Drive Away Fraudsters With Driverless AI

Venkatesh Ramanathan

H2O World 2017, Mountain View, CA Dec 4-5, 2017

Agenda

Problem

Approach

Experiments

Conclusion

PROBLEM

Fraud Prevention @ PayPal

Robust feature engineering, machine learning and statistical models

Highly scalable and multi-layered infrastructure software

Superior team of data scientists, researchers, financial and intelligence analysts

Images source:

Collusion Fraud – An Example Scenario

Buyer purchases an item using PayPal

Seller ships item to buyer

Buyer finds box empty & asks PayPal for refund

PayPal can refund buyer (eligible transactions under Buyer Protection)

Images source:

Collusion Fraud – An Example Scenario

PayPal asks seller for proof

Seller gives proof

PayPal Repays seller (Seller Protection)

PayPal incur loss

Buyer and seller split the money..

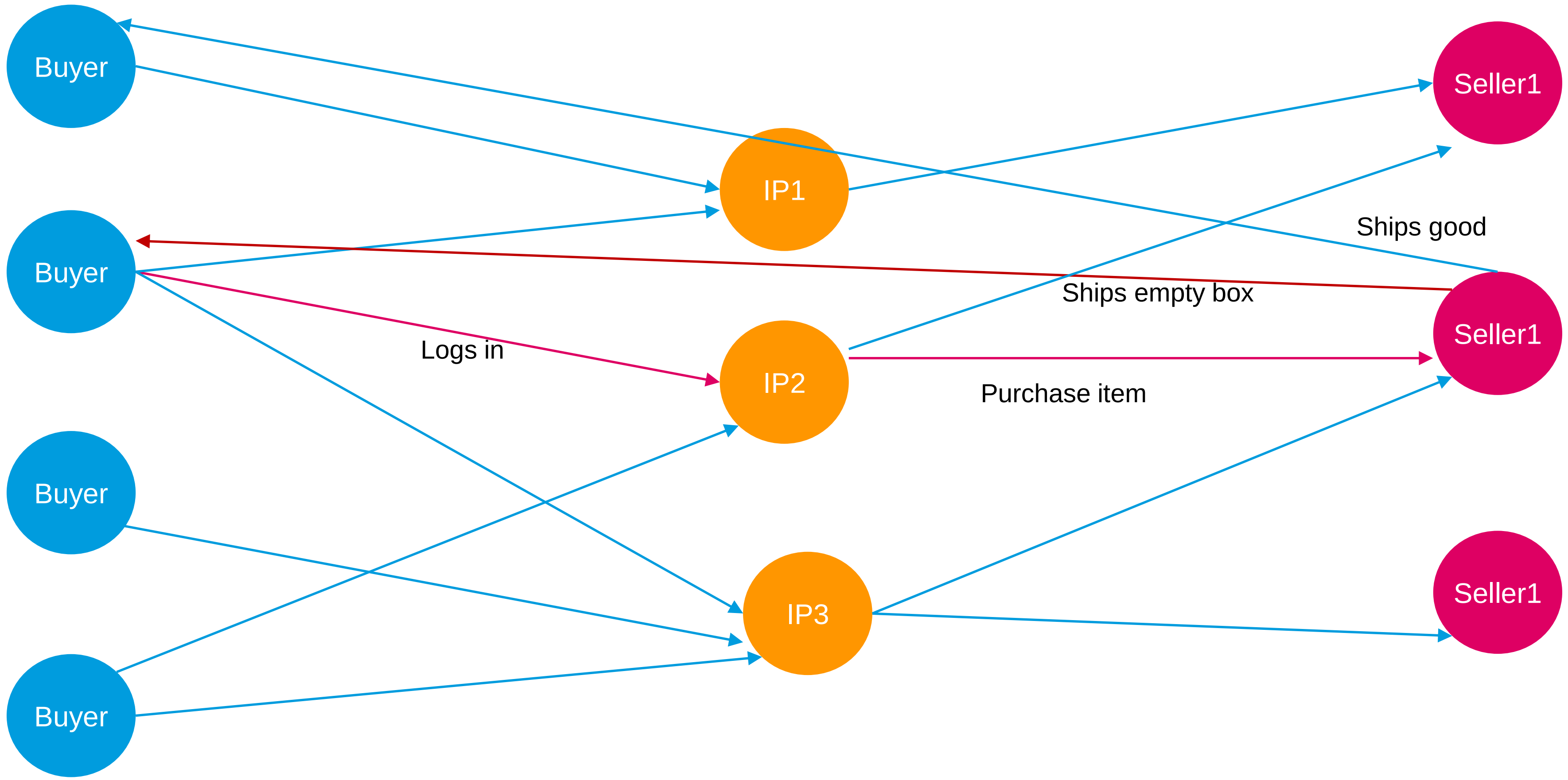
Collusion Fraud

Images source:

Collusion Fraud

- What if there are many such buyers colluding with many such sellers?
- What if such buyers & sellers stay below the radar? ($ Transaction < Threshold)?
- What if such sellers behave well with majority of the buyers & collude with a few?
- How do we detect if the buyers and sellers are legitimate or if they are colluding with each other?

Collusion Fraud

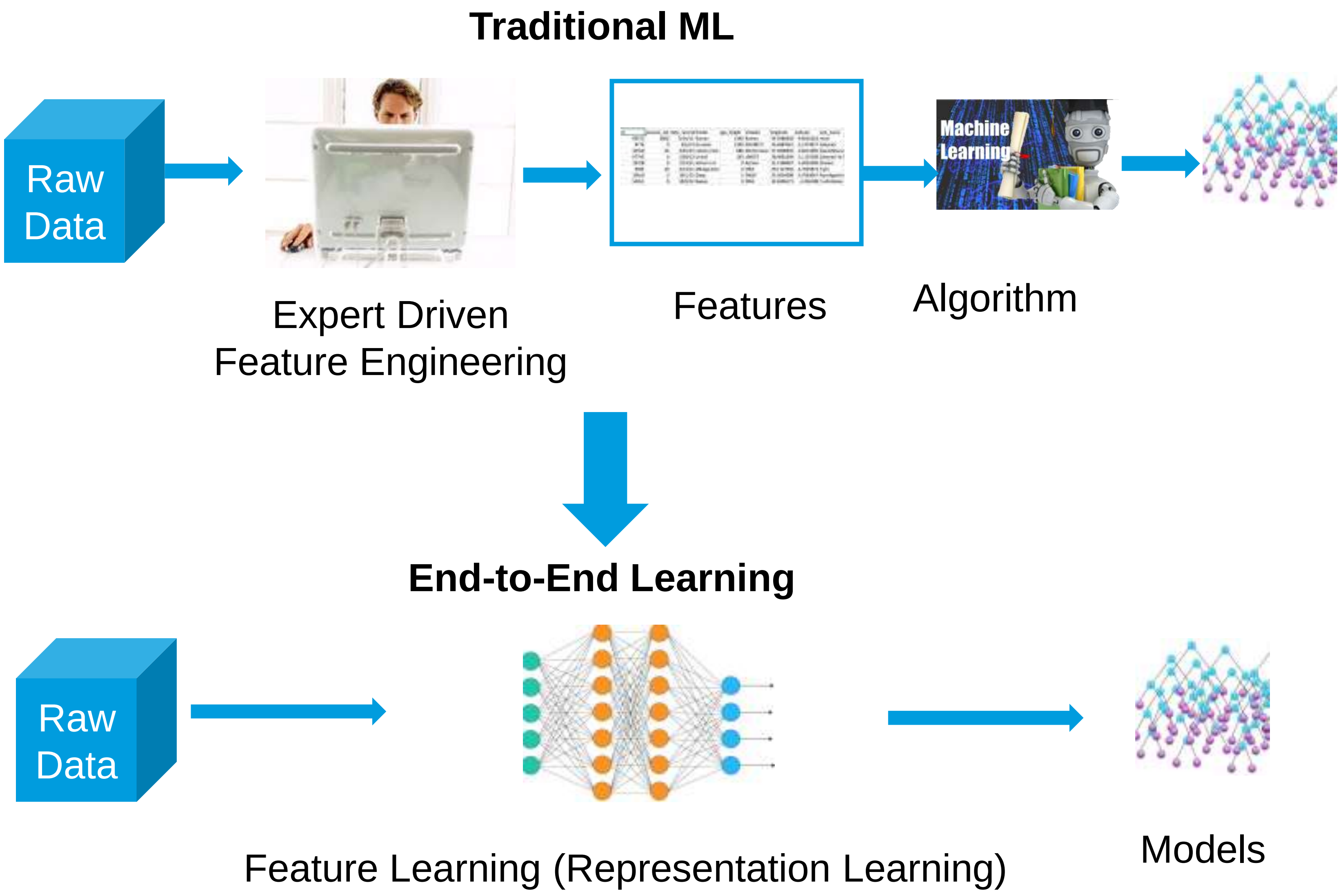


- **Can we exploit network structure of fraudsters to solve collusion fraud?**

APPROACH

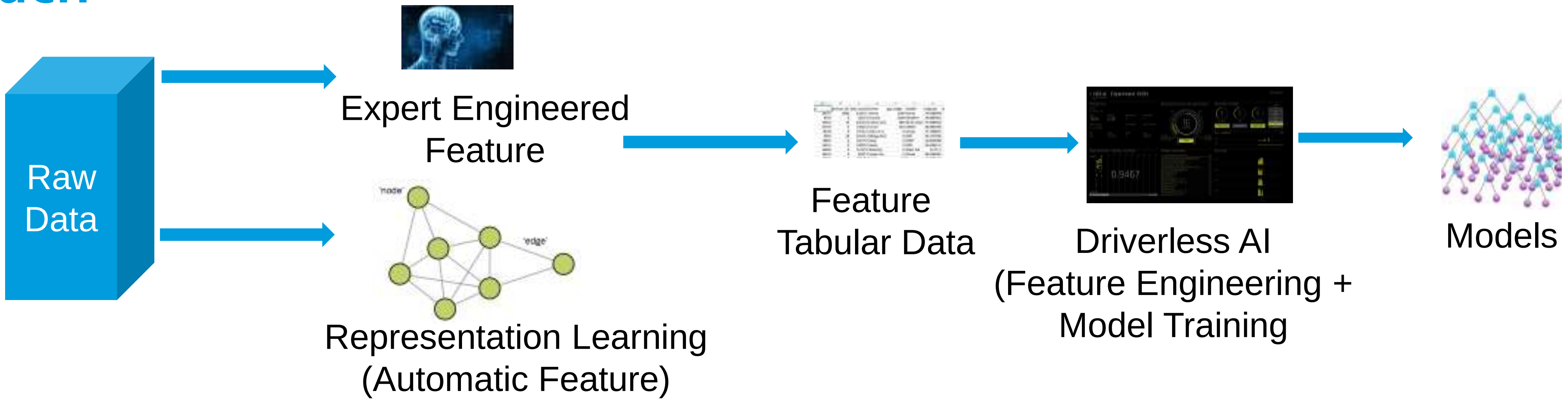
Traditional ML vs End-to-End Learning

- Traditional ML – Significant human effort in engineering features & labelling
- End-to-End Learning – Use algorithm (such as Deep learning) to learn feature representations automatically
  - need tabular data



Images source:

Approach



- Learn features from graphs & use Driverless AI to engineer additional features and build model.
- Feature learning on graphs is fairly new research area & its full potential has not been realized.
- Graph based feature learning helps to understand fraud network & thus prevent collusion and other organized crimes

Images source:

Approach – Graph Based Representation Learning

- Idea
  - You shall know a word (node) by the company (neigbhor) it keep(s)* (Firth, J. R. 1957)
- Word2Vec* - Continuous feature representation for words
  - Suppose user searches for “hotel”, we want to also match “motel”
  - One hot representation (discrete)

$$\text{motel}\ [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0]^T$$
$$\text{hotel}\ [0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0] = 0$$

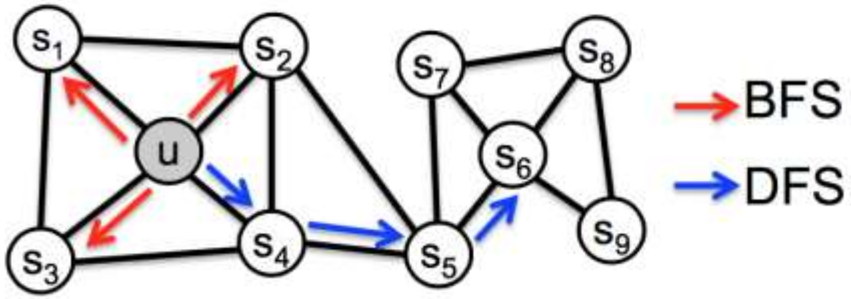
  - Build a dense vector to predict other words from context
  - Two algorithms – Skip Gram (SG) (Predict context words from target) & Continuous Bag of Words (CBOW) (Predict target from context)
- Graph Based Representation Learning
  - Learn continuous feature representation for nodes
  - Representation incorporates community a node belong to & role they play

PayPal

*source: prof. manning’s lecture notes (http://web.stanford.edu/class/cs224n/)

Algorithm – node2vec* Grover & Leskovec, 2016

- Node2Vec
  - s1, s2, s3, s4, u - same community
  - u & s6 also play the role of hub
  - “neigborhood preserving” graph based objective function optimized using SGD
  - MLE optimization problem
  - BFS & DFS sampling to generate neighbors



Algorithm 1 The *node2vec* algorithm.

LearnFeatures (Graph $G = (V, E, W)$, Dimensions d, Walks per node r, Walk length l, Context size k, Return p, In-out q)
  π = PreprocessModifiedWeights(G, p, q)
  $G' = (V, E, \pi)$
  Initialize $walks$ to Empty
  for $iter = 1$ **to** r **do**
    for all nodes $u \in V$ **do**
      $walk$ = node2vecWalk(G', u, l)
      Append $walk$ to $walks$
  f = StochasticGradientDescent($k, d, walks$)
  return f

node2vecWalk (Graph $G' = (V, E, \pi)$, Start node u, Length l)
  Inititalize $walk$ to $[u]$
  for $walk\_iter = 1$ **to** l **do**
    $curr = walk[-1]$
    V_{curr} = GetNeighbors($curr, G'$)
    s = AliasSample(V_{curr}, π)
    Append s to $walk$
  return $walk$

*source: https://arxiv.org/pdf/1607.00653.pdf

Implementation Framework

Human Expert

Raw Events (HDFS)

Graph DB

node2vec

Node Representation Feature vector

Expert Engineered Feature vector

Driverless AI
(Feature Engineering + Model Training)

EXPERIMENTS

Datasets

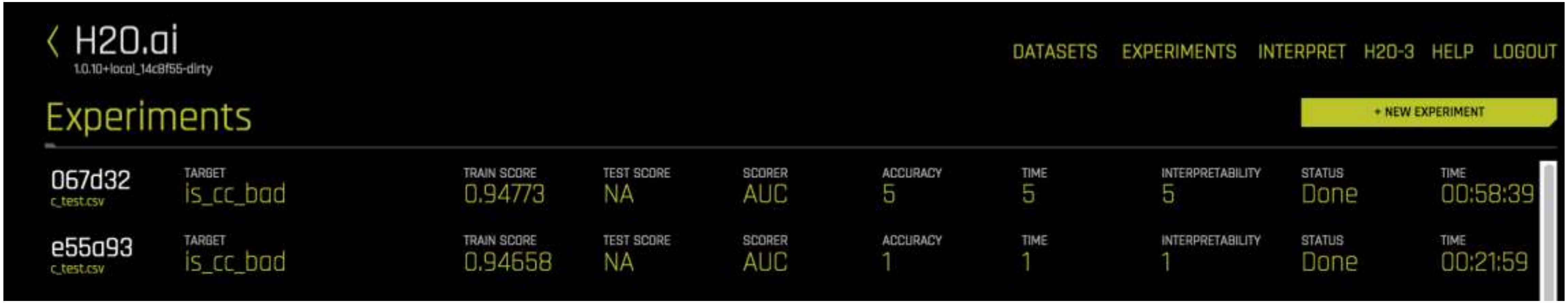
- Training Data
  - Subset of 1 year transactions
  - 1.5 billion edges & 0.5 million nodes
- Test Data
  - 3 months
- # of features
  - 400 - 600

Environment & Tools

- Node2Vec – Node representation learning
- Driverless AI – Feature Engineering & Model Training
- Spark – Data Preparation/Pre-processing
- Hardware – GPU server
  - 4 Pascal 100 GPU
  - 160 cores CPUs
  - 1 TB RAM

Experiment

- Training time (subset of data) – Driverless AI on GPU 6x faster
  - laptop (accuracy 1) - ~ 2 hours
  - GPU (accuracy 1) – 21 minutes; (accuracy 5) – 58 minutes



Experiment

- Top 5 variables from DAI
- AUC – 0.9477

CONCLUSIONS

Conclusions

- Graph based representation learning yield robust feature set for complex fraud patterns such as collusion fraud
- Driverless AI not only help to engineer additional features automatically but also significantly improve model training time (under 2 hours).
- Journey into DAI just beginning…
- Next steps
  - Evaluate Driverless AI results on out-of-time data sets.
  - Evaluate Driverless AI directly on raw data
  - Evaluate representation learning on edges and weighted graphs
  - Machine learning on graphs

Acknowledgements

Driverless AI Team @ H2O.ai