



# FOOTBALL INJURY PREDICTION

TEAM 13

MALAK EL-HAMSHARY (ID:91241072)

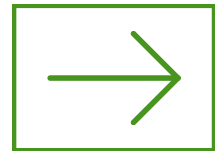
ROWIDA MOHAMMED (ID: 91240303)

MONA MOHAMMED ELKHOLY (ID: 91241075)

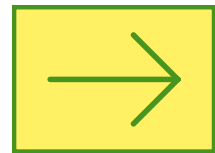
KHADIJA ZAKARIA MABROUK (ID: 91240965)

COURSE CODE: SBEG209

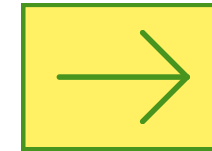
# TABLE OF CONTENT



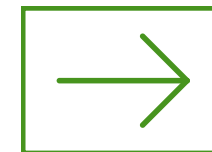
INTRODUCTION



METHODOLOGY



RESULTS & DISCUSSION



CONCLUSIONS

# Introduction



- Football is a high-intensity sport with a significant risk of injury, especially among university athletes balancing training and academics. These injuries affect both performance and long-term health.
- This study uses a **dataset of 800 Chinese university football players**, including physical, fitness, workload, lifestyle, and training features, to predict medically verified injuries in the following season.
- A custom Naïve Bayes classifier is implemented for binary injury prediction and its performance is compared with standard Python models.

# METHODOLOGY



Data  
Pre-processing



Visualization  
and Hypothesis  
Testing



Naive Bayes  
Implementation  
from Scratch



Naive Bayes  
Using  
[scikit-learn](#)

# DATA PRE-PROCESSING

## Dataset Overview

- **Source:** Kaggle – University Football Injury Prediction Dataset
- **Samples:** 800 players
- **Features:** 18 input features + 1 target label
- **Balance:** Well-balanced dataset
- **Task:** Binary classification
  - 0 = No Injury
  - 1 = Injury

## Feature Types

- **Quantitative Features:** Age, Height, Weight, Training Hours, Matches Played, Injury History, Knee Strength, Hamstring Flexibility, Reaction Time, Balance, Sprint Speed, Agility, Sleep Hours, Stress Level, Nutrition Quality, BMI
- **Categorical Features:**
  1. Position
  2. Warm-up Routine Adherence

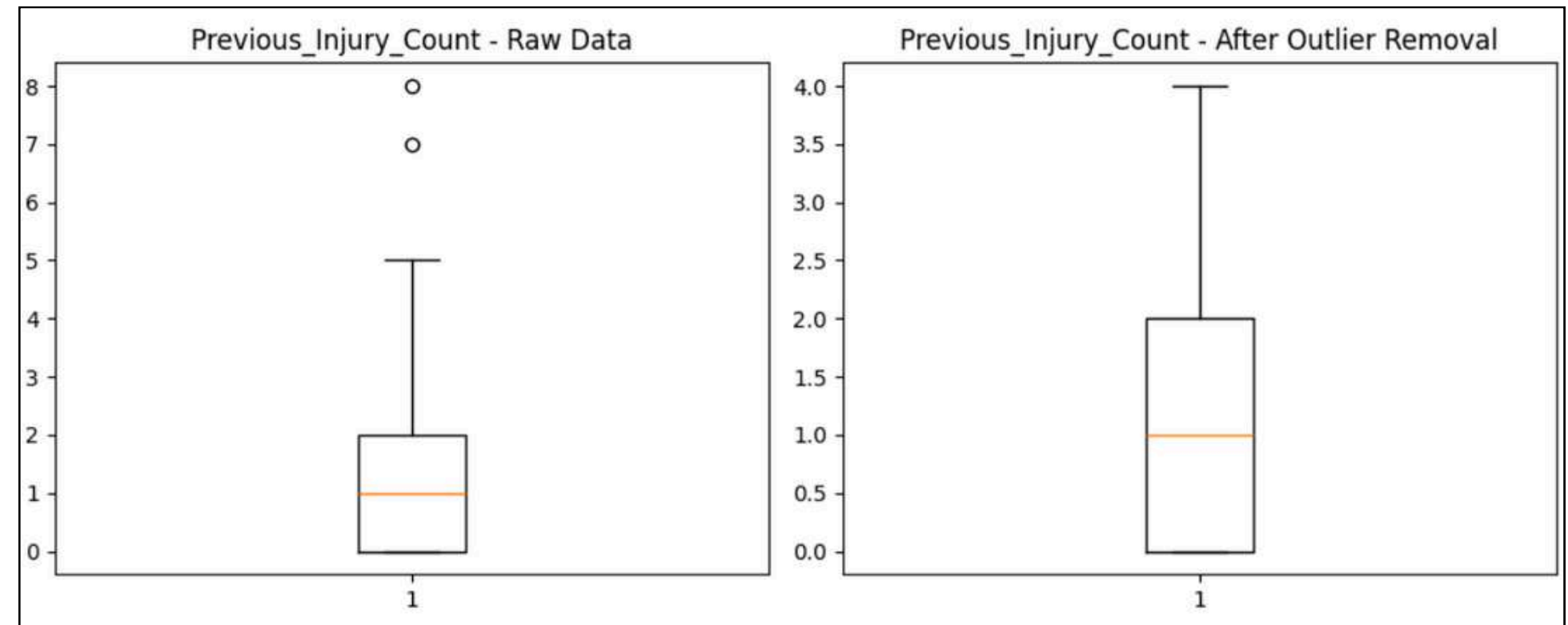
## Packages Used

- **scikit-learn:** pre-implemented Gaussian Naive Bayes model and evaluation metrics
- **NumPy:** numerical operations
- **pandas:** data handling and preprocessing
- **Matplotlib and Seaborn:** For data visualization
- **SciPy:** For statistical tests (Shapiro-Wilk)

# DATA PRE-PROCESSING

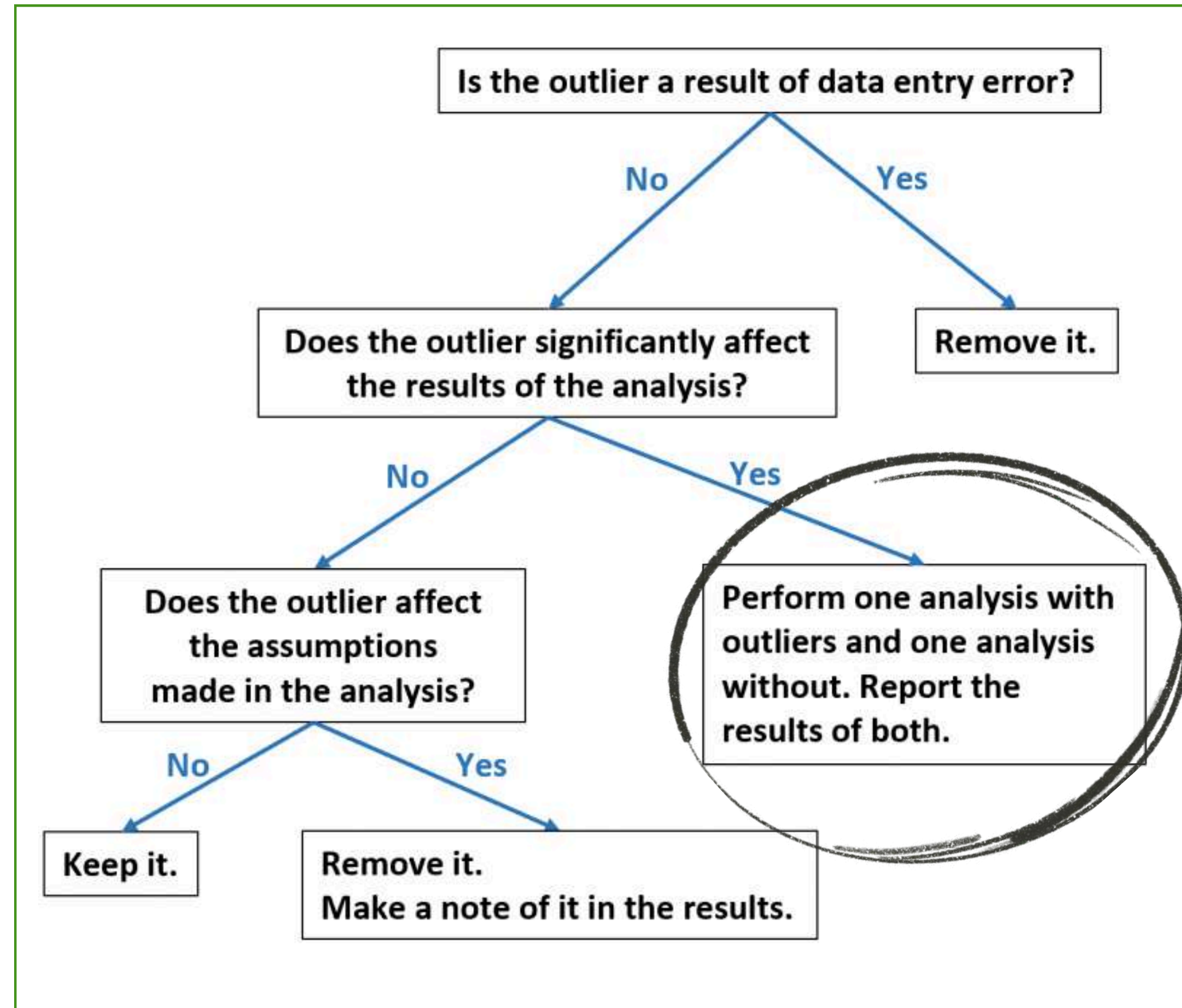
## Outliers Detection and Removal

- Outliers detected using Interquartile Range (IQR) method
- Threshold:  **$Q1 - 2 \times IQR$  and  $Q3 + 2 \times IQR$**
- Rows with at least one outlier removed
- 24 rows removed (776 sample)
- Two datasets created:
  - Raw dataset
  - Cleaned dataset



- Note: The “most used” threshold:  $Q1 - 1.5 \times IQR$ ,  $Q3 + 1.5 \times IQR$  was tested at first, but 115 rows were removed (approx. 14% from the full dataset), so a 2 multiplier was used instead.

# OUR APPROACH



# DATA PRE-PROCESSING

## Descriptive Statistics

- Computed on both datasets (**raw and cleaned**)
- **Quantitative:**  
Mean, Median, Variance, Std, Min, Max
- **Categorical:**  
Mode Calculated before standardization to preserve interpretability

## Train-Test Split

- Both datasets split independently
- **80%** Training, **20%** Testing
- Stratified to preserve injury class distribution

## Feature Standardization

$$z = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}}$$

- **Z-score** normalization applied
- Mean and standard deviation computed from training data only
- Same parameters applied to test set
- Prevents data leakage

# DATA PRE-PROCESSING

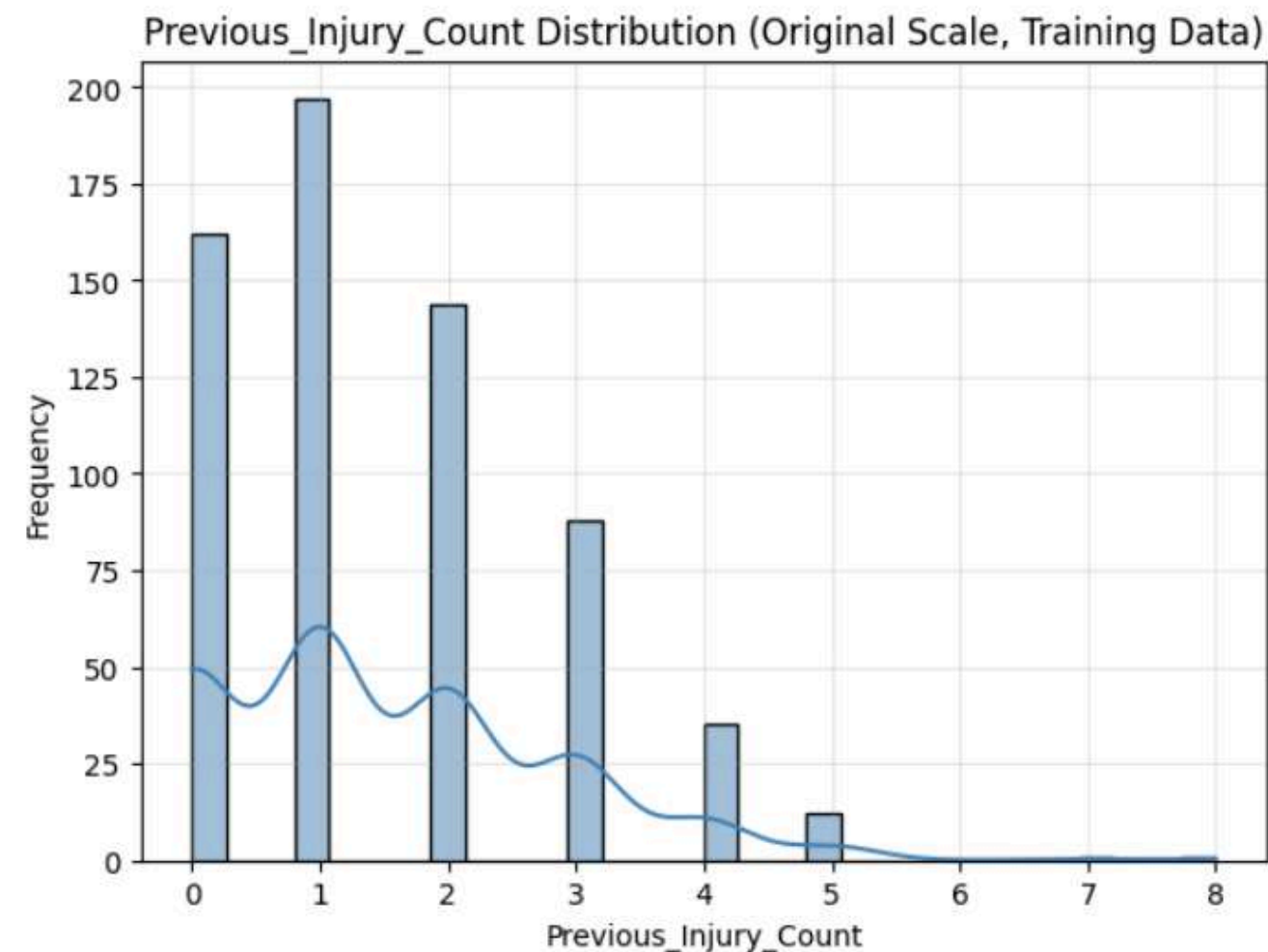
- The **mean** is highly sensitive to outliers; therefore, it was computed before and after outlier removal.
- Most features show negligible change after removal: Age, Height, Weight: error < 0.05%
- Previous Injury Count is still affected.

Feature	Mean of Raw Data	Mean after Outliers Removal	Error (%)
Age	21.135	21.142	0.033
Height_cm	177.407	177.479	0.04
Weight_Kg	73.235	73.255	0.027
Previous_Injury_Count	1.536	1.436	6.5

Outlier removal has minimal impact for most of features

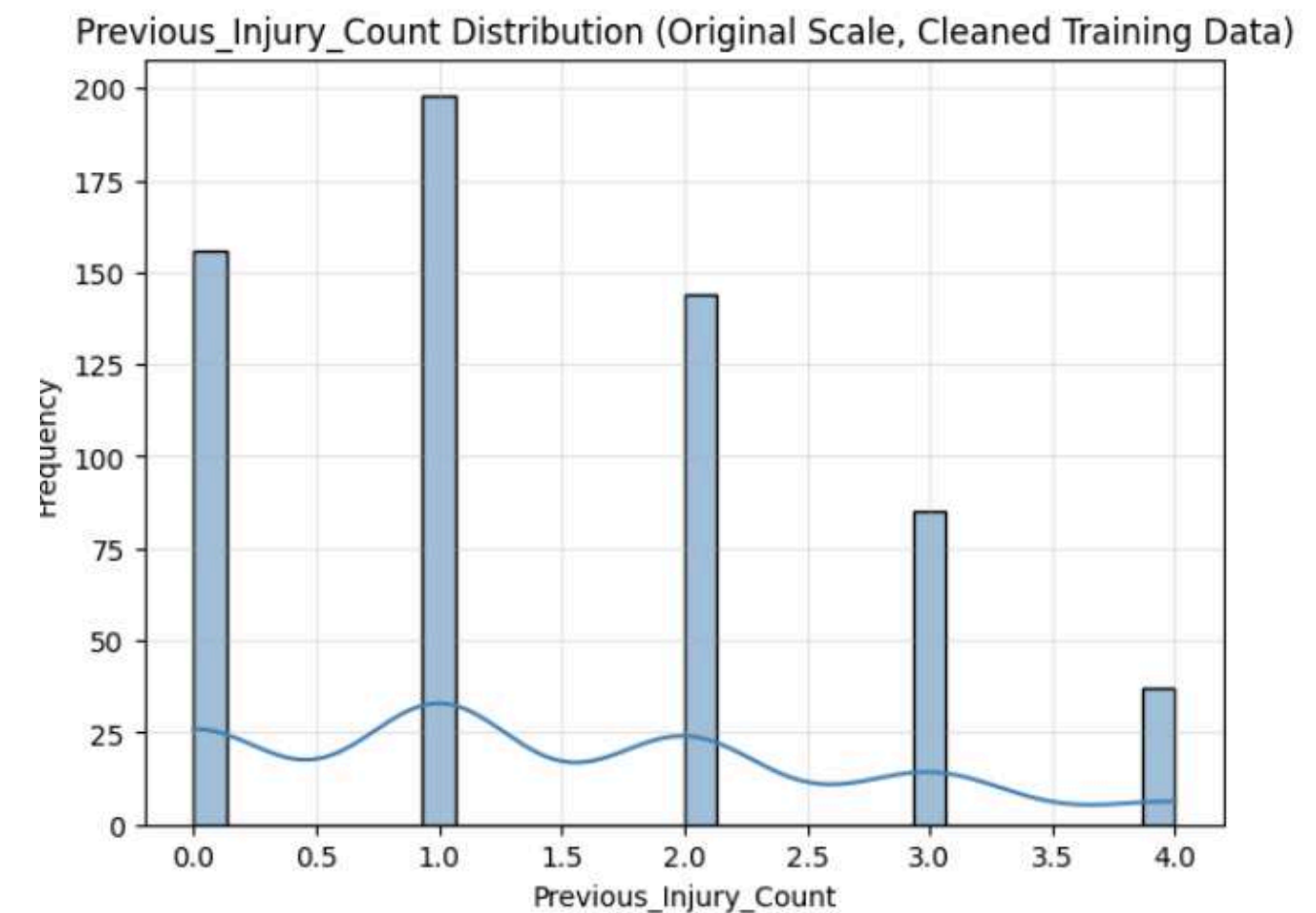
# DISTRIBUTION ANALYSIS

Histograms were used to examine the distribution of quantitative features on their original scale and to compare their shapes before and after outlier removal.



## **Previous Injury Count (with outliers):**

The distribution is heavily right-skewed, with most players having very few previous injuries and a small number of extreme values.

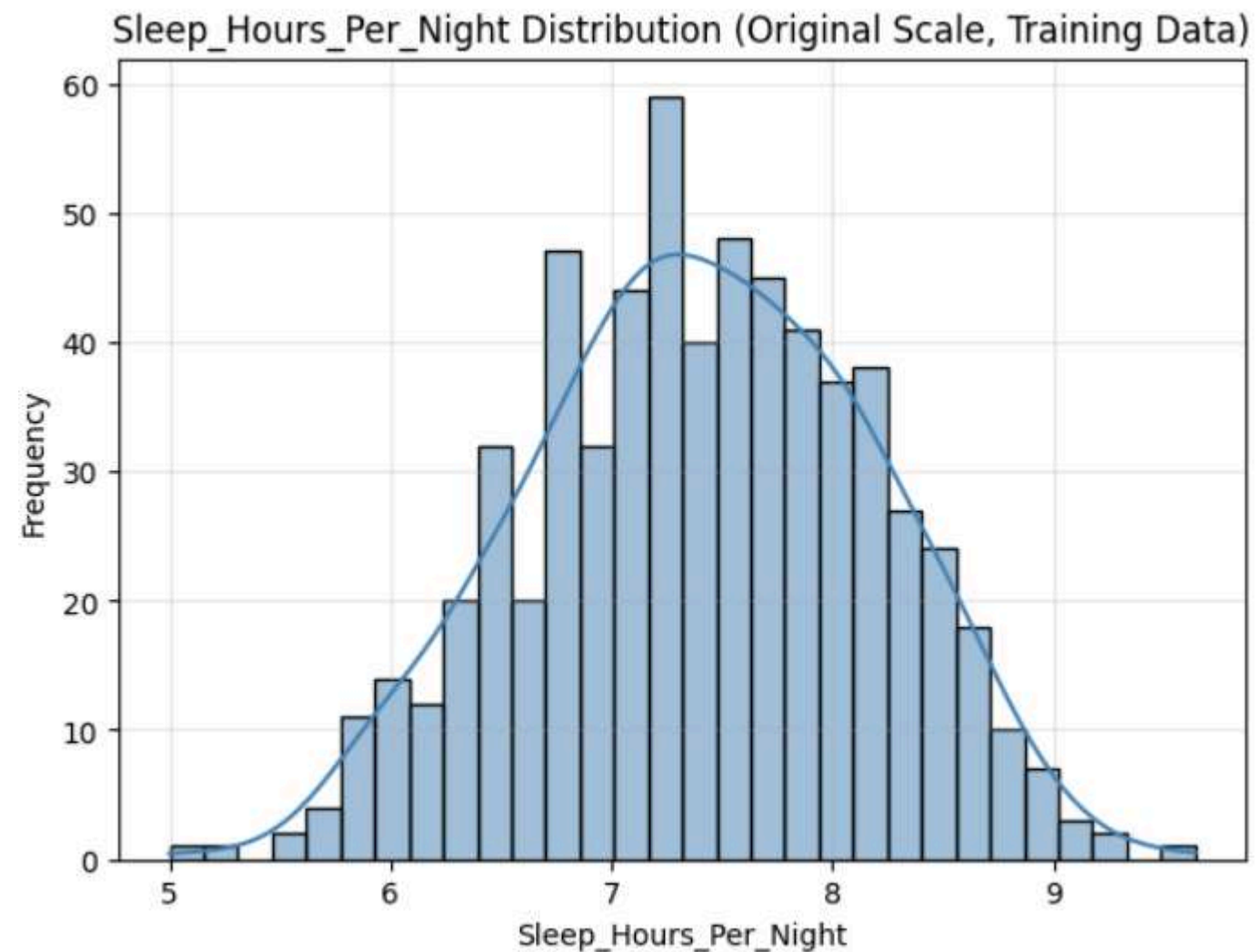


## **Previous Injury Count (outliers removed):**

After outlier removal, the distribution remains right-skewed but is less extreme due to the elimination of unusually high injury counts.

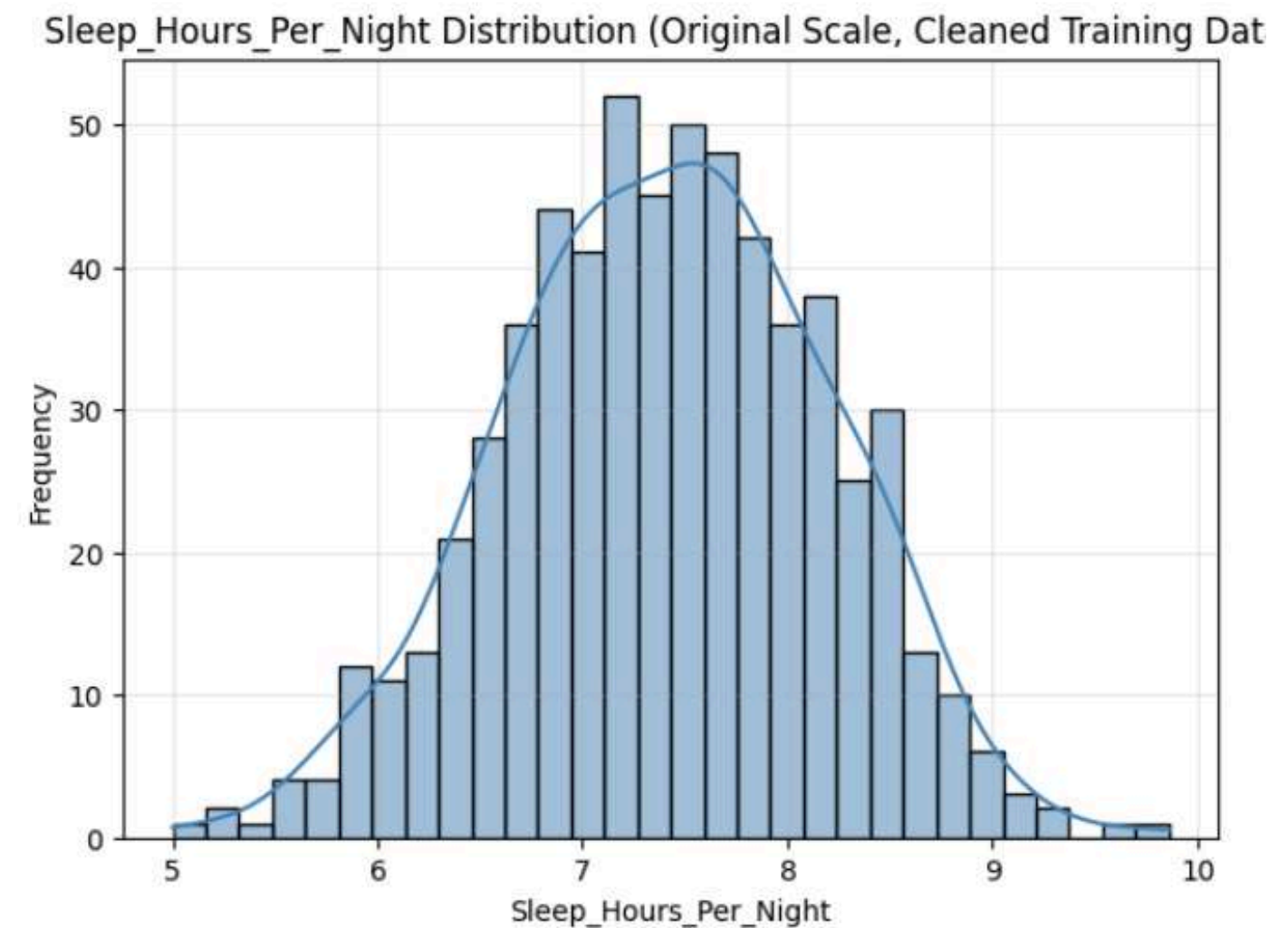
# DISTRIBUTION ANALYSIS

Histograms were used to examine the distribution of quantitative features on their original scale and to compare their shapes before and after outlier removal.



## **Sleep Hours per Night (with outliers):**

The distribution shows a slight left skew, indicating a small number of players reporting unusually long sleep durations.

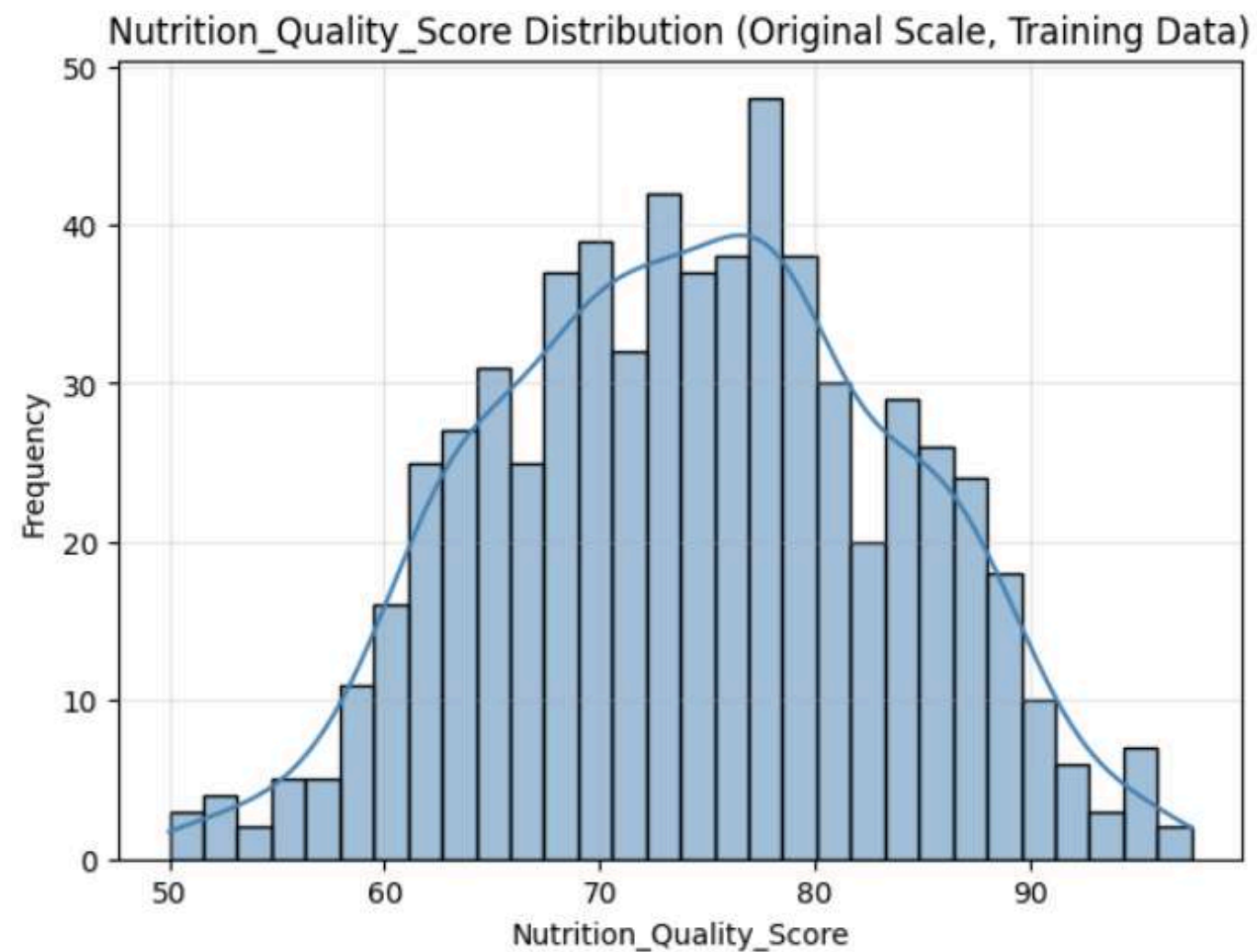


## **Sleep Hours per Night (outliers removed):**

Removing outliers results in a more symmetric, Gaussian-like distribution centered around typical sleep hours.

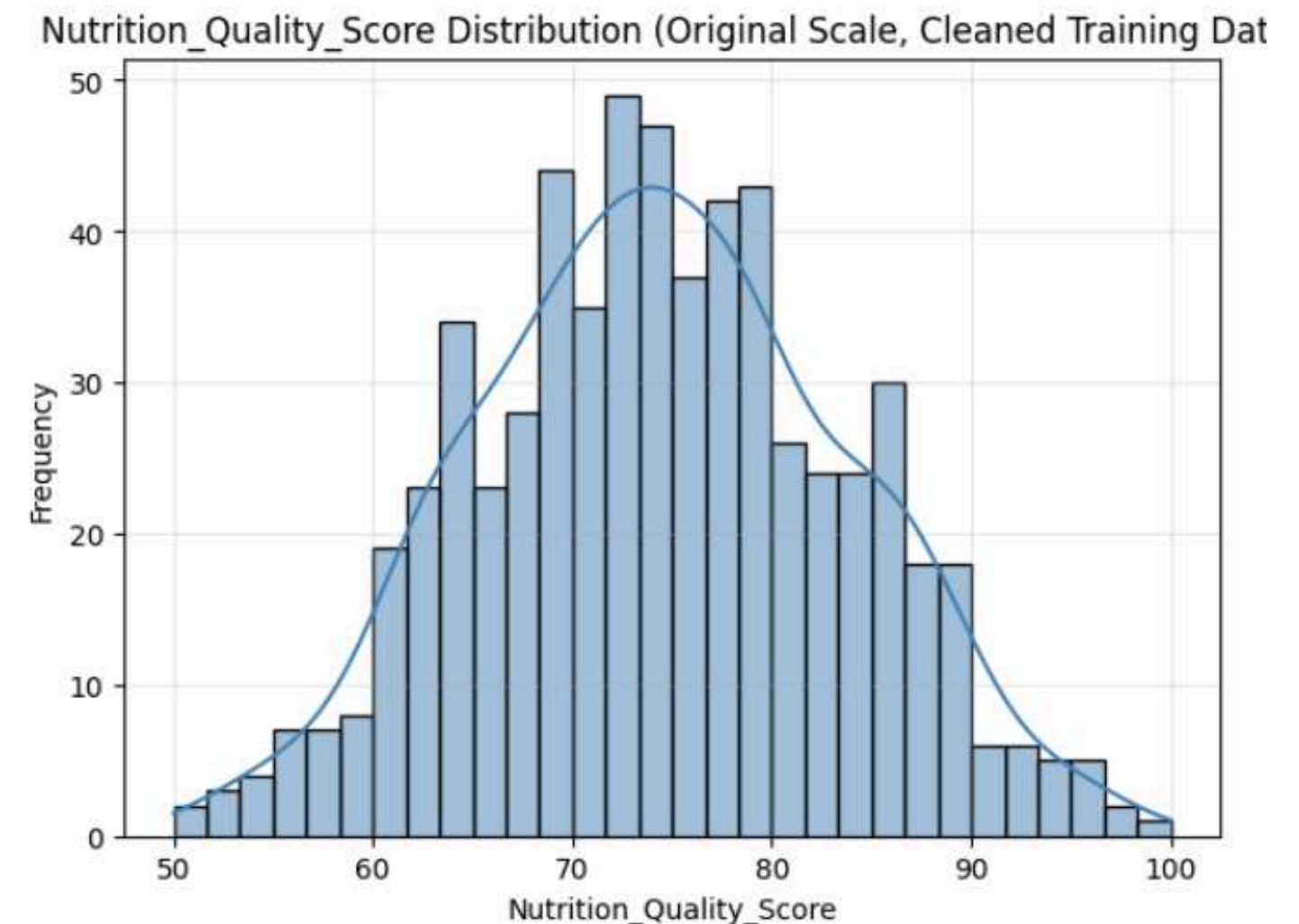
# DISTRIBUTION ANALYSIS

Histograms were used to examine the distribution of quantitative features on their original scale and to compare their shapes before and after outlier removal.



## **Nutrition Quality Score (Before outlier removal):**

The distribution is slightly left-skewed and shows a small secondary peak, indicating variability and deviation from a clear Gaussian shape.



## **Nutrition Quality Score (After outlier removal):**

The distribution becomes more centered and unimodal, with a shape that more closely resembles a Gaussian distribution.

# DISTRIBUTION ANALYSIS

## Key Observations from Distribution Analysis

- Some features already approximately Gaussian (e.g., Height, Knee Strength)
- Others skewed or uniform (e.g., Matches Played)
- Outliers exaggerated non-Gaussian behavior
- Cleaning improved normality for several features

# NORMALITY TESTING

The Shapiro–Wilk test was used to formally evaluate whether each quantitative feature follows a normal distribution by measuring how closely the data matches a Gaussian shape; it was chosen for its reliability on small to moderate sample sizes.

## Hypotheses:

- $H_0$ : Feature is normally distributed
- $H_1$ : Feature is not normally distributed

## Test details:

- Significance level:  $\alpha=0.05$
- Applied to training data only

$$W = \frac{\left(\sum_{i=1}^n a_i x_{(i)}\right)^2}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

**W** measures how closely the sample values follow a normal distribution, with values near 1 indicating normality.

# NORMALITY TESTING

The Shapiro–Wilk results indicate that a few features significantly deviate from normality, while most features approximately satisfy the Gaussian assumption after outlier removal.

The p-value is the probability of observing a test statistic as extreme as W under the assumption that the feature is normally distributed. It quantifies how likely the observed data would occur if the null hypothesis were true.

Feature	W Statistic	p-value
Age	0.9162	<b>0.0000</b>
Height (cm)	0.9964	0.1673
Weight (kg)	0.9971	0.3412
Training Hours / Week	0.9891	<b>0.0002</b>
Matches Played (Past Season)	0.9446	<b>0.000</b>
Previous Injury Count	0.8876	<b>0.0000</b>
Knee Strength Score	0.9977	0.5403
Hamstring Flexibility	0.999	0.9822
Reaction Time (ms)	0.9967	0.241
Balance Test Score	0.9951	<b>0.045</b>
Sprint Speed (10 m, s)	0.9974	0.4441
Agility Score	0.9964	0.1826
Sleep Hours / Night	0.9981	0.7178
Stress Level Score	0.9978	0.5882
Nutrition Quality Score	0.9967	0.2443
BMI	0.9955	0.0689

# CONDITIONAL DISTRIBUTIONS

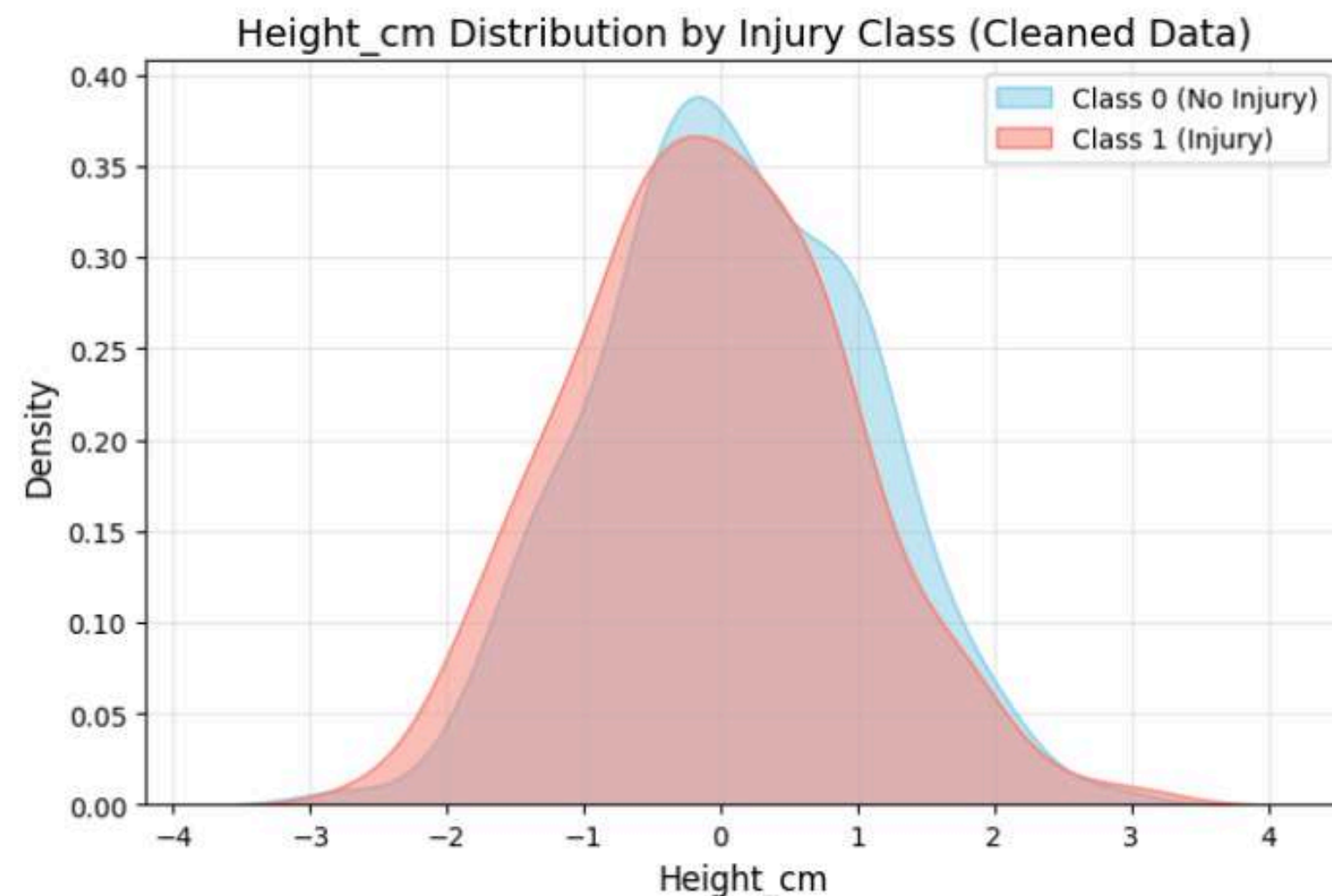
To analyze how individual features relate to injury occurrence, we plotted class-conditional Kernel Density Estimation (KDE) distributions for each quantitative feature.

These plots estimate the probability density of feature values separately for injured and non-injured players, allowing direct comparison between the two classes. By using standardized feature values, differences in scale were removed, making the separation between classes easier to interpret.

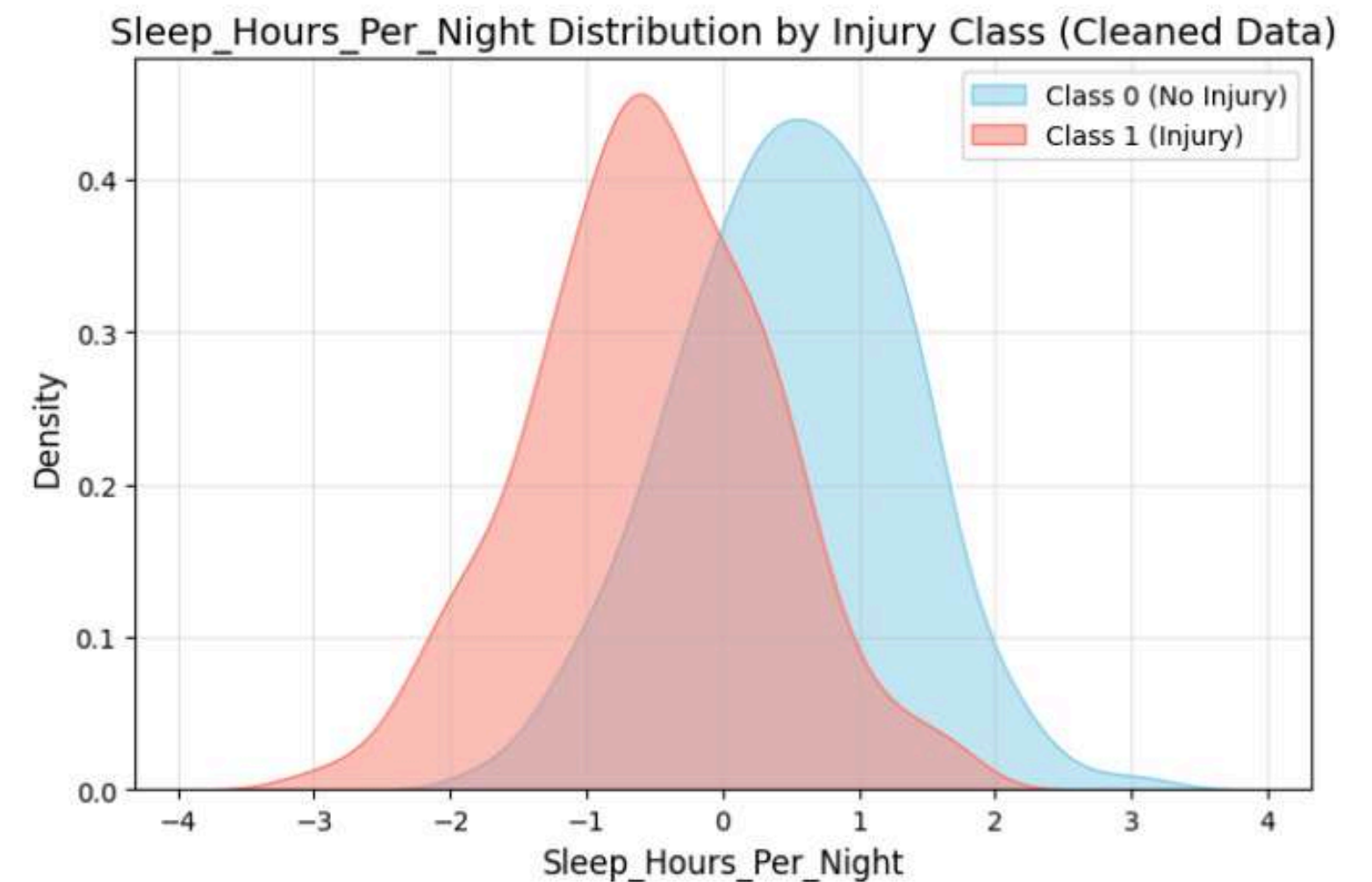
This analysis helps identify features that contribute meaningfully to injury prediction versus those with limited discriminative power.

# CONDITIONAL DISTRIBUTIONS

The following conditional KDE plots show the distribution of feature values separately for injured (Class 1) and non-injured (Class 0) players, illustrating how each feature relates to injury risk.



KDE curves overlap for injured and non-injured players, showing height has little predictive value.



Curves differ between classes, indicating fewer sleep hours are linked to higher injury risk and making this feature more informative.

# GAUSSIAN NAIVE BAYES IMPLEMENTATION FROM SCRATCH

## What is Naive Bayes ?

Naive Bayes is a probabilistic classifier based on Bayes' Theorem. Because our dataset contains continuous features, we used the Gaussian variant, which assumes that features follow a normal distribution within each class

## Why Naive Bayes?

- Probabilistic classifier
- Based on Bayes' Theorem
- Assumes feature independence
- Gaussian version for continuous data

# BAYES' THEOREM

$$P(c|x) \propto P(c) \prod_{i=1}^n P(x_i|c)$$

## Class Prior - $P(c)$

- Represents the probability that a randomly selected sample belongs to class  $c$ .
- Estimated from the training data as:

$$P(c) = \frac{N_c}{N}$$

where  $N_c$  is the number of samples in class  $c$ , and  $N$  is the total number of samples.

## Likelihood - $P(x_i | c)$

- Measures how likely a feature value  $x_i$  is given class  $c$ .
- In Gaussian Naive Bayes, this probability is modeled using the Gaussian Probability Density Function.

## Decision Rule

- Compute the posterior probability for each class.
- Select the class with the maximum posterior probability.

# NUMERICAL STABILITY

$$\log P(c \mid x) = \log P(c) + \sum \log P(x_i \mid c)$$

- Log probabilities used
- Prevents numerical underflow
- Product  $\rightarrow$  Sum

## Gaussian Log PDF

$$\log P(x|c) = -\frac{1}{2}\log(2\pi\sigma^2) - \frac{(x - \mu)^2}{2\sigma^2}$$

- Used for likelihood computation
- Summed across features

## Prediction Phase

$$\underset{c}{\operatorname{argmax}} \log P(c|x)$$

- Compute posterior for each class
- Choose class with max probability

# MODEL STRUCTURE

The Gaussian Naive Bayes classifier consists of **three** core components:

- `fit()` – Model training
- `gaussian_log_pdf()` – Likelihood computation
- `predict()` – Class inference

## Training Phase:

### 1. Class Identification

- Determine the unique class labels in the dataset.

### 2. Feature Statistics Estimation

- Estimate the mean  $\mu$  and variance  $\sigma^2$  of each feature for every class.

### 3. Prior Probability Calculation

- Compute class priors based on class frequencies.

### 4. Zero-Variance Handling

- To prevent numerical instability, variances equal to zero are replaced with a small constant:

$$\sigma^2 \leftarrow 10^{-9}$$

A variance of zero leads to division-by-zero errors in the Gaussian likelihood function.

# SKLEARN NAIVE BAYES MODEL

Implemented using GaussianNB from **scikit-learn** to **benchmark the custom (from-scratch) model**.

Trained and tested on the same raw and cleaned datasets under identical experimental conditions.

## Evaluation:

- Predictions via predict()

## Performance assessed using:

- Accuracy
- Precision, Recall, F1-score
- Confusion Matrix

## Purpose:

- Validate the correctness of the custom implementation
- Provide a standardized baseline for comparison
- Ensure reproducibility and alignment with established ML practices

```
model_clean_skl = GaussianNB()
model_clean_skl.fit(X_train_clean, y_train_clean)

y_pred2_sklearn = model_clean_skl.predict(X_test_clean)

print("Accuracy:", accuracy_score(y_test_clean, y_pred2_sklearn))
print("\nClassification Report:\n", classification_report(y_test_clean, y_pred2_sklearn))
print("\nConfusion Matrix:\n", confusion_matrix(y_test_clean, y_pred2_sklearn))
```

# RESULTS & ANALYSIS

## Performance on Full Dataset (800 Samples)

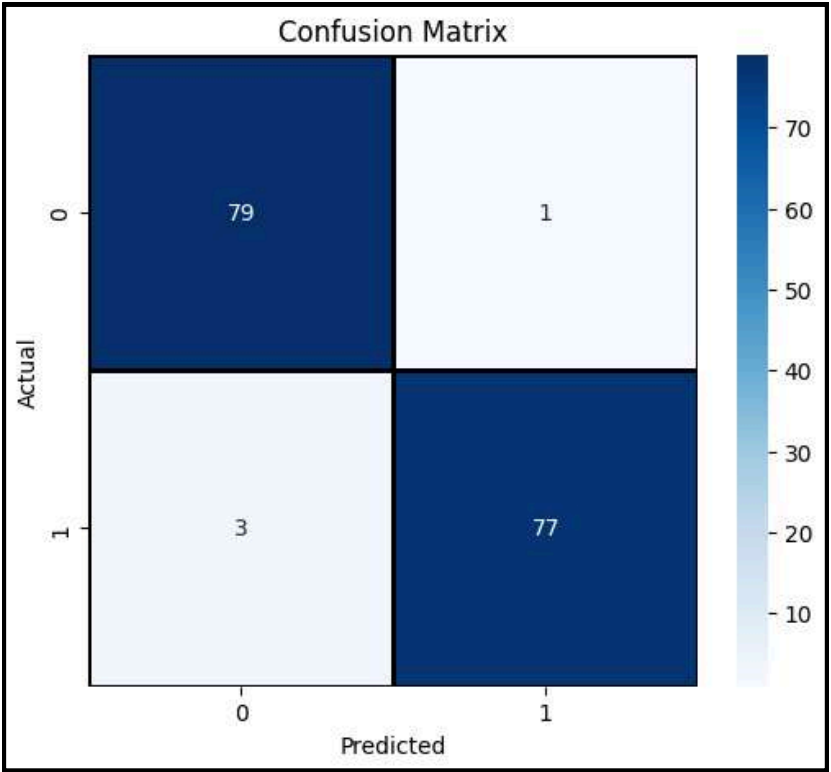
- Compared from-scratch Gaussian NB and scikit-learn GaussianNB
- Identical training/testing settings

### Evaluation performed on:

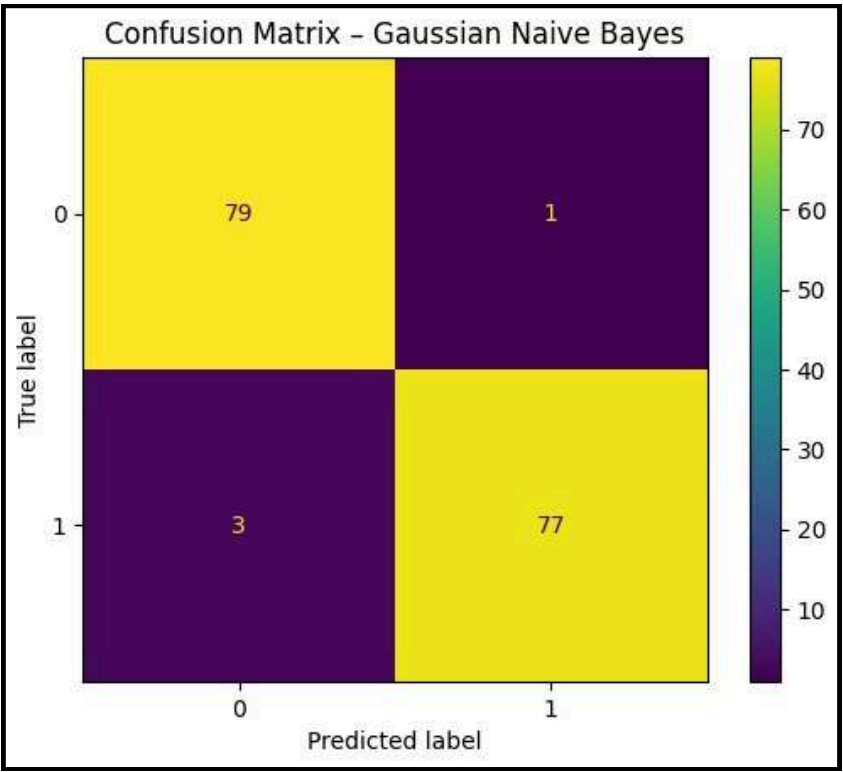
- Full dataset (800 samples)
- Cleaned dataset after outlier removal (24 samples removed)

Metric	From-Scratch NB	SKLearn NB
Accuracy	0.975	0.975
Precision (Class 1)	0.987	0.987
Recall (Class 1)	0.9625	0.9625
F1 Score (Class 1)	0.9747	0.9747

Identical results confirm the correctness and reliability of the custom implementation.



Sklearn Model



Self-implemented model

Both models show very few misclassifications

Only:

- 3 False Negatives
- 1 False Positive

Confusion matrices are identical, indicating consistent behavior across implementations.

# RESULTS & ANALYSIS

## Performance After Outlier Removal (24 Samples)

- Accuracy dropped from 0.975 → 0.9487 after outlier removal
  - Most removed samples belonged to the positive injury class
  - This reduced the prior probability of class 1
- Since Naive Bayes relies heavily on class priors:
- Model becomes biased toward the negative class
  - Leads to a drop in recall for the positive class (0.9625 → 0.921)

Metric	From-Scratch NB	SKLearn NB
Accuracy	0.9487	0.9487
Precision (Class 1)	0.972	0.972
Recall (Class 1)	0.921	0.921
F1 Score (Class 1)	0.946	0.946
Confusion Matrix [TP / FP / FN / TN]	70/2/6/78	70/2/6/78

# CONCLUSION

- Injury prediction for university football players was studied using statistical analysis and Gaussian Naive Bayes (GNB).
- Outlier removal (IQR method) eliminated 24 extreme samples, improving feature distributions and Gaussian normality, confirmed by Shapiro–Wilk tests.

Feature analysis showed:

- Knee Strength Score strongly associated with lower injury risk.
- Some features (e.g., Weight) had limited predictive value.
- Both from-scratch and scikit-learn GNB models achieved high accuracy (0.975), with a drop after outlier removal due to reduced positive-class samples.
- Precision, recall, and F1-score remained robust.
- Strong agreement between implementations validates the modeling approach.

**Takeaway:**

**Gaussian Naive Bayes is effective for injury prediction, and identifying strongly discriminative features can support injury prevention strategies and streamlined data collection.**

# MEMBER CONTRIBUTIONS

## MALAK

- Implemented the sklearn GaussianNB model.
- Computed performance metrics, compared results with the manual implementation.
- Finalized all figures, tables, and outputs for the report and presentation.



## MONA

- Managed the dataset by acquiring, cleaning, and preparing it for modeling.
- Computed descriptive statistics, handled missing values and outliers, and executed an 80%/20% train-test split.



## ROWIDA

- Implemented Gaussian Naive Bayes from scratch
- Gaussian PDF implementation
- Model inference & accuracy evaluation



## KHADIJA

Validated statistical assumptions for Naive Bayes by analyzing feature distributions, conducting normality tests, defining hypotheses, interpreting p-values, and plotting conditional distributions to assess feature relevance.



# REFERENCES

- [1] Yuanchunhong, University Football Injury Prediction Dataset, Kaggle, 2023. Available at:  
<https://www.kaggle.com/datasets/yuanchunhong/university-football-injury-prediction-dataset/data>
- [2] Tukey, J. W. (1977). Exploratory Data Analysis. Addison-Wesley.
- [3] Bobbitt, Z. (2021, November 15). The Complete Guide: When to Remove Outliers in Data.  
Statology. Available at: <https://www.statology.org/remove-outliers/>
- [4] S. S. Shapiro and M. B. Wilk, “An analysis of variance test for normality (complete samples),”  
Biometrika, vol. 52, no. 3–4, pp. 591–611, 1965.

**THANK YOU**

**ANY QUESTIONS?**

