Computer Science Department
CS667 – Practical Data Science (CRN: 72872)
Fall 2025

Project #3 / Due 25-Nov-2025

1. Project Overview

In this project, you will explore the task of **anomaly detection** in a tabular dataset (or multivariate numeric dataset). The goal is to build and compare at least two unsupervised (or semi-supervised) anomaly detection methods one based on a:

- **Gaussian Mixture Model** (**GMM**) and one based on the
- **Isolation Forest** algorithm.

You will carry out the **full pipeline**: data preparation, feature engineering/selection, model building and tuning, model evaluation (despite limited labels for anomalies), interpretation of results, and reporting.

The project is intended to help you understand:

- ✓ How anomaly detection differs from standard supervised classification,
- ✓ How algorithmic choices matter,
- ✓ How feature preparation and contamination assumptions influence results, and
- ✓ How to interpret an "anomaly score" model in practice.

2. Learning Objectives

By the end of the project, you will:

- Understand what constitutes an anomaly (or outlier / novelty) in a data-context, and how anomaly detection differs from standard classification/regression.
- Gain hands-on experience using Python libraries (such as scikit-learn and/or pyod or similar) to implement and compare two anomaly detection methods: GMM and Isolation Forest.
- Develop skills in data preparation specific to anomaly detection: e.g., handling skewed distributions, scaling/normalizing features, dealing with class imbalance or unlabeled anomalies, selecting features that help separate "normal" from "anomalous" behavior.
- Learn to split data appropriately for anomaly detection (often training on mostly "normal" data, then testing on known anomalies or injecting synthetic anomalies).
- Tune model hyperparameters (e.g., number of mixture components in GMM, contamination rate in Isolation Forest) and evaluate model performance using appropriate metrics (e.g., precision-recall, Area Under ROC, F1 for imbalanced data, or anomaly score distributions).
- Interpret and visualize the resulting models: e.g., understand how the GMM clusters correspond to "normal" vs "abnormal", visualize anomaly score distributions, highlight the most anomalous observations and reason about whether they are genuine.

- Produce a professional report (and/or Jupyter Notebook) summarizing methodology, findings, limitations, and practical implications of deploying anomaly detection in a business or operational environment.

3. Project Steps

Here is a recommended (best practices) sequence of steps.

3.1 Data Preparation

- Ingest/load the dataset (see below details about the dataset.)
- Inspect the dataset: feature types, missing values, distributions, basic summary statistics (mean, median, variance, skewness) and correlation matrix.
- Handle missing values (e.g., imputation or dropping features/rows).
- If necessary, convert categorical features (if any) to numeric (one-hot encoding or ordinal encoding) or exclude them if better.
- Scale/normalize features (e.g., using StandardScaler or MinMaxScaler) because many algorithms assume similarly scaled variables.
- Optional: Explore and potentially remove extremely skewed features or transform them (e.g., log transform).
- Visualize feature distributions, pair-wise plots, anomaly score histogram (initially maybe using a simpler method) to understand data structure.

3.2 Feature Selection & Data Splitting

- Decide on which features to include in modelling (for example exclude ID columns, timestamps unless relevant, very high-cardinality features, etc.).
- Split data into “training” and “testing” sets. In many anomaly-detection workflows, one uses the training set to represent mostly “normal” behavior only (if labels permit) and uses the test set to include both normal + known anomalies (if labels exist). If labels don’t exist (pure unsupervised), you may still hold out a subset for evaluation.
- If labeled anomalies exist keep aside a portion of them for final testing only (do not include in training).
- Consider specifying the “contamination” rate (expected proportion of anomalies) if using Isolation Forest (or similarly, the prior for the GMM).
- (Optional) If the number of anomalies is extremely small relative to normal cases, you may consider up-sampling anomalies for evaluation or injecting synthetic anomalies for educational purposes (but clearly document this).

3.3 Model Building

- **GMM Approach:**
- Using something like **sklearn.mixture.GaussianMixture**, fit a GMM on the training set (normal data). Choose number of components (k) via e.g., Bayesian Information Criterion (BIC) or Akaike Information Criterion (AIC).
- Once the mixture model is fitted, compute for each observation the log-likelihood (or negative log-likelihood) under the mixture model. Observations with very low likelihood can be flagged as anomalies.

- Optionally set a threshold (e.g., lowest x% of likelihoods) for classification of anomaly vs normal (or use a percentile).
- Tune parameters (number of components, covariance type, threshold percentile, regularization) based on validation/test set.
- **Isolation Forest Approach:**
- Using **sklearn.ensemble.IsolationForest**, fit on training data. Choose hyperparameters such as n_estimators, max_samples, contamination (expected fraction of anomalies), max_features.
- After training, compute the "anomaly score" (via decision_function or score_samples) for the test set. Lower scores correspond to more anomalous observations.
- Set a threshold (often aligned with contamination) to flag anomalies.
- Tune hyperparameters (e.g., contamination from 0.01 to 0.1, max_features) and compare performance.

3.4 Model Evaluation

- For labelled anomalies (if available), compute metrics: precision, recall, F1-score, ROC-AUC, PR-AUC. Because anomalies are rare, PR-AUC (precision-recall AUC) is often more informative.
- Plot ROC curve, precision-recall curve, and/or histogram of anomaly scores for normal vs anomalous observations to visualize separation.
- Compute confusion matrix at chosen threshold.
- Compare performance of GMM vs Isolation Forest: Which flagged more true anomalies? Which had fewer false positives (normal points flagged as anomaly)? Discuss trade-offs (e.g., more sensitive vs more selective).
- Also assess if the models are overly tuned or overfit: e.g., if training set had no anomalies, ensure you haven't indirectly used anomalies in training.
- (Optional) If you have unlabeled data, you might examine top-k most anomalous points, manually inspect them (or visualize them) and comment on whether they "make sense" as anomalies.

3.5 Model Interpretation & Insights

- For the GMM: Inspect the learned components (means, covariances) and interpret which component(s) represent the "bulk/normal" clusters vs smaller, low-probability clusters. Are anomalies emerging from sparse components?
- For Isolation Forest: Look at feature importances (if available) or look at how many splits / path lengths contributed to anomaly decisions. Investigate which features contribute most to anomaly scores.
- Perform a feature importance or sensitivity analysis: For the flagged anomalies, what are the distinguishing feature values? Use visualization: box plots, violin plots, scatter plots for normal vs anomaly groups, maybe t-SNE or PCA projection colored by anomaly score.
- Discuss any domain-specific insights: e.g., in the dataset context, what might the anomalies represent? Are they plausible? Are there false positives that would make sense in domain but are flagged due to model assumptions?
- Reflect on limitations: e.g., model assumes normal behavior is well-captured by training data, anomalies are rare and different; GMM assumes Gaussian mixtures; Isolation Forest assumes random splits isolate anomalies fast; both may struggle if

anomalies are subtle or very similar to normal; high dimensionality or correlated features may reduce performance; threshold setting is arbitrary; lack of labels may limit evaluation.

4. Reporting & Deliverables

You should submit the following deliverables:

- A **Jupyter Notebook** (or equivalent python script) containing:
  - A clear description of the project and objective (in markdown).
  - Data loading, exploration, preprocessing steps with commentary.
  - Feature selection and data splitting logic.
  - Model building for both GMM and Isolation Forest, including hyper-parameter tuning.
  - Model evaluation: metrics, curves, confusion matrix, visualization.
  - Interpretation: what the models learned, what features mattered, what anomalies were flagged.
  - Reflection on results, limitations, and potential next steps for improvement or deployment.
- A **written report (optional)** (could be a PDF) summarizing:
  - Background & problem statement.
  - Data description.
  - Methodology (briefly describing each step).
  - Results summary (tables and figures).
  - Contrast of GMM vs Isolation Forest (which performed better under what criteria).
  - Domain-insights: what the flagged anomalies might mean, any interesting findings.
  - Practical implications: e.g., in a business context, how you deploy anomaly detection, alerting thresholds, how you would monitor model performance over time, how you handle false positives/negatives.
  - Limitations and proposed future work.

Dataset's (financial_anomaly_data.xlsx) Metadata Info

With this dataset, you could perform various analyses like:

- Detecting anomalies in transaction amounts (e.g., unusually high transactions).
- Identifying irregular transaction types for specific accounts.
- Recognizing unusual patterns based on transaction timestamps or locations.
- Tracking spending behaviors based on merchants.

This Data set contains following columns:

Timestamp: This column records the date and time when the transaction occurred. It helps in understanding the temporal aspect of transactions, such as patterns over time, frequency, and clustering of activities.

TransactionID: An identification number assigned to each transaction. It serves as a unique identifier for referencing or tracking specific transactions.
AccountID: This field represents the unique identifier associated with the bank account involved in the transaction. It links multiple transactions to a specific account, enabling analysis on a per-account basis.
Amount: The monetary value involved in the transaction. This column provides information about the financial magnitude of each transaction, which is crucial for anomaly detection since unusually high or low values might signify irregularities.
Merchant: Specifies the entity or business involved in the transaction. This information helps in categorizing transactions (e.g., retail, online, restaurant) and identifying patterns related to specific merchants.
TransactionType: Describes the nature or category of the transaction, whether it's a withdrawal, deposit, transfer, payment, etc. This column helps in understanding the purpose or direction of the transaction.
Location: Indicates the place where the transaction occurred. It could be a physical location (e.g., city, country) or an identifier (e.g., store code, online portal), aiding in analyzing geographical spending patterns or detecting anomalies based on unusual transaction locations.