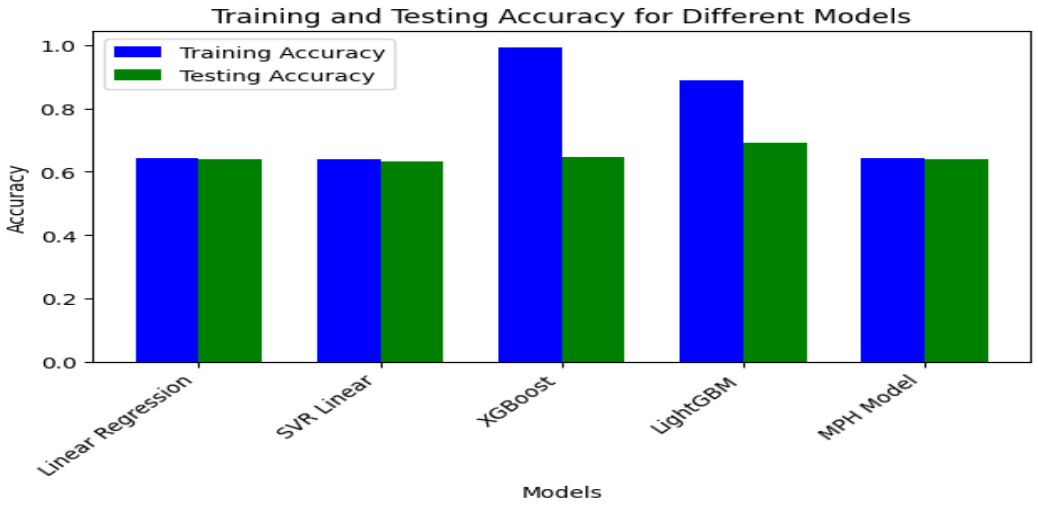


Fig.8.2. Trsining and Testing accuracy before and after outlier removal

Model	Precision	Recall	F1-score	Accuracy
Linear Regression	0.8314	0.9024	0.8654	0.6409
SVR Linear	0.8202	0.8902	0.8538	0.6363
XGBoost	0.7912	0.878	0.8323	0.6462
LightGBM	0.8352	0.8658	0.8502	0.692
MPH Model	0.8314	0.9024	0.8655	0.6409

Fig 8.3: Comparision of Precision, Recall, F1-score among models

Overall, the results highlight the importance of data preprocessing, especially outlier removal, in improving model accuracy. LightGBM stands out as the best-performing model, maintaining high accuracy while avoiding overfitting. The findings also show that traditional models like Linear Regression and MPH provide reasonable performance, but modern ensemble methods offer better predictive power. Future work can explore hyperparameter tuning and additional feature selection techniques to improve accuracy. Further experiments with larger datasets and deep learning models may provide even better insights into height prediction and related growth pattern analysis.

9. CONCLUSION

This project aims to improve personalized adult height prediction by leveraging machine learning and deep learning techniques. Traditional methods like the Mid-Parental Height (MPH) technique often fail to capture the complexity of human growth due to their reliance on limited features. By utilizing Galton's height data and incorporating additional factors such as parental height, birth order, household income, and lifestyle influences, this research enhances predictive accuracy. These features allow the models to account for environmental and genetic variations, making height estimation more precise. The integration of outlier removal further refines model performance, ensuring that predictions are robust and reliable.

The study employs advanced regression models, including Linear Regression, SVR, XGBoost, LightGBM, and the MPH model, to analyze how different algorithms handle height prediction. Among these, LightGBM emerges as the most effective, demonstrating a strong balance between accuracy and generalization. While XGBoost achieves high training accuracy, its lower testing accuracy suggests overfitting, highlighting the need for regularization techniques. By systematically comparing models before and after outlier removal, this research underscores the significance of data preprocessing in improving predictive accuracy. The results indicate that ensemble methods outperform traditional regression techniques, making them more suitable for real-world applications.

Beyond improving predictive frameworks, this project contributes to more informed decision-making in healthcare. Accurate height prediction can benefit pediatricians in diagnosing growth abnormalities at an early stage, allowing for timely interventions. Parents can use these models to estimate their children's potential height, considering both genetic predispositions and external factors. Researchers can further refine predictive models by integrating larger datasets and testing additional features. By bridging the gap between traditional statistical methods and modern machine learning approaches, this project paves the way for enhanced growth assessments.

Additionally, future work can explore integrating real-time data to build dynamic models adaptable to genetic and environmental factors. By leveraging real-world inputs such as nutrition, physical activity, and socioeconomic conditions, these models can continuously refine predictions over time. The incorporation of deep

learning architectures, such as recurrent neural networks (RNNs) or transformers, may enhance the ability to model sequential growth patterns. A hybrid approach combining machine learning with biological insights could improve precision medicine applications, offering more personalized interventions for growth monitoring.

In conclusion, this study highlights the importance of feature selection, model evaluation, and data preprocessing in height prediction. By employing machine learning and deep learning models, the research enhances traditional methods and introduces a more holistic approach to growth estimation. The findings encourage further exploration of AI-driven predictive analytics in healthcare, opening new avenues for personalized growth tracking. Future advancements in genetic analysis and AI integration can make height prediction even more accurate, ultimately benefiting medical professionals, researchers, and families worldwide.

10. FUTURE SCOPE

The future scope of this project is extensive, offering numerous possibilities for enhancement and broader applicability. Expanding the dataset beyond Galton's height data to include diverse populations from various ethnic, geographic, and socioeconomic backgrounds would improve model generalization. By incorporating global datasets, the predictive framework can cater to a wider demographic, making it more applicable for real-world use. Additionally, integrating multi-generational height records and environmental influences can refine the accuracy of height prediction, ensuring that genetic and external factors are comprehensively considered in the modeling process.

Advanced deep learning techniques, such as convolutional neural networks (CNNs) and transformer-based architectures, could significantly improve model precision and computational efficiency. CNNs can analyze patterns in growth trends, while transformers can model complex relationships between multiple influencing factors. By leveraging these methods, the project can transition from traditional regression approaches to more sophisticated predictive models capable of capturing nonlinear dependencies and long-term growth trajectories. Implementing ensemble learning strategies combining multiple models can further optimize predictions and mitigate individual model biases.

Incorporating additional features like genetic markers, real-time nutrition tracking, hormonal changes, and lifestyle habits would create a holistic growth prediction system. By integrating biometric and clinical data, the model could provide a more personalized assessment of an individual's growth potential. A hybrid approach that merges machine learning with genetic analysis could revolutionize pediatric growth monitoring, offering precise and adaptive predictions tailored to an individual's unique physiological traits. The inclusion of real-time health tracking through wearable technology could further enhance predictive capabilities, allowing for dynamic updates in growth forecasts.

Beyond height prediction, this research holds immense potential for applications in personalized healthcare. Pediatricians can use it to monitor developmental milestones, detect early signs of growth disorders, and recommend personalized interventions. In sports medicine, accurate height predictions could assist in talent identification and performance optimization by considering growth potential

in athletic training. Additionally, integrating these predictive models into public health research can aid in population studies, evaluating trends in growth patterns influenced by nutrition, socioeconomic conditions, and genetic predispositions.

In the long term, this project lays the foundation for data-driven healthcare solutions aimed at improving the quality of life. By combining machine learning with medical research, it fosters innovation in predictive analytics, helping individuals, families, and healthcare professionals make informed decisions regarding growth and development. As advancements in AI, genetics, and biomedical data integration continue, the scope for further refinement and real-world implementation of this project remains vast, paving the way for precision medicine and personalized healthcare interventions.

11. REFERENCES

[1] M. Shmoish, A. German, N. Devir, A. Hecht, G. Butler, A. Niklasson, K. Albertsson-Wikland, and Z. Hochberg, "Prediction of Adult Height by Machine Learning Technique," Clinical Endocrinology & Metabolism, vol. 106, no. 7, pp. 301–310, Jan. 2021.

[2] M. Maes, M. Vandeweghe, M. Du Caju, Ch. Ernould, J.-P. Bourguignon, and G. Massa, "A Valuable Improvement of Adult Height Prediction Methods in Short Normal Children," Hormone Research, vol. 48, no. 8, pp. 555–561, Jan. 1997.

[3] Rodari, G., Profka, E., Giacchetti, F., Cavenaghi, I., Arosio, M., & Giavoli, C. (2021). Influence of biochemical diagnosis of growth hormone deficiency on replacement therapy response and retesting results at adult height. *Scientific Reports*, *11*(1), 14553.

[4] Tanner JM, Landt KW, Cameron N, Carter BS, Patel J. Prediction of adult height from height and bone age in childhood. A new system of equations (TW Mark II) based on a sample including very tall and very short children. Arch Dis Child. 1983 Oct;58(10):767-76. doi: 10.1136/adc.58.10.767. PMID: 6639123; PMCID: PMC1628263.

[5] J. Suh, J. Heo, S. J. Kim, S. Park, M. K. Jung, H. S. Choi, Y. Choi, J. S. Oh, H. I. Lee, M. Lee, K. Song, A. Kwon, H. W. Chae, and H.-S. Kim, "Bone Age Estimation and Prediction of Final Adult Height Using Deep Learning," Yonsei Medical, vol. 64, no. 11, pp. 1150–1159, Nov. 2023.

[6] Mlakar M, Gradišek A, Luštrek M, Jurak G, Sorić M, Leskošek B, Starc G. Adult height prediction using the growth curve comparison method. PLoS One. 2023 Feb 16;18(2):e0281960. doi: 10.1371/journal.pone.0281960. PMID: 36795791; PMCID: PMC9934345.

[7] K. Mao, L. Chen, X. Fan, J. Mao, X. Zhou, and K. Fang, "Lsalo-Bp: A Hybrid Model for Children Adulthood Height Prediction," Zhejiang University of Technology, Jan. 2022.

[8] A. Holmgren, A. Niklasson, A. F. M. Nierop, G. Butler, and K. Albertsson-Wikland, "Growth Pattern Evaluation of the Edinburgh and Gothenburg Cohorts by QEPS Height Model," Pediatric Res., vol. 92, no. 2, pp. 592–601, Aug. 2022.

[9] A. Bemporad, "A Piecewise Linear Regression and Classification Algorithm with Application to Learning and Model Predictive Control of Hybrid Systems," IEEE Trans. Autom. Control, vol. 68, no. 6, pp. 3194–3209, Jun. 2023.

[10] H. B. Khazri, S. C. Shimmi, and M. T. H. Parash, "A Multivariate Analysis to Propose Linear Models for the Stature Estimation in the Sabahan Young Adult Population," PLoS ONE, vol. 17, no. 8, Art. no. e0273840, Aug. 2022.

[11] K. Umapavankumar, S. V. N. Srinivasu, S. Nageswara Rao, and S. N. Thirumala Rao, "Machine learning usage in Facebook, Twitter and Google along with the other tools," in Emerging Research in Data Engineering Systems and Computer

Communications: Proceedings of CCODE 2019, pp. 465–471. Singapore: Springer Singapore, 2020.

[12] Michael Shmoish, Alina German, Nurit Devir, Anna Hecht, Gary Butler, Aimon Niklasson, Kerstin Albertsson-Wikland, Ze'ev Hochberg, Prediction of Adult Height by Machine Learning Technique, *The Journal of Clinical Endocrinology & Metabolism*, Volume 106, Issue 7, July 2021, Pages e2700–e2710, https://doi.org/10.1210/clinem/dgab093

[13] Monasterio, X., Gil, S. M., Bidaurrazaga-Letona, I., Lekue, J. A., Santisteban, J., Diaz-Beitia, G., ... & Larruskain, J. (2021). Injuries according to the percentage of adult height in an elite soccer academy. *Journal of science and medicine in sport*, *24*(3), 218-223.

[14] J.-H. Park, M. Lee, D. Kim, H.-W. Kwon, Y.-J. Choi, K.-R. Park, S. Park, S.-B. Park, and J. Cho, "Estimating Adult Stature Using Metatarsal Length in the Korean Population: A Cadaveric Study," Int. J. Environ. Res. Public Health, vol. 19, no. 22, p. 15124, Nov. 2022.