UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

INDIVIDUAL THESIS

May 2024

Individual Thesis

MACHINE LEARNING IN ATHLETICS FOR ANALYZING AND OPTIMIZING RUNNING TECHNIQUES TO ACHIEVE PEAK PERFORMANCE

George Choratta

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

May 2024

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

MACHINE LEARNING IN ATHLETICS FOR ANALYZING AND OPTIMIZING RUNNING TECHNIQUES TO ACHIEVE PEAK PERFORMANCE

George Choratta

Supervisor
Dr George Pallis

The Individual Thesis was submitted for partial fulfilment of the requirements for obtaining a degree in Computer Science from the Department of Computer Science at the University of Cyprus.

May 2024

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Dr George Pallis, for giving me the opportunity to work with him. His invaluable guidance, support, and encouragement have been essential throughout this thesis. I am particularly thankful for the interesting subject matter, which allowed me to merge my passion for track and field with my academic pursuits.
I equally thank George Ioannou for his pivotal guidance and support throughout the process of this thesis. His expertise and constructive feedback have been instrumental in shaping the direction and quality of my study.
I would also like to extend my heartfelt thanks to Maarten Gijssel, CEO of Kinetic Analysis [6], for providing and entrusting me with this novel dataset, which served as the foundation for my thesis. His support has been instrumental in making this study possible.
Additionally, I am grateful to all the teachers and faculty members at the University of Cyprus who have imparted their knowledge and wisdom throughout my academic journey. Their dedication and passion for teaching have provided me with a strong foundation and the skills necessary to complete this thesis.
Lastly, I would like to acknowledge my family and friends for their unwavering support and understanding throughout my academic journey. Their belief in me has been a source of immense motivation and strength.

Abstract

The application of artificial intelligence in sports has surged, driven by significant advancements in data analytics and machine learning. In track and field, where marginal gains can be crucial, coaches and athletes increasingly turn to sophisticated technologies, such as motion capture, to enhance competitive performance. This thesis leverages advanced machine learning techniques to enhance the analysis and prediction of athletic performance using motion capture data.

The primary objectives are to develop predictive models that accurately estimate competition times, analyse running techniques to provide actionable feedback, explore injury prevention strategies through motion data analysis, compare various modelling techniques, and facilitate data-driven decision-making in athletic training. The study emphasizes feature extraction from motion capture data, focusing on key biomechanical insights. Sequential Feature Selection (SFS) is employed to identify the most relevant features, followed by the application of various machine learning models, including Ordinary Least Squares (OLS) regression, ElasticNet, Bayesian Ridge, Lasso, HuberRegressor, and LinearRegression. Hyperparameter tuning is performed to optimize these models for better predictive accuracy.

A unique aspect of this thesis is the utilization of a novel dataset derived from Xsens Custom, provided by Kinetic Analysis [6], which has never been used in prior studies. This thesis represents the first attempt to leverage this specific data for predictive modelling in sports analytics. Additionally, there is no existing work that combines such comprehensive biomechanical feature extraction from Xsens Custom MVNX data with advanced machine learning techniques.

The thesis highlights the efficacy of combining manually and automatically extracted features in predicting performance outcomes. The models developed offer practical insights for coaches and athletes, guiding them to optimize training regimens and improve performance. Furthermore, the study demonstrates the potential of these predictive models in identifying injury risks, thus contributing to safer training practices.

This thesis provides a comprehensive approach to integrating advanced machine learning techniques with sports science, offering a robust framework for enhancing athletic performance and injury prevention. Through detailed data analysis and model comparison, it lays the groundwork for future research and application in the field of sports analytics.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

In recent years, the integration of technology in sports training has seen significant advancements, particularly with the emergence and evolution of Artificial Intelligence (AI) and Machine Learning (ML). These technologies are increasingly being applied across various fields, with track and field emerging as a prime candidate for their application. The ability to analyse vast amounts of data and derive actionable insights is revolutionizing how athletes train and compete.

As a track and field athlete and an undergraduate student specializing in computer science with a focus on AI, I have observed first-hand the potential benefits that AI and ML can bring to athletics. The precise analysis enabled by these technologies can significantly enhance an athlete's understanding of their performance, leading to more tailored and effective training strategies. Moreover, the use of motion capture technology, combined with sophisticated machine learning models, offers a novel approach to analysing and optimizing running techniques, ultimately aiming to boost athletic performance.

This thesis explores the intersection of AI, machine learning, and sports science to develop robust predictive models for athlete performance, specifically focusing on completion times in track and field events. By leveraging detailed motion capture data, this study aims to not only predict performance outcomes with high accuracy but also to uncover underlying patterns and correlations that could inform more efficient training methodologies.

Additionally, this study lays the groundwork for future explorations into the identification of potential biomechanical problems in running techniques. The long-term goal is to evolve these models into tools that can pre-emptively identify risk factors for injuries, thereby helping athletes avoid long-term harm and improve their overall athletic longevity. This dual focus on performance enhancement and injury prevention encapsulates the broader potential of AI and ML applications in sports, marking a significant step forward in how athletes train, compete, and manage their health.

1.2 Problem Statement

The integration of artificial intelligence (AI) in sports, particularly track and field, has grown significantly due to advancements in data analytics and machine learning. Track and field is a sport where even the smallest improvements can lead to substantial competitive advantages. Coaches and athletes increasingly rely on sophisticated technologies, such as motion capture, which was initially developed for animation and gaming but has become a critical tool for biomechanical analysis in sports. Motion capture technology offers detailed insights into athletes' movements with a level of precision that surpasses human observation.

The primary problem is that the human eye, even with expert visual observation, cannot capture all the intricate variables and detailed analytics needed to fully understand and optimize an athlete's technique and form. Traditional analytical methods, including basic statistical analysis and simple biomechanical models, are often inadequate for processing and interpreting the complex and voluminous data generated by motion capture technologies. This inadequacy leads to suboptimal performance prediction and technique optimization.

This thesis aims to address this gap by developing advanced machine learning models capable of effectively assimilating and analysing large datasets from motion capture technologies. These models will provide accurate performance predictions and actionable feedback, enhancing athletic performance and aiding in injury prevention. By leveraging the detailed insights from extracted features, the thesis will offer a new paradigm for training and performance analysis in track and field, ultimately contributing to significant improvements in the application of AI in sports. The successful implementation of these models promises to revolutionize the way athletes train and perform, providing coaches and athletes with the tools to achieve their full potential.

1.3 Study Objectives

The primary objective of this thesis is to enhance the analysis and prediction of athletic performance in track and field using motion capture data by leveraging advanced machine learning techniques. These techniques include Ordinary Least Squares (OLS) regression, ElasticNet, Bayesian Ridge, Lasso, HuberRegressor, LinearRegression, and hyperparameter tuning. Additionally, the study employs Sequential Feature Selection (SFS) and correlation analysis methods like Spearman to identify the most relevant features. The specific objectives are:

1. **Develop Predictive Models**: Construct and evaluate predictive models that accurately estimate competition times for track and field athletes based on motion capture data. These models will utilize both manually extracted features and features automatically derived through tsfresh to determine which method provides greater predictive accuracy. Techniques such as OLS regression, ElasticNet, Bayesian Ridge, Lasso, HuberRegressor, LinearRegression, and hyperparameter tuning will be employed.
2. **Analyse Running Techniques**: Utilize insights gained from predictive modelling to analyse and optimize running techniques. This involves identifying key biomechanical factors that influence performance and providing actionable feedback that athletes and coaches can use to improve training outcomes. Advanced machine learning techniques, including feature importance analysis (such as correlation analysis using Spearman's method) and regression analysis to interpret feature coefficients, will be employed.
3. **Enhance Injury Prevention**: Explore the potential of predictive models to identify patterns and anomalies in motion data that may predispose athletes to injuries. By understanding these patterns, the study aims to contribute to the development of strategies that could help prevent injuries before they occur.
4. **Compare Modelling Techniques**: Compare the effectiveness of different modelling approaches, specifically OLS regression and the best-performing models identified through PyCaret, in handling the complexities of sports performance data. This comparison will help establish best practices for deploying machine learning models in sports analytics.
5. **Facilitate Data-Driven Decisions**: Demonstrate how machine learning can transform traditional coaching methods by integrating data-driven insights into training regimens. This includes evaluating the practical implications of model findings in real-world athletic training and competition settings.

1.4 Overview of the Thesis Structure

This thesis is organized into six chapters, each designed to build upon the information and analysis presented in the previous chapters:

Chapter 1: Introduction

This chapter sets the stage for the thesis by outlining the motivation behind the study, defining the problem statement, and specifying the objectives of the study.

Chapter 2: Background

This chapter introduces foundational concepts and technologies crucial to this thesis, focusing on essential biomechanical metrics. It discusses the Symmetry Index (SI) for evaluating movement efficiency and the significant role of motion capture data in sports analytics. The chapter also covers machine learning tools like PyCaret and tsfresh, explaining their use in feature extraction and predictive modelling. Each section aims to lay a solid foundation for the methodologies and analyses explored in subsequent chapters.

Chapter 3: Related Work

This chapter reviews existing literature and previous research studies that focus on feature extraction, modelling techniques, and the use of AI in sports analytics. It includes an examination of a kinematic analysis study on field hockey players using MVNX files from Xsens, similar to the dataset used in this thesis. Additionally, the chapter reviews the paper [2] highlighting gaps in current methodologies that this study aims to address.

Chapter 4: Methodology

This chapter provides a comprehensive examination of the methodologies and results of the study. It begins with an in-depth look at the data pre-processing techniques and visualizations used to prepare the dataset for analysis. The chapter then explores the feature extraction processes, detailing both manual and automated methods via tools like PyCaret and tsfresh.

Chapter 5: Training and Prediction

This chapter discusses the modelling strategies implemented to predict completion times, presenting the results from different modelling approaches, including OLS and multiple model comparisons within PyCaret. The chapter includes hyperparameter tuning to optimize model performance. It evaluates the models' performance, interpreting the significance of their outputs and discussing the practical implications of the findings. Additionally, it addresses the challenges and limitations encountered during the study, providing a critical perspective on the methods and results presented.

Chapter 6: Conclusion and Future Work

The final chapter concludes the thesis by summarizing key findings and contributions of the study. It begins by restating the study's objectives, followed by a summary of key findings. The chapter also discusses the challenges and limitations encountered during the research. Finally, it outlines potential future work directions that could further enhance the integration of machine learning in sports analytics, specifically in improving performance prediction and injury prevention.

Chapter 2

Background

This section provides a comprehensive understanding of the foundational technologies and concepts that are integral to this thesis. It outlines key biomechanical metrics such as stride length, airtime, and ground contact time, which are integral for analysing track athletes' performance. Additionally, it introduces the Symmetry Index (SI) equation, crucial for evaluating athletes' movement efficiency and balance. The chapter also covers the pivotal role of motion capture data in sports analytics, detailing how this data is captured and its importance in high-precision analysis. Furthermore, descriptions of the machine learning tools PyCaret and tsfresh are provided, explaining their utility in feature extraction and predictive modelling. Every section is carefully structured to elucidate these fundamental concepts, laying the groundwork for the detailed analyses and methodologies to be examined in the following chapters.

2.1 Stride length

Stride length is a critical biomechanical parameter in the analysis of running performance. It is defined as the distance covered between two successive placements of the same foot, encompassing one complete cycle of a leg's movement during running as shown in ***Figure 1 and Figure 2***. Stride length varies among athletes due to factors such as body size, leg length, and the intensity of the exercise.

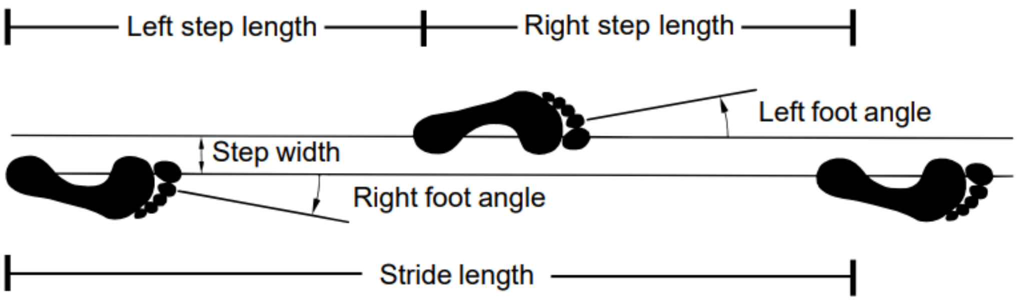


Figure 1 Stride Length [1]

In sports biomechanics, stride length is considered a key determinant of running speed and efficiency. According to research by Novacheck [11], stride length, in conjunction with stride frequency, dictates the pace at which a runner moves. The balance between stride length and stride frequency must be optimized for peak athletic performance, with variations observed between sprinting and long-distance running [15]. For instance, sprinters tend to have longer stride lengths to maximize speed over short distances, while distance runners might optimize stride length for endurance and energy conservation.

Moreover, optimizing stride length is not only crucial for enhancing performance but also for reducing injury risk. Improper stride length can lead to biomechanical inefficiencies and increased strain on specific muscles and joints, potentially resulting in overuse injuries [3].

2.2 Ground Contact Time

Ground Contact Time (GCT) refers to the duration of time a sprinter's foot remains in contact with the ground during each stride as shown in ***Figure 2***. This metric is a fundamental aspect of running biomechanics, reflecting the efficiency with which a sprinter interacts with the running surface. In sprinting, a shorter GCT is often associated with greater running speed and improved efficiency because it indicates rapid force application and quick foot turnover.

The amount of time sprinters spend in contact with the ground directly affects their ability to generate speed. Efficient sprinters typically exhibit a shorter GCT, allowing them to apply more power in less time, thereby increasing their stride frequency and overall speed. Training to reduce GCT can lead to significant improvements in sprint performance by enhancing the athlete's explosive power and reaction time off the ground [4][7][8][11][12][16].

2.3 Airtime/Swing Phase Time

Airtime, in the context of running, refers to the interval during which neither foot is in contact with the ground between consecutive steps from one foot to the other as shown in ***Figure 2***. This occurs from the toe-off of one foot until the moment the opposite foot touches the ground.

This phase is crucial as it reflects the athlete's ability to generate sufficient propulsion and maintain momentum.

Airtime is a critical metric for evaluating sprint performance because it directly relates to the athlete's speed and efficiency. In sprinting, as speed increases, the time spent in swing phase typically increases [7][11], contributing to a greater stride length and enhanced propulsion. Therefore, the focus is not merely on minimizing airtime but on optimizing it along with stride frequency and ground contact mechanics to achieve the best balance for maximal speed.

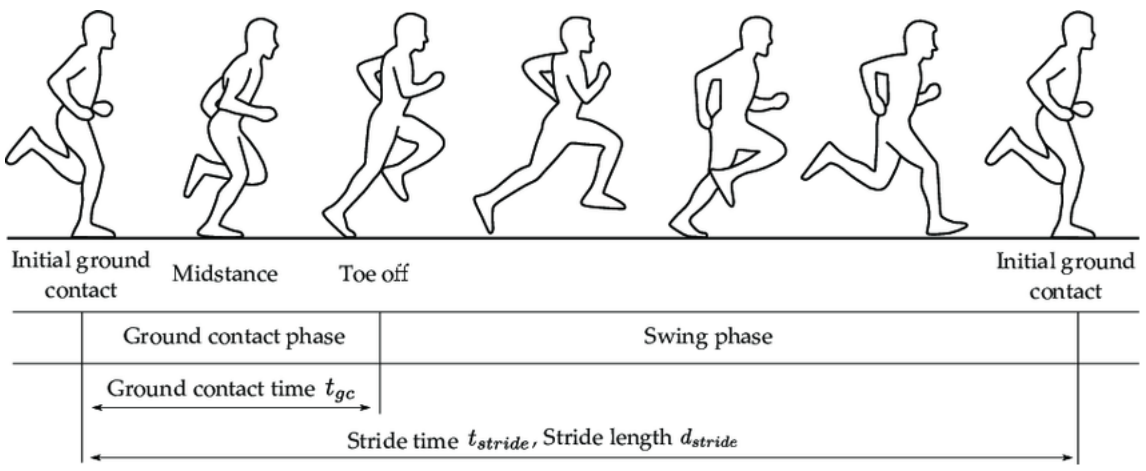


Figure 2 Gait Cycle when running

2.4 Symmetry Index (SI) equation

The Symmetry Index (SI) is a quantitative measure used to assess the symmetry of movement between the left and right limbs of an athlete. This metric is particularly relevant in sports science, where asymmetries can indicate potential inefficiencies or predispositions to injury. Symmetry is often idealized in athletic performance, suggesting that a more balanced movement could lead to improved performance and reduced injury risk [10][17].

The Symmetry Index (SI) is calculated using the equation provided below. This index is a ratio expressed as a percentage, which quantifies the asymmetry between the right and left limbs' movements during performance.

$$\mathbf{SI} = \frac{|\,\boldsymbol{xr} - \boldsymbol{xl}\,|}{\frac{\boldsymbol{xr} + \boldsymbol{xl}}{\mathbf{2}}} * \mathbf{100}$$

In the formula ***xr*** and ***xl*** symbolize the measurements from the right and left limbs, respectively. These measurements could be related to various biomechanical outputs, such as force exerted, power generated, or stride lengths during sprinting. The absolute value of their difference, **|** ***xr* – *xl*** **|**, is divided by the average of the two measurements to determine the relative asymmetry.

Multiplying this ratio by 100% converts it into a percentage. The resulting SI value indicates the level of symmetry, with values approaching zero indicating a high degree of symmetry and higher values indicating greater asymmetry [10][17].

2.5 Max Effort / Sub Max Effort

Max Effort (ME) refers to the highest level of intensity an athlete can exert in a given activity. In sprinting, this would be the athlete's absolute fastest and most powerful performance over a short duration, typically seen in races or short bursts of speed. Max Effort is characterized by a full mobilization of an athlete's physical and mental resources, pushing their physiological limits.

Sub Max Effort (SME), on the other hand, is a level of exertion that is sustainable over a more extended period and is less than the athlete's maximum capacity. While still demanding, it is not as intense as Max Effort and is often used in training to improve endurance, technique, and to simulate race conditions without the strain of a full-out effort.

2.6 Motion Capture Data

Motion Capture Data is a sophisticated method used to record the movements of objects or people. In sports science, it involves tracking the movement of athletes to gather detailed data on their biomechanics. This technology captures the dynamics of movement in several dimensions, allowing for an in-depth analysis of the performance that is not visible to the naked eye. Motion capture technology, particularly from specialized systems like Xsens Custom, enables the collection of precise and multi-dimensional movement data from athletes. These advanced systems use inertial sensors to track and record an athlete's movements without the need for optical cameras or external reference points.

The **Xsens Custom system** is a highly sophisticated motion capture solution tailored to capture the nuanced dynamics of an athlete's movement. By attaching sensors to the athlete’s body, the system provides real-time three-dimensional data on the athlete's kinematics, offering invaluable insights into performance metrics such as joint angles, acceleration, and velocity.

The data used in this study was collected by **KineticAnalysis [6]**, a company that specializes in the analysis of movement for sports performance. Their expertise in processing and interpreting the raw data captured by the Xsens Custom system allows for a detailed understanding of an athlete’s biomechanics and the identification of areas for improvement. A unique aspect of this study is the utilization of a novel dataset derived from Xsens Custom, which has never been used in prior studies. This unprecedented dataset provides a fresh perspective and significant potential for advancing the field of sports performance analytics.

2.7 Pycaret

PyCaret is an open-source, low-code machine learning library in Python that aims to reduce the complexity of performing end-to-end machine learning tasks. It's designed to help data scientists and developers expedite the process of building and deploying machine learning models with a minimal amount of coding effort. [13]

PyCaret is known for its comprehensive suite of features that facilitate everything from data pre-processing, model training, and model tuning to model analysis and final deployment. The library supports various modules for classification, regression, clustering, anomaly detection, natural language processing, and associative rule mining, making it a versatile tool for a wide range of applications.

In this thesis, PyCaret was utilized primarily for training machine learning models and identifying the most effective model for predicting athletic performance from motion capture data. The library's setup allows for quick iteration over multiple model types, facilitating the evaluation of each model based on predefined metrics such as accuracy, recall, precision, and F1 score.

2.8 tsfresh

tsfresh is an open-source Python library that facilitates the extraction of relevant features from time series data automatically. It is specifically designed to handle various types of time series inputs and derive meaningful statistical, and mathematical features that significantly enhance data analysis and predictive modelling. [18]

tsfresh excels in its ability to efficiently extract a large number of features from time series data, which includes basic statistics like mean and median, trends, variability, and more complex characteristics such as Fourier transforms and autocorrelation. One of its key strengths is the automatic selection of relevant features that best contribute to the predictive power of a model, helping to reduce dimensionality and avoid overfitting.

In this thesis, tsfresh was employed alongside my own manual extraction efforts to pre-process motion capture data, deriving key features essential for evaluating athletes' performance. This comprehensive approach combines automated and manual methodologies to form a robust foundation for creating machine learning models. These models are designed to forecast results such as competition completion times, and future work will extend to assess injury risks. Together, these methods enhance the precision and efficiency of the initial feature engineering phase

Chapter 3

Related Work

3.1 Previous Research in Motion Capture Data and Sports Analytics

The study [5] explores the use of wearable sensors to monitor athletes' performance, specifically focusing on field hockey players. This paper, produced by researchers who sourced data from the same company as my study (Kinetic Analysis [6]), provides valuable insights into kinematic analysis using MVNX files. Additionally, the researcher Ioannou, who authored the paper, provided guidance and support throughout my thesis process, especially in pre-processing methodologies.

In their research, Ioannou et al. aimed to minimize the number of sensors required for accurate performance analysis, thereby reducing the invasiveness and potential performance hindrance associated with wearing numerous sensors. By analysing data from 77 field hockey players over four years, they identified correlations between different sensors to determine a minimal yet effective sensor setup. This approach reduced the number of sensors from 23 to 8, maintaining the ability to predict ball speed during a drag-flick with minimal loss in accuracy. The methodology employed involved parsing MVNX files to extract and pre-process data into a structured format suitable for analysis. The researchers used correlation metrics to eliminate redundant sensors, focusing on those that provided the most unique and valuable data. This process highlighted the importance of key sensors placed on specific body parts such as the toes, hands, lower legs, and forearms, as well as the T12 sensor on the spine.

Application to My Study

While my study did not focus on minimizing the number of sensors, it adopted a similar approach in several key areas to predict athletic performance. First, the pre-processing methodologies, developed with guidance from Ioannou, were essential in transforming MVNX files into a pandas DataFrame, enabling structured data analysis. Following this, feature extraction, as in Ioannou's study, was conducted using tools such as TSFresh to derive a comprehensive set of metrics from the motion capture data. Finally, I applied correlation metrics to identify and retain the most relevant features, ensuring that only those most correlated with the target variable were used for better predictive results. These steps were foundational in enabling the detailed analysis conducted in my study.

3.2 Modelling and Predicting Athletic Performance in Running

The paper [2] presents a comprehensive model for predicting individual performance in running using modern machine learning techniques and a database of over 164,000 British runners. The model leverages Local Matrix Completion (LMC) to identify three essential parameters per runner:

1. **Endurance:** This parameter acts as an exponent in an individual power law that reflects the runner's endurance. It explains most performance differences across distances greater than 800m, providing a personalized measure of an athlete's capacity to maintain performance over longer distances.
2. **Balance between Speed and Endurance:** This parameter captures how a runner balances speed versus endurance in races, affecting performance over various distances. It offers insights into each athlete's running style and pacing strategies.
3. **Middle-Distance Specialization:** This parameter accounts for a runner's non-linear corrections to the power law, emphasizing their specialization in middle-distance events. It refines the performance prediction by considering individual variations that deviate from typical pacing trends.

The authors apply these parameters to generate highly accurate predictions, achieving an average prediction error of 3.6 minutes for marathon performances and 0.3 seconds for 100m performances. This model outperforms previous state-of-the-art performance predictors by 30% in root mean square error (RMSE).

By combining these three parameters, the paper's authors produce a holistic, parsimonious model that accurately predicts performance across a wide range of race distances, explaining both physiological and behavioural aspects of running. Their approach unifies principles from power law modelling, scoring tables, and physiological parameters, providing a more comprehensive view of athletic performance.

Differences Between the Paper and This Thesis

The paper [2] and this thesis differ significantly in terms of data sources, analysis approaches, and objectives. The existing paper relies on historical race results from a large database, using predictive models trained on broad datasets to estimate completion times. This approach

focuses on general race performance data without delving into biomechanical features or patterns.
In contrast, this study emphasizes feature extraction from motion capture data to predict completion times, utilizing both manually and automatically extracted features. Manually extracted features include key biomechanical insights such as stride length, ground contact time, and symmetry index, which help identify critical areas for performance optimization. Automatically extracted features, derived using advanced techniques like tsfresh, capture subtle biomechanical patterns and statistical transformations that provide additional layers of analysis.
By combining both manually and automatically extracted features, this thesis offers a more detailed and comprehensive understanding of the factors influencing athletic performance. This dual approach not only highlights the importance of key biomechanical elements but also uncovers intricate patterns that may not be immediately apparent through manual analysis alone.
Another difference lies in the type of feedback provided. The existing paper offers race performance predictions based on aggregated historical data but lacks detailed, actionable recommendations for athletes and coaches to improve performance. This study goes beyond simple predictions by generating detailed, actionable insights that coaches and athletes can use to refine training regimens. The predictive models identify specific features and patterns that athletes need to focus on to achieve better results.
Furthermore, the existing paper primarily focuses on predicting race times and quantifying athletic performance without considering injury risks or related biomechanics. In contrast, this study extracts motion capture features that could signal injury risks. By identifying biomechanical patterns or anomalies, the study provides feedback that allows athletes to adjust training and running mechanics to minimize injury risks.

Chapter 4

Methodology

4.1 Pre-processing of Data

The pre-processing stage lays the foundation for converting unprocessed motion capture data into a well-organized format suitable for detailed analysis. This section provides a comprehensive overview of the MVNX file format, highlighting the intricate and hierarchical nature of the data captured by the motion sensors. Next, it proceeds to explain how the data is converted from an XML-based MVNX format to a pandas DataFrame. The pandas DataFrame is well-known for its analytical flexibility and capability in the Python data science community. The careful reformatting is essential as it allows for the application of advanced data manipulation and machine learning techniques in the latter phases of the thesis.

4.1.1 MVNX File Structure

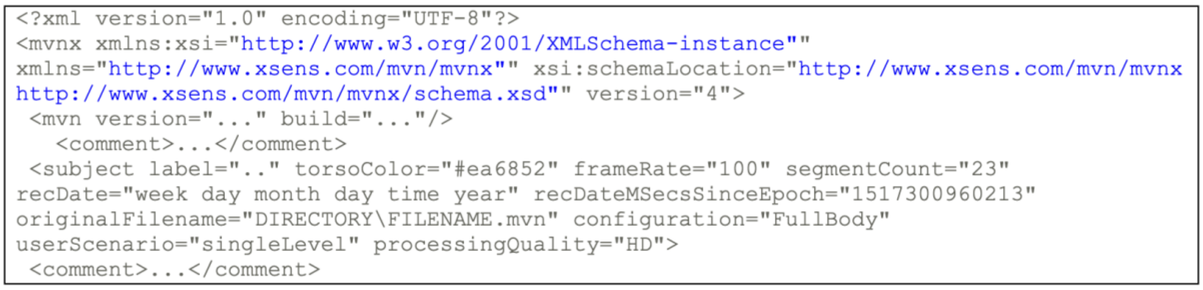
The motion capture data used in this study were stored in MVNX format [9], a sophisticated XML-based format designed to encapsulate a wide range of biomechanical motion data.
This format is notable for its precision in calculating kinematic data, such as the position and orientation of each body segment B, relative to a fixed earth reference coordinate system G.

Reference Coordinate Systems:

- **Global Coordinate System (G)**: By default, the earth-fixed reference coordinate system used in MVNX files is defined as a right-handed Cartesian coordinate system. The axes of this system are aligned as follows:
  - **X:** Points toward the local magnetic North.
  - **Y:** Follows the right-handed coordinate system, pointing West.
  - **Z:** Points upward, perpendicular to the Earth's surface.
- **Body Frame (B)**: Each body segment's frame B is initially aligned with the global reference frame G when the subject assumes a T-pose.

The file structure is comprehensive and includes several critical elements:

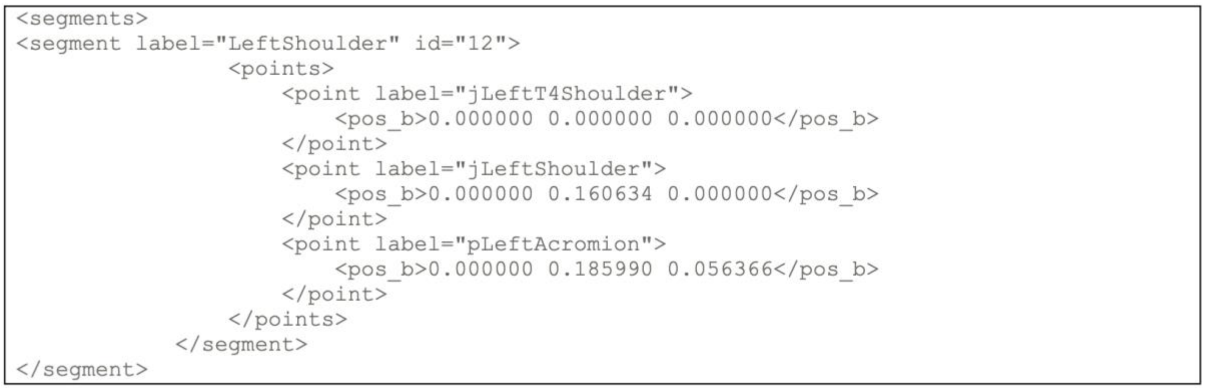
1. **Initialization**: Each MVNX file starts with the XML version declaration and includes the root element **mvnx**, which references the XML Schema Definition (XSD) and the MVNX version. It contains metadata such as the MVN Analyse/Animate version, build details, comments added to the original recording, session information (including the suit label, sample frequency, and number of body segments), the date of the recording, and the filename of the original session. The structure of the initialization element is illustrated in ***Figure 4***.



```
<?xml version="1.0" encoding="UTF-8"?>
<mvnx xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance""
xmlns="http://www.xsens.com/mvn/mvnx"" xsi:schemaLocation="http://www.xsens.com/mvn/mvnx
http://www.xsens.com/mvn/mvnx/schema.xsd"" version="4">
 <mvn version="..." build="..."/>
   <comment>...</comment>
 <subject label=".." torsoColor="#ea6852" frameRate="100" segmentCount="23"
recDate="week day month day time year" recDateMSecsSinceEpoch="1517300960213"
originalFilename="DIRECTORY\FILENAME.mvn" configuration="FullBody"
userScenario="singleLevel" processingQuality="HD">
 <comment>...</comment>
```

Figure 4 MVNX file Initialization

2. **Segments**: This section defines the positions (**pos_b**) of connecting joints (prefixed with "j") and anatomical landmarks (prefixed with "p") relative to the origin of that segment in the body frame B. The configuration of the segments section is illustrated in ***Figure 5***.



```
<segments>
<segment label="LeftShoulder" id="12">
                <points>
                    <point label="jLeftT4Shoulder">
                        <pos_b>0.000000 0.000000 0.000000</pos_b>
                    </point>
                    <point label="jLeftShoulder">
                        <pos_b>0.000000 0.160634 0.000000</pos_b>
                    </point>
                    <point label="pLeftAcromion">
                        <pos_b>0.000000 0.185990 0.056366</pos_b>
                    </point>
                </points>
            </segment>
</segment>
```

Figure 5 MVNX file section Segments

3. **Sensor Data**: Lists the names of segments measured with the motion trackers (MTs). The configuration of the sensors section is illustrated in ***Figure 6***.

```
<sensors>
        < sensor label ="Pelvis"/>
        < sensor label ="Head"/>
        < sensor label ="RightShoulder"/>
        etc.
</sensors>
```

Figure 6 MVNX file section Sensors

4. **Joints Data**: Provides a list of joints, detailing the segments and connections for each joint. The configuration of the joints section is illustrated in ***Figure 7***.

```
<joints>
        <joint label="‟ jLeftHip”>
               <connector1>Pelvis/jLeftHip </connector1>
               <connector2>LeftUpperLeg/jLeftHip </connector2>
         </joint>
         etc.
</joints>
```

Figure 7 MVNX file section Joints

5. **Ergonomic Joint Data**: Includes specific joint angles used in ergonomic analysis along with their segment connections. The configuration of the ergonomic joint section is illustrated in ***Figure 8***.

```
<ergonomicJointAngles>
          <ergonomicJointAngle label="‟ T8_Head” index="0" parentSegment="T8"
childSegment="Head"/>
          etc.
</ergonomicJointAngles>
```

Figure 8 MVNX file section Ergonomic Joint

6. **Foot Contact Detection**: Details the foot contact points, marking each with a value of '1' if in contact with the ground and '0' if not, indexed in the foot contact definition array for every given timeframe. The configuration of the foot contact definition section is illustrated in ***Figure 9***.

```
<footContactDefinition>
                      <contactDefinition label="LeftFoot_Heel" index="0"/>
                      <contactDefinition label="LeftFoot_Toe" index="1"/>
                      <contactDefinition label="RightFoot_Heel" index="2"/>
                      <contactDefinition label="RightFoot_Toe" index="3"/>
 </footContactDefinition>
```

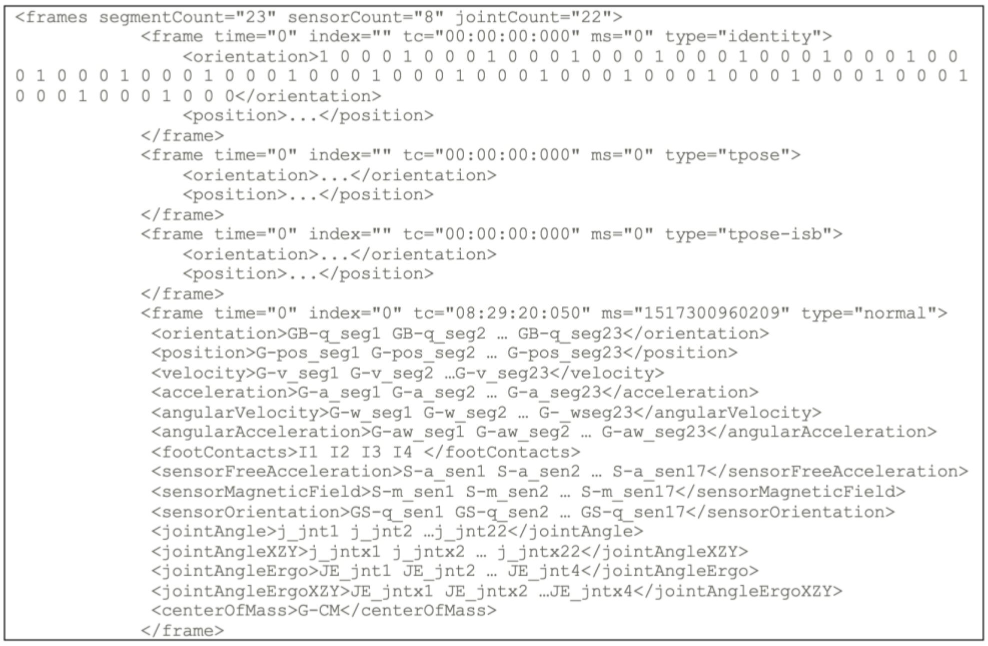
Figure 9 MVNX file section Foot Contact Definition

7. **Frames**:

The frames section of the MVNX file is structured to contain detailed biomechanical data across different types of frames, starting with the general frame attributes and moving to specific frame types:

A. **General Frame Attributes**: Each set of frames begins with an opening tag that includes the total counts for segments, sensors, and joints. This tag sets the context for the detailed data contained in each specific frame type that follows.
B. **Identity Frame**: The first type, known as the "identity frame", is marked by the type "identity". This frame denotes the null pose or identity pose. All segment orientations in this pose are aligned with the global coordinates and have unit quaternion.
C. **Tpose Frame**: Following the identity frame is the "tpose" frame, which describes the positions and orientations of all segments arranged in a T-pose. The positions and orientations of certain segments deviates slightly from the identity pose.
D. **Tpose-ISB Frame**: The frame type “tpose-isb” describes the positions and orientations of all segments in the T-pose, but using the MVN anatomical frame for the body segments. The MVN anatomical frame is used to calculate the joint angles.
E. **Normal Frames**: After the specific pose frames, the section transitions to 'normal' frames that capture actual motion data during a session. These frames are timestamped with their recording time, index, and additional details such as velocity, acceleration, angular velocity, angular acceleration, foot contacts, sensor data, joint angles and center of mass data.

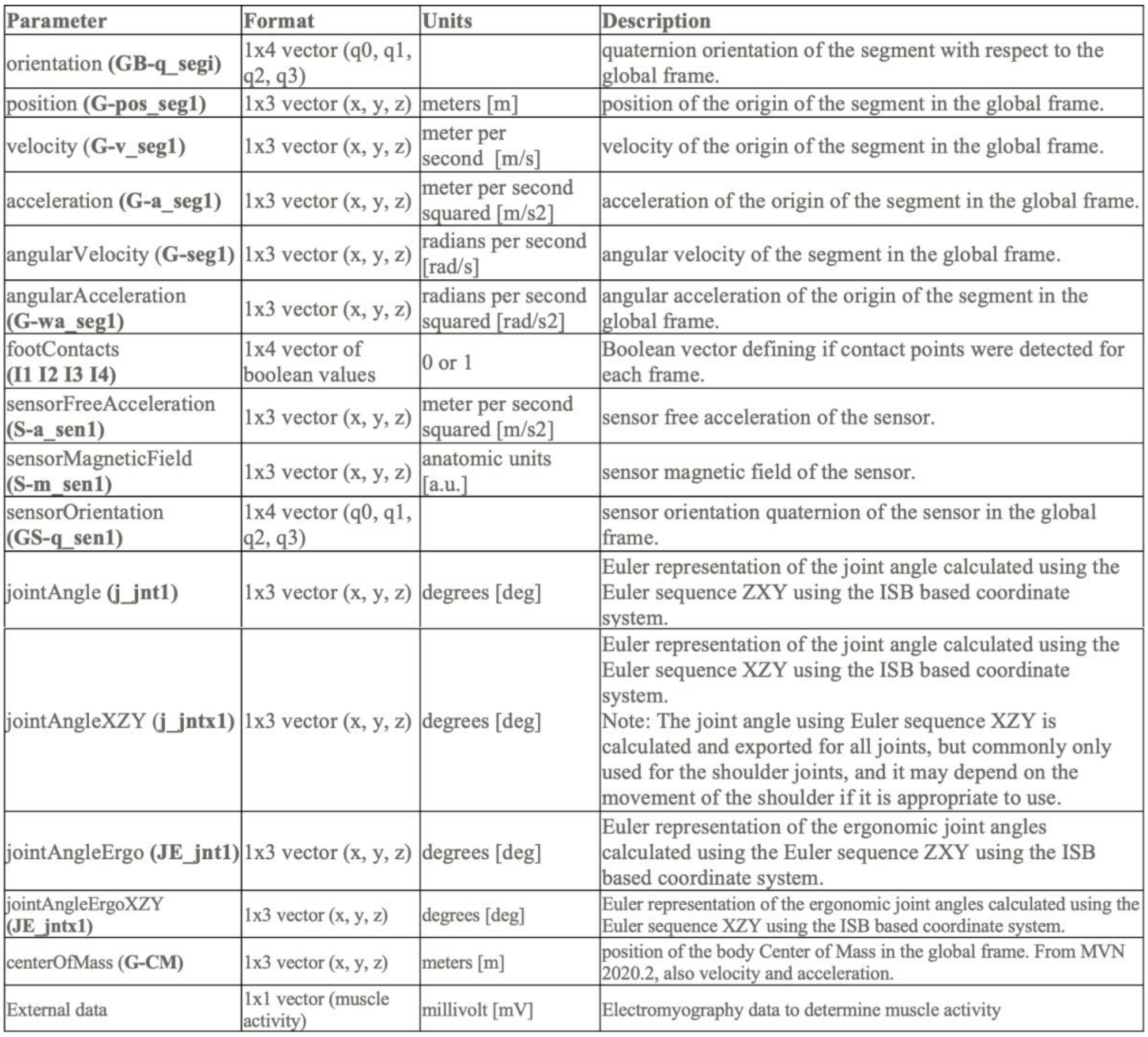
The structure and sequence of the frames as detailed above are illustrated in ***Figure 10***, providing a clear visual representation of the organization and content of data within the frames section.



```
<frames segmentCount="23" sensorCount="8" jointCount="22">
            <frame time="0" index="" tc="00:00:00:000" ms="0" type="identity">
                <orientation>1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0
0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1
0 0 0 1 0 0 0 1 0 0 0</orientation>
                <position>...</position>
            </frame>
            <frame time="0" index="" tc="00:00:00:000" ms="0" type="tpose">
                <orientation>...</orientation>
                <position>...</position>
            </frame>
            <frame time="0" index="" tc="00:00:00:000" ms="0" type="tpose-isb">
                <orientation>...</orientation>
                <position>...</position>
            </frame>
            <frame time="0" index="0" tc="08:29:20:050" ms="1517300960209" type="normal">
             <orientation>GB-q_seg1 GB-q_seg2 … GB-q_seg23</orientation>
             <position>G-pos_seg1 G-pos_seg2 … G-pos_seg23</position>
             <velocity>G-v_seg1 G-v_seg2 …G-v_seg23</velocity>
             <acceleration>G-a_seg1 G-a_seg2 … G-a_seg23</acceleration>
             <angularVelocity>G-w_seg1 G-w_seg2 … G-_wseg23</angularVelocity>
             <angularAcceleration>G-aw_seg1 G-aw_seg2 … G-aw_seg23</angularAcceleration>
             <footContacts>I1 I2 I3 I4 </footContacts>
             <sensorFreeAcceleration>S-a_sen1 S-a_sen2 … S-a_sen17</sensorFreeAcceleration>
             <sensorMagneticField>S-m_sen1 S-m_sen2 … S-m_sen17</sensorMagneticField>
             <sensorOrientation>GS-q_sen1 GS-q_sen2 … GS-q_sen17</sensorOrientation>
             <jointAngle>j_jnt1 j_jnt2 …j_jnt22</jointAngle>
             <jointAngleXZY>j_jntx1 j_jntx2 … j_jntx22</jointAngleXZY>
             <jointAngleErgo>JE_jnt1 JE_jnt2 … JE_jnt4</jointAngleErgo>
             <jointAngleErgoXZY>JE_jntx1 JE_jntx2 …JE_jntx4</jointAngleErgoXZY>
             <centerOfMass>G-CM</centerOfMass>
            </frame>
```

Figure 10 MVNX file section Frames

Parameters description:



Parameter	Format	Units	Description
orientation **(GB-q_segi)**	1x4 vector (q0, q1, q2, q3)		quaternion orientation of the segment with respect to the global frame.
position **(G-pos_seg1)**	1x3 vector (x, y, z)	meters [m]	position of the origin of the segment in the global frame.
velocity **(G-v_seg1)**	1x3 vector (x, y, z)	meter per second [m/s]	velocity of the origin of the segment in the global frame.
acceleration **(G-a_seg1)**	1x3 vector (x, y, z)	meter per second squared [m/s2]	acceleration of the origin of the segment in the global frame.
angularVelocity **(G-seg1)**	1x3 vector (x, y, z)	radians per second [rad/s]	angular velocity of the segment in the global frame.
angularAcceleration **(G-wa_seg1)**	1x3 vector (x, y, z)	radians per second squared [rad/s2]	angular acceleration of the origin of the segment in the global frame.
footContacts **(I1 I2 I3 I4)**	1x4 vector of boolean values	0 or 1	Boolean vector defining if contact points were detected for each frame.
sensorFreeAcceleration **(S-a_sen1)**	1x3 vector (x, y, z)	meter per second squared [m/s2]	sensor free acceleration of the sensor.
sensorMagneticField **(S-m_sen1)**	1x3 vector (x, y, z)	anatomic units [a.u.]	sensor magnetic field of the sensor.
sensorOrientation **(GS-q_sen1)**	1x4 vector (q0, q1, q2, q3)		sensor orientation quaternion of the sensor in the global frame.
jointAngle **(j_jnt1)**	1x3 vector (x, y, z)	degrees [deg]	Euler representation of the joint angle calculated using the Euler sequence ZXY using the ISB based coordinate system.
jointAngleXZY **(j_jntx1)**	1x3 vector (x, y, z)	degrees [deg]	Euler representation of the joint angle calculated using the Euler sequence XZY using the ISB based coordinate system. Note: The joint angle using Euler sequence XZY is calculated and exported for all joints, but commonly only used for the shoulder joints, and it may depend on the movement of the shoulder if it is appropriate to use.
jointAngleErgo **(JE_jnt1)**	1x3 vector (x, y, z)	degrees [deg]	Euler representation of the ergonomic joint angles calculated using the Euler sequence ZXY using the ISB based coordinate system.
jointAngleErgoXZY **(JE_jntx1)**	1x3 vector (x, y, z)	degrees [deg]	Euler representation of the ergonomic joint angles calculated using the Euler sequence XZY using the ISB based coordinate system.
centerOfMass **(G-CM)**	1x3 vector (x, y, z)	meters [m]	position of the body Center of Mass in the global frame. From MVN 2020.2, also velocity and acceleration.
External data	1x1 vector (muscle activity)	millivolt [mV]	Electromyography data to determine muscle activity

4.1.2 Converting MVNX Files to Pandas Data Frame

Overview

MVNX files, which are encoded in XML, contain complex and hierarchical biomechanical data that present challenges for direct analysis. To enable more efficient data manipulation and analysis, this data is parsed and converted into a pandas DataFrame. The DataFrame structure is favoured in data science due to its powerful analytical capabilities and compatibility with a plethora of data analysis tools.

Parsing the XML Data

The process begins with parsing the XML data. Using Python's **xml.etree.ElementTree** module [14], the MVNX XML file is read and converted into a tree structure. This structured representation allows for systematic extraction of data based on its hierarchical organization.

Constructing the DataFrame

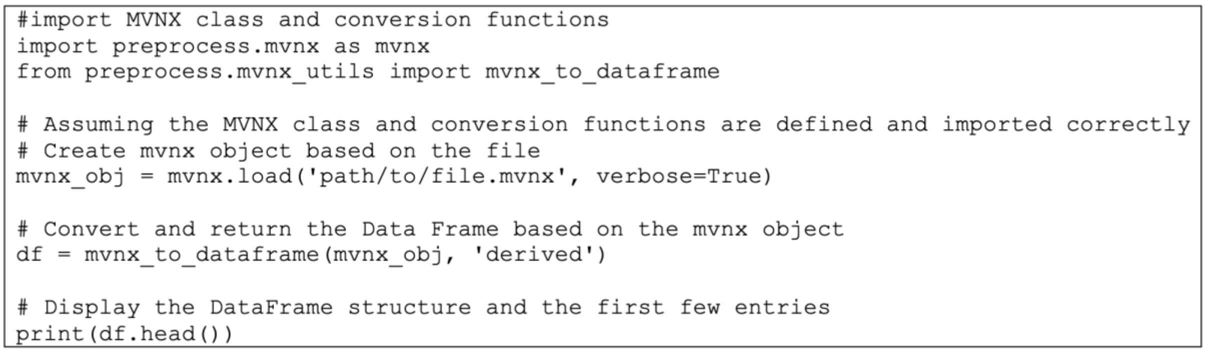
Frame-Based Data Organization: This involves iterating through each frame, extracting data related to various sensors and their metrics, and aligning this data temporally. Each piece of data is associated with a specific time point, ensuring chronological consistency across all data points.

Structuring DataFrame Columns: A separate column is created for each type of measurement from each sensor, such as orientation, position, or velocity, ensuring that each sensor's data is easily accessible. Systematic naming conventions are employed to name columns in a way that reflects the sensor, the type of metric, and the dimension (if applicable), such as 'RightFoot_Acceleration_X'. This naming strategy enhances clarity and facilitates easy data retrieval.

Finalizing the DataFrame:

The structured data is then imported into a pandas DataFrame. Each column represents a sensor and its corresponding metric, while each row represents a frame from the motion capture data.

The process of transforming the MVNX file to a pandas DataFrame and utilizing the structured data within Python is illustrated in the following ***Figure 11***. It visually outlines the steps taken to convert the complex motion capture data into an analytically accessible format.



```
#import MVNX class and conversion functions
import preprocess.mvnx as mvnx
from preprocess.mvnx_utils import mvnx_to_dataframe

# Assuming the MVNX class and conversion functions are defined and imported correctly
# Create mvnx object based on the file
mvnx_obj = mvnx.load('path/to/file.mvnx', verbose=True)

# Convert and return the Data Frame based on the mvnx object
df = mvnx_to_dataframe(mvnx_obj, 'derived')

# Display the DataFrame structure and the first few entries
print(df.head())
```

Figure 11 Python Code for Converting MVNX Data to pandas DataFrame

4.2 Data Visualization

This section emphasizes the crucial role of data visualization in analysing motion capture data, focusing on extracting and validating critical insights about the datasets. We delve into the utilization of visual tools, such as the Xsens Animate and Python, to ensure the integrity and reliability of the data before progressing to feature extraction. By employing these advanced visualization tools, we can thoroughly examine the datasets, identifying any discrepancies or anomalies that need to be addressed.

4.2.1 Visualization with Xsens Animate

This section explores the initial stage of data analysis where visual tools, specifically the Xsens Animate tool, play a crucial role

Tool Overview:

The Xsens Animate tool [19] is an advanced visualization platform specifically designed for motion capture data analysis. This tool transforms raw biomechanical data into detailed three-dimensional animations, providing a dynamic and intuitive means to observe and analyse the complex movements of athletes in real-time.

Technical Features:

Sensor Integration: Xsens Animate seamlessly integrates data from multiple sensors placed on the athlete's body, synthesizing the inputs to create a cohesive animation that reflects the actual movements performed during the capture session.

Frame-by-Frame Analysis: The tool enables frame-by-frame scrutiny of the captured data, an essential feature for identifying precise moments in a performance where biomechanical efficiencies or inefficiencies appear. Paying close attention to little details is crucial when analysing complex actions such as running, as even slight variations over time can significantly impact the efficiency of each step.

Customizable Views: Researchers can customize how data is viewed within the tool, choosing from various display options that highlight different biomechanical aspects, such as skeletal movements, joint angles, or center of mass.

Insights from Xsens Animate

Sensor Placement Observations: During the analysis, it became apparent that the incorrect placement of sensors, particularly at crucial points like the heel and toe, significantly affected the accuracy of the data. ***Figures 12*** illustrate cases where sensors were not correctly placed, leading to erroneous representations of foot mechanics. These errors were critical as they affected the fundamental analysis of foot strike patterns and overall foot dynamics, which are essential for accurate biomechanical assessment.

Additionally, in other captures, sensors incorrectly positioned on the athletes' legs further compromised the kinematic analysis. This misplacement not only distorted the accuracy of leg movement data but also introduced irregular motion patterns that did not truly reflect the athletes' actual performances. Such inaccuracies were particularly problematic for analysing the precise mechanics of running, where detailed leg movements are critical for a thorough understanding of athletic efficiency.

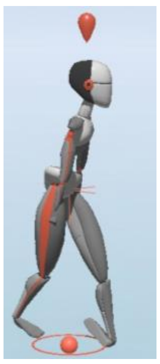


Figure 12 Example of incorrect placement of sensors

Data Exclusion Decisions: Based on the observed sensor errors, decisions were made to exclude certain datasets from further analysis. The process of selectively excluding certain data was crucial in preserving the integrity of the study findings, guaranteeing that only precise and pertinent information was analysed. Implementing such an indicator was necessary to maintain the study's rigorous requirements and ensure reliable insights on sports performance.

Protocols for Future Data Collection: The incident also emphasised the significance of implementing careful sensor setup protocols to prevent similar issues in future captures and studies.

Visual Insights for Feature Selection: The Xsens Animate tool played a vital role in emphasising important visual aspects that guided the process of extracting features. Through careful visual observation of motion dynamics, we determined crucial characteristics that are necessary for thorough analysis. This sets the foundation for the discusses in the Feature Extraction section.

4.2.2 Visualization with Python

This section explores how Python was utilized to complement and extend the insights gained from Xsens Animate and other visualization tools. It will showcase several plots created with Python, discussing how these visualizations support and deepen the initial observations made with dynamic tools like Xsens Animate. The analysis presented here will focus on deriving and explaining the insights that these Python-generated visualizations provide about the motion capture data, further enhancing the feature extraction process by substantiating the findings from Xsens with additional visual evidence.

Right and Left Foot Velocity Analysis

Figure 13 provides a comprehensive analysis of the velocities for both the left and right feet of an athlete, using color-coded indicators to mark ground contact points.

- **Contact Points Indication**: Blue dots signify the right foot's contact with the ground, and red dots represent the same for the left foot.
- **Left Foot Analysis (*Figure 13 - Top Graph*)**: Each red dot, marking the contact of the left foot, is followed by a sharp increase in velocity. This pattern indicates the foot's lift-off, highlighting the critical phase of toe push-off that propels the athlete forward.
- **Right Foot Analysis (*Figure 13 - Bottom Graph*)**: Similarly, for the right foot, the blue dots are succeeded by a rise in velocity. This consistent pattern suggests a toe push-off preceding the foot's lift-off.

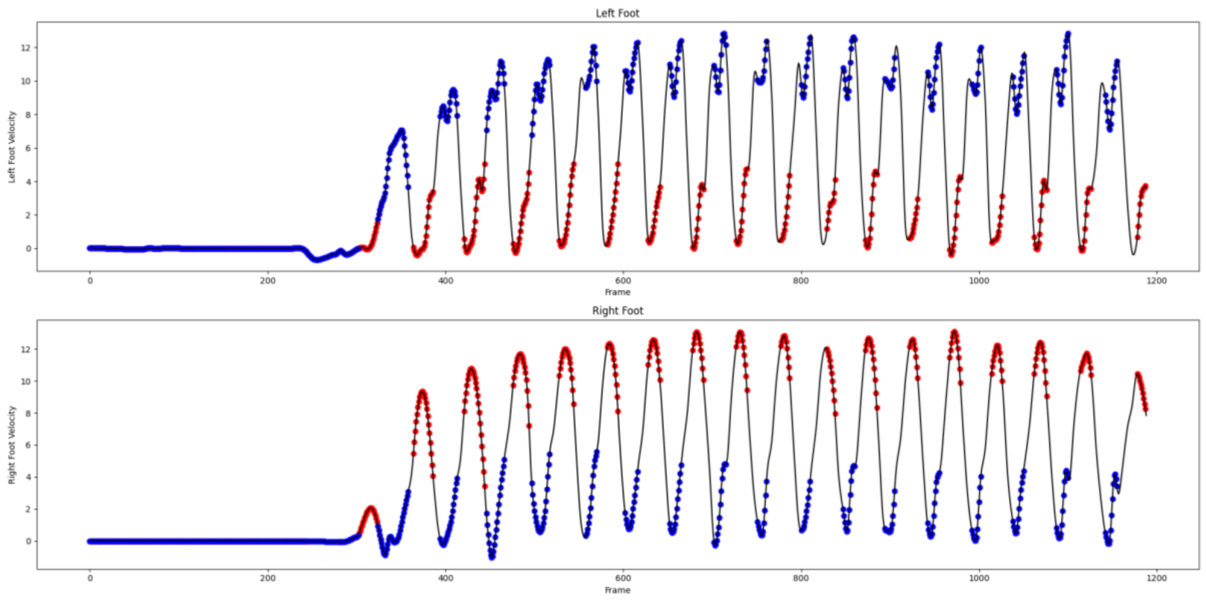


Figure 13 Right and Left Foot Velocity Analysis

Insights from Foot Velocity Analysis:

Consistent Technique and Rhythm: Beyond the approximate 350th frame, a repetitive pattern emerges in both the left and right foot velocity graphs, indicating a consistent running technique and rhythm. This uniform motion is suggestive of efficient energy utilization, which is characteristic of an athlete's optimal running form.

Determining the Beginning of Running Movement: As evident in ***Figure 13***, the initial segments of the capture depict the athlete with a velocity near zero, highlighting a period of stationary rest or preparatory activity. This phase persists until approximately the 350th frame, at which point a noticeable increase in velocity indicates the onset of sustained running. This transition from preparatory activities to actual running is crucial. Accurately identifying this shift is pivotal in data pre-processing to ensure that analyses focus only on frames that capture the full essence of the running motion. The objective is to isolate and retain data from the precise moment an athlete begins running until they complete a specified distance. Implementing this practice across all datasets sharpens the focus and relevance of the analysis, enabling a more accurate assessment of running mechanics and performance.

Differential Ground Contact Analysis

Figure 14 highlights a clear contrast in ground contact data between the athlete's left and right feet. The graphs depict foot velocities over a series of frames again using color-coded indicators to mark ground contact points.

Consistency in Right Foot Contacts: The sequence of blue markers on the bottom graph outlines a consistent pattern of ground contacts for the right foot. Each foot strike, signified by a blue marker, is followed by a peak in velocity, which corresponds to the foot's lift-off, indicative of an established running rhythm.

Absence of Left Foot Contacts: Contrarily, the absence of red markers on both graphs is conspicuous, suggesting an atypical lack of ground contact throughout the captured frames. This anomaly provides a clear visual confirmation of the sensor placement issues previously identified with the Xsens Animate tool, underscoring a critical data integrity issue that needs to be addressed.

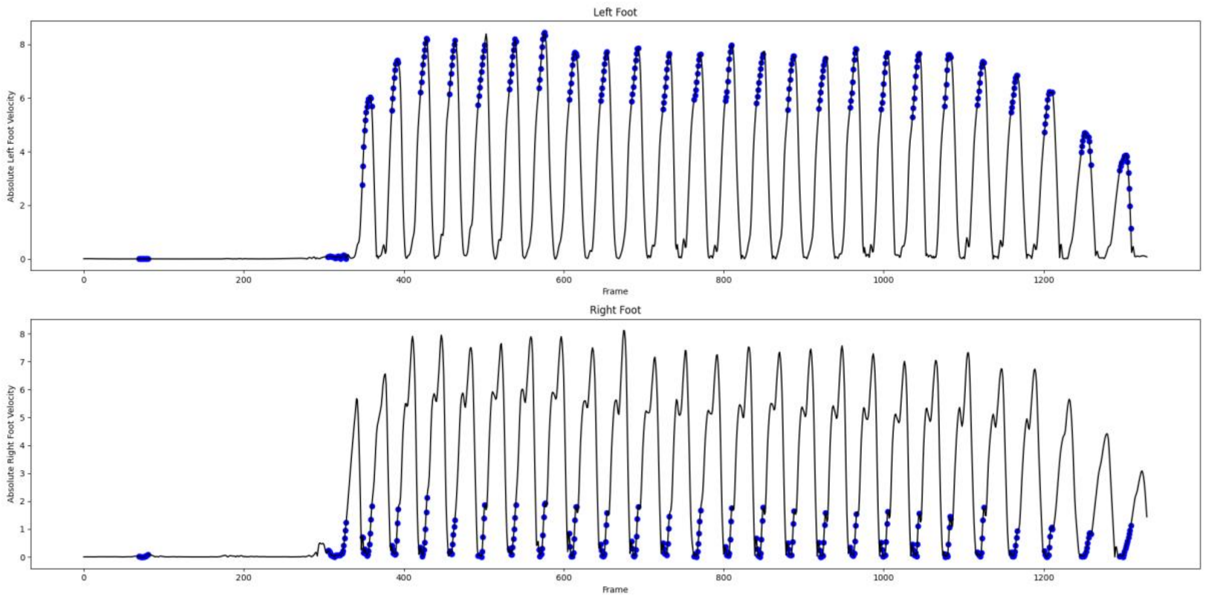


Figure 14 Comparative Ground Contact Analysis for Left and Right Feet

Resolving Data Integrity Issues

The insights drawn from ***Figure 14*** have necessitated decisive action regarding the datasets in question. The notable lack of ground contact data for the left foot corroborates earlier concerns identified with Xsens animation, issues of sensor misplacement or data capture errors. Such findings undermine the reliability of the affected datasets for any substantive biomechanical analysis.

To maintain the scientific rigor of this study, we have determined that datasets exhibiting these discrepancies, specifically those with incorrect sensor placement or problematic capture data cannot be utilized. Consequently, these datasets will be excluded from further analysis. This careful curation ensures that subsequent evaluations, particularly in feature extraction and model training, are based on data of uncompromised accuracy and consistency.

4.3 Feature Extraction

4.3.1 Manual Feature Extraction

In the manual feature extraction phase, we dedicated our efforts to identifying and quantifying crucial aspects of athletic performance tailored to varying effort levels and distances. Through a meticulous analysis of four distinct captures per athlete—two at maximum effort and two at sub-maximum effort, across both 30-meter and 50-meter distances—we derived valuable data points. These data points effectively capture the subtleties of running mechanics, providing insights into the dynamic interplay of biomechanical factors under different running conditions. This careful examination allows us to understand and quantify how athletes adjust their performance strategies across varying intensities and distances, revealing critical elements that influence overall athletic output.

1) Total Steps

Objective: The total number of steps taken by an athlete over a set distance provides insight into their running economy and efficiency. Analysing the step count helps in understanding how athletes manage their energy and pace across different effort levels and distances.

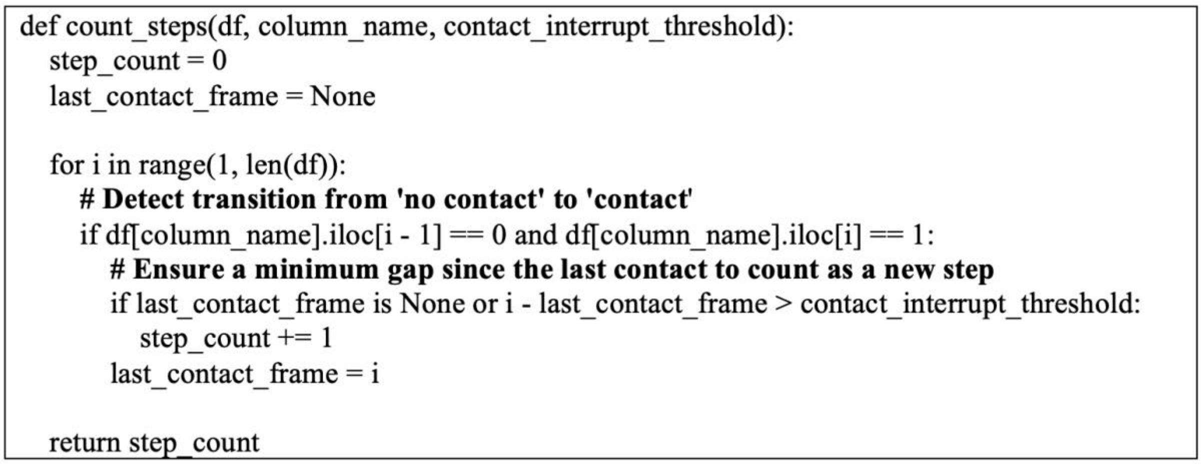
Methodology

Data Preparation

Before counting steps, the motion capture data is carefully pre-processed and loaded into a pandas dataframe. This preparation follows a specific pre-processing function outlined in the previous section, which formats and structures the data to ensure it is ready for detailed analysis. This step involves transforming raw data into a more analysable form by organizing it into clearly defined columns for each sensor metric, including ground contact indicators for both feet.

Algorithm for Calculation:

To accurately quantify the total steps taken by an athlete for each foot during a capture session, we employ a custom Python function named **count_steps**. This function meticulously analyses binary ground contact data from the motion capture system. Each entry in the dataframe corresponds to a frame of motion capture, where a '1' indicates that the foot is in contact with the ground, representing a foot strike, and '0' indicates no contact. This binary system allows precise detection of each step as the foot transitions from air to ground contact. The detailed implementation of this function is illustrated in the Python code shown in ***Figure 15***.



```python
def count_steps(df, column_name, contact_interrupt_threshold):
    step_count = 0
    last_contact_frame = None

    for i in range(1, len(df)):
        # Detect transition from 'no contact' to 'contact'
        if df[column_name].iloc[i - 1] == 0 and df[column_name].iloc[i] == 1:
            # Ensure a minimum gap since the last contact to count as a new step
            if last_contact_frame is None or i - last_contact_frame > contact_interrupt_threshold:
                step_count += 1
            last_contact_frame = i

    return step_count
```

Figure 15 Python Code for the count_steps Function

Analysis Process:

The function iterates through the specified column, identifying transitions from 'no contact' (0) to 'contact' (1), signalling the initiation of a new step.

A new step is counted only if the interval since the last registered contact exceeds the **contact_interrupt_threshold**, effectively filtering out false step counts from brief or accidental contacts. For example, the sensor data switches from '1' (contact) at frame 10 to '0' (no contact) at frame 11, and back to '1' at frame 12—a scenario that's unlikely to represent actual steps due to the physical impossibility of such rapid consecutive foot contacts within just 10 milliseconds.

The function returns the count of steps for the specified foot, allowing separate analysis for each foot's contact data.

Insights from Total Steps Feature

A. Comparative Analysis of Running Efficiency

Scenario 1: Increased Steps with Maximum Effort

During maximum effort, an athlete took 31 steps to cover the distance in 7540 milliseconds, whereas, under sub-maximum effort, the distance was covered in 28 steps over 7700 milliseconds. Notably, at maximum effort, there is a visible increase in tension and a breakdown in running technique, as observed from video analysis using Xsens Animation. This tension appears to disrupt the runner's optimal form, resulting in increased step count and less efficient movement.

Despite the increased effort, this does not translate into improved performance but rather contributes to a less efficient and potentially more fatiguing running style.

Scenario 2: Consistent Steps Across Efforts

In a contrasting example, the athlete consistently covers the distance in 29 steps regardless of the effort level. However, the times differ significantly: 7050 milliseconds at maximum effort vs. 7800 milliseconds at sub-maximum effort. This scenario illustrates that maintaining a consistent step count and technique without excessive tension leads to more efficient and faster performances. The absence of technique breakdown allows the athlete to utilize their energy more effectively, demonstrating that a relaxed but consistent running form can enhance overall speed and efficiency.

Conclusion: Enhancing Predictive Modelling

The analysis of total steps across different effort levels and distances provides valuable data that can significantly enhance the predictive modelling of competition times. Incorporating step count data helps to accurately forecast the effects of different running techniques and efforts on performance outcomes, making it invaluable for optimizing performance in competitive settings.

B. Injury Prevention and Management:

Consistency and Asymmetry: Tracking the consistency of step counts across different sessions and comparing left/right foot contacts can help in identifying asymmetrical running patterns, which are often precursors to injuries. Adjustments in training can be made to correct these imbalances.

2) Ground Contact Time (GCT) and Average GCT

Objective: Ground Contact Time (GCT) quantifies the duration each foot is in contact with the ground during a stride. This metric is fundamental for analysing an athlete's running biomechanics [4][11], assessing efficiency, and identifying potential areas for speed enhancement. Analysing both the individual GCT and average GCT across different running conditions provides insights into technique, fatigue, and efficiency.

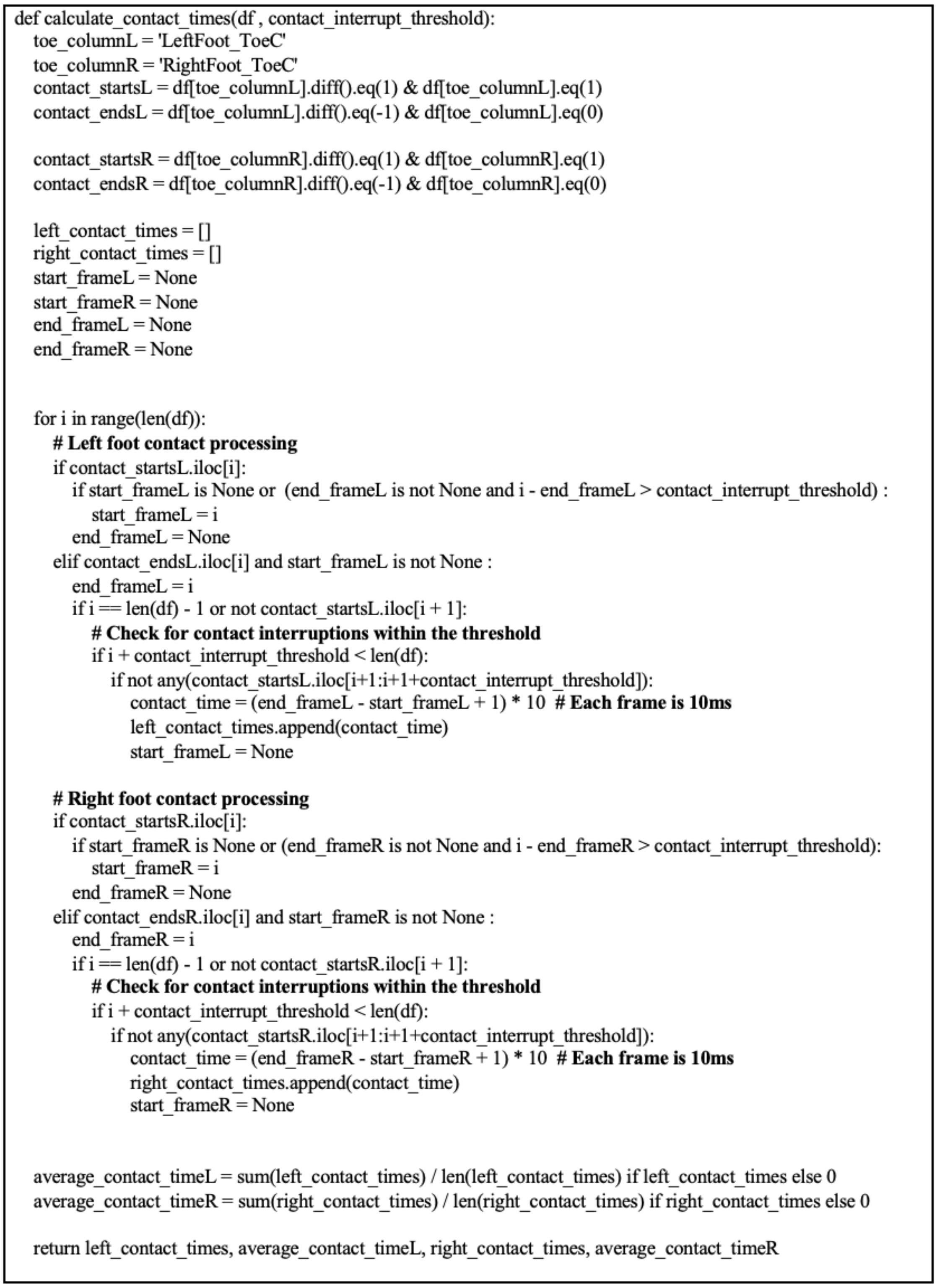
Methodology

Data Preparation:

Before measuring the GCT, the motion capture data is carefully pre-processed and loaded into a pandas dataframe. This preparation follows a specific pre-processing function outlined in the previous section, which formats and structures the data to ensure it is ready for detailed analysis. This step involves transforming raw data into a more analysable form by organizing it into clearly defined columns for each sensor metric, including ground contact indicators for both feet.

Algorithm for Calculation:

The calculation of Ground Contact Times (GCT) for both the left and right feet during a capture session is conducted using a custom Python function, **calculate_contact_times**. This function is designed to process binary ground contact data, which indicates whether each foot is in contact with the ground at each frame captured by the motion capture system. A value of '1' in the data signifies that the foot is in contact with the ground (foot strike), and a value of '0' indicates no contact (foot is in the air). This binary approach allows for precise identification of each contact phase, from the initial touch to the lift-off. The detailed implementation of this function is illustrated in the Python code shown in ***Figure 16***.



```python
def calculate_contact_times(df , contact_interrupt_threshold):
    toe_columnL = 'LeftFoot_ToeC'
    toe_columnR = 'RightFoot_ToeC'
    contact_startsL = df[toe_columnL].diff().eq(1) & df[toe_columnL].eq(1)
    contact_endsL = df[toe_columnL].diff().eq(-1) & df[toe_columnL].eq(0)

    contact_startsR = df[toe_columnR].diff().eq(1) & df[toe_columnR].eq(1)
    contact_endsR = df[toe_columnR].diff().eq(-1) & df[toe_columnR].eq(0)

    left_contact_times = []
    right_contact_times = []
    start_frameL = None
    start_frameR = None
    end_frameL = None
    end_frameR = None


    for i in range(len(df)):
        # Left foot contact processing
        if contact_startsL.iloc[i]:
            if start_frameL is None or  (end_frameL is not None and i - end_frameL > contact_interrupt_threshold) :
                start_frameL = i
            end_frameL = None
        elif contact_endsL.iloc[i] and start_frameL is not None :
            end_frameL = i
            if i == len(df) - 1 or not contact_startsL.iloc[i + 1]:
                # Check for contact interruptions within the threshold
                if i + contact_interrupt_threshold < len(df):
                    if not any(contact_startsL.iloc[i+1:i+1+contact_interrupt_threshold]):
                        contact_time = (end_frameL - start_frameL + 1) * 10  # Each frame is 10ms
                        left_contact_times.append(contact_time)
                        start_frameL = None

        # Right foot contact processing
        if contact_startsR.iloc[i]:
            if start_frameR is None or (end_frameR is not None and i - end_frameR > contact_interrupt_threshold):
                start_frameR = i
            end_frameR = None
        elif contact_endsR.iloc[i] and start_frameR is not None :
            end_frameR = i
            if i == len(df) - 1 or not contact_startsR.iloc[i + 1]:
                # Check for contact interruptions within the threshold
                if i + contact_interrupt_threshold < len(df):
                    if not any(contact_startsR.iloc[i+1:i+1+contact_interrupt_threshold]):
                        contact_time = (end_frameR - start_frameR + 1) * 10  # Each frame is 10ms
                        right_contact_times.append(contact_time)
                        start_frameR = None


    average_contact_timeL = sum(left_contact_times) / len(left_contact_times) if left_contact_times else 0
    average_contact_timeR = sum(right_contact_times) / len(right_contact_times) if right_contact_times else 0

    return left_contact_times, average_contact_timeL, right_contact_times, average_contact_timeR
```

Figure 16 Python Code for the calculate_contact_times Function

Analysis Process:

1. **Identification of Contact Events:**

   Contact Start: A ground contact event starts when the diff() function applied to the toe_column identifies a transition from '0' (no contact) to '1' (contact). This indicates that the foot has just touched the ground.

Contact End: A ground contact event ends when the transition goes from '1' to '0', indicating the foot has left the ground.

2. **Recording Contact Durations:**
   For both feet, the function tracks the frame number when each contact starts and ends. If a contact event meets the criteria of not being a false positive (determined by the contact_interrupt_threshold which filters out any minor breaks in contact that don't surpass the threshold) the duration of the contact is calculated.
3. **Calculation of Contact Times:**
   The duration for each valid contact event is calculated by subtracting the start frame from the end frame, then multiplying by the frame interval (10 milliseconds in this context). This provides the contact time in milliseconds.
   These individual contact times are collected into lists for both the left and right feet.
4. **Calculation of Average Contact Times:**
   The average contact time for each foot is computed by taking the mean of all recorded contact times. This average provides a measure of the typical ground contact duration for the session.

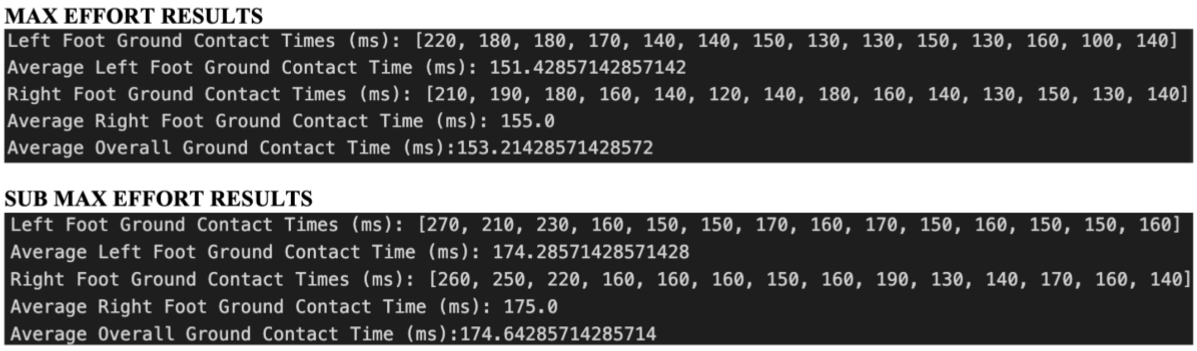
Insights GCT Analysis

This section synthesizes observations from the ground contact time (GCT) analyses, comparing performance across different effort levels and between athletes. The insights are directly linked to biomechanical efficiencies, which are in alignment with established research on running dynamics.

Comparative Analysis Within an Athlete's Performances

From detailed observations of a single athlete's performances, we noted distinct variations in their running efficiency under different conditions. For instance, during a test, the athlete consistently took 29 steps to cover 50 meters, yet the time taken varied with the effort level. At maximum effort, the distance was covered in 7.050 seconds, compared to 7.800 seconds at sub-maximum effort. This 0.750-second difference underscores the influence of exertion level on speed.

Further inspection reveals that ground contact time significantly decreases during maximum effort compared to sub-maximum effort. For example, Figure 17 shows that during maximum effort, the athlete reduced their ground contact time by about 20 milliseconds per step. Over the course of 29 steps, this adjustment cumulatively saved approximately 0.580 seconds due to shorter ground contacts. This observation underscores how effective management of ground contact time can significantly boost running speed, supporting the principle that minimizing ground contact time is essential for enhancing overall running efficiency.



MAX EFFORT RESULTS

```
Left Foot Ground Contact Times (ms): [220, 180, 180, 170, 140, 140, 150, 130, 130, 150, 130, 160, 100, 140]
Average Left Foot Ground Contact Time (ms): 151.42857142857142
Right Foot Ground Contact Times (ms): [210, 190, 180, 160, 140, 120, 140, 180, 160, 140, 130, 150, 130, 140]
Average Right Foot Ground Contact Time (ms): 155.0
Average Overall Ground Contact Time (ms):153.21428571428572
```

SUB MAX EFFORT RESULTS

```
Left Foot Ground Contact Times (ms): [270, 210, 230, 160, 150, 150, 170, 160, 170, 150, 160, 150, 150, 160]
Average Left Foot Ground Contact Time (ms): 174.28571428571428
Right Foot Ground Contact Times (ms): [260, 250, 220, 160, 160, 160, 150, 160, 190, 130, 140, 170, 160, 140]
Average Right Foot Ground Contact Time (ms): 175.0
Average Overall Ground Contact Time (ms):174.64285714285714
```

Figure 17 Ground Contact Time Comparisons for Max Effort vs. Sub Max Effort

Integration with Established Research

Corroboration with Novacheck's [11] **Findings:** Our study's observations align with findings from Tom F. Novacheck, which reveal that elite sprinters achieve remarkably short Ground Contact Times (GCTs), thereby enhancing their speed and efficiency. Notably, Novacheck pointed out that world-class sprinters can achieve toe-off as early as 22% of the gait cycle, signifying superior biomechanical efficiency, as illustrated in ***Figure 18*** from his study "The Biomechanics of Running." (Fig.3 in paper).

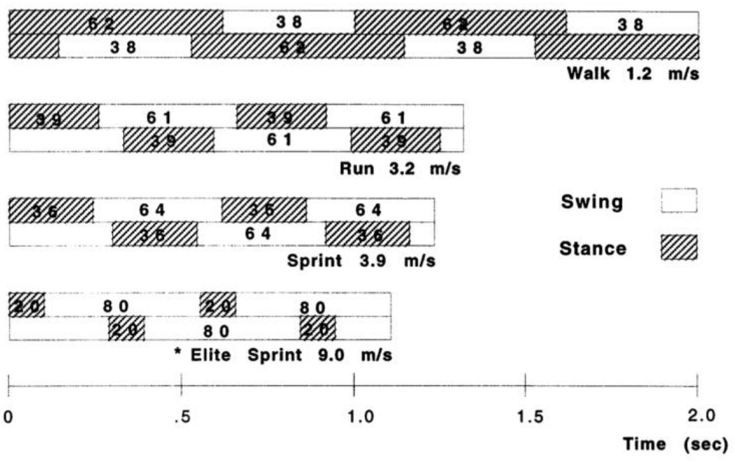


Figure 18 Ground Contact Time and Air Time Across Running Speeds, Adapted from 'The Biomechanics of Running' by Tom F. Novacheck [11] (Fig.3)

Scientific Correlation: These findings are consistent with those reported by Novacheck, who observed that faster runners, particularly elite sprinters, spend significantly less time in stance phases. This reduced stance time translates into shorter GCTs, which are critical for achieving high speeds.

Biomechanical Implications: Our empirical data demonstrates that minimized GCTs correlate strongly with increased running speeds [4][7][11], providing real-world examples of how biomechanical principles can enhance athletic performance. The detailed analysis of GCT across different effort levels substantiates the importance of efficient ground contact management as a key factor in optimizing running mechanics.

Conclusion: Enhancing Predictive Modelling

The insights garnered from the analysis of GCT are invaluable for enhancing the predictive accuracy of models that forecast competition times. By integrating GCT metrics into our models, we can predict how modifications in running technique and effort levels affect an athlete's performance. These metrics prove especially potent in predicting the completion times of races, thereby offering athletes and coaches precise data to fine-tune training and competition strategies.

3) Air Time and Average Air Time

Objective: The objective of analysing Air Time and Average Air Time is to quantify the phase during which an athlete's feet are not in contact with the ground during a running stride. This measurement is crucial for understanding the dynamics of an athlete's gait, particularly the efficiency of their running mechanics and the effectiveness of their stride [11].

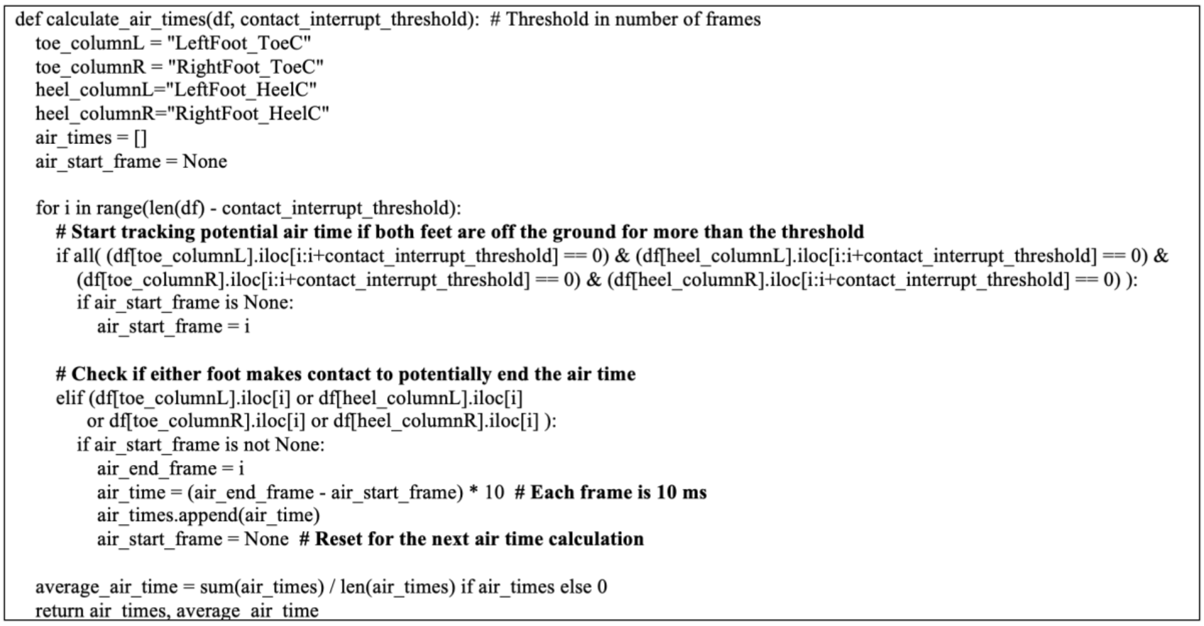
Methodology

Data Preparation:

Prior to calculating Air Time, the motion capture data is processed and organized into a pandas DataFrame as outlined in previous sections. This preparation involves structuring the data to facilitate easy access to specific metrics such as toe and heel contact points for both feet, crucial for determining when both feet are off the ground.

Algorithm for Calculation:

The **calculate_air_times** function is designed to quantify the duration for which both feet of an athlete are airborne during a stride. This function examines sequences of frames to determine continuous periods when both the toes and heels of both feet are not in contact with the ground, surpassing a predefined threshold to ensure it captures genuine Air Time and not brief or accidental losses of contact. The detailed implementation of this function is illustrated in the Python code shown in ***Figure 19***.



```
def calculate_air_times(df, contact_interrupt_threshold):  # Threshold in number of frames
    toe_columnL = "LeftFoot_ToeC"
    toe_columnR = "RightFoot_ToeC"
    heel_columnL="LeftFoot_HeelC"
    heel_columnR="RightFoot_HeelC"
    air_times = []
    air_start_frame = None

    for i in range(len(df) - contact_interrupt_threshold):
        # Start tracking potential air time if both feet are off the ground for more than the threshold
        if all( (df[toe_columnL].iloc[i:i+contact_interrupt_threshold] == 0) & (df[heel_columnL].iloc[i:i+contact_interrupt_threshold] == 0) &
            (df[toe_columnR].iloc[i:i+contact_interrupt_threshold] == 0) & (df[heel_columnR].iloc[i:i+contact_interrupt_threshold] == 0) ):
            if air_start_frame is None:
                air_start_frame = i

        # Check if either foot makes contact to potentially end the air time
        elif (df[toe_columnL].iloc[i] or df[heel_columnL].iloc[i]
              or df[toe_columnR].iloc[i] or df[heel_columnR].iloc[i] ):
            if air_start_frame is not None:
                air_end_frame = i
                air_time = (air_end_frame - air_start_frame) * 10  # Each frame is 10 ms
                air_times.append(air_time)
                air_start_frame = None  # Reset for the next air time calculation

    average_air_time = sum(air_times) / len(air_times) if air_times else 0
    return air_times, average_air_time
```

Figure 19 Python Code for the calculate_air_times Function

Analysis Process:

Contact Detection: The function iterates through the DataFrame, checking for intervals where both the toe and heel of each foot register no contact (**0**) continuously over a set threshold of frames. This method helps in identifying the start of an actual Air Time period, minimizing false detections.

Airtime Calculation: Once a potential Air Time start is identified, the function looks for any foot contact (**toe** or **heel** changing to **1**) to mark the end of Air Time. The duration is calculated based on the number of frames between the start and end of the Air Time, with each frame representing 10 milliseconds.

Average Air Time: The function finally computes the average Air Time across all detected periods within a session, providing a metric of the athlete's average flight phase per stride.

Insights from Air Time Analysis

In this analysis, we examine the air time metrics of two sprint sessions involving the same athletes, referred to as Runner A and Runner B, across two distances: 30 meters and 50 meters. This approach allows us to assess their consistency and changes in performance under different conditions.

Correlation with Biomechanical Research: Our data corroborates the biomechanical insights presented by Tom Novacheck [11], specifically his observation that as speed increases, so does the time spend in the swing phase (air time). This phenomenon is vividly illustrated in ***Figure 18*** and further supported by our comparative analysis of two athletes over distances of 30 meters and 50 meters.

Comparative Performance Analysis:

30-meter Performance Comparison:

In the 30-meter performance comparison, Runner A completed the distance in 3.05 seconds, taking 19 steps with an average overall airtime of 74.21 milliseconds. Runner B outperformed this, finishing faster in 2.7 seconds and with fewer steps (17), despite having a slightly higher average overall airtime of 78.75 milliseconds. This indicates that Runner B's more effective utilization of vertical force may have enhanced their overall speed.

50-meter Performance Comparison:

For the 50-meter race, Runner A finished in 4.86 seconds, taking 31 steps with an average overall airtime of 82 milliseconds. Runner B again showed superior performance, covering the distance in a quicker 4.46 seconds using only 26 steps, and recording an average overall airtime of 85.38 milliseconds. This consistent ability of Runner B to maintain more airtime and efficiently utilize vertical force significantly contributes to their quicker finishes in both distances.

Biomechanical Implications and Conclusion:

These comparisons highlight the critical role of airtime in sprinting efficiency. Extending the duration of the swing phase not only allows athletes to optimize stride mechanics but also helps reduce step count and improve overall race times. The consistent results across both distances for Runners A and B illustrate how enhancing airtime can significantly enhance performance. This aligns with Novacheck's biomechanical theories, providing empirical support for theoretical concepts and offering actionable insights for training programs focused on maximizing biomechanical advantages to boost competitive sprinting performance.

Conclusion: Enhancing Predictive Modelling

The detailed analysis of air time and average air time enhances the accuracy of predictive models aimed at forecasting athletes' completion times. By integrating these metrics, models can more precisely account for the dynamic elements of running mechanics that directly influence speed and efficiency. This enables more accurate predictions and tailored training approaches that can significantly improve athletes’ competitive performance.

4) Stride Length

Objective:

The objective of analysing Stride Length is to measure the distance covered in each step during a running stride. Stride Length is a critical metric for assessing the efficiency of a runner's gait. In sprinting, longer strides are often associated with greater speed and efficiency, provided they can be maintained without excessive energy expenditure [11][15]. In distance running, optimal stride length helps maintain endurance and speed over longer periods, balancing energy conservation with pace [12].

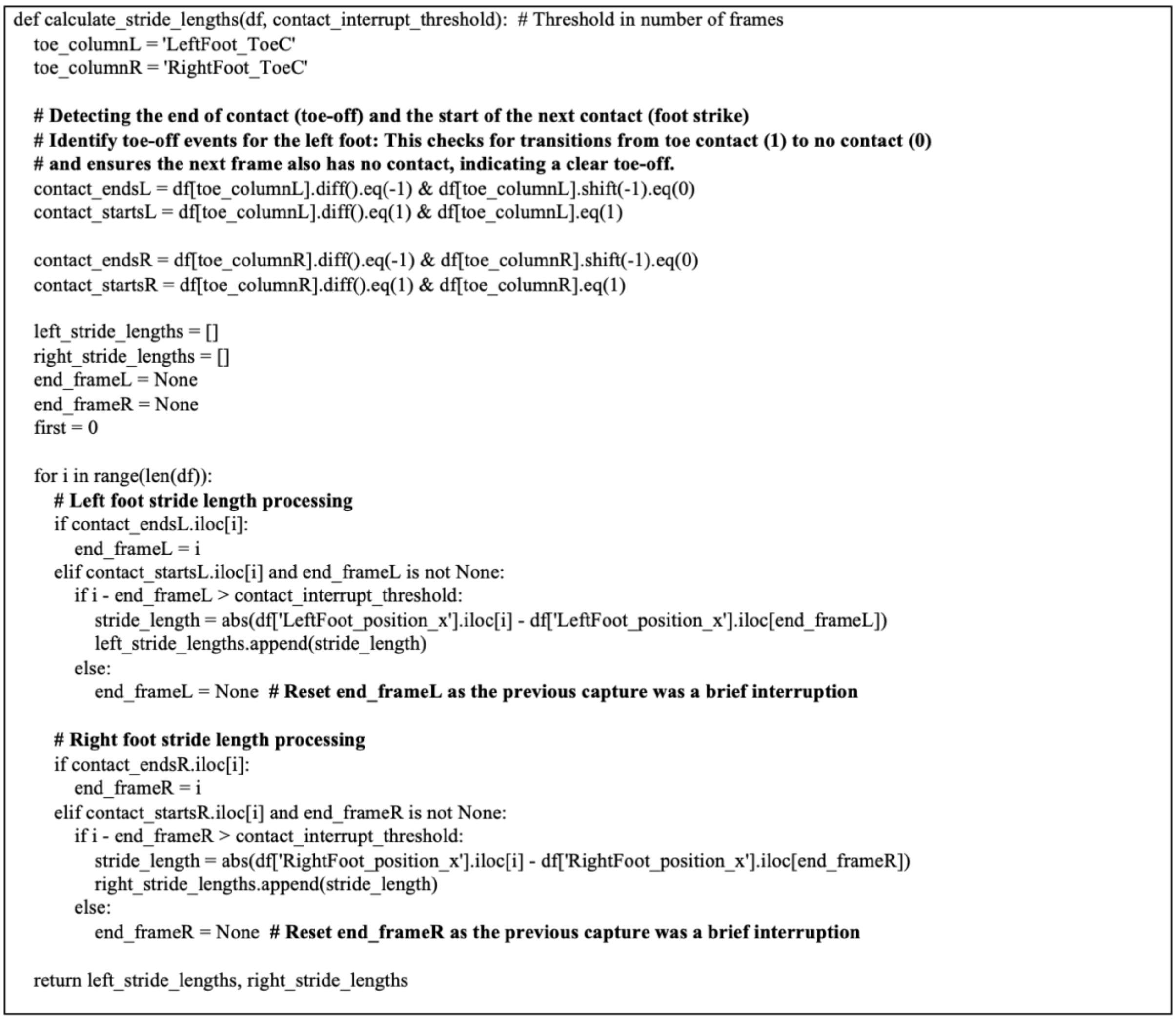
Methodology

Data Preparation:

Before calculating Stride Length, it is essential to ensure the motion capture data is pre-processed and organized correctly into a pandas DataFrame, as previously outlined. The data should include accurate positional information for each foot, which is critical for measuring the distance between consecutive foot contacts of the same foot.

Algorithm for Calculation:

The **calculate_stride_lengths** function is carefully designed to compute the stride lengths for both the left and right feet by determining the distance between successive ground contacts made by the same foot. This measurement is critical for assessing the efficiency and mechanics of a runner’s gait. The detailed implementation of this function is illustrated in the Python code shown in ***Figure 20***.



```python
def calculate_stride_lengths(df, contact_interrupt_threshold):  # Threshold in number of frames
    toe_columnL = 'LeftFoot_ToeC'
    toe_columnR = 'RightFoot_ToeC'

    # Detecting the end of contact (toe-off) and the start of the next contact (foot strike)
    # Identify toe-off events for the left foot: This checks for transitions from toe contact (1) to no contact (0)
    # and ensures the next frame also has no contact, indicating a clear toe-off.
    contact_endsL = df[toe_columnL].diff().eq(-1) & df[toe_columnL].shift(-1).eq(0)
    contact_startsL = df[toe_columnL].diff().eq(1) & df[toe_columnL].eq(1)

    contact_endsR = df[toe_columnR].diff().eq(-1) & df[toe_columnR].shift(-1).eq(0)
    contact_startsR = df[toe_columnR].diff().eq(1) & df[toe_columnR].eq(1)

    left_stride_lengths = []
    right_stride_lengths = []
    end_frameL = None
    end_frameR = None
    first = 0

    for i in range(len(df)):
        # Left foot stride length processing
        if contact_endsL.iloc[i]:
            end_frameL = i
        elif contact_startsL.iloc[i] and end_frameL is not None:
            if i - end_frameL > contact_interrupt_threshold:
                stride_length = abs(df['LeftFoot_position_x'].iloc[i] - df['LeftFoot_position_x'].iloc[end_frameL])
                left_stride_lengths.append(stride_length)
            else:
                end_frameL = None  # Reset end_frameL as the previous capture was a brief interruption

        # Right foot stride length processing
        if contact_endsR.iloc[i]:
            end_frameR = i
        elif contact_startsR.iloc[i] and end_frameR is not None:
            if i - end_frameR > contact_interrupt_threshold:
                stride_length = abs(df['RightFoot_position_x'].iloc[i] - df['RightFoot_position_x'].iloc[end_frameR])
                right_stride_lengths.append(stride_length)
            else:
                end_frameR = None  # Reset end_frameR as the previous capture was a brief interruption

    return left_stride_lengths, right_stride_lengths
```

Figure 20 Python Code for the calculate_stride_lengths Function

Analysis Process:

1. **Identification of Contact Events:**

    Contact End (Toe-Off): The end of a ground contact event (toe-off) is detected when the diff() function applied to the toe contact columns (toe_columnL and toe_columnR) identifies a transition from '1' (contact) to '0' (no contact). This transition indicates that the foot has lifted off the ground.

    Subsequent Foot Strike: After a toe-off event, the subsequent foot strike event is identified when the transition goes from '0' to '1', indicating the foot has touched the ground again.

2. **Recording Stride Events:**

    Tracking Events: For both the left and right feet, the function records the frame numbers where each toe-off and the next corresponding foot strike of the same foot. It checks these events to ensure they are separated by more than the contact_interrupt_threshold to avoid counting minor, false-positive displacements as new strides.

Distance Measurement: Once a valid pair of toe-off and foot strike events is identified, the horizontal distance between these two points (along the x-axis) is measured to calculate the stride length.

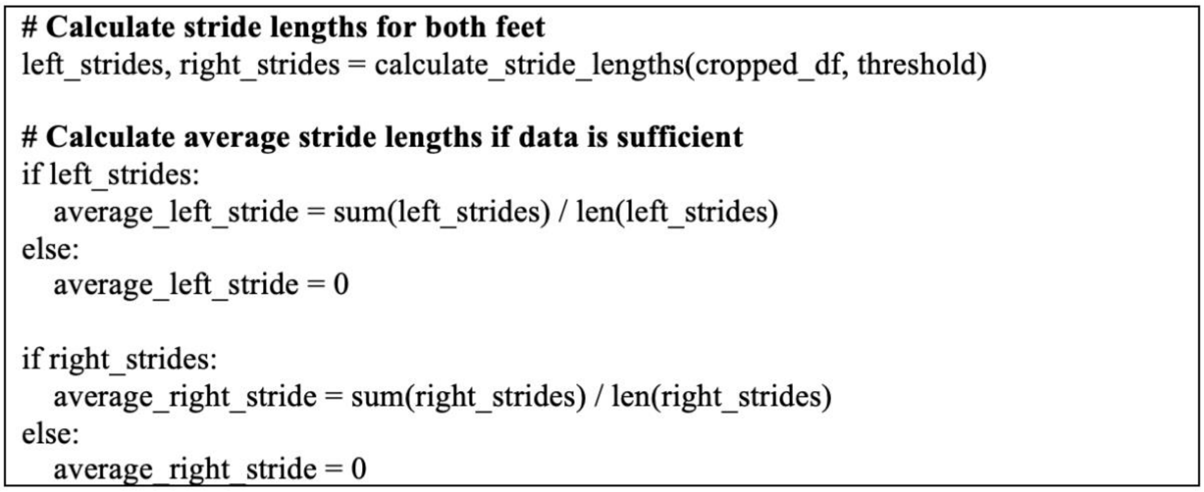
3. **Calculation of Stride Lengths:**

   Stride Measurement: The stride length for each valid event pair is calculated by taking the absolute difference between the x-coordinates of the toe-off and the subsequent foot strike positions. This provides the stride length in meters (or the units used for positional data).

   Collection of Stride Data: These individual stride lengths are collected into separate lists for both the left and right feet, allowing for analysis of stride patterns and differences between feet.

Calculation of Average Stride Lengths:

For a broader analysis that summarizes stride data over a session or compares across sessions, you can optionally calculate the average stride lengths for each foot. This average provides insights into the general efficiency and consistency of an athlete's gait over the analysed period. The Python code demonstrating this calculation is presented in ***Figure 21***.



```python
# Calculate stride lengths for both feet
left_strides, right_strides = calculate_stride_lengths(cropped_df, threshold)

# Calculate average stride lengths if data is sufficient
if left_strides:
   average_left_stride = sum(left_strides) / len(left_strides)
else:
   average_left_stride = 0

if right_strides:
   average_right_stride = sum(right_strides) / len(right_strides)
else:
   average_right_stride = 0
```

Figure 21 Python Code for the calculation of average strides

Insights from Stride Length Analysis

In this analysis, we explore the stride length metrics from two sprint sessions, focusing on Runner A and Runner B across distances of 30 meters and 50 meters. By comparing their stride lengths across these distances, we aim to evaluate the consistency of their running mechanics and observe variations in performance under differing conditions.

Comparative Performance Analysis Over 30 Meters:

Runner A completed the 30-meter distance in 3.05 seconds, taking 19 steps. His average stride length was 2.775 meters, with the average left stride measuring of 2.82 meters and the average right stride measuring of 2.73 meters. Runner B outperformed this by demonstrating a longer average stride length of 3.445 meters, with average left stride of 3.45 meters and average right stride of 3.44 meters, allowing him to finish faster in 2.7 seconds and with fewer steps, totalling only 17.

Observations from 30-Meter Performance

Runner B's increased stride length facilitated fewer steps to cover the same distance, exemplifying more efficient running mechanics. This efficiency not only reflects in his faster completion time but also indicates that effectively executed longer strides can significantly enhance running speed. Runner B's performance underscores the importance of optimizing stride length to improve overall sprinting efficiency and speed.

Comparative Performance Analysis Over 50 Meters:

Runner A took 31 steps to complete the 50-meter distance in 4.86 seconds, with an average stride length of 2.98 meters, breaking down to 3.00 meters for the average left stride and 2.97 meters for the average right stride. Conversely, Runner B demonstrated a more efficient stride, achieving a faster completion time of 4.46 seconds with an average stride length of 3.64 meters, detailed further as 3.635 meters for the average left stride and 3.646 meters for the average right, while requiring only 26 steps.

Observations from 50-Meter Performance:

The considerable difference in average stride lengths between Runner A and Runner B significantly influenced their race dynamics. Runner B's ability to maintain longer strides not only reduced the number of steps required but also contributed to a swifter overall performance, indicating an optimized stride mechanism. Moreover, Runner B's extended strides across the 50-meter distance highlight the critical role of stride length efficiency in enhancing both speed and endurance. This consistent capability to maintain elongated strides underscores potential long-term benefits, suggesting that enhancing stride length could lead to substantial improvements in overall racing performance.

Conclusion: Enhancing Predictive Modelling

The detailed analysis of stride length across various running conditions and distances offers critical insights that can significantly enhance the accuracy of predictive models in athletics. By integrating stride length data, these models gain a deeper, more nuanced understanding of the intricate ways in which stride dynamics influence overall performance times. This integration not only improves the precision of predictions but also helps in tailoring training programs to optimize athletes' performance based on their unique biomechanical characteristics.

5) Stride Length Symmetry Index

Objective:

The Stride Length Symmetry Index (SI) aims to quantify the degree of symmetry in an athlete’s stride lengths between the left and right legs. This metric is critical for assessing balance and uniformity in an athlete's gait, which are essential for optimal performance and injury prevention. Symmetrical stride lengths can indicate efficient biomechanics, whereas asymmetrical strides may signal potential biomechanical inefficiencies or underlying issues that could lead to injuries.

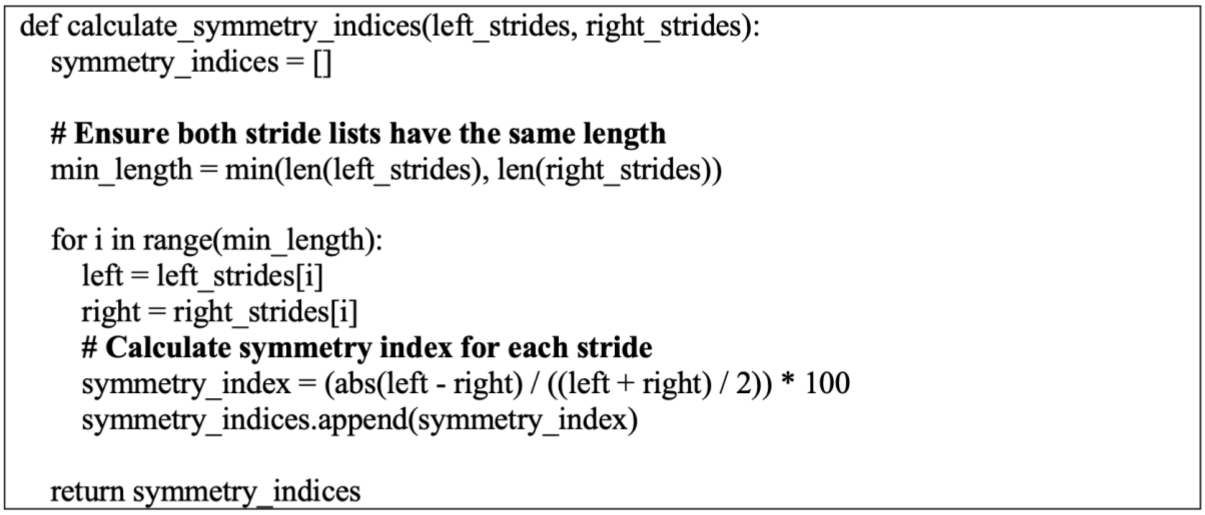
Methodology

Data Preparation:

The data from the motion capture system is processed to calculate the stride lengths for both the left and right legs, as previously described. This data is essential for computing the Symmetry Index.

Algorithm for Calculation:

The function **calculate_symmetry_indices** computes the Stride Length Symmetry Index for each pair of left and right stride lengths recorded during a session. This calculation is crucial for evaluating the balance and uniformity in an athlete's running mechanics. The detailed implementation of this function is illustrated in the Python code shown in ***Figure 22***.



```
def calculate_symmetry_indices(left_strides, right_strides):
    symmetry_indices = []

    # Ensure both stride lists have the same length
    min_length = min(len(left_strides), len(right_strides))

    for i in range(min_length):
        left = left_strides[i]
        right = right_strides[i]
        # Calculate symmetry index for each stride
        symmetry_index = (abs(left - right) / ((left + right) / 2)) * 100
        symmetry_indices.append(symmetry_index)

    return symmetry_indices
```

Figure 22 Python Code for the calculate_stride_lengths Function

Equalizing Data Lengths: The function begins by determining the shorter length between the two lists of stride lengths (**left_strides** and **right_strides**) to ensure that the comparison is only made where paired data exists.

Symmetry Index Calculation: For each pair of corresponding left and right strides, the function calculates the Symmetry Index using the formula:

$$SI = \frac{|\text{Left Stride Length} - \text{Right Stride Length}|}{\frac{\text{Left Stride Length} + \text{Right Stride Length}}{2}} * 100$$

This formula measures the percentage difference between the two stride lengths relative to their average, providing a clear quantification of symmetry.

Compiling Indices: All calculated Symmetry Indices are compiled into a list, which can later be analysed to assess the overall symmetry of the athlete's gait across the session.

Calculating Average Symmetry Index: To determine the overall symmetry for the session or across multiple sessions, the average Symmetry Index is calculated by summing all the indices and then dividing by the number of indices: $Average\ Symmetry\ Index = \frac{\sum(\text{symmetry indices})}{Count\ of\ Symmetry\ indices}$

This average provides a single percentage value that represents the typical symmetry level for the athlete's strides during the analysed period.

This added step in the analysis process allows for a more comprehensive understanding of an athlete's stride symmetry over time, giving insight into their biomechanical consistency and potential areas for improvement. By quantifying and averaging the Symmetry Index, you can effectively monitor changes in gait symmetry as a response to training interventions or due to other factors such as fatigue or injury.

Insights from Stride Length Symmetry Index Analysis

In this analysis, we focused on uncovering potential asymmetries in the runners' stride patterns. The goal was to understand how such asymmetries could affect overall running efficiency and the athletes' risk of injury. The Stride Length Symmetry Index (SI) was calculated for each runner across multiple sessions and distances to provide a comprehensive overview of their stride balance. Lower Symmetry Indices indicate a more balanced and efficient running gait, while higher indices may reveal underlying issues that need addressing through targeted training or biomechanical adjustments.

Key Findings:

The results from the Stride Length Symmetry Index (SI) calculations across all athletes and sessions consistently revealed low symmetry indices. This consistent finding suggests a high level of balance in the runners’ stride patterns, indicating efficient biomechanical alignment and movement. Well-balanced strides are crucial for maintaining high running efficiency, and the low symmetry indices observed in our athletes imply that their energy is being optimally utilized during each stride. This efficiency is essential not only for sprinting but also for distance running.

Additionally, a balanced stride symmetry is often associated with a lower risk of injuries [10][17]. Asymmetries in stride length can lead to compensatory movements that increase stress on specific muscles or joints, potentially resulting in overuse injuries. The absence of significant asymmetry in our athletes' stride patterns suggests a reduced likelihood of such issues. This supports their ability to train consistently and compete safely, underlining the importance of monitoring stride symmetry not only for performance enhancement but also for injury prevention.

Conclusion: Enhancing Predictive Modelling

The insights derived from analysing the Stride Length Symmetry Index are crucial for refining predictive models aimed at forecasting athletic performance, especially in terms of completion times. By integrating symmetry index data, these models can better account for biomechanical efficiency and potential asymmetries, which are key factors influencing an athlete's speed and endurance.

Incorporating these biomechanical variables into predictive models not only improves their accuracy but also makes them more reflective of real-world conditions. This enhanced understanding allows coaches and athletes to develop targeted training strategies that address any identified asymmetries proactively. Such strategic interventions optimize performance and reduce the risk of injury, ensuring athletes can maintain high levels of competitiveness and longevity in their sporting careers.

4.3.2 tsfresh Feature Extraction

In this subsection, we delve into the use of tsfresh for automated feature extraction from motion capture data of each athlete. tsfresh is a powerful Python library designed to automatically calculate a vast array of time series characteristics, or features, from a dataset. This tool is particularly useful for capturing the inherent complexities within time series data without manual intervention. This process aims to identify features that significantly correlate with the target variable, optimize the feature set for model training, and enhance the predictive accuracy.

Methodology

Data Preparation:

The data preparation stage is crucial for ensuring that the motion capture data is accurately processed for tsfresh feature extraction. This step involves loading and formatting each session's data into a pandas DataFrame, as described in the previous sections. Here’s a detailed breakdown:

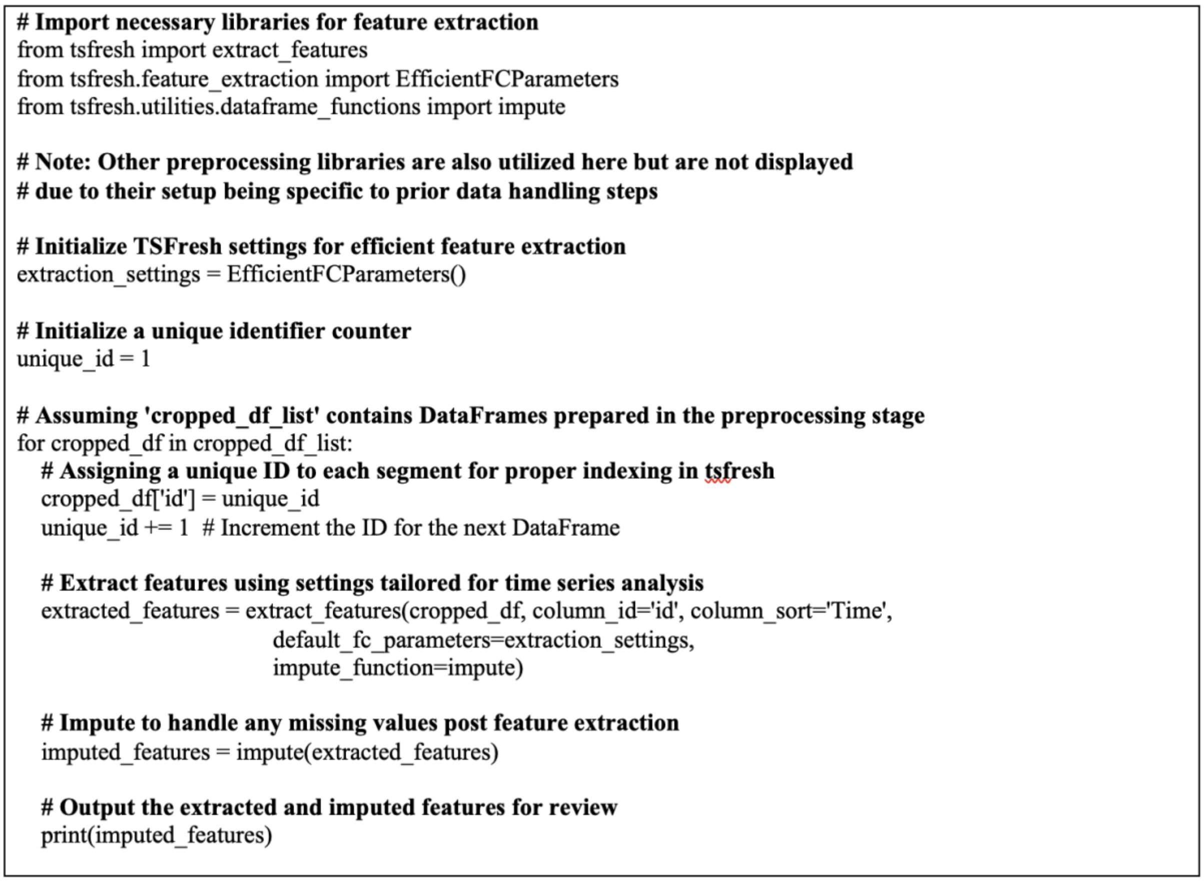
1. **Loading Data:**

Each motion capture file, typically stored in MVNX format, is loaded into the pandas DataFrame. These files contain detailed biomechanical data capturing each athlete's movements throughout various performance tests.
Exclusions: Specific files identified for exclusion based on predetermined criteria, such as errors in data capture or irrelevance to the current study, are not loaded into the DataFrame. This preventive measure ensures the quality and relevance of the data being analysed.

2. **Data Segmentation:** The main goal of data segmentation in this context is to isolate the segment of the data that represents the athlete's performance from the start to the completion of the 30-meter distance. This ensures that the analysis focuses specifically on the performance metrics relevant to this sprint distance.

Feature Extraction Process:

Using tsfresh [18], features are automatically extracted based on various time series characteristics. The process involves specifying parameters to efficiently capture relevant features from the dataset. The code block provided in ***Figure 23*** details how the extract_features function from tsfresh is used to extract a wide array of features based on the configured settings.



```
# Import necessary libraries for feature extraction
from tsfresh import extract_features
from tsfresh.feature_extraction import EfficientFCParameters
from tsfresh.utilities.dataframe_functions import impute

# Note: Other preprocessing libraries are also utilized here but are not displayed
# due to their setup being specific to prior data handling steps

# Initialize TSFresh settings for efficient feature extraction
extraction_settings = EfficientFCParameters()

# Initialize a unique identifier counter
unique_id = 1

# Assuming 'cropped_df_list' contains DataFrames prepared in the preprocessing stage
for cropped_df in cropped_df_list:
    # Assigning a unique ID to each segment for proper indexing in tsfresh
    cropped_df['id'] = unique_id
    unique_id += 1  # Increment the ID for the next DataFrame

    # Extract features using settings tailored for time series analysis
    extracted_features = extract_features(cropped_df, column_id='id', column_sort='Time',
                                          default_fc_parameters=extraction_settings,
                                          impute_function=impute)

    # Impute to handle any missing values post feature extraction
    imputed_features = impute(extracted_features)

    # Output the extracted and imputed features for review
    print(imputed_features)
```

Figure 23 Implementation of tsfresh's extract_features Function for Automated Feature Extraction

Choice of EfficientFCParameters for Feature Extraction:

In this study, the feature extraction settings are defined using **tsfresh.feature_extraction.settings.EfficientFCParameters()**. This configuration is selected to balance between extracting a comprehensive set of features and maintaining manageable computation times, which is crucial due to the extensive nature of the biomechanical data being processed [18].

The **EfficientFCParameters** function in TSFresh is designed to extract a robust set of features from time series data, similar to those obtained using the

ComprehensiveFCParameters. However, it omits features that are tagged with a “high computational cost” attribute. The rationale behind this selection is to reduce the runtime, especially important when processing large datasets like those in this study, where each athlete’s motion capture data involves numerous data points across multiple sessions.

Using **EfficientFCParameters** ensures that while essential and diverse features are extracted to capture the dynamics of the athletes’ performances adequately, the computational overhead is kept in check. This is particularly advantageous in scenarios where the data needs to be processed in a reasonable timeframe without sacrificing the depth of analysis required to make informed conclusions about athletic performance.

This approach aligns with the thesis’s need to efficiently handle extensive motion capture data, allowing for the extraction of meaningful features that can later be used to model and predict athletic performance outcomes effectively.

Saving Common Features:

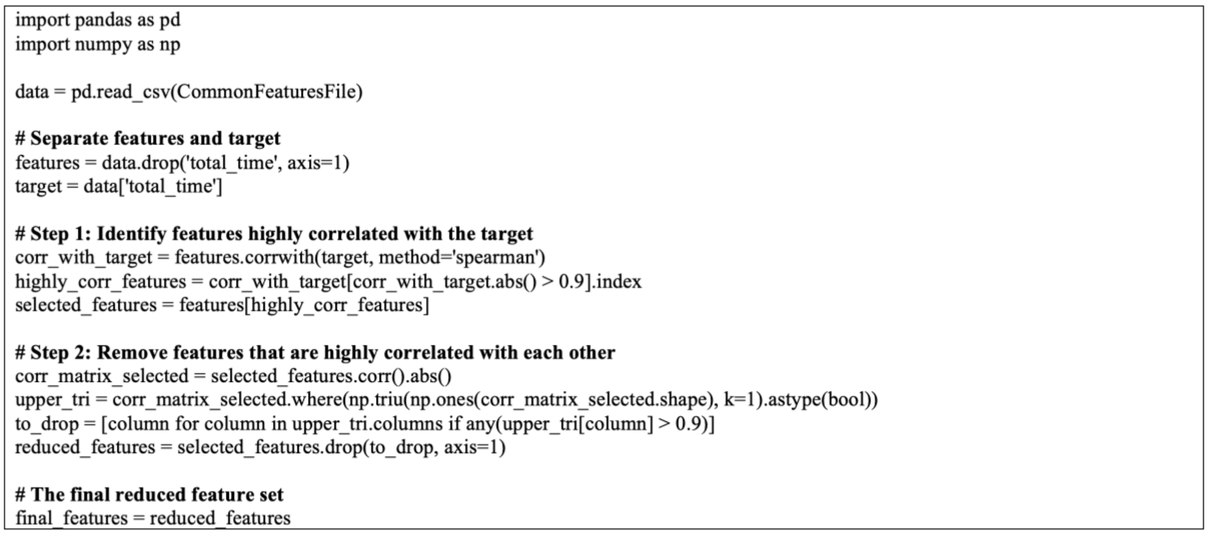
After the feature extraction process using tsfresh, we identified features that are common across different datasets. These common features, numbering 342,657, were then consolidated and saved into a CSV file. This critical step ensures that the subsequent analysis phases are based on features that are consistently present across all sessions, thereby maintaining the reliability and validity of the analyses. This CSV file serves as a foundational dataset for further detailed examination and modelling.

Final Feature Set Preparation

Objective:
Given the large number of common features (342,657) identified from the tsfresh extraction across multiple datasets, it was imperative to refine the feature set to enhance model performance and avoid overfitting. The goal was to retain features that have a strong correlation with the target variable ('total_time') and to eliminate features that are highly correlated with each other. This approach helps in minimizing redundancy and potential multicollinearity in the modelling phase.

Code Implementation:

The Python code shown in ***Figure 24*** demonstrates the steps taken to refine the feature set, ensuring that we only retain the most informative features for our predictive modelling. This code is responsible for the final feature set refinement, highlighting our methodical approach to handling a large number of features and ensuring that each feature contributes uniquely to our model's predictive capabilities.



```
import pandas as pd
import numpy as np

data = pd.read_csv(CommonFeaturesFile)

# Separate features and target
features = data.drop('total_time', axis=1)
target = data['total_time']

# Step 1: Identify features highly correlated with the target
corr_with_target = features.corrwith(target, method='spearman')
highly_corr_features = corr_with_target[corr_with_target.abs() > 0.9].index
selected_features = features[highly_corr_features]

# Step 2: Remove features that are highly correlated with each other
corr_matrix_selected = selected_features.corr().abs()
upper_tri = corr_matrix_selected.where(np.triu(np.ones(corr_matrix_selected.shape), k=1).astype(bool))
to_drop = [column for column in upper_tri.columns if any(upper_tri[column] > 0.9)]
reduced_features = selected_features.drop(to_drop, axis=1)

# The final reduced feature set
final_features = reduced_features
```

Figure 24 Python Code for Feature Set Refinement Process

Process Overview:

1. **Data Loading and Preparation:**

The data, including the extracted features, is loaded from a CSV file into a pandas DataFrame. This dataset includes all features deemed common across different motion capture sessions, ensuring a consistent basis for analysis.

2. **Feature Selection Based on Target Correlation:**

Correlation with the target variable ('total_time') is calculated using Spearman's rank correlation to identify features that have a strong influence on the target. A high correlation threshold (e.g., 0.9) ensures only the most impactful features are selected.

3. **Removal of Highly Inter-correlated Features:**

A correlation matrix of the selected features is created to identify and remove features that have high correlations with each other (threshold >= 0.9). This step helps to minimize multicollinearity, ensuring that the remaining features provide unique informational value to the models.

4. **Consolidation of the Final Feature Set:**

The remaining features after the above filtering steps form the final set used for modelling. This set is optimized for model performance, balancing the need for comprehensive data representation with the practical considerations of model complexity and overfitting. From the initial pool of 342,657 features, this rigorous process reduces the set to only 16 features, achieving a significant dimensionality reduction.

Conclusion: This structured approach to feature selection and refinement helps ensure that the final model is built on a robust, interpretable, and highly predictive set of features. By systematically reducing the feature set from an initial 342,657 features to just 16, we significantly enhance the model's manageability and focus on elements most critical to predicting the total time effectively. This process not only optimizes performance but also aids in the understanding and application of the model in practical scenarios, ensuring that the features used are the most impactful for predicting outcomes.

4.4 Training and Prediction

In this section, we explain the comprehensive methodology followed for the predictive modelling discussed in Chapter 5. Our approach involved leveraging both Ordinary Least Squares (OLS) regression and various machine learning models provided by the PyCaret library, applied to both manually and automatically extracted features.

Ordinary Least Squares (OLS) Regression

We employed the OLS model to predict race completion times using both manually and automatically extracted features. Through iterative modelling and strategic feature refinement, we identified key predictors of completion times. This process involved applying Sequential Feature Selection (SFS) to retain the most relevant features, experimenting with different training and testing dataset sizes to ensure robust model performance, and refining the feature set to enhance the model's predictive capabilities.

To ensure the reliability and generalizability of the OLS models, we employed 10-fold cross-validation throughout the modelling process. This rigorous validation method helped to prevent overfitting and provided a robust measure of model performance across different subsets of the dataset.

To evaluate the effectiveness of the OLS models using manually extracted features, we benchmarked their performance against a baseline model. The baseline model predicted performance times by simply using the average completion time derived from the training dataset. Comparing the OLS models to this baseline highlighted the improvements achieved through more sophisticated modelling techniques, with performance assessed using Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) metrics.

For the automatically extracted features, we compared the best model derived from manually extracted features with various OLS models trained using automatically extracted features, with performance assessed using MAE and RMSE. This comparison aimed to determine the superiority and accuracy of models based on different feature extraction methods.

The best OLS models for both manually and automatically extracted features were further analysed for the statistical significance and influence of each feature on the prediction of completion times. This detailed examination provided insights into how each feature contributed to the model's performance and identified the key predictors of completion times.

PyCaret for Multiple Model Training

Our application of PyCaret to the dataset was systematic and iterative, designed to maximize the efficiency and effectiveness of our experiments for both manually and automatically extracted features. Initially, we conducted model screening by training and evaluating various machine learning models available in PyCaret based on their Mean Absolute Error (MAE), a critical metric for our regression analysis. From this assessment, we identified the top three models that demonstrated consistent performance and superior average weighted MAE scores. In the next phase, we focused on hyperparameter tuning to refine the parameters of these top models. PyCaret's automated optimization tools were used to quickly adjust model parameters, complemented by GridSearchCV for conducting an exhaustive search for optimal parameter combinations, allowing for more granular control over the tuning process. After tuning, we compared the performance of the models to determine which configuration—either tuned by PyCaret or GridSearchCV—yielded the best results. This comparison was critical in selecting the final model, ensuring it met the analytical needs of our project.

Throughout the training and tuning phases, 10-fold cross-validation was employed to ensure the models' reliability and to prevent overfitting, thus enhancing their generalizability. The methodology culminated in the selection of the final model, which was then rigorously evaluated to confirm its predictive accuracy. The selected model was assessed to ensure it met the project's analytical needs, providing actionable insights for predicting race completion times.

Comparative Analysis of OLS and PyCaret Outcomes

Finally, we compared the results of the OLS models with the best-performing models from PyCaret. This comparative analysis focused on evaluating the effectiveness of each approach in terms of results, model complexity, training time, robustness, and generalization. By analysing the performance of both manually and automatically extracted features, we aimed to determine which modelling approach offers the best balance between accuracy, simplicity, and computational efficiency. This comparison provided valuable insights into the strengths and trade-offs of each method, guiding the selection of the most appropriate modelling technique for enhancing athletic performance prediction.

Chapter 5

Training and Prediction

5.1 Ordinary Least Squares (OLS) Regression

Introduction to OLS

Ordinary Least Squares (OLS) regression is a fundamental statistical method used extensively in predictive modelling and econometrics. It is specifically designed for estimating the relationships among variables in linear regression models. By fitting a linear equation to observed data, OLS minimizes the sum of the squared differences between the observed responses in the dataset and those predicted by the linear approximation.

Advantages of OLS

The primary advantage of OLS is its simplicity and interpretability. It provides clear, quantifiable insights into how predictor variables affect the response variable, making it invaluable for understanding and predicting outcomes. In the context of this study, Ordinary Least Squares (OLS) regression helps elucidate the relationships between various race features such as (stride length, air time, ground contact time etc.) and the athletes' performance times. This approach enables a detailed understanding of how different aspects of an athlete’s performance correlate with their speed and efficiency, providing actionable insights that can directly influence training methodologies and competitive strategies.

Baseline Comparison

To assess the effectiveness of the OLS models, their performance is benchmarked against a baseline model. The baseline model predicts performance times by simply using the average completion time derived from the training dataset. This comparison not only highlights the superior predictive capability of the OLS models over simpler statistical methods but also establishes a standard for evaluating the incremental value brought by employing more sophisticated features and modelling techniques.

By employing OLS regression, this study aims to provide a robust statistical framework for predicting athletic performance, thereby offering insights that are critical for athletes and coaches to optimize training and competition strategies.

Model Implementation:

1) **Manual Extracted Features Implementation**:

Data Preparation

The manually extracted features were carefully prepared to ensure the integrity of the dataset for modelling. The selected features for the analysis were:

- **Total Steps**
- **Average Left and Right Foot Ground Contact Time (ms)**
- **Average Overall Ground Contact Time (ms)**
- **Average Overall Air Time (ms)**
- **Stride Length Symmetry Index**
- **Adjusted Stride Length Symmetry Index**
- **Average Stride Length (Left, Right, and Combined)**
- **Gender**

These features were derived from detailed analysis in the previous sections, focusing on capturing essential aspects of running mechanics that influence performance.

Models Setup and Evaluation

i. Model 1:

Selected Features

For this model, all manually extracted features previously detailed were utilized:

- **Total Steps**
- **Average Left and Right Foot Ground Contact Time (ms)**
- **Average Overall Ground Contact Time (ms)**

- **Average Overall Air Time (ms)**
- **Stride Length Symmetry Index**
- **Adjusted Stride Length Symmetry Index**
- **Average Stride Length (Left, Right, and Combined)**

Data Splitting

The dataset was divided into a training set (70% of the data) and a test set (30% of the data). This split was chosen to ensure that the model could be trained on a comprehensive sample of the data while still retaining a substantial portion for unbiased evaluation.

ii. Model 2:

Selected Features

Model 2 applied Sequential Forward Selection (SFS) using the Ordinary Least Squares (OLS) as the estimator. The goal was to identify an optimal subset of features that enhance predictive accuracy while minimizing redundancy. The SFS identified the following features as most effective for predicting race completion times:

- **Total Steps**
- **Average Overall Ground Contact Time (ms)**
- **Average Overall Air Time (ms)**
- **Stride Length Symmetry Index**

Data Splitting

As with Model 1, the data was split into 70% for training and 30% for testing, maintaining consistency in evaluation methodology to ensure comparability across models.

iii. Model 3:

Selected Features:

For Model 3, I expanded the feature set from Model 2 by incorporating the gender of each athlete, hypothesizing that this demographic detail might have a significant impact on the model's accuracy due to physiological differences that could affect race dynamics.

Data Splitting:

As with Model 1 and 2, the data was split into 70% for training and 30% for testing, maintaining consistency in evaluation methodology to ensure comparability across models.

iv. Model 4:

Selected Features:

The best-performing model utilized an expanded set of features based on comprehensive analysis from previous models:

- **Total Steps**
- **Stride Length Symmetry Index**
- **Average Stride Length**
- **Average Overall Ground Contact Time (ms)**
- **Average Overall Air Time (ms)**
- **Gender**

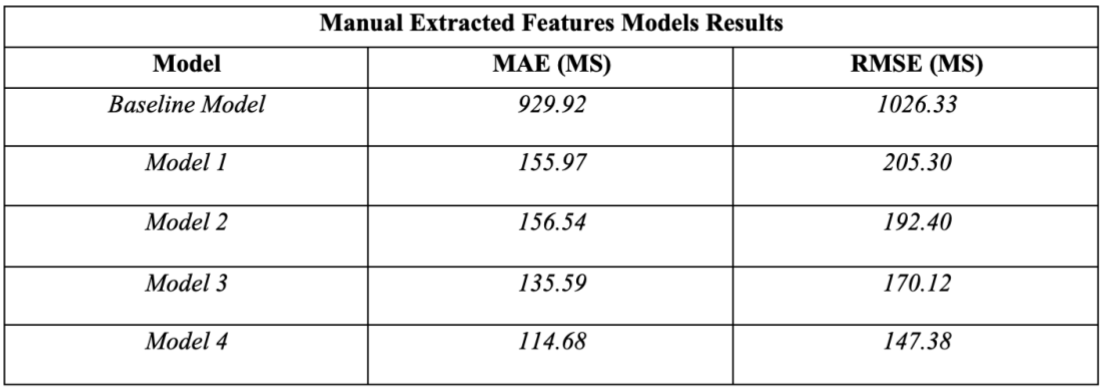
These features were chosen for their strong correlation with the target variable and their potential to provide a nuanced understanding of the dynamics influencing race performance.

Data Splitting:

For this model, the data was split into 80% training and 20% testing. This adjustment in the training-test ratio was made to maximize the training data available for the model, enhancing its ability to learn and generalize from a larger dataset. Increasing the training proportion can often lead to better model performance, especially when the feature space is complex and varied, as it provides a richer set of examples from which the model can learn.

Results and Interpretation

The predictive models developed for forecasting completion times demonstrated significant improvements through incremental refinements, as detailed in the accompanying table. Each model's performance offers valuable insights into the interaction between feature selection and predictive accuracy, guiding future enhancements and research directions.



Manual Extracted Features Models Results		
Model	**MAE (MS)**	**RMSE (MS)**
Baseline Model	*929.92*	*1026.33*
Model 1	*155.97*	*205.30*
Model 2	*156.54*	*192.40*
Model 3	*135.59*	*170.12*
Model 4	*114.68*	*147.38*

Figure 25 Results of Manual Extracted Features OLS Models

Model 1 marked a substantial advance over the baseline, reducing the Mean Absolute Error (MAE) from 929.92 milliseconds (ms) to 155.97 ms, and the Root Mean Square Error (RMSE) from 1026.33 ms to 205.30 ms. This improvement highlights the value of the manually extracted biomechanical features, confirming their critical role in predictive modelling. The reduction in error metrics emphasizes the features' effectiveness, though the remaining errors indicate potential for further optimization through advanced feature engineering or alternative regression methodologies.

Model 2 utilized a refined set of features determined through Sequential Feature Selection (SFS), achieving notable performance gains over the baseline, albeit with slight increases in error metrics compared to Model 1. This variation underscores the SFS's efficiency in identifying essential features but suggests that some beneficial predictors might have been excluded. The results illustrate the delicate balance required in feature selection to optimize both model accuracy and interpretability.

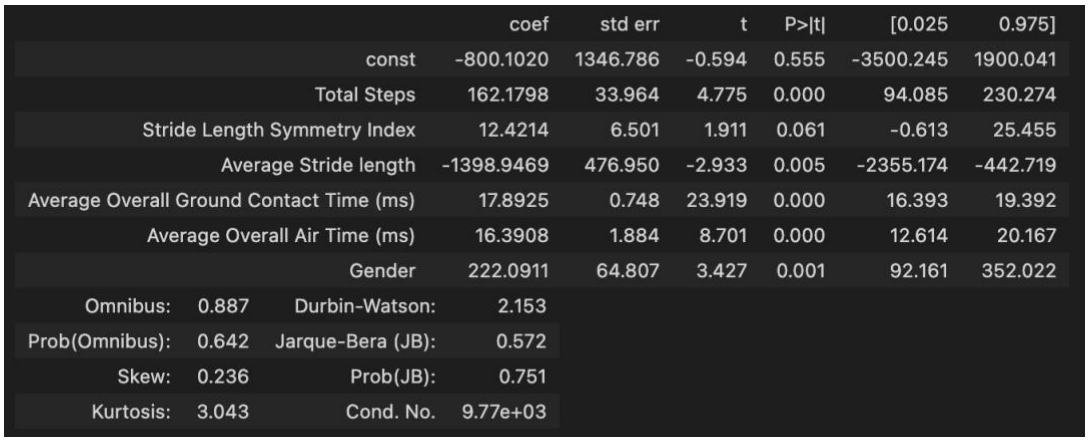
Model 3 incorporated gender as an additional feature, resulting in further improvements with an MAE of 135.59 ms and an RMSE of 170.12 ms. The inclusion of gender not only reduced errors but also added a layer of depth to the analysis, reflecting physiological and biomechanical differences that impact race outcomes. This underscores the importance of demographic variables in enhancing the model's explanatory power and predictive accuracy.

Model 4 achieved the best performance with the lowest MAE and RMSE of 114.68 ms and 147.38 ms, respectively. By strategically selecting features and using a larger training dataset, this model showcased the profound impact of careful data management and methodological precision on predictive outcomes. The success of **Model 4** effectively demonstrates how careful feature selection, alongside optimal data splitting, can profoundly impact the predictive capabilities of regression models in sports analytics. The results not only validate the chosen methodology but also highlight the critical role of demographic factors like gender in predicting athletic performance.

Collectively, these results underscore the progressive enhancement of predictive accuracy through methodical feature selection and the thoughtful integration of demographic variables. Each step forward provides critical insights into the dynamics influencing race performance, setting a foundation for further investigations aimed at refining predictive models in sports analytics.

Detailed Analysis of the Best Model

Below, ***Figure 26*** displays the summary of the best-performing model derived from our analysis. This summary encapsulates the statistical significance and influence of each feature on the prediction of completion times.



	coef	std err	t	P>|t|	[0.025	0.975]
const	-800.1020	1346.786	-0.594	0.555	-3500.245	1900.041
Total Steps	162.1798	33.964	4.775	0.000	94.085	230.274
Stride Length Symmetry Index	12.4214	6.501	1.911	0.061	-0.613	25.455
Average Stride length	-1398.9469	476.950	-2.933	0.005	-2355.174	-442.719
Average Overall Ground Contact Time (ms)	17.8925	0.748	23.919	0.000	16.393	19.392
Average Overall Air Time (ms)	16.3908	1.884	8.701	0.000	12.614	20.167
Gender	222.0911	64.807	3.427	0.001	92.161	352.022

Omnibus:	0.887	Durbin-Watson:	2.153
Prob(Omnibus):	0.642	Jarque-Bera (JB):	0.572
Skew:	0.236	Prob(JB):	0.751
Kurtosis:	3.043	Cond. No.	9.77e+03

Figure 26 Best Model Summary

The table summarizes the regression output, detailing each predictor's coefficients, standard errors, t-values, P-values, and confidence intervals. This section interprets these statistical metrics to illuminate their significance in influencing race performance. By understanding the quantitative impact of each feature, we can better appreciate how variations in athlete performance are explained through our model.

Total Steps: The coefficient for total steps is 162.1798, with a standard error of 33.964, a t-value of 4.775, and a p-value of less than 0.001. The confidence interval ranges from 94.085 to 230.274. This feature is statistically significant, indicating a strong positive relationship between the total number of steps and race completion time. Each additional step is associated with an increase in race time, emphasizing the importance of stride efficiency for better performance.

Stride Length Symmetry Index: The coefficient for the stride length symmetry index is 12.4214, with a standard error of 6.501, a t-value of 1.911, and a p-value of 0.061.

The confidence interval ranges from -0.613 to 25.455. While this coefficient approaches significance, it suggests that higher asymmetry might increase race time, although this effect is not statistically robust at the usual levels of significance.

Average Stride Length: The coefficient for average stride length is -1398.9469, with a standard error of 476.950, a t-value of -2.933, and a p-value of 0.005. The confidence interval ranges from -2355.174 to -442.719. This significant negative coefficient indicates that longer strides are effectively associated with shorter completion times, reinforcing the importance of stride efficiency.

Average Overall Ground Contact Time (ms): The coefficient for average overall ground contact time is 17.8925, with a standard error of 0.748, a t-value of 23.919, and a p-value of less than 0.001. The confidence interval ranges from 16.393 to 19.392. This positive and highly significant relationship suggests that longer contact times are correlated with increased completion times, highlighting the performance cost of extended ground contact.

Average Overall Air Time (ms): The coefficient for average overall air time is 16.3908, with a standard error of 1.884, a t-value of 8.701, and a p-value of less than 0.001. The confidence interval ranges from 12.614 to 20.167. Similar to ground contact time, longer air times are significantly associated with increased completion times, reflecting the biomechanical trade-offs in running techniques.

Gender: The coefficient for gender is 222.0911, with a standard error of 64.807, a t-value of 3.427, and a p-value of 0.001. The confidence interval ranges from 92.161 to 352.022. Gender shows a significant effect on race times, indicating potential physiological and biomechanical differences affecting performance.

Model Insights and Practical Implications:

Efficiency in Stride: The data highlights the critical role of stride length and its symmetry in running efficiency. Properly balanced strides ensure more effective propulsion and less energy wastage, which directly translates into improved performance times. Athletes and coaches should consider drills and techniques that enhance stride length and promote symmetry across both feet.

Ground and Air Time Optimization: Our findings indicate that minimizing ground contact time without compromising the air time essential for forward propulsion can lead to better race results. Focused training on quick toe-offs and efficient mid-air motion can help athletes achieve a balance that optimizes speed.

Data-Driven Training: This model provides valuable insights that can inform targeted interventions in an athlete's training regimen. By understanding the specific variables that most significantly impact performance, training can be tailored to address these areas, resulting in more focused improvements. Regularly revisiting this data-driven approach, as athletes evolve, ensures that training remains aligned with optimal performance goals.

2) **tsfresh Extracted Features**:

Data Preparation

After segmenting the motion capture data as previously outlined, the tsfresh library was used to automatically extract features from each athlete's performance data. This process leverages complex algorithms designed to explore and construct a comprehensive set of time-series features from the biomechanical data collected.

Feature Extraction and Selection

From the extensive range of features generated by tsfresh, we initially identified 342,657 potential features. Through a systematic process of identifying common features present in all datasets, we narrowed this list down significantly to ensure consistency and manageability.

Final Feature Set

The final set of features used for modelling includes 16 specific attributes that demonstrated the highest correlation with our target variable, completion time, and minimal inter-feature correlation. This diverse range of data points includes specific body part positions and movements, along with more complex statistical transformations, categorized as follows:

- **Frequency Domain Features**: Fourier coefficients help capture repetitive motion patterns, which are essential for identifying consistent performance attributes in athletes. For example:
    1) **Right Upper Leg Position X (FFT Coefficient Absolute, Coefficient 82)**
    2) **L5 Orientation Q3 (FFT Coefficient Angle, Coefficient 99)**
    3) **Head Position Z (FFT Coefficient Angle, Coefficient 92)**

- **Statistical Features**: These features assess the time series data's underlying properties, such as stationarity and trends, which are important for understanding movement consistency and predicting performance.
    4) **Left Upper Leg Position X (Augmented Dickey-Fuller, Used Lag, Auto Lag: AIC)**
    5) **Head Velocity Y (Augmented Dickey-Fuller, Used Lag, Auto Lag: AIC)**
    6) **T8 Acceleration Z (Augmented Dickey-Fuller, Used Lag, Auto Lag: AIC)**
    7) **Left Lower Leg Orientation Q3 (Augmented Dickey-Fuller, Used Lag, Auto Lag: AIC)**

- **Peaks and Other Descriptive Statistics**: These features highlight critical points and descriptive metrics in the sensor data, offering insights into the dynamics of motion.
    8) **Neck Acceleration Z (Number of Peaks)**
    9) **Right Shoulder Position X (Last Location of Minimum)**
    10) **Right Forearm Velocity Y (Number of Peaks)**
    11) **Left Lower Leg Position Y (Longest Strike Above Mean)**

- **Dynamic Time-Warping and Complexity Measures**: These complex metrics provide insights into the temporal dynamics and structural complexity of the motion capture data.
    12) **Left Toe Position Y (Lempel-Ziv Complexity, Bins: 10)**

- **Trend and Variation Analysis**: Analysing trends and variations in sensor data helps identify patterns related to athletic performance and potential biomechanical issues.
    13) **Right Hand Angular Velocity Y (Aggregated Linear Trend, Standard Error, Chunk Length: 10, Aggregate Function: Maximum)**
    14) **Right Upper Leg Acceleration X (Aggregated Linear Trend, Standard Error, Chunk Length: 5, Aggregate Function: Mean)**

- **Change Quantiles Analysis**: This type of analysis measures changes within specified quantile ranges of the data. It helps identify variability within certain segments of the distribution, which can be crucial for understanding how movements vary under different conditions or phases of activity.
    15) **Right Toe Velocity X (Change in Quantiles, Function: Variance, Non-Absolute Values, Quantile Range: 0.0 to 0.6)**

- **Count Features**: Count features quantify the frequency of certain conditions being met within the dataset. These are useful for identifying recurring patterns or events.

    16) T8 Orientation Q4 (Count Above Mean)

Models Setup and Evaluation

i. Model 1

Selected Features:

Utilized all 16 tsfresh extracted features, including Fourier coefficients, augmented dickey-fuller attributes, change quantiles, number of peaks, and various other dynamic and statistical measures from the sensor data.

Data Splitting:

For this model, the data was split into 80% training and 20% testing. This adjustment in the training-test ratio was made to maximize the training data available for the model, enhancing its ability to learn and generalize from a larger dataset. Increasing the training proportion can often lead to better model performance, especially when the feature space is complex and varied, as it provides a richer set of examples from which the model can learn.

ii. Model 2

Selected Features:

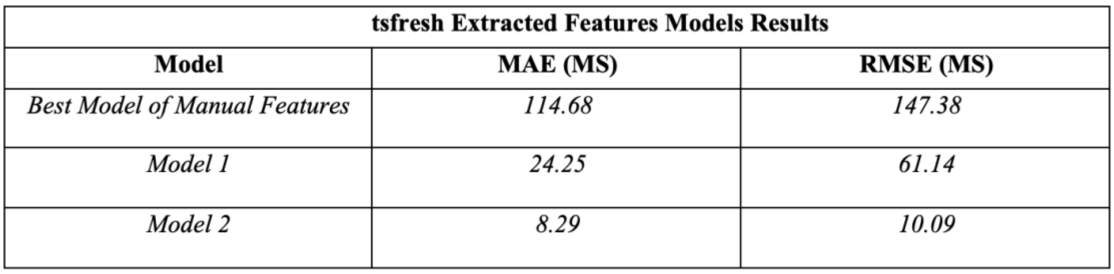
Model 2 utilizes a refined subset of features identified through Sequential Feature Selection (SFS) with the Ordinary Least Squares (OLS) as the estimator. The selected features are particularly potent in capturing essential aspects of the dynamics of race, emphasizing the potential of combining frequency domain features, count features, and statistical metrics:

- **Right Upper Leg Position X (FFT Coefficient Absolute, Coefficient 82)**
- **L5 Orientation Q3 (FFT Coefficient Angle, Coefficient 99)**
- **T8 Orientation Q4 (Count Above Mean)**
- **Neck Acceleration Z (Number of Peaks)**
- **Right Shoulder Position X (Last Location of Minimum)**

- **Right Hand Angular Velocity Y (Aggregated Linear Trend, Standard Error, Chunk Length 10, Aggregate Function: Max)**
- **Left Lower Leg Orientation Q3 (Augmented Dickey-Fuller, Used Lag, Autolag: AIC)**
- **Left Lower Leg Position Y (Longest Strike Above Mean)**
- **T8 Acceleration Z (Augmented Dickey-Fuller, Used Lag, Autolag: AIC)**

Results and Interpretation

The development and refinement of predictive models using tsfresh extracted features demonstrated significant advancements in forecasting race completion times, as depicted in the accompanying table. The tsfresh library facilitated an extensive and nuanced feature extraction process, ultimately contributing to substantial improvements in model performance.



tsfresh Extracted Features Models Results		
Model	**MAE (MS)**	**RMSE (MS)**
Best Model of Manual Features	*114.68*	*147.38*
Model 1	*24.25*	*61.14*
Model 2	*8.29*	*10.09*

Figure 26 Results of tsfresh Extracted Features OLS Models

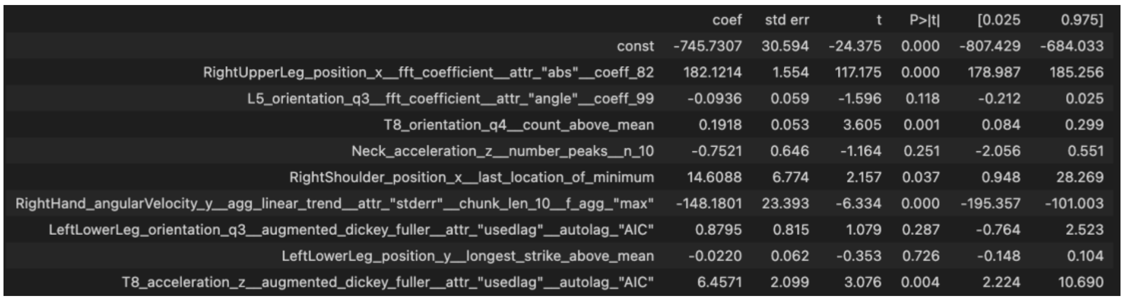
Model 1 leveraged all 16 tsfresh extracted features, which include complex attributes like Fourier coefficients, augmented Dickey-Fuller attributes, and dynamic time-warping measures. This model achieved a Mean Absolute Error (MAE) of 24.25 milliseconds (ms) and a Root Mean Square Error (RMSE) of 61.14 ms, which significantly outperformed the best model using manually extracted features that had an MAE of 114.68 ms and an RMSE of 147.38 ms. The substantial reduction in error metrics with Model 1 illustrates the robustness and efficacy of automated feature extraction techniques. These techniques excel at capturing intricate patterns and dynamics of biomechanical data, which are often challenging to identify through manual processes.

Model 2 refined the feature set further through Sequential Feature Selection (SFS) using Ordinary Least Squares (OLS) as the estimator. This model, focusing on a tailored subset of powerful features, dramatically improved the prediction accuracy, achieving an MAE of 8.29 ms and an RMSE of 10.09 ms. This represents an unprecedented reduction in error metrics, highlighting the critical importance of targeted feature selection. The focused approach allowed Model 2 to not only enhance predictive accuracy but also to increase the interpretability of the model. By concentrating on fewer, more impactful features, this model provided deeper insights into the specific biomechanical and physiological variables that most directly affect race performance. Moreover, the simplification of the feature set enhances the practical applicability of the model's findings, potentially influencing training and performance optimization strategies.

These results validate the powerful capability of tsfresh for automatic feature extraction in sports analytics. They demonstrate how strategic feature selection and advanced modelling techniques can synergistically improve the precision of predictions while also providing actionable insights into the underlying dynamics of athletic performance. The progression from a broad array of automatically extracted features to a meticulously curated subset exemplifies a successful approach in leveraging complex data for practical outcomes.

Detailed Analysis of the Best Model

Below, ***Figure 27*** displays the summary of the best-performing model derived from our analysis. This summary encapsulates the statistical significance and influence of each feature on the prediction of race completion times



	coef	std err	t	P>|t|	[0.025	0.975]
const	-745.7307	30.594	-24.375	0.000	-807.429	-684.033
RightUpperLeg_position_x__fft_coefficient__attr_"abs"__coeff_82	182.1214	1.554	117.175	0.000	178.987	185.256
L5_orientation_q3__fft_coefficient__attr_"angle"__coeff_99	-0.0936	0.059	-1.596	0.118	-0.212	0.025
T8_orientation_q4__count_above_mean	0.1918	0.053	3.605	0.001	0.084	0.299
Neck_acceleration_z__number_peaks__n_10	-0.7521	0.646	-1.164	0.251	-2.056	0.551
RightShoulder_position_x__last_location_of_minimum	14.6088	6.774	2.157	0.037	0.948	28.269
RightHand_angularVelocity_y__agg_linear_trend__attr_"stderr"__chunk_len_10__f_agg_"max"	-148.1801	23.393	-6.334	0.000	-195.357	-101.003
LeftLowerLeg_orientation_q3__augmented_dickey_fuller__attr_"usedlag"__autolag_"AIC"	0.8795	0.815	1.079	0.287	-0.764	2.523
LeftLowerLeg_position_y__longest_strike_above_mean	-0.0220	0.062	-0.353	0.726	-0.148	0.104
T8_acceleration_z__augmented_dickey_fuller__attr_"usedlag"__autolag_"AIC"	6.4571	2.099	3.076	0.004	2.224	10.690

Figure 27 Best Model Summary

This section delves into the coefficients that are statistically significant, providing insights into their impact on race performance. The table includes each predictor's coefficients, standard errors, t-values, P-values, and confidence intervals, helping us to understand the quantitative effects of these variables on race times.

Statistically Significant Model Coefficients:

The **Right Upper Leg Position X (FFT Coefficient Abs Coeff 82)** has a coefficient of 182.1214, a standard error of 1.554, a t-value of 117.175, and a p-value of less than

0.001. The confidence interval ranges from 178.987 to 185.256. This feature shows a significant positive relationship with completion times, suggesting that increased movement or variation in the right upper leg's position correlates with longer completion times. This underscores the need for controlled and efficient leg movements during the run.

The **T8 Acceleration Z (Augmented Dickey-Fuller Usedlag AIC)** has a coefficient of 6.4571, a standard error of 2.099, a t-value of 3.076, and a p-value of 0.004. The confidence interval ranges from 2.224 to 10.690. This feature reflects the importance of forward and backward acceleration at the T8 segment, likely related to running posture and vertical oscillations. Effective management of these accelerations can improve overall completion times by ensuring core stability and efficient energy utilization, which are vital for competitive running.

The **Right Hand Angular Velocity Y (Aggregated Linear Trend, Stderr, Max)** has a coefficient of -148.1801, a standard error of 23.393, a t-value of -6.334, and a p-value of less than 0.001. The confidence interval ranges from -195.357 to -101.003. This negative coefficient demonstrates that higher angular velocities of the right hand are associated with shorter completion times. Efficient arm swing dynamics contribute positively to propulsion and momentum, highlighting the necessity of proper synchronization of arm movements with leg strides to maximize speed and efficiency.

The **Right Shoulder Position X (Last Location of Minimum)** has a coefficient of 14.6088, a standard error of 6.774, a t-value of 2.157, and a p-value of 0.037, with a confidence interval ranging from 0.948 to 28.269. This indicates that the position of the right shoulder at the last minimum has a notable impact on completion times. Such positions, likely occurring at critical moments during the race, can significantly influence the athlete's performance.

Model Insights and Practical Implications

The analysis of the best-performing model using tsfresh extracted features offers valuable insights into the biomechanical factors impacting race performance, and here are the practical implications derived from those insights:

Upper Body Mechanics: The model highlights the significant role of the right upper leg's position and the right shoulder's movements. This emphasizes the need for balanced strength and coordination between the upper and lower body. Athletes should focus on enhancing upper body mechanics through drills that improve the strength and mobility of shoulders and upper legs, like plyometrics and targeted resistance training, which can directly improve running efficiency and race times.

Arm Dynamics: The negative relationship between the right hand's angular velocity and race times suggests that efficient arm dynamics are crucial. Proper arm swings synchronized with leg movements can increase propulsion and overall speed. Athletes should practice drills that refine arm swing techniques to ensure they contribute effectively to forward momentum.

Vertical Motion Management: The association of T8 spinal acceleration with slower race times underlines the importance of minimizing vertical motion to conserve energy. Training to improve core stability and strength can help athletes maintain a stable torso and reduce vertical oscillations, leading to more efficient energy use during races.

Data-Driven Training Adjustments: Insights from this model enable targeted, data-driven adjustments in training. Understanding specific biomechanical influences allows for personalized training approaches, focusing on the individual needs of each athlete. Regular use of sensor data during training can aid in real-time adjustments and track progress effectively.

Holistic Training Approach: The interaction of different biomechanical factors suggests that a holistic approach to training is essential. A comprehensive program that addresses stride mechanics, upper body strength, core stability, and arm dynamics ensures a well-rounded development and enhances overall athletic performance.

Conclusion

This part of the thesis has provided significant insights into athletic performance by exploring both manually and automatically extracted features through Ordinary Least Squares (OLS) regression. Our progression from initial models using manually extracted features to more advanced models employing tsfresh extracted features demonstrates a clear enhancement in both predictive accuracy and interpretative power.

Through iterative modelling and strategic feature refinement, we have identified key predictors of completion times. This journey from a broad collection of manually analysed features to a refined set of features extracted by sophisticated automated methods illustrates the value of blending domain knowledge with advanced data analytics to enhance predictive models. These models not only improve upon baseline predictions but also offer actionable insights that are directly applicable to training and performance optimization.

The standout performance of the models using tsfresh extracted features highlights the considerable benefits of automated feature extraction within the realm of sports analytics. By harnessing complex algorithms to pinpoint critical predictors from extensive data sets, this thesis has set new benchmarks in the field. This approach enriches our understanding of crucial performance influences and opens up further avenues for research and practical application in sports science.

As we continue to delve deeper into the potential of data analytics in sports, the subsequent sections will explore the use of the PyCaret tool, expanding on the foundation laid here to further refine our methods and insights. This transition ensures a seamless progression towards even more sophisticated modelling techniques, continuing to support athletes and coaches in their pursuit of peak performance.

5.2 PyCaret for Multiple Model Training

Introduction to PyCaret

PyCaret is an open-source, low-code machine learning library in Python that aims to reduce the hypothesis to insights cycle time in a machine learning experiment. It enables data scientists and analytics professionals to perform end-to-end experiments quickly and efficiently. PyCaret automates machine learning workflows, enabling fast experimentation and deployment of models.

Advantages of PyCaret

- **Efficiency**: Automates many of the repetitive tasks involved in a typical machine learning workflow, allowing for more efficient model development.
- **Accessibility**: Offers a simple and easy-to-use interface that helps new practitioners adopt machine learning techniques without deep programming expertise.
- **Flexibility**: Supports multiple algorithms and models, providing a broad toolkit for testing various approaches to find the best solution.
- **Deployment Ready**: Includes modules for model deployment and monitoring, making it easier to transition from a development environment to a production setting.

Methodology for PyCaret Model Training and Selection

In applying PyCaret to our dataset, our approach was systematic and iterative, leveraging the library's extensive capabilities to maximize the efficiency of our experiments. We began with an initial model screening where various machine learning models provided by PyCaret were trained and evaluated based on their Mean Absolute Error (MAE), a critical metric for our regression analysis. From this initial assessment, we identified the top three models, selected for their performance consistency and superior average weighted MAE scores. The next phase involved hyperparameter tuning using PyCaret’s automated optimization tools to quickly refine model parameters, complemented by an exhaustive search with GridSearchCV for more granular control over the tuning process. After optimizing the parameters, we compared the performance of the models to determine which configuration—either tuned by PyCaret or GridSearchCV—yielded the best results. Throughout the training and tuning phases, we employed 10-fold cross-validation to ensure the models' reliability and to prevent overfitting, thus enhancing their generalizability. The methodology culminated in selecting the final model, which was then rigorously evaluated to confirm its predictive accuracy and ensure it met the analytical needs of our project. This comprehensive process allowed us to systematically identify and refine the best models for predicting race completion times, harnessing the full potential of PyCaret's features in our data-driven analysis.

1) **Manual Extracted Features Implementation**

Data Preparation and Feature Selection

In this phase of the analysis, we utilized manually extracted features that were identified as significant in the earlier sections of the study. These features were chosen based on their demonstrated impact on race performance metrics and included:

- Total Steps
- Average Left and Right Foot Ground Contact Time (ms)
- Average Overall Ground Contact Time (ms)
- Average Overall Air Time (ms)
- Stride Length Symmetry Index
- Average Stride Length (Left, Right, and Combined)
- Gender

Data Splitting

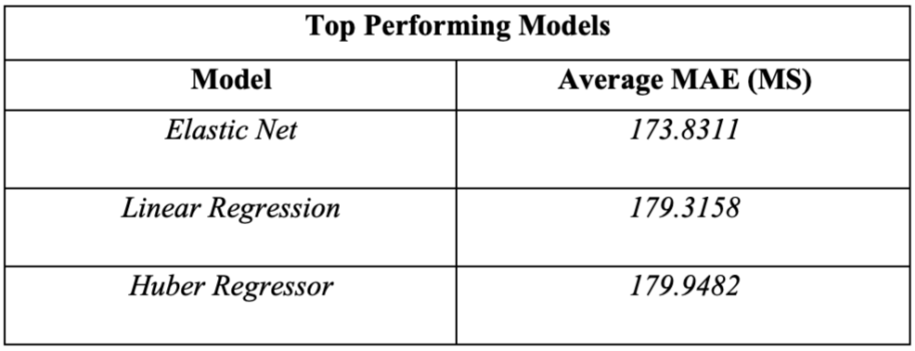
For the multiple models trained using PyCaret, the dataset was partitioned into 80% for training and 20% for testing. This 80/20 split is widely utilized in machine learning to ensure that each model has a substantial amount of data to learn from while also reserving a significant portion for unbiased evaluation of their performance. This standard practice helps maintain a balance between effectively learning underlying patterns and avoiding overfitting across various models.

Model Training with PyCaret

We employed PyCaret, an open-source machine learning library, to train multiple regression models using the manually extracted features. The process involved:

10-Fold Cross-Validation: This technique was used to validate the models' performance, ensuring robustness and the ability to generalize across different datasets.

Model Selection: PyCaret's comparative model analysis helped identify the top-performing models based on their Average Mean Absolute Error (MAE). The initial selection of the top three models is illustrated in ***Figure 28***.



Top Performing Models	
Model	**Average MAE (MS)**
Elastic Net	*173.8311*
Linear Regression	*179.3158*
Huber Regressor	*179.9482*

Figure 28 Best Performing Models Based on Average Mean Absolute Error

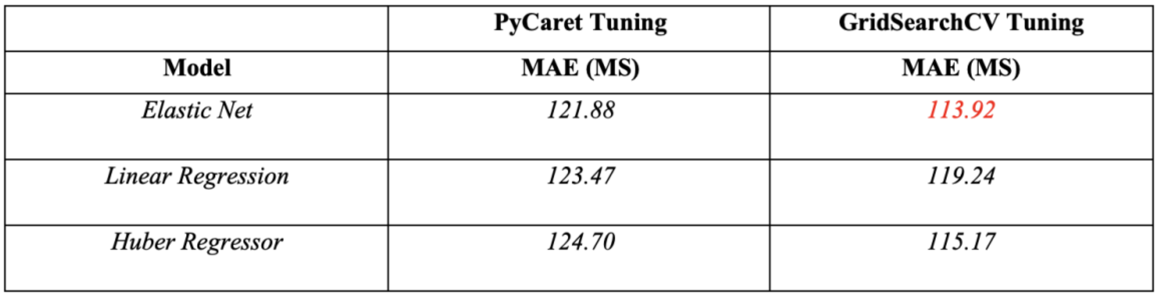
This figure highlights the initial results from the comparative analysis of regression models trained using PyCaret. The models are ranked based on their Average MAE, with the Elastic Net model showing the best performance at 173.8311 ms, followed closely by the Linear Regression and Huber Regressor models with MAEs of 179.3158 ms and 179.9482 ms, respectively. These results facilitate the identification of the most promising models for further tuning and evaluation. This selection process is crucial as it narrows down the candidates to those with the highest potential for precision in predicting race completion times, setting the stage for the next phase of hyperparameter tuning.

Hyperparameter Tuning

Hyperparameter tuning was undertaken for the top-performing models identified through initial model screenings, aiming to further enhance their prediction accuracy for race completion times. We employed two distinct methods for this purpose:

PyCaret Tuning: Utilizing PyCaret's built-in optimization tools allowed for rapid iteration and tuning of model parameters.

GridSearchCV Tuning: This method conducted a more exhaustive search across a broader range of parameter configurations to finely tune the models.



	PyCaret Tuning	**GridSearchCV Tuning**
Model	**MAE (MS)**	**MAE (MS)**
Elastic Net	*121.88*	*113.92*
Linear Regression	*123.47*	*119.24*
Huber Regressor	*124.70*	*115.17*

Figure 29 Tuning Results from Pycaret and GridSearchCV for Selected Models

The results of the hyperparameter tuning are illustrated in Figure 29, which presents the Mean Absolute Error (MAE) for each model following adjustments made via PyCaret and GridSearchCV. The table clearly demonstrates that both tuning methods significantly improved the models' predictive accuracy. The **Elastic Net** model, in particular, showed superior performance with the lowest MAE of 113.92 ms after GridSearchCV tuning, confirming its effectiveness in predicting race times using the selected features. The **Linear Regression** and **Huber Regressor** models also saw improved accuracy, with MAEs of 119.24 ms and 115.17 ms, respectively, indicating robust enhancements though slightly trailing behind the Elastic Net.

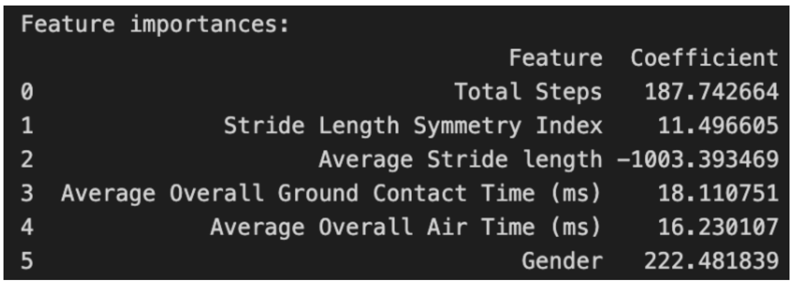
The best settings for the ElasticNet model were:

- Alpha: 0.1
- L1 Ratio: 0.99
- Positive: False
- Selection: 'random'
- Tolerance: 0.001

These parameters suggest a model finely tuned for this specific dataset, balancing the complexity of the model with the need to prevent overfitting. This level of precision in parameter tuning enhances the model's performance and adaptability to new data, ensuring robustness in practical applications.

ElasticNet Model Coefficients

Figure 30 presents the summary of the ElasticNet model, which has been refined through rigorous hyperparameter tuning. This model provides critical insights into the variables that significantly impact race completion times, highlighting the interplay of physical attributes and biomechanical efficiency.



```
Feature importances:
                                  Feature  Coefficient
0                             Total Steps   187.742664
1            Stride Length Symmetry Index    11.496605
2                   Average Stride length -1003.393469
3  Average Overall Ground Contact Time (ms)    18.110751
4            Average Overall Air Time (ms)    16.230107
5                                  Gender   222.481839
```

Figure 30 ElasticNet Summary

The model identifies **Average Stride Length** with a coefficient of -994.670468, indicating a strong negative relationship with race times. This suggests that longer strides are beneficial, as they are associated with decreased completion times, reflecting higher efficiency in an athlete's stride mechanics. It's an indication that training aimed at lengthening the stride could effectively improve race performance.

Gender shows a coefficient of 222.570542, revealing notable differences in race times between genders. This difference points to underlying physiological or biomechanical disparities that could be explored further to develop gender-specific training programs, potentially optimizing performance through tailored approaches.

The positive coefficient of 188.326301 for **Total Steps** implies that a higher count of steps correlates with increased race times. This finding underscores the importance of stride optimization, where reducing the number of steps through efficient stride length and frequency could lead to faster race completions.

Average Overall Ground Contact Time and **Average Overall Air Time** have coefficients of 18.115142 and 16.226401, respectively. Both metrics suggest that longer times spent either in contact with the ground or in the air are linked with slower race times. This insight is crucial for training focus, as it suggests benefits from drills that enhance foot strike efficiency and reduce non-productive air time, thereby promoting a more effective racing motion.

Lastly, the **Stride Length Symmetry Index** has a small coefficient of 11.466058, mirroring previous findings that athletes exhibit a high level of biomechanical balance with consistently low symmetry indices. This inclusion underlines its importance, slight changes in stride symmetry, whether improvements or deteriorations, can significantly affect race times. Thus, even small deviations in this area warrant attention in training programs to maintain and enhance athletic performance.

Interpretation and Practical Implications

This refined analysis using the **ElasticNet** model through PyCaret illustrates the substantial impact of advanced machine learning techniques in sports analytics. By enhancing models with precise parameter tuning, we have significantly reduced prediction errors, demonstrating the capabilities of sophisticated analytical methods in improving predictions of athletic performance.

The model's accuracy and predictive power, bolstered by comprehensive tuning, effectively show how machine learning can be integrated with sports science to offer actionable insights. Such data-driven approaches are pivotal in advancing our understanding of athlete performance, facilitating the development of scientifically backed, personalized training regimens tailored to the unique needs and characteristics of each athlete.

2) **tsfresh Extracted Features**

Data Preparation and Feature Selection

In this phase of the analysis, we utilized tsfresh extracted features that were previously identified as significant in earlier sections of the study. These features were specifically selected based on their demonstrated impact on race performance metrics and robustness in capturing essential dynamics of athletic performance. The selection was informed by comprehensive testing, which highlighted these features' ability to significantly enhance model accuracy. The selected features included:

- Right Upper Leg Position X (FFT Coefficient Absolute, Coefficient 82)
- L5 Orientation Q3 (FFT Coefficient Angle, Coefficient 99)
- T8 Orientation Q4 (Count Above Mean)
- Neck Acceleration Z (Number of Peaks)
- Right Shoulder Position X (Last Location of Minimum)
- Right Hand Angular Velocity Y (Aggregated Linear Trend, Standard Error, Chunk Length 10, Aggregate Function: Max)

- Left Lower Leg Orientation Q3 (Augmented Dickey-Fuller, Used Lag, Autolag: AIC)
- Left Lower Leg Position Y (Longest Strike Above Mean)
- T8 Acceleration Z (Augmented Dickey-Fuller, Used Lag, Autolag: AIC)

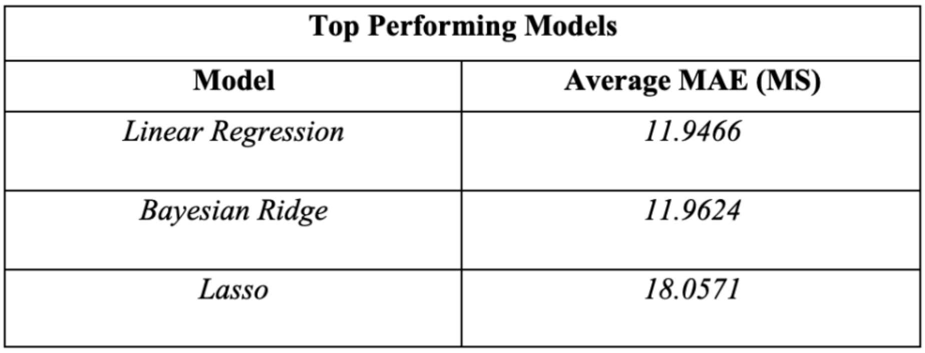
These features were chosen because previous analyses and model iterations demonstrated their effectiveness in predicting race completion times, providing a solid foundation for the current predictive modelling efforts.

Data Splitting

To maintain consistency across our modelling strategies, we continued with an 80/20 data partition—80% for training and 20% for testing. This standard practice in machine learning ensures that models have enough data to effectively learn the underlying patterns while also providing a substantial dataset for performance evaluation and avoiding overfitting.

Model Training with PyCaret

Using PyCaret, we trained a variety of models to find the best predictors of race times using the tsfresh extracted features. The initial training phase helped identify the top performers based on their average Mean Absolute Error (MAE). The following figure showcases the three models that excelled in terms of predictive accuracy.



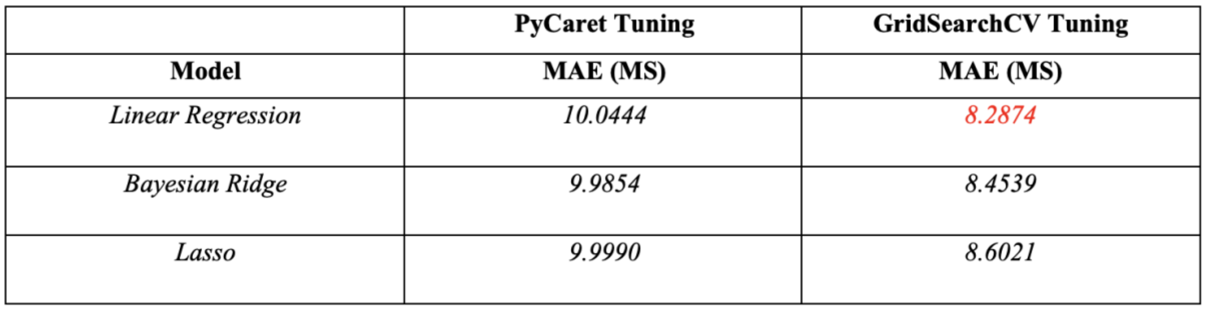
Top Performing Models	
Model	**Average MAE (MS)**
Linear Regression	*11.9466*
Bayesian Ridge	*11.9624*
Lasso	*18.0571*

Figure 31 Best Performing Models Based on Average Mean Absolute Error

In the competitive assessment of models**, Linear Regression** emerged as the leader with the lowest Average Mean Absolute Error (MAE) of 11.9466 ms, demonstrating its high accuracy in forecasting race completion times with the selected tsfresh extracted features. Closely following was **Bayesian Ridge**, which achieved an Average MAE of 11.9624 ms, indicative of its proficiency in managing the complex interactions inherent in the dataset. **Lasso**, though slightly less accurate with an Average MAE of 18.0571 ms, still marked itself as a strong contender, showcasing the robustness of these predictive models in the domain of sports analytics. This trio of models now prepares the groundwork for deeper analysis and hyperparameter tuning aimed at refining their predictive capabilities and optimizing their real-world application to athletic performance forecasting.

Hyperparameter Tuning

In the pursuit of optimizing model performance, we conducted extensive hyperparameter tuning using both PyCaret and GridSearchCV. This stage was crucial to refine the accuracy of our selected models: Linear Regression, Bayesian Ridge, and Lasso.



	PyCaret Tuning	**GridSearchCV Tuning**
Model	**MAE (MS)**	**MAE (MS)**
Linear Regression	*10.0444*	*8.2874*
Bayesian Ridge	*9.9854*	*8.4539*
Lasso	*9.9990*	*8.6021*

Figure 32 Tuning Results from Pycaret and GridSearchCV for Selected Models

As shown in Figure 32, both tuning methods successfully improved the models' Mean Absolute Error (MAE) scores. PyCaret’s automated tuning tools provided substantial enhancements, while GridSearchCV's more exhaustive parameter search yielded even better performance, demonstrating the power of thorough parameter optimization.

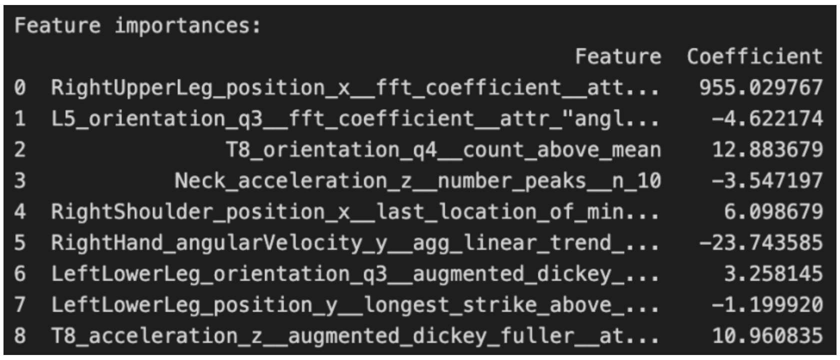
For **Linear Regression**, the best results were obtained with GridSearchCV tuning, which delivered an impressive MAE reduction to 8.2874 ms. This level of precision was achieved using optimal settings that included fitting the model with an intercept and without constraining the coefficients to be positive, allowing for a flexible adaptation to the underlying data patterns.

The **Bayesian Ridge** and **Lasso** models also saw significant improvements, with GridSearchCV tuning bringing their MAEs down to 8.4539 ms and 8.6021 ms, respectively. These results underscore the effectiveness of meticulous parameter adjustments in enhancing the predictive capabilities of our models.

The detailed tuning of these models not only ensures greater predictive accuracy but also enhances their applicability to real-world scenarios in sports analytics. By achieving such low MAE scores, the models demonstrate their potential for precise and reliable race time predictions, setting a strong foundation for further validation and application.

LinearRegression Model Coefficients

Figure 33 presents the summary of the LinearRegression model derived from our analysis, highlighting the statistically significant coefficients and their implications for race times.



```
Feature importances:
                                            Feature  Coefficient
0  RightUpperLeg_position_x__fft_coefficient__att...   955.029767
1  L5_orientation_q3__fft_coefficient__attr_"angl...    -4.622174
2                T8_orientation_q4__count_above_mean    12.883679
3             Neck_acceleration_z__number_peaks__n_10    -3.547197
4  RightShoulder_position_x__last_location_of_min...     6.098679
5  RightHand_angularVelocity_y__agg_linear_trend_...   -23.743585
6  LeftLowerLeg_orientation_q3__augmented_dickey_...     3.258145
7  LeftLowerLeg_position_y__longest_strike_above_...    -1.199920
8  T8_acceleration_z__augmented_dickey_fuller__at...    10.960835
```

Figure 33 LinearRegression Summary

The **Right Upper Leg Position X (FFT Coefficient Abs Coeff 82)** has a coefficient of 955.029767, indicating a significant impact on race times. This suggests that increased movement or variation in the right upper leg's position correlates with longer completion times, emphasizing the need for controlled and efficient leg movements during the run.

For the **T8 Orientation Q4 Count Above Mean**, with a coefficient of 12.883679, the data implies that higher counts above the mean orientation at the T8 spinal segment enhance race completion times. This may indicate more consistent upper body posture or stability, which is crucial for maintaining speed and reducing fatigue over the duration of the race.

The coefficient of 10.960835 for **T8 Acceleration Z (Augmented Dickey-Fuller Usedlag AIC)** reflects the importance of forward and backward acceleration at the T8 segment, likely related to the running posture and vertical oscillations. Managing these accelerations effectively can improve overall completion times by ensuring core stability and efficient energy utilization, which are vital for competitive running.

A negative coefficient of -23.743585 for **Right Hand Angular Velocity Y (Aggregated Linear Trend, Stderr, Max)** demonstrates that higher angular velocities of the right hand are associated with shorter completion times. Efficient arm swing dynamics contribute positively to propulsion and momentum, highlighting the necessity of proper synchronization of arm movements with leg strides to maximize speed and efficiency.

Interpretation and Practical Implications

This analysis using PyCaret and GridSearchCV illustrates the profound impact of advanced machine learning techniques in sports analytics. Sophisticated tools for hyperparameter tuning have led to a significant reduction in prediction error, showcasing the potential of modern analytical methods to enhance athletic performance predictions.

The **Linear Regression** model, enhanced through meticulous tuning, has demonstrated remarkable accuracy and predictive power. This model not only confirms the effectiveness of integrating machine learning with sports science but also offers actionable insights. These insights are pivotal for coaches, suggesting focused training on critical biomechanical aspects such as upper leg mechanics, arm swing dynamics, core stability and posture. By optimizing these areas, significant improvements in athletes' performance times can be achieved.

These findings highlight that strategic training, informed by data-driven insights, can lead to marked improvements in athletic performance. They offer valuable guidance for coaches, suggesting targeted areas for training enhancements and more precise athlete monitoring. This approach ensures that training is scientifically grounded and tailored to meet individual athlete needs, potentially transforming conventional training methodologies in sports.

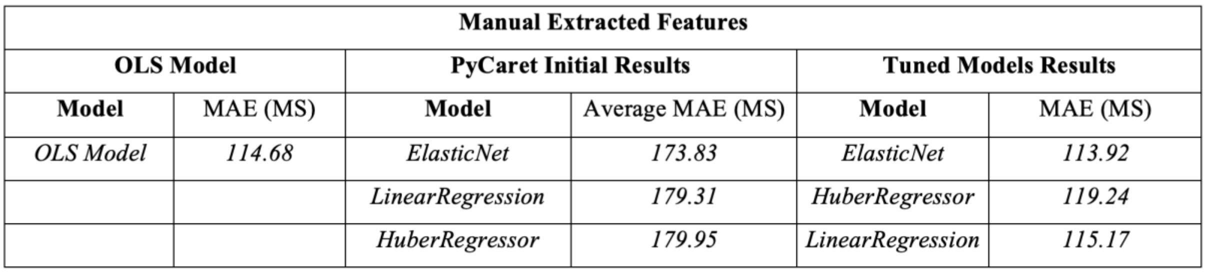
5.3 Comparative Analysis of OLS and PyCaret Outcomes

Introduction to Comparative Analysis

This analysis compares the effectiveness of the Ordinary Least Squares (OLS) regression with various machine learning models developed using PyCaret. The focus is on evaluating how OLS, a model without hyperparameter tuning capabilities, stands against potentially more complex models from PyCaret that benefit from extensive tuning. By analysing their performance on both manually and automatically extracted features, this comparison aims to elucidate which modelling approach yields the best balance between accuracy, simplicity, and computational efficiency.

Performance Metrics Comparison

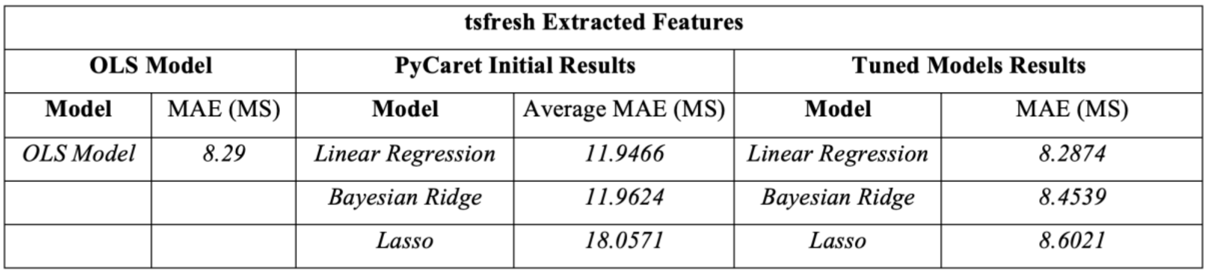
The comparative analysis initially reveals a noteworthy finding: before tuning, PyCaret models did not outperform the simpler, untuned OLS model. This initial performance is depicted in the tables below, which contrast the results for manually and automatically extracted features.



Manual Extracted Features					
OLS Model		**PyCaret Initial Results**		**Tuned Models Results**	
Model	MAE (MS)	**Model**	Average MAE (MS)	**Model**	MAE (MS)
OLS Model	*114.68*	*ElasticNet*	*173.83*	*ElasticNet*	*113.92*
		LinearRegression	*179.31*	*HuberRegressor*	*119.24*
		HuberRegressor	*179.95*	*LinearRegression*	*115.17*

Figure 34 Comparison of OLS and PyCaret Models Using Manually Extracted Features

This table shows the Mean Absolute Error (MAE) for the OLS model alongside the initial and tuned results of the PyCaret models. Initially, all PyCaret models—ElasticNet, LinearRegression, and HuberRegressor—recorded higher MAEs than the OLS model. However, after tuning, the PyCaret models demonstrated significant improvements, with ElasticNet showing the most considerable enhancement, reducing its MAE to closely match that of the OLS model.



tsfresh Extracted Features					
OLS Model		**PyCaret Initial Results**		**Tuned Models Results**	
Model	MAE (MS)	**Model**	Average MAE (MS)	**Model**	MAE (MS)
OLS Model	*8.29*	*Linear Regression*	*11.9466*	*Linear Regression*	*8.2874*
		Bayesian Ridge	*11.9624*	*Bayesian Ridge*	*8.4539*
		Lasso	*18.0571*	*Lasso*	*8.6021*

Figure 35 Comparison of OLS and PyCaret Models Using tsfresh Extracted Features

Similarly, the second table presents the outcomes for models utilizing tsfresh extracted features. The OLS model initially outperformed all PyCaret models. Nevertheless, following tuning, all PyCaret models substantially improved their performance, with LinearRegression achieving the most notable reduction in MAE, surpassing the OLS model.

These tables clearly illustrate the impact of tuning on the performance of PyCaret models. While the initial results were less impressive, the tuned models align more closely with or even surpass the simplicity and effectiveness of the OLS model. This underscores the potential enhancements that advanced machine learning techniques can offer over traditional models when appropriately tuned for specific datasets.

The outcomes from this comparative analysis not only highlight the importance of hyperparameter tuning in enhancing model performance but also demonstrate the efficacy of machine learning models in sports analytics when they are optimally adjusted. This comprehensive comparison provides valuable insights into the balance between model simplicity and computational sophistication, guiding future modelling strategies in sports performance prediction.

Model Complexity and Training Time

OLS, with its absence of hyperparameter tuning, offers a straightforward and computationally efficient modelling approach, ideal for situations where simplicity is prioritized. In contrast, while PyCaret models initially underperformed, they reached competitive performance levels after tuning. However, this enhancement required additional computational resources and time, illustrating a trade-off between simplicity and the need for complex adjustments to achieve optimal performance.

Robustness and Generalization

Both the OLS and PyCaret models underwent rigorous evaluation, including 10-fold cross-validation, to ensure their robustness and generalizability. The notable improvement in PyCaret models post-tuning suggests that with meticulous calibration, these models can provide robust predictions, though this comes with an increase in training complexity.

Practical Implications

In scenarios where computational resources and time are limited, OLS provides a viable and effective solution. However, if conditions permit extended model exploration and tuning, PyCaret's models offer considerable customization and potentially greater accuracy, making them suitable for more in-depth analytical tasks.

Conclusion

This comparative analysis highlights the distinct strengths of OLS and PyCaret within various contexts. OLS excels in delivering simplicity and satisfactory performance without the necessity for tuning, making it well-suited for quicker deployments. Conversely, PyCaret offers enhanced flexibility and potentially superior performance but requires significant tuning and more computational resources to fully leverage its capabilities. The choice between these modelling approaches should be guided by the specific needs of the project, the available resources, and the desired complexity of the model.

5.4 Challenges and Limitations

The process of modelling completion times using OLS and PyCaret has highlighted several challenges and exposed limitations inherent in predictive modelling within sports analytics. These challenges range from data quality and feature selection to model complexity and generalizability.

Data Quality and Availability

The data for this study was sourced from a specific group of athletes, which limits the diversity and potentially affects the generalizability of the findings. The motion capture sessions were well-conducted but limited in number, leading to models that may perform well for this particular group but less so for a broader athlete population. This highlights the need for datasets that include a wider range of athletes to enhance the robustness and applicability of the predictive models.

Feature Selection and Engineering

The study employed both manually extracted and tsfresh automated features. While manual selection allows for domain expertise to guide the process, it risks omitting potentially informative features. Conversely, automated methods can introduce features that, although statistically significant, might not provide practical insights and could lead to overfitting or reduced interpretability. These tsfresh features are not only complex and numerous, presenting significant challenges in interpretation and obscuring model transparency, but their extraction also demands considerable computational resources. This requirement can be particularly prohibitive in scenarios where rapid model training and deployment are essential.

Model Selection and Hyperparameter Tuning The effectiveness of the models, particularly those trained with PyCaret, heavily relied on hyperparameter tuning. This tuning is resource-intensive and can limit the feasibility of these models in scenarios requiring quick responses or limited computational resources. Furthermore, while complex models can capture intricate data patterns, they also risk overfitting, making it essential to balance complexity with generalization capabilities.

Generalizability and Validation

Although 10-fold cross-validation helps assess model robustness, it doesn't guarantee performance on unseen data from different populations or under varied conditions.

The predictive models developed in this study are tailored to the specifics of short-distance sprint data, derived from a controlled group of athletes during motion capture sessions explicitly focused on short sprints. The nature of our dataset, which includes only short-distance sprint sections, precludes the evaluation of these models for long-distance races like marathons or extended athletic events. This limitation restricts the applicability of our findings outside of short sprint contexts, emphasizing the need for targeted data collection across different distances and formats to validate and potentially extend the utility of these models to broader athletic applications.

Chapter 6

Conclusion & Future Work

6.1 Restating Objectives

The primary objective of this thesis was to leverage advanced machine learning techniques to enhance the analysis and prediction of athletic performance in track and field using motion capture data. To achieve this, the study aimed to construct and evaluate predictive models that accurately estimate competition times for track and field athletes. These models utilized both manually extracted features, informed by theoretical and practical knowledge of biomechanics, and automatically derived features using the tsfresh tool to identify the most predictive features. In analysing running techniques, the models provided insights into key factors affecting performance, enabling the provision of actionable feedback for athletes and coaches to optimize their training outcomes. Moreover, this study explored the potential of predictive modelling for injury prevention by identifying patterns and anomalies in motion data that could predispose athletes to injuries, contributing to strategies for proactive injury prevention. The study also compared different modelling approaches, including OLS regression and various models through PyCaret, to establish best practices in handling complex sports performance data. Finally, it demonstrated how integrating machine learning insights into coaching regimens can transform traditional training methods, facilitating data-driven decisions with practical implications for real-world athletic training and competition settings. Ultimately, this study sought to empower athletes and coaches with valuable insights to enhance training strategies and performance.

6.2 Summary of Key Findings

Predictive Models: The predictive models constructed in this study successfully estimated competition times by utilizing both manually and automatically extracted features. The Linear Regression model, which used features derived from the tsfresh tool, achieved the most accurate predictions with a Mean Absolute Error of 8.29 milliseconds. This finding underscores the significant predictive power of automated feature extraction techniques.

Running Techniques: The analysis of running techniques provided valuable insights into the key biomechanical factors influencing athletic performance. By using both manually and automatically extracted features, this study identified specific aspects of an athlete's movement that significantly affect competition times.
Manually extracted features emphasized the importance of balancing stride length, stride frequency, and symmetry. Reducing both ground contact time and air time proved crucial for efficient motion, as shorter times were associated with faster race outcomes. Optimizing the total number of steps also emerged as important, reinforcing the need to balance stride frequency and length.
Automatically extracted features highlighted subtle but essential patterns in leg positioning, core stability, and arm-leg synchronization, revealing intricate biomechanical patterns crucial for effective performance. These features emphasized the importance of controlled and efficient leg movements, core stability, and arm swing coordination in enhancing athletic speed and minimizing fatigue.
Together, these insights offer a comprehensive understanding of how different running technique elements affect race outcomes and provide practical strategies for refining training regimens. By leveraging these findings, coaches and athletes can tailor their training to optimize individual performance while minimizing injury risks.

Injury Prevention: Although this study primarily focused on performance improvement, the predictive models provide a foundational framework for identifying patterns that may predispose athletes to injury. Enhanced analysis could identify problematic patterns early.

Comparing Modelling Techniques: The comparative analysis between Ordinary Least Squares (OLS) regression and machine learning models developed using PyCaret highlighted the strengths and trade-offs of each approach. OLS, despite its lack of hyperparameter tuning, initially outperformed the untuned PyCaret models, showcasing its ability to deliver reasonable predictive accuracy with minimal complexity.

After tuning, the PyCaret models significantly improved their predictive performance, aligning with or even surpassing OLS results for both manually and automatically extracted features. However, this improvement required additional computational resources and time, highlighting a trade-off between simplicity and optimization.

OLS is ideal for quick deployment in scenarios where computational efficiency is essential. In contrast, PyCaret provides greater customization and accuracy through hyperparameter tuning but demands more computational resources. Choosing between these approaches depends on specific project needs, available resources, and the desired level of model complexity. This analysis ultimately underscores the importance of model selection and optimization in sports analytics.

Data-Driven Decisions: The study demonstrated the potential of machine learning models to inform training strategies by providing actionable insights into each athlete's performance. Coaches can tailor training regimens using detailed metrics from predictive modelling, which identify individual strengths and weaknesses. These insights give coaches a nuanced understanding of each athlete's biomechanics, allowing them to focus on training areas that will yield the greatest improvement.

6.3 Challenges and Limitations

This study encountered several challenges related to data quality, computational resources, feature selection, and model generalizability. The dataset, drawn from a specific group of athletes, lacked diversity, which affected generalizability. The limited number of motion capture sessions resulted in models that were highly tailored to this group, restricting their broader applicability. A more varied dataset would improve the robustness of the predictive models.

Manual feature extraction, guided by domain expertise, may overlook key features, while automated selection can introduce statistically significant but less interpretable features. The automated extraction process via tsfresh is resource-intensive and generates complex features that can lead to interpretation challenges and computational burdens.

The performance of models trained via PyCaret was highly dependent on tuning, which is computationally expensive and time-consuming. Additionally, balancing model complexity with the ability to generalize to new data remains a critical challenge.

While 10-fold cross-validation aided model robustness, it does not guarantee performance on unseen data. These models, tailored to short-distance sprint data, were not validated on long-distance races, limiting their applicability beyond the sprint context. Data collection across different race lengths is necessary for broader validation.

6.4 Future Work

Enhanced Data Collection: Future studies should focus on gathering a more diverse dataset across different athlete populations and race formats. This will improve model robustness, enable better generalization, and enhance our understanding of performance patterns in various track and field events.

Advanced Modelling Techniques: Exploring advanced modelling techniques, such as deep learning and ensemble methods, could further refine predictive accuracy. Additionally, testing models that incorporate real-time data streams and external variables, such as environmental conditions, can yield richer insights and more dynamic predictions.

Real-World Applications: Integrating predictive models into real-world coaching environments is crucial for validating their practicality and effectiveness in competitive contexts. Wearable technology and training software can serve as valuable tools for implementing these models, providing coaches and athletes with real-time feedback and actionable insights

Improving Interpretability: Developing more interpretable models will enhance the understanding of predictive insights among coaches and athletes. This improved interpretability will facilitate more effective communication and integration of data into training strategies, ensuring that the insights are practically applied to improve performance.

Injury Prevention: Future work in injury prevention can be significantly advanced by combining a comprehensive review of theoretical research on track and field injuries with deeper, more meaningful feature extraction to identify critical biomechanical patterns. Delving into this body of knowledge, alongside collecting richer data on the biomechanics of athletes, could reveal subtle patterns or specific risk factors associated with injuries. For instance, as demonstrated with the stride symmetry index, these patterns may provide early indications of imbalances or inefficiencies in running mechanics that predispose athletes to injuries.

Through detailed analysis of these movement patterns, we can develop predictive models that offer immediate, actionable feedback on each athlete's biomechanical health. These insights would help coaches and medical staff identify potential injury risks, customize training to address specific vulnerabilities, and improve recovery protocols. Moreover, such models would empower athletes to proactively adjust their running mechanics to minimize injury risks while maximizing performance. Real-time, data-driven body analysis could ultimately lead to the development of promising tools and technologies that not only safeguard athlete health but also enhance their competitive edge through optimized running mechanics and improved performance results.

References

[1] Abhinandan Aggarwal, Rohit Gupta, Ravinder Agarwal. "Design and Development of Integrated Insole System for Gait Analysis."

[2] Duncan A. J. Blythe, Franz J. Király. "Prediction and Quantification of Individual Athletic Performance of Runners."

[3] Elizabeth S. Chumanov, Bryan C. Heiderscheit, Darryl G. Thelen. "The effect of speed and influence of individual muscles on hamstring mechanics during the swing phase of sprinting."

[4] Cornelis J. de Ruiter, Ben van Oeveren, Agnieta Francke, Patrick Zijlstra, Jaap H. van Dieen. "Running Speed Can Be Predicted from Foot Contact Time during Outdoor over Ground Running."

[5] Giorgos Ioannou, Andrei Kazlouski, Thomas Marchioro, Maarten Gijssel. "Minimal wearable setup via sensor correlation: a case study of field hockey players."

[6] "Kinetic Analysis." URL: https://www.kinetic-analysis.com

[7] S. Manzer, K. MatteS, K. HOllänDer. "Kinematic Analysis of Sprinting Pickup Acceleration versus Maximum Sprinting Speed."

[8] J.B. Morin, P. Samozino, K. Zameziati, A. Belli. "Effects of altered stride frequency and contact time on leg-spring behavior in human running."

[9] "MVNX Version 4 File Structure."
URL: https://base.movella.com/s/article/MVNX-Version-4-File-Structure

[10] Sandro Nigg, Jordyn Vienneau, Christian Maurer, Benno M. Nigg. "Development of a symmetry index using discrete variables."

[11] Tom F. Novacheck. "The biomechanics of running."

[12] A. Nummela, T. Keränen, L. O. Mikkelsson. "Factors Related to Top Running Speed and Economy."

[13] "PyCaret 3.0 Documentation." URL: https://pycaret.gitbook.io/docs

[14] "Python xml.etree.ElementTree — The ElementTree XML API Documentation." URL: https://docs.python.org/3/library/xml.etree.elementtree.html

[15] Marzena Paruzel-Dyja, Anna Walaszczyk, Janusz Iskra. "Elite Male and Female Sprinters' Body Build, Stride Length and Stride Frequency."

[16] Peter G. Weyand, Deborah B. Sternlight, Matthew J. Bellizzi, Seth Wright. "Faster top running speeds are achieved with greater ground forces not more rapid leg movements."

[17] Rebecca Avrin Zifchock, Irene Davis, Jill Higginson, Todd Royer. "The symmetry angle: A novel, robust method of quantifying asymmetry."

[18] "tsfresh Documentation." URL: https://tsfresh.readthedocs.io/en/latest/text/introduction.html

[19] "Xsens Animate Tool" URL: https://www.movella.com/products/motion-capture/xsens-mvn-animate