# Machine Learning for Fraud Detection in Online Financial Transactions: A Literature Review

## Introduction and Domain Overview

Financial fraud in online transactions is a pervasive and costly problem, affecting industries worldwide and undermining trust in financial systems (PricewaterhouseCoopers, 2022; KPMG, 2022). For example, a 2022 global survey by PwC reported that 56% of companies experienced fraud that year (PricewaterhouseCoopers, 2022). Fraud can take many forms – from payment card fraud and money laundering to insurance scams and financial statement falsification – and it has severe economic and reputational consequences for businesses and economies (Abdallah et al., 2016; PricewaterhouseCoopers, 2022). Traditionally, organizations relied on manual reviews and expert-defined rules to detect fraud. However, these approaches struggle with the scale, complexity, and adaptive nature of modern fraud: high volumes of transactions, subtle fraudulent patterns, and continuously evolving tactics by fraudsters make purely manual or static rule-based systems inadequate (West & Bhattacharya, 2016; Ryman-Tubb et al., 2018). In one case, a large bank’s legacy rule system caught barely **5.6%** of fraudulent transactions (recall) with about **40%** precision, illustrating the extreme limitations of manual rules (Ryman-Tubb et al., 2018).

Given these challenges, machine learning (ML) has become a crucial tool in fraud detection. ML algorithms can automatically learn patterns of legitimate and fraudulent behavior from historical data, enabling more accurate and timely detection of anomalies and suspicious activities (Phua et al., 2010; West & Bhattacharya, 2016). Unlike static rules, ML models can leverage a multitude of features and adapt to new fraud patterns, helping organizations catch fraud that would evade simple heuristic checks (Phua et al., 2010; West & Bhattacharya, 2016). The importance of ML in this domain is underscored by the growing body of research on the topic: over the past decade, numerous studies have applied ML techniques to financial fraud detection, with a particular focus on online transaction fraud (especially in banking and payment platforms) as well as related areas like corporate accounting fraud and insurance fraud (Ali et al., 2022; Al-Hashedi & Magalingam, 2021). Scholars often classify fraud into categories such as banking fraud, corporate fraud, and insurance fraud, with banking fraud (e.g. credit card and online payment fraud) being the most studied due to its prevalence (Nicholls et al., 2021; Al-Hashedi & Magalingam, 2021). Overall, the domain of online financial fraud detection is both high-impact and inherently challenging – making it an important area for advanced ML solutions.

## Machine Learning Approaches to Fraud Detection

A wide range of machine learning approaches have been explored for fraud detection, each with its own strengths and weaknesses. *Figure 1* illustrates the major categories of ML techniques applied in this field and their typical usage.

### Supervised Learning Techniques

**Supervised learning** (using labeled examples of fraud and non-fraud) is by far the most widely used approach in fraud detection research (Al-Hashedi & Magalingam, 2021; Hernandez Aros et al., 2024). In supervised methods, classification algorithms are trained on historical transaction data labeled as legitimate or fraudulent, then used to predict the class of new transactions. A variety of classifiers have been applied, including:

* **Logistic Regression (LR):** A simple and interpretable linear model often used as a baseline. Despite its simplicity, LR has been effective in many fraud studies (Achakzai & Juan, 2022). Its learned coefficients can provide insight into feature importance, aiding explainability. However, LR may struggle with complex non-linear patterns unless feature engineering is thorough.
* **Decision Trees (DT):** Tree-based classifiers (e.g. CART) that split data on feature thresholds. They are intuitive and explainable, modeling decision rules that analysts can interpret (Ahmed et al., 2016). Single trees can overfit, but they form the basis of powerful ensemble methods.
* **Support Vector Machines (SVM):** SVMs find an optimal hyperplane separating classes and are effective in high-dimensional feature spaces. They have been popular in fraud detection literature, often yielding good accuracy especially with balanced data (Ali et al., 2022). SVMs can handle non-linear boundaries via kernel functions, but can be slow on large datasets and their results are less interpretable than tree models.
* **Nearest Neighbor Methods (k-NN):** Used in some studies for anomaly or outlier detection by measuring distance to known examples. k-NN is simple and can detect clusters of fraud by proximity, but it scales poorly to large datasets and does not provide reasoning behind a classification.
* **Naïve Bayes (NB):** A probabilistic classifier assuming feature independence. Perhaps surprisingly, recent surveys indicate NB was among the most frequently used algorithms in financial fraud papers up to around 2020 (Ali et al., 2022). Its ease of implementation and speed are advantages, though its independence assumption is often violated by real transaction data.
* **Ensemble Classifiers:** Methods like Random Forests (RF) and gradient boosting (e.g. XGBoost, LightGBM) combine many decision trees to improve accuracy. Ensembles have proven very effective in fraud detection: Random Forests reduce overfitting by bagging (training multiple trees on bootstrap samples) and generally outperform individual models in many studies (Alarfaj et al., 2022), while boosting methods like XGBoost handle class imbalance and complex patterns well (Alwadain et al., 2023). Ensemble models often achieve the highest precision and recall in benchmarks, though at the cost of interpretability. Recent work also explores **stacking** ensembles (combining heterogeneous models via a meta-learner) to further improve robustness (Achakzai & Juan, 2022; Alarfaj et al., 2022).

Supervised methods benefit from directly optimizing discrimination between fraud and non-fraud transactions, and they can leverage rich labeled datasets when available. Indeed, literature reviews show that a majority of fraud detection studies have used supervised classification, with algorithms like logistic regression, decision trees, SVM, neural networks and their ensembles being among the top choices (Ali et al., 2022; Hernandez Aros et al., 2024). The downside is their reliance on large labeled datasets – which in fraud scenarios can be limited or skewed, since frauds are rare and labels may be delayed or incomplete. This motivates the use of unsupervised and semi-supervised methods, discussed next.

### Unsupervised and Semi-Supervised Approaches

**Unsupervised learning** methods detect fraud by identifying anomalies or patterns in data without the need for labeled examples. Given that fraudulent transactions are by nature outliers in the distribution of normal transactions, anomaly detection is a logical strategy, especially when labels are scarce. Techniques include:

* **Clustering algorithms:** e.g. k-Means, DBSCAN, or hierarchical clustering to group similar transactions and flag those that fall in very small or distant clusters as potential frauds. Clustering has been applied to detect unusual spending patterns or customer segments, but surveys find it has been used considerably less than supervised classification in fraud contexts (Ahmed et al., 2016; Al-Hashedi & Magalingam, 2021). One challenge is tuning such methods to avoid flagging too many innocuous anomalies (high false positives).
* **One-class classification:** Methods like one-class SVM or autoencoders (discussed below under deep learning) which train on the distribution of **legitimate** transactions only, then identify outliers that don’t fit this learned "normal" profile (Ahmed et al., 2016). These approaches are useful when confirmed fraud examples are extremely few – the model essentially learns the “normal” class. For instance, one-class SVM has shown success in credit card fraud detection by under-sampling normal data to better learn the boundary between normal and abnormal transactions.
* **Statistical outlier detection:** Simpler analytics examine transactions for extreme values or deviations on key features (e.g. very large amounts or an unusual time/location). While easy to implement and often overlapping with expert rules, these can suffer high false-positive rates if used alone, since not every outlier is fraudulent.

Semi-supervised techniques, which combine a small labeled set with a large unlabeled set, have also been explored. For example, *active learning* has been suggested as a way to iteratively label the most suspicious unlabeled transactions and incrementally improve the model (Ali et al., 2022). Semi-supervised clustering or graph-based methods can propagate limited label information through clusters or networks of transactions (some researchers have applied label propagation in transaction graphs to catch frauds linked to known bad actors). Overall, unsupervised and semi-supervised methods are valuable for detecting new or evolving fraud patterns that weren’t present in the training labels. However, they can be difficult to validate (since flagged anomalies may or may not be true fraud) and typically need careful tuning. Recent surveys note that unsupervised anomaly detection, while conceptually important, remains underutilized in published fraud studies compared to supervised methods (Al-Hashedi & Magalingam, 2021; Ali et al., 2022), indicating a gap where more research is expected in the future.

### Deep Learning Techniques

Deep learning has increasingly been applied to fraud detection as larger datasets and more complex sequential or relational patterns are involved. Neural networks can automatically learn feature representations from raw data, which is useful in a domain where high-level temporal or network patterns (e.g. spending sequences, or connections between entities) matter. Key deep learning approaches include:

* **Artificial Neural Networks (ANN / MLP):** Multi-layer perceptrons have been used in fraud detection for years, often with one or two hidden layers (Ryman-Tubb et al., 2018). They can capture non-linear relationships between features better than linear models. Studies frequently include a basic ANN in model benchmarks, and they have shown solid performance – in one recent review, ANN was among the top five most-used model types (Ryman-Tubb et al., 2018). However, simple MLPs may not dramatically outperform tree ensembles on tabular transaction data unless the feature space is very complex.
* **Convolutional Neural Networks (CNN):** Although CNNs are best known for image data, they have been applied to transaction fraud by treating input features or time-series as “grids” or images. For instance, some work transforms sequences of transactions into a 2D temporal matrix or "transaction amount heatmap" and applies CNNs to detect abnormal patterns. CNNs can automatically perform a form of feature extraction, and combining CNN-extracted features with traditional classifiers has been found to improve detection performance in some cases.
* **Recurrent Neural Networks (RNN):** RNNs, including LSTM (Long Short-Term Memory) and GRU networks, are well-suited for sequential data and have been used to model the sequence of transactions on an account or card. An LSTM can learn temporal patterns such as a fraudster testing a stolen card with small transactions before making large purchases. RNN-based models have shown promise in capturing long-term dependencies in fraud data, though they require careful handling of issues like class imbalance and can be slower to train.
* **Autoencoders:** These are unsupervised neural networks trained to reconstruct input data, thereby learning a compressed representation of normal patterns. In fraud detection, autoencoder-based anomaly detection works by training on predominantly genuine transactions – then fraudulent transactions, being rare and poorly reconstructed, yield higher reconstruction error and can be flagged. Autoencoders (including variants like Variational Autoencoders) have been used to detect novel fraud patterns and for dimensionality reduction (Ali et al., 2022). They are particularly useful when labeled fraud examples are scarce, and have been reported in several works. However, autoencoders require careful tuning to avoid simply learning the identity function, and they may not capture all types of fraud behavior if those patterns are very subtle.
* **Generative Adversarial Networks (GANs):** GANs involve a generator and discriminator network in competition. Though more common for data synthesis, some studies use GANs to generate synthetic fraudulent examples for training (as data augmentation) or to detect fraud by training a generator to mimic normal transactions and having the discriminator spot the real vs. generated instances. GANs are harder to train, but they have been proposed as a way to tackle class imbalance by generating additional minority-class samples.
* **Graph Neural Networks (GNN):** A newer frontier in fraud detection uses GNNs to exploit the graph structure of financial transactions. Transactions can be represented as networks connecting entities (customers, merchants, IP addresses, etc.), and GNNs such as Graph Convolutional Networks can propagate information along these links to detect suspicious subgraphs or entities. GNNs have achieved excellent results in scenarios like detecting organized fraud rings or money laundering networks, outperforming some traditional models (Zhao & Bai, 2022). Their strength is in modeling relational data – for example, if multiple fraudulent accounts all send money to the same payee, a graph-based model can catch this pattern even if each account individually had no prior fraud record. The challenge is that graph models are complex and not yet as widely understood outside of specialized researchers, but interest in this area is growing.

Deep learning models generally require more data and computational resources, but they offer the ability to capture complex, high-dimensional patterns (and even derive features automatically, reducing some need for manual feature engineering). In recent years, *hybrid* models have appeared that combine deep learning with other methods – for example, using an autoencoder to generate features for an XGBoost classifier, or combining a GNN with an ensemble method (this aims to get the best of both worlds: deep models’ pattern recognition with the interpretability or simplicity of more traditional models). Such hybrid approaches are an emerging trend to maximize performance and robustness (Zhao & Bai, 2022).

One notable direction is applying sequence models and temporal ensembling for fraud, acknowledging that fraud is often a dynamic phenomenon. **Reinforcement learning**, discussed next, also taps into this sequential aspect.

### Reinforcement Learning and Adaptive Systems

**Reinforcement learning (RL)** is a relatively novel approach in fraud detection, in which an agent learns to make decisions (e.g. whether to flag a transaction) through trial-and-error to maximize a reward (such as catching fraud while minimizing false alarms). While not as commonly used as other methods, researchers have begun to investigate RL for *real-time* fraud detection. For instance, an RL agent could dynamically adjust the thresholds for flagging transactions based on the current estimated fraud risk, or determine when to trigger additional verification steps for a suspicious transaction. The potential advantage of RL is its **adaptability**: it can, in theory, continuously learn from the environment and from adversaries’ responses. Early studies integrating deep RL (e.g. Deep Q-Networks) with fraud detection indicate that such models could optimize the trade-off between fraud catch rate and customer friction, outperforming static models in some cases (Zhao & Bai, 2022). However, RL approaches require careful formulation of reward functions (to capture business costs of fraud vs. false positives) and very large amounts of interaction data to train. They also raise the complexity of deployment in real financial systems. As of the latest literature, RL is seen as an **emerging frontier** – a promising direction for future research rather than a mature, widely adopted solution. The literature suggests that more exploration of RL and other advanced learning paradigms (like active learning, transfer learning, and incremental/online learning) is needed to keep up with adaptive fraudsters (Ali et al., 2022; Hernandez Aros et al., 2024).

### Comparative Strengths and Weaknesses

In summary, supervised learning remains the cornerstone of fraud detection due to its high accuracy when ample labeled data is available. Ensemble tree models and neural networks tend to achieve the best performance in comparative studies (Alarfaj et al., 2022), though no single algorithm dominates universally. Unsupervised methods play a crucial role in detecting new schemes and in scenarios of limited labels, but they often require domain insight to interpret their outputs. Deep learning methods excel at capturing complex patterns (e.g. sequential behaviors, network structures) at the cost of higher data requirements and reduced interpretability. A clear trend is the **combination of methods** – for example, using unsupervised anomaly detectors to generate features for supervised models, or ensembling multiple model types – to leverage their complementary strengths (Al-Hashedi & Magalingam, 2021; Ali et al., 2022). *Table 1* (below) provides a high-level comparison of the main approach categories in terms of their advantages and limitations for fraud detection:

**Table 1. Comparison of ML Approach Categories for Fraud Detection**

| Approach | Strengths | Weaknesses |
| --- | --- | --- |
| **Supervised ML** | High accuracy with sufficient labeled data; well-studied algorithms (LR, DT, SVM, RF, etc.) (West & Bhattacharya, 2016); straightforward evaluation using labels. | Requires labeled fraud data (scarce and often biased); may not detect novel fraud types; retraining needed to handle concept drift (Ali et al., 2022). |
| **Unsupervised ML** | No need for labels – can find novel or emerging anomalies (Ahmed et al., 2016); useful for exploratory analysis; can process massive unlabeled datasets. | Higher false positive rates if not tuned; results can be hard to interpret; not all anomalies are fraud (requires validation by analysts). |
| **Ensembles** | Improved accuracy and robustness by combining models (Alarfaj et al., 2022); can handle imbalance better (e.g. boosting focuses on difficult cases); often top performance in competitions. | Less interpretable (especially boosting); longer training and inference times; risk of overfitting if not properly regularized. |
| **Deep Learning** | Captures complex non-linear patterns; can learn feature representations automatically; well-suited for sequential data (RNNs) and relational data (GNNs) (Zhao & Bai, 2022). | Requires large datasets and computational power; typically "black-box" models (limited explainability); risk of overfitting, and tuning is complex. |
| **Reinforcement Learning** | Adapts to changing environment or adversary strategies; optimizes decisions over long-run outcomes (sequential decision making). | Very data-intensive (needs many interactions); difficult to design appropriate reward signals; not widely tested in production for fraud yet; can be unstable to train. |

No single method is a silver bullet; effective fraud detection often involves a **hybrid strategy** – for example, combining business rules and anomaly detectors to catch immediate red flags, with supervised ML models to provide a risk score, and then periodically retraining models as new fraud patterns emerge. As fraudsters adapt, the ability to update and combine models becomes crucial (Zhao & Bai, 2022; Ali et al., 2022).

## Datasets and Benchmarks in the Domain

Research in online fraud detection relies on a variety of datasets, ranging from public benchmarks to private industry data. Obtaining quality fraud datasets is a known challenge: real transaction data is often proprietary and sensitive, and fraud cases are rare events, so any public data tend to be highly imbalanced. Nonetheless, several datasets and benchmarks have become common in the literature:

* **Credit Card Transaction Data:** The most famous benchmark is the **Credit Card Fraud Detection** dataset originally made public by researchers at ULB (often accessed via Kaggle). It contains 284,807 credit card transactions with only 492 frauds (~0.17% fraud rate), and features are 30 numeric values which are principal components (PCA-transformed) of the original inputs, plus the transaction amount and time. This dataset is extremely imbalanced and has become a de facto standard testbed for fraud detection models. Indeed, it has been noted as “one of the most important” datasets in many comparative studies. Models are typically evaluated on this data using metrics like AUC and precision/recall, making it a common benchmark in academic papers. Another related dataset is the **IEEE-CIS Fraud Detection** competition dataset (from 2019), which includes e-commerce payment transactions with labeled frauds and additional identity features. The IEEE-CIS data (also popular on Kaggle) has on the order of 1-3% fraud rates and hundreds of features (some anonymized) capturing detailed transaction and user information. It is widely used in machine learning competitions, though less so in academic literature due to licensing restrictions.
* **Mobile Payments Simulation (PaySim):** **PaySim** is a synthetic dataset generated from real mobile money transaction logs to simulate mobile payment fraud. A commonly used version (on Kaggle) contains ~6.3 million simulated transactions with about 8,213 frauds (~0.13%). Each record includes features such as transaction type (e.g. cash-in, cash-out, transfer), amount, origin and destination account IDs, balance updates, and a fraud label. PaySim has been used in numerous studies as a proxy for mobile money fraud data. Its advantage is scale and realistic behavior patterns; however, being synthetic, it may not capture all real-world complexities.
* **BankSim (Financial Payments Simulation):** **BankSim** is another simulator-generated dataset (by López-Rojas *et al.*, 2016) based on a sample of point-of-sale bank transactions in Spain. It comprises roughly 594,643 transactions with around 1.2% labeled as fraud. Features include customer and merchant IDs, transaction amounts, customer age, location, etc., reflecting a realistic mix of transaction types. BankSim and PaySim are often used to evaluate fraud algorithms, especially when real transaction data is unavailable or to augment limited real data.
* **Financial Statement Fraud Data:** For **corporate fraud** detection (identifying companies that issue fraudulent financial statements), specialized datasets exist. The **China Stock Market & Accounting Research (CSMAR)** dataset includes financial reports of Chinese listed companies, with labels indicating which reports were later found fraudulent (e.g., overstating earnings). One published compilation from CSMAR contains 35,574 samples (company-year financial statements) with 337 labeled fraud cases. Another example is a dataset derived from Compustat for U.S. firms, used to detect accounting fraud: one study compiled data on 228 companies’ financial metrics, where half had fraudulent financial reporting. These datasets typically contain features like financial ratios, revenues, expenses, and other accounting indicators. Researchers use them to evaluate ML models on fraud detection in an accounting context, which differs from transaction fraud but shares issues of class imbalance and feature selection.
* **Insurance and Loan Fraud Data:** A known example is the **Insurance Company (CoIL 2000) Challenge** dataset, which contains customer data (9,822 instances, 86 attributes) from a Dutch insurance company; it has been used for tasks like predicting fraudulent insurance claims. While not exclusively labeled for fraud, creative use of such data (e.g., labeling known bad claims) has appeared in the literature. Similarly, credit scoring datasets (like the UCI “Default of Credit Card Clients” dataset of 30,000 records) are sometimes repurposed to study fraud-related behaviors (for example, treating loan default or credit default as an analog of fraudulent behavior or financial risk).
* **Cryptocurrency Fraud Data:** With the rise of cryptocurrencies, there are datasets focusing on crypto transactions and scams. One example is a Bitcoin transaction network dataset (from Kaggle) that includes metadata on Bitcoin addresses and transactions, with ~30,000 instances labeled as fraud or malicious behavior (such as transactions associated with Ponzi schemes or money laundering). Graph-based fraud detection techniques are often evaluated on such network data, where the challenge is to find illicit activity within a web of transactions.

In addition to these public datasets, many studies rely on **proprietary industry datasets**, such as a bank’s internal credit card transaction logs or an e-commerce platform’s purchase and account data. While results on such private data are often reported (sometimes involving millions of transactions and extremely low fraud rates), the datasets themselves are not publicly accessible. This has hampered direct benchmarking across studies, since two research teams might be testing their models on very different data. Researchers have called for more publicly available, up-to-date fraud datasets to enable consistent benchmarks and reproducible research (Hernandez Aros et al., 2024). The community has also noted that many widely used benchmark datasets (like the ULB credit card data from 2013) are somewhat dated or simplified, potentially not reflecting newer fraud patterns (Hernandez Aros et al., 2024). There is a recognized need for more diverse and current datasets covering various fraud types.

Data imbalance is a common trait across all these benchmarks – genuine transactions vastly outnumber fraudulent ones (often by 100:1 or 1000:1). This heavily influences model training and evaluation, and has led to specific techniques (discussed later) to handle imbalance. Moreover, *distribution shift* or concept drift over time is another issue: a model trained on last year’s fraud data may falter on this year’s data as fraud tactics evolve. Unfortunately, most public datasets are static snapshots and do not capture temporal evolution, so researchers sometimes resort to splitting data chronologically to simulate future tests, or using synthetic drift scenarios to evaluate adaptability.

Finally, it is worth noting the role of **competitions and shared challenges** (such as Kaggle competitions or the IEEE-CIS Fraud Detection challenge). These have driven progress by providing larger unified datasets and rallying practitioners to develop better models. Winning solutions often employ ensemble models and intensive feature engineering, which then inform academic research. However, competition setups may not fully reflect real-time deployment constraints (they typically allow training on past data with all labels known, whereas in production one deals with streaming data and unknown future fraud patterns).

In summary, a handful of well-known datasets (Credit Card, PaySim, CSMAR, etc.) serve as benchmarks in the literature, but there is a push for new datasets that reflect modern fraud scenarios. Research is increasingly venturing beyond traditional credit card fraud into areas like online banking fraud, mobile payments fraud, and cross-institution fraud, though data availability in these areas remains a challenge.

## Feature Engineering for Fraud Detection

Feature engineering is often cited as one of the most critical steps in building effective fraud detection models (Ti et al., 2022; Ali et al., 2022). Since raw transaction logs can be sparse or high-dimensional (especially for categorical variables like merchant or account IDs), deriving informative features that capture patterns of fraudulent behavior is key to improving model performance (Ti et al., 2022). In fact, many advances in fraud detection accuracy are driven not just by better algorithms, but by *richer features* that give the models more predictive signal.

Typical features used in online transaction fraud detection can be grouped into a few major categories:

* **Transactional Features (Monetary):** Basic attributes of each transaction, such as the transaction amount, currency, and transaction type. These are always included, but by themselves often have limited power (fraudulent amounts can range widely). More informative are aggregate features, like the average, minimum, and maximum transaction amount for a customer over various time windows, or relative amount features (e.g., this transaction’s amount compared to the customer’s historical average). These monetary-based features help identify if a transaction is abnormally large or small for a given account.
* **Temporal Features (Recency/Frequency):** Features capturing time-based patterns, often referred to as **RFM features** (Recency, Frequency, Monetary) in the fraud literature. **Recency** features might include the time since the last transaction on the account, or time since account opening or last password change. **Frequency** features include the number of transactions in the past hour/day/week, the count of failed login or payment attempts, etc. Fraudsters often make multiple rapid transactions in a short period (burst behavior) or strike at unusual times (e.g., in the middle of the night or right after an account becomes active). RFM features have been widely used; for example, a common strategy is to flag accounts with very high transaction counts in short spans or long dormant periods followed by sudden activity. One prior analysis grouped most transaction features into these R, F, M categories, reflecting that many fraud signals relate to how recently and frequently money is moving, and in what amounts.
* **Anomaly/Deviation Features:** Beyond simple aggregates, researchers derive features that reflect deviations or anomalies in behavior. For instance, one can compute the difference between the current transaction amount and the average amount for that merchant (to see if this purchase is unusually large relative to normal behavior at that store), or include a binary feature indicating if the current transaction is the largest the user has made in the past year. Other examples are *outlier scores* from unsupervised models (e.g., the reconstruction error from an autoencoder, or distance to nearest neighbors) which can then be fed as input features into a supervised model (if a transaction looks highly deviant compared to normal activity, that information helps the classifier). Some studies create features from the output of clustering algorithms or PCA components that summarize how “normal” or “abnormal” a transaction appears. However, generic anomaly-based features alone have often performed poorly relative to other feature categories, likely because they are too general and can flag many benign outliers.
* **Identity and Geographic Features:** These include static or semi-static attributes of the entities and context – e.g., the user’s age, location, IP address, device identifier or fingerprint, past fraud history flag, merchant category, etc. While not present in all datasets (some public sets are anonymized), when available these features can be crucial. For example, if a transaction comes from a device or IP address never seen before for that user, it might be deemed higher risk (many fraud schemes involve new device identities). Likewise, transactions involving high-risk countries or merchant categories known for fraud (e.g., online electronics, gambling) can be given higher risk scores.
* **Network Features:** By linking transactions through shared entities (card numbers, email addresses, devices, geolocation, etc.), one can derive graph/network metrics as features. For instance, a feature like “the number of unique credit cards that used the same device in the past 24 hours” could indicate a device that is being used by many different customers (potentially a sign of a fraud ring or mule device). Graph analytics can produce features such as the degree (number of connections) of a node (e.g., how many accounts are linked to an IP address), clustering coefficient of a node in the transaction network, or PageRank score of an entity. Such *network features* have been shown to help identify collusive fraud or money laundering clusters (Zhao & Bai, 2022). Incorporating relational features often significantly boosts the detection of organized fraud beyond what transaction-level features alone can achieve.
* **Behavioral and Domain-Specific Features:** These incorporate domain knowledge or known fraud patterns. For instance, in online banking fraud, one useful feature might be “the number of failed login attempts in the last 24 hours” or an indicator for whether the transaction is happening from an unusual location/device for the user. In credit card fraud, *velocity* features like “total spending in the last 5 minutes” or “number of distinct merchants used in the last hour” are used to catch rapid-fire fraud bursts. There are documented cases where features like the use of certain channels in combination with withdrawal amounts can signal fraud; for example, fraudsters often max out ATM withdrawals using the fast-cash option, so a feature capturing whether a withdrawal left the account balance near zero could indicate such behavior. Domain-specific features like these, often inspired by anti-money laundering (AML) red-flag rules or known fraud patterns, can dramatically improve detection when added to models (these were precisely the kind of expert-driven features that helped identify new fraud cases in some studies).

Researchers have compiled features from prior work to guide new projects. **Ti et al. (2022)** provide a useful categorization: they observed that most features in the literature fall into Recency, Frequency, Monetary (RFM) groups or a generic anomaly/deviation category, and they noted that these common statistical features alone *“do not fully describe users’ historical transaction records nor the relationship between legal and fraudulent behaviors”* (Ti et al., 2022). In their work, Ti et al. introduced additional features based on financial domain knowledge (e.g., patterns common in known money-laundering cases) and even features characterizing a user’s normal account behavior to better distinguish it from fraudulent behavior. Such features included counts of certain transaction types, usage of online vs. in-person channels, and indicators of unusual withdrawal patterns, crafted with the help of bank fraud experts (Ti et al., 2022). The result was an improved detection performance over using only traditional features, demonstrating the value of expert-driven feature engineering (Ti et al., 2022).

Feature selection is another important aspect: not all engineered features will be useful, and some may be redundant or noisy. Techniques like mutual information ranking, decision tree feature importance analysis, or stepwise selection are often used to filter features. A common finding is that a relatively small subset of features often accounts for most of a model’s predictive power – typically a mix of a few RFM features with some identity or network features. For example, Ti et al. (2022) found that their top engineered features (as ranked by feature importance or information gain) overlapped significantly with those identified as most important in earlier studies, suggesting certain key features consistently provide the bulk of detection capability.

It is also worth noting the interplay between feature engineering and the choice of ML model. Deep learning approaches can automatically create new feature representations (e.g., learned embeddings for categorical variables, or latent features via an autoencoder). For instance, recent works use **word embedding** techniques (Word2Vec, Doc2Vec, even BERT) on textual data (like transaction descriptions or IP addresses encoded as “sentences”) to generate features for fraud models (Hernandez Aros et al., 2024). These approaches blur the line between feature engineering and modeling. Nonetheless, even deep models benefit from sensible input representations (for example, feeding an LSTM a sequence of time-sorted transactions with engineered summary attributes, rather than raw unprocessed data). In practice, combining human-designed features with automatically learned features often yields the best results.

In summary, feature engineering for fraud detection typically involves creating a rich set of domain-driven features that capture transaction patterns over time, across entities, and relative to historical norms. Many studies credit feature engineering as a decisive factor in performance gains (Ti et al., 2022; Ali et al., 2022). As fraud tactics evolve, continuous updating of feature sets (for example, devising new proxies for emerging fraud behaviors) is needed. Moreover, incorporating unstructured data (like customer emails, chat logs, or call center notes) as features is a growing area – though it requires natural language processing techniques, it could uncover red flags that purely numeric features might miss (Hernandez Aros et al., 2024).

## Model Evaluation and Metrics

Evaluating fraud detection models is challenging due to the extreme class imbalance and the differing costs of errors. In this domain, a false negative (missed fraud) can lead to direct financial loss, whereas a false positive (flagging a legitimate transaction as fraud) causes inconvenience to customers and potential revenue loss if transactions are declined. Therefore, model evaluation must consider more than just overall accuracy.

Common evaluation metrics used in the literature include:

* **Accuracy:** the overall fraction of transactions correctly classified. Accuracy can be misleading in fraud detection because a trivial model that labels *everything* as non-fraud might achieve 99.9% accuracy if frauds are only 0.1% of the data. Thus, accuracy is usually reported alongside more sensitive metrics, but not relied upon solely in this field.
* **Precision (Positive Predictive Value):** among transactions predicted as fraudulent, how many are truly fraud. Precision reflects the *false alarm rate* – low precision means many false positives. In highly imbalanced settings, precision is crucial since a model that casts a wide net might catch most fraud (high recall) but at the expense of flagging too many legitimate transactions. A precision of, say, 10% means 90% of alerts are false alarms, which may be untenable for an operations team.
* **Recall (Sensitivity or Detection Rate):** among actual fraudulent transactions, how many the model correctly identifies. This is vital because missing fraud directly correlates to financial loss. Most banks and payment processors aim to maximize recall (catch as much fraud as possible) up to a certain precision or false-positive rate limit. High recall with low precision, however, can overwhelm investigators or annoy customers by falsely declining transactions.
* **F1-Score:** the harmonic mean of precision and recall (F1 = 2 · precision · recall / (precision + recall)). F1 provides a single measure that balances the two; it’s often used to compare models when there is class imbalance. A model with F1 closer to 1 is generally better in balancing fraud catch with false alarms.
* **Specificity:** the true negative rate (how many legitimate transactions are correctly left un-flagged). In fraud detection, specificity is usually extremely high for any reasonable model because the vast majority of transactions are non-fraud. Therefore, specificity is less informative by itself – even a poor model might achieve >99% specificity. It is sometimes reported for completeness, but the focus is typically on the minority class performance (fraud detection).
* **ROC-AUC (Area Under the ROC Curve):** The ROC curve plots the true positive rate (recall) against the false positive rate as the classification threshold is varied. The AUC summarizes the model’s ability to rank frauds higher than non-frauds, independent of a specific threshold. A perfect model yields AUC = 1.0 and a random model 0.5. Many published works report AUC values to demonstrate improvements (e.g., “Model X achieved ROC-AUC of 0.98 on the credit card fraud dataset”). However, with extreme imbalance, improvements in parts of the ROC curve that correspond to high false-positive rates (which might be irrelevant in practice) can inflate AUC, so it must be interpreted carefully.
* **Precision-Recall Curve and PR-AUC:** The Precision-Recall curve focuses on the minority class performance, plotting precision vs recall. The area under this curve (PR-AUC) is often more indicative than ROC-AUC when the positive class is very rare, as it zooms in on how well the model identifies positives without being overwhelmed by the numerous negatives. Researchers have increasingly reported PR-AUC, especially if they care about operating in high-recall regimes. For instance, one recent study evaluated their fraud model using both ROC-AUC and PR-AUC, and presented precision/recall values at certain operating points to give a fuller picture of performance (Baghdadi et al., 2024).
* **Confusion Matrix & Derived Rates:** It is common to present the confusion matrix (the counts of true positives, true negatives, false positives, false negatives) or derived metrics like False Positive Rate (FPR) and False Negative Rate (miss rate). In fraud detection, a false negative (missed fraud) is often considered more costly than a false positive, but both are important. Some papers specifically mention the *false alarm rate* (which is the FPR among legitimate transactions) and the *detection rate* (recall among frauds). A model with a low FPR at a given high recall is highly desirable.

Given the imbalanced nature of the problem, evaluation must account for the base rate. This sometimes means using resampling or setting specific decision thresholds. For example, a model might be trained to output a fraud risk score; then the threshold is tuned to achieve a particular target (like 90% recall) and the corresponding precision is observed. This threshold tuning is often guided by business constraints (how many alerts can be handled, how much loss is acceptable). Cost-based metrics are also employed in some research: assigning a monetary cost to a missed fraud (e.g., the fraud amount lost) and a cost to a false alert (e.g., operational review cost or customer dissatisfaction), then aiming to minimize total cost. Cost-sensitive learning and evaluation align closely with business goals but require quantifying those costs accurately (Ali et al., 2022).

Cross-validation must be used carefully in fraud datasets to avoid overfitting. Time-based splits are recommended for evaluation (training on past data, testing on a future period) to mimic real deployment. Also, due to class imbalance, stratified sampling or ensuring each fold has some fraud instances is important when using cross-validation.

Finally, evaluation of **unsupervised** methods poses a unique challenge: if no labels are used in training, one still needs labels (or some ground truth) to assess detection performance. Many papers that propose unsupervised fraud detection will simulate fraud anomalies or inject known fraud patterns into a dataset to test if the method can detect them, or they run the unsupervised method on real data and then manually check a sample of the flagged cases. In academic literature, unsupervised techniques are often still evaluated against a labeled test set (treating it like a binary classification evaluation), even though the model itself didn’t use labels to train. Clustering methods might be evaluated by cluster purity or by how well clusters align with fraud vs. non-fraud if labels are available for evaluation.

**Summary of metrics usage:** A systematic review by Ali et al. (2022) noted that the most common performance measures in fraud detection studies were precision, recall (sensitivity), F1-score, and AUC, along with accuracy and occasionally specificity. No single metric is sufficient; authors often report a suite of metrics to provide a complete picture. Importantly, because of the precision-recall tradeoff, it’s recommended to compare models at similar recall levels or to use F1/PR-AUC to ensure a fair comparison across models.

## Challenges and Limitations in Applying ML to Fraud Detection

Despite substantial progress, there remain numerous challenges when deploying machine learning for fraud detection in online financial systems. Key limitations and active problem areas include:

**Class Imbalance:** As emphasized earlier, fraud datasets are extremely imbalanced, with fraud cases often <1% of all transactions (which is typical in credit card fraud, for example). This imbalance can bias ML models to favor the majority class (non-fraud) and ignore the minority class. A model might achieve very high overall accuracy simply by predicting “no fraud” most of the time. Techniques to address this include data-level methods like oversampling the minority class (SMOTE is one of the most commonly used oversampling methods in fraud research) or under-sampling the majority (less common in practice due to the loss of information) (Ali et al., 2022). Algorithm-level approaches include using class weight adjustments (penalizing fraud misclassification more than non-fraud) and specialized algorithms like one-class classifiers or anomaly detectors. Class imbalance also motivates ensemble strategies (e.g., combining detectors focused on the rare class). Despite these measures, the imbalance problem is not fully “solved” – high false negative rates can persist if models are not carefully tuned, and oversampling can lead to overfitting synthetic patterns. Research suggests exploring more advanced resampling and augmentation techniques, as well as better evaluation frameworks for imbalanced data (Ali et al., 2022).

**Evolving Fraud Patterns (Concept Drift):** Fraudsters continuously adapt their strategies to evade detection, leading to *concept drift* – the statistical properties of fraud data change over time (Zhao & Bai, 2022). A model that was effective last year may gradually perform worse as new fraud tactics emerge (for instance, once EMV chip cards reduced card-present fraud, criminals shifted to online card-not-present fraud). Handling drift requires models that can adapt. This may involve retraining periodically with recent data, using online learning algorithms, or designing features that capture new behavior patterns. However, retraining has operational overhead and there is a gap in performance between retraining events. Some proposed solutions are incremental learning (continuously updating the model as new labeled data comes in) and transfer learning (transferring a model trained in one context to a new but related context). Yet, these approaches are not widely adopted in practice for fraud, partly due to complexity and the risk of catastrophic forgetting. Future research is expected to focus on drift, perhaps using techniques from data stream mining or dedicated drift detection algorithms to signal when a model update is needed (Ali et al., 2022; Hernandez Aros et al., 2024).

**Explainability and Transparency:** Financial institutions operate in regulated environments, and when ML models flag a transaction or account as fraudulent, they often need to explain or justify that decision – to auditors, regulators, or customers. Many effective models (ensemble trees, neural networks) are *black boxes*, making it hard to explain why a transaction was scored as fraud. This lack of transparency can hinder trust in the system and complicate compliance with regulations (for example, the EU’s GDPR includes a “right to explanation” for algorithmic decisions). As a result, **Explainable AI (XAI)** techniques are gaining attention in fraud detection. Methods like SHAP and LIME are being used to provide feature-attribution explanations for model outputs (e.g., highlighting which features contributed most to a particular fraud score). For instance, a bank might generate an explanation such as “Transaction flagged due to an unusually high amount and a new device location.” XAI not only helps human analysts verify and trust the alerts, but also ensures that the model is not inadvertently biased or discriminatory (e.g., not unfairly targeting certain demographic groups) (Rakowski et al., 2021). Transparency and explainability are essential to ensure fairness – as noted by Rakowski et al. (2021), the growing use of AI in finance raises ethical questions about the status of humans and potential biases in automated decisions. A related concern is **ethical AI**: ensuring the model’s decisions do not reflect biases in historical data (for example, if past fraud investigations disproportionately scrutinized a certain region or customer profile, a model might carry that bias forward). Thus, explainability is linked to detecting and correcting such biases. While XAI tools are helpful, they add computational overhead and complexity. There is ongoing research into inherently interpretable models for fraud (e.g., hybrid rule/ML systems, or models that naturally output human-readable reasons). In practice, some institutions use a two-tier system: a complex model to score transactions, followed by an interpretable set of rules or an explanation module to justify each positive prediction.

**Privacy and Data Sharing Constraints:** Fraud detection could be greatly improved by sharing data across institutions (since fraudsters often attack multiple banks or merchants), but privacy regulations and competitive concerns limit data pooling. Financial data is highly sensitive, and transferring customer transaction data to a centralized repository for modeling may violate privacy laws or customer agreements. **Federated learning (FL)** has emerged as a promising solution: it allows multiple institutions to collaboratively train a global model without sharing raw data (Aljunaid et al., 2025). In FL, each institution trains the model on its local data and only model updates (gradients) are shared and aggregated. This approach preserves privacy while still benefiting from broader fraud patterns learned across organizations. For example, an FL system could let many banks jointly train a fraud model that has “seen” fraud examples from all participants, without any bank ever seeing another’s customer data (Aljunaid et al., 2025). Early research (as of 2025) demonstrates that FL can maintain model performance close to a pooled-data scenario, offering a way forward for industry-wide fraud defenses (Aljunaid et al., 2025). However, FL comes with its own challenges: ensuring all participants’ data are fairly represented, preventing a malicious participant from infering others’ data from the model updates, and handling variability in data quality across institutions. Another privacy-preserving approach is using fully *synthetic data* generation to share patterns without real data – though synthetic data may not capture all the nuances of real fraud. (PaySim, discussed earlier, is essentially an example of releasing a simulator instead of raw logs.) Overall, privacy constraints mean most models today are trained in silos, which is a limitation since fraudsters exploit gaps between those silos (e.g., moving to a different bank if one catches them).

**Scalability and Real-Time Detection:** Fraud detection systems often need to operate in real-time (or near real-time), scoring transactions as they occur (e.g., within milliseconds for a credit card swipe at a point-of-sale). This imposes engineering constraints: the ML model must be efficient in terms of latency and throughput. Complex ensemble or deep models might be too slow for real-time use unless optimized. Some banks still rely on simpler models or a two-stage filtering approach (apply quick business rules first to narrow down candidates, then an ML model on the remaining subset) to meet strict time requirements. Moreover, the volume of data can be huge – major payment processors handle *millions* of transactions daily, so models must also scale for batch retraining and storage of features. While this is more of a systems engineering issue, it influences model selection: a slightly less accurate but much faster model might be chosen for deployment if it dramatically reduces latency. Researchers are increasingly aware of this; for instance, one 2024 study explicitly considered computational cost and online response time in proposing their fraud detection architecture, aiming to balance predictive power with speed (Baghdadi et al., 2024). The challenge of real-time fraud detection is not the focus of most academic papers (which typically assume offline evaluation on static datasets), but for practical adoption it is critical. This includes handling data streams, updating models on the fly as new data arrives, and avoiding latency spikes that could interfere with user transactions.

**False Positives and Operational Load:** Even with decent precision, the absolute number of false positives can be large due to the base rate fallacy. For example, if the fraud rate is 0.1% and a model has 90% recall and 90% precision – which is excellent performance – out of 1 million transactions, around 900 will be fraud (of which 810 are caught) but also ~900 false alerts will be raised. This means 900 customers impacted falsely (transactions incorrectly flagged or stopped), which might still be too high for a business to handle every day. False positives carry significant operational costs: each alert might require manual review, a customer phone call, or some intervention. Therefore, many institutions layer their models with post-processing rules or case-based reasoning to further reduce false positives. It’s a challenging balance between catching fraud and not "crying wolf" too often. In practice, classifier threshold tuning and combining ML with expert rules (to override obvious false alarms) are common approaches. Research-wise, this issue is less discussed, but some works incorporate cost-sensitive training or optimize metrics like Fβ (which weight precision vs recall) depending on business priorities.

**Adversarial Attacks on ML Models:** A growing concern is that sophisticated fraudsters might attempt to *probe* and evade machine learning models, effectively launching adversarial attacks. For instance, if they can deduce that transactions just below a certain dollar amount don’t trigger the model, they may adjust their fraud amounts accordingly. Adversarial machine learning – where inputs are intentionally manipulated to fool the model – is a relevant threat in fraud detection. There have been theoretical studies on how a fraudster could manipulate transaction features to slip past a classifier (analogous to adversaries generating images that fool image classifiers). While such systematic probing is not widely documented in real banking systems (it’s difficult for attackers to get feedback on why transactions are declined), it remains a looming challenge. Building models that are robust to such manipulation (through adversarial training or by detecting anomalous patterns indicative of an attack) could become an important direction, especially as fraudsters become more tech-savvy.

In summary, ML-based fraud detection must contend with **data-related challenges** (imbalance, drift, limited labels, privacy constraints) and **model-related challenges** (need for interpretability, real-time operation, adversary-aware robustness). These factors mean that pure ML is not a "set-and-forget" solution – it requires continuous monitoring, updating, and integration with domain knowledge. Many practical systems adopt a hybrid approach: ML models combined with rule-based filters, human expert review for edge cases, and ongoing model governance (to recalibrate for drift and check for bias). The literature reflects this as well, as more recent papers emphasize the deployment context (like real-time constraints and human-AI interaction) in addition to just classifier performance (Baghdadi et al., 2024).

## Tools, Libraries, and Implementation Frameworks

Implementing fraud detection models involves a mix of data engineering, machine learning libraries, and sometimes big data or streaming frameworks. Over the years, a set of common tools and platforms have emerged in both research and industry:

* **Programming Languages:** *Python* is the dominant language in recent fraud detection research, thanks to its rich ecosystem of ML libraries and tools. Researchers frequently use Python notebooks (often on platforms like Jupyter or Kaggle) to experiment with models using scikit-learn, TensorFlow/PyTorch, etc. *R* is also used in some academic works, especially for statistical approaches or where interpretability is valued (with packages like randomForest or caret). In older literature, MATLAB or Java (e.g., Weka) were sometimes used, but these have largely been supplanted by Python/R in the last decade.
* **Machine Learning Libraries:** For classical ML algorithms, **scikit-learn** (Python) is a go-to library, offering efficient implementations of logistic regression, SVM, decision trees, ensemble methods, and more. Many comparative studies rely on scikit-learn for a consistent implementation of algorithms to benchmark (Ali et al., 2022). In R, packages like randomForest, e1071 (for SVM), or glmnet (for logistic regression) are common. **XGBoost** has its own library (Python, R, C++), which became popular for its efficiency and high performance in Kaggle competitions. **LightGBM** and **CatBoost** are other gradient boosting libraries occasionally seen in recent work.
* **Deep Learning Frameworks:** **TensorFlow** (with Keras) and **PyTorch** are widely used for developing neural network models for fraud detection. For example, a study building an LSTM on transaction sequences might use TensorFlow/Keras, while one implementing a custom Graph Neural Network might prefer PyTorch (often with libraries like PyTorch Geometric). These frameworks provide GPU acceleration, which is useful since training on millions of transactions can be time-consuming otherwise.
* **Anomaly Detection Toolkits:** There are libraries specialized for anomaly detection that have been applied to fraud. For instance, **PyOD** (Python Outlier Detection) is a library that implements a variety of detection algorithms (LOF, Isolation Forest, one-class SVM, autoencoders, etc.) and has been used by practitioners for quick anomaly detection experiments. In R, packages like tsoutliers or oddstream might be used for time-series anomaly detection relevant to fraud.
* **Big Data Processing:** In industry or large-scale experiments, tools like **Apache Spark** (with MLlib) or **Flink** have been used to scale fraud detection pipelines. Spark’s MLlib offers distributed implementations of machine learning algorithms, and companies often use Spark or similar big-data platforms to handle streaming transaction data for real-time scoring. For example, a bank might have a Spark Streaming pipeline that applies a pre-trained model to each transaction as it flows in, ensuring low-latency responses even under high volume. Some academic papers addressing scalability report using Spark to handle datasets with tens of millions of transactions, especially for feature extraction and data preparation.
* **Rule Engines and Hybrid Systems:** Many practical solutions combine ML models with rule engines (like Drools or proprietary rule systems). While not an “ML library” per se, these rule engines often work alongside ML models to enforce business constraints or catch certain edge cases that the model might miss. Researchers sometimes mention using a set of expert rules as a baseline and then layering an ML model on top (this hybrid approach can yield better performance than either alone). The rule engine can also serve to ensure interpretability for critical conditions (e.g., always flag transactions from a sanctioned country, regardless of model score).
* **Model Deployment and Monitoring:** Tools for model deployment (such as TensorFlow Serving, ONNX runtime, or cloud services like AWS SageMaker, Google Cloud AI Platform, etc.) are relevant when taking a fraud model from the lab to production. While academic papers don’t typically detail deployment, a thesis or industry report might note that these frameworks allow integration of trained models into transaction processing systems (e.g., wrapping the model in a REST API that the payment system calls). Additionally, monitoring tools (like custom dashboards or alerting systems) are used to track model performance in production – essential to detect when retraining is needed due to drift or when a model starts generating too many false positives.

In academic research contexts, much of the focus is on the modeling, so authors will often mention using scikit-learn or Keras for their experiments, sometimes with hyperparameter tuning libraries (like Optuna or scikit-learn’s GridSearchCV). Reproducibility has improved with more researchers sharing code on GitHub or Kaggle for their fraud models. For instance, many **Kaggle competition winners** share their notebooks (often using Python with scikit-learn/XGBoost and sometimes deep learning models), which then serve as reference implementations for others.

Another aspect is specialized **fraud detection platforms or services**. Companies like SAS, FICO, or IBM offer commercial fraud detection solutions (e.g., SAS Fraud Management, FICO Falcon) which incorporate ML techniques under the hood. These are often proprietary “black-box” systems to end-users but set benchmarks for what is used in industry. Open-source alternatives or research prototypes often attempt to replicate or improve on those using the aforementioned libraries and frameworks.

In summary, a practitioner building an ML-based fraud detection system today is likely to use Python as the main programming language; scikit-learn for baseline models and experimentation; XGBoost/LightGBM for powerful ensemble models; PyTorch or TensorFlow for any deep learning components; and possibly Spark or a cloud pipeline for handling real-world data volumes and streaming. The tooling ecosystem for fraud detection is mature and largely overlaps with general data science and big data tools, with some unique additions (like simulation tools for generating synthetic fraud data, or graph databases for linking entities and deriving network features).

## Research Gaps and Future Directions

Despite extensive research, there are several areas in fraud detection that warrant further investigation. Current literature has identified multiple gaps and emerging trends that point to how future work can enhance the effectiveness of ML in fraud detection:

* **More Focus on Un- and Semi-Supervised Methods:** As noted, most studies to date have emphasized supervised learning on known fraud scenarios. However, a gap exists in detecting previously **unseen fraud patterns**. Future research is encouraged to explore advanced unsupervised techniques, such as deep anomaly detectors (e.g., combining autoencoders with additional criteria) and hybrid models that can handle the one-class or limited-label setting more robustly (Ali et al., 2022). Some authors suggest that methods popular in other domains – for example, active learning to selectively label new data, or self-supervised representation learning to exploit unlabeled data – have not yet been fully utilized in fraud detection. Closing this gap could improve detection of novel frauds without waiting for large labeled samples to accumulate.
* **Adaptive and Online Learning:** Handling concept drift remains an open issue. Future systems may need to incorporate online learning algorithms that update continuously as new transactions and fraud labels come in. Research into drift detection techniques specifically tailored to financial transaction data could help trigger model updates at the right time (Zhao & Bai, 2022). Additionally, reinforcement learning approaches (or generally sequential decision models) are still in their infancy in this field; exploring these could yield models that better adapt to adversaries over time. For example, an RL agent that adjusts its strategy as it observes fraudster behavior could potentially outmaneuver static models. This area intersects with game-theoretic perspectives on fraud (treating detection as a game between defender and attacker), which is an interesting theoretical lens.
* **Integration of Diverse Data Sources:** A promising direction is to bring in data beyond the immediate transaction features. Studies have pointed out that incorporating unstructured data – such as text from payment descriptions, customer emails or chat logs, IP address WHOIS information, or even voice call data – could significantly enhance fraud detection but is currently under-explored (Hernandez Aros et al., 2024). For instance, fraud investigators often look at email correspondence or device/browser metadata; an ML model could do the same if fed those inputs via NLP or graph features. Some recent works have used news articles or financial disclosures to predict fraud or defaults (text mining for fraud cues in financial statements), which could be extended to transaction fraud contexts. Moreover, network and relationship data (e.g., knowing if multiple customers share a device or address) might help catch collusion fraud. The gap here is often the difficulty of acquiring or fusing these data sources, but future research might employ multi-modal learning to combine them effectively.
* **Real-World Deployment and Case Studies:** There is a relative lack of published case studies on the *long-term deployment* of fraud ML systems. Many research papers test models on static datasets, but do not report on what happens when the model is deployed in production (and fraudsters potentially react to it). Sharing such experiences, even qualitatively, could inform better model design – for example, how often models need updating in practice, or what types of fraud slip through a deployed system and why. Future literature could focus on **MLOps** for fraud detection, i.e., best practices in monitoring model performance over time, handling model updates with minimal service disruption, and integrating human analysts’ feedback into the model improvement loop. This would bridge the gap between research prototypes and effective production systems.
* **Explainability and Fairness Research:** While XAI methods are being applied post-hoc to explain black-box models, a gap remains in designing fraud detection models that are interpretable by design and ensuring they are fair (not inadvertently biased against certain groups). Future research might explore **interpretable ML** algorithms in fraud (e.g., rule-based learners, generalized additive models with pairwise interactions) that can approach black-box performance. Additionally, systematic studies on the *fairness* of fraud models – ensuring that fraud detection doesn’t, say, unfairly target transactions from specific regions or demographics without justification – are still sparse. As AI ethics and fairness become more prominent in all domains, we expect more work auditing fraud detection models for bias and developing techniques to mitigate any unfairness (Rakowski et al., 2021).
* **Creating Better Public Datasets and Benchmarks:** As identified in surveys (Hernandez Aros et al., 2024), many fraud types (e.g., internal fraud, procurement fraud, mortgage fraud) are under-represented in research simply due to lack of available data. One future direction is **collaborative dataset creation**: financial institutions and researchers could work together (with privacy protections) to release anonymized, representative datasets for these fraud domains. A suggestion in a 2024 review was to create a freely accessible, up-to-date fraud dataset with modern fraud scenarios for the community (Hernandez Aros et al., 2024). This would spur research in neglected areas and allow benchmarking of new methods. Additionally, future benchmarks could focus on streaming/temporal evaluation (with time-ordered data to test adaptability) rather than the standard random train-test splits that ignore temporal drift.
* **Advanced Algorithms from Other Fields:** Techniques like graph neural networks, which have shown state-of-the-art results in detecting fraud rings, still have room for development in terms of scalability and ease of use. Also, **transformer-based models**, which revolutionized NLP, are starting to be applied to sequential transaction data (treating each transaction as a “token” in a sequence) – future work could assess if transformers outperform RNNs for modeling long transaction sequences. Some emerging research looks at **meta-learning** (models that can quickly adapt to new fraud patterns with very few examples) and **federated learning** across institutions (as discussed earlier) to leverage knowledge without sharing data. Incorporating these advanced ML concepts could address current limitations around data scarcity and evolving patterns.
* **Holistic Fraud Prevention Systems:** Moving beyond just detection, future directions include how ML can assist in **fraud prevention and response**. This involves not only flagging a transaction, but deciding what action to take (block it, hold for review, ask the customer for two-factor authentication, etc.). Reinforcement learning or adaptive decision strategies could play a role here in optimizing such actions to minimize loss and customer friction. Another aspect is **link analysis for investigations** – once a fraud alert is raised, ML could help trace related transactions or accounts (clustering alerts into likely cases, identifying connected entities automatically). Some papers hint at this (using graph connectivity to group alerts), but it’s not a well-studied area academically.

In conclusion, while machine learning has greatly enhanced our ability to detect online financial fraud, the arms race between fraudsters and detection systems continues. Research gaps highlight the need for adaptability, data sharing, and explainability. The incorporation of new data types and learning paradigms, the emphasis on ethical and fair AI, and closer industry-academic cooperation (for data and deployment knowledge) are all important for the next generation of fraud detection systems. As one recent survey noted, *“the need for robust and updated approaches to face new fraud modalities is particularly highlighted”* (Hernandez Aros et al., 2024). The coming years will likely see fraud detection evolve into an even more interdisciplinary field, combining ML with cybersecurity, network science, and domain expertise in finance to stay ahead of increasingly sophisticated fraud schemes.

## References (Harvard Style)

Abdallah, A., Maarof, M.A. & Zainal, A. (2016). Fraud detection system: a survey. *Journal of Network and Computer Applications*, 68, 90–113.

Achakzai, M.A.K. & Juan, P. (2022). Using machine learning meta-classifiers to detect financial frauds. *Finance Research Letters*, 48, 102915.

Ahmed, M., Mahmood, A.N. & Islam, M.R. (2016). A survey of anomaly detection techniques in the financial domain. *Future Generation Computer Systems*, 55, 278–288.

Al-Hashedi, K.G. & Magalingam, P. (2021). Financial fraud detection applying data mining techniques: a comprehensive review from 2009 to 2019. *Computer Science Review*, 40, 100402.

Ali, A., Abd Razak, S., Othman, S.H., Eisa, T.A.E., Al-Dhaqm, A., Nasser, M. *et al.* (2022). Financial fraud detection based on machine learning: a systematic literature review. *Applied Sciences*, 12(19), 9637.

Aljunaid, S.K., Almheiri, S.J., Dawood, H. & Khan, M.A. (2025). Secure and Transparent Banking: Explainable AI-Driven Federated Learning Model for Financial Fraud Detection. *Journal of Risk and Financial Management*, 18(4), 179.

Alarfaj, F.K., Malik, I., Khan, H.U., Almusallam, N., Ramzan, M. & Ahmed, M. (2022). Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms. *IEEE Access*, 10, 39700–39715.

Alwadain, A., Ali, R.F. & Muneer, A. (2023). Estimating financial fraud through transaction-level features and machine learning. *Mathematics*, 11(5), 1184.

Baghdadi, N., Rachdi, M. & Ouanan, M. (2024). Energy-based deep learning architecture for credit card fraud detection. *Preprint* (unpublished manuscript).

Hernandez Aros, L., Bustamante Molano, L.X., Gutierrez-Portela, F., Moreno Hernandez, J.J. & Rodríguez Barrero, M.S. (2024). Financial fraud detection through the application of machine learning techniques: a literature review. *Humanities and Social Sciences Communications*, 11, Article 1130.

KPMG (2022). *KPMG Fraud Outlook 2022*. KPMG International.

Nicholls, C.M., Arning, A. & Ong, S.H. (2021). Financial services fraud – trends and emerging threats. *Journal of Financial Crime*, 28(4), 1114–1129.

Phua, C., Lee, V., Smith, K. & Gayler, R. (2010). A comprehensive survey of data mining-based fraud detection research. *arXiv preprint*, arXiv:1009.6119.

PricewaterhouseCoopers (PwC) (2022). *Global Economic Crime and Fraud Survey 2022*. PwC Report.

Rakowski, R., Polak, P. & Kowalikova, P. (2021). Ethical aspects of the impact of AI: the status of humans in the era of artificial intelligence. *Society*, 58(3), 196–203.

Ryman-Tubb, N.F., Krause, P. & Garn, W. (2018). How Artificial Intelligence and machine learning research impacts payment card fraud detection: a survey and industry benchmark. *Engineering Applications of Artificial Intelligence*, 76, 130–157.

Ti, Y.W., Hsin, Y.Y., Dai, T.S., Huang, M.C. & Liu, L.C. (2022). Feature generation and contribution comparison for electronic fraud detection. *Scientific Reports*, 12, Article 18042.

West, J. & Bhattacharya, M. (2016). Intelligent financial fraud detection: a comprehensive review. *Computers & Security*, 57, 47–66.

Zhao, Z. & Bai, H. (2022). Financial fraud detection using machine learning: fundamental issues and key challenges. *IEEE Transactions on Knowledge and Data Engineering*, 34(5), 2391–2402.