

Deep Fraud

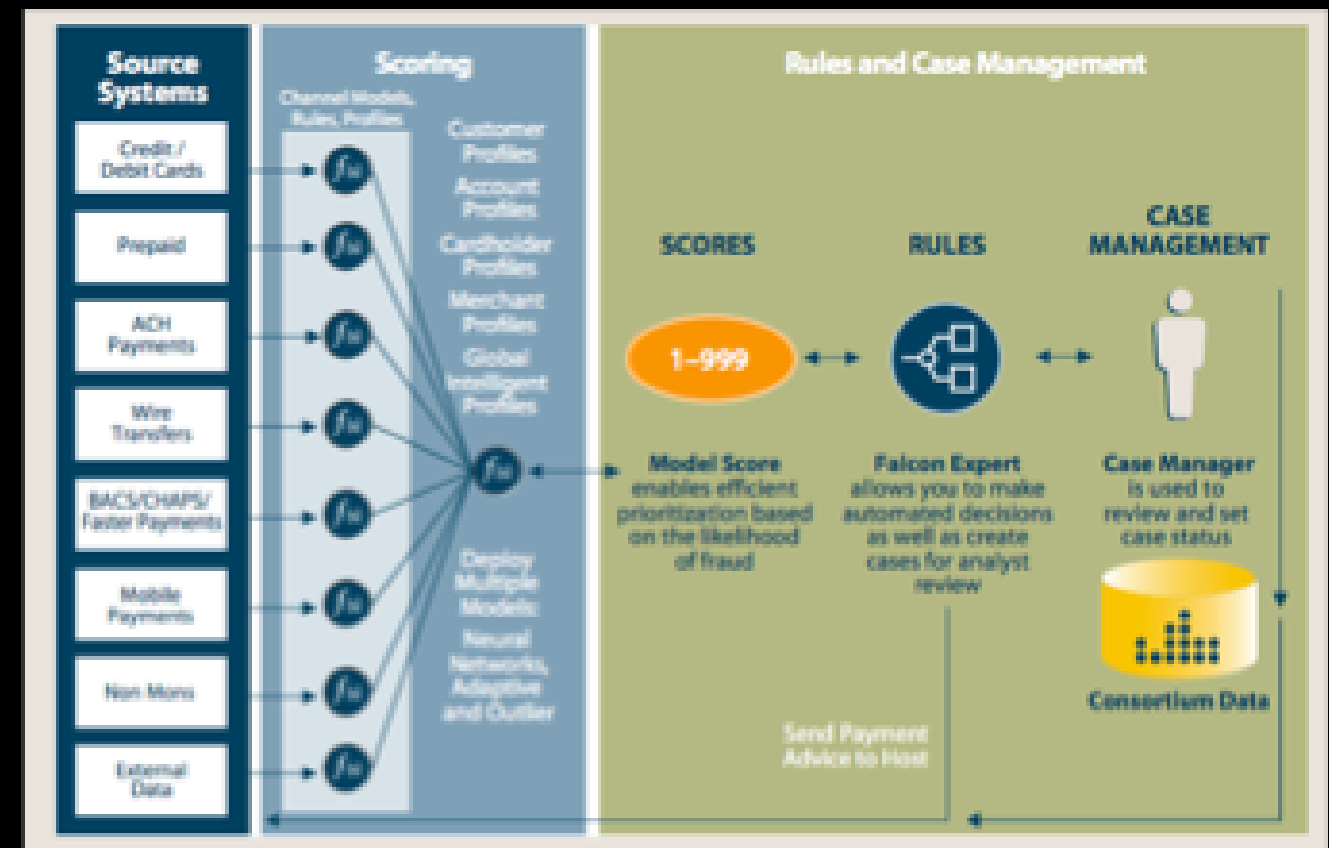
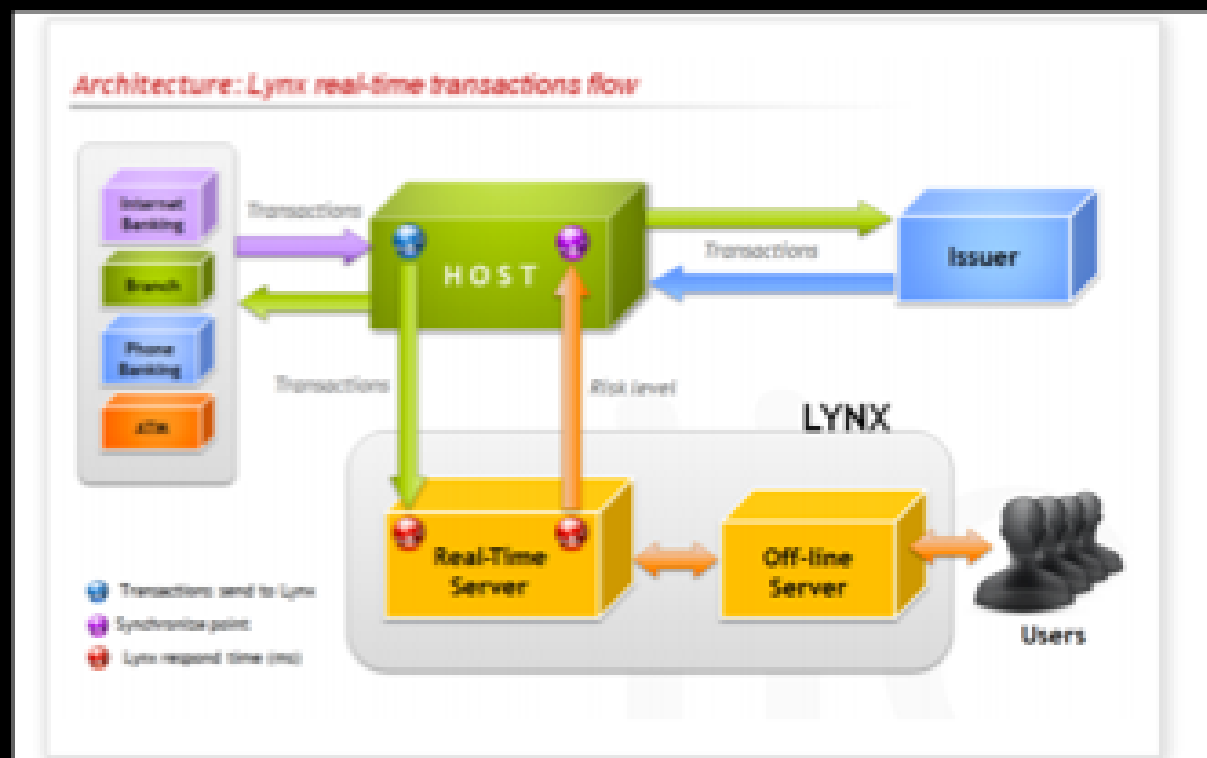
- Doing research with financial data

A black and white photograph of a hand holding a piece of chalk, writing on a chalkboard. The hand is positioned on the right side of the frame, with the index finger pointing towards the left. The chalkboard surface is visible in the background, with some faint, illegible markings. Overlaid on the center of the image is the text 'Chapter 1: Problem Statement' in a large, bold, white sans-serif font.

Chapter 1: Problem Statement

- Each time a new card payment arrives to our mainframe, provide a risk score (from 0 to 99)
- Apply this score together with some business rules to decide if the payment is accepted or not

How is being doing this?

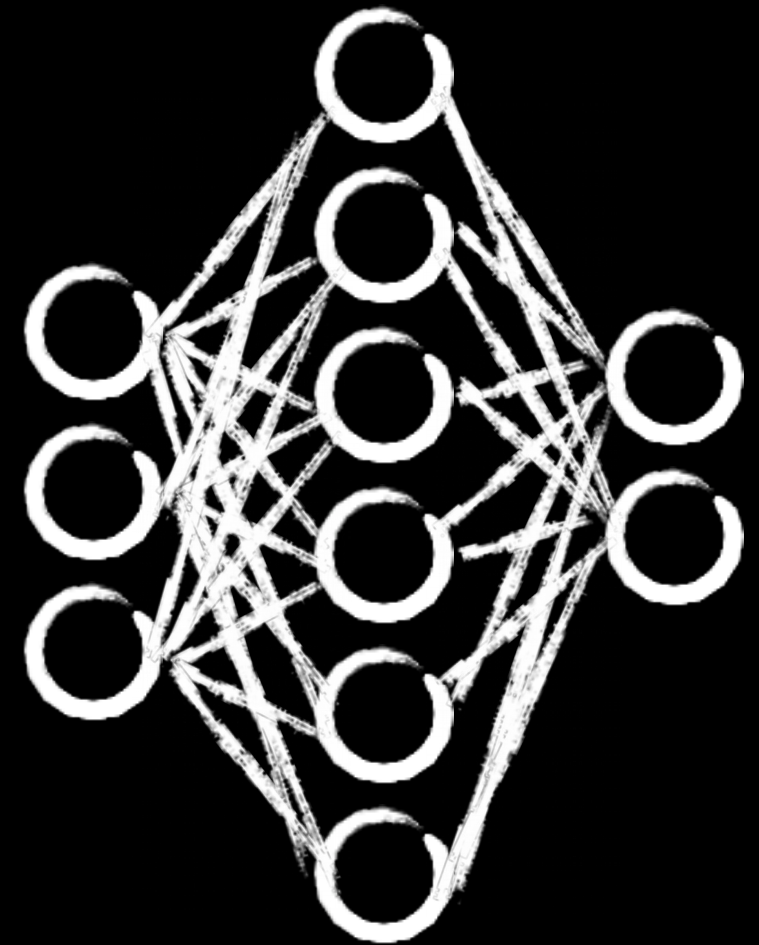


FICO® Falcon® Fraud Manager

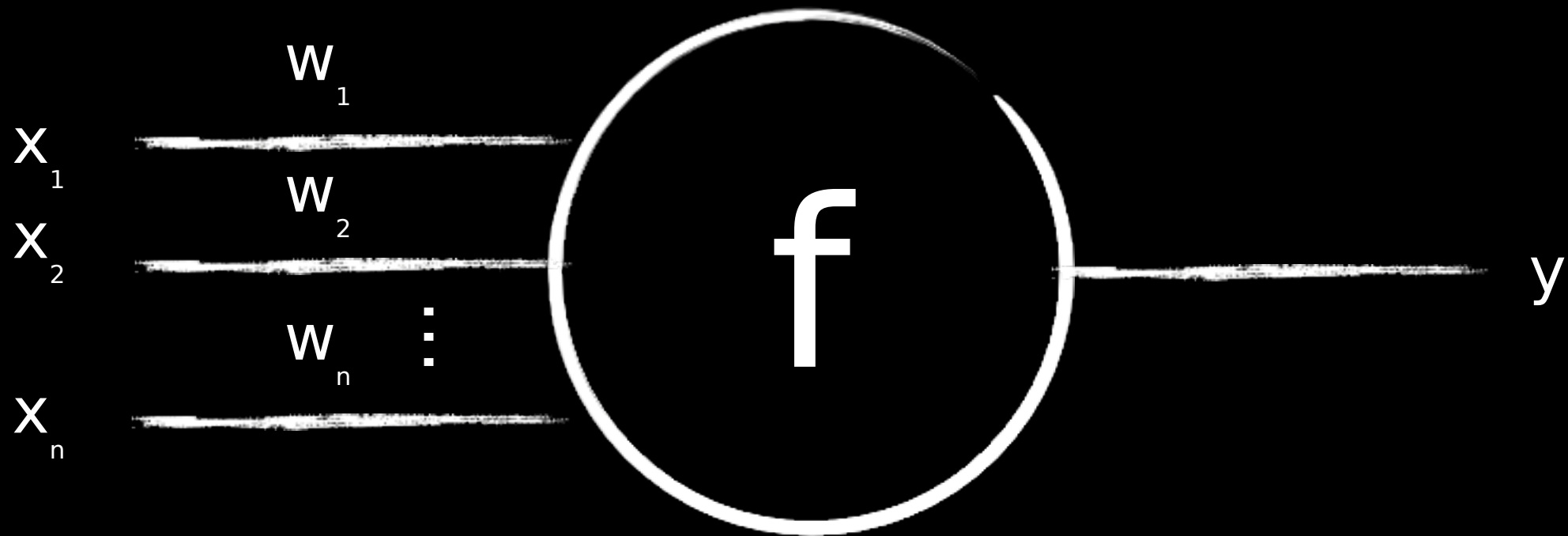
External companies with very expensive prices

We want to but ... how?

- Using an Artificial Neural Network
- Training data: labeled past transactions



A First Look at NN

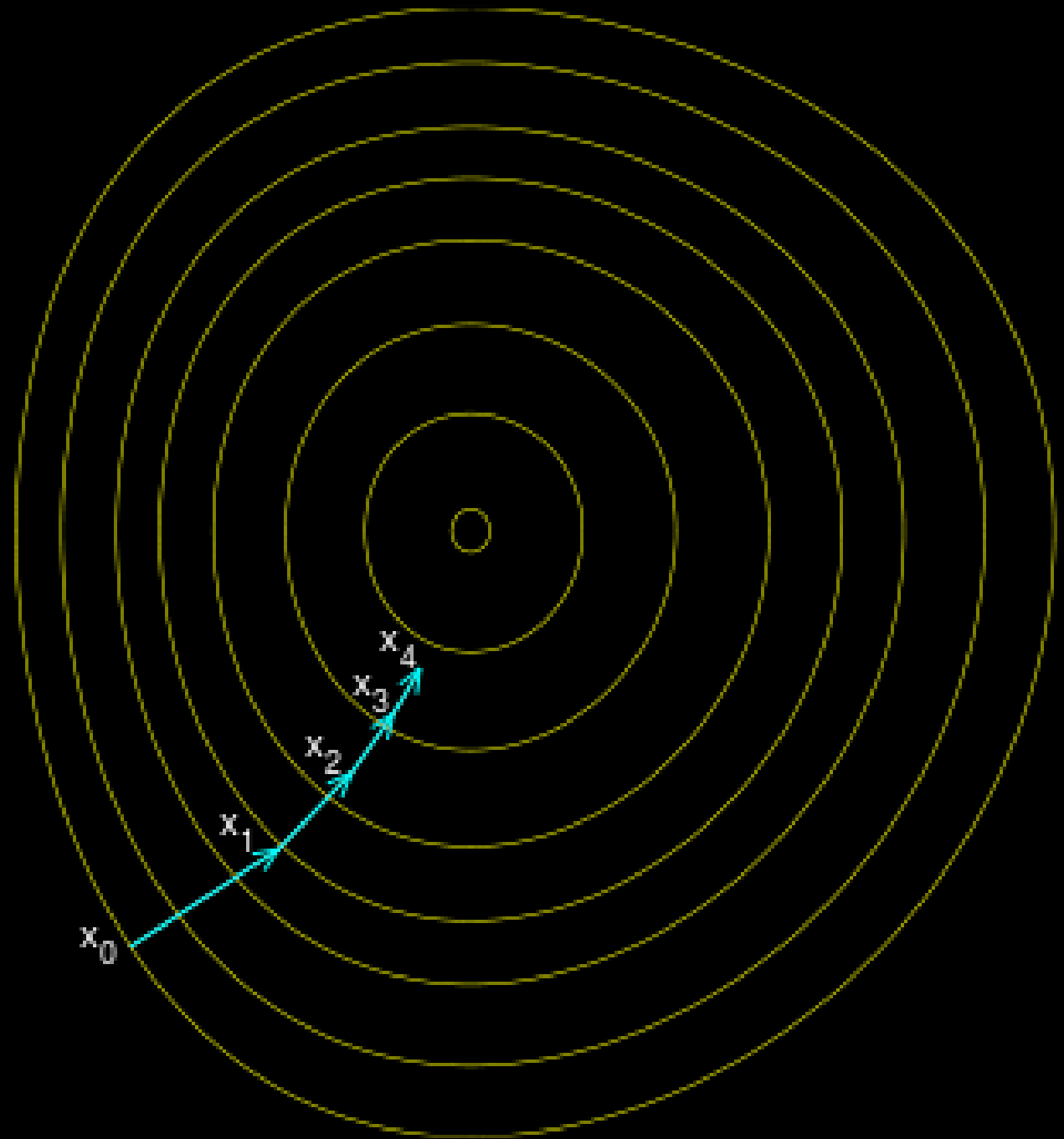


Schematic for a neuron in a neural net

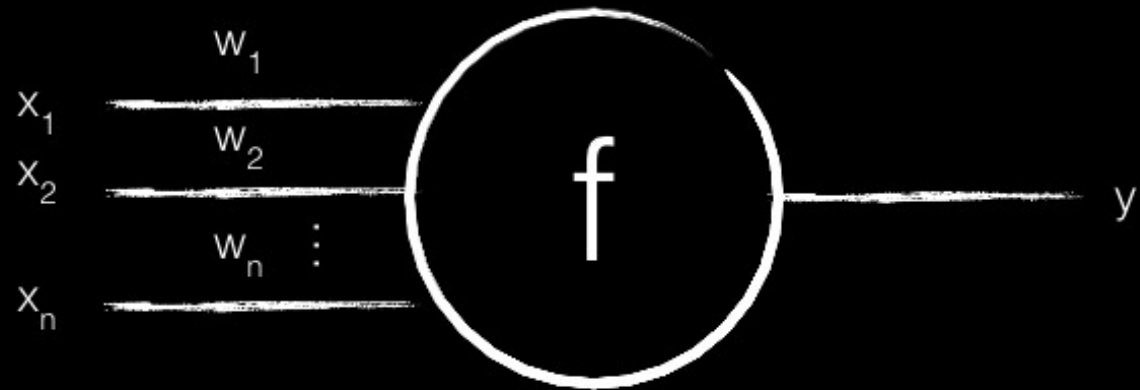
Gradient Descent

it's based on the next idea:

- if $F(x)$ is **defined** and **differentiable** in a point a
- $F(x)$ decreases fastest if one goes from a in the direction of the negative gradient of F



Training a Single Neuron

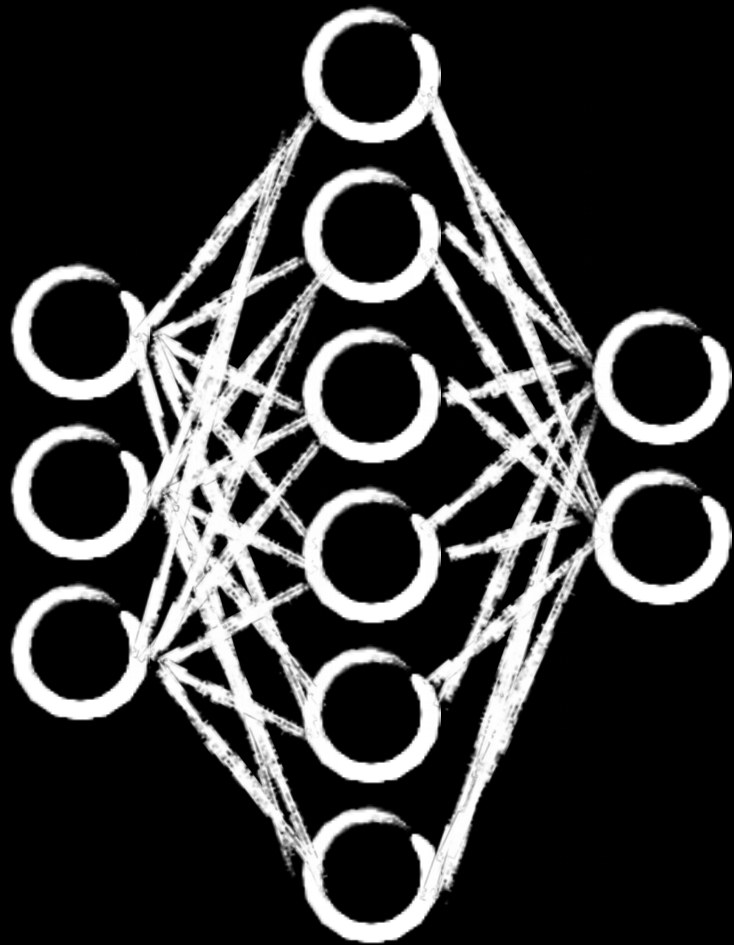


$$z^{(i)} = w_1^{(i)} x_1^{(i)} + w_2^{(i)} x_2^{(i)} + \dots + w_n^{(i)} x_n^{(i)}$$

$$E = \frac{1}{2} \sum_i (t^{(i)} - y^{(i)})^2$$

$$\Delta w_k = \sum_i \epsilon x_k^{(i)} (t^{(i)} - y^{(i)})$$

But there are many neurons...



By applying the chain rule:

$$\frac{\delta E_{total}}{\delta w_{o_1}^1} = \frac{\delta E_{total}}{\delta out_{o_1}} \times \frac{\delta out_{o_1}}{\delta net_{o_1}} \times \frac{\delta net_{o_1}}{\delta w_{o_1}^1}$$

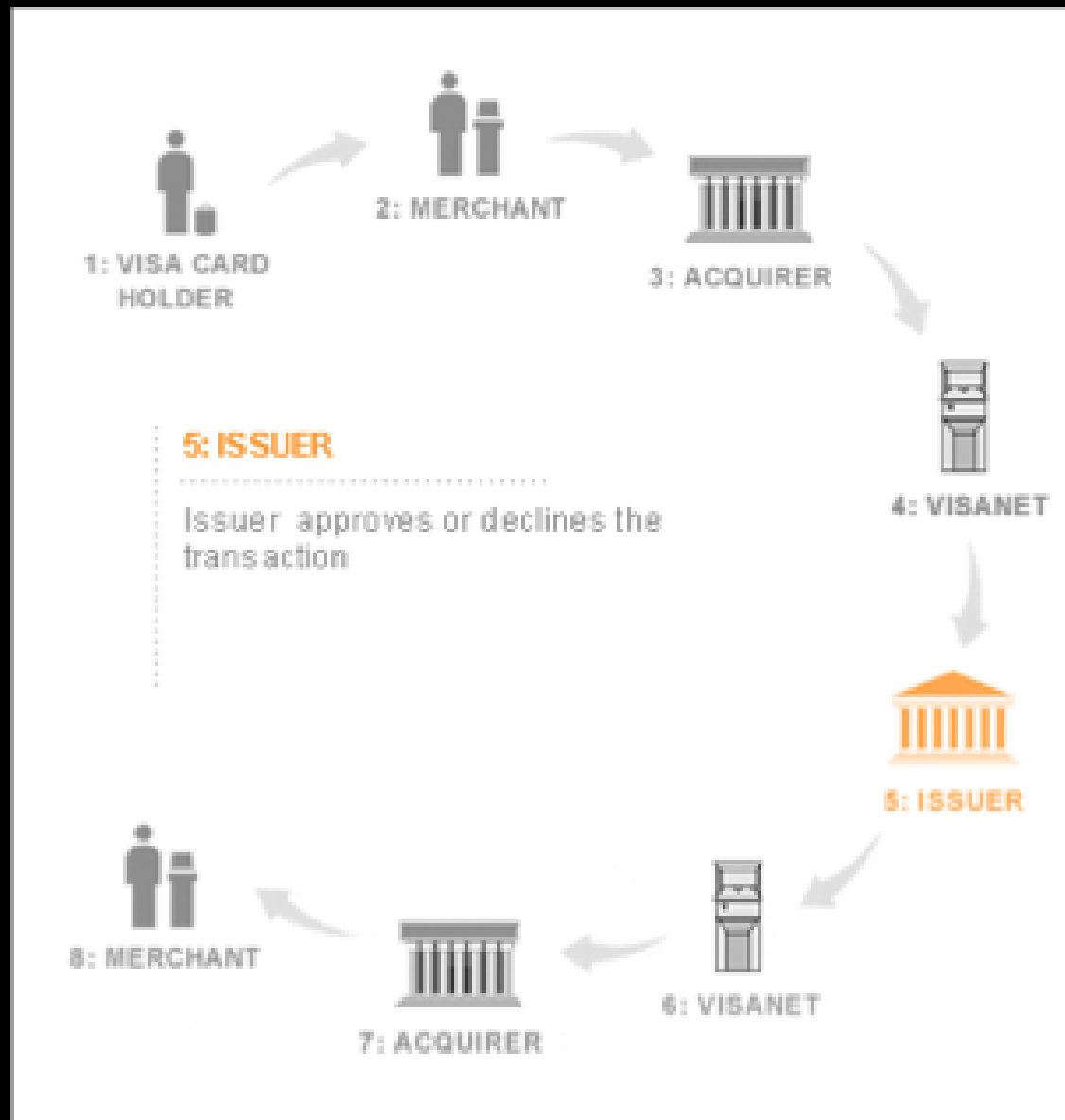
This seems tedious...

- With the advent of Deep Learning, there are many libraries to make you life easier:
 - TensorFlow (google)
 - Theano (U. of Montreal)
 - Caffe (U. of Berkley)
 - Torch (Facebook, Twitter, Google)
 - ...



Chapter 2: Are Credit card payments big data?

- BBVA processes 903,646,280 transactions during 2014 (around 30 transactions per sec.)
— Spain



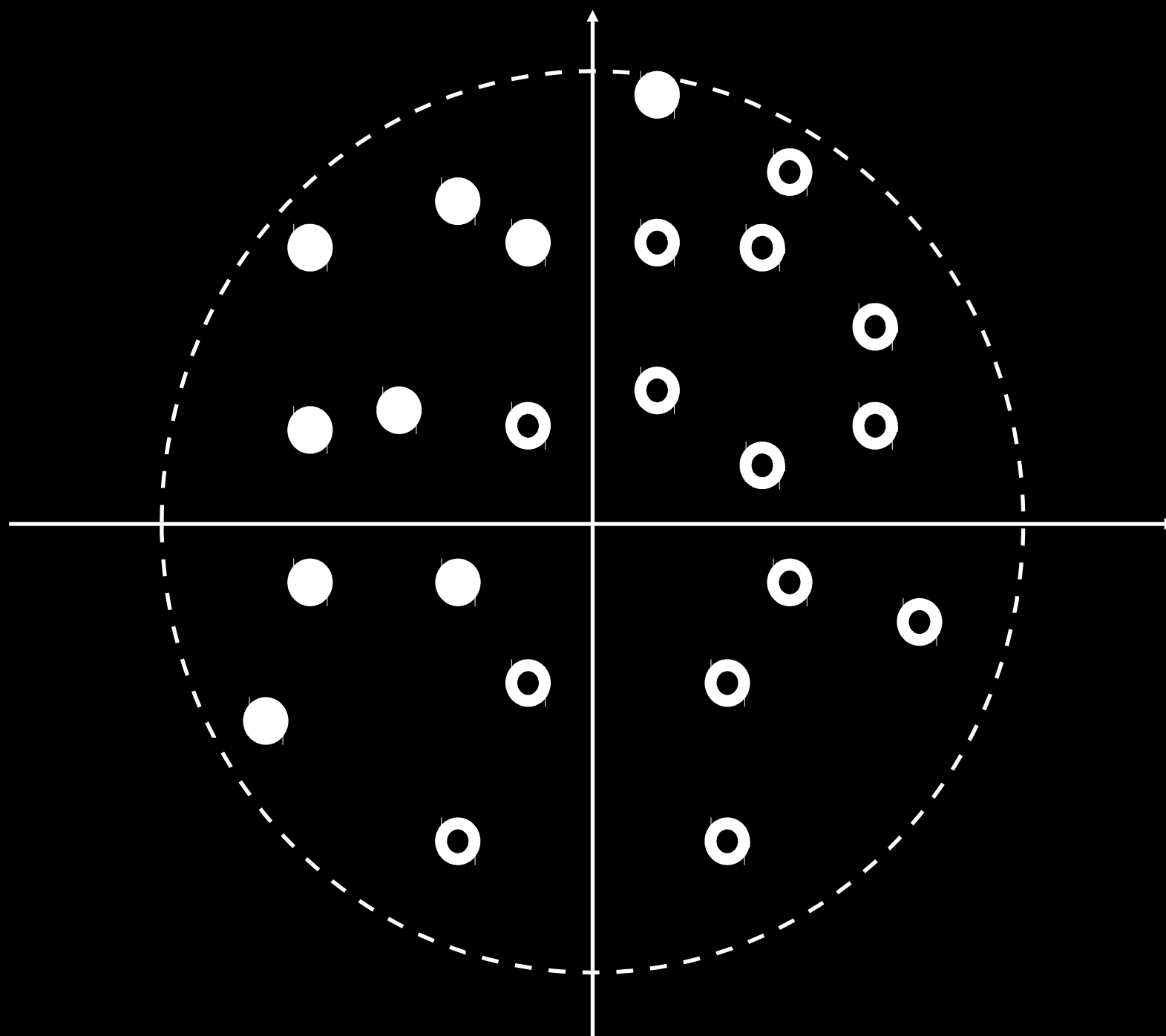
- Local merchant transaction
- Verified by Visa

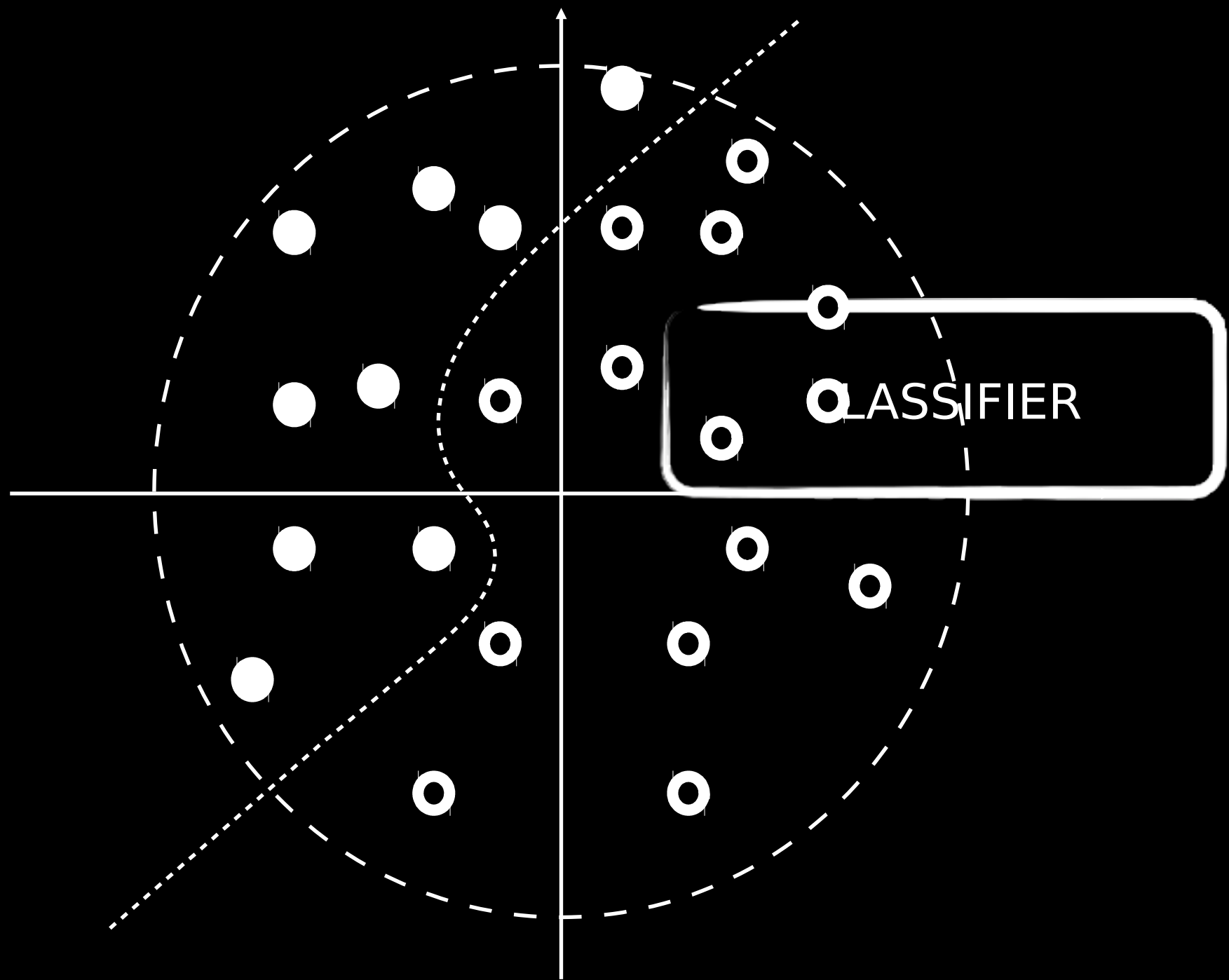
It doesn't matter the type of loop, the answer must be provided in less than 2 seconds!

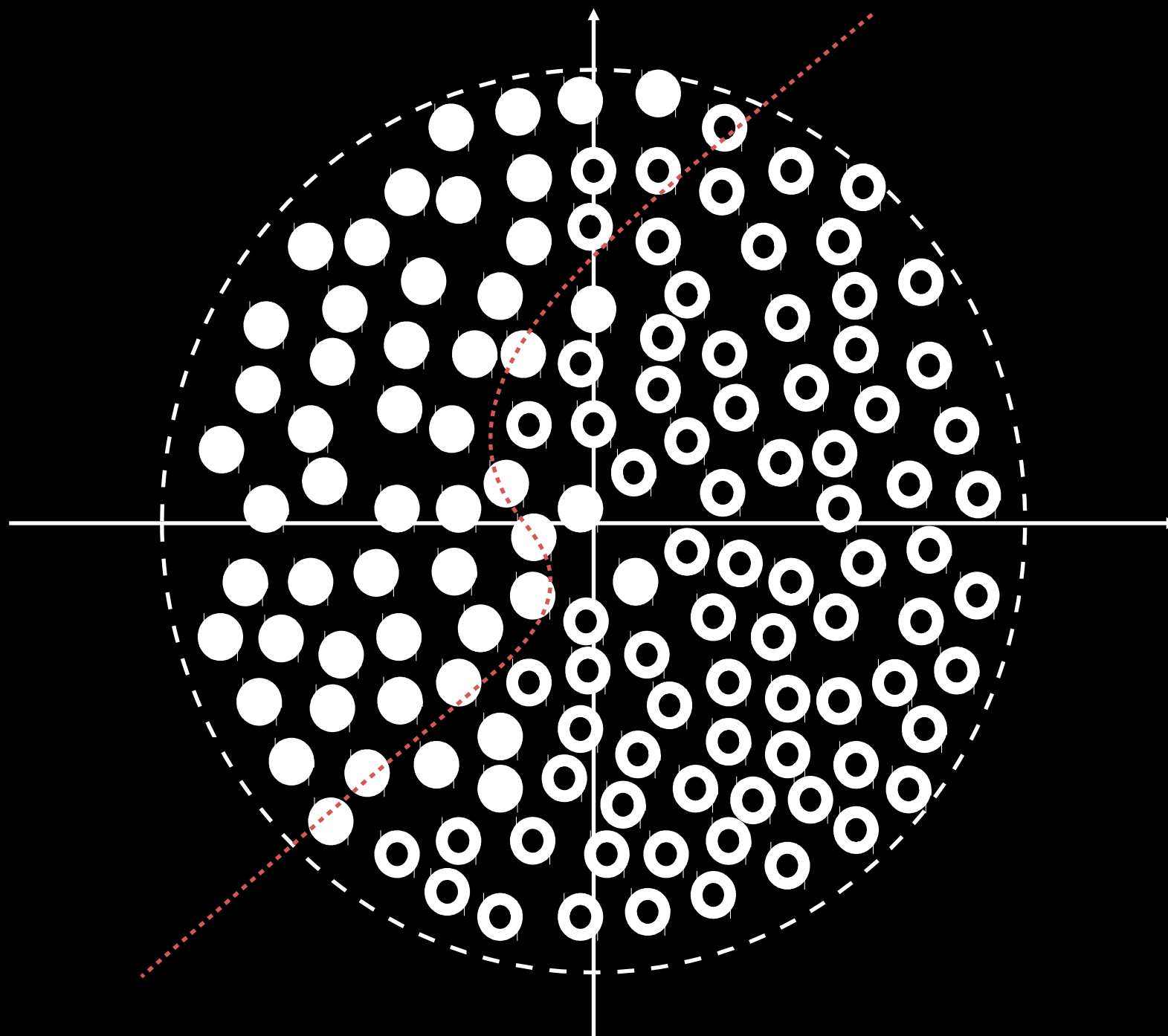


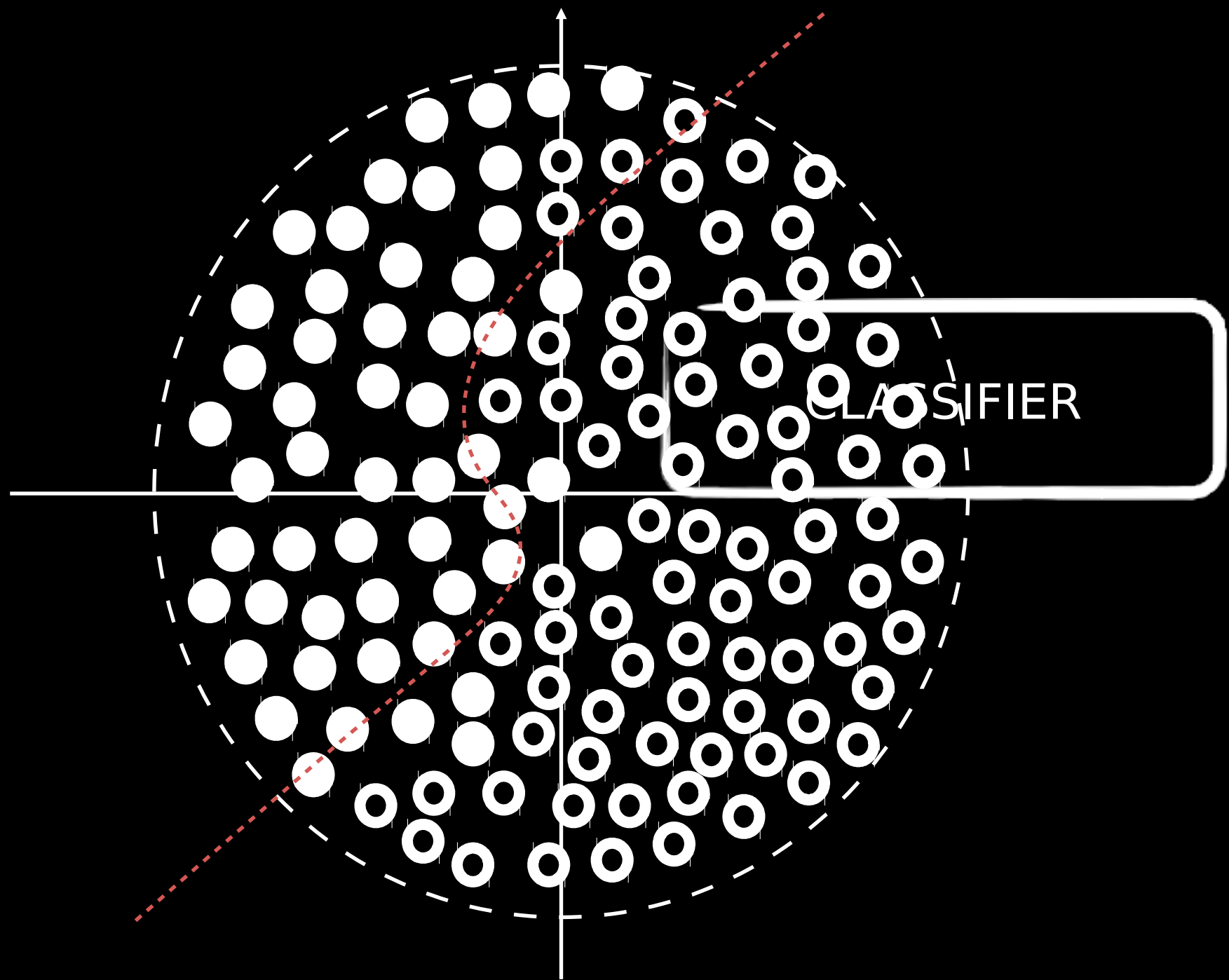
NN + big data
learning by Boredom

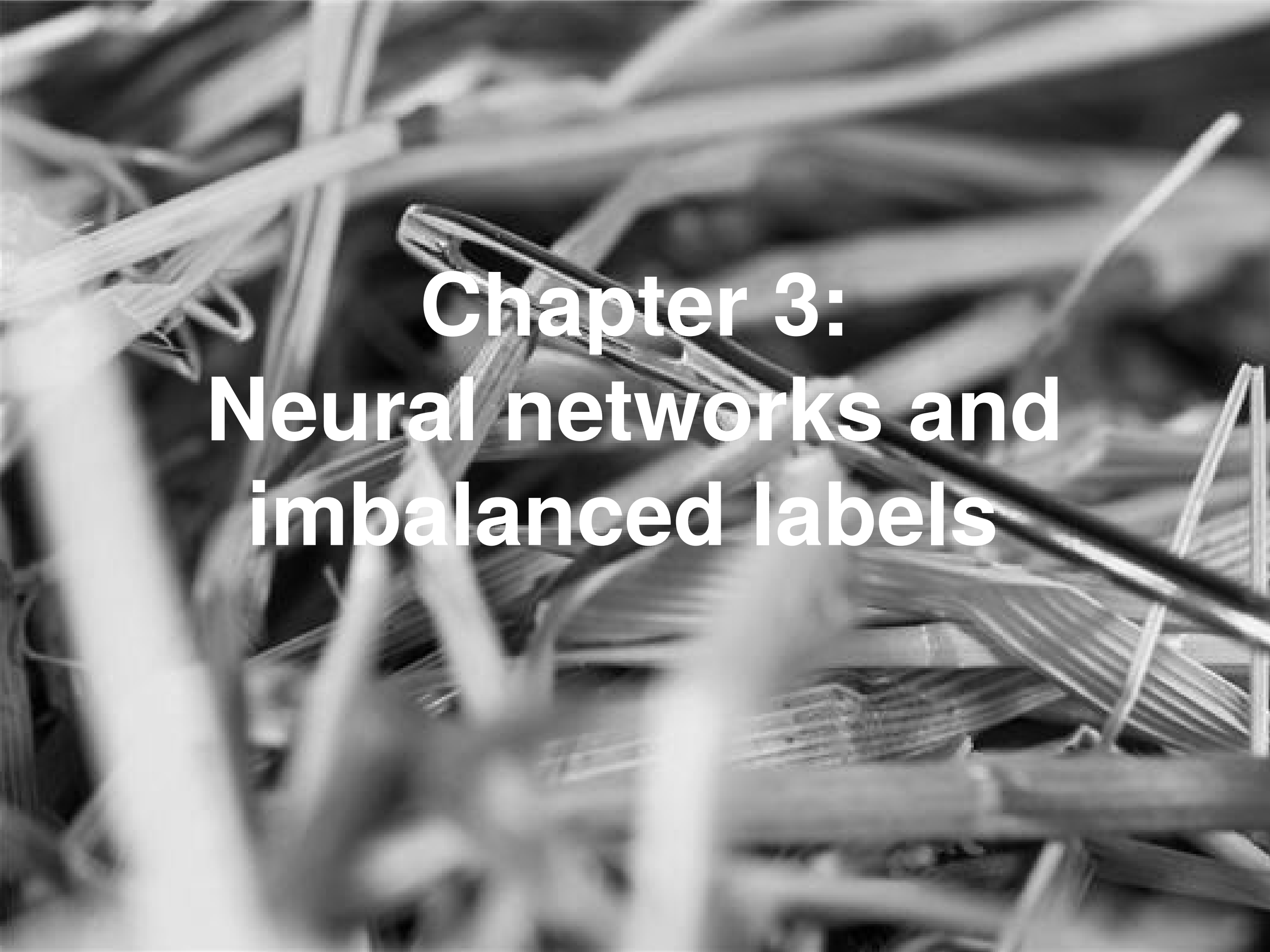










A black and white photograph of a dense thicket of reeds or tall grasses. A fountain pen is resting on one of the stalks in the center. The text "Chapter 3: Neural networks and imbalanced labels" is overlaid in white, bold, sans-serif font.

Chapter 3: Neural networks and imbalanced labels

The enormity of the tragedy

- 1 / 6000 fraudulent transactions (0,016667%), it's more likely to find a cat in a youtube video
- even worse! Fraud is not correctly labeled (noise)

Fraud database information



- Fraud database does not include the timestamp, only the date.
- Amounts are not exactly the same
- The number of transactions composing a reported fraud is unknown.

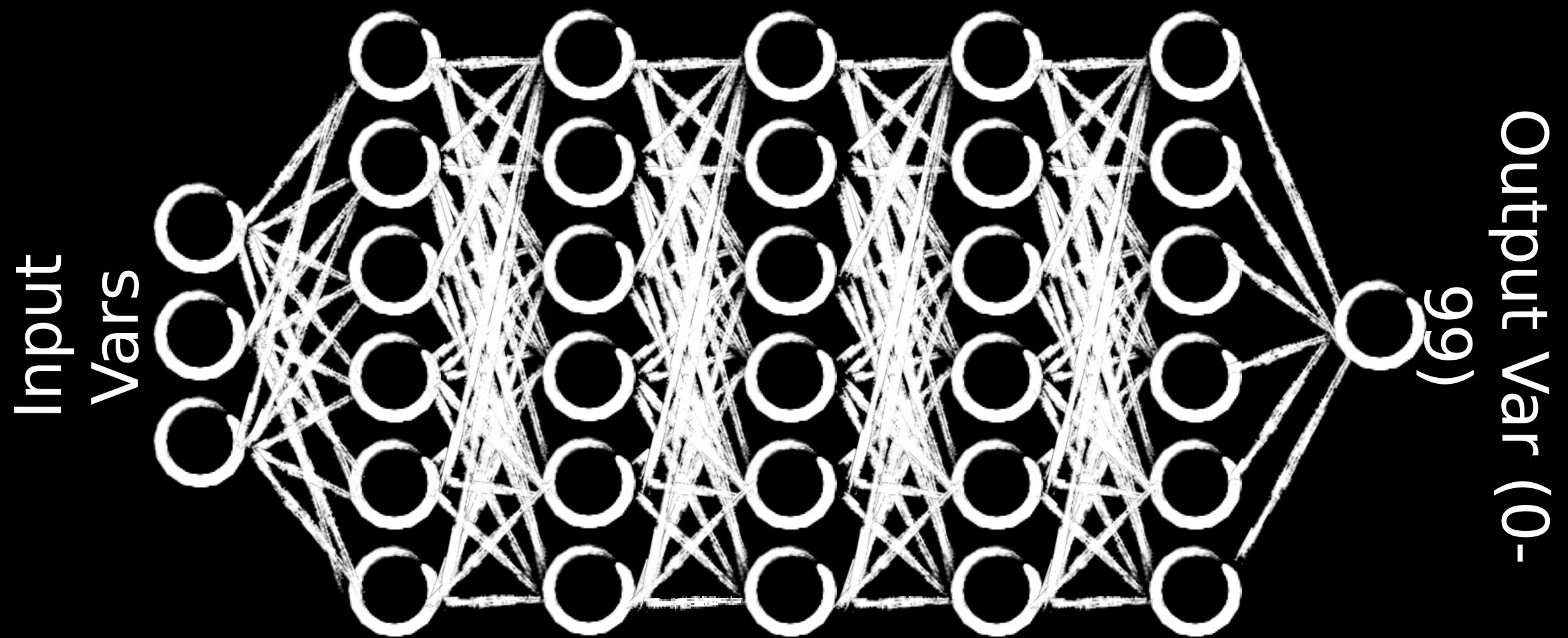
Record linkage rules

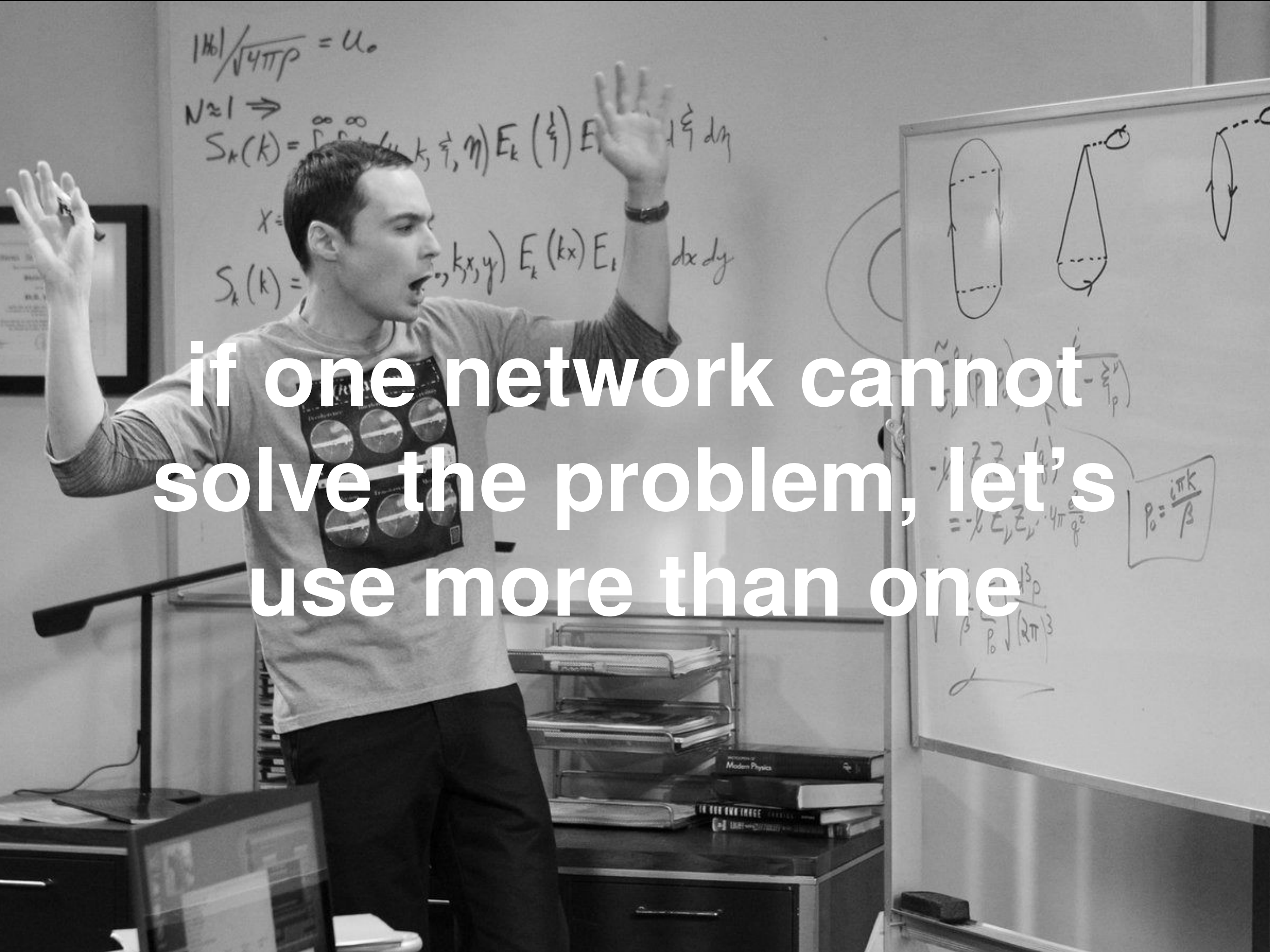


- If a client reports a fraud between +/- 36 hours of a transaction, and such transaction differs less than 10% of the reported fraud, the transaction is considered fraud.
- We did not try to group different transactions within the fraud time period (to be improved)
- With these simple rules a only of fraud 13% is not linked

First attempt

- If deep learning is so cool and powerful, let's create a single big network

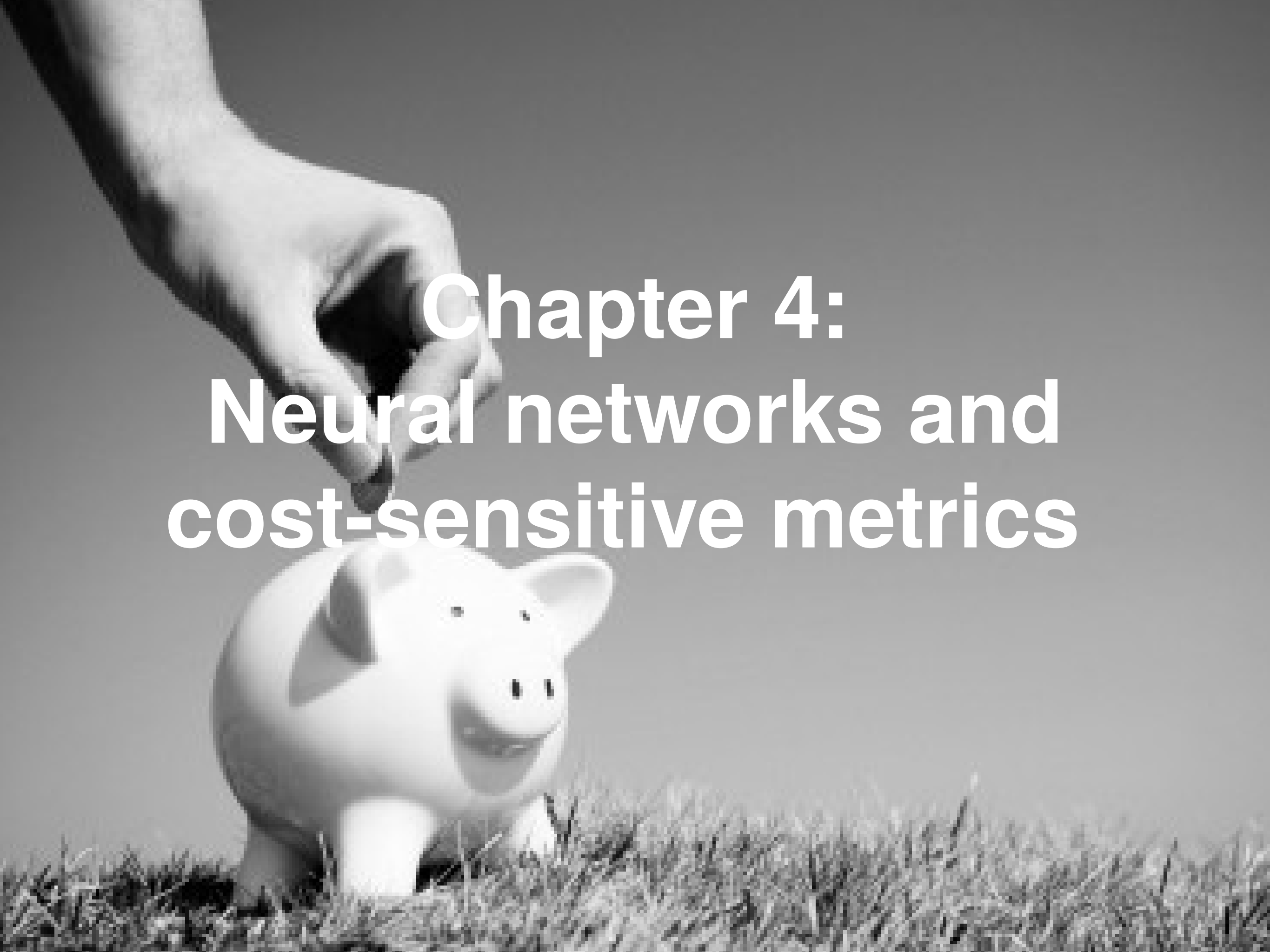




if one network cannot
solve the problem, let's
use more than one

- **New goal:** remove as many non-fraud transactions as possible while conserving the maximum number of fraud transactions
- **Filters with NN:**
 - Output: [1,0] non-fraud, [0,1] for fraud
 - If $(\text{output0} - \text{output1}) > 0$ then discarded
 - **Downsampling** in some sense

Dataset	# Trans.	%	Fraud
Original	903.646.28 0	100 %	100 %
1st Filter	646.669.68 1	71,6 %	95 %
2nd Filter	53.254.006	5,9 %	88 %
3rd Filter	9.955.055	1,1 %	45 %

A black and white photograph of a hand dropping a coin into a piggy bank. The piggy bank is a pig-shaped container, and the coin is being dropped from the top. The background is a textured, grassy surface.

Chapter 4: Neural networks and cost-sensitive metrics

- A cost Sensitive metric evaluates the cost associated with misclassifying observations.
- Normally, it's defined as $C(\text{FN})$, $C(\text{FP})$ in the confusion matrix
- In our case cost is data-driven, the amount of the payment

Business metrics

- Value detected rate: ~25%
 - Blocked €25 / €100 fraud transaction
- Positive predictive rate (TFPR): ~20%
 - Detected 1 fraud transaction with 4 false positive
- Model stability is a must

Neural Network Details

- MultiLayer Perceptron
- Rectified Linear (ReLu) hidden units: $f(x)=\max(0,x)$
- For the sake of stability and convergence of the SGD back propagation we use batch normalisation (50% F - 50% NF)
- Combines two networks, one with 1 hidden layer and another with 2 (256 neurones each) and 1 output neurone.
- MaxNorm regularisation is set to 3.0



Chapter 5: Conclusions

Research Conclusions

- Real datasets are not simple (human + legacy)
- In 6 months we were able to obtain decent results
- At least for this topic, ANN are flexible enough to overtake all the problems we had to handle

ANY

QUESTIONS

?