# Package 'autoFlagR'

January 15, 2026

**Type** Package

**Title** AI-Driven Anomaly Detection for Data Quality

**Version** 1.0.0

**Description** Automated data quality auditing using unsupervised machine learning. Provides AI-driven anomaly detection for data quality assessment, primarily designed for Electronic Health Records (EHR) data, with benchmarking capabilities for validation and publication. Methods based on: Liu et al. (2008) <doi:10.1109/ICDM.2008.17>, Breunig et al. (2000) <doi:10.1145/342009.335388>.

**License** MIT + file LICENSE

**URL** https://github.com/vikrant31/autoFlagR, https://vikrant31.github.io/autoFlagR/

**BugReports** https://github.com/vikrant31/autoFlagR/issues

**Encoding** UTF-8

**Imports** isotree, dbscan, dplyr, ggplot2, pROC, PRROC, knitr, gt, scales, rmarkdown (>= 2.0)

**Suggests** testthat, pkgdown, ggnewscale

**VignetteBuilder** knitr

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**Author** Vikrant Dev Rathore [aut, cre]

**Maintainer** Vikrant Dev Rathore <rathore.vikrant@gmail.com>

**Repository** CRAN

**Date/Publication** 2026-01-15 17:40:31 UTC

## Contents

---

`calculate_benchmark_metrics`
*Calculate Benchmarking Metrics*

---

## Description

Computes AUC-ROC, AUC-PR, and Top-K Recall metrics for evaluating anomaly detection performance against ground truth.

## Usage

```
calculate_benchmark_metrics(scores, ground_truth, contamination = 0.05)
```

## Arguments

| | |
|---|---|
| `scores` | Numeric vector of anomaly scores |
| `ground_truth` | Binary vector (0/1) of true anomaly labels |
| `contamination` | Expected proportion of anomalies |

## Value

List of benchmarking metrics

---

`calculate_feature_importance`
*Calculate Feature Importance for Anomalies*

---

## Description

Calculates which feature contributes most to each record's anomaly score. This provides a "reason code" explaining why each record was flagged as anomalous.

## Usage

```
calculate_feature_importance(flagged_data, metadata, top_k = 1, max_cols = 10)
```

### Arguments

| | |
|---|---|
| flagged_data | A data frame with anomaly scores and is_anomaly flags, typically the output of flag_top_anomalies(). |
| metadata | Metadata from prep_for_anomaly(), containing information about numeric and categorical columns. |
| top_k | Integer indicating how many top contributing features to consider. Default is 1 (returns only the most important feature). |
| max_cols | Integer indicating maximum number of columns to consider for feature importance. If NULL, uses all columns. Default is 10 for performance. |

### Value

The input data frame with additional columns:

| | |
|---|---|
| reason_feature | Name of the feature contributing most to the anomaly |
| reason_value | The value of that feature for this record |
| reason_code | A brief description combining feature name and value |
| reason_deviation | The standardized deviation from the median (for numeric) or frequency (for categorical) |

### Examples

```
data <- data.frame(
  patient_id = 1:50,
  age = rnorm(50, 50, 15),
  cost = rnorm(50, 10000, 5000)
)
scored_data <- score_anomaly(data, id_cols = "patient_id")
flagged_data <- flag_top_anomalies(scored_data)
metadata <- attr(scored_data, "metadata")
flagged_data <- calculate_feature_importance(flagged_data, metadata)
```

---

## create_rmd_template *Create R Markdown Template*

---

### Description

Creates a default R Markdown template for audit reports.

### Usage

```
create_rmd_template(template_path, output_format = "pdf")
```

### Arguments

| | |
|---|---|
| `template_path` | Path where the template should be created |
| `output_format` | Output format ("pdf", "html", or "docx") |

### Value

No return value, called for side effects. Creates an R Markdown template file at the specified path.

---

`extract_benchmark_metrics`

*Extract Benchmark Metrics from Scored Data*

---

### Description

Extracts benchmarking metrics from a data frame that was scored with `score_anomaly()` using ground truth labels.

### Usage

```
extract_benchmark_metrics(scored_data)
```

### Arguments

| | |
|---|---|
| `scored_data` | A data frame with anomaly scores, typically the output of `score_anomaly()` with a `ground_truth_col` specified. |

### Value

A list containing benchmarking metrics, or NULL if no metrics are available.

### Examples

```
data <- data.frame(
  patient_id = 1:50,
  age = rnorm(50, 50, 15),
  is_error = sample(c(0, 1), 50, replace = TRUE, prob = c(0.95, 0.05))
)
scored_data <- score_anomaly(data, ground_truth_col = "is_error")
metrics <- extract_benchmark_metrics(scored_data)
print(metrics$auc_roc)
```

flag_top_anomalies *Flag Top Anomalies Based on Score Threshold*

## Description

Categorizes records as anomalous or normal based on their anomaly scores, using either a fixed threshold or a contamination rate.

## Usage

```
flag_top_anomalies(data_with_scores, threshold = NULL, contamination = 0.05)
```

## Arguments

| | |
|---|---|
| data_with_scores | A data frame containing an anomaly_score column, typically the output of score_anomaly(). |
| threshold | Numeric value between 0 and 1. Records with anomaly_score >= threshold are flagged as anomalous. If NULL (default), uses the contamination rate from the score_anomaly() attributes. |
| contamination | Numeric value between 0 and 1. If threshold is NULL, this proportion of records with the highest scores will be flagged. Default is 0.05 (5%). |

## Value

The input data frame with an additional is_anomaly logical column indicating whether each record is flagged as anomalous.

## Examples

```
data <- data.frame(
  patient_id = 1:50,
  age = rnorm(50, 50, 15),
  cost = rnorm(50, 10000, 5000)
)
scored_data <- score_anomaly(data)
flagged_data <- flag_top_anomalies(scored_data, contamination = 0.05)
```

`generate_audit_report` *Generate Automated Data Quality Audit Report*

## Description

Executes the complete anomaly detection pipeline (preprocessing, scoring, flagging) and generates a professional PDF, HTML, or DOCX report with visualizations and prioritized audit listings.

## Usage

```
generate_audit_report(
  data,
  filename = "dq_audit_report",
  output_dir = NULL,
  output_format = "pdf",
  method = "iforest",
  contamination = 0.05,
  top_n = 100,
  id_cols = NULL,
  exclude_cols = NULL,
  ground_truth_col = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `data` | A data frame containing the data to be audited. |
| `filename` | Character string for the output file (without extension). Default is "dq_audit_report". |
| `output_dir` | Character string specifying the directory for the output file. If NULL (default), uses tempdir(). Users should specify a directory explicitly for production use. |
| `output_format` | Character string indicating the output format. Options: "pdf" (default), "html", or "docx" (for editable Word document). Note: PDF format provides the best color rendering for heat map tables. DOCX format is generated by first creating a PDF, then converting to DOCX. |
| `method` | Character string indicating the anomaly detection method. Passed to `score_anomaly()`. Default is "iforest". |
| `contamination` | Numeric value between 0 and 1. Passed to `score_anomaly()`. Default is 0.05. |
| `top_n` | Integer indicating the number of top anomalous records to display in the prioritized audit listing. Default is 100. |
| `id_cols` | Character vector of column names to exclude from scoring. Passed to `prep_for_anomaly()`. |
| `exclude_cols` | Character vector of additional columns to exclude. Passed to `prep_for_anomaly()`. |
| `ground_truth_col` | Character string naming a column with ground truth labels. If provided, benchmarking metrics will be included in the report. |
| `...` | Additional arguments passed to `score_anomaly()`. |

## Value

Invisibly returns the path to the generated report file.

## Examples

```
data <- data.frame(
  patient_id = 1:50,
  age = rnorm(50, 50, 15),
  cost = rnorm(50, 10000, 5000),
  gender = sample(c("M", "F"), 50, replace = TRUE)
)
# Generate HTML report (fastest, no LaTeX/pandoc required)
generate_audit_report(data, filename = "my_audit", output_format = "html",
                      output_dir = tempdir())
```

---

# get_top_anomalies *Get Top Anomalous Records*

---

## Description

Convenience function to extract the top N most anomalous records from scored data.

## Usage

```
get_top_anomalies(scored_data, n = 100)
```

## Arguments

| | |
|---|---|
| scored_data | A data frame with anomaly scores. |
| n | Integer indicating the number of top records to return. Default is 100. |

## Value

A data frame containing the top N most anomalous records, sorted by anomaly_score (descending).

## Examples

```
data <- data.frame(
  patient_id = 1:50,
  age = rnorm(50, 50, 15),
  cost = rnorm(50, 10000, 5000)
)
scored_data <- score_anomaly(data)
top_10 <- get_top_anomalies(scored_data, n = 10)
```

prep_for_anomaly *Prepare Data for Anomaly Detection*

## Description

Preprocesses data for unsupervised anomaly detection by handling identifiers, scaling numerical features, and encoding categorical variables.

## Usage

```
prep_for_anomaly(
  data,
  id_cols = NULL,
  exclude_cols = NULL,
  scale_method = "mad"
)
```

## Arguments

| | |
|---|---|
| data | A data frame containing the data to be preprocessed. |
| id_cols | Character vector of column names to exclude from scoring (e.g., patient IDs, encounter IDs). If NULL, attempts to auto-detect common ID column patterns. |
| exclude_cols | Character vector of additional columns to exclude from scoring. Default is NULL. |
| scale_method | Character string indicating the scaling method for numerical variables. Options: "mad" (Median Absolute Deviation, default), "minmax" (min-max normalization), or "none" (no scaling). |

## Value

A list containing:

| | |
|---|---|
| prepared_data | A numeric matrix ready for anomaly detection |
| metadata | A list with mapping information:<br>• original_data: The original data frame<br>• id_cols: Column names used as identifiers<br>• numeric_cols: Column names of numeric variables<br>• categorical_cols: Column names of categorical variables<br>• excluded_cols: Column names excluded from scoring |

## Examples

```
data <- data.frame(
  patient_id = 1:20,
  age = rnorm(20, 50, 15),
  cost = rnorm(20, 10000, 5000),
  gender = sample(c("M", "F"), 20, replace = TRUE)
)
prep_result <- prep_for_anomaly(data, id_cols = "patient_id")
```

---

`score_anomaly` *Score Anomalies Using Unsupervised Machine Learning*

---

## Description

Calculates anomaly scores for each record using Isolation Forest or Local Outlier Factor algorithms. Optionally evaluates performance against ground truth labels for benchmarking.

## Usage

```
score_anomaly(
  data,
  method = "iforest",
  contamination = 0.05,
  ground_truth_col = NULL,
  id_cols = NULL,
  exclude_cols = NULL,
  ...
)
```

## Arguments

| | |
|---|---|
| `data` | A data frame containing the data to be scored. |
| `method` | Character string indicating the anomaly detection method. Options: "iforest" (Isolation Forest, default) or "lof" (Local Outlier Factor). |
| `contamination` | Numeric value between 0 and 1 indicating the expected proportion of anomalies in the data. Default is 0.05 (5%). |
| `ground_truth_col` | Character string naming a column in `data` that contains binary ground truth labels (0/1 or FALSE/TRUE) for known anomalies. If provided, benchmarking metrics will be calculated. Default is NULL. |
| `id_cols` | Character vector of column names to exclude from scoring. Passed to `prep_for_anomaly()`. |
| `exclude_cols` | Character vector of additional columns to exclude. Passed to `prep_for_anomaly()`. |
| `...` | Additional arguments passed to the underlying algorithm. For Isolation Forest: `ntrees`, `sample_size`, `max_depth`. For LOF: `minPts` (number of neighbors; deprecated `k` is converted to `minPts`). |

## Value

A data frame with the original data plus an `anomaly_score` column. If `ground_truth_col` is provided, the result includes an attribute `benchmark_metrics` containing: `auc_roc` (Area Under the ROC Curve), `auc_pr` (Area Under the Precision-Recall Curve), `top_k_recall` (List of recall values for top K records: K = 10, 50, 100, 500), and `contamination_rate` (Actual proportion flagged as anomalous).

## Examples

```
data <- data.frame(
  patient_id = 1:50,
  age = rnorm(50, 50, 15),
  cost = rnorm(50, 10000, 5000)
)
scored_data <- score_anomaly(data, method = "iforest", contamination = 0.05)
```

# Index