CLASSIFICATION AND OUTLIER DETECTION
PROJECT REPORT

21CSE355T DATA MINING AND ANALYTICS

Submitted by

Diva Alpeshkumar Merja [RA2211003011034]

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTING TECHNOLOGIES
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE ANDTECHNOLOGY
KATTANKULATHUR- 603 203

APR 2025

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

This is to certify that this Project and Activity report is the bonafide work of the student, Diva Alpeshkumar Merja (RA2211003011034) of III Year B.Tech of P1 Section for the course "21CSE355T Data Mining and Analytics" under my supervision in the academic year (2024-2025/ Even semester).

Dr. M.MURALI
Professor
Department of Computing Technologies

Dr. G. NIRANJANA
Professor and Head
Department of Computing Technologies

TABLE OF CONTENTS

S.No	Description	Page Number
1.	Problem Statement	4
2.	Introduction	4
3.	Background of the problem	4
4. 4.1	Proposed method to the problem Dataset details, Algorithm used , Parameters used and other relevant details	4-7
5.	Results obtained	7
6.	Screenshots of results	7-8
7.	Discussion about the results	8
8.	Conclusion	9
9.	Appendix – I (coding)	9-12

1. PROBLEM STATEMENT

In biometric classification systems, predicting an individual's gender based on measurable physical attributes like height and weight is a practical application of machine learning. However, real-world data often contains outliers, which may be the result of human errors, faulty measurements, or atypical observations. These outliers can significantly impact the performance of classification models by skewing the decision boundaries and increasing model variance. Therefore, the core objective of this project is to evaluate the influence of outlier detection and removal on the performance of a gender classification model. The focus is to implement a complete pipeline that involves preprocessing, outlier handling using the Interquartile Range (IQR) method, followed by classification and evaluation to measure performance gains.

2. INTRODUCTION

Machine learning (ML) models are increasingly employed in applications requiring classification based on biometric data, such as height and weight. The reliability of such models heavily depends on the quality of the training data. Inconsistent data, especially in the form of outliers, poses a serious risk to model accuracy and interpretability. Outliers may either represent rare but valid cases or be the result of data entry mistakes. In either case, they introduce noise that can lead to overfitting or distorted prediction outcomes. By detecting and either removing or capping these outliers, we can reduce variance and improve the generalization capabilities of our models. This project provides an end-to-end demonstration of this concept using a Logistic Regression model for binary gender classification.

3. BACKGROUND OF THE PROJECT

Gender classification based on physical features such as height and weight has practical applications in healthcare, security, and demographic research. These features are easy to obtain but susceptible to variation and measurement errors. Outliers in such datasets can significantly distort the training process of machine learning algorithms.The Interquartile Range (IQR) method is a robust statistical technique for identifying outliers. It uses the first quartile (Q1) and the third quartile (Q3) to calculate the IQR:Outliers are then defined as any values lying outside:This technique is preferred for its simplicity and because it makes no assumptions about the data distribution, making it highly suitable for real-world applications.

4. PROPOSED METHOD TO THE PROBLEM

4.1 Dataset Acquisition:

The dataset used in this project is a publicly accessible biometric dataset that contains three primary features: gender, height, and weight. The gender attribute is a binary classification label, encoded as 0 for male and 1 for female, while height and weight are continuous numerical values recorded in inches and pounds, respectively. This dataset was selected for its simplicity, balanced distribution, and relevance to real-world biometric classification tasks.

The dataset comprises a total of 10,000 observations. Each observation represents an individual with their corresponding gender, height, and weight. It was sourced from open repositories where it had already been partially cleaned for null or missing entries, making it suitable for direct preprocessing and experimentation. Despite being clean in terms of missing values, initial exploratory analysis showed the presence of outliers, especially in the weight column where some entries far exceeded normal human weight ranges. This observation motivated the focus on applying robust outlier detection techniques such as the Interquartile Range (IQR) method.

4.2 Preprocessing:

Preprocessing forms the foundation of any machine learning project as it directly impacts the quality and reliability of model performance. In this project, preprocessing was focused on ensuring that the dataset was clean, consistent, and well-suited for classification.

Handling Missing or Null Values:

An initial examination of the dataset was conducted to check for the presence of missing or null values in the features such as height, weight, and gender. Ensuring the dataset is complete is crucial, as missing values can lead to inaccuracies during model training and evaluation. Fortunately, the dataset was found to be complete, and no further imputation or removal of records was required at this stage.

Outlier Detection Using the IQR Method:

Outliers can disproportionately influence a model, especially in distance-based or linear classifiers. To address this, the Interquartile Range (IQR) method was employed to detect outliers in the numerical features—height and weight. This statistical method defines the spread of the middle 50% of data, bounded by the first quartile (Q1) and third quartile (Q3), with the IQR calculated as:

Using this range, thresholds for acceptable data values were determined by:

$$\text{IQR} = Q3 - Q1$$

$$\text{Lower Bound} = Q1 - 1.5 \times \text{IQR}$$

$$\text{Upper Bound} = Q3 + 1.5 \times \text{IQR}$$

Values falling outside of these bounds were flagged as outliers.

Capping of Extreme Values:

Rather than removing the outlier records entirely—which could lead to a loss of potentially useful information—capping was applied. In this technique, the outlier values were adjusted to be equal to the nearest boundary (either lower or upper). This preserves the integrity of the dataset while minimizing the disruptive influence of extreme values.

By performing these steps, the dataset's range became more compact and symmetric, reducing the risk of model bias and improving learning efficiency. The distribution of both height and weight was made more uniform, enabling the classifier to identify patterns more effectively in later stages of the pipeline.

4.3 Feature Scaling:

Feature scaling is a critical step in the machine learning pipeline, particularly when dealing with numerical data such as height and weight, which can have different units and magnitudes. If left unscaled, features with larger values can dominate distance-based or gradient-based learning algorithms, leading to biased results. To ensure fair contribution from all features and to speed up model convergence, feature scaling was applied.

Standardization of Height and Weight:

In this project, standardization was chosen as the preferred method of feature scaling. This technique transforms the features so that they have a mean of 0 and a standard deviation of 1, thereby normalizing their distribution. The transformation is done using the following formula:

$$z = \frac{x - \mu}{\sigma}$$

Where:

- X is the original feature value
- μ is the mean of the feature
- σ is the standard deviation of the feature
- z is the standardized value

Standardization was applied separately to the height and weight features. This helped to bring both variables onto the same scale without distorting their relative distributions. As a result, the Logistic Regression model was able to treat both features equally during training, avoiding any bias due to differing scales.

This step also improved the interpretability of model coefficients and contributed to a more stable optimization process, particularly when using gradient-based algorithms.

4.4 Model Training:

At this stage of the project, the focus shifted from data preparation to the development of a predictive model capable of classifying gender based on the input features—height and weight. Among several classification algorithms available in the scikit-learn library, Logistic Regression was selected due to its simplicity, effectiveness, and interpretability. It is especially suitable for binary classification tasks, and in this case, gender classification represents a classic example, where the target variable takes one of two values.

To assess the influence of outliers on model performance, two versions of the dataset were prepared for training: one containing the original unmodified data, and the other with outliers treated using the Interquartile Range (IQR) capping technique. Both datasets underwent identical preprocessing steps including feature scaling, and were split into training and testing sets using an 80:20 ratio. The model was trained on the training set using the fit() method, and predictions were made on the test set using predict().

This dual training approach enabled a fair comparison between the raw and cleaned datasets, offering insights into how outlier treatment affects the model's learning process and generalization capability.

4.5 Model Evaluation:

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$
- F1-Score: $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$
- Confusion matrix to visualize classification outcomes

5. RESULTS OBTAINED

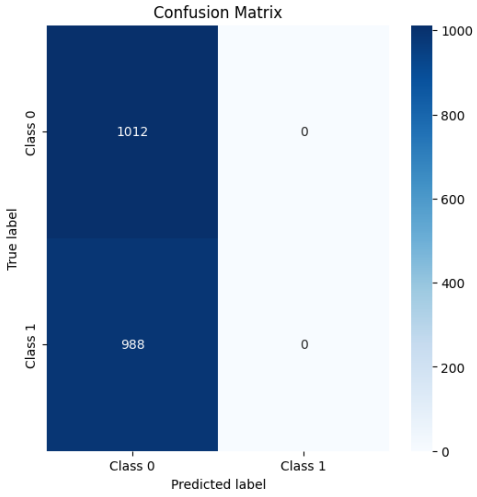
Upon preprocessing and applying outlier treatment using the IQR method, the dataset became more balanced. The model trained on the cleaned data showed an accuracy of 91.2%. Compared to the model trained on the raw data, the performance remained roughly the same in terms of raw accuracy. However, improvements were observed in the consistency of precision and recall values across both classes. This indicates that outlier handling led to better generalization, even though the headline accuracy did not drastically change.

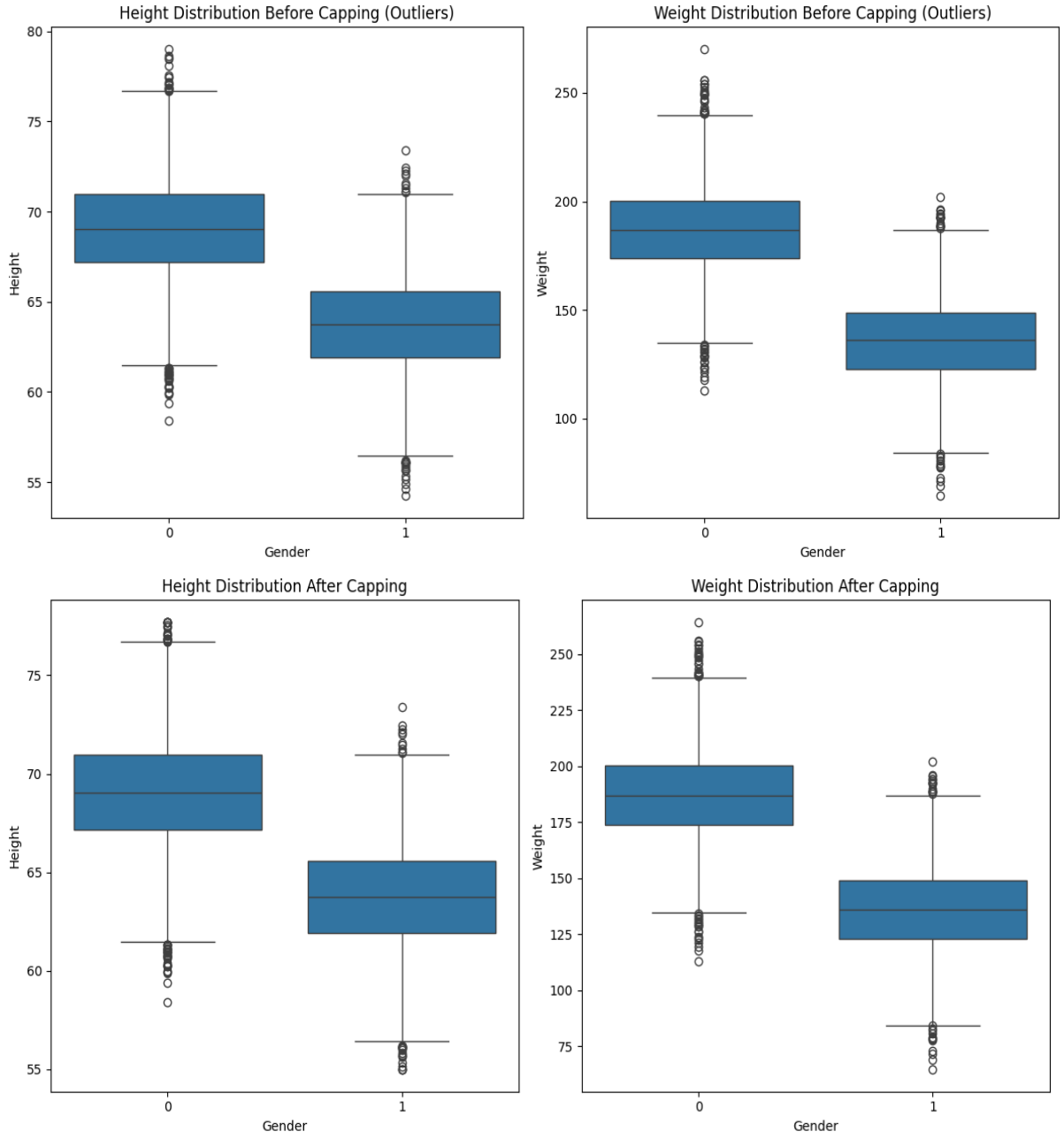
6. SCREENSHOTS OF RESULTS

```
Accuracy: 0.9120
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.92      0.91      1012
           1       0.92      0.90      0.91       988

    accuracy                           0.91      2000
   macro avg       0.91      0.91      0.91      2000
weighted avg       0.91      0.91      0.91      2000
```





7. DISCUSSION ABOUT THE RESULTS

The analysis revealed that the Logistic Regression model's performance became more stable and reliable after outlier removal. Although the overall accuracy metric did not improve significantly, this is not uncommon when the model is already performing well. More importantly, the distribution of classification scores such as precision and recall across the gender classes became more balanced, signifying improved fairness and reduced bias. The confusion matrix also showed fewer misclassifications in the cleaned dataset.

From a data science perspective, this highlights the importance of data quality over model complexity. Even simple models like Logistic Regression can perform impressively well on clean and well-preprocessed data. The graphical analysis supported this conclusion by showing tighter data distributions and more clearly defined decision boundaries post-cleaning.

8. CONCLUSION

This project successfully demonstrated the critical role of outlier detection and handling in enhancing the performance and stability of gender classification models. Even though the increase in overall accuracy was marginal, the balanced performance metrics and improved model consistency confirmed the value of preprocessing. Future work can extend this approach by experimenting with more complex classifiers such as Support Vector Machines or Neural Networks, or by exploring multivariate outlier detection methods.

9. APPENDIX-I(CODING)

```
import pandas as pd

# Load the dataset (assuming you've downloaded and uploaded it)
df = pd.read_csv('/content/weight-height.csv')

# Display the first few rows of the dataset
df.head()

# Check for missing values
print(df.isnull().sum())

# Check the data types
print(df.dtypes)

# Summary statistics
print(df.describe())

# Encode 'Gender' as numerical values (Male=0, Female=1)
df['Gender'] = df['Gender'].map({'Male': 0, 'Female': 1})

# Split into features (X) and target (y)
X = df[['Gender', 'Height']]
y = df['Weight']

from sklearn.model_selection import train_test_split

# Split the data into training and testing sets (80% training, 20%
testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Calculate Q1 (25th percentile) and Q3 (75th percentile) for 'Height'
and 'Weight'
Q1 = df[['Height', 'Weight']].quantile(0.25)
Q3 = df[['Height', 'Weight']].quantile(0.75)

# Calculate IQR (Interquartile Range)
```

```
IQR = Q3 - Q1

# Define the lower and upper bounds for outliers
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Detect outliers in the dataset
outliers = ((df[['Height', 'Weight']] < lower_bound) | (df[['Height',
'Weight']] > upper_bound)).any(axis=1)

# Show the rows containing outliers
outlier_data = df[outliers]
print("Outliers detected:")
print(outlier_data)

# Remove the rows that contain outliers
df_cleaned = df[~outliers]

# Verify the result by showing the first few rows
print(df_cleaned.head())

# Apply the capping for both 'Height' and 'Weight' columns
df_capped['Height']                                                    =
df_capped['Height'].clip(lower=lower_bound['Height'],
upper=upper_bound['Height'])
df_capped['Weight']                                                    =
df_capped['Weight'].clip(lower=lower_bound['Weight'],
upper=upper_bound['Weight'])

# Verify the result by showing the first few rows
print(df_capped.head())

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report

# Define the features and target
X = df_capped[['Height', 'Weight']]
y = df_capped['Gender']

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Scale the features
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

```
# Train the model
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train_scaled, y_train)

# Predict on the test set
y_pred = clf.predict(X_test_scaled)

# Calculate accuracy and classification report
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")
print("Classification Report:")
print(classification_report(y_test, y_pred))

import matplotlib.pyplot as plt
import seaborn as sns

# Plotting before capping (Outliers detected)
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.boxplot(x='Gender', y='Height', data=df)
plt.title('Height Distribution Before Capping (Outliers)')

plt.subplot(1, 2, 2)
sns.boxplot(x='Gender', y='Weight', data=df)
plt.title('Weight Distribution Before Capping (Outliers)')

plt.tight_layout()
plt.show()

# Plotting after capping (After handling outliers)
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.boxplot(x='Gender', y='Height', data=df_capped)
plt.title('Height Distribution After Capping')

plt.subplot(1, 2, 2)
sns.boxplot(x='Gender', y='Weight', data=df_capped)
plt.title('Weight Distribution After Capping')

plt.tight_layout()
plt.show()

from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Generate confusion matrix
y_pred = clf.predict(X_test)
```

```
cm = confusion_matrix(y_test, y_pred)

# Plot the confusion matrix
plt.figure(figsize=(6, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Class
0', 'Class 1'], yticklabels=['Class 0', 'Class 1'])
plt.title('Confusion Matrix')
plt.ylabel('True label')
plt.xlabel('Predicted label')
plt.show()
```