Fraud Data Analysis

Austin Phillips & Ivo Schossow

Background info

- The rise of digital commerce has dramatically increased credit card use, and with it, the risk of fraud.
- Banks and financial institutions now rely heavily on data mining and machine-learning models to detect suspicious transactions. A major challenge arises with this: huge volumes of data, which makes training models slow and computationally expensive.
- Our project aimed to answer the following question: What is the ideal training-set size that maintains accuracy while reducing computation time?
- We used a 1-million-observation Kaggle dataset with clear variable names and an unusually high fraud rate (8%), allowing easier model testing despite it being simulated.

Methodology

- Data Prep: Use 1M-observation Kaggle dataset; split into 50% training and 50% testing.
- Sampling: Create training sample sizes of 100 -> 500 -> 2500 -> 10000 ->25000 -> 125000 -> 500000 (approx. multiplicative increases).
- Modeling: Train logistic regression on each sample subset.
- Evaluation: Compare models using AUC, Accuracy, Sensitivity, Precession, Specificity F1, and Run Time on the test set.
- Optimization: Identify the smallest sample size that performs “close enough” to the full dataset.
- Outcome: Determine ideal training size for efficient, accurate fraud detection.

Multicollinearity testing

Our first concern with logistic regression analysis was **multicollinearity**

- This occurs when predictor variables are linearly correlated

The two sets of variables that I believed **may be correlated** were:

- Distance from Home & Distance from Last Transaction.
- Used Chip & Used Pin.

Multicollinearity testing

Distance from Home & Distance from Last Transaction are both **Continuous** variables
To test a possible relationship, we used ***Pearson*** and ***Spearman*** correlation tests.

- Both tests indicate a **near-zero relationship** between the two variables

```
> cat(sprintf("Pearson  r = %.4f, p = %
lue))
Pearson  r = 0.0002, p = 8.47e-01
> cat(sprintf("Spearman ρ = %.4f, p = %
value))
Spearman ρ = -0.0011, p = 2.85e-01
```

Multicollinearity testing

Used Chip and Used PIN are both categorical variables

- We must use a different test - ***Cramér's V***

Once again, we received a result that shows **near-zero correlation**

```
> cramv <- assocstats(tab)$cramer
> cat(sprintf("Cramér's V = %.4f\n", cramv))
Cramér's V = 0.0014
>
```

Define Test Statistics

We have 6 test statistics that we will use to determine model effectiveness, all ranging from 0-1

AUC (Area Under the Curve)

- Select both a fraud and non-fraud observation. AUC is the probability that the model rates the true fraud as a higher likely to be fraud than the non-fraud

Accuracy

- This is the total correct, divided by the total tested
- Can be misleading, as a model that always selects selects no fraud will have 94% accuracy

Sensitivity

- The rate at which the model will accurately flag real fraud cases

Precision

- The percent of true fraud transaction, out of all that had been flagged

Specificity

- percent of non-fraud transactions that are correctly not-flagged

F1

- A harmonic mean of precision and specificity which tries to balance both

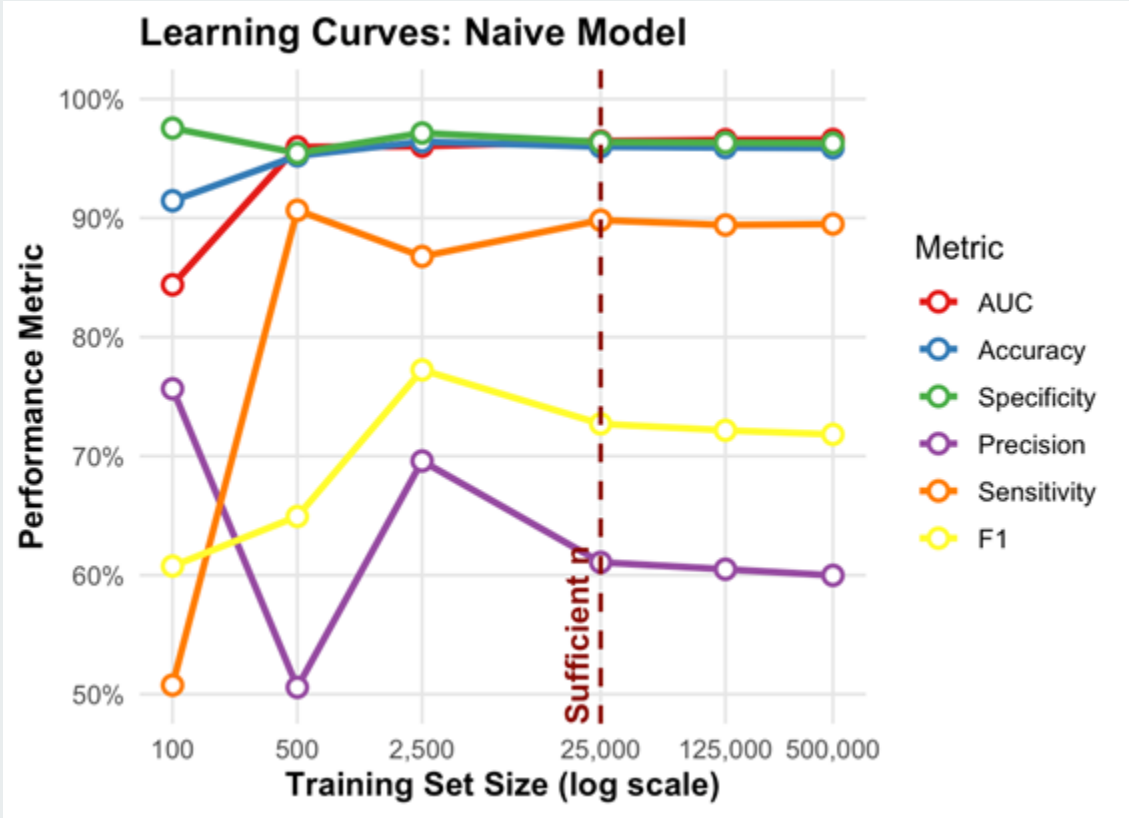
Naive Model

- This Naive Model will be used as our baseline estimate
- It is called the “Naive” model because we **do not validate any assumptions**, data has not been transformed in any way
- Note our **low precision** and **sensitivity**
  - 40% of flagged transactions are NOT Fraud
  - 10% of true fraud cases are not caught

	AUC	Accuracy	Sensitivity	Precision	Specificity	F1	TrainSize
1	0.8439142	0.914512	0.5076090	0.7565621	0.9755137	0.6075724	100
2	0.9596882	0.952206	0.9065202	0.5057618	0.9545499	0.6492801	500
3	0.9602953	0.964120	0.8677368	0.6958798	0.9713911	0.7723639	2500
4	0.9646288	0.959896	0.8981843	0.6107554	0.9637987	0.7270946	25000
5	0.9656874	0.959176	0.8940930	0.6049479	0.9632702	0.7216343	125000
6	0.9658568	0.958834	0.8949272	0.5998034	0.9628139	0.7182302	500000

Naive Model

- Our metrics seem to bound around a bit, but level out around a sample size 25,000
- Can we do better?



Let's Test Our Assumptions

- Log. Reg. has an assumption of a **linear relationship** between variables and the log-odds of the response
- ***Box-Cox Test*** tells us how to transform our variables

	Variable	Lambda
distance_from_home	distance_from_home	0
distance_from_last_transaction	distance_from_last_transaction	0
ratio_to_median_purchase_price	ratio_to_median_purchase_price	0

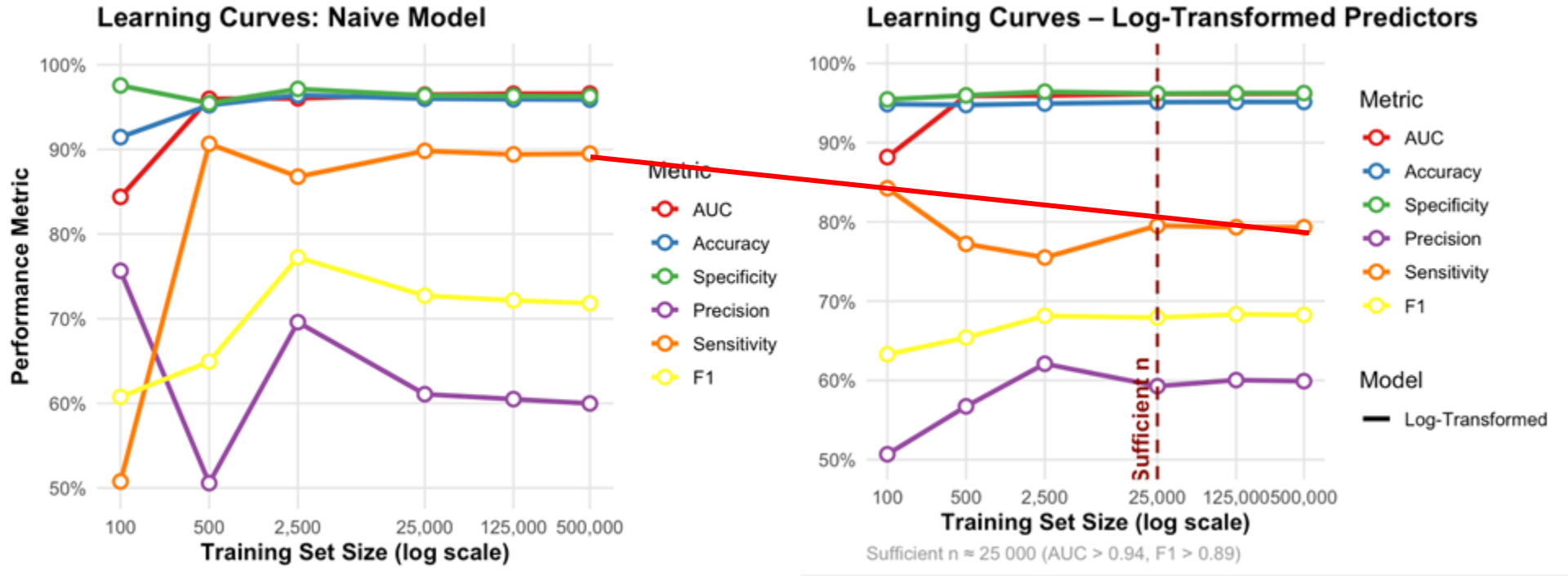
- Results: Our data **needs to be transformed** with a logistic transformation
  - Lambda = 0 indicates log(x+1) transformation

Log Transform

- After testing for this issue, we re-ran our logistic regression with the transformed variables

	AUC	Accuracy	Sensitivity	Precision	Specificity	F1	TrainSize
1	0.8817133	0.948562	0.8423853	0.5067679	0.9544596	0.6328322	100
2	0.9590656	0.947502	0.7721373	0.5672215	0.9595444	0.6540038	500
3	0.9594538	0.949216	0.7549761	0.6209530	0.9642737	0.6814372	2500
4	0.9613750	0.951024	0.7952328	0.5927154	0.9618893	0.6791993	25000
5	0.9615151	0.951354	0.7932212	0.6003750	0.9625656	0.6834550	125000
6	0.9616958	0.951264	0.7932058	0.5990031	0.9624431	0.6825595	500000

Log transform



Naive vs Log

Naive

	AUC	Accuracy	Sensitivity	Precision	Specificity	F1	TrainSize
1	0.8439142	0.914512	0.5076090	0.7565621	0.9755137	0.6075724	100
2	0.9596882	0.952206	0.9065202	0.5057618	0.9545499	0.6492801	500
3	0.9602953	0.964120	0.8677368	0.6958798	0.9713911	0.7723639	2500
4	0.9646288	0.959896	0.8981843	0.6107554	0.9637987	0.7270946	25000
5	0.9656874	0.959176	0.8940930	0.6049479	0.9632702	0.7216343	125000
6	0.9658568	0.958834	0.8949272	0.5998034	0.9628139	0.7182302	500000

Log

	AUC	Accuracy	Sensitivity	Precision	Specificity	F1	TrainSize
1	0.8817133	0.948562	0.8423853	0.5067679	0.9544596	0.6328322	100
2	0.9590656	0.947502	0.7721373	0.5672215	0.9595444	0.6540038	500
3	0.9594538	0.949216	0.7549761	0.6209530	0.9642737	0.6814372	2500
4	0.9613750	0.951024	0.7952328	0.5927154	0.9618893	0.6791993	25000
5	0.9615151	0.951354	0.7932212	0.6003750	0.9625656	0.6834550	125000
6	0.9616958	0.951264	0.7932058	0.5990031	0.9624431	0.6825595	500000

Why did our model get worse?

- Transforming the data didn't help us - even after the transformation there is still a non-linear relationship in the log-odds of our response variable
  - Tested using the ***Box-Tidwell*** test below
- There is no easy fix for this. We need a new tool!
- GAM (Generalized Additive Models) can track these **non-linear** relationships!

```
+ }
→ dist_home: p = 0.000000 → NON-LINEAR in log-odds (reject linearity)
→ dist_last: p = 0.000000 → NON-LINEAR in log-odds (reject linearity)
→ ratio: p = 0.000000 → NON-LINEAR in log-odds (reject linearity)
>
```

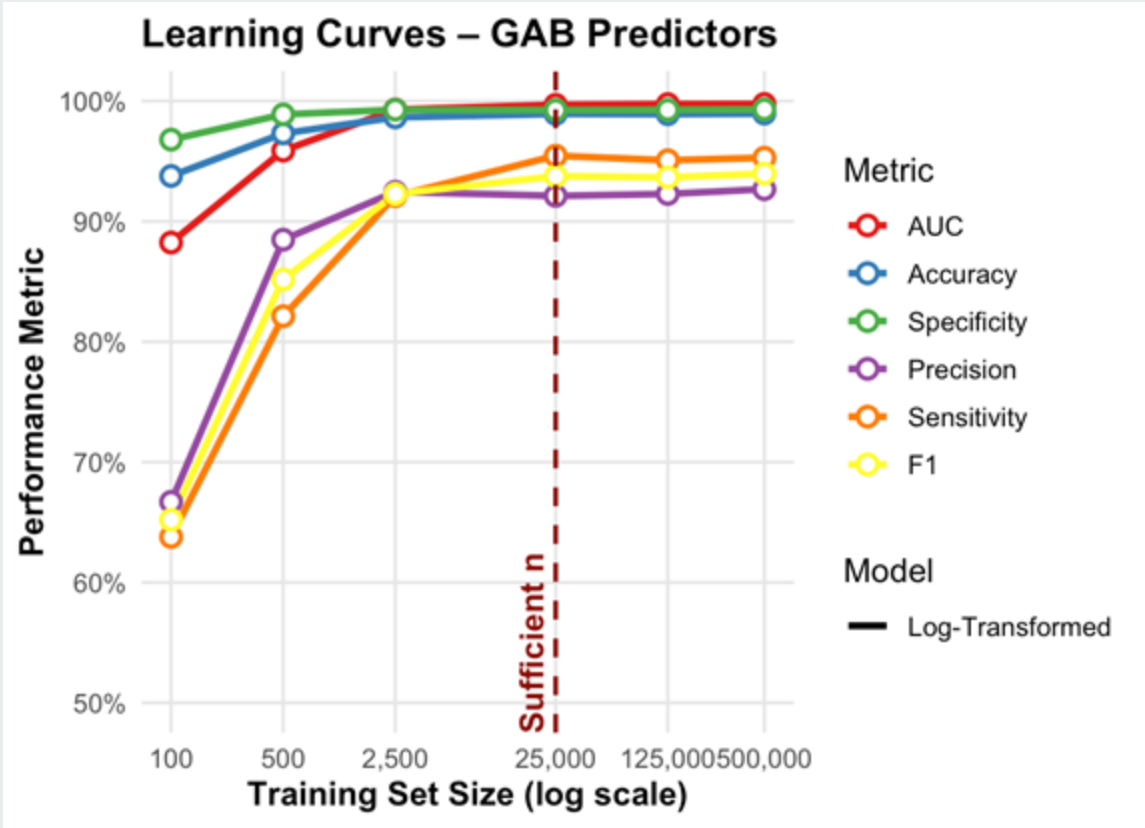
GAM Model

- Our GAM model is much better, even at smaller n

	AUC	Accuracy	Sensitivity	Precision	Specificity	F1	TrainSize
1	0.8824248	0.937742	0.6378266	0.6669791	0.9679372	0.6520772	100
2	0.9587862	0.973074	0.8212262	0.8847860	0.9888734	0.8518221	500
3	0.9929526	0.986468	0.9210323	0.9245701	0.9927669	0.9227978	2500
4	0.9968956	0.989272	0.9545799	0.9211862	0.9924704	0.9375858	25000
5	0.9974299	0.989068	0.9507868	0.9227867	0.9926194	0.9365776	125000
6	0.9975787	0.989588	0.9529105	0.9267651	0.9929984	0.9396560	500000

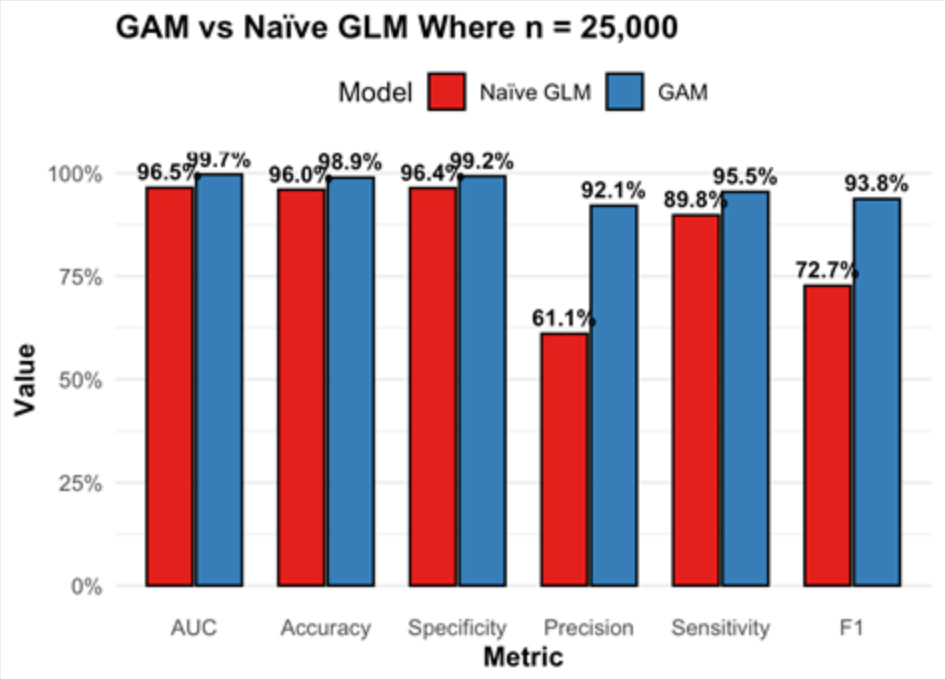
GAM Model

- All of our metrics seem to perform highly, **all above 90%**
- Even at a sample size of 2,500, metrics seem to do well (but don't level our until 25,000)



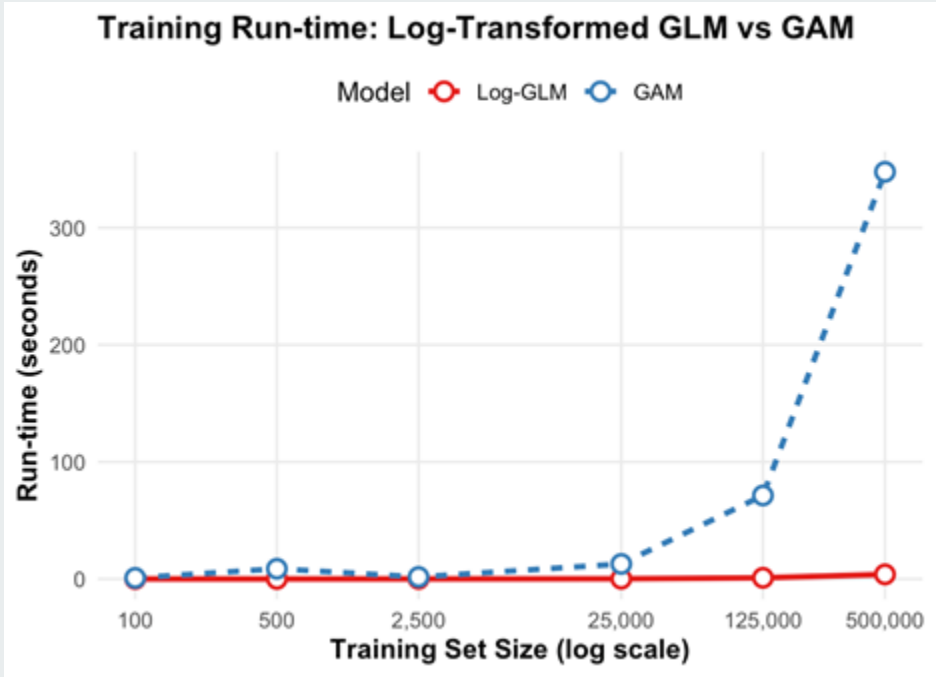
Gam vs Naive Model at n = 25,000

- When compared to the naive model, **all** of our metrics see a marginal increase
- Precision now at 92%, meaning **only 8% of flagged transactions are non-fraudulent**
- The chance of a true fraud cases not being flagged has been cut in half! (10.2% to 4.5%)



GAM vs Log. Reg. - Run Time

- Main issue with GAM is run-time
- Our Logistic Regression model took only 4 seconds to run, even at 500,000 observations.
- GAM model took over a minute for 125,000 samples and almost 6 minutes for 500,000 samples
- Good News: 25,000 seems to be sufficient, where the GAM model only took 12.8 seconds (still 64x longer than the log-reg model.



Conclusions & Future Research Directions

- It appears that both Log-Reg Modeling and GAMs have limitations where model accuracy converges, and adding more data does not have a benefit
- Logistic Regression models work to an extent - however are underpowered for true credit card fraud identifications
- Our idealized scenario may underrepresent the true amount of samples needed due to unrealistically high fraud rate - making GAM models even slower/resource intensive
- This preliminary research shows that GAM models can be useful in identifying fraud, while keeping error-rates marginally low.
- Further research should apply GAM to real fraud data - may require the use of high-end hardware

Potential Applications

- Credit card fraud detection: A more optimized and faster model would help banks flag fraudulent transactions in real time.
- Findings can guide how organizations handle extremely large datasets without unnecessary computational cost.
- Broader Optimization: Sampling strategies can be applied in healthcare analytics, retail forecasting, and any field that relies on large-scale predictive modeling.
- Baseline Algorithms: Results may validate when simpler models (like logistic regression) are sufficient before moving to more complex models (GAM/ Neural Networks)

Bibliography

Narayanan, D.R. (2022) *Credit Card Fraud*, *Kaggle*. Available at: https://www.kaggle.com/datasets/dhanushnarayananr/credit-card-fraud/data (Accessed: 30 September 2025).

John, G H, & Langley, P (1996). Static versus dynamic sampling for data mining.
Available at https://cdn.aaai.org/KDD/1996/KDD96-069.pdf

Y. Sahin and E. Duman, "Detecting credit card fraud by ANN and logistic regression," *2011 International Symposium on Innovations in Intelligent Systems and Applications*, Istanbul, Turkey, 2011, pp. 315-319, doi: 10.1109/INISTA.2011.5946108.
Available at https://ieeexplore.ieee.org/abstract/document/5946108

B. Teh, M. B. Islam, N. Kumar, M. K. Islam and U. Eaganathan, "Statistical and Spending Behavior based Fraud Detection of Card-based Payment System," 2018 International Conference on Electrical Engineering and Informatics (ICELTICs), Banda Aceh, Indonesia, 2018, pp. 78-83, doi: 10.1109/ICELTICS.2018.8548878.
Available at https://ieeexplore.ieee.org/abstract/document/8548878

Yufeng Kou, Chang-Tien Lu, S. Sirwongwattana and Yo-Ping Huang, "Survey of fraud detection techniques," IEEE International Conference on Networking, Sensing and Control, 2004, Taipei, Taiwan, 2004, pp. 749-754 Vol.2, doi: 10.1109/ICNSC.2004.1297040.
Available at https://ieeexplore.ieee.org/abstract/document/1297040