Using Markov Chain Approximation to Identify Financial Fraud

Floriana Zefi

floriana.zefi@ing.com

Advanced Analytics @ING Group, Amsterdam

ING

PyData Amsterdam 2019

Markov Chains are Widely Used

Outline

- Introduction
  - Financial fraud
  - Artificial Intelligence (AI) in Financial Sector
  - Combating Fraud with AI
- Anomaly Detection with Markov Chains
  - Markov Chain Approximation
  - Sequential Data
  - Encoding Strategy
- Demo Notebook
  - Build Markov Features from Sequential Data
- Conclusions and Prospects

Artificial Intelligence in Financial Sector

> " Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed."

AI is widely employed to solve complex problems in different areas

Some applications of machine learning in the Financial sector:

- Trading strategies
- Credit risk assessment
- Fraud detection

AI is a successful tool to battle financial fraud

Financial Fraud

- There are different types of fraud in financial sector.
- **Commonly known:**
  - Transactional fraud
  - Loan fraud
  - CxO fraud

Combating Fraud with AI

- Advanced methods are developed to:
  - Identify fraudulent patterns
  - Separate them from legitimate clients
  - Suppress fraud

The main challenges for model building are:

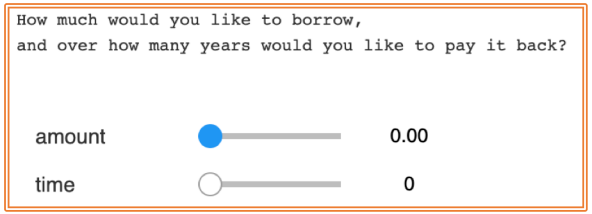
- Imbalanced datasets
- Concept drift: financial fraud is evolving
- Normally we do not have a target / labels

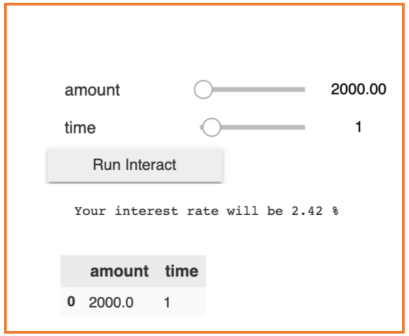
Also new opportunities for detecting fraud:

- Web-collected data

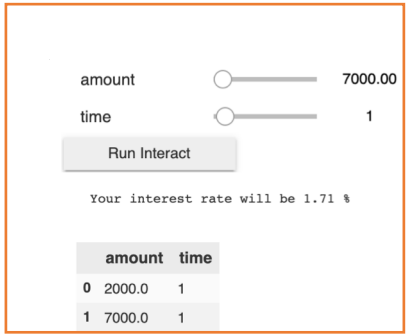
Sequential Data

Applying for financial products online, e.g. request a loan:

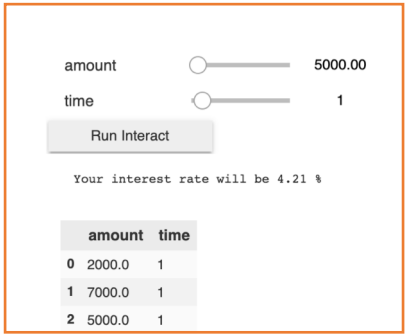




1



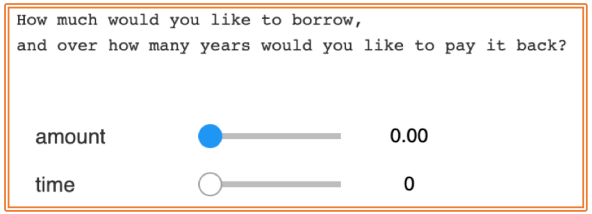
2

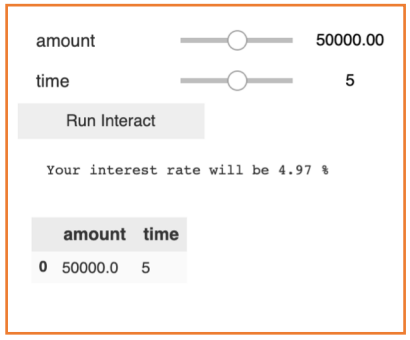


3

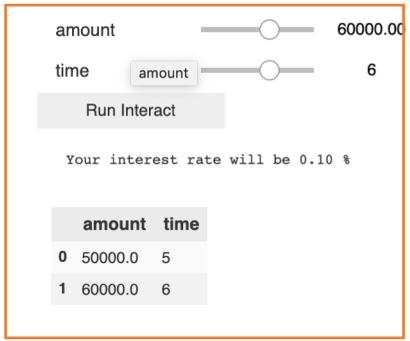
Sequential Data

Applying for financial products online, e.g. request a loan:

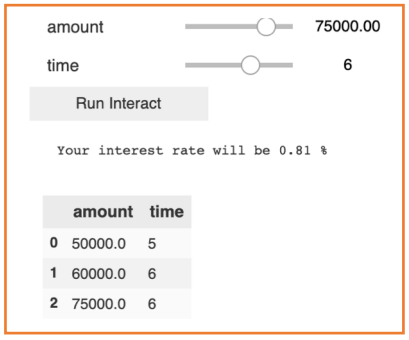




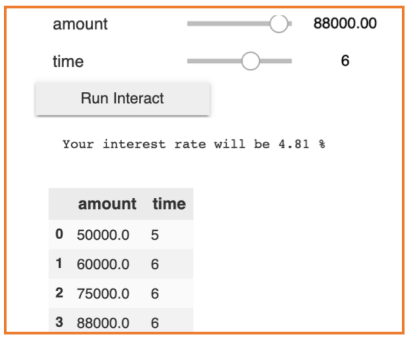
1



2



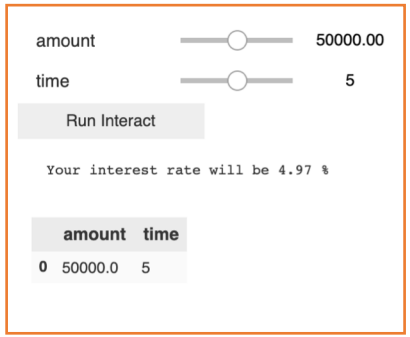
3



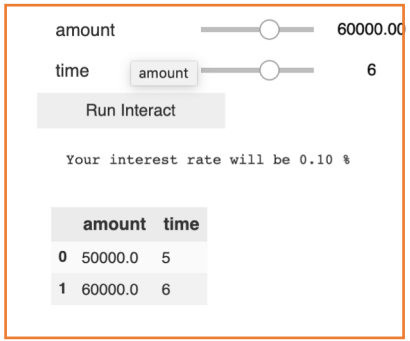
4

Sequential Data

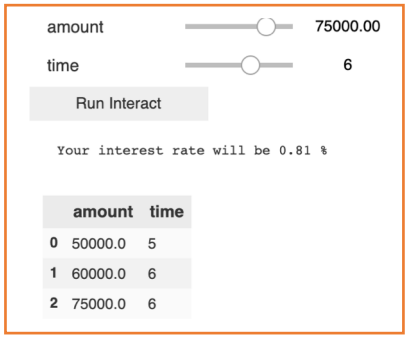
This is just a small part of many sequential interactions you could capture from web interactions



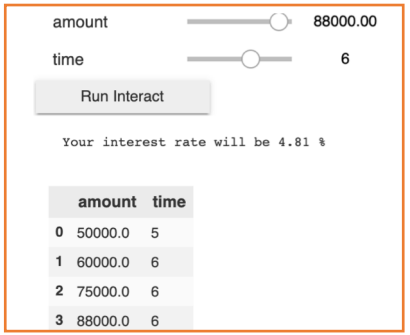
1



2



3



4

Encoding Strategy

- The range of the numbers can be wide and different form one case to another.
- The direction of changes could give some information about behavioral patterns
- Encoding / binning of changes required
- The encoding strategy for some features from the web-data:

Δ(feature_1)	Δ(feature_2)	Δ(feature_3)	Δ(feature_4)	Δ(feature_5)	Δ(feature_5)
1	1	1	1	1	1
1 ↗	0 —	-1 ↘	1 ↗	0 —	-1 ↘

- -1 : smaller value than in the previous trial
- 1 : larger value than in the previous trial
- 0 : same value as in the previous trial

Encoding Strategy

person_nr_1

feature_1	feature_2	feature_3	Δ(feature_1)	Δ(feature_2)	Δ(feature_)	STATE
x	**y**	**z**	**1**	**1**	**1**	**1**
x1 (> x)	y1 (= y)	z1 (= z)	1	0	0	5
x2 (> x1)	y2 (= y1)	z2 (= z1)	1	0	1	10
x3 (> x2)	y3 (= y2)	z3 (= z2)	1	0	0	5

person_nr_2

feature_1	feature_2	feature_3	Δ(feature_1)	Δ(feature_2)	Δ(feature_)	STATE
a	**b**	**c**	**1**	**1**	**1**	**1**
a1 (< a)	b1 (< b)	c1 (= c)	-1	-1	0	24
a2 (= a1)	b2 (> b1)	c2 (= c1)	0	1	0	9

And so on . . .

Combating Fraud with AI

- Our modeling philosophy is to combine several well understood features in a complex model
- Using web-collected data we created encoded sequential data
- From the sequential: have features that contain information on how these values change
- In general, the fraudulent class shows different patterns from the legitimate clients.
- **Can we use this data to identify anomalies?**

Anomaly Detection

Anomaly detection by matching probabilities:

We model the 'non-fraudulent' population and then measure how likely it is that a new observation belongs to the normal population

Main assumption:

- *Fraudulent behavior is different from non-fraudulent behavior*
- The most legitimate applications follow a similar path and fraudulent applications follow a lot of different, more creative paths

If these assumptions are true, it allows to separate fraudulent cases from legitimate cases

Markov Chain Approximation

We want to e**stimate the sequence probability**:

$$P(x_1, x_2, x_3, \ldots, x_n) = P(x_1) \cdot P(x_2|x_1) \cdot P(x_3|x_1, x_2) \cdots P(x_n|x_0, \ldots, x_{n-1})$$

In the **Markov chain approximation**, the probability for each state **only depends on the previous state**:

$$P(x_1, x_2, x_3, \ldots, x_n) = P(x_1) \cdot P(x_2|x_1) \cdot P(x_3|x_2) \cdots P(x_n|x_{n-1})$$

The **conditional probabilities** for bi-states can be learned from data as:

$$P(x_n|x_{n\_1}) = \frac{\text{count}(x_{n-1}, x_n)}{\text{count}(x_{n-1})}$$

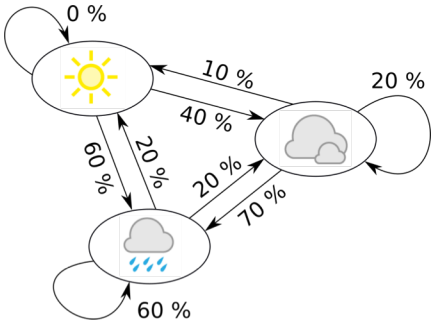
Example:

$$P(\text{I do not like chocolate}) = P(\text{do}|\text{I}) \cdot P(\text{not}|\text{do}) \cdot P(\text{like}|\text{not}) \cdot P(\text{chocolate}|\text{like})$$

Markov Chain Approximation

Example:

Weather in Amsterdam
sunny –rainy – cloudy



Probability Matrix

$$
\begin{array}{cc} & \\ X_t & \begin{matrix} \text{sunny} \\ \text{cloudy} \\ \text{rainy} \end{matrix} \end{array}
\begin{array}{c} X_{t+1} \\ \begin{matrix} \text{sunny} & \text{cloudy} & \text{rainy} \end{matrix} \\ \begin{pmatrix} 0.0 & 0.4 & 0.6 \\ 0.1 & 0.2 & 0.7 \\ 0.2 & 0.2 & 0.6 \end{pmatrix} \end{array}
$$

Transition Probability Matrix

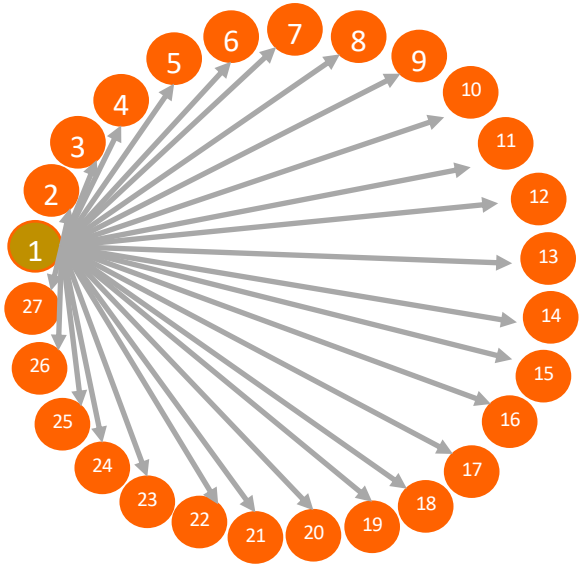
- For three features, which take three possible values: there are 27 (3 x 3 x 3) possible states.
- The goal of the Markov chain training algorithm is to calculate the probabilities matrix, a 27 x 27 state transition probability matrix from the training data.
- An element P(i,j) for the matrix tells us that that given the last change of type i, the probability of the next state being of type j is P(i,j).
- The encoding startegy is used to calculate the transition probability matrix :

$$X_t \left\{ \begin{pmatrix} p_{11}, p_{12}, \ldots\ldots, p_{1i-1}, p_{1i} \\ \ldots\ldots\ldots\ldots\ldots \\ \ldots \\ \ldots \\ \ldots \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ p_{1i}, p_{2i}, \ldots\ldots, p_{i-1j-1} p_{ij} \end{pmatrix} \right.$$

$$X_{t+1}$$

State Transition Probabilities

State 1 can be followed by any of the other states:

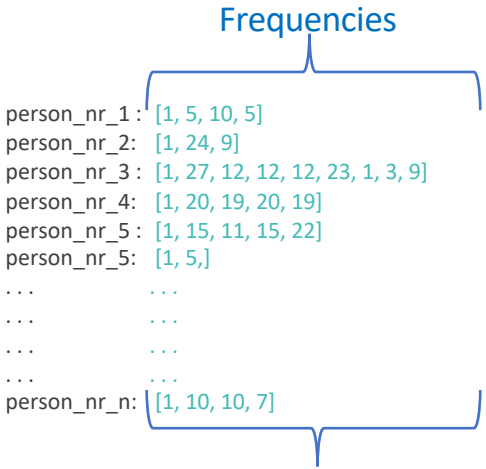


The sequences in the example are:

person_nr_1 : [1, 5, 10, 5]		P(1, 5, 10, 5) = P(1|5) *x* P(10|5) *x* P(5|10)
person_nr_2: [1, 24, 9]		P(1, 24, 9) = P(1|24) x P(24|9)
. . .	. . .	. . .
. . .	. . .	. . .

State Transition Probabilities

Use the training sample to calculate probability matrix



Sequence probabilities

P(1, 5, 10, 5) = P(1|5) *x* P(10|5) *x* P(5|10)
P(1, 24, 9) = P(1|24) x P(24|9)
. . .
. . .

From this array of probabilities, we can extract Markov based features

Building Markov Features

Time for the demo notebook