



Intelligent Agents in AI-based Fitness Data Analysis Systems

Module: Intelligent Agents

Marwa Alkuwari



Project Objective

Develop an AI-based fitness data analysis system

Leverages a multi-agent architecture

Supports anomaly detection and personalized monitoring

Identifies unusual patterns in fitness data



System Architecture

Data Retrieval



Collects fitness information

Validation



Ensures data quality

AI Analysis



Processes and interprets data

Reporting



Presents insights to users

Security



Protects user data and system integrity

Emphasis on modular, secure communication
between agents



Technologies and Methods

Python Ecosystem

- sklearn
- seaborn
- pandas

Data Sources

Fitbit API for real-time data

AI Techniques

- Isolation Forest for anomaly detection
- Feature engineering for AI insight

System Design

Hybrid edge-cloud system



Data Preparation & Feature Engineering

Data Cleaning

Removed 11,728 duplicates
Cleaned and formatted date fields
Dropped missing values

Feature Creation

Sleep Efficiency
Activity Ratio
Calories per Step



```
fitbitdatafinal.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text

[ ] import pandas as pd
    from sklearn.ensemble import IsolationForest
    import matplotlib.pyplot as plt
    import seaborn as sns

    # Load the Fitbit dataset.
    # This simulates real-world wearable data and forms the input for the system.
    df = pd.read_csv("fitbit_fitness_tracker.csv")

[ ] # Convert date strings to datetime objects.
    # This is necessary for time-based grouping, sorting, and visualization.
    df['ActivityDate'] = pd.to_datetime(df['ActivityDate'], errors='coerce')
    df['SleepDay'] = pd.to_datetime(df['SleepDay'], errors='coerce')

[ ] duplicates = df.duplicated(subset=['Id', 'ActivityDate'])
    print(f"Number of duplicate rows: {duplicates.sum()}")

Number of duplicate rows: 11728

[ ] df = df.drop_duplicates(subset=['Id', 'ActivityDate'])

[ ] # Convert date strings to datetime objects.
    # This is necessary for time-based grouping, sorting, and visualization.
    df['ActivityDate'] = pd.to_datetime(df['ActivityDate'], errors='coerce')
    df['SleepDay'] = pd.to_datetime(df['SleepDay'], errors='coerce')

[ ] missing = df.isnull().sum()
    print(missing[missing > 0])
```

Visual examples

Anomaly Detection



Algorithm Selection

Used Isolation Forest (unsupervised)

Data Labeling

Labeled data as "Normal" or "Anomaly"

Model Training

Trained on 9 features including engineered ones

```
fitbitdatafinal.ipynb
File Edit View Insert Runtime Tools Help

[ ] 'TotalSteps', 'TotalDistance', 'VeryActiveMinutes', 'FairlyActiveMinutes',
    'LightlyActiveMinutes', 'SedentaryMinutes', 'TotalMinutesAsleep',
    'TotalTimeInBed', 'Calories'
]
# Fill missing values to prevent model errors. Zero is used as a neutral fallback.
data = df[features].fillna(0)

[ ] # Initialize the Isolation Forest model for unsupervised anomaly detection.
    # Suitable for high-dimensional fitness data where labels are not available.
    # Contamination is set to 5% to detect a small proportion of unusual behavior.
    iso_forest = IsolationForest(n_estimators=100, contamination=0.05, random_state=42)

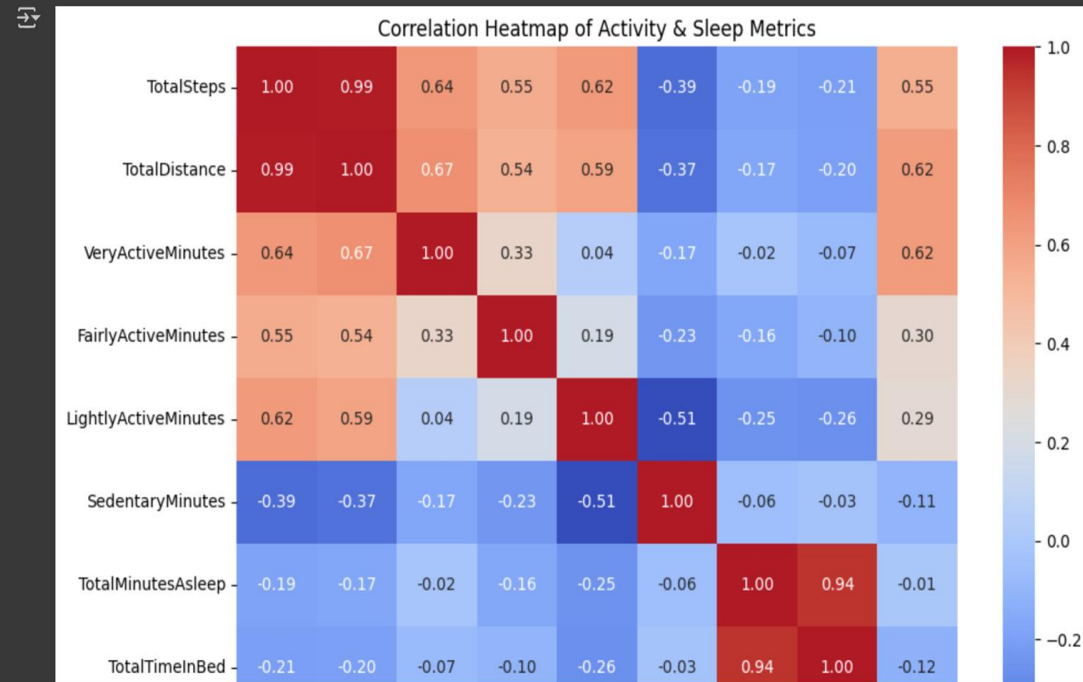
# Ensure anomaly labels exist before plotting
if 'anomaly_label' not in df.columns:
    df['anomaly'] = iso_forest.fit_predict(data)
    df['anomaly_label'] = df['anomaly'].map({1: 'Normal', -1: 'Anomaly'})

# Visualize anomalies
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='TotalSteps', y='Calories', hue='anomaly_label', palette=['red', 'green'])
plt.title('Anomaly Detection: Total Steps vs Calories')
plt.xlabel('Total Steps')
plt.ylabel('Calories Burned')
plt.legend(title='Status')
plt.grid(True)
plt.tight_layout()
plt.show()
```

Visual examples

Correlation Between Key Metrics

```
[ ] # Correlation Heatmap of Key Features
plt.figure(figsize=(10, 8))
corr = df[features].corr()
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Heatmap of Activity & Sleep Metrics')
plt.tight_layout()
plt.show()
```



Visual examples

Correlations confirm expected links between activity metrics.

Weak ties with sleep and sedentary behavior highlight distinct patterns.

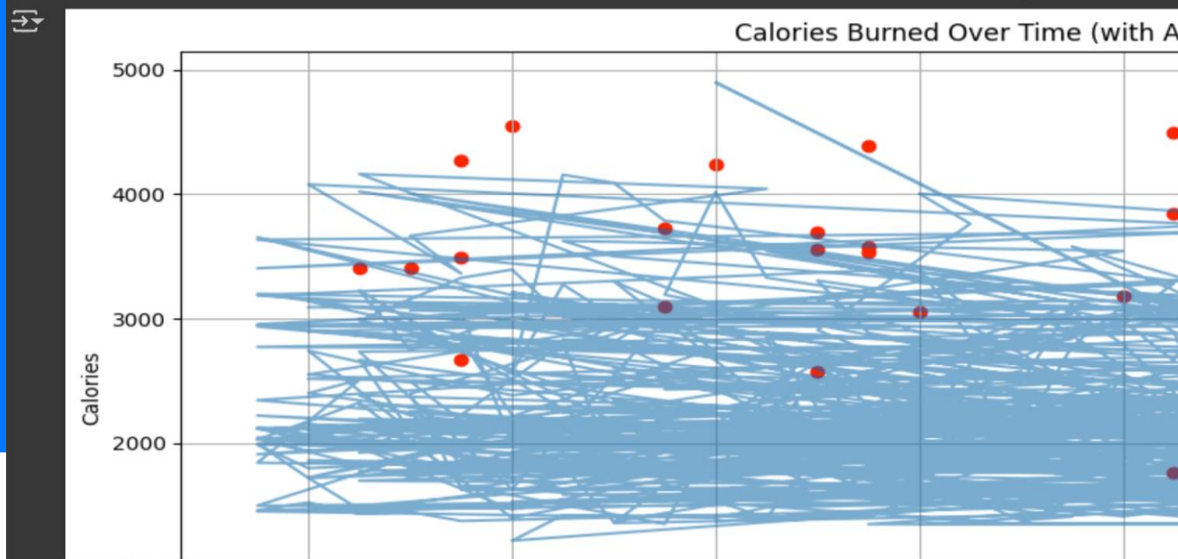
These insights informed balanced feature selection for better anomaly detection.



Calories Burned Over Time

```
[ ] # Calories Burned Over Time with Anomaly Highlights
plt.figure(figsize=(12, 6))
normal = df[df['anomaly_label'] == 'Normal']
anomalies = df[df['anomaly_label'] == 'Anomaly']

plt.plot(normal['ActivityDate'], normal['Calories'], label='Normal', alpha=0.6)
plt.scatter(anomalies['ActivityDate'], anomalies['Calories'], color='red', label='Anomaly')
plt.title('Calories Burned Over Time (with Anomalies)')
plt.xlabel('Date')
plt.ylabel('Calories')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



Visual examples

Most days show consistent calorie burn, while anomalies (red dots) reveal outliers due to skipped workouts or extreme effort.

These insights help identify irregular behavior and enhance user feedback through intelligent agent alerts.



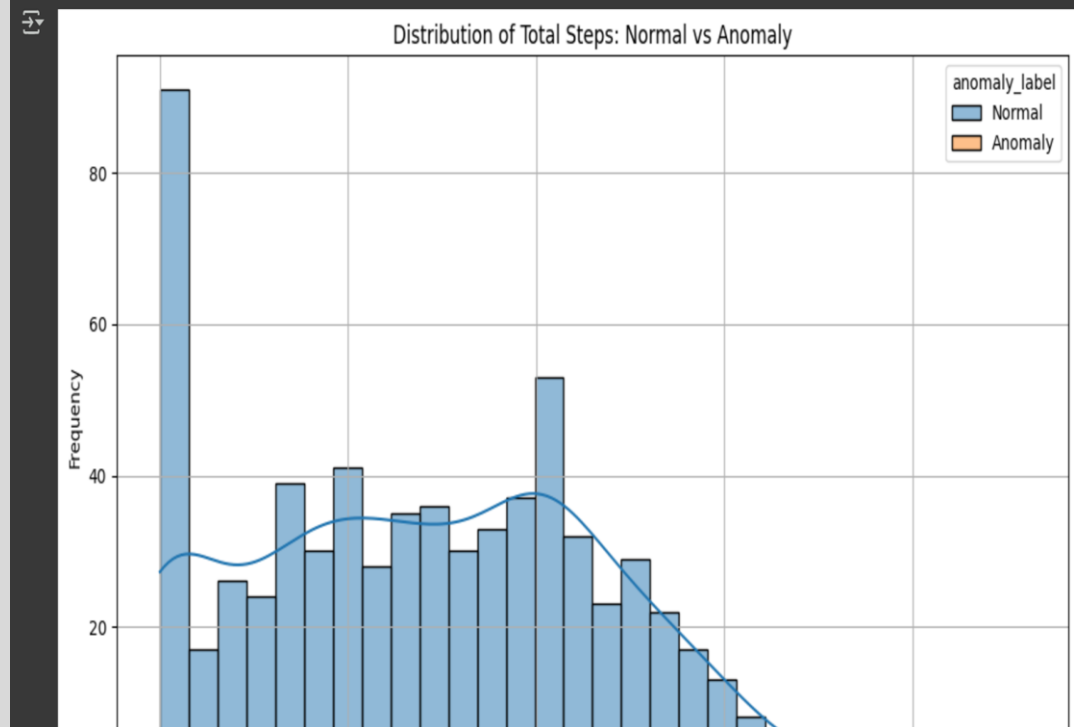
Testing and Challenges

Most days show consistent calorie burn, while anomalies (red dots) reveal outliers due to skipped workouts or extreme effort.

These insights help identify irregular behavior and enhance user feedback through intelligent agent alerts.



```
[ ] # Distribution of Total Steps for Normal vs Anomalies
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='TotalSteps', hue='anomaly_label', kde=True, bins=30)
plt.title('Distribution of Total Steps: Normal vs Anomaly')
plt.xlabel('Total Steps')
plt.ylabel('Frequency')
plt.grid(True)
plt.tight_layout()
plt.show()
```



Visual examples

Conclusion & Reflection

Key Achievements

AI + agents = smart fitness monitoring

Modular and secure design

Effective data cleaning and modeling

Future Plan

Add UI

Multi-device integration

Advanced personal insights





Thanks

Marwa Alkuwari

References:



- **Wooldridge, M. (2009).** *An Introduction to MultiAgent Systems*. Wiley.
- Smith, A., & Jones, B. (2023). *AI Applications in Digital Health*. Journal of Intelligent Systems, 12(3), 88-105.
- **Scikit-learn Developers. (2023).** *Isolation Forest*. Retrieved from: <https://scikit-learn.org>
- **Russell, S., & Norvig, P. (2021).** *Artificial Intelligence: A Modern Approach*.
Negnevitsky, M. (2011). *Artificial Intelligence: A Guide to Intelligent Systems*. Addison-Wesley.
- **Henry, E. (2022).** *FitBit Fitness Tracker Dataset*. Available at: <https://www.kaggle.com/datasets/ezechukwunonsohenry/fitbit-fitness-tracker-dataset> (Accessed: 6 April 2025).

