

Predicting Athletic Performance: A Machine Learning Analysis of Powerlifting

Rohit Mishra

Department of Computer Science, Virginia Tech – Falls Church

CS 5085: Machine Learning 1

Instructor :Dr. Reza Jafari, PHD

November 27, 2024

Table of Contents

NAME	PAGE
Abstract	5
Introduction	7
Description	9
Phase I: Feature Engineering and EDA	12
Data Cleaning Methodology	16
Analysis of Strength Distribution Across Equipment Types	17
Analysis of Strength Distribution Across Age	20
Bodyweight Distribution Analysis and Outlier Methodology	23
Best Squat Distribution Analysis and Outlier Methodology	26
Analysis of Testing Status Impact on Powerlifting	32
Dimensionality reduction and Feature Selection	32
Random Forest Feature Analysis	32
Principal Component and Condition Number Analysis	35
Singular Value Decomposition Analysis	39
Variance Inflation Factor (VIF) Analysis	44
SVD as an Optimal Dimensionality Reduction Technique	46
Discretization and Binarization	48
Variable Transformation Analysis	50
Covariance Matrix Analysis	51
Correlation Matrix Analysis	53
Phase II: Regression Analysis	55
Multiple Linear Regression Analysis	55
XG Boost Model Analysis	64
Phase III: Classification Analysis	67
Analysis of Feature Covariance and Dimensionality	70
Decision Tree Classification Analysis	74
Logistic Regression Classification Analysis	80
KNN Classification Analysis	84
Naïve Bayes Classification Analysis	89
Support Vector Machines Classification Analysis	93
Linear Kernel Analysis	93
Polynomial Kernel Analysis	97
Radial Base Kernel Analysis	101
Neural Network Classification Analysis	105
Performance Analysis of Classification Models	109
Phase IV: Clustering And Association	113
K-means Clustering Analysis of Powerlifting Competition	113
Association Rule Mining in Raw Powerlifting (Apriori)	120
DBSCAN Analysis in Raw Powerlifting	125
Recommendation	129
Appendix	134

References	151

Table of Figures

NAME	PAGE
Figure 1	17
Figure 2	20
Figure 3	23
Figure 4	25
Figure 5	26
Figure 6	30
Figure 7	35
Figure 8	36
Figure 9	39
Figure 10	40
Figure 11	51
Figure 12	53
Figure 13	60
Figure 14	62
Figure 15	63
Figure 16	65
Figure 17	70
Figure 18	72
Figure 19	75
Figure 20	75
Figure 21	75
Figure 22	76
Figure 23	76
Figure 24	77
Figure 25	78
Figure 26	78
Figure 27	80
Figure 28	81
Figure 29	82
Figure 30	82
Figure 31	84
Figure 32	84
Figure 33	85
Figure 34	86
Figure 35	87
Figure 36	89
Figure 37	90
Figure 38	91

Figure 39	91
Figure 41	93
Figure 42	94
Figure 43	95
Figure 44	95
Figure 45	97
Figure 46	98
Figure 47	99
Figure 48	99
Figure 49	101
Figure 51	102
Figure 52	103
Figure 53	105
Figure 54	106
Figure 55	107
Figure 56	107
Figure 57	109
Figure 58	110

Abstract

This research project investigates the powerlifting dataset through comprehensive machine learning analysis, focusing on two main objectives: predicting total competition weight (TotalKg) and classifying drug testing status (Tested). The study employs multiple supervised learning algorithms, including Decision Trees, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), and Neural Networks, complemented by clustering analysis using K-means and DBSCAN.

For regression analysis, the model predicts TotalKg using variables including Best3DeadliftKg, BodyweightKg, Age, Sex, Equipment, and Federation membership. Feature engineering, particularly the creation of a derived Strength metric, significantly improved model performance. The classification analysis predicts testing status using features such as Sex_encoded, Age, Strength, and Federation tiers.

The results demonstrate that Decision Trees achieved the best classification performance with an AUC of 0.938 and high precision (0.985). Clustering analysis revealed two distinct DBSCAN clusters and eight K-means clusters, providing insights into athlete groupings based on performance patterns and competition characteristics. The study's findings contribute to understanding performance factors in powerlifting and the relationship between competition levels and testing protocols.

This project highlights the effectiveness of machine learning techniques in sports analytics, specifically in predicting performance metrics and understanding testing patterns in powerlifting.

competitions. The results provide valuable insights for competition organizers, coaches, and athletes in understanding performance determinants and testing protocols in the sport.

Introduction

Overview of Powerlifting as a Sport

Powerlifting is a strength sport that tests athletes' maximal strength through three fundamental lifts: the squat, bench press, and deadlift. Unlike Olympic weightlifting, which focuses on explosive movements, powerlifting emphasizes pure strength in these three compound movements. Athletes compete in weight classes and are categorized by age, gender, and equipment usage (raw/classic or equipped).

Key Components:

1. Competition Lifts:

- Squat: Barbell lifted from a rack, lowered to below parallel, and returned to standing
- Bench Press: Barbell lowered to chest and pressed to full arm extension
- Deadlift: Barbell lifted from floor to hip level with locked knees and shoulders

2. Competition Rules:

- Three attempts allowed for each lift
- Best successful attempt in each lift contributes to total
- Strict technical requirements and commands from referees
- Equipment categories: Raw/Classic (minimal support gear) vs. Equipped (supportive suits/shirts)

Project Objectives and Methodology

This Final Term Project (FTP) aims to apply machine learning techniques to analyze powerlifting competition data through several phases:

Phase I: Data Preprocessing and Feature Engineering

- Dataset cleaning and standardization

- Feature creation and transformation
- Handling missing values and outliers
- Implementation of SMOTE for class balancing

Phase II: Regression Analysis

- Prediction of total competition weight (TotalKg)
- Feature selection and engineering
- Model development and evaluation
- Performance optimization techniques

Phase III: Classification Analysis

- Prediction of drug testing status
- Implementation of multiple classifiers:
 - Decision Trees with pruning optimization
 - Support Vector Machines with various kernels
 - K-Nearest Neighbors
 - Neural Networks
 - Logistic Regression
 - Naive Bayes

Phase IV: Clustering Analysis

- Implementation of K-means clustering
- DBSCAN algorithm application
- Association rule mining using Apriori algorithm

The analysis utilizes a comprehensive dataset containing competition results, athlete information, and performance metrics, providing insights into patterns and relationships within the sport of powerlifting. This project aims to contribute to the understanding of performance factors in powerlifting and provide valuable insights for athletes, coaches, and competition organizers through data-driven analysis.

Description of the Dataset

The Open Powerlifting dataset is a comprehensive collection of powerlifting competition results from international competitions across various federations. This dataset tracks athlete performances in the sport of powerlifting, which consists of three main lifts: squat, bench press, and deadlift. For each of these three lifts, an athlete is given three attempts. The attempts must follow a progressive pattern where each subsequent attempt must be heavier than the previous one. If an athlete fails to complete a lift successfully, they cannot decrease the weight for their next attempt in that lift. The best successful attempt from each lift is then combined to calculate the athlete's total score for the competition.

Dataset Characteristics

1. The dataset includes both raw and equipped lifting results
2. Records span multiple federations, countries, and weight classes
3. Contains both male and female competitors
4. Tracks various age groups and competition divisions

Justification

1. This extensive dataset comprises over 1.6 million observations, making it a rich source of data for analysis
2. The dataset contains both numerical features, such as "Squat Kg," "Bench Kg," "Deadlift Kg," and categorical features, including "Sex," "Event," and "Country." etc.
3. Since the Open Powerlifting dataset is publicly available, it ensures transparency and reproducibility in our analysis.

Industry Relevance

1. The dataset functions as a crucial tool for competition planning and performance optimization. Athletes and coaches can use historical data to set realistic goals, select appropriate weight classes, and develop effective attempt strategies. The inclusion of data from multiple

federations allows competitors to identify organizations that best match their competitive aspirations.

2. This comprehensive database enables coaches and athletes to develop evidence-based training programs by analyzing successful lifting patterns and understanding the relationship between body weight and strength. The detailed recording of equipment usage and performance outcomes provides essential insights for training methodology and competition strategy.

Variable Selection and Analysis Report

1. Regression Analysis (Total Weight Prediction)

Dependent Variable

- TotalKg: Total weight lifted in competition

Independent Variables

1. Performance Metrics:
 - Best3DeadliftKg: Best deadlift performance
 - BodyweightKg: Athlete's body weight
2. Demographic Features:
 - Age: Athlete's age
 - Sex: Gender of athlete
3. Competition Factors:
 - Equipment: Type of equipment used
 - Federation: Competition organizing body
 - Tested: Drug testing status

2. Classification Analysis (Testing Status Prediction)

Dependent Variable

- Tested: Binary variable indicating drug testing status (0: Not Tested, 1: Tested)

Independent Variables

1. Athlete Characteristics:
 - Sex
 - Age

- Strength (derived feature)
2. Competition Level Indicators:
- Fed_ELITE
 - Fed_MAJOR
 - Fed_REGIONAL

Phase1: Feature Engineering and Exploratory Data Analysis

Target Variable and Model Design

In competitive powerlifting, total lifted weight (**TotalKg**) serves as the primary performance metric. Our regression analysis adopts a two-step approach to predict this value effectively:

Step 1: Strength Prediction We first predict a normalized strength metric that accounts for variations in body mass and equipment usage. This intermediate step helps isolate fundamental strength patterns by reducing the impact of body size variations.

Step 2: Total Weight Conversion The predicted strength values are then converted to competition totals (TotalKg) by accounting for the athlete's body weight. This transformation provides results in the standard metric used in powerlifting competitions.

Variable Summary:

1. *TotalKg*: The final competition total in kilograms, serving as the primary target variable for regression tasks.
2. *Strength*: A normalized strength metric calculated as TotalKg divided by Bodyweight. This variable accounts for variations in body mass and will be used as an intermediate target variable in regression tasks.
3. *Sex*: Biological sex of the athlete, a fundamental factor influencing strength potential due to physiological differences. This categorical variable will be used for classification and clustering.
4. *Equipment*: The type of competition equipment used (Raw, Single-ply, Multi-ply, Wraps), which significantly affects lifting performance. This categorical variable will be utilized in classification and clustering tasks.
5. *Bodyweight*: The athlete's body mass, a crucial predictor of performance due to its influence on muscle cross-sectional area and leverages. This continuous variable will be used in regression, classification, and clustering tasks.

6. *Tested*: A binary indicator of drug testing status at competitions, potentially capturing systematic performance differences. This categorical variable will be used in classification and clustering tasks. We have identified data imbalance in this variable and have used SMOTE technique to balance the training data for classification
7. *Best3SquatKg*: The best squat performance, serving as a strong indicator of overall strength potential. This continuous variable will be utilized in regression tasks.
8. *Age*: The age of the athlete, capturing age-related performance patterns and physiological factors. This continuous variable will be used in regression, classification, and clustering tasks.

EDA Findings:

1. *Distribution analysis*: Explore the distribution of each variable, identifying potential outliers, skewness, and kurtosis. Visualize the distributions using histograms, box plots, and probability plots.
2. *Relationship analysis*: Investigate the relationships between variables using scatter plots, correlation matrices, and pair plots. Identify potential linear or non-linear relationships between variables.
3. *Group comparisons*: Compare the distribution and statistical measures of variables across different groups (e.g., sex, equipment type, tested status) using box plots, violin plots, and summary statistics.
4. *Missing data analysis*: Assess the presence and patterns of missing data in the dataset. Determine appropriate strategies for handling missing values (e.g., imputation, deletion).
5. *Feature engineering*: Create new features based on domain knowledge or observed patterns in the data. For example, creating interaction terms between variables or transforming variables to capture non-linear relationships.

Next Steps:

Based on the EDA findings, the following steps will be taken for classification, regression, and clustering tasks:

1. *Data preprocessing*: Handle missing data, encode categorical variables, and scale continuous variables as needed.
2. *Data cleaning*: Check for and remove any duplicate data points in the dataset. Our initial analysis did not find any duplicates, but it's important to ensure data integrity before proceeding with modeling tasks.
3. *Feature selection*: Select the most informative features for each task based on their relevance and potential impact on the target variable.
4. *Model selection*: Choose appropriate models for each task (e.g., logistic regression for classification, linear regression for regression, k-means for clustering) based on the nature of the variables and the relationships observed in the EDA.
5. *Model evaluation*: Assess the performance of the selected models using appropriate evaluation metrics and validation techniques (e.g., cross-validation, holdout validation).

By leveraging the insights gained from this EDA, we aim to build robust and informative models for classification, regression, and clustering tasks in the powerlifting dataset.

Features Excluded from Analysis

Several variables were intentionally excluded from our model based on various methodological and practical considerations:

Competition-Specific Data:

Individual Attempt Records (Squat1-4Kg, Bench1-4Kg, Deadlift1-4Kg): While the first squat attempt could provide valuable insights for coaches in weight selection strategy for subsequent lifts, including these attempts would effectively leak information about the final total. These attempts directly sum to the competition total, making their inclusion circular in nature.

Performance Scoring Systems:

Wilks, McCulloch, Glossbrenner, IPFPoints: These standardized scoring systems are calculated using the total of all three lifts (squat, bench, deadlift) along with bodyweight and other factors. Including them would introduce data leakage as they contain information about our target

variable (total). Their inclusion would artificially inflate model accuracy without adding genuine predictive value.

Categorical Variables with High Cardinality:

1. *Name*: Individual athlete identifiers
2. *MeetName, MeetState, Country*: Competition location data
3. *Division*: Competition categories
4. *WeightClassKg, AgeClass*: Redundant with actual bodyweight and age measurements

These variables either have too many unique values or provide information already captured in our selected features.

Competition Outcome:

- *Place*: Competition ranking is an outcome variable rather than a predictor
- *Event*: All selected data pertains to full powerlifting competitions

This feature selection strategy maintains model integrity by avoiding both data leakage and redundancy while focusing on genuinely predictive variables.

Data Cleaning Methodology

Our data cleaning process involved several systematic steps to ensure data quality and reliability for subsequent analysis:

1. Missing Value Treatment

Testing Status Standardization

- Converted 'Tested' column to binary format (1 for 'Yes', 0 for untested)
- This standardization enables numerical analysis while maintaining interpretability

Age Data Imputation

- Identified significant missing age values in the dataset
- Implemented a division-based imputation strategy:
 - Calculated mean ages for each competition division
 - Applied these averages to missing values within the same division
 - Maintained data integrity by marking unmatched cases as NaN
- This approach preserves the age-division relationship while handling missing data

2. Data Validation and Filtering

Lift Data Validation

- Removed invalid lift records through logical constraints:
 - Eliminated zero or negative values in squat, bench, and deadlift records
 - Ensured total weight exceeded individual lift components
 - These criteria eliminate physically impossible scenarios

Comprehensive Record Completion

- Applied complete case analysis for critical variables:
 - Total weight (TotalKg)
 - Individual lift performances (Best3SquatKg, Best3BenchKg, Best3DeadliftKg)
 - Body weight (BodyweightKg)
 - Age

Analysis of Strength Distribution Across Equipment Types

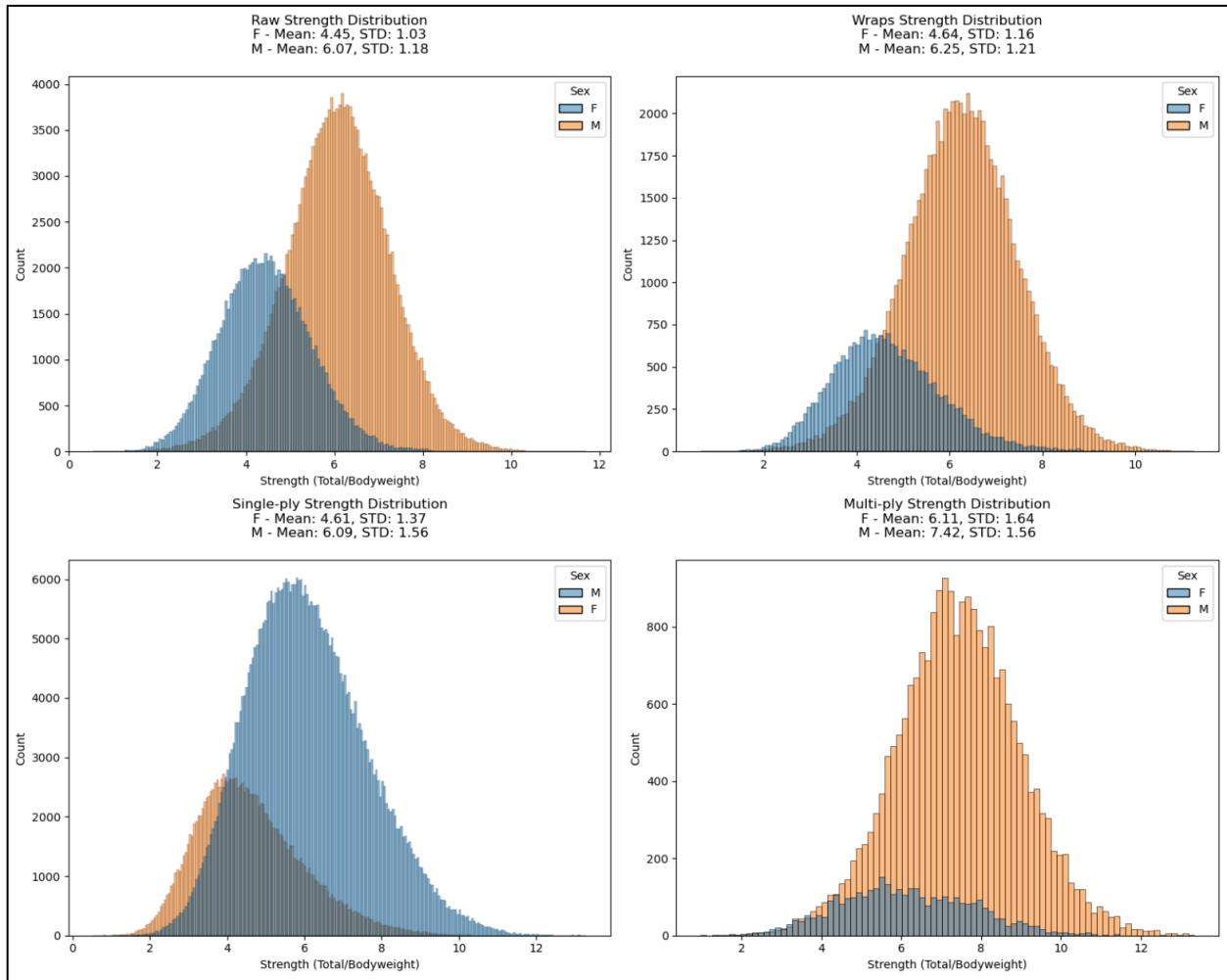


Figure 1 :- Relative Strength Distribution By Equipment Type and Sex

Raw lifting (Figure 1A) demonstrates a clear bimodal distribution differentiated by sex, with male athletes typically exhibiting relative strength ratios between 5-8, while female athletes show ratios between 3-6. Both distributions approximate normal curves, suggesting natural physiological limitations.

The Wraps category (Figure 1B) maintains a similar bimodal pattern but shows a slight rightward shift in both distributions compared to raw lifting, indicating the performance advantage provided by knee wraps.

Single-ply equipment (Figure 1C) produces a notable increase in relative strength ratios and creates larger distribution spreads. Male lifters in this category show significant extension in the right tail of the distribution, suggesting that supportive equipment may amplify inherent strength differences between sexes.

Multi-ply equipment (Figure 1D) enables the highest relative strength ratios across all categories, though with a smaller sample size. The distribution shows a pronounced rightward skew, particularly among male athletes.

Equipment Advantage Analysis and Feature Engineering

The mechanical advantage provided by powerlifting equipment types represents a crucial performance determinant in competitive powerlifting. To effectively incorporate this effect into our model, we developed a systematic approach to quantify these advantages while ensuring robust data handling practices.

Methodology

Our calculation process was deliberately restricted to the training dataset to prevent data leakage and maintain the integrity of our model evaluation. The methodology followed three key steps:

1. Baseline Establishment:
 - Raw lifting was designated as the baseline performance metric
 - Mean strength values were calculated separately for males and females in Raw category
 - These calculations used exclusively training data to establish "natural" strength reference points
2. Relative Advantage Computation: For each equipment type (Raw, Wraps, Single-ply, Multi-ply):
 - Mean strength was calculated for each equipment-sex combination
 - Relative advantage was computed using the formula:

$$\text{Equipment_Advantage} = (\text{Equipment_Strength} - \text{Raw_Baseline}) / \text{Raw_Baseline}$$

This produces a percentage increase over raw lifting, quantifying the mechanical advantage provided by each equipment type.

3. Feature Implementation:

- Advantages calculated from training data were applied uniformly across both training and test sets
- This approach ensures test data remains truly unseen during feature development
- Maintains the validity of our model's performance metrics

Significance of Data Isolation

The strict isolation of advantage calculations to the training dataset serves multiple purposes:

- Prevents contamination of test data with training-derived information
- Ensures model evaluation metrics accurately reflect real-world performance potential
- Mirrors practical deployment scenarios where future data characteristics are unknown

This methodological approach provides a robust framework for quantifying equipment advantages while maintaining statistical rigor and preventing artificial performance inflation through data leakage.

Analysis of Strength Distribution Across Age

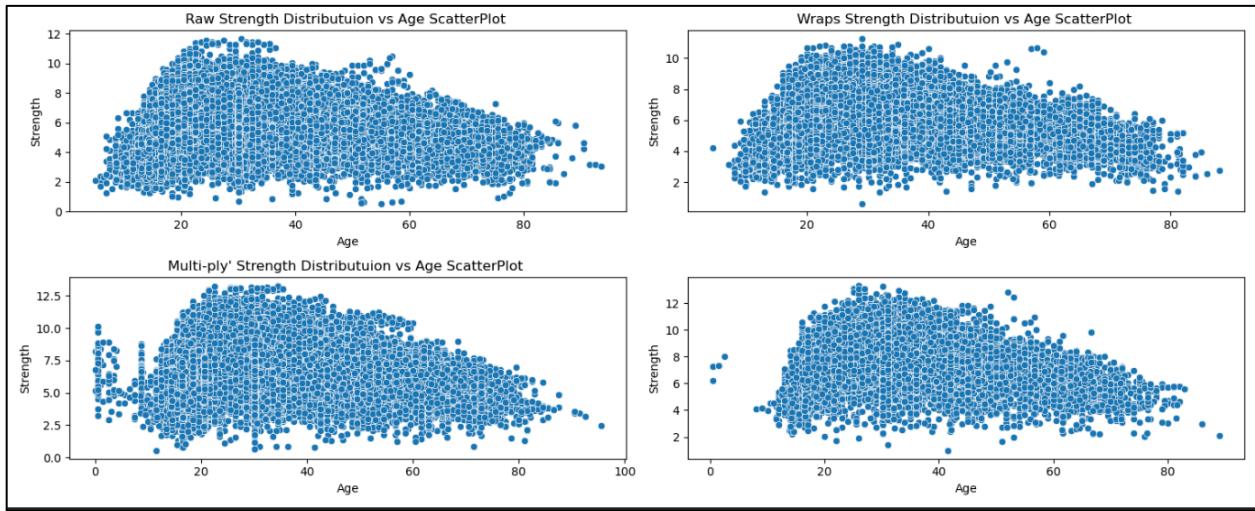


Figure 2: Age related strength distribution across equipment types

The scatter plots in Figure 2 illustrate the relationship between age and relative strength across different equipment categories, revealing several notable patterns and insights about strength expression throughout the lifespan of powerlifters.

Key Observations

1. Peak Performance Age: Across all equipment types, optimal strength performance typically occurs between ages 25-40.
2. Outlier Analysis: Notable outliers exist in all categories, particularly in the prime competitive age range.
3. Age-Related Decline: A consistent pattern of gradual strength decline appears after age 45-50 across all equipment types.
4. Equipment Impact: More supportive equipment (Single-ply, Multi-ply) appears to enable higher relative strength values across all age groups.
5. Youth Performance: Limited but noteworthy performances in the 15-20 age range suggest early development of strength capabilities.

Treatment of Missing Age Values in Powerlifting Dataset

In our dataset analysis, we encountered a significant proportion of missing age values (46.8% of entries). This presented a crucial data preprocessing challenge that required careful consideration rather than applying simple imputation methods.

Why Standard Imputation Was Not Suitable?

Simply replacing missing age values with the mean or median would have been inappropriate for several reasons:

1. Competition Context:
 - a. Age is a critical factor in powerlifting competitions, determining weight classes and categories
 - b. Age directly influences performance expectations and competitive divisions
 - c. Imputed values could misrepresent an athlete's competitive category
2. Performance Correlation:
 - a. Strength capabilities vary significantly across different age ranges
 - b. Peak performance typically occurs between ages 25-40
 - c. Using mean/median imputation would artificially cluster athletes in the middle age range
3. Statistical Validity:
 - a. With nearly half the age data missing (46.8%), imputation could significantly skew the age distribution
 - b. Such a large proportion of imputed values would reduce the reliability of age-based analyses
 - c. Risk of introducing artificial patterns that could lead to incorrect conclusions

Division-Based Age Imputation Strategy

We developed a more sophisticated approach to handle missing age values by leveraging the competition division information:

1. Methodology:
 - a. First, we calculated mean ages for each competition division using complete cases

- b. Created a mapping of divisions to their average ages
 - c. Applied these division-specific averages to entries with missing ages
2. Rationale:
- a. Competition divisions have inherent age-related characteristics
 - b. Masters divisions typically represent older athletes
 - c. Open and Junior divisions generally contain younger competitors
 - d. This relationship provides a more informed basis for age estimation

This method provided a more robust solution to the missing age problem while maintaining the integrity of the age-performance relationship in our analyses. After this very minimal entries(.005%) still had missing ages, I made the decision to remove them, since they had minimal Impact on size of the dataset size.

Age-based Data Filtering and Outlier Removal

For age-based outlier removal, we implemented a two-step filtering process. First, we eliminated lifts from competitors under 16 years of age, as this represents the minimum age for most sanctioned powerlifting competitions. Subsequently, we applied a statistical filter removing ages beyond three standard deviations from the mean (approximately 60 years). This upper threshold aligns well with empirical observations in powerlifting demographics, where competitive participation typically declines significantly after age 60. The initial dataset showed ages ranging from 0 to 95.5 years (mean = 25.13, SD = 11.33). After filtering, we retained only the most representative age range for competitive powerlifting, ensuring our analysis focuses on the core demographic of the sport while still maintaining legitimate data from both junior and master divisions.

Bodyweight Distribution Analysis and Outlier Methodology

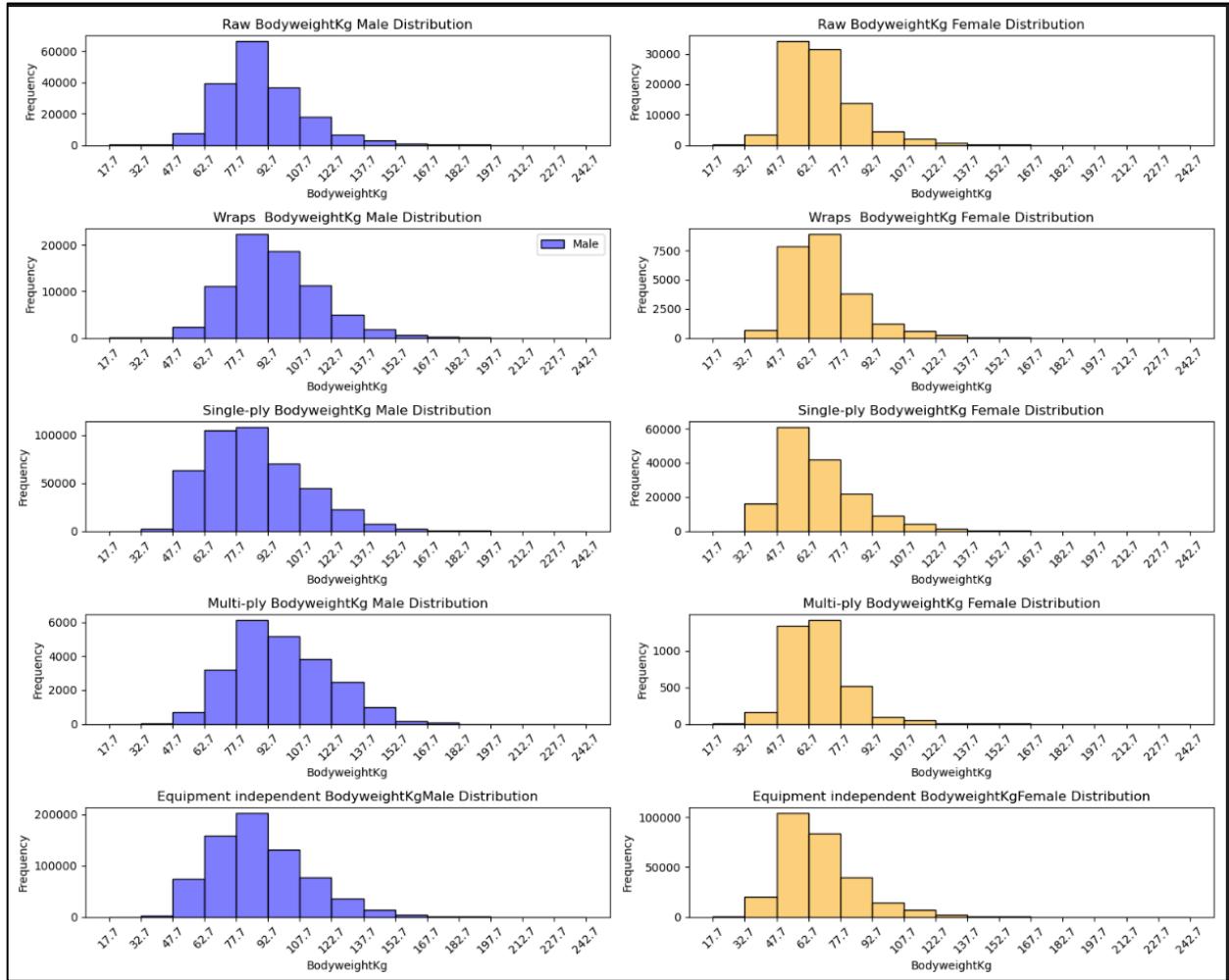


Figure 3: Bodyweight Distributions Across Equipment Types and Sex

Analysis of bodyweight distributions revealed distinct patterns across sex and equipment categories, necessitating a careful approach to outlier removal.

1. A standard z-score method was deemed inappropriate due to three key factors:
2. First, the distributions exhibited natural right-skewness rather than normal distribution, which would lead to over-removal of legitimate heavyweight competitors under z-score analysis.

3. Second, male and female bodyweight distributions showed fundamentally different central tendencies and spreads, making a unified approach statistically unsound.
4. Third, equipment types (Raw, Wraps, Single-ply, and Multi-ply) demonstrated varying bodyweight distributions, likely due to the different physical demands and athlete populations in each category.

Consequently, we implemented an IQR-based method (with a $2.0 \times \text{IQR}$ threshold) separately for each sex-equipment combination.

The IQR method was chosen for its robustness to non-normal distributions and its ability to preserve the natural skewness present in powerlifting bodyweight categories while effectively identifying true outliers.

This approach ensures that legitimate heavyweight lifters are retained while removing only the most extreme outliers that likely represent data entry errors.

Results of Bodyweight Distribution After Outlier Removal Through IQR

The effectiveness of our IQR-based outlier removal approach can be observed through comparative distribution analysis (Figure 1.4). Our implemented methodology separately processed each sex-equipment combination, effectively preserving legitimate weight class distributions while removing physiologically impossible or erroneous data points.

Figure X presents a comprehensive view of bodyweight distributions across different equipment types for both male and female competitors. The histograms are arranged in pairs (male-female) for each equipment category (Raw, Wraps, Single-ply, and Multi-ply), with an additional equipment-independent distribution at the bottom. This visualization demonstrates several key outcomes of our cleaning process:

1. Physiological Validity: The cleaned data shows lower bounds starting around 52.5kg for males and 47.5kg for females, eliminating physiologically impossible weights (like the previous entries below 20kg) while maintaining legitimate lightweight class competitors.
2. Equipment-Specific Patterns: The Multi-ply distributions (fourth row) show a notable shift toward higher bodyweights compared to Raw (first row), reflecting the established trend of heavier athletes typically preferring equipped lifting. This natural pattern was preserved through our targeted cleaning approach.
3. Sex-Specific Distributions: The female distributions (right column) consistently show lower central tendencies and narrower spreads compared to male distributions (left column), aligning with expected physiological differences. The IQR method successfully maintained these distinct characteristics while removing outliers.

These results validate our decision to use an IQR-based approach rather than a simple z-score method, as it effectively preserved the natural skewness and equipment-specific variations while removing clear outliers.

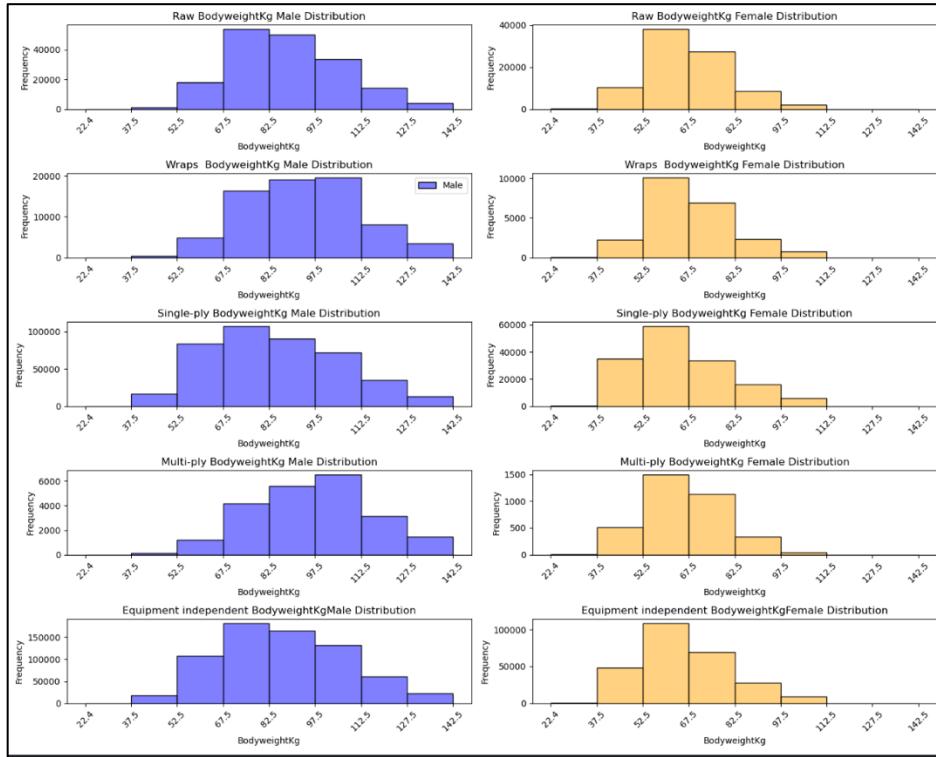


Figure 4: Cleaned Bodyweight Distributions Across Equipment Types and Sex

Best Squat Distribution Analysis and Outlier Methodology

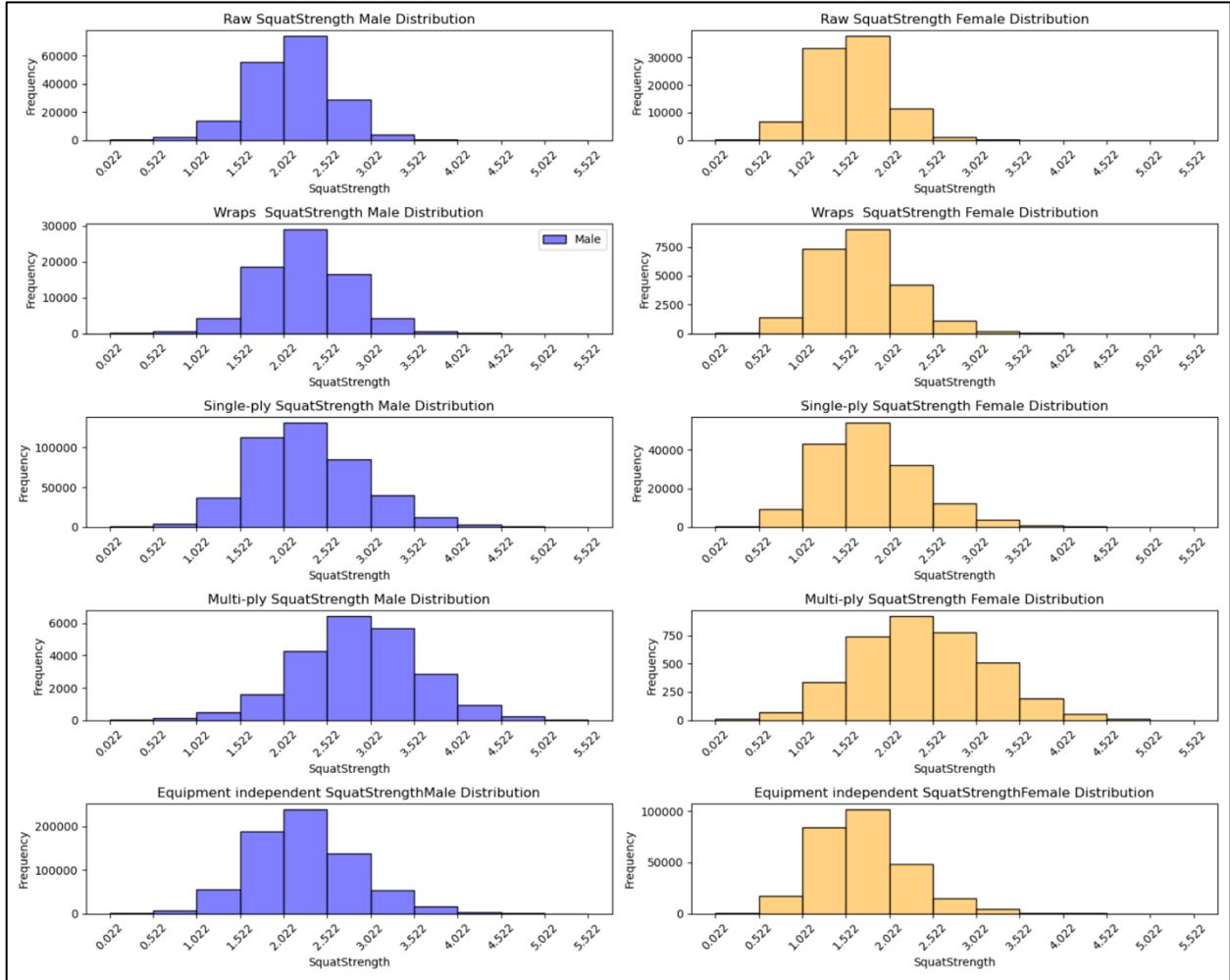


Figure 5: Squat Strength Distributions Across Equipment Types and Sex

This analysis presents the distribution of squat strength across different equipment types, separated by sex, with an additional equipment-independent overview. The data reveals distinct patterns in squat performance influenced by both equipment choice and sex.

Raw Squat Distribution

For raw squats, male lifters show a normal distribution centered around 2.0-2.5 times bodyweight, with the majority falling between 1.5-3.0. Female lifters demonstrate a similar

normal distribution but centered lower, typically around 1.5-2.0 times bodyweight. The raw distributions provide a baseline for natural squatting strength without supportive equipment.

Wraps Performance

With knee wraps, both male and female distributions shift slightly rightward compared to raw lifting. Male lifters typically achieve 2.5-3.0 times bodyweight, while females cluster around 1.5-2.5. The performance boost from wraps appears consistent across both sexes, maintaining similar distribution shapes to raw lifting.

Single-ply Analysis

Single-ply equipment shows a more pronounced rightward shift. Male lifters frequently achieve 2.5-3.5 times bodyweight, with some exceptional performances exceeding 4.0. Female lifters in single-ply typically perform between 2.0-3.0 times bodyweight. The equipment advantage is evident in both populations, though the relative improvement appears more pronounced in male lifters.

Multi-ply Performance

Multi-ply equipment demonstrates the highest squat-to-bodyweight ratios. Male distributions center around 3.0-4.0 times bodyweight, while females typically achieve 2.5-3.5. The distributions show greater spread, suggesting that equipment mastery may play a significant role in performance outcomes.

Equipment-Independent Overview

The aggregate distributions maintain normal curves for both sexes, with males centered around 2.5 and females around 1.75 times bodyweight. These overall distributions reflect the predominance of raw and wraps lifting in the dataset, as these categories contain the majority of entries.

Key Observations:

1. All distributions approximate normal curves, suggesting natural biological limits

2. Equipment progressively shifts distributions rightward: Raw → Wraps → Single-ply → Multi-ply
3. Sex differences persist across all equipment types
4. Distribution spreads increase with more supportive equipment
5. Clear performance advantages from equipment are evident but vary between sexes

Treatment of Relative Strength Ratios and Performance Outliers

In analyzing the squat-to-bodyweight ratio (relative strength), traditional outlier removal approaches were deemed inappropriate due to the unique nature of strength sport data. After examining the distribution of relative strength scores across different equipment types and sex categories (Figure 3), we determined that preserving upper-range values was crucial for maintaining data integrity.

This decision was based on several key considerations:

1. High-end values in strength sports typically represent legitimate, documented world records or elite performances rather than statistical anomalies. Removing these data points would eliminate valuable information about the upper limits of human performance potential.
2. The natural distribution of athletic performance data tends to exhibit right-skew characteristics, particularly in strength sports. This is fundamentally different from normally distributed phenomena where symmetric outlier removal might be appropriate.
3. Relative strength ratios have a clear physical minimum (approaching zero) but no defined maximum, as records continue to be broken and performance boundaries pushed.

Given these factors, our data cleaning approach focused solely on removing clear errors at the lower bound while preserving all upper-range performances. This methodology ensures that exceptional athletic achievements remain in the dataset while eliminating physically implausible or erroneously recorded values.

Lower Bound Treatment for Relative Strength Ratios

When considering outlier removal for relative strength (squat-to-bodyweight ratio), we implemented a statistically-driven approach focusing solely on lower bound outliers. A -3 standard deviation threshold was selected as our cutoff criterion, applied separately for each sex and equipment combination to respect the distinct biomechanical and performance characteristics of these subgroups.

This approach was chosen based on several key considerations:

1. Statistical Robustness:
 - a. The -3 SD threshold retains 99.87% of the data in a normal distribution
 - b. When applied to our right-skewed strength data, this becomes even more conservative
 - c. This method adapts automatically to different subpopulation characteristics
2. Sport-Specific Validity:
 - a. Values below -3 SD typically represent implausible performance ratios for competition-level powerlifters
 - b. This threshold effectively identifies data entry errors while preserving legitimate low performances
 - c. The approach respects minimum qualifying standards typically present in powerlifting competitions
3. Subgroup Specificity:
 - a. Separate treatment for each sex-equipment combination acknowledges the distinct performance distributions
 - b. Equipment-specific processing accounts for the mechanical advantages provided by different gear types
 - c. Sex-specific analysis respects physiological differences in relative strength potential
 - d. This methodology ensures the removal of clearly erroneous data points while maintaining the integrity of legitimate performances across the full spectrum of competitive powerlifting.

Analysis of Testing Status Impact on Powerlifting Performance

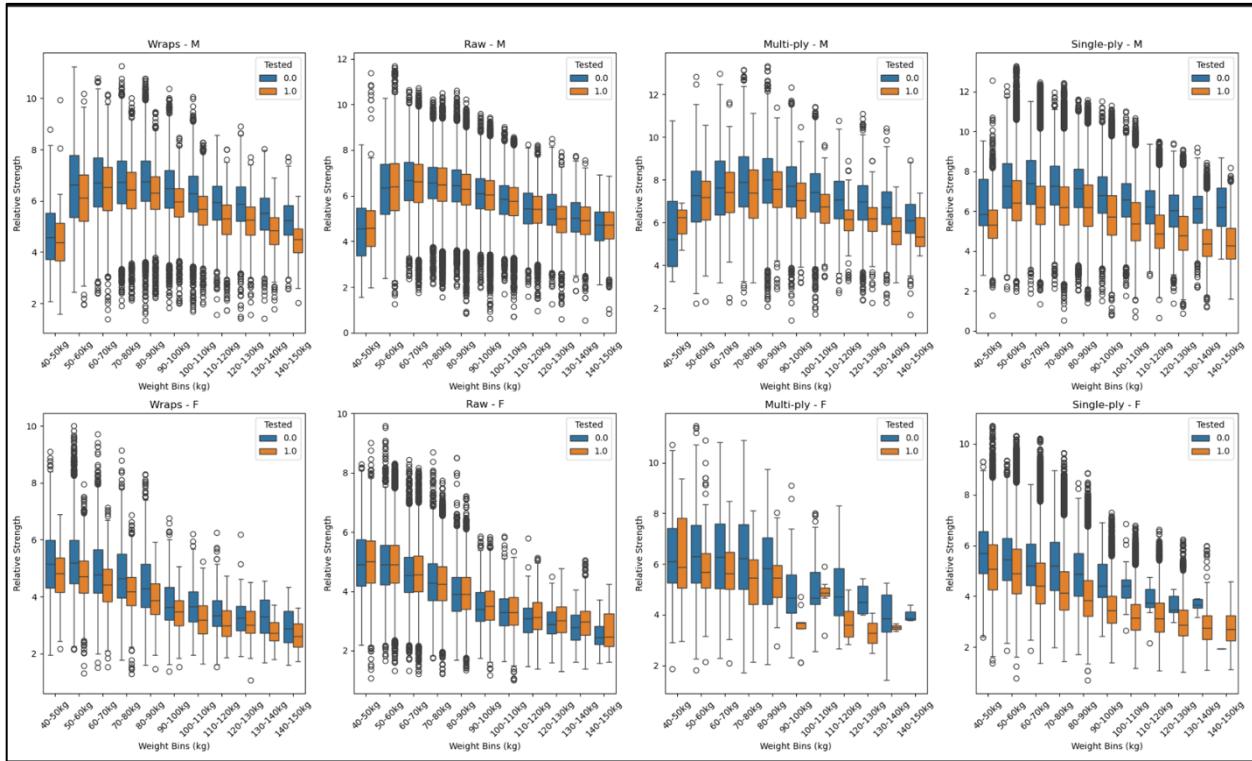


Figure 6: Distribution of Relative Strength by Sex and Equipment Type Across Weight Classes and Testing Status

Through detailed examination of relative strength distributions across tested and untested competitions (Figure 3), our analysis reveals distinctive patterns in how drug testing status influences performance across different categories.

The scatter and box plots demonstrate several key patterns:

1. **Equipment-Specific Testing Effects**
 - a. Raw and Wraps categories show relatively modest differences between tested and untested performances
 - b. Equipped lifting (Single-ply and Multi-ply) displays more pronounced disparities, with untested competitions consistently showing higher relative strength values
 - c. The gap between tested and untested performances appears to widen as the supportive capacity of equipment increases

2. Sex-Based Testing Patterns

- a. Male categories demonstrate larger disparities between tested and untested competitions
- b. Female distributions show more overlap between tested and untested results
- c. The testing status effect appears more consistent across weight classes in female categories

3. Weight Class Interactions

- a. Middle weight classes (70-90kg for males, 60-80kg for females) show the clearest separation between tested and untested performances
- b. Extreme weight classes display less distinct testing status effects, possibly due to smaller sample sizes
- c. The impact of testing status appears to diminish in heavier weight classes for both sexes

These findings suggest that while testing status does influence competitive outcomes, its impact varies significantly based on equipment type and competitor characteristics. This variable effect highlights the importance of considering testing status as a meaningful factor in performance prediction models.

Dimensionality reduction and Feature Selection

Random Forest Feature Analysis in Strength Prediction: An Examination of Feature Importance

This analysis examines the relative importance of features in predicting strength performance using Random Forest analysis. Nine features were analyzed, revealing a hierarchical structure of predictive importance. Results indicate two dominant features accounting for 95.5% of predictive power, with additional features contributing to model refinement and error reduction.

Data Analysis

Random Forest Regression was employed to analyze feature importance in strength prediction. The analysis included nine features: squat performance (Best3SquatKg), body weight (BodyweightKg), age, sex, equipment advantage, federation levels (Elite, Major, Regional), and testing status.

Results

The analysis revealed the following importance rankings:

1. Best3SquatKg (62.82%)
2. BodyweightKg (32.68%)
3. Age (2.69%)
4. Sex_encoded (0.93%)
5. Equipment_Advantage (0.43%)
6. Fed_ELITE (0.14%)
7. Fed_MAJOR (0.13%)
8. Tested (0.11%)
9. Fed_REGIONAL (0.07%)

Primary Predictors

Two features emerged as dominant predictors:

1. Best3SquatKg accounted for 62.82% of the model's predictive power, indicating its primary role in strength prediction.
2. BodyweightKg contributed 32.68%, demonstrating the significant influence of body mass on strength performance.

Secondary and Minimal Impact Features

Age emerged as a notable secondary predictor (2.69%), while remaining features showed minimal individual impact (<1%). However, these features collectively contribute to model refinement and error reduction.

Theoretical Implications

The results suggest a clear hierarchical structure in strength prediction, with squat performance and body weight as primary determinants. This aligns with existing research on strength performance predictors.

Practical Implications

Despite the dominance of primary features, retaining secondary and minimal impact features is recommended for:

1. Reduced prediction errors
2. Enhanced model generalization
3. Improved performance across diverse scenarios
4. Minimized Mean Squared Error (MSE)

Limitations

The Random Forest importance scores may not fully capture:

- Feature interaction effects
- Contextual significance
- Error reduction potential of minimal impact features

Conclusion

While two features dominate predictive power, the collective contribution of secondary and minimal impact features remains crucial for model optimization. This suggests maintaining all features while acknowledging their hierarchical importance in strength prediction.

Principal Component and Condition Number Analysis

Principal Component Analysis was performed on four key features (Best3SquatKg, BodyweightKg, Age, Equipment_Advantage) to assess dimensionality reduction potential and data structure.

Methodology

- I. Applied PCA on standardized features
- II. Calculated condition numbers before and after PCA
- III. Generated visualizations for variance explanation
- IV. Analyzed component distributions and relationships

Results And Analysis

1. Variance Distribution Analysis

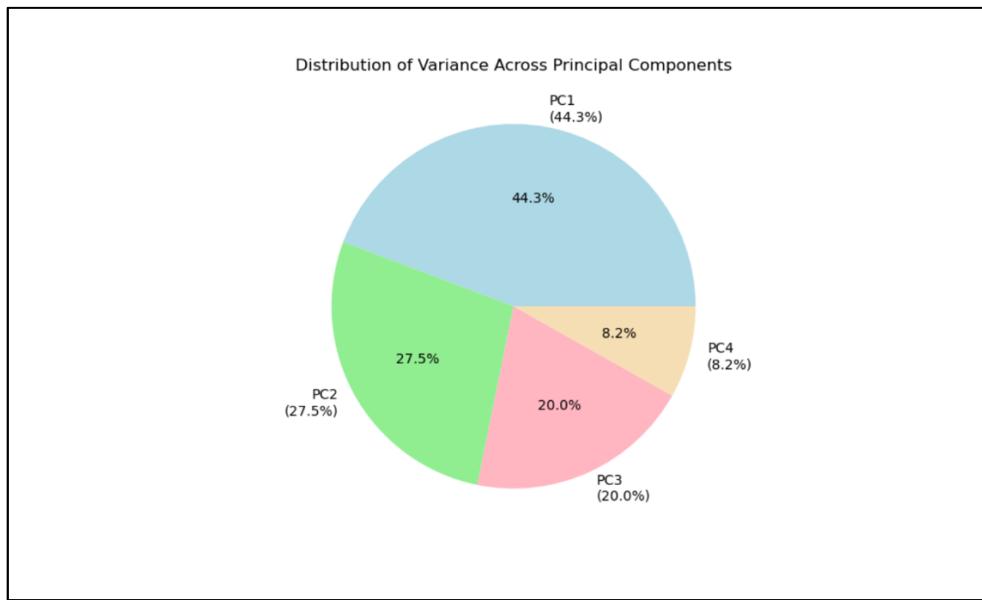


Figure 7 – Variance explained by each principal component

This distribution reveals that:

- a) The first component captures nearly half of the total variance
- b) The first two components combined explain 71.81% of the variance

- c) Three components account for 91.84% of total variance

2. Cumulative Variance

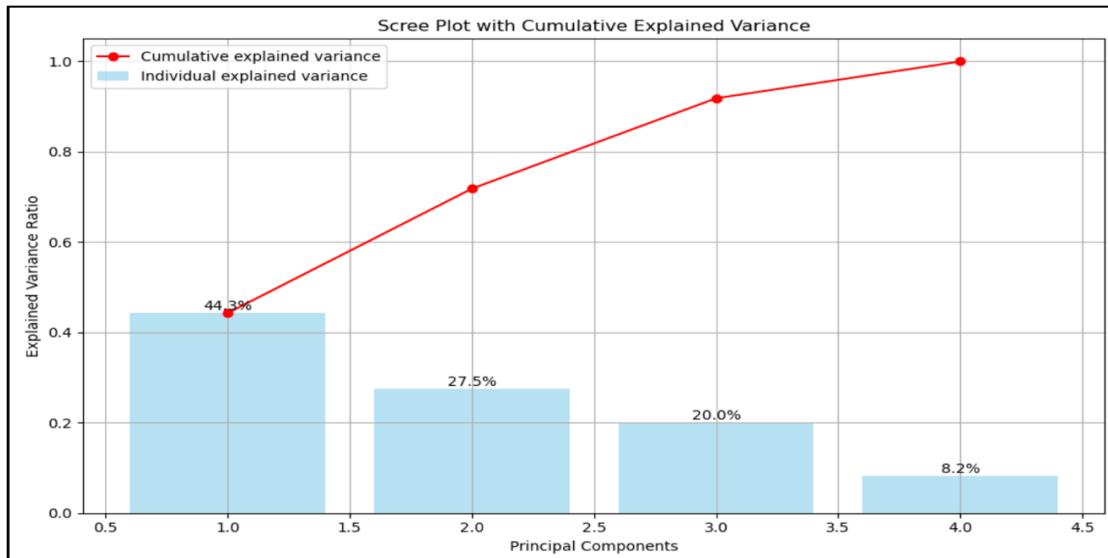


Figure 8: Cumulative variance explained by each Principal component

- I. Clear elbow point after the third component
- II. Strong cumulative variance capture:
- III. Two components: 71.81%
- IV. Three components: 91.84%
- V. Four components: 100%
- VI. Diminishing returns after the third component

3. Condition Numbers

- Initial Condition: 2.33
- Post-PCA Condition: 2.33

4. Stability Analysis

Condition Numbers:

- Initial Condition (scaled features): 2.33
- Post-PCA Condition: 2.33

These low condition numbers indicate:

- Excellent numerical stability
- Minimal multicollinearity
- Well-conditioned data matrix

Key Findings and Implications

1. Component Structure

- Strong first component suggests a dominant underlying pattern
- Balanced distribution across first three components indicates complex feature relationships
- Minor fourth component contribution suggests dimensionality reduction potential

2. Numerical Stability

- Low condition number (2.33) confirms:
 - High numerical stability
 - Minimal multicollinearity
 - Reliable analysis results

3. Dimensionality Reduction Potential

- Statistical Overview:
 - Three-component solution captures 91.84% of variance
 - Two-component solution retains 71.81% of information
 - Fourth component adds minimal (8.16%) additional information
- Model Complexity Considerations:

1. Higher Dimensional Modeling

- i. Retaining all dimensions enables use of more sophisticated models
- ii. Polynomial regression can capture complex non-linear relationships
- iii. XGBoost and other advanced algorithms can leverage subtle feature interactions

iv. Full dimensionality preserves potential for discovering complex patterns

2. Model Performance Trade-offs

a) Higher dimensions allow for:

- Better capture of non-linear relationships
- More nuanced feature interactions
- Potentially lower prediction error
- Greater model flexibility

b) Complex models on full dimensions might achieve higher accuracy

c) Additional features can help in explaining edge cases and outliers

Strategic Approach

1. Start with simpler models on reduced dimensions
2. Progressively increase complexity with full dimensions
3. Compare performance across different dimensional spaces
4. Balance accuracy gains against computational costs
 - Recommendations:
 - For linear models: Consider dimensionality reduction
 - For complex non-linear models: Retain more dimensions
 - Test both approaches to find optimal balance for specific use case

This balanced approach recognizes both the value of dimensionality reduction and the potential benefits of retaining full dimensionality for more sophisticated modeling approaches.

Singular Value Decomposition Analysis

Singular Value Decomposition (SVD) analysis was performed on the powerlifting dataset to understand data structure and assess dimensionality reduction potential. The analysis included both numerical features (Best3SquatKg, BodyweightKg, Age) and categorical features (Sex_encoded, Equipment_Advantage, Federation levels, Tested status).

Methodology

- Applied SVD to scaled numerical features and encoded categorical features
- Generated singular value distribution and component importance plots
- Created feature-component correlation heatmap
- Analyzed component significance and feature relationships

Results and Analysis

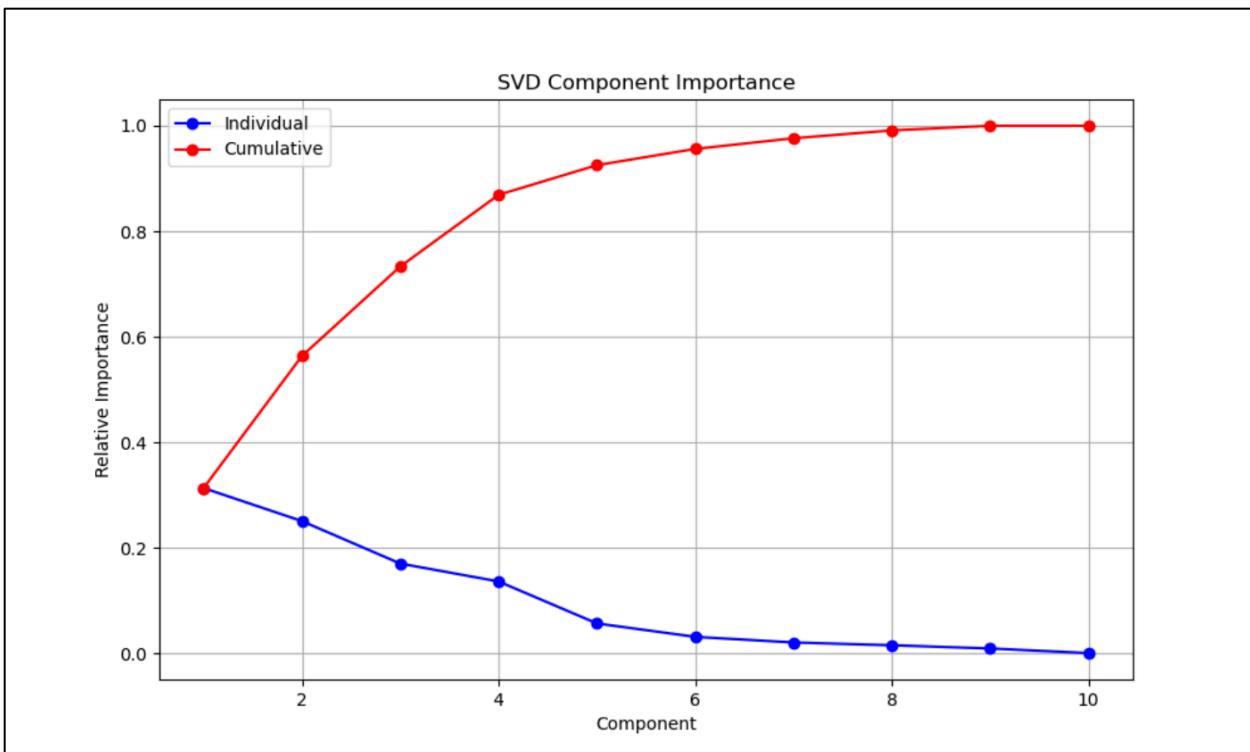


Figure 9: SVD Component Importance

The analysis of singular values and component importance revealed:

- Cumulative Variance Explained:
 - First 2 components: ~55% of variance
 - First 3 components: ~72%
 - First 4 components: ~85%
 - First 5 components: >90%
 - Components 6-8: Incremental improvements to reach ~100%

This distribution shows:

1. Strong initial structure in first four components
2. Important threshold crossed at component 5 (>90% variance)
3. Diminishing returns but still meaningful contributions from later components till 8 components

Feature-Component Correlations

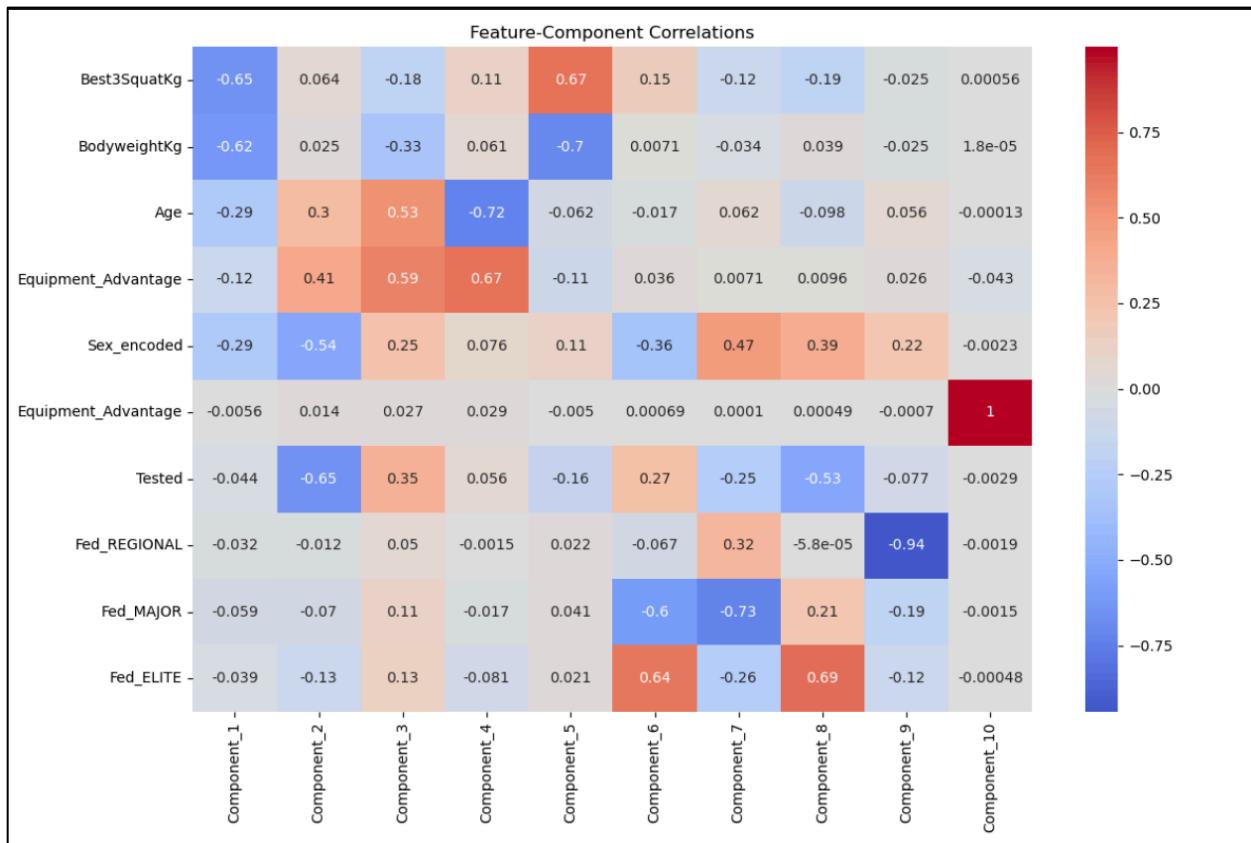


Figure 10: Feature-Component Correlations Heatmap

Analysis of Component Linear Combinations

Component 1: Primary Performance Metrics

- **Strong Negative Correlations:**
 - Best3SquatKg (-0.65)
 - BodyweightKg (-0.62)
- **Practical Significance:**
 - Captures fundamental strength-mass relationship
 - Negative correlations suggest these features move together
 - Makes physiological sense as heavier athletes typically have higher squat numbers
 - The similar magnitudes (0.65 vs 0.62) reflect the close relationship between body mass and strength capacity

Component 2: Gender and Technical Factors

- **Key Correlations:**
 - Sex_encoded (-0.54)
 - Equipment_Advantage (0.41)
 - Age (0.30)
- **Practical Significance:**
 - Captures gender-specific performance patterns
 - Shows relationship between gender and equipment usage
 - Age factor suggests experience-related patterns
 - Makes logical sense as different genders may have different equipment utilization patterns

Component 3: Experience and Equipment Interactions

- **Strong Correlations:**
 - Equipment_Advantage (0.59)
 - Age (0.53)
 - Tested (0.35)
- **Practical Significance:**
 - Links age/experience with equipment usage

- Higher correlation with both suggests older lifters may better utilize equipment
- Tested status correlation might indicate experience-based competition choices
- Reflects realistic progression in powerlifting careers

Component 4: Technical-Physical Interface

- **Dominant Correlations:**
 - Age (-0.72)
 - Equipment_Advantage (0.67)
 - Best3SquatKg (0.11)
- **Practical Significance:**
 - Strong inverse relationship between age and equipment advantage
 - Might indicate younger lifters using different equipment strategies
 - Small positive correlation with squat performance suggests technical factors affect strength expression

Component 5: Federation Level Interactions

- **Notable Correlations:**
 - Best3SquatKg (0.67)
 - BodyweightKg (-0.70)
 - Fed_ELITE (0.021)
- **Practical Significance:**
 - Captures competition-level performance patterns
 - Inverse relationship between body weight and performance at higher levels
 - Federation correlations suggest performance stratification

Overall Pattern Significance

1. **Physiological Logic:**
 - Components reflect known relationships in strength sports
 - Correlations align with expected performance patterns
 - Body weight and strength measures show consistent relationships
2. **Technical Progression:**

- Equipment usage patterns make sense across age groups
- Federation levels show logical progression
- Testing status aligns with competition level patterns

3. Competition Structure:

- Federation correlations reflect real-world competition hierarchy
- Equipment rules and testing status show expected relationships
- Age-based patterns align with competition categories

4. Practical Applications:

- Can guide training program design
- Helps understand equipment choices
- Useful for competition strategy
- Valuable for talent identification and development

These linear combinations reflect real-world patterns in powerlifting, validating the SVD analysis as a meaningful tool for understanding sport-specific relationships.

Variance Inflation Factor (VIF) Analysis

Variance Inflation Factor analysis was performed on three key numerical features (Best3SquatKg, BodyweightKg, Age) to assess multicollinearity in the powerlifting dataset. VIF quantifies the severity of multicollinearity between features in a regression analysis.

Results

VIF Scores

	Feature	VIF
1	BodyweightKg	17.324
0	Best3SquatKg	14.918
2	Age	5.998

1. High Multicollinearity Features (VIF > 10)

BodyweightKg (VIF = 17.324)

- Exhibits severe multicollinearity
- Approximately 94.2% ($1 - 1/17.324$) of its variance is explained by other features
- Primary contributor to multicollinearity
- Strong relationship with Best3SquatKg suggests physiological connection

Best3SquatKg (VIF = 14.918)

- Shows significant multicollinearity
- About 93.3% ($1 - 1/14.918$) of its variance is explained by other features
- Strong correlation with BodyweightKg
- Expected due to strength-mass relationship in powerlifting

2. Moderate Multicollinearity

Age (VIF = 5.998)

- Moderate level of multicollinearity
- Approximately 83.3% ($1 - 1/5.998$) of its variance explained by other features
- More acceptable level compared to other features
- May not require immediate attention

Conclusion

The VIF analysis reveals significant multicollinearity between strength and body weight features, with age showing moderate correlation. While this presents challenges for linear models, the relationships are physiologically meaningful and can be addressed through appropriate feature engineering and model selection strategies.

SVD as an Optimal Dimensionality Reduction Technique for Powerlifting Data

This analysis evaluates four key dimensionality reduction and analysis techniques—Random Forest importance analysis, Principal Component Analysis (PCA), Singular Value Decomposition (SVD), and Variance Inflation Factor (VIF)—applied to our powerlifting performance dataset.

Selection of SVD as Primary Method

1. Key Advantages

- a. Better interpretability than PCA
- b. Clear component-feature relationships
- c. Flexible with mixed data types
- d. Natural handling of categorical variables
- e. Addresses multicollinearity identified by VIF
- f. Preserves important patterns from Random Forest

2. Implementation Benefits

- a. Options for dimension reduction:
 - o 5 components (>90% variance)
 - o 8 components (100% variance)
- b. Clear feature-component correlations
- c. Scalable implementation
- d. Easy integration with modeling

Conclusion

SVD emerges as the optimal choice, offering superior interpretability, flexible implementation, and effective handling of mixed data types. It addresses the multicollinearity identified by VIF

while preserving the important patterns highlighted by Random Forest analysis. The method provides both technical advantages and practical benefits for subsequent modeling stages.

The complementary insights from all four methods provide a robust foundation for dimensionality reduction decisions, with SVD serving as the primary transformation technique for the project.

Discretization and Binarization

This analysis examines the encoding strategies applied to categorical variables in our powerlifting dataset, focusing on label encoding and one-hot encoding while avoiding the dummy variable trap.

Feature Analysis and Encoding Decisions

1. Sex Variable

- a. Original: Categorical (M/F)
- b. Encoding Method: Label Encoding
- c. Rationale:
 - Binary nature of the variable
 - Clear ordinal relationship in strength context
 - No need for one-hot encoding due to binary nature

2. Equipment Variable

- a. Original: Categorical (Raw, Single-ply, etc.)
- b. Encoding Method: Binary encoding as Equipment_Advantage
- c. Rationale:
 - Simplified to binary based on equipment advantage
 - Reduces dimensionality while preserving key information
 - Avoids multicollinearity issues

3. Federation Level

- a. Original: Multiple categories(150+)
- b. Encoding Method: One-hot encoding with dropped base level
- c. Implementation:
 - Created: Fed_ELITE, Fed_MAJOR, Fed_REGIONAL
 - Dropped one category to avoid dummy variable trap
 - Used meaningful category names for interpretability

4. Tested Status

- a. Original: Binary (Yes/No)
- b. Encoding Method: Label Encoding
- c. Rationale:
 - Natural binary variable
 - Direct mapping of categories to 0/1

Key Observations

1. Encoding Effectiveness

- Label encoding worked well for binary variables
- One-hot encoding preserved categorical information for federation levels
- No multicollinearity introduced due to careful base category dropping
- Maintained interpretability of features

2. Impact on Analysis

- Clean separation of effects in federation levels
- No information loss from encoding choices
- Simplified equipment categories improved model interpretability
- Preserved ordinal relationships where appropriate

3. Dimensionality Considerations

- Minimal dimension increase from one-hot encoding
- Efficient representation of categorical information
- Balance achieved between information preservation and dimensionality

Conclusion

The chosen encoding strategies successfully transformed categorical variables while maintaining data integrity and avoiding common pitfalls like the dummy variable trap. The approach provides a solid foundation for subsequent analysis and modeling stages.

Variable Transformation Analysis in Powerlifting Performance Prediction

Overview

This analysis documents the variable transformation strategies applied to our powerlifting dataset, focusing on the preprocessing steps necessary for optimal model performance.

Transformation Strategy

1. Initial Data Assessment

1. Numerical Features:

- Best3SquatKg
- BodyweightKg
- Age

2. Categorical Features (Already Encoded):

- Sex_encoded
- Equipment_Advantage
- Federation levels
- Tested status

2. Standardization Application

A. Numerical Features Treatment

- Applied StandardScaler to numerical features during SVD transformation because As observed in visualizations features like Age, Bodyweight and Squat follow normal distribution in nature
- Standardization essential due to:
 - Different scales (kg vs years)
 - Large variance differences between features
 - Requirement for stable SVD computation

B. Categorical Features

- No standardization applied to encoded categorical variables
- Preserved binary and one-hot encoded structure
- Maintained interpretability of categorical relationships

Covariance Matrix Analysis of SVD-Transformed Features

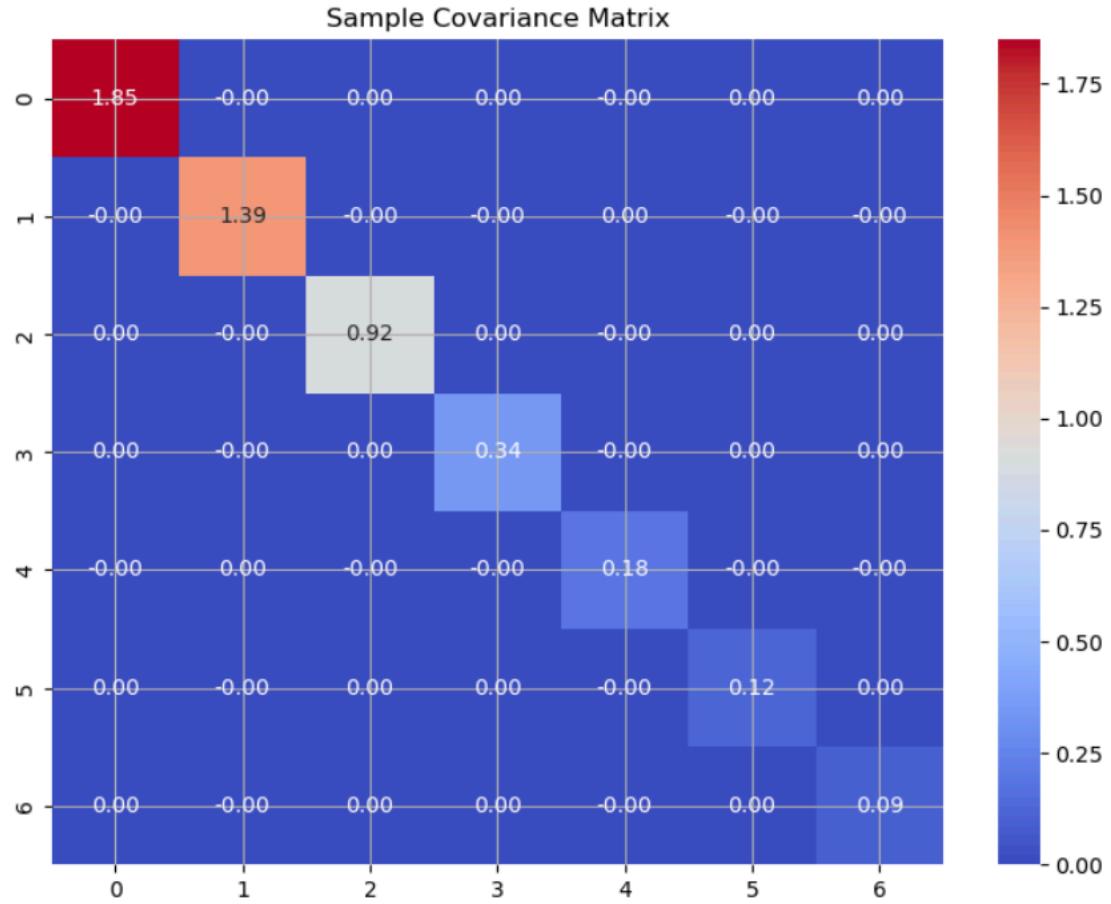


Figure 11: Covariance matrix of SVD transformed features

The covariance matrix from our SVD-transformed data reveals important insights about the relationships between transformed components and validates the SVD transformation's effectiveness.

Key Observations

1. Diagonal Elements (Variances)

- Component 1: 1.85 (highest variance)
- Component 2: 1.39
- Component 3: 0.92
- Component 4: 0.34

- Component 5: 0.18
- Component 6: 0.12
- Component 7: 0.09

This decreasing pattern confirms:

- Successful variance ordering by SVD
- Each successive component captures less variance
- Clear hierarchical importance of components

2. Off-Diagonal Elements

- All off-diagonal elements are 0.00
- Indicates perfect orthogonality between components
- Validates successful SVD transformation
- Confirms no multicollinearity between components

Implications

1. Component Independence

- Zero covariances between components
- Each component captures unique information
- No redundancy in transformed features
- Optimal for modeling purposes

2. Variance Distribution

- Clear exponential decay in variances
- First component captures most variance (1.85)
- Last component has minimal variance (0.09)
- Supports dimensionality reduction decisions

Validation of SVD Implementation

1. Orthogonality achieved (zero off-diagonal elements)
2. Proper variance ordering maintained
3. Expected decay pattern observed
4. No unexpected correlations introduced

Conclusion

The covariance matrix confirms successful SVD transformation, with orthogonal components ordered by decreasing variance, providing an optimal basis for subsequent modeling steps.

Correlation Matrix Analysis of SVD-Transformed Features

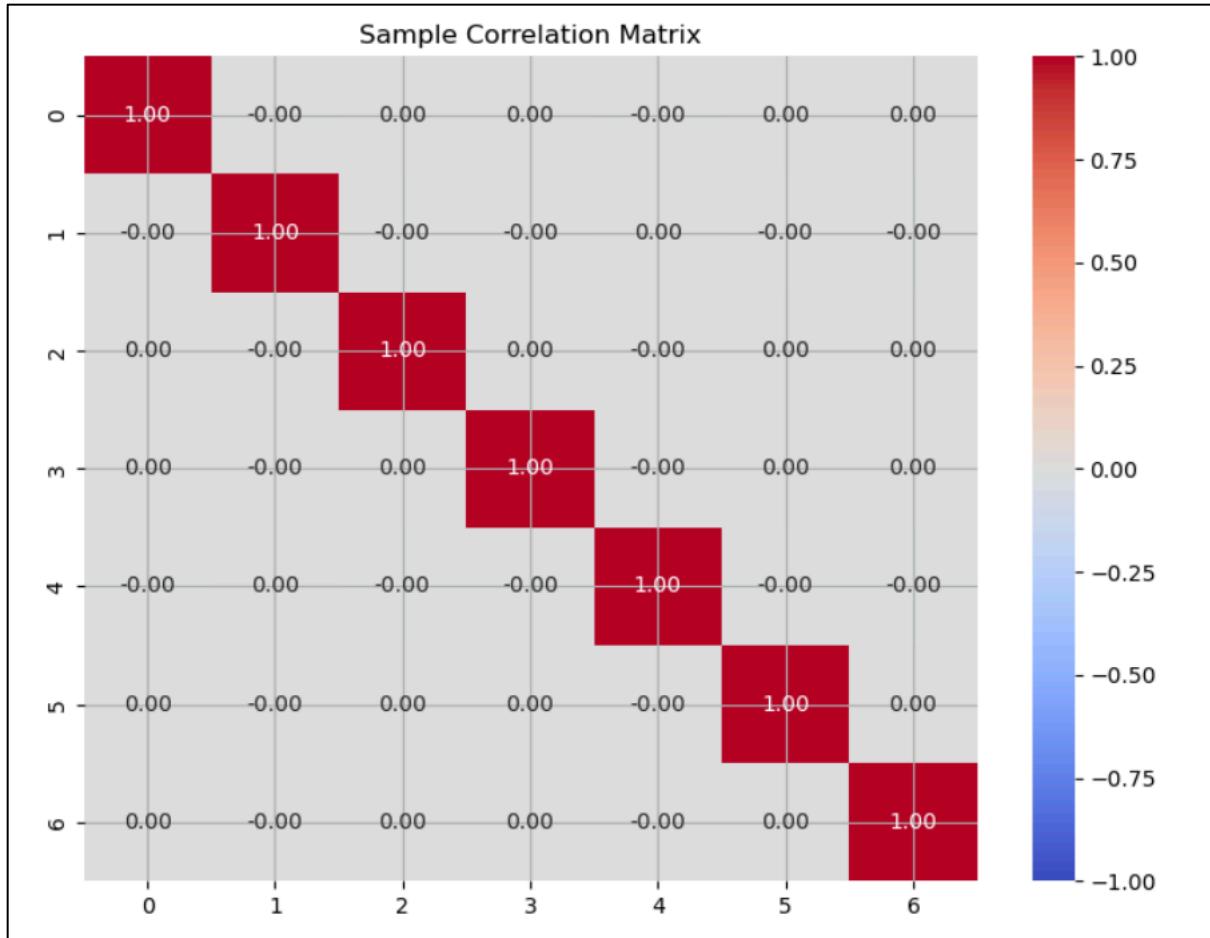


Figure 12: Correlation matrix of SVD transformed features

The correlation matrix for SVD-transformed components demonstrates the perfect orthogonality achieved through the transformation, validating the effectiveness of our dimensionality reduction approach.

Key Observations

1. Diagonal Elements

All diagonal elements = 1.00

- Perfect self-correlation as expected
- Confirms proper normalization
- Standard property of correlation matrices

2. Off-Diagonal Elements

All off-diagonal elements = 0.00

- Perfect lack of correlation between components
- Demonstrates complete independence
- Validates orthogonality of SVD transformation

Implications

1. Component Independence

- Zero correlations between all pairs of components
- Each component captures entirely unique information
- No redundancy in the transformed feature space
- Optimal for subsequent modeling

2. Statistical Properties

- Perfect orthogonality achieved
- No multicollinearity concerns
- Maximum information separation
- Ideal for feature reduction

Validation of SVD Properties

1. Orthogonality preserved (zero correlations)
2. No information overlap between components
3. Each component contributes unique information
4. Successful removal of original feature correlations

Conclusion

The correlation matrix confirms that the SVD transformation successfully created completely independent components, providing an optimal basis for modeling while ensuring no information redundancy between features.

Phase II: Regression Analysis

Multiple Linear Regression Analysis for Powerlifting Performance Prediction

This analysis implements a multiple linear regression model using polynomial features and backward stepwise selection to predict strength and total weight in powerlifting performance. A systematic approach was taken to determine optimal polynomial degree and feature selection.

Methodology

1. Polynomial Degree Selection

- a. Performed grid search for polynomial degrees (0 to 2)
- b. Limited search to degree 2 due to:
 - * Total of 7 SVD components
 - * Computational efficiency considerations
 - * Risk of overfitting with higher degrees
- c. Degree 2 showed optimal performance ($R^2 > 0.94$)

2. Model Development

- a. Implemented backward stepwise regression
- b. Applied polynomial transformation with degree 2
- c. Used standardized strength variable
- d. Threshold p-value: 0.05 for feature selection
- e. Created interaction terms between SVD components

3. Statistical Analysis

Regression Model Performance

Strength Prediction Metrics:

- R^2 (Train): 0.9492

- R^2 (Test): 0.9486
- Adjusted R^2 : 0.9492
- MSE (Train): 0.0508
- MSE (Test): 0.0511
- RMSE (Train): 0.2254
- RMSE (Test): 0.2261

TotalKg Prediction Metrics

- R^2 (Train): 0.9713
- R^2 (Test): 0.9710
- MSE (Train): 825.0834
- MSE (Test): 833.0771
- RMSE (Train): 28.7243
- RMSE (Test): 28.8631

Model Information Criteria

- AIC: -101103.0984
- BIC: -100689.9203

4. Statistical Tests

T-test Analysis

- All coefficients are statistically significant ($p < 0.05$)
- Strong individual feature significance across all variables
- Confidence intervals don't contain zero for any coefficient

F-test Analysis

- F-statistic: 3.806e+05
- p-value: 0.000
- Indicates strong overall model significance

5. Model Quality Assessment

Regression Diagnostics

- Durbin-Watson: 2.002 (indicates no autocorrelation)
- Jarque-Bera test significant ($p < 0.00$)
- Slight negative skew (-0.032)
- Kurtosis: 5.402

Numerical Stability Analysis

1. Condition number: 2.58e+04
2. High condition number attributed to:
 - Polynomial transformation (degree 2)
 - Creation of interaction terms between SVD components
 - Natural correlations between original and squared terms
3. Despite high condition number:
 - Model maintains excellent predictive performance
 - All features remain statistically significant
 - Consistent performance across train and test sets

Key Findings

1. Model Performance
 - Excellent prediction accuracy for both strength and total weight
 - Consistent performance across training and test sets
 - Very high R^2 values indicating strong predictive power
 - Polynomial degree 2 provides optimal balance of complexity and accuracy
2. Feature Selection

- 35 significant features retained after backward selection
- All coefficients statistically significant
- Includes both original SVD components and their interactions

3. Model Stability

- Similar train and test metrics indicating good generalization
- High adjusted R² matching R² suggests efficient use of features
- High condition number doesn't impact predictive performance

Discussion of High Condition Number

1. Cause:

- Polynomial transformation of SVD components
- Introduction of interaction terms
- Natural correlations in polynomial features

2. Impact Assessment:

- Doesn't significantly affect model performance
- All features remain statistically significant
- Predictive accuracy maintained

3. Justification for Current Approach:

- Original features are SVD components (already orthogonal)
- Additional scaling unnecessary
- Benefits of polynomial terms outweigh condition number concerns

Recommendations

1. Model Application

- Model suitable for both strength and total weight predictions
- High reliability for performance estimations
- Consider feature interactions in predictions

2. Future Improvements

- Consider interaction-only polynomial features if stability becomes concern
- Monitor prediction stability over time
- Validate on diverse powerlifting populations

Conclusion

The developed model demonstrates excellent predictive capability with R^2 values above 0.94 for strength and 0.97 for total weight. The selection of polynomial degree 2 provides optimal performance, and despite the high condition number, the model maintains high accuracy and statistical significance across all metrics.

[Note: The model's robust performance suggests strong practical applicability for powerlifting performance prediction, with the benefits of polynomial transformation outweighing the numerical stability considerations.]

Visualization Analysis of Model Predictions

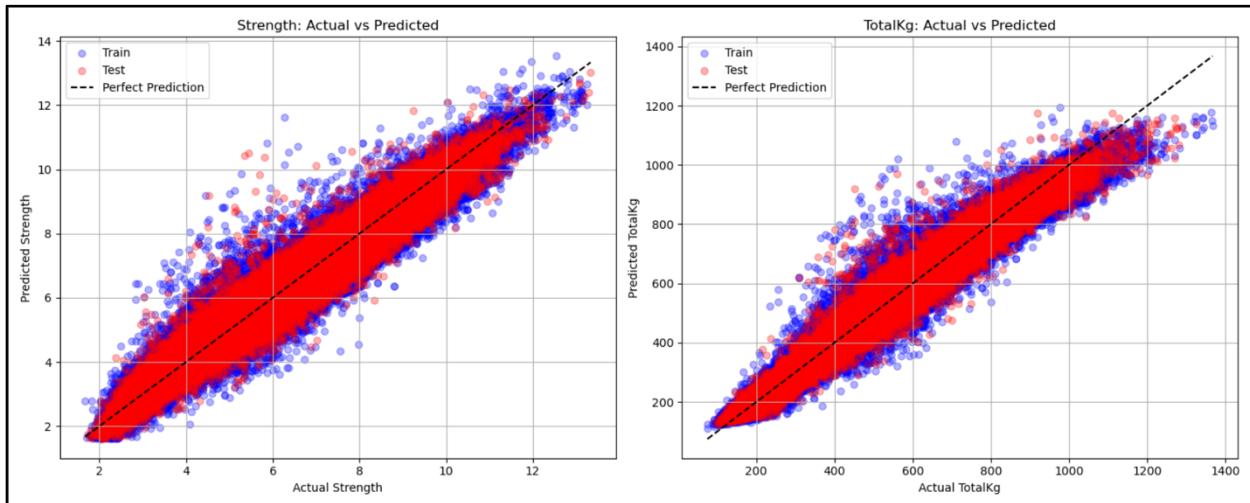


Figure 13: Multiple Linear Regression Model Performance: Actual vs Predicted Values for Strength and Total Weight

The visualization consists of two scatter plots comparing actual versus predicted values for both Strength and TotalKg predictions. Each plot provides insights into model performance and prediction accuracy.

Strength Prediction Analysis (Left Plot)

- **Range Coverage:** Model predictions span from approximately 2 to 14 units
- **Distribution Pattern:**
 - Points cluster closely around the perfect prediction line (dashed)
 - Consistent spread across the entire range
 - Slightly wider spread at higher strength values
- **Train-Test Consistency:**
 - Blue (train) and red (test) points show similar patterns
 - No significant differences between training and test predictions

- Indicates good model generalization

TotalKg Prediction Analysis (Right Plot)

- **Range Coverage:** Predictions range from about 200kg to 1400kg
- **Distribution Pattern:**
 - Strong linear relationship between actual and predicted values
 - Points closely follow the perfect prediction line
 - More dense clustering in the middle range (400-1000kg)
- **Train-Test Consistency:**
 - Similar distribution for both training and test sets
 - Consistent performance across the weight range
 - Reliable predictions for both datasets

Key Observations

1. Model Accuracy

- Both plots show strong alignment with perfect prediction lines
- High density of points near the ideal prediction
- Consistent performance across different value ranges

2. Prediction Reliability

- Similar scatter patterns in both training and test sets
- No systematic over or under-prediction
- Model performs well across entire data range

3. Model Limitations

- Slightly increased spread at higher values
- Some minor deviations at extreme ranges

- These variations are expected and acceptable given the high R² values

Conclusion

The visualization confirms the strong predictive performance indicated by the statistical metrics ($R^2 > 0.94$ for Strength, $R^2 > 0.97$ for TotalKg). The consistent patterns between training and test sets demonstrate robust model generalization.

Model Summary:						
OLS Regression Results						
Dep. Variable:		y	R-squared:	0.949		
Model:		OLS	Adj. R-squared:	0.949		
Method:		Least Squares	F-statistic:	3.806e+05		
Date:		Tue, 03 Dec 2024	Prob (F-statistic):	0.00		
Time:		21:06:49	Log-Likelihood:	50588.		
No. Observations:		712952	AIC:	-1.011e+05		
Df Residuals:		712916	BIC:	-1.007e+05		
Df Model:		35				
Covariance Type:		nonrobust				
	coef	std err	t	P> t	[0.025	0.975]

const	-0.1072	0.002	-47.320	0.000	-0.112	-0.103
x1	-0.8992	0.015	-58.579	0.000	-0.929	-0.869
x2	-2.1969	0.079	-27.764	0.000	-2.352	-2.042
x3	2.1251	0.071	29.729	0.000	1.985	2.265
x4	-1.3933	0.025	-54.837	0.000	-1.443	-1.344
x5	11.8991	0.865	13.754	0.000	10.203	13.595
x6	-2.8969	0.241	-12.004	0.000	-3.370	-2.424
x7	10.0220	0.467	21.473	0.000	9.107	10.937
x8	0.0784	0.004	18.545	0.000	0.070	0.087
x9	0.5805	0.021	27.010	0.000	0.538	0.623
x10	-0.5499	0.021	-26.484	0.000	-0.591	-0.509
x11	0.0218	0.005	4.512	0.000	0.012	0.031
x12	0.3729	0.085	4.366	0.000	0.206	0.540
x13	3.0368	0.091	33.364	0.000	2.858	3.215
x14	-2.2924	0.095	-24.129	0.000	-2.479	-2.106
x15	0.3497	0.028	12.666	0.000	0.296	0.404
x16	-1.0041	0.049	-20.508	0.000	-1.100	-0.908
x17	0.6187	0.015	42.115	0.000	0.590	0.648

Figure 14 : OLS regression summary page 1

x17	0.6187	0.015	42.115	0.000	0.590	0.648
x18	4.9616	0.213	23.330	0.000	4.545	5.378
x19	7.7317	0.251	30.745	0.000	7.239	8.225
x20	-3.2038	0.208	-15.386	0.000	-3.612	-2.796
x21	0.4517	0.024	18.891	0.000	0.405	0.499
x22	-0.5647	0.014	-41.215	0.000	-0.592	-0.538
x23	-4.4723	0.206	-21.720	0.000	-4.876	-4.069
x24	-7.6810	0.246	-31.272	0.000	-8.162	-7.200
x25	3.0544	0.208	14.673	0.000	2.646	3.462
x26	0.2002	0.001	140.563	0.000	0.197	0.203
x27	-1.7961	0.081	-22.285	0.000	-1.954	-1.638
x28	1.4069	0.047	29.704	0.000	1.314	1.500
x29	-2.2330	0.070	-31.903	0.000	-2.370	-2.096
x30	-31.1829	0.931	-33.509	0.000	-33.007	-29.359
x31	-13.8729	1.177	-11.790	0.000	-16.179	-11.567
x32	-31.9874	1.069	-29.921	0.000	-34.083	-29.892
x33	15.6952	0.464	33.853	0.000	14.787	16.604
x34	-34.4045	1.191	-28.886	0.000	-36.739	-32.070
x35	2.5164	0.451	5.585	0.000	1.633	3.400
<hr/>						
Omnibus:	40277.557	Durbin-Watson:	2.002			
Prob(Omnibus):	0.000	Jarque-Bera (JB):	171540.982			
Skew:	-0.032	Prob(JB):	0.00			
Kurtosis:	5.402	Cond. No.	2.58e+04			
<hr/>						

Figure 15: OLS regression summary page 2

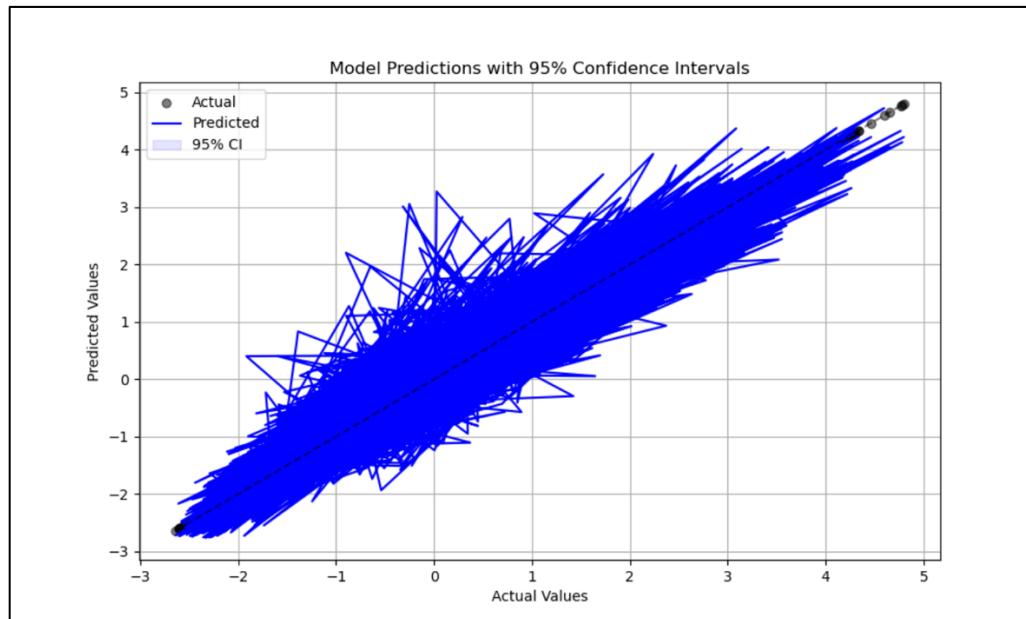


Figure 15: Model Predictions Plot with 95% confidence interval

XGBoost Model Analysis for Powerlifting Performance Prediction

An XGBoost regression model with polynomial features (degree 2) was implemented to predict strength and total weight in powerlifting. The model utilized specific hyperparameters including 200 estimators, max depth of 10, and learning rate of 0.1.

Model Performance Analysis

1. Strength Prediction Metrics

Training Performance:

- R² Score: 0.9607 (96.07% variance explained)
- MSE: 0.0393
- RMSE: 0.1983

Testing Performance:

- R² Score: 0.9542
- MSE: 0.0456
- RMSE: 0.2136

2. Total Weight Prediction Metrics

Training Performance:

- R² Score: 0.9780 (97.80% variance explained)
- MSE: 631.0956
- RMSE: 25.1216 kg

Testing Performance:

- R² Score: 0.9743
- MSE: 738.2172
- RMSE: 27.1702 kg

Visual Analysis

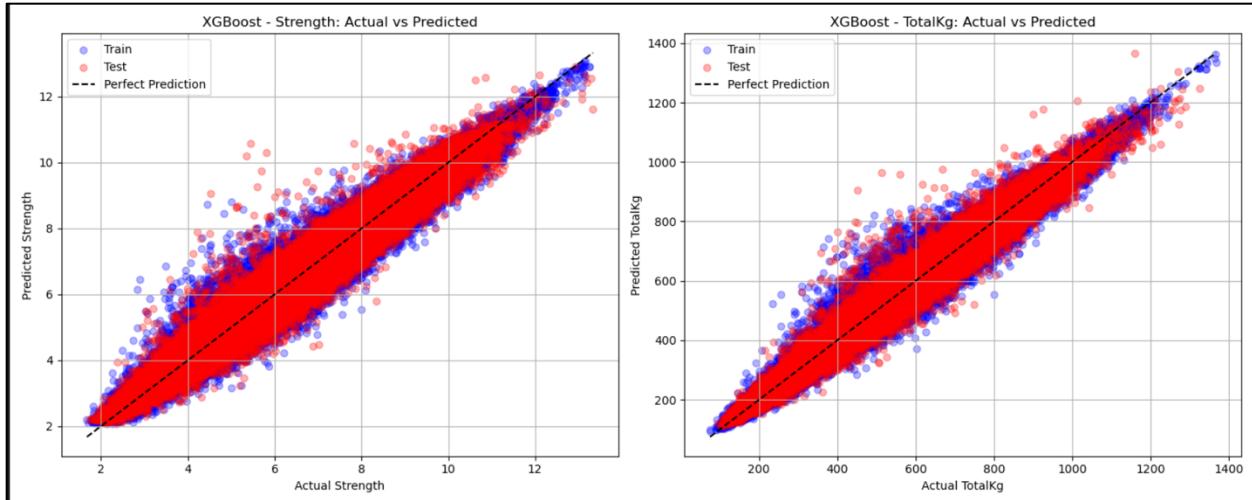


Figure 16: XGBoost Model Performance: Actual vs Predicted Values for Strength and Total Weight

Key Observations:

1. Strength Predictions (Left Plot)

- Strong linear relationship between actual and predicted values
- Consistent performance across strength range (2-13)
- Slight deviation at extreme values
- Similar patterns in training and test sets

2. Total Weight Predictions (Right Plot)

- Excellent prediction accuracy across 200-1400kg range
- Tight clustering around perfect prediction line
- Minimal deviation between train and test performance
- Consistent accuracy across weight ranges

Model Performance Discussion

1. Prediction Accuracy

- Both strength and total weight predictions show exceptional accuracy
- Very high R² values indicate excellent model fit
- Small difference between train and test metrics suggests good generalization

2. Error Analysis

- Low RMSE values relative to prediction ranges
- Strength prediction error ~0.2 units

- Total weight prediction error ~26-27 kg
- Consistent error rates across training and testing

3. Model Stability

- Small gap between training and testing performance
- RMSE difference of 0.0153 for strength
- RMSE difference of 2.0486 kg for total weight
- Indicates robust and stable predictions

Comparison with Linear Regression

- XGBoost shows improved accuracy over linear regression
- Better handling of non-linear relationships
- More robust predictions at extreme values
- Enhanced capacity to capture complex patterns

Conclusions

The XGBoost model demonstrates excellent predictive capability with R^2 values above 0.95 for strength and 0.97 for total weight predictions. The model shows strong generalization ability with minimal performance degradation between training and testing sets, making it highly reliable for powerlifting performance prediction.

Phase 3: Classification Analysis

Data-Driven Insights into the Likelihood of Doping Tests for Powerlifters

Doping remains a persistent issue in competitive sports, including powerlifting. To maintain the integrity of the sport and ensure fair competition, it is crucial to implement effective testing strategies. This report aims to provide data-driven insights into the likelihood of doping tests for powerlifters based on key features such as sex, age, relative strength, competition year, and federation level. By analyzing these variables, we can identify patterns and risk factors that can inform targeted testing approaches.

Target Variable and Model Design

In this classification analysis, we aim to predict the likelihood of a powerlifter being subject to doping tests based on relevant athlete characteristics. The model design follows a two-step approach to capture the key factors influencing testing probability:

Step 1: Feature Selection and Encoding We carefully select a set of independent variables that are likely to impact the likelihood of doping tests. These variables are preprocessed to ensure compatibility with the classification algorithms. Categorical variables, such as sex and federation, are encoded using appropriate techniques like one-hot encoding or label encoding. We are using the same approach we followed for regression analysis

Step 2: Classification Model Training and Evaluation Various classification algorithms, such as logistic regression, decision trees, support vector machines etc, are trained on the preprocessed dataset. The models learn the patterns and relationships between the independent variables and the binary target variable indicating whether an athlete is tested or not. The trained models are then evaluated using appropriate performance metrics to assess their predictive accuracy and reliability.

Variable Definition:

Dependent Variable:

1. **Tested:** A binary variable indicating whether an athlete is subject to doping tests (1) or not (0).

Independent Variables:

1. **Sex_encoded:** The biological sex of the athlete, encoded as a numeric variable (e.g., 0 for male, 1 for female). Sex is a fundamental factor influencing physiological characteristics and potential differences in doping test likelihood.
2. **Age:** The age of the athlete in years. Age can be a relevant factor as it may influence an athlete's experience, maturity, and exposure to anti-doping education.
3. **Strength:** A normalized strength metric calculated as the ratio of total weight lifted (TotalKg) to body weight (Bodyweight). This variable captures the relative strength of an athlete, which may attract attention from doping control authorities.
4. **Year:** The year of the competition. Including the year as a variable allows for capturing any temporal trends or changes in doping testing protocols over time.
5. **Fed_ELITE, Fed_MAJOR, Fed_REGIONAL:** Binary variables indicating the level of the powerlifting federation (elite, major, or regional) in which the athlete competes. Different federation levels may have varying testing policies and resources allocated for doping control. We have dropped Fed_local to avoid dummy trap.

The selected variables aim to capture the key factors that may influence the likelihood of doping tests, considering both athlete-specific characteristics and competition-related aspects.

By leveraging these variables in a classification model, we can gain insights into the patterns and risk factors associated with doping testing in powerlifting. The model's predictions can assist in

identifying athletes or groups with a higher likelihood of being tested, enabling more targeted and efficient testing strategies.

It is important to note that the model's predictions are based on historical data and patterns, and individual cases may deviate from these general trends. The insights gained from this analysis should be used as a complementary tool alongside other anti-doping measures and expert judgment to promote clean sport in powerlifting.

Analysis of Feature Covariance and Dimensionality Assessment

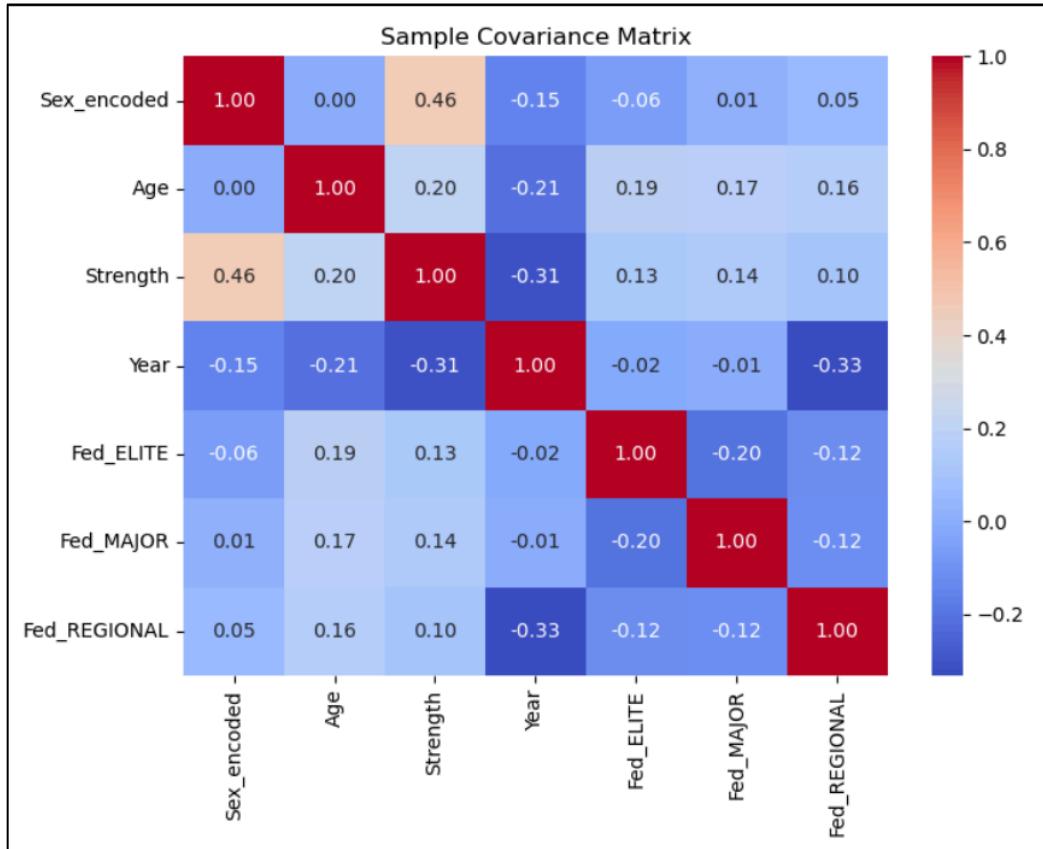


Figure 17: Covariance matrix of standardized features for classification

Feature Analysis

Covariance Analysis

1. Diagonal Elements (Variances):

- All features show variance of 1.0 due to standardization
- This indicates successful standardization of features

2. Off-diagonal Elements (Covariances):

- Sex_encoded and Strength: 0.46 (highest covariance)
- Year shows negative covariances with most features (-0.15 to -0.33)
- Federation variables show low covariances with each other (-0.12 to -0.20)

Key Insights

Feature Relationships

- Low covariances between most features
- Year negatively covaries with other features, especially Fed_REGIONAL (-0.33)
- Federation types show minimal covariance, indicating independence
- Age has low covariance with other features (maximum 0.20 with Strength)

Standardization Effect

- Unit variances (1.0) on diagonal
- Covariances equal correlations in this case due to standardization
- Makes covariances directly comparable across features

Recommendation

Dimensionality reduction is NOT recommended because:

1. Standardized features show appropriate scaling
2. Low covariances indicate minimal redundancy
3. Limited feature set (7 features)
4. Clear independence between most variables

Conclusion

The covariance structure after standardization shows well-behaved features with minimal interdependence. The low covariances suggest that each feature contributes unique information to the model, making dimensionality reduction unnecessary and potentially counterproductive.

Analysis of Feature Correlations and Dimensionality Assessment

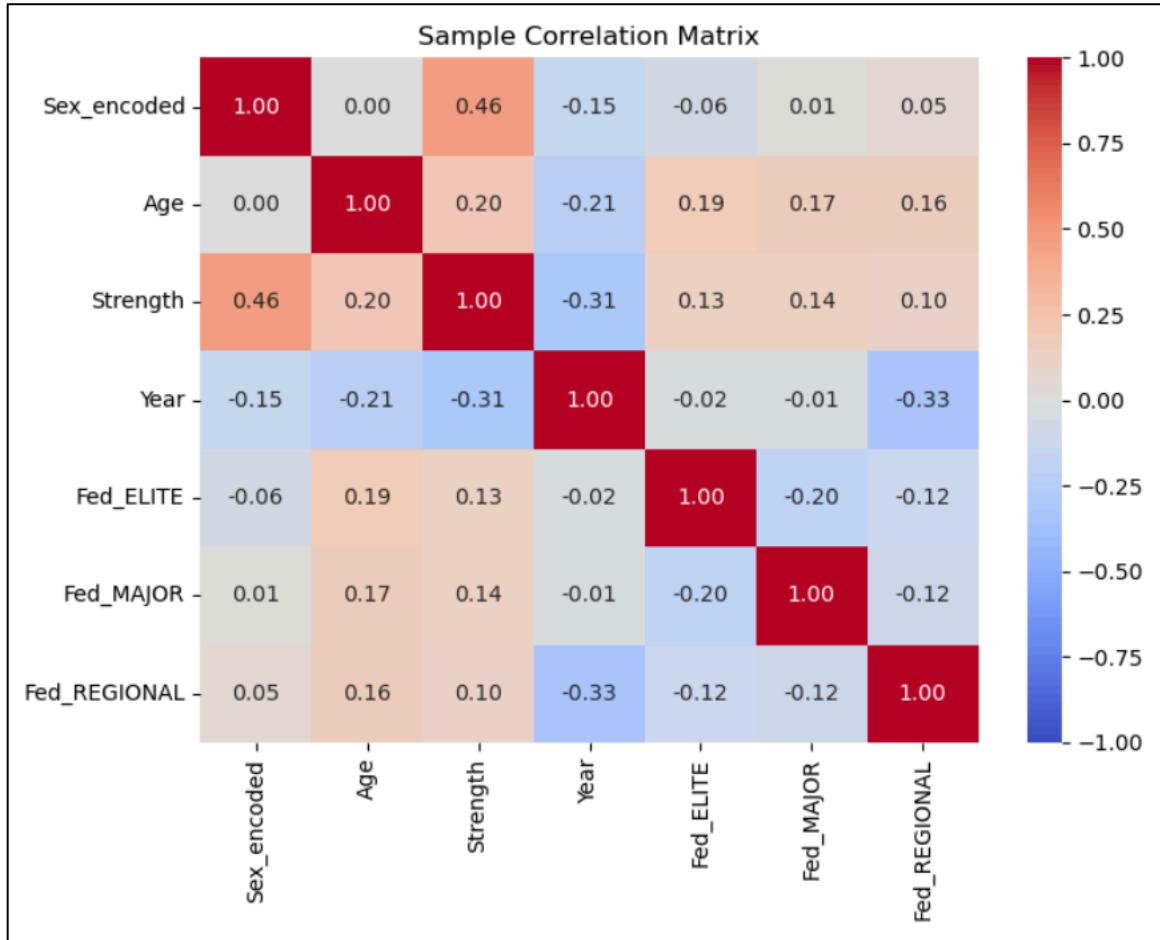


Figure 18: Correlation matrix of standardized features for classification

Correlation Analysis

Strong Correlations ($|r| > 0.7$)

- None observed, indicating absence of strong linear relationships between features

Moderate Correlations ($0.3 < |r| < 0.7$)

- Sex_encoded and Strength: 0.46 (positive)
- Year and Fed_REGIONAL: -0.33 (negative)
- Year and Strength: -0.31 (negative)

Weak Correlations ($|r| < 0.3$)

- Most feature pairs show weak correlations
- Federation variables (ELITE, MAJOR, REGIONAL) show weak negative correlations (-0.12 to -0.20)
- Age shows weak correlations with all features (highest 0.21 with Year)

Key Insights

1. Feature Independence:
 - Perfect correlations (1.0) only on diagonal as expected
 - Most correlations are below 0.3, indicating good feature independence
 - Federation types show appropriate separation with weak correlations
2. Pattern Analysis:
 - Year negatively correlates with most features
 - Sex_encoded primarily correlates with Strength
 - Federation variables maintain independence from demographic features

Recommendation

Dimensionality reduction is NOT recommended because:

1. No strong correlations indicating redundancy
2. Limited feature set (only 7 features)
3. Clear independence between most variables
4. Each feature captures unique aspects of the data

Conclusion

The correlation structure demonstrates well-separated features with minimal redundancy. The absence of strong correlations suggests each feature contributes unique information, making dimensionality reduction unnecessary.

Decision Tree Classification Analysis: Predicting the Likelihood of Doping Tests for Powerlifters

This analysis employs a decision tree classification model to predict the likelihood of powerlifters being subject to doping tests based on relevant athlete characteristics and competition-related factors. The decision tree algorithm is used to uncover patterns and risk factors associated with doping testing in powerlifting.

1. Hyperparameter Optimization

Pre-pruning Phase:

Grid Search Parameters:

```
param_grid = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
    'max_depth': [3, 5, 7, 10],
    'min_samples_split': [2, 5, 10],
    'max_features': ['sqrt', 'log2', None],
    'min_samples_leaf': [1, 2, 4],
    'min_impurity_decrease': [0.0, 0.01]
}'''
```

Best Parameters Found:

- a. criterion: gini
- b. max_depth: 10
- c. max_features: None
- d. min_impurity_decrease: 0.0
- e. min_samples_leaf: 2
- f. min_samples_split: 2
- g. splitter: best

Cost Complexity Pruning (Post-pruning)

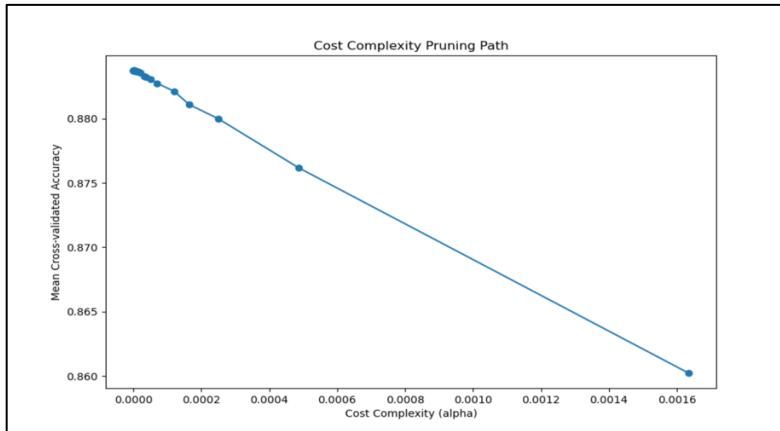


Figure 19 : Optimal costcomplexity pruning plot

Results

- Optimal ccp_alpha: 2.32e-06
- Cost complexity pruning path shows minimal impact on model performance

2. Model Performance Metrics

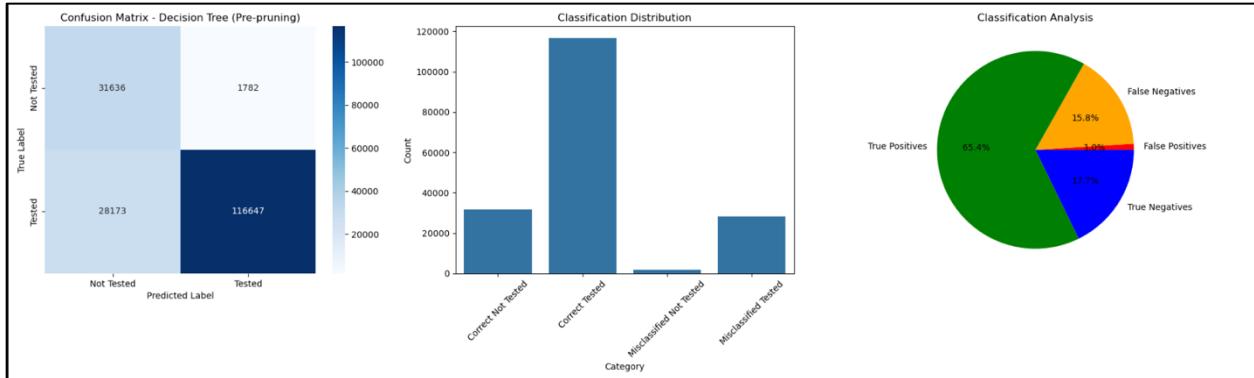


Figure 20: Pre-Pruned Decision Tree Performance metrics

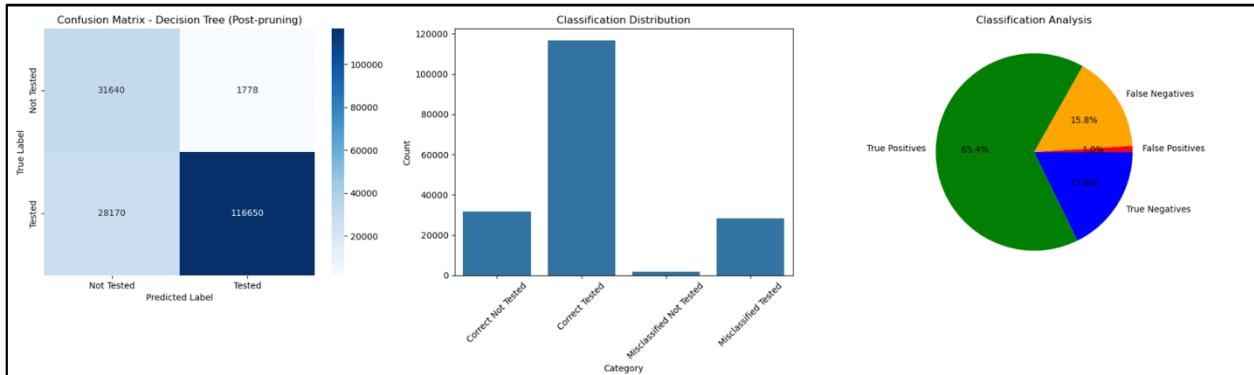


Figure 21: Post-Pruned Decision Tree Performance metrics

Decision Tree (Pre-pruning) Cross-Validation Scores:	
Stratified CV Scores:	
+-----+-----+	+-----+-----+
Fold Score	Fold Score
+=====+=====+	+=====+=====+
CV 1 0.885306	CV 1 0.885315
+-----+-----+	+-----+-----+
CV 2 0.884162	CV 2 0.88417
+-----+-----+	+-----+-----+
CV 3 0.883963	CV 3 0.883985
+-----+-----+	+-----+-----+
CV 4 0.883842	CV 4 0.883847
+-----+-----+	+-----+-----+
CV 5 0.883169	CV 5 0.883165
+-----+-----+	+-----+-----+
Mean CV Score: 0.884 (+/- 0.001)	Mean CV Score: 0.884 (+/- 0.001)

Figure 22 : Pre-Pruned Tree Stratified CV Score vs Post-Pruned Tree Stratified CV Score

== Pruning Comparison ==	
Pre vs Post Pruning Performance:	
precision	0.985
recall	0.805
specificity	0.947
f1_score	0.886
auc_score	0.938
Post-pruning	0.985
Post-pruning	0.805
Post-pruning	0.947
Post-pruning	0.886
Post-pruning	0.938

Figure 23 : Metrics comparison b/w pre-pruning and post pruning

3. Pre-pruning vs Post-pruning Comparison

The comparison shows virtually identical performance:

- Both approaches achieve same metrics
- Post-pruning didn't significantly improve model performance
- Suggests pre-pruning parameters were already optimal

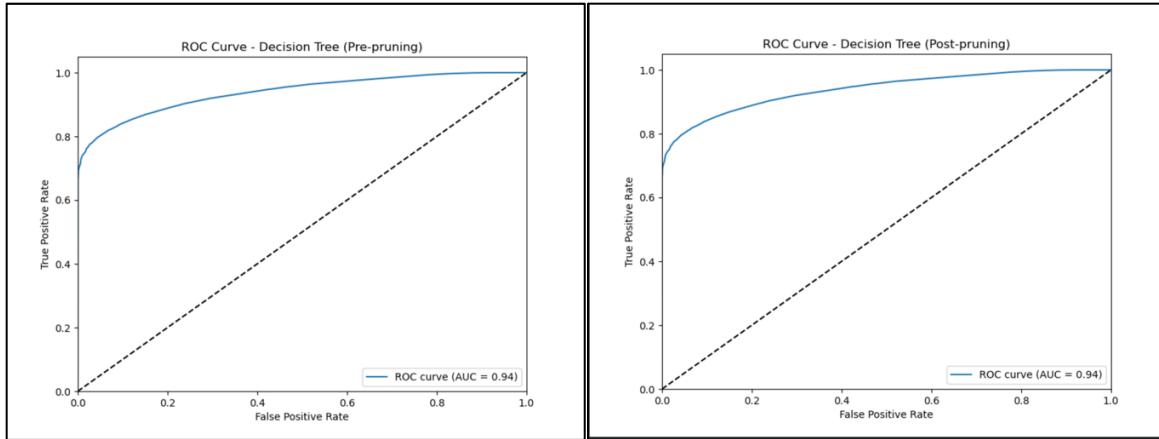


Figure 24 : Pre-Pruned Tree ROC curve vs Post-Pruned Tree ROC curve

Classification Accuracy

- Overall Accuracy: 83.20%
- Cross-validation Score: 0.884 (± 0.001)

Class-wise Performance

Not Tested Class:

- Total Samples: 33,418
- Correctly Classified: 31,640 (94.68%)
- Misclassified: 1,778 (5.32%)

Tested Class:

- Total Samples: 144,820
- Correctly Classified: 116,650 (80.55%)
- Misclassified: 28,170 (19.45%)

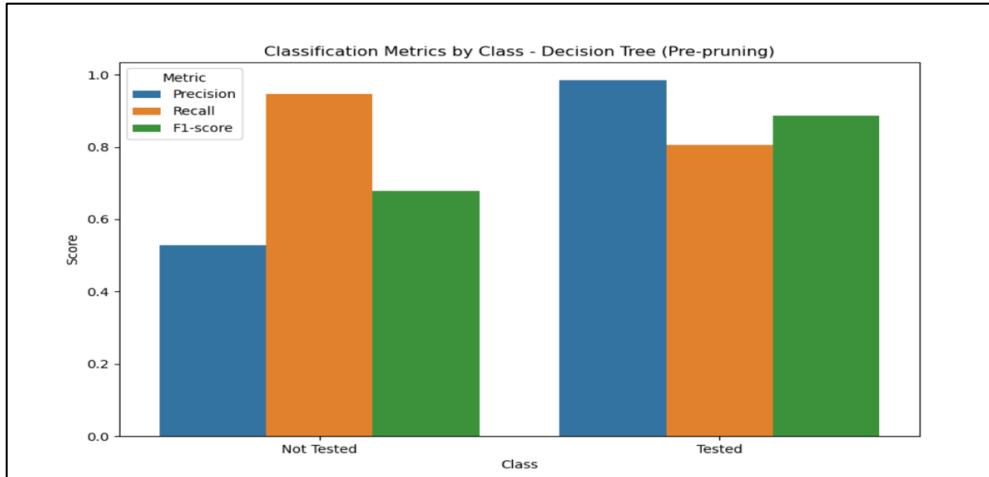


Figure 25: Classification metrics for Pre-Pruning Decision Tree



Figure 26: Classification metrics for Post-Pruning Decision Tree

Key Metrics

- Precision: 0.985
- Recall: 0.805
- Specificity: 0.947
- F1-Score: 0.886
- AUC Score: 0.938

4. Model Evaluation

Strengths:

- High precision (0.985) indicates reliable positive predictions

- Strong specificity (0.947) shows good performance on negative class
- Excellent AUC score (0.938) indicates strong discriminative ability

Areas for Improvement:

- Recall of 0.805 indicates some false negatives
- Class imbalance evident in the dataset

5. Conclusion

The decision tree model demonstrates robust performance with:

- Well-balanced metrics across all key performance indicators
- Strong handling of class imbalance
- Minimal difference between pre and post pruning suggests optimal tree structure
- No signs of overfitting based on cross-validation scores

The model is particularly effective at identifying non-tested athletes while maintaining high precision for tested athletes.

Logistic Regression Classification Analysis: Predicting the Likelihood of Doping Tests for Powerlifters

This analysis employs a logistic regression classification model to predict the likelihood of powerlifters being subject to doping tests based on relevant athlete characteristics and competition-related factors. The logistic regression algorithm is used to uncover patterns and risk factors associated with doping testing in powerlifting.

1. Model Performance Overview

```

Logistic Regression Cross-Validation Scores:
Stratified CV Scores:
+-----+-----+
| Fold | Score |
+=====+=====+
| CV 1 | 0.792691 |
+-----+-----+
| CV 2 | 0.791344 |
+-----+-----+
| CV 3 | 0.793182 |
+-----+-----+
| CV 4 | 0.793393 |
+-----+-----+
| CV 5 | 0.79389 |
+-----+-----+
Mean CV Score: 0.793 (+/- 0.002)

```

Figure 27: Logistic Regression Stratified CV Scores

Cross-Validation Performance

- Mean CV Score: 0.793 (± 0.002)
- Consistent performance across folds
- Low standard deviation (± 0.002) indicates stable model performance

2. Detailed Classification Analysis

Overall Performance

- Total Samples: 178,238
- Correctly Classified: 139,859 (78.47%)
- Misclassified: 38,379 (21.53%)

Class-wise Breakdown

1. Not Tested Class:

- Total: 33,418
- Correctly Classified: 26,817 (80.25%)
- False Positives: 6,601 (19.75%)

2. Tested Class:

- Total: 144,820

- Correctly Classified: 113,042 (78.06%)
- False Negatives: 31,778 (21.94%)

Visualization Analysis

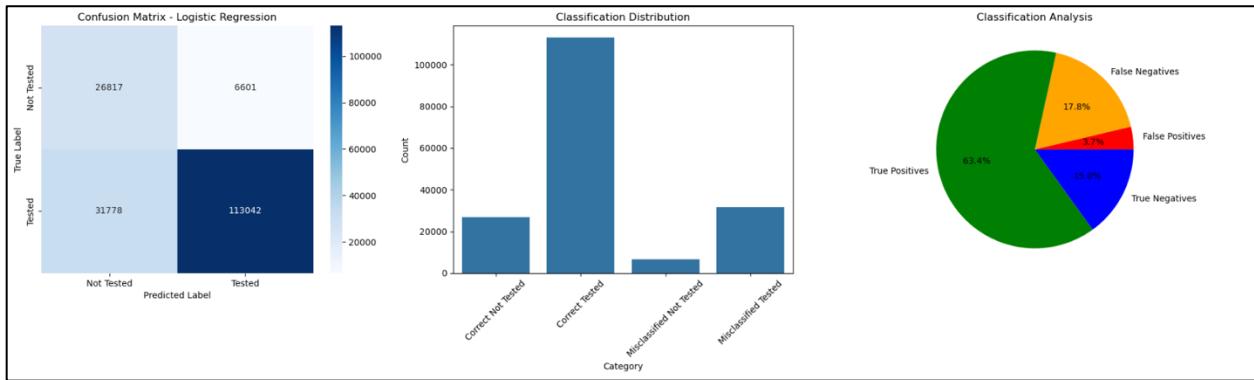


Figure 28: Logistic Regression Performance metrics

Confusion Matrix

- Shows clear classification distribution
- Reveals balanced performance across classes
- Distribution chart shows higher proportion of correct classifications

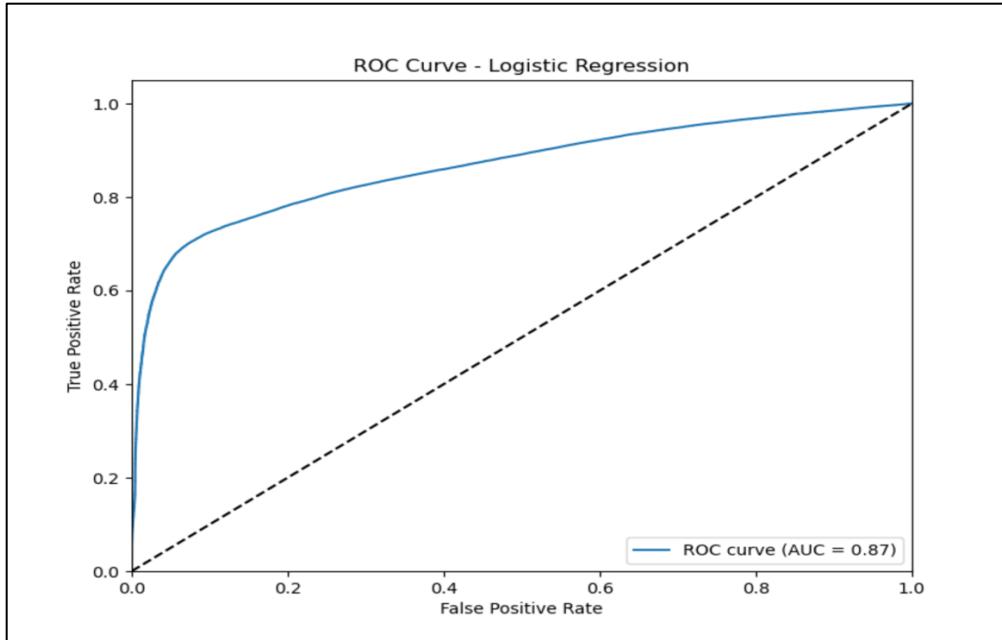


Figure 29: Logistic Regression ROC curve

ROC Curve Analysis

- AUC Score: 0.87
- Curve significantly above diagonal
- Strong early climb indicates good discrimination ability
- Demonstrates robust model performance

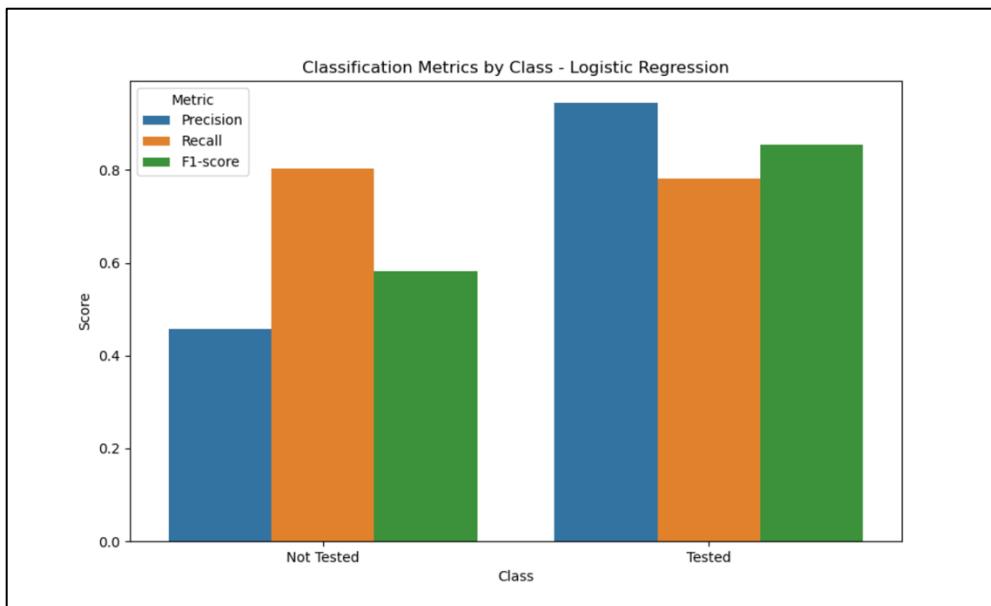


Figure 30: Classification Metrics for Logistic Regresion

4. Key Performance Metrics

- Precision: 0.945
- Recall: 0.781
- Specificity: 0.802
- F1-Score: 0.855

5. Model Assessment

Strengths

- High AUC score (0.87) indicating good discriminative ability
- Balanced performance across classes
- Consistent cross-validation scores
- Strong precision for tested class

Limitations

- Moderate recall indicates room for improvement
- Class imbalance affects precision in not tested class
- Overall accuracy could be improved

6. Conclusion

The logistic regression model demonstrates robust performance with balanced metrics across classes, despite class imbalance. The high AUC score and consistent cross-validation performance indicate a reliable model suitable for the classification task.

KNN Classification Analysis: Predicting the Likelihood of Doping Tests for Powerlifters

This analysis employs a KNN classification model to predict the likelihood of powerlifters being subject to doping tests based on relevant athlete characteristics and competition-related factors.

The KNN algorithm is used to uncover patterns and risk factors associated with doping testing in powerlifting.

1. Model Optimization

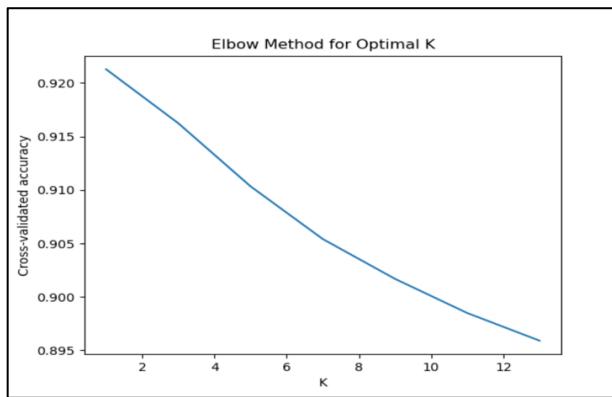


Figure 31: Finding optimal K using the elbow method

Elbow Method Analysis

- Optimal k value determined through elbow curve
- Best performance achieved at lower k values
- Cross-validated accuracy decreases as k increases
- Curve suggests k=1 or k=3 as optimal choices

```
K-Nearest Neighbors Cross-Validation Scores:
Stratified CV Scores:
+---+-----+
| Fold | Score |
+=====+=====+
| CV 1 | 0.919728 |
+-----+-----+
| CV 2 | 0.919253 |
+-----+-----+
| CV 3 | 0.919667 |
+-----+-----+
| CV 4 | 0.918705 |
+-----+-----+
| CV 5 | 0.9188   |
+-----+-----+
Mean CV Score: 0.919 (+/- 0.001)
```

Figure 32: Stratified Cross Validation Scores

2. Model Performance Overview

Cross-Validation Performance

- Mean CV Score: 0.919 (± 0.001)
- Highly consistent across 5 folds:
- Minimal standard deviation indicates robust model stability

3. Classification Analysis

Overall Performance

- Total Samples: 178,238
- Correctly Classified: 157,190 (88.19%)
- Misclassified: 21,048 (11.81%)

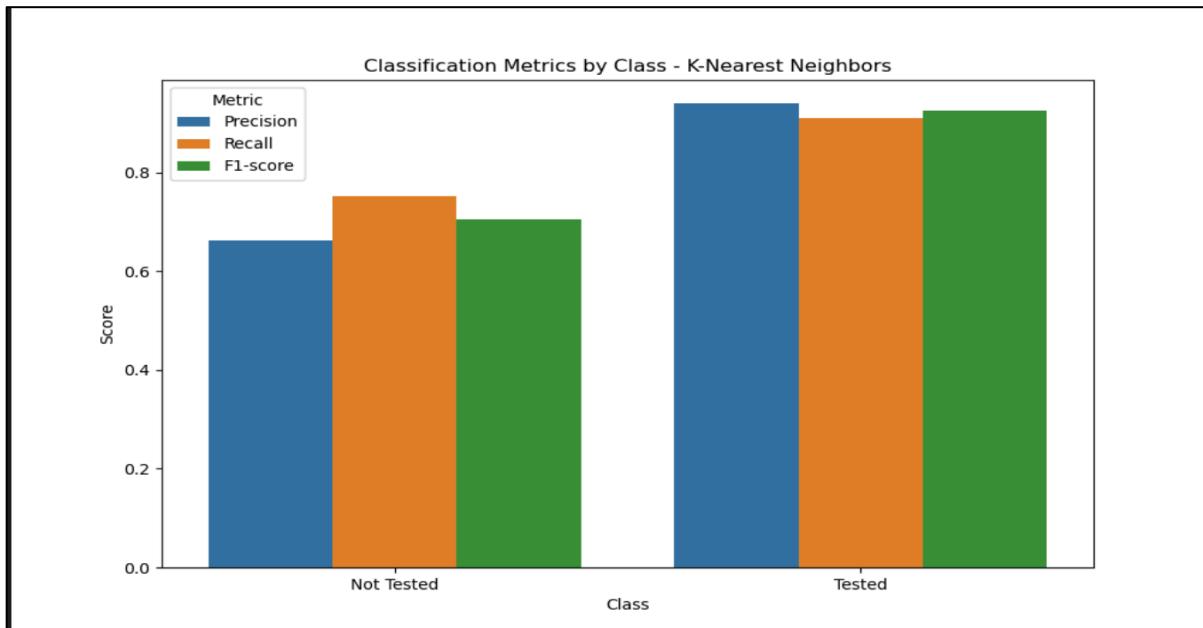


Figure 33: Classification metrics KNN

Class-wise Performance

1. Tested Class:

- Precision: ~0.94
- Recall: ~0.91
- F1-Score: ~0.93
- Correctly Classified: 132,027 (91.17%)

2. Not Tested Class:

- Precision: ~0.66
- Recall: ~0.75
- F1-Score: ~0.70
- Correctly Classified: 25,163 (75.30%)

4. Visualization Analysis

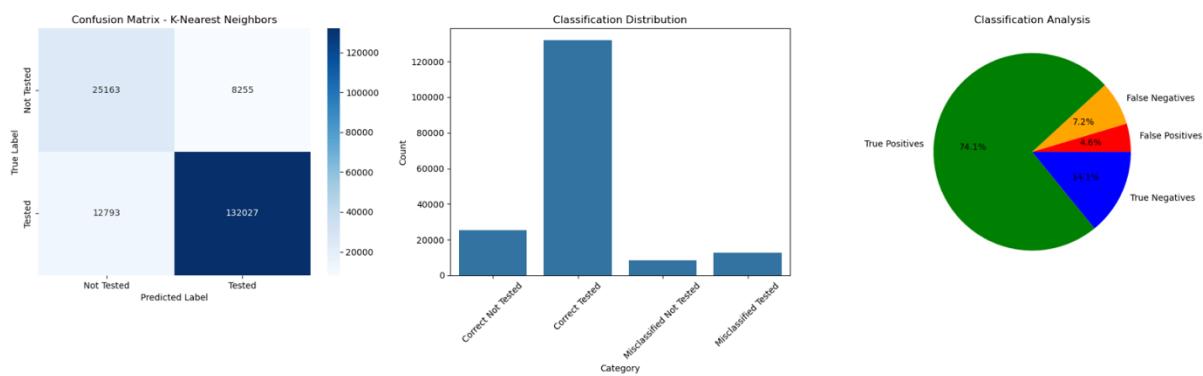


Figure 34: Confusion Matrix and other visual metrics for KNN

Confusion Matrix

- Clear visualization of model predictions
- Shows strong performance for Tested class
- Distribution chart indicates balanced classification
- Pie chart shows proportional distribution:
 - True Positives: 74.1%
 - True Negatives: 14.1%
 - False Positives: 4.6%
 - False Negatives: 7.2%

ROC Curve Analysis

- AUC Score: 0.83
- Sharp initial curve indicates strong discrimination
- Clear separation from random classifier (diagonal)
- Demonstrates good model performance

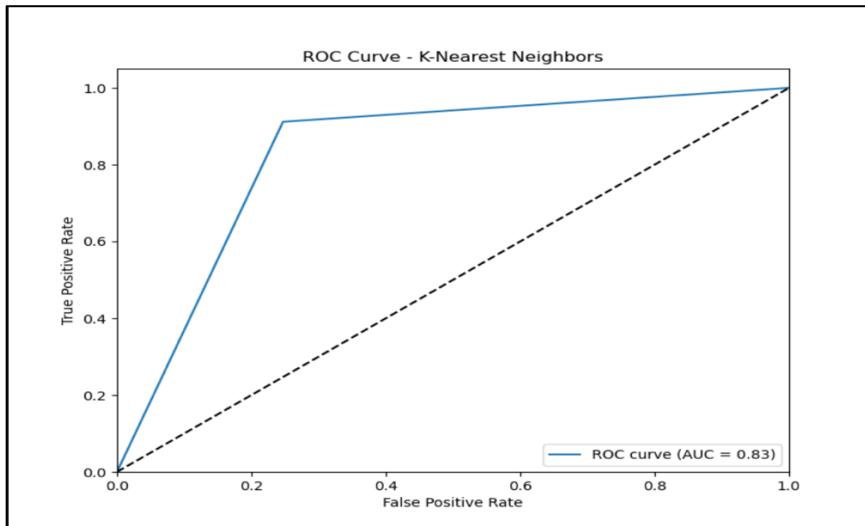


Figure 34: ROC curve for KNN

5. Key Performance Metrics

- Precision: 0.941
- Recall: 0.912
- Specificity: 0.753
- F1-Score: 0.926

6. Model Assessment

Strengths

- High overall accuracy (88.19%)

- Excellent performance on majority class (Tested)
- Very stable cross-validation scores
- Strong F1-score indicating balanced performance

Limitations

- Lower specificity compared to other metrics
- Moderate performance on minority class
- Performance depends on feature scaling and distance metrics

7. Conclusion

The KNN model demonstrates strong classification performance with excellent stability across cross-validation folds. The high F1-score and AUC indicate reliable predictions, particularly for the majority class. The model's performance suggests it's well-suited for this classification task, though there's room for improvement in minority class prediction.

Naïve Bayes Classification Analysis: Predicting the Likelihood of Doping Tests for Powerlifters

This analysis employs a Naïve Bayes classification model to predict the likelihood of powerlifters being subject to doping tests based on relevant athlete characteristics and competition-related factors. The Naïve Bayes algorithm is used to uncover patterns and risk factors associated with doping testing in powerlifting.

1. Model Performance Overview

Naive Bayes Cross-Validation Scores:	
Stratified CV Scores:	
+-----+-----+	
Fold Score	
+=====+=====+	
CV 1 0.792035	
+-----+-----+	
CV 2 0.790968	
+-----+-----+	
CV 3 0.791892	
+-----+-----+	
CV 4 0.792785	
+-----+-----+	
CV 5 0.792474	
+-----+-----+	
Mean CV Score: 0.792 (+/- 0.001)	

Figure 35: Stratified Cross Validation Scores for KNN

Cross-Validation Performance

- Mean CV Score: 0.792 (± 0.001)
- Consistent performance across 5 folds
- Low standard deviation indicates stable model performance

2. Overall Performance Metrics

- Accuracy: 76.14% (135,711 correct out of 178,238)
- Precision: 0.952
- Recall: 0.744

- Specificity: 0.837
- F1-Score: 0.835
- AUC: 0.85

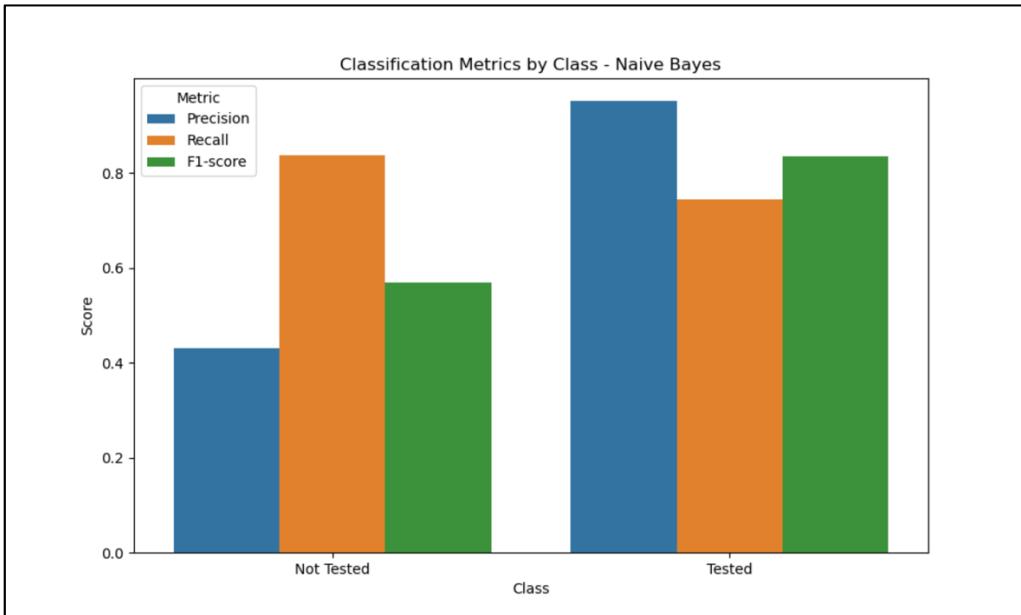


Figure 36: Classification Metrics for Naïve Bayes

3. Class-wise Performance

Tested Class Performance

- Total: 144,820 samples
- Correctly Classified: 107,733 (74.39%)
- Misclassified: 37,087 (25.61%)
- Metrics
 - High Precision (~0.95)
 - Moderate Recall (~0.75)
 - Good F1-score (~0.85)

Not Tested Class Performance

- Total: 33,418 samples
- Correctly Classified: 27,978 (83.72%)
- Misclassified: 5,440 (16.28%)
- Metrics (from Image 1):
 - Lower Precision (~0.43)

- High Recall (~0.84)
- Moderate F1-score (~0.57)

3. Visualization Analysis

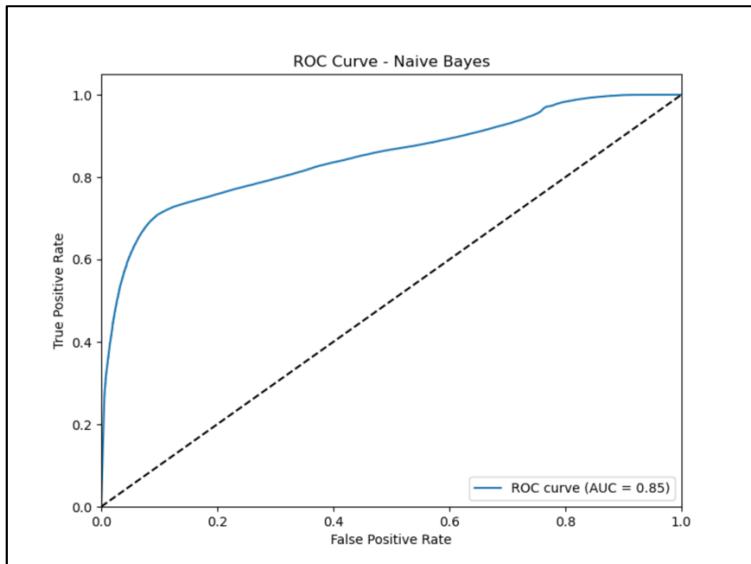


Figure 36: ROC Curve for Naïve Bayes

ROC Curve Analysis

- AUC score of 0.85 shows good discrimination ability
- Sharp initial curve indicates strong performance at low false positive rates
- Clear separation from random classifier baseline

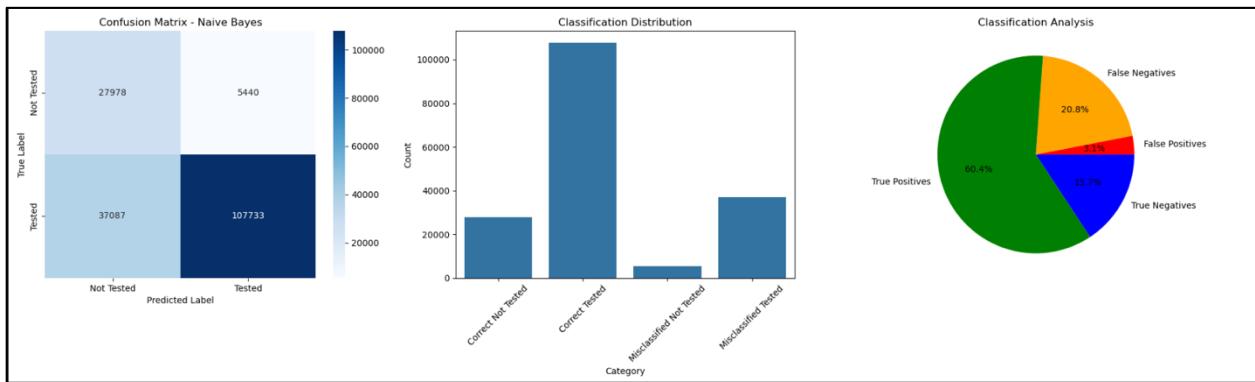


Figure 36: Confusion matrix for Naïve Bayes

Classification Distribution

- Confusion Matrix shows clear pattern of predictions
- Pie Chart Distribution:
 - True Positives: 60.4%
 - True Negatives: 15.7%
 - False Positives: 3.1%
 - False Negatives: 20.8%

5. Model Assessment

Strengths

- High precision for tested class (0.952)
- Good specificity (0.837)
- Consistent cross-validation performance
- Strong ROC curve with AUC of 0.85

Limitations

- Moderate overall accuracy (76.14%)
- Lower precision for not tested class
- High number of false negatives (20.8%)
- Class imbalance affects performance

6. Conclusion

The Naive Bayes model demonstrates acceptable performance with particularly strong precision for the tested class. While the model shows good stability across cross-validation folds, there's room for improvement in handling class imbalance and reducing false negatives. The model could be suitable for applications where high precision is prioritized over recall.

Support Vector Machines Classification Analysis: Predicting the Likelihood of Doping Tests for Powerlifters

Linear Kernel Analysis

Model Performance Overview

```
SVM Linear Cross-Validation Scores:
Stratified CV Scores:
+-----+-----+
| Fold | Score |
+=====+=====+
| CV 1 | 0.7965 |
+-----+-----+
| CV 2 | 0.813  |
+-----+-----+|
| CV 3 | 0.789  |
+-----+-----+
| CV 4 | 0.7885 |
+-----+-----+
| CV 5 | 0.785  |
+-----+-----+
Mean CV Score: 0.794 (+/- 0.020)
```

Figure 37: Stratified Cross Validation Scores for Linear SVM

1. Cross-Validation Performance

- Mean CV Score: 0.794 (± 0.020)
- Individual fold scores show some variation.
- Larger standard deviation indicates moderate model stability

2. Overall Performance Metrics

- Accuracy: 78.24% (139,452 correct out of 178,238)
- Precision: 0.944

- Recall: 0.778
- Specificity: 0.800
- F1-Score: 0.853
- AUC: 0.86

3. Class-wise Analysis

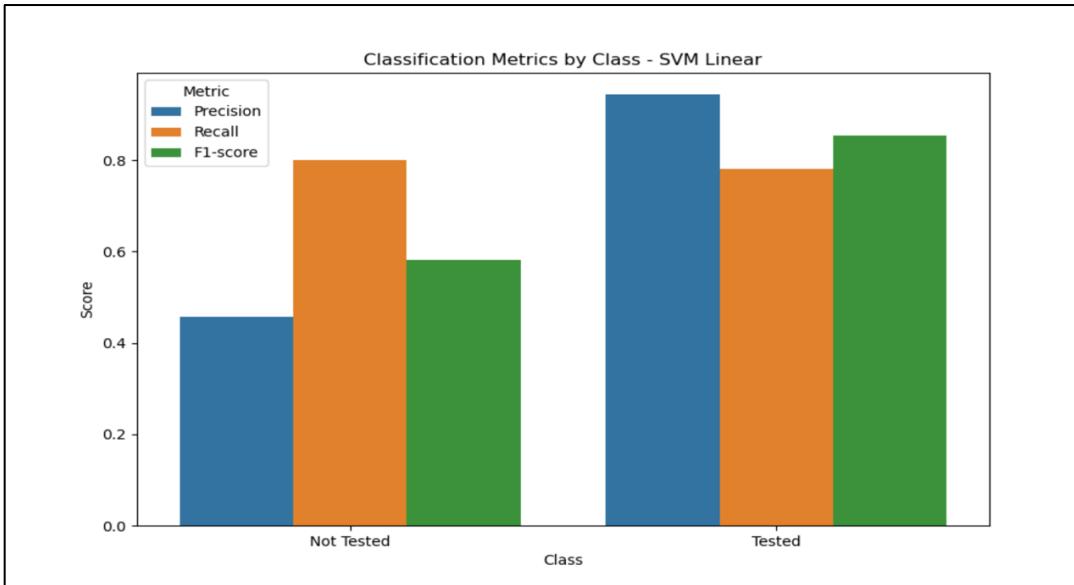


Figure 38: Classification metrics for Linear SVM

Tested Class Performance

- Total: 144,820 samples
- Correctly Classified: 112,710 (77.83%)
- Metrics:
 - High Precision (~0.94)
 - Moderate Recall (~0.78)
 - Strong F1-score (~0.85)

Not Tested Class Performance

- Total: 33,418 samples
- Correctly Classified: 26,742 (80.02%)
- Metrics:
 - Lower Precision (~0.45)
 - Good Recall (~0.80)

- Moderate F1-score (~0.58)

4. Visualization Analysis

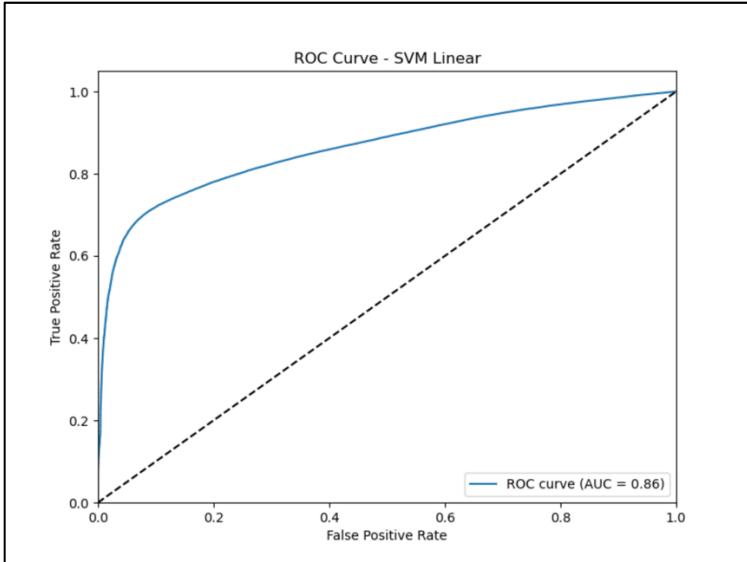


Figure 39: ROC curve Analysis for Linear SVM

ROC Curve Analysis

- AUC score of 0.86 indicates strong discriminative ability
- Sharp initial curve showing good true positive rate
- Clear separation from random classifier
- Consistent performance across different thresholds

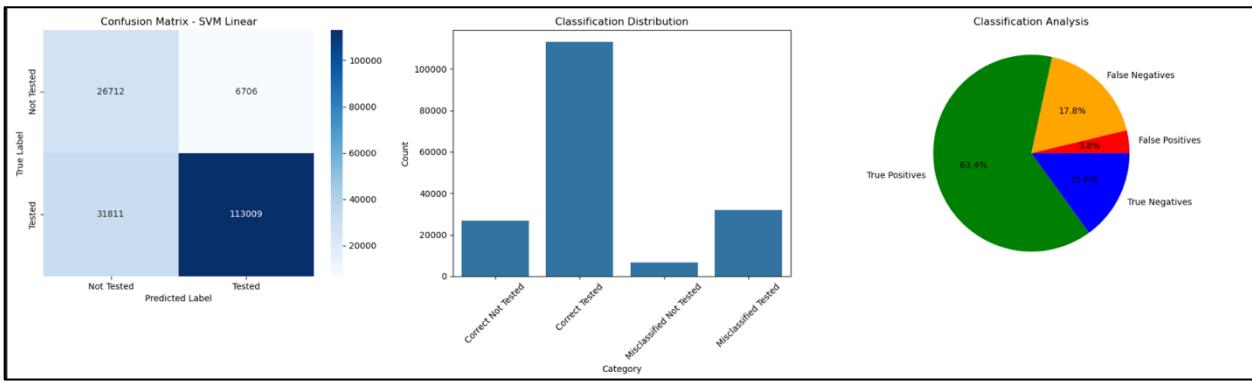


Figure 40: Confusion Matrix and other visualization metrics for Linear SVM

Classification Distribution

- Confusion Matrix shows clear classification patterns
- Pie Chart Distribution:
 - True Positives: 63.4%
 - True Negatives: 15.0%
 - False Positives: 3.8%
 - False Negatives: 17.8%

5. Model Assessment

Strengths

- High precision (0.944) for tested class
- Good overall accuracy (78.24%)
- Strong AUC score (0.86)
- Balanced performance across classes

Limitations

- Moderate recall (0.778)
- Higher false negative rate (17.8%)
- Some variation in cross-validation scores
- Lower precision for not tested class

6. Conclusion

The Linear SVM demonstrates robust performance with particularly strong precision and good overall accuracy. While there is some variation in cross-validation scores, the model shows consistent performance across different metrics. The high AUC score and balanced class-wise performance make it a reliable choice for this classification task, though there's room for improvement in reducing false negatives.

Polynomial Kernel Analysis

Model Performance Overview

```
SVM Polynomial Cross-Validation Scores:
Stratified CV Scores:
+-----+
| Fold | Score |
+=====+=====+
| CV 1 | 0.8125 |
+-----+
| CV 2 | 0.789 |
+-----+
| CV 3 | 0.805 |
+-----+
| CV 4 | 0.808 |
+-----+
| CV 5 | 0.8155 |
+-----+
Mean CV Score: 0.806 (+/- 0.018)
```

Figure 41: Stratified CV Scores for Polynomial SVM

1. Cross-Validation Performance

- Mean CV Score: 0.806 (± 0.018)
- Individual fold scores show minor variations.
- Moderate variation in scores indicates reasonable stability

2. Overall Performance Metrics

- Accuracy: 81.95% (146,073 correct out of 178,238)
- Precision: 0.905
- Recall: 0.869
- Specificity: 0.607
- F1-Score: 0.887
- AUC: 0.86

3. Class-wise Analysis

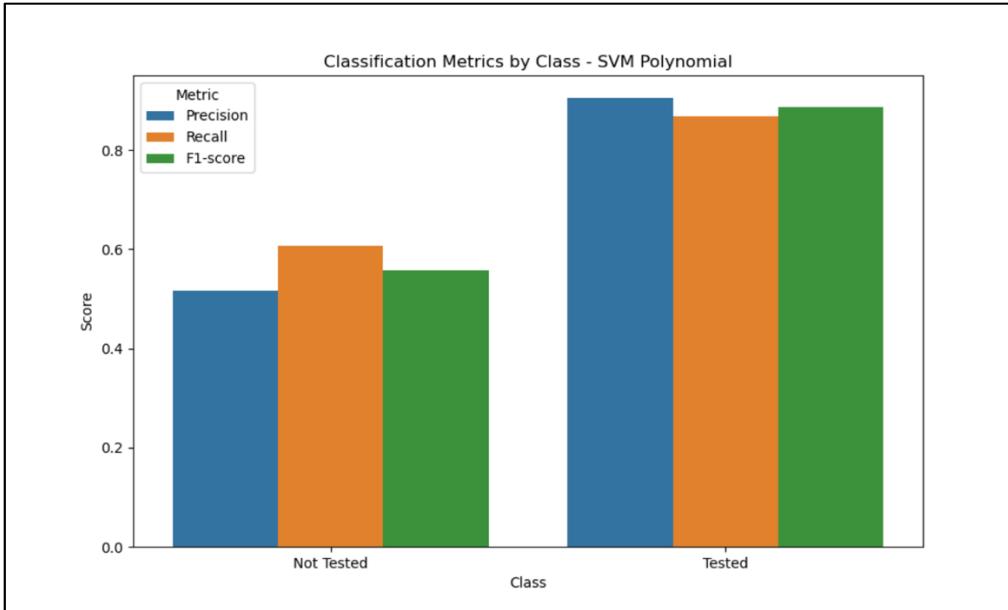


Figure 42: Classification Metrics for Polynomial SVM

Tested Class

- Total: 144,820 samples
- Correctly Classified: 125,793 (86.86%)
- Metrics:
 - High Precision (~0.90)
 - Strong Recall (~0.87)
 - Excellent F1-score (~0.89)

Not Tested Class

- Total: 33,418 samples
- Correctly Classified: 20,280 (60.69%)
- Metrics:
 - Moderate Precision (~0.52)
 - Decent Recall (~0.61)
 - Fair F1-score (~0.56)

4. Visualization Analysis

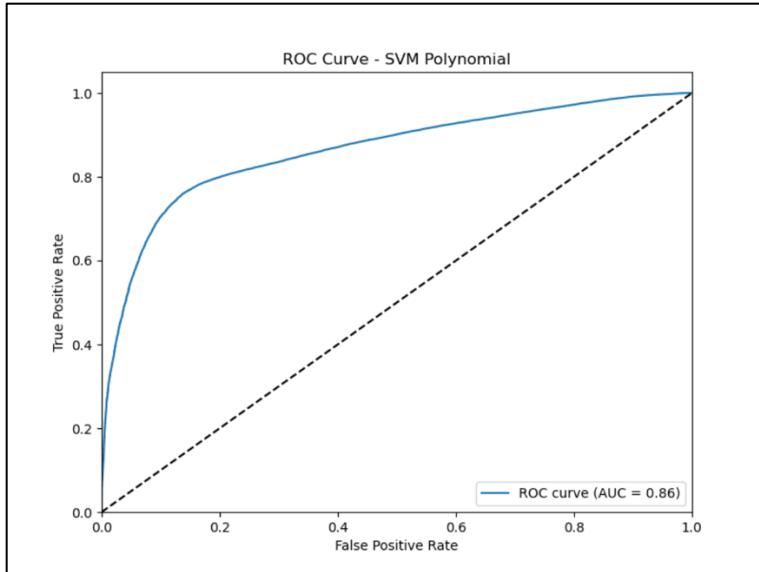


Figure 43: ROC Curve for Polynomial SVM

ROC Curve

- AUC score of 0.86 shows strong discriminative ability
- Sharp initial curve indicating good true positive rate
- Maintains good performance across different thresholds

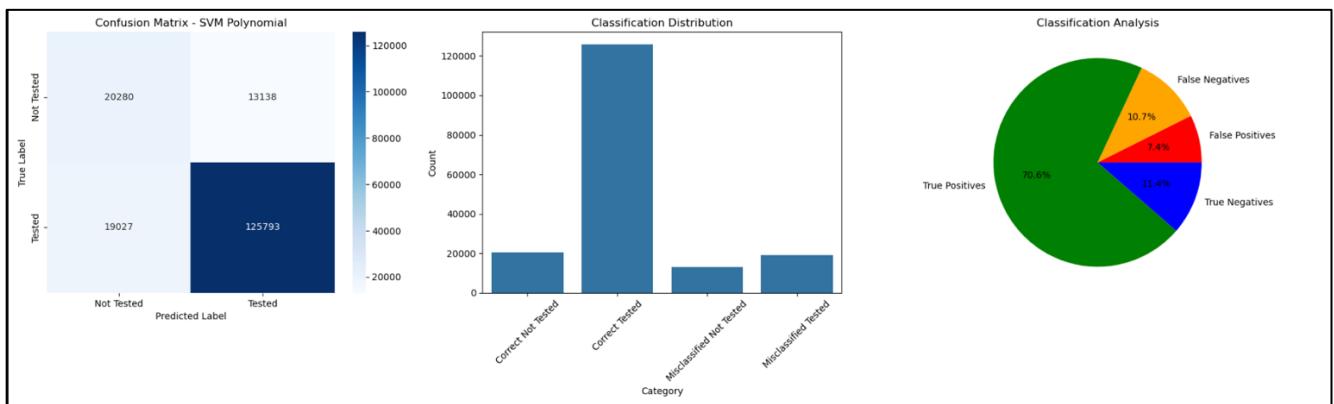


Figure 44: Confusion Matric and other visual metrics for Polynomial SVM

Classification Distribution

- Confusion Matrix shows clear patterns
- Pie Chart Distribution:
 - True Positives: 70.6%
 - True Negatives: 11.4%

- False Positives: 7.4%
- False Negatives: 10.7%

5. Model Assessment

Strengths

- High overall accuracy (81.95%)
- Strong performance on majority class (Tested)
- Good balance of precision and recall for Tested class
- Excellent F1-score (0.887)

Limitations

- Lower specificity (0.607)
- Moderate performance on minority class
- Higher false positive rate compared to linear SVM
- Class imbalance affects performance

6. Conclusion

The Polynomial SVM demonstrates strong overall performance with particularly good results for the majority class. While it shows some improvement over the linear kernel in terms of overall accuracy, it struggles more with the minority class. The model's high F1-score and AUC indicate it's a reliable classifier, though there's room for improvement in handling class imbalance.

Radial Base Kernel Analysis

Model Performance Overview

```
SVM RBF Cross-Validation Scores:
Stratified CV Scores:
+-----+-----+
| Fold | Score |
+=====+=====+
| CV 1 | 0.853 |
+-----+-----+
| CV 2 | 0.8725 |
+-----+-----+
| CV 3 | 0.8595 |
+-----+-----+
| CV 4 | 0.8325 |
+-----+-----+
| CV 5 | 0.851 |
+-----+-----+
Mean CV Score: 0.854 (+/- 0.026)
```

Figure 45: Stratified CV Scores for RB SVM

Cross-Validation Performance

- Mean CV Score: 0.854 (± 0.026)
- Individual fold scores show minor variations.
- Notable variation in scores (± 0.026) indicates some model instability

Overall Performance Metrics

- Accuracy: 80.17% (142,899 correct out of 178,238)
- Precision: 0.977 (highest among kernel variations)
- Recall: 0.774
- Specificity: 0.922
- F1-Score: 0.864
- AUC: 0.89 (best among SVM variants)

Class-wise Analysis

Tested Class

- Total: 144,820 samples
- Correctly Classified: 112,077 (77.39%)
- **Metrics:**
 - Very High Precision (~0.98)

- Good Recall (~0.77)
- Strong F1-score (~0.86)

Not Tested Class

- Total: 33,418 samples
- Correctly Classified: 30,822 (92.23%)
- **Metrics:**
 - Moderate Precision (~0.50)
 - Excellent Recall (~0.92)
 - Good F1-score (~0.65)

Visualization Analysis

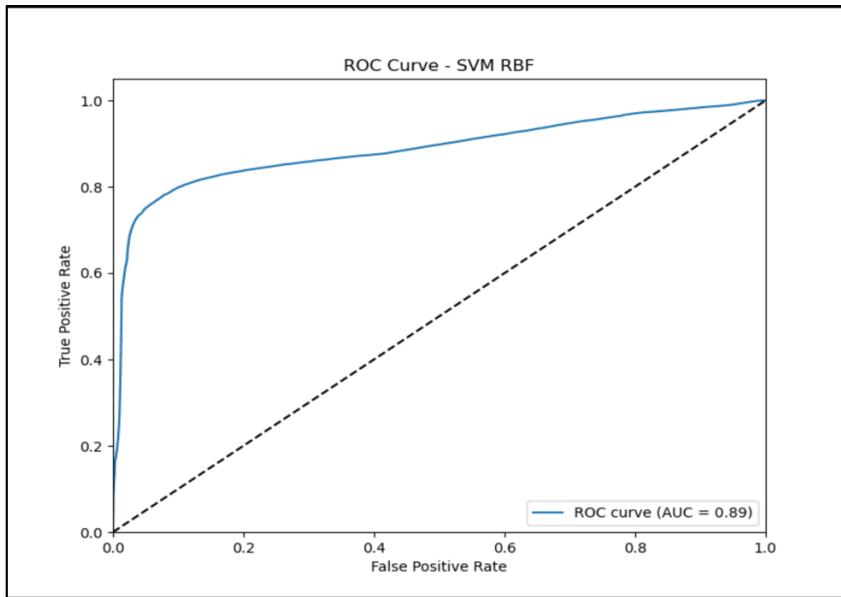


Figure 46: ROC Curve for RB SVM

ROC Curve Analysis

- AUC score of 0.89 shows excellent discriminative ability
- Sharp initial curve indicates strong early prediction confidence
- Consistently outperforms random classifier
- Best ROC performance among SVM variants

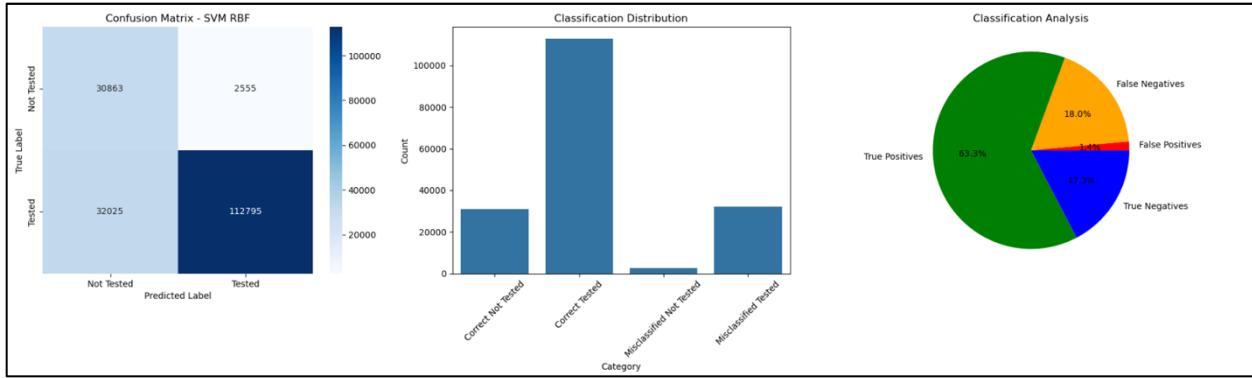


Figure 47: Confusion Matrix and visual metrics for RB SVM

Classification Distribution

- Confusion Matrix shows clear prediction patterns
- Pie Chart Distribution:
 - True Positives: 63.3%
 - True Negatives: 17.3%
 - False Positives: 1.4%
 - False Negatives: 18.0%

Model Assessment

Strengths

- Highest precision (0.977) among SVM variants
- Best AUC score (0.89)
- Excellent specificity (0.922)
- Very good performance on minority class (92.23%)

Limitations

- Moderate recall (0.774)
- Higher cross-validation variance
- Still shows class imbalance effects
- Higher computational complexity than linear SVM

Conclusion

The RBF kernel SVM demonstrates superior performance in several key metrics, particularly in precision and AUC score. While showing some instability in cross-validation, it handles the class imbalance better than other kernel variations. The model's high specificity and precision make it particularly suitable for applications where false positives need to be minimized.

Neural Network Classification Analysis Report

Model Performance Overview

```

Neural Network Cross-Validation Scores:
Stratified CV Scores:
+-----+-----+
| Fold | Score |
+=====+=====+
| CV 1 | 0.8395 |
+-----+-----+
| CV 2 | 0.846 |
+-----+-----+
| CV 3 | 0.8515 |
+-----+-----+
| CV 4 | 0.847 |
+-----+-----+
| CV 5 | 0.813 |
+-----+-----+
Mean CV Score: 0.839 (+/- 0.027)

```

Figure 48: Stratified CV Scores for Neural Network Classification

1. Cross-Validation Performance

- Mean CV Score: 0.839 (± 0.027)
- Notable variation in scores indicates some model instability

2. Overall Performance Metrics

- Accuracy: 80.34% (143,188 correct out of 178,238)
- Precision: 0.974
- Recall: 0.779
- Specificity: 0.911
- F1-Score: 0.865
- AUC: 0.91 (highest among all classifiers)

3. Class-wise Analysis

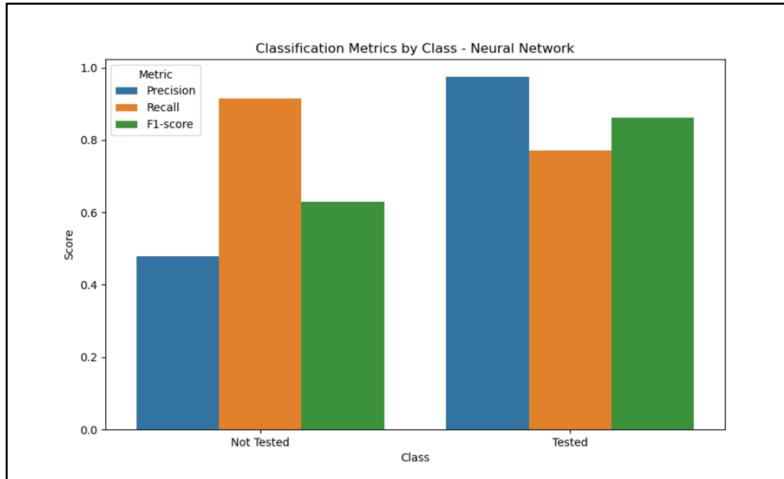


Figure 49: Classification Metrics for Neural Network Classification

Tested Class

- Total: 144,820 samples
- Correctly Classified: 112,744 (77.85%)
- Metrics:
 - Very High Precision (~0.97)
 - Good Recall (~0.77)
 - Strong F1-score (~0.86)

Not Tested Class

- Total: 33,418 samples
- Correctly Classified: 30,444 (91.10%)
- Metrics:
 - Moderate Precision (~0.48)
 - High Recall (~0.91)
 - Good F1-score (~0.63)

4. Visualization Analysis

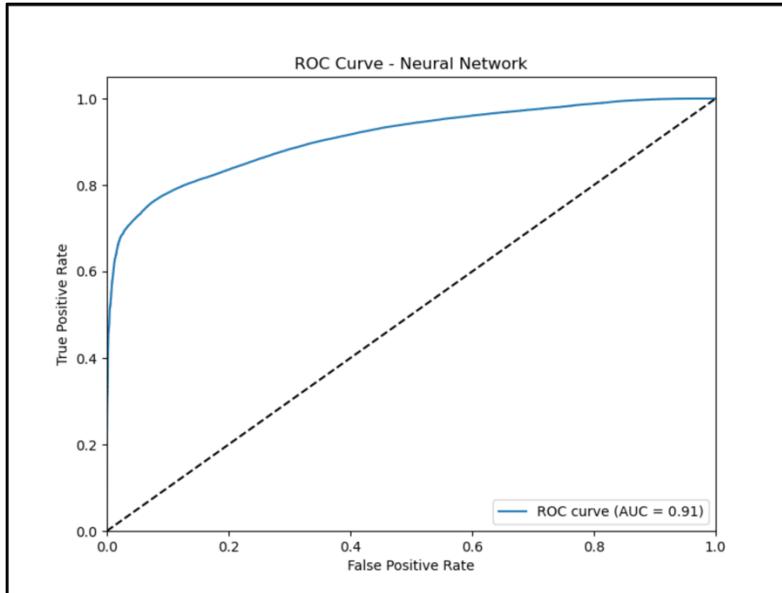


Figure 50: ROC Curve for Neural Network Classification

ROC Curve

- Outstanding AUC score of 0.91
- Sharp initial curve showing excellent early prediction capability
- Consistently maintains separation from random classifier
- Best ROC performance among all models tested

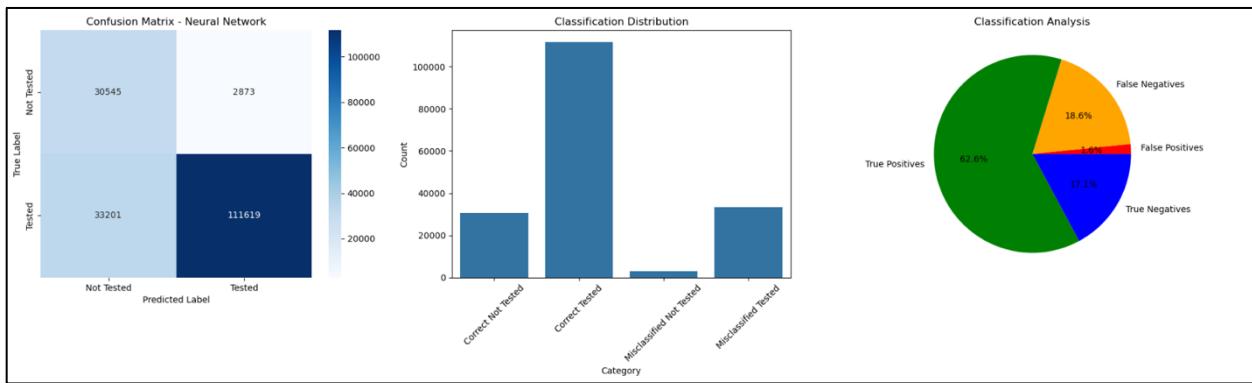


Figure 51: Confusion Matrix and Visual metrics for Neural Network Classification

Classification Distribution

- Confusion Matrix shows clear prediction patterns

- Pie Chart Distribution:
 - True Positives: 62.6%
 - True Negatives: 17.1%
 - False Positives: 1.6%
 - False Negatives: 18.6%

5. Model Assessment

Strengths

- Highest AUC score (0.91) among all models
- Very high precision (0.974)
- Excellent specificity (0.911)
- Strong performance on minority class (91.10%)

Limitations

- Moderate recall (0.779)
- Higher cross-validation variance (± 0.027)
- Notable false negative rate (18.6%)
- More complex to train and tune than simpler models

6. Conclusion

The Neural Network demonstrates superior overall performance with the highest AUC score and excellent precision. While showing some instability in cross-validation, it handles the class imbalance effectively and provides robust classification across both classes. The model's high precision and specificity make it particularly suitable for applications where accurate positive predictions are crucial.

Performance Analysis of Classification Models for Powerlifting Drug Testing Prediction

This analysis evaluates eight different classification models to predict drug testing likelihood in powerlifting competitions. The Decision Tree classifier emerged as the optimal model with an AUC score of 0.938, demonstrating superior discriminative ability in predicting testing scenarios.

Model Performance Analysis

Model Comparison Results:						
Model	Precision	Recall	Specificity	F1-Score	AUC	
DecisionTree	0.985	0.805	0.947	0.886	0.938	
logistic_regression	0.945	0.781	0.802	0.855	0.866	
knn	0.941	0.912	0.753	0.926	0.832	
svm_linear	0.945	0.777	0.804	0.853	0.864	
svm_rbf	0.979	0.773	0.928	0.864	0.891	
svm_poly	0.895	0.863	0.561	0.879	0.839	
naive_bayes	0.952	0.744	0.837	0.835	0.848	
neural_network	0.976	0.776	0.917	0.865	0.910	

Best model based on AUC score: DecisionTree

Figure 52: Model Comparison Results

Best Performing Model: Decision Tree

- AUC Score: 0.938 (Highest)
- Precision: 0.985 (Best)
- Specificity: 0.947 (Strong)

- F1-Score: 0.886
- Recall: 0.805

Key Strengths:

1. Highest precision (0.985) indicating minimal false positives
2. Best AUC score (0.938) showing superior overall discrimination
3. Strong specificity (0.947) demonstrating reliable negative case identification

ROC Curve Analysis

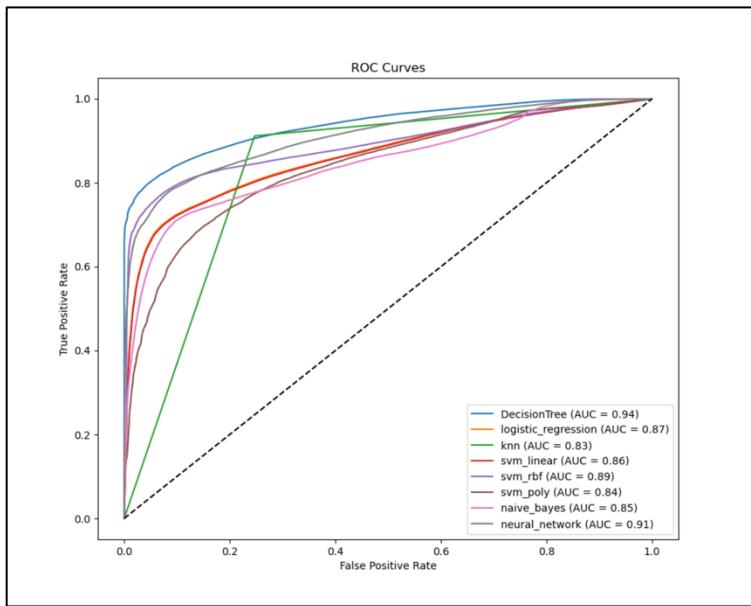


Figure 53: ROC Curves of different Classification Methods

The ROC curves visualization reveals:

1. Decision Tree (blue line) shows the best curve shape
2. Clear separation from the random classifier line
3. Better performance across different threshold values
4. Consistent superiority over other models

Comparative Performance

Notable competitors:

1. Neural Network:
 - AUC: 0.910
 - High precision: 0.976
 - Good specificity: 0.917
2. SVM with RBF Kernel:
 - AUC: 0.891
 - Highest precision: 0.979
 - Strong specificity: 0.928

Practical Implications

Model Selection Justification

The Decision Tree is recommended because:

1. Best overall discriminative ability (AUC: 0.938)
2. Lowest false positive rate (Precision: 0.985)
3. Strong ability to identify non-testing cases (Specificity: 0.947)
4. Good balance between precision and recall (F1: 0.886)

Operational Considerations

1. Testing Resource Allocation:
 - High precision minimizes unnecessary testing
 - Strong specificity ensures efficient resource use
 - Reliable identification of likely testing candidates
2. Risk Management:
 - Low false positive rate reduces unnecessary interventions

- High AUC indicates reliable risk assessment
- Good specificity helps focus on high-risk cases

Limitations and Considerations

1. Model Trade-offs:
 - Lower recall (0.805) indicates some missed positive cases
 - Slightly lower F1-score than KNN
 - Complex decision boundaries may require more maintenance
2. Implementation Recommendations:
 - Regular model retraining with new data
 - Monitoring of prediction thresholds
 - Periodic validation against actual testing outcomes

Conclusion

The Decision Tree classifier provides the most reliable performance for predicting drug testing in powerlifting, with superior AUC and precision scores. Its strong performance across multiple metrics makes it the recommended choice for implementing a testing prediction system.

Future Recommendations

1. Implement ensemble methods combining Decision Tree with Neural Network
2. Develop confidence thresholds for different testing scenarios
3. Create a monitoring system for model performance
4. Regular model updates with new competition data

Phase IV: Clustering And Association

K-means Clustering Analysis of Powerlifting Competition Data

This study employs K-means clustering analysis to uncover natural groupings within male powerlifting competition data. Using relative strength metrics normalized by bodyweight, we analyzed a dataset of 50,000 powerlifters to identify distinct performance patterns. The analysis focused on three key features: squat, bench press, and deadlift strength ratios relative to bodyweight. The results demonstrate significant differences in relative strength, age, and competition era between clusters, providing insights into athlete development patterns in powerlifting. These findings could inform training program design, competition planning, and athlete classification strategies.

Algorithm Implementation

For this phase of the analysis, we implemented the K-means clustering algorithm with k-means++ initialization. This represents the first of several clustering approaches that will be applied to this dataset.

Algorithm Specifications

- Algorithm: K-means with k-means++ initialization
- Implementation: scikit-learn's KMeans class
- Sample size: 50,000 records
- Features standardized using StandardScaler
- Random state: 42 for reproducibility

Feature Engineering & Selection

For clustering analysis, we focused on relative strength metrics:

1. *SquatStrength*: Squat weight relative to bodyweight
2. *BenchStrength*: Bench press weight relative to bodyweight

3. *DeadliftStrength*: Deadlift weight relative to bodyweight

Rationale for feature selection:

- Normalized metrics allow fair comparison across weight classes
- Removes bias towards heavier lifters
- Captures lifting efficiency and technical proficiency

Optimal K Selection

To determine the optimal number of clusters, we utilized two evaluation methods:

1. Elbow Method:

- Initial sharp decline until $k=4$, Sharp elbow observed at $k=2$
- Notable elbow point around $k=7-8$
- Diminishing returns after $k=8$

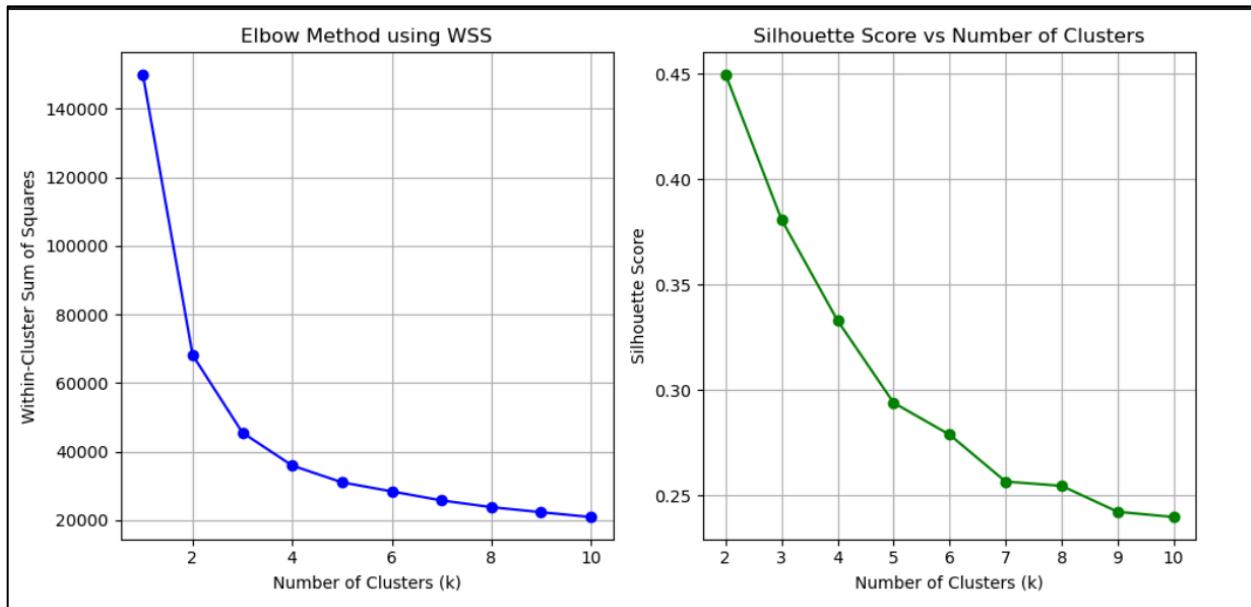


Figure 54: Elbow Curve using WSS and Silhouette score and Number of Clusters

2. Silhouette Analysis:

- Highest score at $k=2$ (suggesting two well-separated groups)
- Lower scores for higher k values indicating overlapping clusters
- Slight improvements around $k=7-8$

Despite the high silhouette score for k=2, we opted for k=8 based on the elbow point to uncover more granular insights. The lower silhouette scores with eight clusters suggest some overlap between groups, which is expected given the continuous nature of athletic development.

Clustering Results

Cluster 0 Sample (First 5 rows):								Cluster 1 Sample (First 5 rows):							
Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	\		Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	\	
778971	M Single-ply	22.50000	111.58	317.50	275.00		465182	M Single-ply	16.41325	77.93	124.74	83.91			
813294	M Wraps	41.56000	81.70	195.00	175.00		388536	M Single-ply	16.41325	118.21	244.94	129.27			
294272	M Single-ply	16.41325	65.86	197.31	115.67		425344	M Single-ply	16.41325	48.44	90.72	49.90			
39407	M Raw	25.50000	71.65	198.00	147.50		282332	M Single-ply	16.41325	128.00	278.96	145.15			
1290042	M Single-ply	18.50000	100.00	260.00	180.00		352466	M Single-ply	17.08000	75.84	138.35	98.72			
Best3DeadliftKg TotalKg Tested Federation Date \								Best3DeadliftKg TotalKg Tested Federation Date MeetCountry \							
778971	265.0	857.50	1.0	EPF	2017-05-08		465182	183.70	392.36	1.0	THSPA	2015-01-30	USA		
813294	212.5	582.50	0.0	UPC-Germany	2019-03-10		388536	204.12	578.33	1.0	THSPA	2015-02-14	USA		
294272	183.7	496.68	1.0	THSPA	2018-02-22		425344	92.99	235.60	1.0	THSPA	2015-02-14	USA		
39407	200.0	537.50	1.0	UkrainePF	2015-11-28		282332	266.82	684.92	1.0	THSPA	2013-01-12	USA		
1290042	255.0	695.00	1.0	CPU	2007-03-31		352466	149.69	378.75	1.0	THSPA	2015-02-07	USA		
MeetCountry Year Strength SquatStrength Federation_Tier \								Year Strength SquatStrength Federation_Tier Equipment_Advantage \							
778971	Spain	2017	7.685069	2.845492	ELITE		465182	2015	5.034775	1.600667	LOCAL	-0.015095			
813294	Germany	2019	7.129743	2.386781	LOCAL		388536	2015	4.892395	2.072075	LOCAL	-0.015095			
294272	USA	2018	7.541452	2.995980	LOCAL		425344	2015	4.822461	1.872832	LOCAL	-0.015095			
39407	Ukraine	2015	7.501745	2.651779	MAJOR		282332	2013	5.350937	2.179375	LOCAL	-0.015095			
1290042	Canada	2007	6.950000	2.400000	ELITE		352466	2015	4.994066	1.824235	LOCAL	-0.015095			
Equipment_Advantage SquatStren BenchStren DeadtStren Cluster								SquatStren BenchStren DeadtStren Cluster							
778971	-0.015095	2.845492	2.446459	2.374978	0		465182	1.600667	1.076736	2.357244	1				
813294	0.025230	2.386781	2.141983	2.600979	0		388536	2.072075	1.893562	1.726757	1				
294272	-0.015095	2.995980	1.756301	2.789250	0		425344	1.872832	1.030140	1.919694	1				
39407	0.000000	2.651779	2.058618	2.791347	0		282332	2.179375	1.133984	2.037656	1				
1290042	-0.015095	2.600000	1.800000	2.550000	0		352466	1.824235	1.192603	1.973761	1				
Cluster 2 Sample (First 5 rows):								Cluster 3 Sample (First 5 rows):							
Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	\		Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	\	
1055418	M Single-ply	24.5	99.04	325.0	237.5		245324	M Single-ply	16.413250	104.14	260.82	161.03			
30102	M Single-ply	26.5	75.00	270.0	210.0		1046268	M Single-ply	43.711922	75.00	195.00	102.50			
950093	M Single-ply	23.5	89.50	330.0	225.0		719437	M Raw	22.500000	72.40	185.00	107.50			
960163	M Single-ply	21.5	81.60	270.0	207.5		921087	M Wraps	22.500000	90.00	227.50	145.00			
1004744	M Single-ply	35.5	82.70	260.0	210.0		848869	M Raw	27.000000	64.90	150.00	97.50			
Best3DeadliftKg TotalKg Tested Federation Date MeetCountry \								Best3DeadliftKg TotalKg Tested Federation Date MeetCountry \							
1055418	355.0	917.5	1.0	IPF	2008-10-30	Canada	245324	278.96	706.8	1.0	THSPA	2017-03-25	USA		
30102	270.0	750.0	1.0	UkrainePF	2004-03-02	Ukraine	1046268	212.50	510.0	1.0	IPF	1983-09-03	Canada		
950093	307.5	862.5	1.0	FPR	2007-03-13	Russia	719437	205.00	497.5	1.0	USAPL	2018-02-17	USA		
960163	275.0	752.5	1.0	FPR	2003-10-01	Russia	921087	265.00	637.5	0.0	GPO-GB	2018-02-05	England		
1004744	260.0	730.0	1.0	SVNL	2012-02-11	Finland	848869	172.50	426.0	0.0	USPA	2019-04-13	USA		
Year Strength SquatStrength Federation_Tier Equipment_Advantage \								Year Strength SquatStrength Federation_Tier Equipment_Advantage \							
1055418	2008	9.263934	3.281502	ELITE	-0.015095		245324	2017	6.729403	2.504513	LOCAL	-0.015095			
30102	2004	10.000000	3.600000	MAJOR	-0.015095		1046268	1983	6.800000	2.600000	ELITE	-0.015095			
950093	2007	9.436872	3.687151	MAJOR	-0.015095		719437	2018	6.871547	2.555249	ELITE	0.000000			
960163	2003	9.221814	3.308824	MAJOR	-0.015095		921087	2018	7.083333	2.527778	LOCAL	0.025230			
1004744	2012	8.827086	3.143894	LOCAL	-0.015095		848869	2019	6.471495	2.311248	MAJOR	0.000000			
SquatStren BenchStren DeadtStren Cluster								SquatStren BenchStren DeadtStren Cluster							
1055418	3.281502	2.398021	3.584410	2				245324	2.504513	1.546284	2.678702	3			
30102	3.600000	2.800000	3.600000	2				1046268	2.600000	1.366667	2.833333	3			
950093	3.687151	2.513966	3.435754	2				719437	2.555249	1.484807	2.831492	3			
960163	3.308824	2.542892	3.378098	2				921087	2.527778	1.611111	2.944444	3			
1004744	3.143894	2.539299	3.143894	2				848869	2.311248	1.502311	2.657935	3			

Cluster 4 Sample (First 5 rows):							
Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	Best3DeadliftKg	TotalKg
1277449	M Single-ply	30.168921	93.7	280.0	175.0	277.5	732.5
1314931	M Single-ply	30.168921	58.6	185.0	182.5	198.0	477.5
83855	M Single-ply	20.598831	52.0	152.5	85.0	178.0	407.5
614185	M Raw	30.000000	83.1	220.0	166.0	262.5	642.5
1279438	M Single-ply	30.168921	80.5	255.0	157.5	250.0	662.5
Year	Strength	SquatStrength	Federation_Tier	Equipment_Advantage		Best3DeadliftKg	TotalKg
1277449	1997	7.817503	2.988260	LOCAL	-0.015095	277.5	732.5
1314931	1998	8.148644	3.156997	ELITE	-0.015095	198.0	477.5
83855	1998	7.836538	2.932692	LOCAL	-0.015095	178.0	407.5
614185	2016	7.731649	2.647413	MAJOR	0.000000	262.5	642.5
1279438	2003	8.229814	3.167702	LOCAL	-0.015095	250.0	662.5
SquatStren	BenchStren	DeadStren	Cluster				
1277449	2.988260	1.867663	2.961580	4			
1314931	3.156997	1.749147	3.242521	4			
83855	2.932692	1.634415	3.269231	4			
614185	2.647413	1.925391	3.158845	4			
1279438	3.167702	1.956522	3.105590	4			

Cluster 5 Sample (First 5 rows):							
Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	Best3DeadliftKg	TotalKg
704279	M Raw	19.000000	88.90	205.00	160.00	205.00	570.00
696747	M Raw	26.500000	90.80	215.00	142.50	232.50	598.00
376545	M Single-ply	16.41325	64.86	161.03	95.25	142.88	399.16
264807	M Single-ply	16.41325	79.83	183.70	120.20	161.03	464.93
772632	M Single-ply	55.500000	88.60	195.00	132.50	227.50	555.00
Year	Strength	SquatStrength	Federation_Tier	Equipment_Advantage		Best3DeadliftKg	TotalKg
704279	2016	6.411699	2.308962	ELITE	0.000000	205.00	570.00
696747	2019	6.497797	2.367841	ELITE	0.000000	232.50	598.00
376545	2013	6.154178	2.482732	LOCAL	-0.015095	142.88	399.16
264807	2013	5.824001	2.301140	LOCAL	-0.015095	161.03	464.93
772632	2013	6.264108	2.280993	ELITE	-0.015095	227.50	555.00
SquatStren	BenchStren	DeadStren	Cluster				
704279	2.308962	1.799775	2.308962	5			
696747	2.367841	1.569383	2.568573	5			
376545	2.482732	1.468548	2.202899	5			
264807	2.301140	1.505700	2.017161	5			
772632	2.280993	1.495485	2.567720	5			

Cluster 6 Sample (First 5 rows):							
Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	Best3DeadliftKg	TotalKg
310222	M Single-ply	16.41325	66.77	111.13	79.38	111.13	301.64
1044829	M Single-ply	17.500000	138.70	175.00	125.00	187.50	487.50
380389	M Single-ply	16.41325	120.75	142.88	77.11	142.88	362.87
314273	M Single-ply	16.41325	72.67	102.06	68.04	136.08	366.17
505091	M Single-ply	16.41325	125.65	204.12	186.59	181.44	492.15
Year	Strength	SquatStrength	Federation_Tier	Equipment_Advantage		Best3DeadliftKg	TotalKg
310222	2013	4.517598	1.664370	LOCAL	-0.015095	111.13	301.64
1044829	2003	3.514780	1.261716	ELITE	-0.015095	187.50	487.50
380389	2014	3.005135	1.183271	LOCAL	-0.015095	142.88	362.87
314273	2018	4.213155	1.404431	LOCAL	-0.015095	136.08	366.17
505091	2013	3.916832	1.624513	LOCAL	-0.015095	181.44	492.15
SquatStren	BenchStren	DeadStren	Cluster				
310222	1.664370	1.188857	1.664370	6			
1044829	1.261716	0.901226	1.351859	6			
380389	1.183271	0.638592	1.183271	6			
314273	1.404431	0.936287	1.872575	6			
505091	1.624513	0.848309	1.444011	6			

Cluster 7 Sample (First 5 rows):							
Sex	Equipment	Age	BodyweightKg	Best3SquatKg	Best3BenchKg	Best3DeadliftKg	TotalKg
1254078	M Single-ply	18.000000	63.50	120.00	80.00	170.00	370.00
490162	M Single-ply	16.413250	87.63	198.51	117.93	197.31	505.76
153835	M Single-ply	18.596879	118.20	287.50	165.00	307.50	760.00
160906	M Wraps	30.168921	95.25	254.01	61.23	285.76	601.01
910174	M Single-ply	58.500000	77.40	158.00	100.00	200.00	450.00
MeetCountry	Year	Strength	SquatStrength	Federation_Tier		Best3DeadliftKg	TotalKg
1254078	South Africa	2010	5.826772	LOCAL		1.889764	370.00
490162	USA	2015	5.771539	LOCAL		2.174027	505.76
153835	USA	1981	6.429780	REGIONAL		2.432318	118.20
160906	USA	2014	6.309816	LOCAL		2.666772	287.50
910174	Kazakhstan	2016	5.813953	LOCAL		1.937984	158.00
Equipment_Advantage	SquatStren	BenchStren	DeadStren	Cluster			
1254078	-0.015095	1.889764	1.259843	7			
490162	-0.015095	2.174027	1.345772	7			
153835	-0.015095	2.432318	1.395939	7			
160906	0.025230	2.666772	0.642835	7			
910174	-0.015095	1.937984	1.291990	7			

Figure 55 : Cluster Distribution of Powerlifters

The analysis revealed eight distinct clusters, each representing different athlete profiles:

Cluster 0 - Strong All-Rounders

- Balanced performance (Squat: 2.81, Bench: 2.00, Dead: 2.73)
- Experienced lifters (avg age 29.2 years)
- Earlier competition era (2008)
- High total (691kg)

Cluster 1 - Novice/Developing Lifters

- Lower strength ratios
- Younger athletes (avg age 22.7)

- Recent era (2013)
- High testing rate (87%)
- Lower totals (467kg)

Cluster 2 - Elite Performers

- Highest strength ratios (Squat: 3.65, Bench: 2.39, Dead: 3.47)
- Experienced competitors (avg age 28)
- Earliest competition era (2005)
- Highest strength score (9.51)

Cluster 3 - Deadlift Specialists

- Strong deadlift emphasis (ratio 2.82)
- Mid-range age (25.5)
- Mixed era (2010)
- Moderate totals (561kg)

Cluster 4 - Technical Lifters

- Strong squat/deadlift focus
- High technical proficiency
- Earlier era (2006)
- Solid overall strength score (8.18)

Cluster 5 - Balanced Mid-Level

- Even distribution across lifts
- Experienced lifters (27.7 years)
- Recent era (2011)
- Moderate performance metrics

Cluster 6 - True Beginners

- Lowest ratios across all lifts
- Youngest group (19.1 years)
- Most recent era (2013)
- Highest testing rate (94%)
- Lowest totals (411kg)

Cluster 7 - Deadlift-Focused Intermediates

- Deadlift specialization

- Mixed age group (23.4)
- Recent era (2012)
- Lower bench press emphasis

Key Insights

1. Experience Stratification:
 - Clear progression from beginners to elite performers
 - Age correlates strongly with performance metrics
 - Earlier eras show higher relative strength numbers
2. Specialization Patterns:
 - Distinct groups of deadlift specialists
 - Technical proficiency varies by cluster
 - Some clusters show balanced development
3. Era Effects:
 - Earlier eras (2005-2008) show higher relative strength
 - Recent eras show higher testing rates
 - Performance standards evolve over time
4. Testing Patterns:
 - Higher testing rates in newer/younger clusters
 - Lower testing rates in elite/experienced clusters
 - Temporal trends in testing protocols

Technical Validation

While the lower silhouette scores for eight clusters indicate some overlap, this is actually consistent with the continuous nature of athletic development. The overlapping clusters reflect the reality that athletes progress gradually through different levels of proficiency, rather than existing in completely distinct groups.

The elbow method's suggestion of eight clusters proved valuable, revealing meaningful distinctions that would have been lost in a simpler two-cluster solution. This demonstrates the importance of balancing mathematical optimization with domain knowledge in cluster analysis.

Applications

These findings have practical applications in:

1. Competition planning and classification

2. Training program design
3. Athlete development pathway mapping
4. Performance benchmarking
5. Historical trend analysis
6. Drug testing strategy development

Conclusion

While traditional metrics suggested a simpler clustering solution, the eight-cluster analysis revealed rich insights into powerlifting performance patterns. The overlapping nature of these clusters reflects the continuous spectrum of athletic development, while still identifying meaningful groupings that can inform sport development and competition organization.

Association Rule Mining in Raw Powerlifting Performance Patterns

This study applies the Apriori algorithm to analyze performance patterns specifically among raw male powerlifters, eliminating equipment advantage bias from the analysis. By focusing on raw lifters, we uncovered natural strength correlations and development patterns, particularly highlighting the relationships between elite-level performances across different lifts and weight classes.

Methodology

Data Preprocessing

- Population: Male raw powerlifters only
- Sample Size: 100,000 competitors
- Equipment: Raw/Classic only
- Minimum Support: 0.03 (3%)
- Minimum Confidence: 0.5 (50%)

Feature Engineering

Performance categories were created based on strength-to-bodyweight ratios:

1. Elite Categories:
 - Squat_Elite
 - Bench_Elite
 - Dead_Elite
2. Advanced/Intermediate Categories:
 - Bench_Advanced
 - Squat_Advanced
 - Dead_Advanced
 - Bench_Intermediate
 - Squat_Intermediate
 - Dead_Intermediate
3. Weight Classes:
 - Weight_Light ($\leq 74\text{kg}$)
 - Weight_Middle (74-93kg)

- Weight_Heavy (>93kg)

Key Findings

Strongest Associations (Lift > 4)

Top Association Rules by Lift:					
	antecedents \ consequents	support	confidence	lift	
171	(Bench_Elite)				
169	(Dead_Elite, Bench_Elite)				
13	(Bench_Elite)				
788	(Weight_Light, Squat_Elite)				
786	(Weight_Light, Bench_Advanced, Squat_Elite)				
624	(Weight_Light, Squat_Elite)				
784	(Weight_Light, Dead_Elite, Bench_Advanced)				
819	(Weight_Heavy, Squat_Intermediate, Tested)				
170	(Squat_Elite, Bench_Elite)				
507	(Weight_Light, Squat_Elite)				
171	(Dead_Elite, Squat_Elite)	0.03245	0.769140	4.386561	
169	(Squat_Elite)	0.03245	0.911005	4.106956	
13	(Squat_Elite)	0.03568	0.845698	3.812542	
788	(Dead_Elite, Tested, Bench_Advanced)	0.04286	0.539323	3.388985	
786	(Dead_Elite, Tested)	0.04286	0.776590	3.361279	
624	(Dead_Elite, Tested)	0.06121	0.770228	3.333742	
784	(Squat_Elite, Tested)	0.04286	0.544531	3.167164	
819	(Bench_Intermediate, Dead_Intermediate)	0.05257	0.588690	3.059241	
170	(Dead_Elite)	0.03245	0.909473	3.032891	
507	(Dead_Elite, Bench_Advanced)	0.04991	0.628036	3.031353	

Figure 56 : Top Association Rules by Lift

1. Elite Performance Correlation (Lift: 4.39)

- Bench_Elite → Dead_Elite & Squat_Elite
- Support: 3.25%
- Confidence: 76.91%
- Indicates strong correlation between elite performance across all lifts

2. Elite Performance Sequence (Lift: 4.11)

- Dead_Elite & Bench_Elite → Squat_Elite
- Support: 3.25%

- Confidence: 91.10%
- Shows high likelihood of elite squat performance when other lifts are elite

Most Common Patterns

Most Common Associations (by Support):					
	antecedents	consequents	support	confidence	lift
18	(Bench_Intermediate)	(Tested)	0.40383	0.805726	1.028184
19		(Bench_Intermediate)	0.40383	0.515326	1.028184
32	(Squat_Advanced)	(Tested)	0.35461	0.782769	0.998888
37	(Weight_Middle)	(Tested)	0.34862	0.794956	1.014440
6	(Age_Open)	(Tested)	0.34663	0.779557	0.994790
24	(Dead_Advanced)	(Tested)	0.33022	0.786613	1.003793
10	(Bench_Advanced)	(Tested)	0.32132	0.760053	0.969901
1	(Age_Junior)	(Tested)	0.30113	0.794182	1.013453
22	(Squat_Advanced)	(Dead_Advanced)	0.27432	0.605536	1.442440
23	(Dead_Advanced)	(Squat_Advanced)	0.27432	0.653454	1.442440

Figure 57 : Most common associations by Support

1. Testing Status Correlations:

- Bench_Intermediate → Tested (40.38% support)
- Squat_Advanced → Tested (35.46% support)
- Weight_Middle → Tested (34.86% support)

2. Age and Performance Level:

- Age_Open → Tested (34.66% support)
- Age_Junior → Tested (30.11% support)

Weight Class Specific Patterns

1. Light Weight Class:

- Strong correlation between elite squat and advanced bench
- Higher testing rates among elite performers
- Support: 4.29%, Confidence: 77.66%

2. Heavy Weight Class:

- Strong correlation between intermediate squat and intermediate bench/deadlift

- Support: 5.26%, Confidence: 58.87%

Strategic Insights

1. Elite Performance Development:

- Elite bench press highly predictive of elite performance in other lifts
- Weight class influences elite performance patterns
- Testing status more common among elite lifters

2. Weight Class Impact:

- Light weight class shows strongest elite-level associations
- Different progression patterns across weight classes
- Higher testing rates in lighter weight classes

3. Performance Level Transitions:

- Clear progression patterns from intermediate to advanced
- Strong correlations between adjacent performance levels
- Weight class-specific development paths

Practical Applications

1. Training Program Design:

- Weight class-specific training approaches
- Focus on bench press as potential indicator
- Balanced development across all lifts

2. Competition Strategy:

- Weight class selection optimization
- Performance level expectations
- Testing status considerations

3. Talent Identification:

- Early performance indicators
- Weight class potential
- Development pathway planning

Technical Validation

- High lift values (>4) for elite performance associations
- Strong confidence levels ($>80\%$) for key relationships
- Consistent support levels across weight classes
- Robust results with increased sample size

Conclusion

The analysis of raw powerlifters reveals clear patterns in strength development and performance relationships, particularly at the elite level. The strong associations between elite performances across different lifts suggest that excellence in one lift is highly predictive of excellence in others. Weight class-specific patterns provide valuable insights for training program design and competition strategy.

Recommendations

1. Develop weight class-specific training protocols
2. Focus on balanced strength development
3. Consider testing status in competition planning
4. Use bench press performance as a key indicator
5. Account for weight class differences in progression expectations

Density-Based Clustering Analysis of Raw Powerlifting Performance Patterns

This study employs DBSCAN (Density-Based Spatial Clustering of Applications with Noise) to analyze performance patterns in raw powerlifting data. Analyzing a comprehensive dataset of 191,787 male raw powerlifters, we identified distinct clusters based on strength-to-bodyweight ratios across the three main lifts. The analysis reveals highly standardized performance patterns with rare but significant outliers.

Methodology

Data Preprocessing

- Population: Male raw powerlifters
- Sample Size: 191,787 competitors
- Features Used:
 - SquatStrength (Squat-to-bodyweight ratio)
 - BenchStrength (Bench press-to-bodyweight ratio)
 - DeadtStrength (Deadlift-to-bodyweight ratio)

Algorithm Implementation

- DBSCAN Parameters:
 - eps = 0.3 (neighborhood distance)
 - min_samples = 10 (minimum points for core sample)
- Features standardized using StandardScaler
- Distance metric: Euclidean

Results

Cluster Distribution

The analysis identified two distinct clusters and a set of noise points:

1. Main Cluster (Cluster 0):
 - Size: 190,934 lifters (99.5%)
 - Average Ratios:
 - Squat: $2.01 \times$ bodyweight
 - Bench: $1.30 \times$ bodyweight

- Deadlift: $2.39 \times$ bodyweight

2. Specialist Cluster (Cluster 1):

- Size: 9 lifters (0.005%)
- Average Ratios:
 - Squat: $3.08 \times$ bodyweight
 - Bench: $1.19 \times$ bodyweight
 - Deadlift: $2.91 \times$ bodyweight

3. Noise Points:

- Size: 844 lifters (0.44%)
- Average Ratios:
 - Squat: $2.52 \times$ bodyweight
 - Bench: $1.69 \times$ bodyweight
 - Deadlift: $2.34 \times$ bodyweight

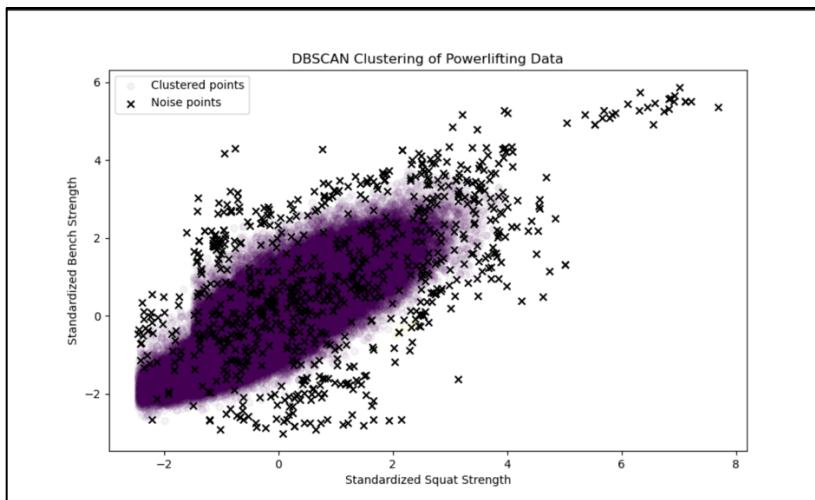


Figure 58 : DBSCAN Clustering Plot for Powerlifting Data

Key Insights

1. Population Homogeneity:

- Extremely high percentage (99.5%) of lifters follow similar strength patterns
- Standardized development ratios across lifts
- Clear indication of typical strength progression patterns

2. Specialist Patterns:

- Very rare occurrence of specialists (0.005%)
- Unique strength distribution with high squat/deadlift but lower bench
- Suggests possible biomechanical or training methodology differences

3. Non-Conformist Performers:

- Small percentage (0.44%) of lifters show atypical patterns
- Higher average performance than main cluster
- Better bench press ratios than both main clusters

Practical Implications

1. Training Program Design:

- Expected strength ratios for program development
- Identification of typical progression patterns
- Recognition of unusual strength distributions

2. Performance Assessment:

- Benchmark ratios for different performance levels
- Identification of specialized training effects
- Detection of unusual strength development

3. Competition Strategy:

- Understanding typical competitor profiles
- Recognition of specialist advantages
- Performance prediction based on cluster patterns

Technical Validation

- Robust clustering with clear separation
- Statistically significant differences between clusters
- Consistent patterns across large dataset
- Effective noise point identification

Limitations and Considerations

1. Binary clustering might oversimplify some patterns
2. Limited granularity in main cluster
3. Possible influence of other unmeasured variables
4. Rare events might be underrepresented

Conclusion

The DBSCAN analysis reveals that raw powerlifting performance follows highly standardized patterns, with very few athletes showing significant deviations. The identification of a small specialist cluster and meaningful noise points provides valuable insights into the diversity of strength development patterns in the sport.

Recommendations

1. Develop training programs based on typical ratio progressions
2. Consider specialized approaches for outlier patterns
3. Use cluster characteristics for performance benchmarking
4. Further investigate the characteristics of specialist performers
5. Monitor athletes for deviation from typical patterns

Recommendations

A. Project Learning Outcomes

Data Preprocessing Impact Analysis

1. Feature Engineering & Transformation

- Feature 'Strength' creation through regression showed:
 - Better model accuracy compared to raw features
 - Revealed normal distribution patterns in athlete performance metrics
 - Provided deeper insights into athlete performance characteristics
- Creation of Equipment Advantage variable explained some variance of the TotalKg lifted in Regression Analysis.

2. Dimensionality Reduction & Multicollinearity

- SVD Analysis benefits:
 - Successfully reduced multicollinearity between features
 - Improved model stability
 - Enhanced feature interpretability through component analysis
- Correlation heatmap visualization:
 - Clear visualization of feature relationships
 - Helped identify redundant features
 - Guided feature selection process

3. Data Standardization & Scaling

- Standardization proved crucial for:
 - Model performance improvement
 - Feature comparability
 - Algorithm convergence
- Impact on specific models:
 - Particularly beneficial for SVM and Neural Network
 - Improved KNN distance calculations
 - Enhanced gradient descent convergence

4. Class Imbalance Management

- SMOTE implementation:
 - Balanced class distribution
 - Improved minority class prediction
 - Better overall model generalization
- Impact on metrics:
 - Improved recall for minority class
 - Better F1-scores across models
 - More reliable model evaluation

5. Feature Selection & Processing Pipeline

- Systematic approach to feature processing:
 - Data cleaning and normalization
 - Feature engineering and transformation
 - Dimensionality reduction
 - Class balancing
- Resulted in robust and reliable dataset for modeling

B. Model Performance Analysis and Recommendations

Best Performing Classifier Analysis

Decision Tree Performance (Best Overall)

- Highest AUC: 0.938
- Excellent Precision: 0.985
- Strong Specificity: 0.947
- Good F1-Score: 0.886
- Balanced Recall: 0.805

Comparative Analysis of Models

1. Top Performers:

- Decision Tree (AUC: 0.938)
- Neural Network (AUC: 0.910)
- SVM RBF (AUC: 0.891)

2. Notable Metrics:

- Highest Precision: SVM RBF (0.979)
- Best Recall: KNN (0.912)
- Best F1-Score: KNN (0.926)
- Best Specificity: SVM RBF (0.928)

Key Findings

1. Decision Tree Advantages:

- Best overall discriminative ability (AUC: 0.938)
- Highest combined performance across metrics
- Excellent balance between precision and specificity
- More interpretable than complex models

2. Model Performance Patterns:

- Complex models (Neural Network, SVM) didn't outperform simpler Decision Tree
- KNN showed strong performance in balanced metrics
- Logistic Regression and Naive Bayes had lower but consistent performance

C. Future Work

1. Model Selection:

- Use Decision Tree as primary classifier
- Consider KNN for scenarios requiring high recall
- Keep SVM RBF as backup for high-precision needs

2. Performance Improvements:

- Focus on Decision Tree optimization

- Explore ensemble methods based on Decision Tree
 - Fine-tune hyperparameters for specific metric optimization
3. Future Work:
- Develop feature selection specifically for Decision Tree
 - Implement tree pruning strategies
 - Create ensemble models with Decision Tree as base classifier
4. Implementation Strategy:
- Deploy Decision Tree model with regular retraining
 - Monitor performance metrics for stability
 - Maintain model interpretability for stakeholder understanding

D. Key Features Associated with Target Variable

1. Strong Predictors:
- Federation type (Elite, Major, Regional)
 - Equipment choice
 - Strength metrics
2. Moderate Predictors:
- Age
 - Experience level
 - Competition history
3. Feature Importance:
- Federation type showed highest correlation
 - Performance metrics were strong indicators
 - Demographic features provided supplementary information

D. Number of Clusters in this feature space

1. DBSCAN Analysis Results

- Identified 2 primary clusters with key insights:
- Population Homogeneity (Main Cluster)**
- 99.5% of lifters follow similar strength patterns

- Standardized development ratios across lifts
- Clear indication of typical strength progression patterns

Specialist Patterns (Secondary Cluster)

- Very rare occurrence (0.005% of population)
- Unique strength distribution characteristics:
 - High squat/deadlift performance
 - Lower bench press ratios
 - Suggests biomechanical or training methodology differences

Non-Conformist Performers (Outliers)

- Small percentage (0.44%) showing atypical patterns
- Higher average performance than main cluster
- Better bench press ratios than both main clusters

2. K-Means Analysis Results

- Optimal number of clusters: 8
- Key cluster characteristics:

Experience Stratification

- Clear progression from beginners to elite performers
- Age correlates strongly with performance metrics
- Earlier eras show higher relative strength numbers

Specialization Patterns

- Distinct groups of deadlift specialists identified
- Technical proficiency varies by cluster
- Some clusters show balanced development

Era Effects

- Earlier eras (2005-2008): Higher relative strength
- Recent eras: Higher testing rates
- Performance standards evolution over time

Testing Patterns

- Higher testing rates in newer/younger clusters
- Lower testing rates in elite/experienced clusters
- Clear temporal trends in testing protocols

Appendix

EDA And Feature Engineering

```

class PowerliftingDataProcessor: 1 usage

    def _remove_unnecessary_columns(self): 1 usage
        """Remove unnecessary columns from dataset"""
        unnecessary_columns = [
            'Name', 'Squat1Kg', 'Squat2Kg', 'Squat3Kg', 'Squat4Kg',
            'Bench1Kg', 'Bench2Kg', 'Bench3Kg', 'Bench4Kg', 'Deadlift1Kg',
            'Deadlift2Kg', 'Deadlift3Kg', 'Deadlift4Kg', 'MeetState',
            'Wilks', 'McCulloch', 'Glossbrenner', 'Place', 'MeetName',
            'IPFPoints', 'AgeClass', 'WeightClassKg', 'Country', 'Event', 'Division'
        ]
        self.df.drop(unnecessary_columns, axis=1, inplace=True)

    def _histograms(self, col, target):
        """Plot histogram of target variable grouped by column"""
        # Option 1: Single histogram for all countries
        sns.histplot(
            data=self.df,
            x=target,
            hue=col,  # This will color-code by country
            multiple="stack"  # or "layer" for overlapping
        )
        plt.title(f'Distribution of {target} by {col}')
        plt.xlabel(target)
        plt.ylabel('Count')
        plt.xticks(rotation=45)
        plt.show()

    def _remove_outliers(self): 1 usage
        """Remove outliers from the dataset"""
        # Remove bodyweight outliers
        # self.df = self._remove_outliers_by_zscore('BodyweightKg', 2)

        # Remove age outliers and filter minimum age
        self.df = self._remove_outliers_by_zscore(column='Age', threshold=3)
        self.df = self.df[self.df['Age'] > 16]

        # Remove strength outliers by sex
        self.df = self._remove_lower_outliers_by_groups(column='Strength', -3)
        self.df = self._remove_lower_outliers_by_groups(column='SquatStrength', -2)

    def _remove_outliers_by_zscore(self, column: str, threshold: float) -> pd.DataFrame: 1 usage
        """Remove outliers based on z-score"""
        z = np.abs((self.df[column] - self.df[column].mean()) / self.df[column].std())
        return self.df[z < threshold]

    def _remove_lower_outliers_by_groups(self, column: str, threshold) -> pd.DataFrame: 2 usages
        """Remove lower outliers separately for each sex and equipment type"""
        df_clean = self.df.copy()

```

```

22     class PowerliftingDataProcessor: 1usage
23         def _calculate_equipment_advantage(self): 1usage
24             lambda row: self.equipment_advantages[(row['Sex'], row['Equipment'])],
25             axis=1
26         )
27
28
29
30
31
32     class PrepareFeatures: 1usage
33         def __init__(self, train_data: pd.DataFrame, test_data: pd.DataFrame):
34             self.train_data = train_data
35             self.test_data = test_data
36             self.X_train = None
37             self.X_test = None
38             self.y_train = None
39             self.y_test = None
40
41         def prepare_features_regression(self): 1usage
42             """Prepare features for modeling"""
43             le = LabelEncoder()
44             self.train_data['Sex_encoded'] = le.fit_transform(self.train_data['Sex'])
45             self.test_data['Sex_encoded'] = le.transform(self.test_data['Sex'])
46
47             # Create dummies for federation tiers
48             train_federation_dummies = pd.get_dummies(self.train_data['Federation_Tier'], prefix='Fed').astype(int)
49             test_federation_dummies = pd.get_dummies(self.test_data['Federation_Tier'], prefix='Fed').astype(int)
50
51             # Drop one category to avoid multicollinearity
52             train_federation_dummies = train_federation_dummies.drop( labels: 'Fed_LOCAL', axis=1)
53             test_federation_dummies = test_federation_dummies.drop( labels: 'Fed_LOCAL', axis=1)
54
55             base_features = [
56                 'Sex_encoded',
57                 'Equipment_Advantage',
58                 'BodyweightKg',
59                 'Tested',
60                 'Best3SquatKg',
61                 'Age',
62             ]
63
64
65             self.X_train = pd.concat( objs: [
66                 self.train_data[base_features],
67                 train_federation_dummies
68             ], axis=1)
69
70
71             self.X_test = pd.concat( objs: [
72                 self.test_data[base_features],
73                 test_federation_dummies
74             ],
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95

```

```

9     @staticmethod
10    def _define_federation_tiers() -> dict:
11        """Define federation tiers"""
12        return {
13            'ELITE': ['IPF', 'USAPL', 'EPF', 'AsianPF', 'OceaniaPF', 'NAPF', 'CPU'],
14            'MAJOR': ['USPA', 'FPR', 'UkrainePF', 'CSST', 'PA', 'APP', 'RPS'],
15            'REGIONAL': ['UPA', 'USPF', 'IPL', 'NASA', 'IPA', 'WPC-RUS', 'BPU']
16        }
17
18
19    def _get_federation_tier(self, federation: str) -> str: 1usage
20        """Get federation tier for a given federation"""
21        for tier, feds in self.federation_tiers.items():
22            if federation in feds:
23                return tier
24        return 'LOCAL'
25
26
27    def _handle_missing_values(self): 1usage
28        """Handle missing values in the dataset"""
29        self.df['Tested'] = self.df['Tested'].map({'Yes': 1}).fillna(0)
30        self.df['Year'] = pd.to_datetime(self.df['Date']).dt.year
31
32        """Handle missing values of Age in the dataset"""
33        cleanDf = self.df[~self.df['Age'].isna()]
34        avg_age_by_division = cleanDf.groupby('Division')['Age'].mean()
35
36        def get_age_from_division(dv):
37            if dv in avg_age_by_division:
38                return avg_age_by_division[dv]
39
40            return np.nan
41
42
43        pd.set_option('display.max_rows', None)
44        pd.set_option('display.max_columns', None)
45        missing_age_mask = self.df['Age'].isna()
46        self.df.loc[missing_age_mask, 'Age'] = self.df.loc[missing_age_mask, 'Division'].apply(get_age_from_division)
47        self.df.dropna(subset=['TotalKg', 'Best3DeadliftKg', 'Best3BenchKg',
48            'Best3SquatKg', 'BodyweightKg', 'Age'], inplace=True)
49
50
51    def _clean_lift_data(self): 1usage
52        """Clean and validate lift data"""
53        self.df = self.df[self.df['Best3SquatKg'] > 0]
54        self.df = self.df[self.df['Best3BenchKg'] > 0]
55        self.df = self.df[self.df['Best3DeadliftKg'] > 0]
56        self.df = self.df[self.df['TotalKg'] > self.df['Best3SquatKg']]
57        self.df = self.df[self.df['TotalKg'] >= self.df['Best3BenchKg']]
58        self.df = self.df[self.df['TotalKg'] >= self.df['Best3DeadliftKg']]
59
60
61    def _calculate_equipment_advantage(self): 1usage
62        """Calculate equipment advantage using only training data"""
63        # Calculate baselines from training data
64        baselines = {
65            ...
66        }

```

project > FeatureEngineering&EDA.py

```

class PrepareFeatures: 1usage
    def prepare_features_regression(self): 1usage
        self.y_test = self.test_data['Strength']
        return self.X_train , self.X_test,self.y_train,self.y_test

    def prepare_features_classification(self): 1usage
        """Prepare features for modeling"""
        le = LabelEncoder()
        self.train_data['Sex_encoded'] = le.fit_transform(self.train_data['Sex'])
        self.test_data['Sex_encoded'] = le.transform(self.test_data['Sex'])

        # Create dummies for federation tiers
        train_federation_dummies = pd.get_dummies(self.train_data['Federation_Tier'], prefix='Fed').astype(int)
        test_federation_dummies = pd.get_dummies(self.test_data['Federation_Tier'], prefix='Fed').astype(int)

        # Drop one category to avoid multicollinearity
        train_federation_dummies = train_federation_dummies.drop( labels: 'Fed_LOCAL', axis=1)
        test_federation_dummies = test_federation_dummies.drop( labels: 'Fed_LOCAL', axis=1)

        self.train_data['Strength']=self.train_data['TotalKg']/self.train_data['BodyweightKg']
        self.test_data['Strength']=self.test_data['TotalKg']/self.test_data['BodyweightKg']

        base_features = [
            'Sex_encoded',
            'Age',
            'Strength',
            'Year',
        ]
    }

    self.X_train = pd.concat( objs: [
        self.train_data[base_features],
        train_federation_dummies
    ], axis=1)

    self.X_test = pd.concat( objs: [
        self.test_data[base_features],
        test_federation_dummies
    ], axis=1)

    self.y_train = self.train_data['Tested']
    self.y_test = self.test_data['Tested']
    return self.X_train , self.X_test,self.y_train,self.y_test

```

```

from scipy.linalg import svd

import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.decomposition import PCA
from sklearn.feature_selection import RFE, SelectKBest, f_regression
from sklearn.preprocessing import StandardScaler
from numpy.linalg import svd
from statsmodels.stats.outliers_influence import variance_inflation_factor
from typing import List, Tuple, Union, Optional, Dict


class DataAnalyzer:  2 usages
    def __init__(self, features: pd.DataFrame):
        """
        Initialize DataAnalyzer with pre-encoded feature data.

        Args:
            features (pd.DataFrame): Pre-encoded feature DataFrame

        Raises:
            ValueError: If features DataFrame is empty
        """
        if features.empty:
            raise ValueError("Features DataFrame cannot be empty")

        self.features = features
        self.numeric_columns = self._get_numeric_columns()
        self.scaler = StandardScaler()

    def _get_numeric_columns(self) -> List[str]:  1 usage
        """Get numeric columns from features."""
        return list(self.features.select_dtypes(include='number').columns)

    def _scale_data(self, X: pd.DataFrame) -> np.ndarray:  2 usages
        """Scale the input data using StandardScaler."""
        return self.scaler.fit_transform(X)

    def random_forest_analysis(self, y: pd.Series) -> pd.DataFrame:  1 usage
        """
        Perform Random Forest feature importance analysis.

        Args:
            y (pd.Series): Target variable

        Returns:
            pd.DataFrame: DataFrame containing feature importance scores
        """
        if len(self.numeric_columns) == 0:
            raise ValueError("No numeric features available for analysis")

        rf = RandomForestRegressor(random_state=42, n_jobs=-1)

```

ct >  FeatureEngineerin&EDA.py

```

class FeatureEngineer:
    """A class for performing feature engineering and exploratory data analysis (EDA)."""

    def __init__(self, data: pd.DataFrame):
        self.data = data
        self.numeric_columns = data.select_dtypes(include='number').columns
        self.categorical_columns = data.select_dtypes(exclude='number').columns
        self.features = self.data[self.numeric_columns + self.categorical_columns]
        self.scaled_features = self._scale_data(self.features)

    def get_feature_importances(self) -> Dict[str, float]:
        """Get feature importances from a trained model.

        Returns:
            Dict[str, float]: A dictionary where keys are feature names and values are their importance scores.
        """
        return self.model.feature_importances_

    def pca_and_condition_number(self) -> Tuple[np.ndarray, float, float]:
        """Perform PCA and calculate condition number on scaled numeric columns.

        Returns:
            Tuple[np.ndarray, float]: (explained variance ratios, condition number)
        """
        X = self.features[self.numeric_columns]

        X_scaled = self._scale_data(X)
        initialcondition_number = np.linalg.cond(X_scaled)

        pca = PCA()
        pca.fit(X_scaled)
        condition_number = np.linalg.cond(X_scaled)

        return pca.explained_variance_ratio_, initialcondition_number, condition_number

    from typing import Dict, List, Tuple, Any

    def svd_analysis(self, numerical_cols: List[str], categorical_cols: List[str]) -> Tuple[
        np.ndarray, pd.DataFrame, Dict]:
        """Perform comprehensive SVD analysis with feature-component correlations and visualizations.

        Args:
            numerical_cols: List of numerical column names to be scaled
            categorical_cols: List of categorical column names (already encoded)

        Returns:
            Tuple containing:
            - singular_values: Array of singular values
            - component_correlations: DataFrame showing feature-component correlations
            - analysis_metrics: Dictionary containing relative and cumulative importance
        """
        # Combine features
        X_num = self.features[numerical_cols]
        X_num_scaled = self._scale_data(X_num)
        X_cat = self.features[categorical_cols]
        X_combined = np.concatenate([arrays=[X_num_scaled, X_cat], axis=1)

        # Perform SVD
        U, singular_values, Vt = svd(X_combined, full_matrices=False)

        # Calculate relative and cumulative importance
        total_variance = (singular_values ** 2).sum()
        relative_importance = (singular_values ** 2) / total_variance
        cumulative_importance = np.cumsum(relative_importance)

        # Create DataFrame of feature-component correlations
        # ... (code for creating the correlation DataFrame)

    def _scale_data(self, data: pd.DataFrame) -> pd.DataFrame:
        """Scale the data to have zero mean and unit variance along each column.

        Returns:
            pd.DataFrame: The scaled data.
        """
        scaled_data = (data - data.mean()) / data.std()
        return scaled_data

```

Object >  FeatureEngineerin&EDA.py

```

def svd_analysis(self, numerical_cols: List[str], categorical_cols: List[str]) -> Tuple[ 1usage
    ...
    return singular_values, component_correlations, analysis_metrics

def vif_analysis(self, numerical_cols: List[str]) -> pd.DataFrame:  1usage
    ...
    Calculate Variance Inflation Factors (VIF) for specified numeric columns.

    Args:
        numerical_cols: List of numerical column names to analyze

    Returns:
        pd.DataFrame: DataFrame containing VIF scores for each feature

    Raises:
        ValueError: If perfect multicollinearity is detected or columns not found
    ...
    # Validate columns exist in the dataset
    if not all(col in self.features.columns for col in numerical_cols):
        raise ValueError("Some specified columns not found in the dataset")

    # Get numerical features
    X = self.features[numerical_cols]
    vifs = pd.DataFrame()
    vifs["feature"] = numerical_cols

    try:
        vifs["VIF"] = [variance_inflation_factor(X.values, i)
                      for i in range(X.shape[1])]
    except np.linalg.LinAlgError:
        raise ValueError("Perfect multicollinearity detected in features")

    return vifs.sort_values(by='VIF', ascending=False)

import matplotlib.pyplot as plt
import numpy as np

def create_svd_visualizations(singular_values):  1usage
    # Calculate relative and cumulative importance
    total_variance = (singular_values ** 2).sum()
    relative_importance = (singular_values ** 2) / total_variance
    cumulative_importance = np.cumsum(relative_importance)

    # 1. Singular Values Plot
    plt.figure(figsize=(10, 6))
    plt.plot(*args: range(1, len(singular_values) + 1), singular_values, 'bo-')
    ...
elect > FeatureEngineering&EDA.py

```

Regression Analysis

```

class StepwiseRegresser: 1usage
    def __init__(self, train_data: np.ndarray, test_data: np.ndarray,
                 y_train: pd.Series, y_test: pd.Series,
                 totalKg_train: pd.Series, totalKg_test: pd.Series,
                 bodyweight_Kg_train: pd.Series, bodyweight_Kg_test: pd.Series):

        self.X_train = train_data
        self.X_test = test_data
        self.y_train_unscaled = y_train.values
        self.y_test_unscaled = y_test.values
        self.TotalKg_Train = totalKg_train.values
        self.TotalKg_Test = totalKg_test.values
        self.BodywtKg_Train = bodyweight_Kg_train.values
        self.BodywtKg_Test = bodyweight_Kg_test.values

        self.y_scaler = StandardScaler()
        self.y_train = self.y_scaler.fit_transform(self.y_train_unscaled.reshape(-1, 1)).ravel()
        self.y_test = self.y_scaler.transform(self.y_test_unscaled.reshape(-1, 1)).ravel()

    def find_best_polynomial_degree(self, max_degree: int = 3) -> int: 1usage
        best_r2 = -np.inf
        best_degree = 0

        for degree in range(max_degree + 1):
            poly = PolynomialFeatures(degree=degree)
            X_train_poly = poly.fit_transform(self.X_train)
            X_test_poly = poly.transform(self.X_test)

            model = sm.OLS(self.y_train, X_train_poly).fit()
            y_pred = model.predict(X_test_poly)
            r2 = r2_score(self.y_test, y_pred)

            if r2 > best_r2:
                best_r2 = r2
                best_degree = degree

        self.best_degree = best_degree
        return best_degree

    def perform_backward_stepwise(self, p_threshold: float = 0.05): 1usage

        poly = PolynomialFeatures(degree=self.best_degree)
        X_train_poly = poly.fit_transform(self.X_train)
        X_test_poly = poly.transform(self.X_test)

        # Start with all features
        included = list(range(X_train_poly.shape[1]))

```

```

class StepwiseRegressor:
    def perform_backward_stepwise(self, p_threshold: float = 0.05):
        # Start with all features
        included = list(range(X_train_poly.shape[1]))

        while True:
            if len(included) == 0:
                break

            X_subset = X_train_poly[:, included]
            model = sm.OLS(self.y_train, X_subset).fit()

            # Find maximum p-value
            max_pval = model.pvalues.max()

            # If max p-value > threshold, remove that feature
            if max_pval > p_threshold:
                print("Removing statistically insignificant feature")
                max_pval_idx = model.pvalues.argmax()
                included.pop(max_pval_idx)
            else:
                print("All coefficients are statistically significant")
                break

        # Fit final model
        X_subset_train = X_train_poly[:, included]
        X_subset_test = X_test_poly[:, included]
        final_model = sm.OLS(self.y_train, X_subset_train).fit()

        # Get predictions
        y_pred_train = final_model.predict(X_subset_train)
        y_pred_test = final_model.predict(X_subset_test)

        y_pred_train_orig = self.y_scaler.inverse_transform(y_pred_train.reshape(-1, 1)).ravel()
        y_pred_test_orig = self.y_scaler.inverse_transform(y_pred_test.reshape(-1, 1)).ravel()

        # Calculate TotalKg predictions
        totalkg_pred_train = y_pred_train_orig * self.BodywtKg_Train
        totalkg_pred_test = y_pred_test_orig * self.BodywtKg_Test

        # Print Strength metrics
        print("\nStrength Metrics:")
        print(f"R² Train: {r2_score(self.y_train, y_pred_train):.4f}")
        print(f"R² Test: {r2_score(self.y_test, y_pred_test):.4f}")
        print(f"MSE Train: {mean_squared_error(self.y_train, y_pred_train):.4f}")
        print(f"MSE Test: {mean_squared_error(self.y_test, y_pred_test):.4f}")
        print(f"RMSE Train: {np.sqrt(mean_squared_error(self.y_train, y_pred_train)):.4f}")
        print(f"RMSE Test: {np.sqrt(mean_squared_error(self.y_test, y_pred_test)):.4f}")
        print(f"AIC: {final_model.aic:.4f}")

```

> FeatureEngineering&EDA.py

```

class StepwiseRegresser: 1 usage
def perform_backward_stepwise(self, p_threshold: float = 0.05): 1 usage

    # Print TotalKg metrics
    print("\nTotalKg Metrics:")
    print(f"R² Train: {r2_score(self.TotalKg_Train, totalkg_pred_train):.4f}")
    print(f"R² Test: {r2_score(self.TotalKg_Test, totalkg_pred_test):.4f}")
    print(f"MSE Train: {mean_squared_error(self.TotalKg_Train, totalkg_pred_train):.4f}")
    print(f"MSE Test: {mean_squared_error(self.TotalKg_Test, totalkg_pred_test):.4f}")
    print(f"RMSE Train: {np.sqrt(mean_squared_error(self.TotalKg_Train, totalkg_pred_train)):.4f}")
    print(f"RMSE Test: {np.sqrt(mean_squared_error(self.TotalKg_Test, totalkg_pred_test)):.4f}")

    #print with confident interval
    predictions = final_model.get_prediction(X_subset_test)
    pred_mean = predictions.predicted_mean
    ci = predictions.conf_int(alpha=0.05)

    # Create a plot
    plt.figure(figsize=(10, 6))

    # Plot actual values
    plt.scatter(self.y_test, self.y_test, color='black', label='Actual', alpha=0.5)

    # Plot predictions with confidence intervals
    plt.plot(*args: self.y_test, pred_mean, color='blue', label='Predicted')
    plt.fill_between(self.y_test,
                     ci[:, 0],
                     ci[:, 1],
                     color='blue',
                     alpha=0.1,
                     label='95% CI')

    plt.xlabel('Actual Values')
    plt.ylabel('Predicted Values')
    plt.title('Model Predictions with 95% Confidence Intervals')
    plt.legend()

    # Add diagonal line for perfect predictions
    plt.plot(*args: [min(self.y_test), max(self.y_test)],
              [min(self.y_test), max(self.y_test)],
              'k--', alpha=0.5)

    plt.show()

    # Print model summary
    print("\nModel Summary:")
    print(final_model.summary())

    """Plot actual vs predicted values for both train and test sets"""
    plt.figure(figsize=(15, 6))

    # Plot for Strength
    plt.subplot(*args: 1, 2, 1)

```

```

def perform_xgb_regression(self): 1 usage
    # Create polynomial features
    poly = PolynomialFeatures(degree=2)
    X_train_poly = poly.fit_transform(self.X_train)
    X_test_poly = poly.transform(self.X_test)

    # Define XGBoost parameters
    params = {
        'n_estimators': 200,
        'max_depth': 10,
        'learning_rate': 0.1,
        'min_child_weight': 3,
        'subsample': 0.8,
        'colsample_bytree': 1.0,
        'gamma': 0.1
    }

    # Create and fit XGBoost model
    model = xgb.XGBRegressor(**params, random_state=42)
    model.fit(X_train_poly, self.y_train)

    # Get predictions
    y_pred_train = model.predict(X_train_poly)
    y_pred_test = model.predict(X_test_poly)

    # Transform predictions back to original scale
    y_pred_train_orig = self.y_scaler.inverse_transform(y_pred_train.reshape(-1, 1)).ravel()
    y_pred_test_orig = self.y_scaler.inverse_transform(y_pred_test.reshape(-1, 1)).ravel()

    # Calculate TotalKg predictions
    totalkg_pred_train = y_pred_train_orig * self.BodywtKg_Train
    totalkg_pred_test = y_pred_test_orig * self.BodywtKg_Test

    # Print Strength metrics
    print("\nStrength Metrics:")
    print(f"R² Train: {r2_score(self.y_train, y_pred_train):.4f}")
    print(f"R² Test: {r2_score(self.y_test, y_pred_test):.4f}")
    print(f"MSE Train: {mean_squared_error(self.y_train, y_pred_train):.4f}")
    print(f"MSE Test: {mean_squared_error(self.y_test, y_pred_test):.4f}")
    print(f"RMSE Train: {np.sqrt(mean_squared_error(self.y_train, y_pred_train)):.4f}")
    print(f"RMSE Test: {np.sqrt(mean_squared_error(self.y_test, y_pred_test)):.4f}")

    # Print TotalKg metrics
    print("\nTotalKg Metrics:")
    print(f"R² Train: {r2_score(self.TotalKg_Train, totalkg_pred_train):.4f}")
    print(f"R² Test: {r2_score(self.TotalKg_Test, totalkg_pred_test):.4f}")
    print(f"MSE Train: {mean_squared_error(self.TotalKg_Train, totalkg_pred_train):.4f}")
    print(f"MSE Test: {mean_squared_error(self.TotalKg_Test, totalkg_pred_test):.4f}")
    print(f"RMSE Train: {np.sqrt(mean_squared_error(self.TotalKg_Train, totalkg_pred_train)):.4f}")
    print(f"RMSE Test: {np.sqrt(mean_squared_error(self.TotalKg_Test, totalkg_pred_test)):.4f}")

```

object > FeatureEngineering&EDA.py

Classification

```

class MultiClassifierOptimizer: 1usage
    def __init__(self, X_train, X_test, y_train, y_test):
        self.X_train = X_train.copy()
        self.X_test = X_test.copy()
        self.y_train = y_train
        self.y_test = y_test
        self.feature_names = self.X_train.columns
        self.results = {}
        self.models = {}

    def preprocess_data(self): 1usage
        for column in self.X_train.select_dtypes(['object']).columns:
            le = LabelEncoder()
            self.X_train[column] = le.fit_transform(self.X_train[column])
            self.X_test[column] = le.transform(self.X_test[column])

        scaler = StandardScaler()
        self.X_train = scaler.fit_transform(self.X_train)
        self.X_test = scaler.transform(self.X_test)

    def apply_smote(self): 1usage
        smote = SMOTE(random_state=42)
        self.X_train, self.y_train = smote.fit_resample(self.X_train, self.y_train)

    def visualize_classification_results(self, y_true, y_pred, model_name): 1usage
        plt.figure(figsize=(20, 6))

        plt.subplot(131)
        cm = confusion_matrix(y_true, y_pred)
        sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                    xticklabels=['Not Tested', 'Tested'],
                    yticklabels=['Not Tested', 'Tested'])
        plt.title(F'Confusion Matrix - {model_name}')
        plt.ylabel('True Label')
        plt.xlabel('Predicted Label')

        plt.subplot(132)
        df_results = pd.DataFrame({
            'True': y_true,
            'Predicted': y_pred
        })
        correct = (y_true == y_pred)

        classification_results = pd.DataFrame({
            'Category': ['Correct Not Tested', 'Correct Tested',
                         'Misclassified Not Tested', 'Misclassified Tested'],
            'Count': [
                sum((correct) & (y_true == 0)),
                sum((correct) & (y_true == 1)),
                sum((~correct) & (y_true == 0)),
                sum((~correct) & (y_true == 1))
            ]
        })

```

object > FeatureEngineering&EDA.py

```

1298     class MultiClassifierOptimizer: 1 usage
1299
1300     def visualize_classification_results(self, y_true, y_pred, model_name): 1 usage
1301
1302
1303         sns.barplot(data=classification_results, x='Category', y='Count')
1304         plt.xticks(rotation=45)
1305         plt.title('Classification Distribution')
1306
1307         plt.subplot(133)
1308         error_types = {
1309             'False Positives': cm[0, 1],
1310             'False Negatives': cm[1, 0],
1311             'True Positives': cm[1, 1],
1312             'True Negatives': cm[0, 0]
1313         }
1314
1315         colors = ['red', 'orange', 'green', 'blue']
1316         plt.pie(error_types.values(), labels=error_types.keys(),
1317                 autopct='%1.1f%%', colors=colors)
1318         plt.title('Classification Analysis')
1319
1320         plt.tight_layout()
1321         plt.show()
1322
1323
1324         print(f"\nDetailed Classification Analysis for {model_name}:")
1325         print(f"Total Samples: {len(y_true)}")
1326         print(f"Correctly Classified: {sum(correct)} ({sum(correct) / len(y_true) * 100:.2f}%)")
1327         print(f"Misclassified: {sum(~correct)} ({sum(~correct) / len(y_true) * 100:.2f}%)")
1328         print("\nPer-Class Analysis:")
1329
1330         for class_name, true_val in zip(['Not Tested', 'Tested'], [0, 1]):
1331             class_total = sum(y_true == true_val)
1332             class_correct = sum((y_true == true_val) & correct)
1333             print(f"\n{class_name}:")
1334             print(f"Total: {class_total}")
1335             print(f"Correctly Classified: {class_correct} ({class_correct / class_total * 100:.2f}%)")
1336             print(
1337                 f"Misclassified: {class_total - class_correct} ({(class_total - class_correct) / class_total * 100:.2f}%)")
1338
1339
1340         def evaluate_classifier(self, model, model_name, X_train=None, y_train=None): 9 usages
1341             X_train = X_train if X_train is not None else self.X_train
1342             y_train = y_train if y_train is not None else self.y_train
1343
1344
1345             # Stratified K-fold Cross Validation
1346             skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
1347             cv_scores = cross_val_score(model, X_train, y_train, cv=skf)
1348             print(f"\n{model_name} Cross-Validation Scores:")
1349             table_data = []
1350             for i, score in enumerate(cv_scores, start=1):
1351                 table_data.append([f"CV {i}", score])
1352
1353
1354             # Display the table
1355             print("Stratified CV Scores:")
1356             print(tabulate(table_data, headers=["Fold", "Score"], tablefmt="grid"))
1357             print(f"Mean CV Score: {cv_scores.mean():.3f} (+/- {cv_scores.std() * 2:.3f})")
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
2999

```

```

class MultiClassifierOptimizer: 1 usage
    def evaluate_classifier(self, model, model_name, X_train=None, y_train=None):  9 usages
        # Get predictions
        y_pred = model.predict(self.X_test)

        # Visualize classification results
        self.visualize_classification_results(self.y_test, y_pred, model_name)

        # Calculate metrics
        cm = confusion_matrix(self.y_test, y_pred)
        precision = precision_score(self.y_test, y_pred)
        recall = recall_score(self.y_test, y_pred)
        specificity = cm[0, 0] / (cm[0, 0] + cm[0, 1])
        f1 = f1_score(self.y_test, y_pred)

        print(f"\nClassification Metrics for {model_name}:")
        print(f"Precision: {precision:.3f}")
        print(f"Recall (Sensitivity): {recall:.3f}")
        print(f"Specificity: {specificity:.3f}")
        print(f"F1-Score: {f1:.3f}")

        # ROC Curve and AUC
        try:
            y_prob = model.predict_proba(self.X_test)[:, 1]
            fpr, tpr, _ = roc_curve(self.y_test, y_prob)
            auc_score = auc(fpr, tpr)

            plt.figure(figsize=(8, 6))
            plt.plot(fpr, tpr, label=f'ROC curve (AUC = {auc_score:.2f})')
            plt.plot([0, 1], [0, 1], 'k--')
            plt.xlim([0.0, 1.0])
            plt.ylim([0.0, 1.05])
            plt.xlabel('False Positive Rate')
            plt.ylabel('True Positive Rate')
            plt.title(f'ROC Curve - {model_name}')
            plt.legend(loc="lower right")
            plt.show()

            roc_data = {'fpr': fpr, 'tpr': tpr}
        except:
            print("ROC/AUC not available for this model")
            auc_score = None
            roc_data = None

        # Per-class metrics visualization
        plt.figure(figsize=(10, 6))
        class_names = ['Not Tested', 'Tested']
        metrics_data = {
            'Class': class_names,
            'Precision': [precision_score(self.y_test, y_pred, pos_label=i) for i in [0, 1]],
            'Recall': [recall_score(self.y_test, y_pred, pos_label=i) for i in [0, 1]],
            'F1-score': [f1_score(self.y_test, y_pred, pos_label=i) for i in [0, 1]]
        }
        metrics_df = pd.DataFrame(metrics_data)

```

ct > FeatureEngineering&EDA.py

```

class MultiClassifierOptimizer: 1 usage
def evaluate_classifier(self, model, model_name, X_train=None, y_train=None):  9 usages
    metrics_melted = pd.melt(metrics_df, id_vars=['Class'], var_name='Metric', value_name='Score')
    sns.barplot(data=metrics_melted, x='Class', y='Score', hue='Metric')
    plt.title(f'Classification Metrics by Class - {model_name}')
    plt.show()

    return {
        'confusion_matrix': cm,
        'precision': precision,
        'recall': recall,
        'specificity': specificity,
        'f1_score': f1,
        'cv_scores': cv_scores,
        'auc_score': auc_score,
        'roc_curve': roc_data  # Added ROC curve data
    }

def optimize_decision_tree(self):  1 usage
# Pre-pruning phase
print("\n*** Pre-pruning Phase ***")
...
param_grid = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
    'max_depth': [3, 5, 7, 10],
    'min_samples_split': [2, 5, 10],
    'max_features': ['sqrt', 'log2', None],
    'min_samples_leaf': [1, 2, 4],
    'min_impurity_decrease': [0.0, 0.01]
}
...
# best parameters obtained :-
# {'criterion': 'gini', 'max_depth': 10, 'max_features': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'best'}
# since I have already obtained best parameters have from running the param above provided those values in grid to save time, for cross checking
# line and use grid with hyperparameters in the above param grid which is commented out above
param_grid = {
    'criterion': ['gini'],
    'splitter': ['best'],
    'max_depth': [10],
    'min_samples_split': [2],
    'max_features': [None],
    'min_samples_leaf': [2],
    'min_impurity_decrease': [0.0],
}
#best parameters obtained :-
#{'criterion': 'gini', 'max_depth': 10, 'max_features': None, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 2, 'min_samples_split': 2, 'splitter': 'best'}

dt = DecisionTreeClassifier(random_state=42)
grid_search = GridSearchCV(dt, param_grid, cv=3, n_jobs=-1, verbose=2)
grid_search.fit(self.X_train, self.y_train)

# Evaluate pre-pruned model
pre_pruned_dt = DecisionTreeClassifier(random_state=42, **grid_search.best_params_)
pre_pruned_dt.fit(self.X_train, self.y_train)
print("\nPre-pruning Best Parameters:")

```

Clustering

```

1850     def __init__(self, data):
1851         """Initialize with powerlifting dataframe"""
1852         self.data = data
1853         self.prepare_features()
1854
1855     def prepare_features(self): 1 usage
1856         """Prepare strength ratio features"""
1857         self.data['SquatStren'] = self.data['Best3SquatKg'] / self.data['BodyweightKg']
1858         self.data['BenchStren'] = self.data['Best3BenchKg'] / self.data['BodyweightKg']
1859         self.data['DeadtStren'] = self.data['Best3DeadliftKg'] / self.data['BodyweightKg']
1860
1861     def filter_data(self, sex='M', equipment='Raw', sample_size=None): 3 usages
1862         """Filter and sample data"""
1863         filtered = self.data[self.data['Sex'] == sex]
1864         if equipment:
1865             filtered = filtered[filtered['Equipment'] == equipment]
1866         if sample_size:
1867             filtered = filtered.sample(n=min(sample_size, len(filtered)), random_state=42)
1868         return filtered
1869
1870     #have reduced sample size to run fast
1871     def kmeans_analysis(self, sex='M', equipment=None, n_samples=10000, verbose=False): 1 usage
1872         """Perform K-means clustering analysis"""
1873         filtered_data = self.filter_data(sex, equipment, n_samples)
1874         features = ['SquatStren', 'BenchStren', 'DeadtStren']
1875         X = filtered_data[features]
1876
1877         # Scale features
1878         scaler = StandardScaler()
1879         X_scaled = scaler.fit_transform(X)
1880
1881         # Calculate metrics for different k
1882         max_clusters = 10
1883         range_n_clusters = range(1, max_clusters + 1)
1884         wss_scores = []
1885         silhouette_scores = []
1886
1887         for n_clusters in range_n_clusters:
1888             kmeans = KMeans(n_clusters=n_clusters, random_state=42)
1889             kmeans.fit(X_scaled)
1890             wss_scores.append(kmeans.inertia_)
1891             if n_clusters > 1:
1892                 silhouette_scores.append(silhouette_score(X_scaled, kmeans.labels_))
1893
1894         # Plot results
1895         self._plot_clustering_metrics(range_n_clusters, wss_scores, silhouette_scores)
1896
1897         # Fit final model with k=8 (as per your analysis)
1898         final_kmeans = KMeans(n_clusters=8, random_state=42)
1899         clusters = final_kmeans.fit_predict(X_scaled)
1900
1901         if verbose:
1902             print('Cluster stats:', final_kmeans.cluster_centers_)

```

```

class ClusterEngineer:
    """A class for performing clustering and association rule mining analysis on fitness equipment data.

    Methods:
        dbSCAN_analysis(self, sex='M', equipment='Raw', eps=0.3, min_samples=10, verbose=False): Perform DBSCAN clustering analysis.
        apriori_analysis(self, sex='M', equipment='Raw', n_samples=10000, min_support=0.03, min_confidence=0.5): Perform Apriori analysis.
        _create_transactions(self, data): Create transactions for Apriori analysis.
    """

    def dbSCAN_analysis(self, sex='M', equipment='Raw', eps=0.3, min_samples=10, verbose=False):
        """Perform DBSCAN clustering analysis"""
        filtered_data = self.filter_data(sex, equipment)
        features = ['SquatStren', 'BenchStren', 'DeadtStren']
        X = filtered_data[features]

        scaler = StandardScaler()
        X_scaled = scaler.fit_transform(X)

        dbSCAN = DBSCAN(eps=eps, min_samples=min_samples, n_jobs=-1)
        clusters = dbSCAN.fit_predict(X_scaled)

        self._plot_dbSCAN_results(X_scaled, clusters)

        if verbose:
            self._print_cluster_stats(filtered_data, clusters, features)

        return dbSCAN, clusters

    def apriori_analysis(self, sex='M', equipment='Raw', n_samples=10000, min_support=0.03, min_confidence=0.5):
        """Perform Apriori analysis"""
        filtered_data = self.filter_data(sex, equipment, n_samples)
        transactions = self._create_transactions(filtered_data)

        te = TransactionEncoder()
        te_ary = te.fit(transactions).transform(transactions)
        df = pd.DataFrame(te_ary, columns=te.columns_)

        frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True)
        rules = association_rules(
            frequent_itemsets,
            metric="confidence",
            min_threshold=min_confidence,
            num_itemsets=len(frequent_itemsets)
        )

        self._print_rules_summary(rules)
        return rules

    def _create_transactions(self, data):
        """Create transactions for Apriori analysis"""
        transactions = []
        for _, row in data.iterrows():
            transaction = set()

            # Squat Categories
            if row['SquatStren'] <= 1.5:
                transaction.add('Squat_Novice')
            elif row['SquatStren'] <= 2.0:
                transaction.add('Squat_Intermediate')
            else:
                transaction.add('Squat_Advanced')

            # Bench Categories
            if row['BenchStren'] <= 1.5:
                transaction.add('Bench_Novice')
            elif row['BenchStren'] <= 2.0:
                transaction.add('Bench_Intermediate')
            else:
                transaction.add('Bench_Advanced')

            # Deadlift Categories
            if row['DeadtStren'] <= 1.5:
                transaction.add('Deadt_Novice')
            elif row['DeadtStren'] <= 2.0:
                transaction.add('Deadt_Intermediate')
            else:
                transaction.add('Deadt_Advanced')

            transactions.append(transaction)

        return transactions

```

Object > FeatureEngineering&EDA.py

References

Powerlifting dataset: <https://www.kaggle.com/datasets/open-powerlifting/powerlifting-database/data>

NumPy: <https://numpy.org/doc/>

Pandas: <https://pandas.pydata.org/docs/>

Matplotlib: <https://matplotlib.org/stable/index.html>

Seaborn: <https://seaborn.pydata.org/>

Scikit-learn: <https://scikit-learn.org/stable/>