Step	Action	Description	Rationale	Key Outcome/Result
1	Review All Provided Datasets	Explored the three datasets: safety_monitoring.csv, daily_reminder.csv, and health_monitoring.csv. Checked structure, size, and relevant features.	Needed to identify which dataset most directly supports the project's objective of real-time ADL classification and anomaly (fall/inactivity) detection.	Found that safety_monitoring.csv contains direct activity/event labels relevant to detecting safety-critical incidents.
2	Inspect safety_monitoring.csv Features	Reviewed columns, datatypes, and sample values. Looked for sensor-based features and clearly defined labels.	Direct mapping to ADL monitoring tasks requires sensor inputs + event/label data.	Confirmed presence of relevant sensor readings and explicit "safety" status labels, making it suitable for classification and anomaly detection.
3	Inspect daily_reminder.csv	Analyzed structure and fields related to scheduled reminders and alerts.	While reminders can be part of an elderly care solution, they do not directly provide continuous behavioral or sensor data for ML-based ADL recognition.	Determined dataset is not useful for initial ML modeling but may be supplementary for future system integration.
4	Inspect health_monitoring.csv	Checked health vitals (heart rate, blood pressure, etc.) and timestamps.	Health vitals are valuable for context but less directly linked to ADL labeling unless combined with other sensor data.	Considered as a secondary dataset to potentially enrich features after core ADL model is established.
5	Compare Dataset Relevance	Compared the three datasets against criteria: (a) presence of time-series sensor data, (b) labeled activities/events, (c) potential for anomaly detection.	Needed a primary dataset that aligns tightly with project goal of elderly safety monitoring through ADL recognition.	safety_monitoring.csv scored highest on all three criteria, making it the primary dataset for initial modeling.
6	Final Dataset Selection	Chose safety_monitoring.csv for milestone work.	Contains event labels tied to safety (e.g., falls, emergencies) and enough diversity of normal activities to train a balanced model with targeted anomaly detection strategies.	Clear focus on core objective: detecting safety-critical events while tracking ADLs for elderly independence.

Stage	Step	Action / Method	Description	Rationale	Key Outcome / Result
Exploratory Data Analysis (EDA)	1	Initial Data Loading & Inspection	Loaded safety_monitoring.csv into Pandas, checked column names, types, and previewed data.	Understand dataset structure, spot immediate issues before preprocessing.	10,000 rows × 10 columns loaded; identified empty Unnamed: 9 column; placeholders like - found in numeric fields.
	2	Handle Numeric Missing Values	Replaced - with NaN, imputed gaps in numeric features using median; imputed all-missing Impact Force Level with 0.	Prevent bias from missing values; median preserves distribution; zeros for all-missing field avoid data loss.	Complete numeric dataset without large gaps; consistent for modeling.
	3	Handle Categorical Missing Values	Dropped rows missing critical categorical values (Movement Activity, Fall Detected, Alert Triggered).	Missing labels undermine supervised learning and accuracy metrics.	Removed incomplete rows, ensuring reliable labels for training/testing.
	4	Outlier Detection	Used IQR method and box plots to flag outliers in Impact Force Level and Post-Fall Inactivity Duration.	Identify potential anomalies in sensor data; guide later filtering or weighting.	Outliers confirmed visually; kept for now as they may be genuine fall events.
Data Preparation	5	Convert Data Types	Converted timestamps to datetime; ensured numeric columns are floats; dropped unused Unnamed: 9 column.	Correct types allow time-based feature extraction and prevent computation errors.	Enabled rolling windows and temporal analysis.
	6	Remove Unrealistic Sensor Values	Filtered values outside plausible physiological/motion ranges.	Avoid skewing model with sensor glitches or faulty readings.	Cleaned dataset with reduced noise.
	7	Normalize & Standardize	Used StandardScaler on numeric features for zero mean, unit variance.	Many ML models (e.g., SVM) are sensitive to feature scale.	All features on comparable scale; improves convergence.
	8	Address Class Imbalance	Assessed label distribution; planned SMOTE oversampling for minority classes.	Falls are rare but critical; balanced training improves recall.	Documented imbalance; SMOTE-ready dataset for modeling.
	9	Train-Test Split	Split data preserving class distribution; applied balancing only to training set.	Prevents data leakage; ensures test set reflects real-world distribution.	Fair evaluation setup.
Feature Engineering	10	Time-Based Features	From timestamps, created day_of_week, hour_of_day, minute_of_day.	Temporal patterns may reveal ADL routines or risk periods.	Dataset enriched with temporal context.
	11	Inactivity Feature	Created time_since_last_event to measure inactivity duration.	Long inactivity may indicate a fall or health issue.	Added a key safety-related feature.
	12	Statistical Features	Computed mean, std, min, max for accelerometer/gyroscope signals in fixed windows.	Summarizes movement magnitude and variability for classification.	More informative input features for ADL recognition.
	13	Categorical Encoding	One-hot encoded categorical variables (Movement Activity, Location, day_of_week).	Converts categorical features into ML-compatible format.	Expanded dataset with encoded columns ready for modeling.

Stage	Step	Action / Method	Description	Rationale	Key Outcome / Result
Data Loading & Cleaning	1	Load Raw Data	Loaded three CSVs: safety, health, and reminder monitoring datasets.	To gather the full dataset needed for activity and anomaly tasks.	Dataframes loaded; safety and health cleaned for invalid values.
Data Cleaning & Preprocessing	2	Drop Invalid Entries	Removed “Unnamed” columns, NaNs, and invalid timestamps.	To ensure consistency and prevent errors during training.	Incomplete rows removed; consistent column names established.
Data Cleaning & Preprocessing	3	Handle Outliers	Filtered Heart Rate (<30 or >220) and SpO_2 (<70 or >100).	To retain only physiologically plausible sensor readings.	Health dataset cleaned of extreme noise.
Data Cleaning & Preprocessing	4	Convert Data Types	Converted timestamps to datetime; extracted hour and day-of-week.	To enable time-based feature engineering for daily/weekly routines.	Time features (hour, day-of-week) added to predictors.
Data Cleaning & Preprocessing	5	Encode & Standardize	OneHotEncoder for categorical variables; StandardScaler for numeric.	To prepare data for ML algorithms that require normalized inputs.	Final feature matrix ready with scaled numeric and encoded categorical features.
Data Labeling	6	Define Targets	Defined multi-class labels for activity and binary labels for anomaly.	To frame supervised ML tasks clearly.	y_act: 4 classes (Movement Activities); y_anom: 2 classes (normal vs anomaly).
Model Training & Evaluation	7	Baseline Model Sweep	Trained Logistic Regression, Random Forest, SVC, KNN, Gradient Boosting on both safety-only and combined datasets.	To identify the strongest base algorithm per task.	Activity: SVC best (F1 ≈ 0.31); Anomaly: Logistic Regression selected (F1 = 1.0, likely artifact).
Hyperparameter Tuning & Optimization	8	RandomizedSearchCV / GridSearchCV	Performed tuning on the best models.	To optimize performance and reduce bias/variance.	SVC: kernel=poly, γ=scale, C≈8.11 → F1=0.3055. Logistic Regression: penalty=l2, C≈0.0052 → F1=1.0 (suspect).
Model Evaluation	9	Holdout Validation	Split data into training/testing and generated classification reports.	To evaluate generalization beyond cross-validation.	Weak performance for activity; anomaly results flagged as unrealistic.
Model Evaluation	10	Ensemble Learning	Tested Voting (LR+RF+SVC) and Stacking (LR+RF+SVC→LR).	To check if combining models improved robustness.	Activity: F1=0.3050 (Voting), 0.2921 (Stacking) — no gain. Anomaly ensembles ≈ 0.64–0.66 F1.
Robustness Checks	11	LOSO Cross-Validation	Performed leave-one-subject-out CV.	To assess robustness to unseen subjects.	LOSO mean F1=0.0499; Clean/Noisy sets reported F1=1.0 (artifact/leakage).
Additional Experiments	12	Gradient Boosting with Rolling Features	Tested rolling feature representation for activity.	To explore temporal signal representations.	Columns generated, but no stable performance metrics recorded.

Stage	Step	Action / Method	Description	Rationale	Key Outcome / Result
Data Cleaning & Preprocessing	Handle Missing Data	Impute with median and 0	Replaced '-' placeholders with NaN and filled missing values.	To create a complete dataset without missing values that could cause model errors.	Impact Force Level imputed with 0, Post-Fall Inactivity Duration with 0.0, resulting in no missing data.
Data Cleaning & Preprocessing	Feature Engineering	Create time_since_last_event and time-based features	A new feature was created to measure time between events. The Timestamp column was used to extract hour_of_day, minute_of_day, and day_of_week.	To capture inactivity patterns and daily routines, providing valuable signals for the model.	A new feature, time_since_last_event, was added, and time-based columns were extracted.
Data Cleaning & Preprocessing	Categorical Encoding	OneHotEncoder	Categorical features like Movement Activity and Location were converted into a numerical binary format.	To prepare data for machine learning models, which require numerical inputs.	25 total features, including new one-hot encoded columns, were created.
Data Cleaning & Preprocessing	Data Standardization	StandardScaler	The numerical features were scaled to have a mean of 0 and a standard deviation of 1.	To prevent features with larger values from dominating the model.	Numerical features were standardized, preparing the data for models like SVM.
Data Splitting & Resampling	Split Data	train_test_split with stratify=y	The dataset was divided into 70% for training and 30% for testing.	To evaluate the model on unseen data. stratify ensures class distribution is maintained.	Training set: 7000 samples; Testing set: 3000 samples.
Data Splitting & Resampling	Handle Imbalance	SMOTE (Synthetic Minority Oversampling Technique)	SMOTE was applied to the training data to create synthetic samples of the minority "Fall" class.	To prevent the model from becoming biased towards the majority "No Fall" class.	The training set became perfectly balanced with 6651 samples for each class.
Model Training & Evaluation	Train Baseline Model	Logistic Regression	A baseline model was trained on the resampled training data and evaluated on the test data.	To establish an initial performance benchmark.	The model achieved a perfect fall recall of 1.00, indicating data leakage.
Model Training & Evaluation	Evaluate Models	Decision Tree, Random Forest, SVM, kNN, Gradient Boosting	Multiple models were trained and evaluated on the test set.	To identify the most suitable algorithm for the task.	Gradient Boosting was selected as the best option, with Fall Recall: 0.80 and ROC AUC: 0.89.
Hyperparameter Tuning & Optimization	Fine-tune Model	GridSearchCV	The Gradient Boosting model's hyperparameters (n_estimators, learning_rate, max_depth) were tuned.	To optimize the model's performance and find the best configuration.	The best parameters were found to be {'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 50}.
Hyperparameter Tuning & Optimization	Final Evaluation	Final Model on Test Set	The optimized Gradient Boosting model was evaluated on the test set.	To get a realistic measure of its performance on unseen data.	The model achieved a Fall Recall of 0.92, a significant improvement.
Post-Training Analysis & Finalization	Analyze Feature Importance	feature_importances_	The model's feature importance scores were visualized.	To understand which features were most influential in the model's predictions.	Movement Activity_No Movement was identified as the overwhelmingly most important feature.
Post-Training Analysis & Finalization	Adjust Threshold	A new threshold of 0.4 was applied to the model's predictions.	To prioritize the most critical metric (recall) for this safety-first application.	The fall recall was increased to an exceptional 0.97, correctly identifying 146 of 150 falls.	
Post-Training Analysis & Finalization	Save Model	joblib.dump	The final model, scaler, encoder, feature names and optimal threshold were saved.	To make the model and its entire preprocessing pipeline ready for deployment in a production environment.	All necessary artifacts (fall_detection_model.joblib, scaler.joblib, encoder.joblib, feature_names.joblib and prediction_threshold.txt) were successfully saved.