Data Band

2025 FRAUD DETECTION

COM-480

Prepared By
Abdul Karim Mouakeh
Nadine Al Fadel Raad
Jean-Daniel Rouveyrol

website
EPFL

Overview:

Fraudulent transactions are an issue that all financial institutions face. As fraudsters improve their ways of committing various crimes, a static system often falls short in detecting these complex or subtle anomalies. This is where the use of visual analytics becomes powerful; it enables analysts to interactively explore the different patterns of criminals' suspicious behaviors at a larger scale.

Our motivation came from the need to make the exploration of fraud related data more intuitive through data storytelling. By transforming dense transaction data into interactive visual insights, we aimed to uncover behavioral patterns and trends over time or across individuals to better help analysts understand the context of these suspicious activities.

Key Questions

- Do fraud rates vary significantly in different time frames?
- Are certain types of transactions (e.g., POS, ATM, Online) more prone to fraud?
- How do user behaviors, such as failed transactions or high transaction volumes, correlate with fraud?
- Are there geographic patterns in fraudulent activity?
- Can we identify clusters or similarities among users linked to suspicious behavior?
- What is the relationship between risk scores and fraud probability?

We're Investigating

Time Patterns

Transaction Types

Geographic Trends

User Behavior

Risk Scores

Our Process:

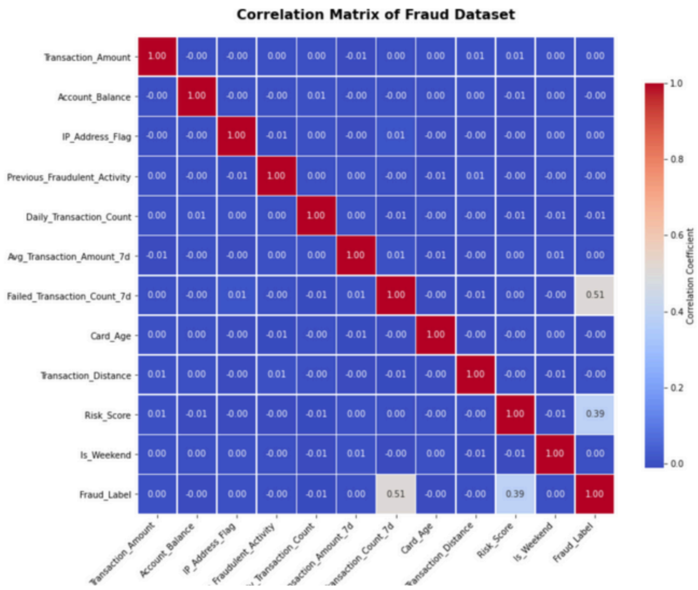
📁 Raw Dataset 📊 EDA

❓ Key Questions 📈 Visualization Design

Timeline:

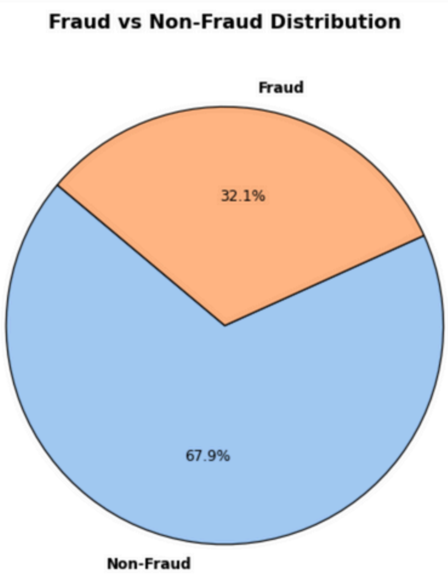
- Week 1–2: EDA & Question Formation
- Week 3: Sketches & MVP Planning
- Week 4–5: Implementation
- Week 6+: Finalization & Visual Tuning

Dataset Summary



We worked with a Kaggle dataset containing 50,000 financial transactions labeled as either fraudulent (1) or legitimate (0). Each transaction included detailed metadata, such as type, amount, device, timestamp, risk score, and behavioral signals like failed transactions or volume over time.

Preprocessing



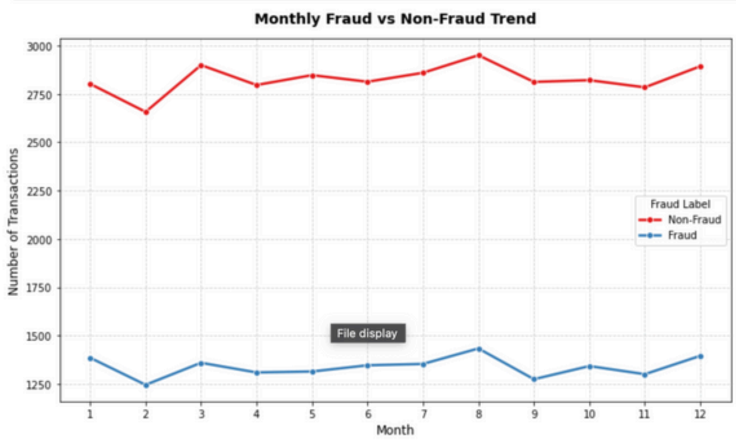
The dataset was largely clean, with no missing values. We performed light preprocessing, including:

- Converting timestamps to datetime objects for temporal analysis
- Parsing and encoding categorical features
- Normalizing key numerical values (e.g., risk scores, transaction amounts)

EDA:

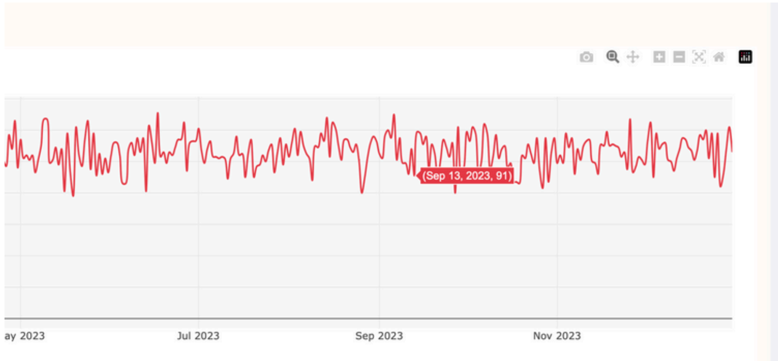
Initial EDA revealed that:

- Failed transactions (7-day count) had a moderate correlation with fraud (r = 0.51)
- Risk score also showed a meaningful relationship (r = 0.38)
- Transaction types and device usage varied notably (POS and Online were most common)
- Tokyo and Mumbai had the highest number of transactions



Insights to Interface

After completing the exploratory data analysis, we started turning our ideas into actual visual drafts. In addition to the hand drawn sketches, we created rough first pass visualizations directly in code to test our layout and direction.

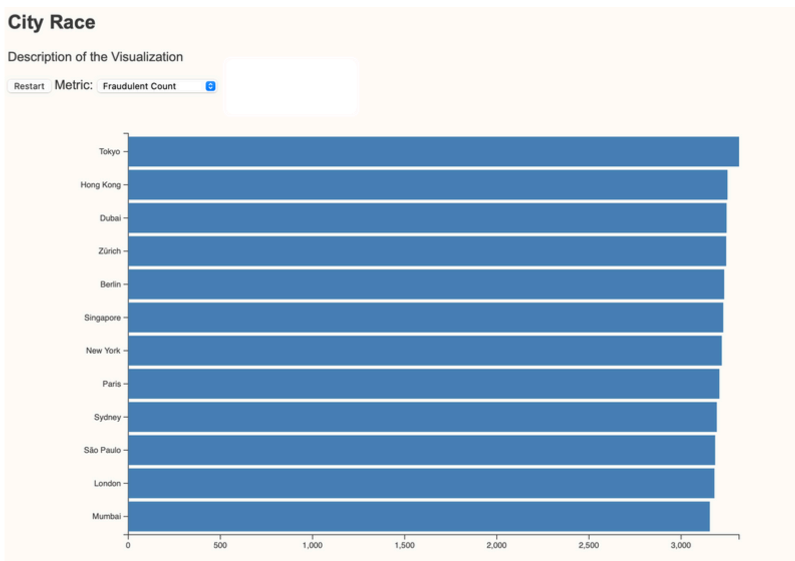


In Milestone 2, we built three core components:

- A simple line chart to track fraudulent transactions over time.
- A bubble based world map to display fraud distribution by city
- A static bar chart comparing fraud volume across major cities

At this stage, our visualizations were basic just enough to validate the key views we wanted to explore. We didn’t have interactivity, filters, or styling in place yet, but the structure was there.

These early drafts helped us decide what was working and what needed improvement. For example:



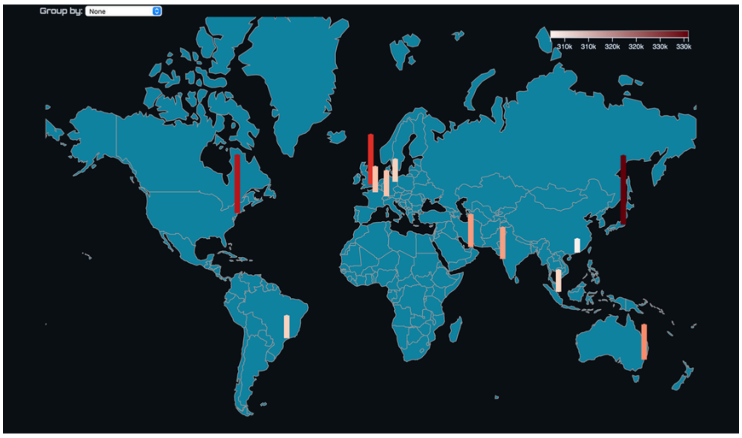
- We noticed the time series chart needed smoothing and more context (like a moving average)
- The choropleth map felt too flat, so we added grouped bar heights and a color scale in the final version
- The static bar chart was upgraded into a city race animation, which added a temporal dynamic and made it more engaging

Final Visualizations

Fraud Over Time:

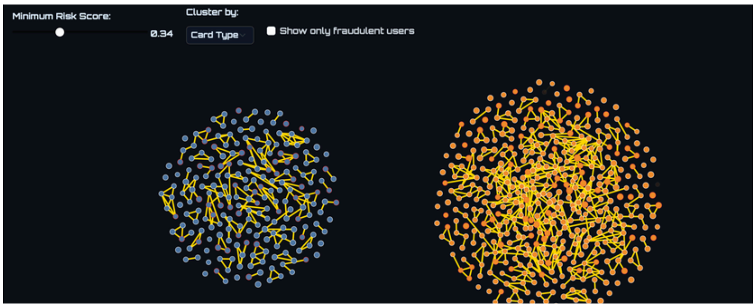
We built a time series chart to look at how fraud trends change throughout the year. You can filter the data by transaction type, like POS, bank transfer, or ATM withdrawal, and by city. There are also options to view raw values, smoothed averages, or scaled fraud rates. One of the best parts is that it clearly shows the peak fraud periods, so you can spot spikes, seasonal patterns, and any unexpected drops that might hint at changing tactics or external events.

Choropleth Map:



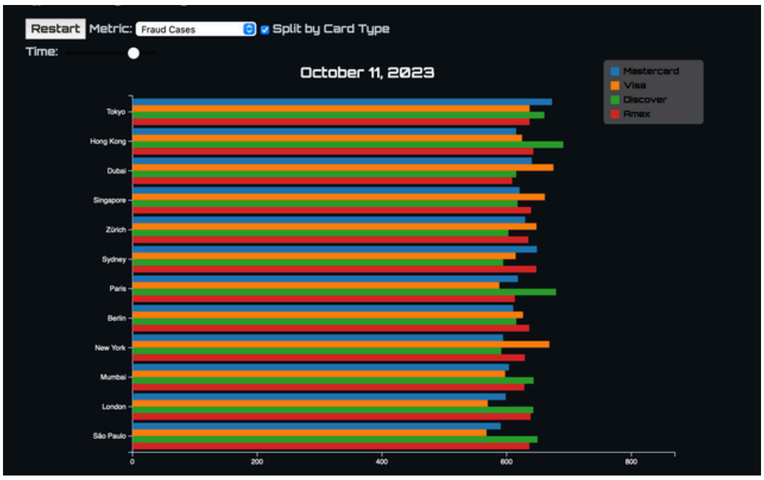
To explore where fraud happens the most, we built a choropleth map that colors each country and adds vertical bars on top. At first, we thought about going with circular markers, but they made it harder to compare and overlapped a lot, so we switched to bars, which made the map easier to read. You can group the data by card type, device type, or merchant category to dig into what's really driving fraud in different regions. It's a great way to spot patterns and connect them to user behavior across locations.

Network Graph:



We also created a force directed graph that links users based on how similar their behavior is like using the same device or having similar risk scores. You can filter by risk score or only show fraudulent users, and group them by things like card type or location. This chart helps highlight clusters of users that might be acting suspiciously together, stuff that's hard to catch in other charts.

City Race Chart:



This one's an animated bar chart that shows how the top cities change over time in terms of fraud volume or transaction value. It's engaging to watch cities rise and fall. We also added the option to break it down by card type, so you can understand what's going on behind the scenes.

How we built it:

To create the final visualizations, we mainly worked with D3.js along with some HTML/CSS to lay everything out. We chose a scroll-based layout to let the user go through the story in a more guided and interactive way.

All of the charts from the time series to the city race, choropleth map, and network graph were made from scratch using D3. Each one came with its own set of challenges. For example, building the time series chart required syncing axes and tooltips properly, while the city race chart had a lot of logic to handle transitions and updates over time. Here's a small example from that:

```
.attr("transform", d => `translate(0, ${yScale(d.name)})`)
.transition()
.duration(duration)
.attr("transform", d => `translate(0, ${yScale(d.name)})`)
```

That transition made the bars move smoothly to their new position each time the data updated.

The network graph was probably the trickiest to get right. We used D3's force simulation, and setting the right parameters took a lot of trial and error. For example:

```
forceSimulation(nodes)
  .force("link", forceLink(links).distance(100))
  .force("charge", forceManyBody().strength(-200))
  .force("center", forceCenter(width / 2, height / 2))
```

We also added interactive filters using dropdowns and sliders like the ones for transaction type and city. These let the user explore different aspects of the data without needing any technical background.

One thing we did, was that we added a few new cities to have a better pallet. To keep it fidel to the original data, the attributes where resampled from original cities and we gave a number of records for each new city in the range of the number of records from the original cities.

Challenges & Lessons Learned

Working on this project definitely taught us a lot and not just about D3

- D3 Learning Curve: At first, we really struggled with getting D3 to behave how we wanted. The documentation isn't always beginner friendly, so we had to rely a lot on examples and trial and error. Figuring out how to properly bind data, use scales, and handle transitions took time.

- Layout Troubles: Trying to make the dashboard scroll based while keeping everything aligned and responsive was more complicated than we expected. Sometimes elements would break or overlap between sections. It made us appreciate how tricky front end design can be.

- Network Graph Confusion: This one gave us the most trouble. Getting the simulation to behave well (no flying nodes) and figuring out the right parameters for distance, force, and link strength took a lot of tuning. We also had to write custom filters for clustering and highlighting specific users, which was tricky with our dataset.

- Data Simplification: We initially tried to include too many features, which slowed everything down and made the visualizations hard to read. Eventually, we had to simplify, reduce attributes, clean up values, and focus on what would actually provide insight.

- Time Management: Finally, managing our time was harder than expected. Balancing this with other deadlines, syncing tasks between teammates, and finalizing the story flow took a lot of late nights

Final Reflection

When we started designing our visualizations, we thought about how someone would naturally explore fraud patterns, not just jumping into one chart, but following a logical story. So we built the flow around four key questions: When? Where? Who? And How much?

We kicked things off with the Fraud Over Time chart because time gives immediate context, if fraud is seasonal, increasing, or randomly spiking, that's something analysts want to see right away. It also helps spot weird behavior that might link to events or external changes.

Next, we moved to the Choropleth Map, which breaks things down geographically. After seeing when fraud peaks, the next question is where it's happening the most. We made sure you could group by card type, merchant type, and other filters to get a deeper picture.

Once we had the when and where, we introduced the Network Graph, this shifted the focus to the who. We wanted to give users a tool to explore connections between suspicious users. Clustering based on behavior helped us uncover groups that might be acting together or just showing similar patterns, even across locations.

Then we closed the story with the City Race Chart, which visualizes how cities compare over time. It wraps things up by showing not just static fraud counts, but how cities rise and fall in fraud activity. We thought it was a nice way to bring everything together in a dynamic, easy to digest way.

Peer Assesment

Milestone 1

We split the work into three equal parts. Each of us proposed 2–3 interesting datasets and ideas, and after some discussion, we selected the one we're working on now. For the Exploratory Data Analysis (EDA), we each explored the dataset independently, and then combined our findings into one unified analysis. The README was also divided equally, with everyone writing a section.

Milestone 2

In the second milestone, we continued the same approach. Everyone took ownership of a section in the report and contributed their own visualization sketch idea. We each built a basic implementation of our sketch, and brought everything together in our prototype website.

Milestone 3

For the final milestone, we needed to be more strategic to avoid overlap and stay efficient. So, each one of us was responsible for implementing one of the core visualizations. We also divided the process book editing equally and made sure everyone's input was reflected. As for the final visualization, the network graph, we sat down and worked on it together as a team.

We also co-wrote the script for the screencast and intended for all of us to appear in the final video. Unfortunately, after several takes, we had to limit it to just two people due to technical and timing constraints, but the entire team was involved in scripting, reviewing, and filming the video.