CREDIT CARD FRAUD DETECTION USING HIDDEN MARKOV MODELS

A Term Project for course

Introduction to Stochastic Processes and Their Applications (IME625A)

By

ASHWIN ASHOK KUMAR
(Roll Number-18114007)

DANISH NAWAZ
(Roll Number-18114008)

PATEL DHRUV
(Roll Number-18114010)

PARAS ARORA
(Roll Number – 18114016)

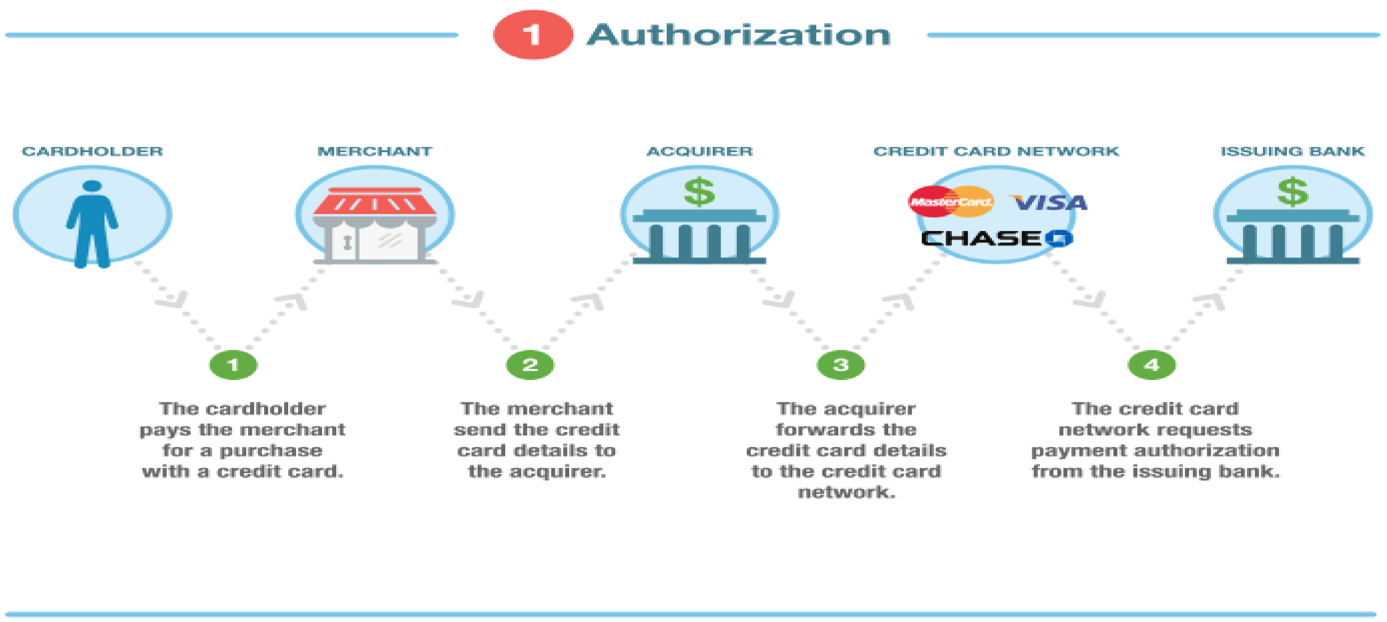
Guided By:
PROF. AVIJIT KHANRA

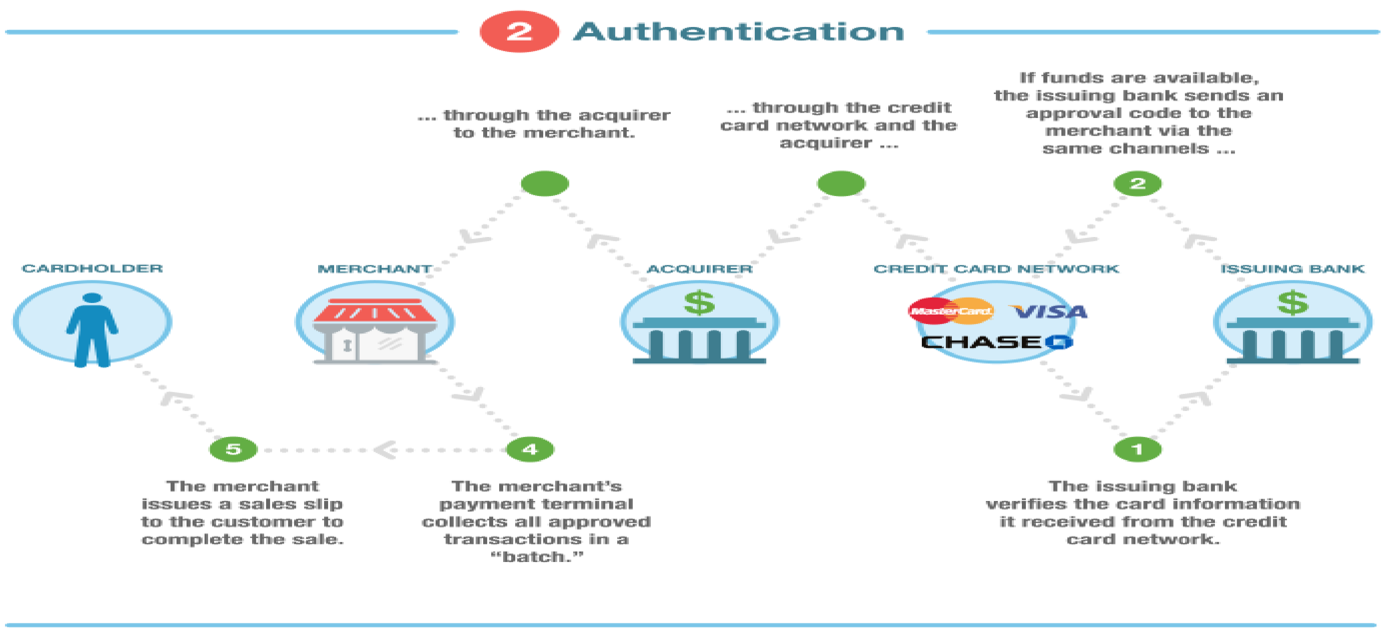
Contents

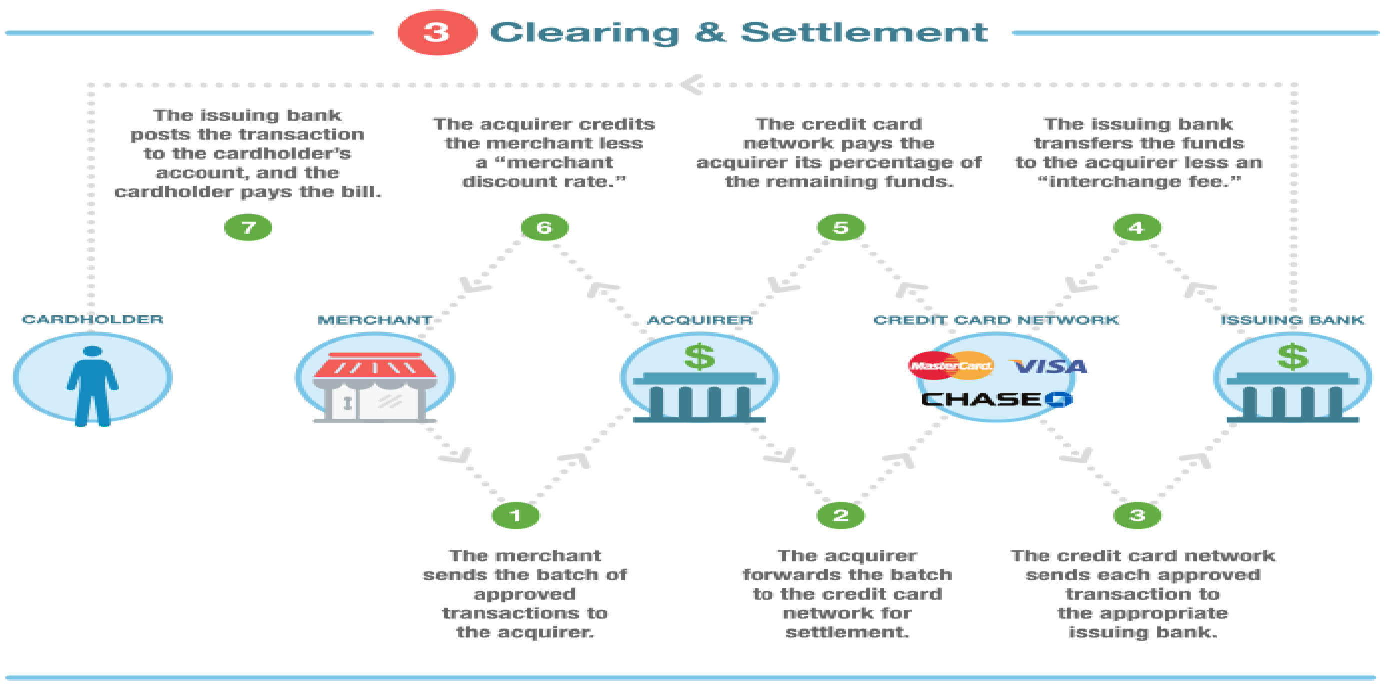
1. ABSTRACT:

Due to a rapid advancement in the electronic commerce technology, the use of credit cards has dramatically increased. As credit card becomes the most popular mode of payment for both online as well as regular purchase, cases of fraud associated with it are also rising. In this project, we studied how to model the sequence of operations in credit card transaction processing using a Hidden Markov Model (HMM) and showed how it can be used for the detection of frauds. An HMM is initially trained with the normal behaviour of a cardholder. If an incoming credit card transaction is not accepted by the trained HMM with sufficiently high probability, it is considered to be fraudulent. At the same time, we try to ensure that genuine transactions are not rejected.

2. A TYPICAL CREDIT CARD TRANSACTION PROCESS:


1 Authorization
CARDHOLDER
MERCHANT
ACQUIRER
CREDIT CARD NETWORK
ISSUING BANK
MasterCard
VISA
CHASE
1 The cardholder pays the merchant for a purchase with a credit card.
2 The merchant send the credit card details to the acquirer.
3 The acquirer forwards the credit card details to the credit card network.
4 The credit card network requests payment authorization from the issuing bank.


2 Authentication
... through the acquirer to the merchant.
... through the credit card network and the acquirer ...
If funds are available, the issuing bank sends an approval code to the merchant via the same channels ...
2
CARDHOLDER
MERCHANT
ACQUIRER
CREDIT CARD NETWORK
ISSUING BANK
MasterCard
VISA
CHASE
5 The merchant issues a sales slip to the customer to complete the sale.
4 The merchant's payment terminal collects all approved transactions in a "batch."
1 The issuing bank verifies the card information it received from the credit card network.


3 Clearing & Settlement
The issuing bank posts the transaction to the cardholder's account, and the cardholder pays the bill.
The acquirer credits the merchant less a "merchant discount rate."
The credit card network pays the acquirer its percentage of the remaining funds.
The issuing bank transfers the funds to the acquirer less an "interchange fee."
7
6
5
4
CARDHOLDER
MERCHANT
ACQUIRER
CREDIT CARD NETWORK
ISSUING BANK
1
2
3
The merchant sends the batch of approved transactions to the acquirer.
The acquirer forwards the batch to the credit card network for settlement.
The credit card network sends each approved transaction to the appropriate issuing bank.

3. INTRODUCTION TO HIDDEN MARKOV MODELS:

A Markov chain is useful for computing a probability for a sequence of events that we can observe in the world. But in many cases, the events we are interested in may not be directly observable in the world. A Hidden Markov Model (HMM) talks about both observed events and hidden events that we think of as causal factors in our probabilistic model. (Daniel Jurafsky & James H. Martin (2013))

Hidden Markov Model (**HMM**) is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved (*i.e. hidden*) states. Hidden Markov models are especially known for their application in reinforcement learning and temporal pattern recognition such as speech, handwriting, gesture recognition, part-of-speech tagging, musical score following, partial discharges and bioinformatics.(Hidden Markov model (2018))

FORMAL DEFINITION:

A HMM is specified by the following components: $W = \omega 1, \omega 2 \ldots \omega N$

a set of N hidden states.

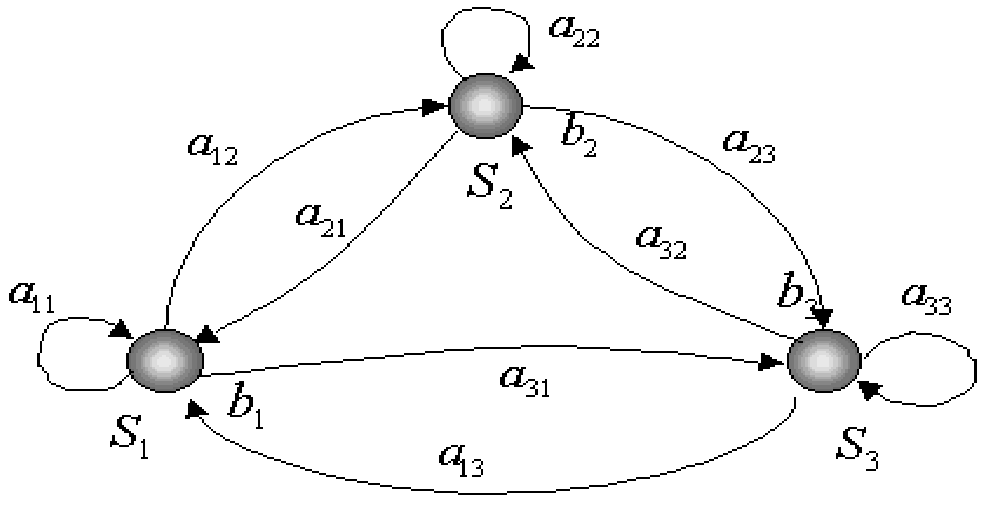
$V = v1; v2; \ldots\ldots vT$ a sequence of T observations (observed states).

A = {aij : ωi to ωj } **transition probability** matrix A is the probability of moving from state i to state j, such that $\sum_{j=1}^{n} aij = 1$ for all i.

B = {bjk} a sequence of observation likelihoods, also called **emission probabilities**, each expressing the probability of an observation v_k being generated from a state ω_j such that

$$\sum_{k=1}^{T} b_{ij} = 1 \text{ for all J.}$$

Absorbing state- ω_0 is the final state where the machine ends up and it emits only 1 signal v_0 from this state.



$$O = \{ Y, G, R, G, G, B, R, R, Y, B \}$$

$$S = \{ 2, 1, 1, 3, 2, 2, 2, 3, 3, 1 \}$$

-An Example of HMM

4. CENTRAL ISSUES IN HMM

Now, given such a hidden Markov model you find that in hidden Markov models, there are three central issues which need to be addressed.

- Evaluation Problem or Likelihood Problem
- Decoding Problem
- Learning Problem

A. EVALUATION PROBLEM

Our first problem is to compute the likelihood of a particular observation sequence.

Computing Likelihood: Given an **HMM θ= (W, V, aij, bjk)** and an observation sequence V^T, determine the likelihood $P(V^T \mid \theta)$ or what is the probability that this sequence V^T is generated by θ.

This can be done by a Crude approach and a better Recursive algorithm. But first we will discuss the crude approach.

In the Crude Approach we find the required probability using the following formula-

$$P(V^T|\theta) = \sum_{r=1}^{rmax} P(V^T|\omega_r^T).P(\omega_r^T)$$

Where ω_r^T is one of the possible sequences of hidden states and it is given by-

$\omega_r^T = \{\omega(1), \omega(2) \dots\dots\dots\dots\dots\dots \omega(T)$ and t=1,2…. T are the discrete time steps.

Now if there are N number of hidden states, then maximum number of possible sequences will be of the order of N^T. In order to calculate the probability of a particular sequence, we use

$$P(\omega_r^T) = \pi_{t=1}^T P(\omega(t)|\omega(t-1))$$

Given that we know a particular sequence has occurred, the probability of getting the observed pattern is found by-

$$P(V^T|\omega_r^T) = \pi_{t=1}^T P(V(t)|\omega(t))$$

So the final formula boils down to

$$P(V^T|\theta) = \sum_{r=1}^{rmax} \prod_{t=1}^{T} P(V(t)|\omega(t))P(\omega(t)|\omega(t-1))$$

But the order of complexity of this process is $N^T * T$ which is huge. This is why we move onto the Forward Algorithm.

THE FORWARD ALGORITHM

In the forward algorithm we must know the value of $\alpha_j(t)$ which is the probability of being in state j after seeing the first t observations, given the HMM θ. The probability distribution of $\alpha_j(t)$ is given by-

$\alpha_j(t) = 0$ t = 0 & j ≠ initial state

= 1 t = 0 & j = initial state

= $[\sum_{i=1}^{N} \alpha_i(t-1)a_{ij}]b_{(jkv(t))}$]otherwise

$\alpha_i(t-1)$ is the previous forward path probability from the previous time step a_{ij} the transition

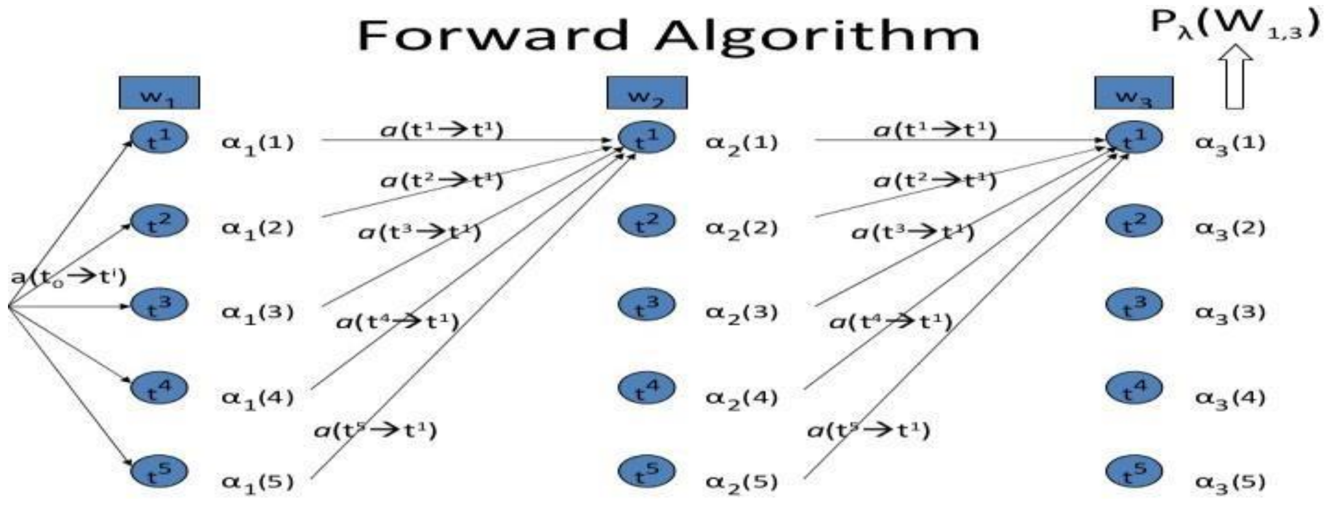
probability from previous state ω_i to current state ω_j

$b_{(jkv(t))}$ the state observation likelihood of the observation symbol v(t) given the current state j. The recursive

algorithm is-

1. Initialization: At time step t=0, initialize the values of a_{ij} and b_{jk}.We also know the sequence V_T and the initial state $\alpha_j(0)$.
2. Increment: For t←t+1

   $\alpha_j(t) = b_{(jkv(t))} [\sum_{i=1}^{N} \alpha_i(t-1)a_{ij}]$ until t=T.
3. Return $P(V^T|\theta)$ which is nothing but α_0(T).
4. End



B. LEARNING PROBLEM- THE FORWARD BACKWARD ALGORITHM

We turn to the third problem for HMMs: learning the parameters of an HMM, that is, the A={a_{ij}} and B={b_{jk}} matrices. Formally-

Learning: Given a large number of observation sequences V^T and the set of possible states in the HMM, we need to learn the HMM parameters A and B. The input to such a learning algorithm would be an unlabeled sequence of observations V and a vocabulary of potential hidden states.

The standard algorithm for HMM training is the forward-backward, or Baum Welch algorithm. The algorithm will let us train both the transition probabilities A and the emission probabilities B of the HMM. But before that we need to know about **backward probability**.

βi(t) = The probability that the model will be in state $\omega_i(t)$ and will generate remaining of the given target sequence V^T i.e from v(t+1) to v(T).

The probability distribution of β_i (t) is given by-

β_i (t) = 0 $\omega_i(t) \neq \omega_0$ and t=T

= 1 $\omega_i(t) = \omega_0$ and t=T

=$[\sum_{j=1}^{N} \beta_i(t+1) a_{ij} b_{(jkv(t))}]$ otherwise

otherwise

Now in order to compute $\beta_i$we perform the following algorithm.

1. Initialize β_ii(t) a_{io} ; $1 \leq i \leq N$
2. Recursion : For t←t-1

$$\beta_i(t) = [\sum_{j=1}^{N} \beta_i(t+1) a_{ij} b_{(jkv(t))}]$$

Until t=1

3. Return $P(V^T|\theta)$ which is nothing but$\beta_i(0)$.
4. End

Forward Backward Algorithm:

Having computed the forward and backward probabilities we need to use them to build an algorithm to estimate the transition probability and emission probability matrix A and B.

1. For this we define

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1)a_{ij}b_{jk}\beta_j(t)}{P(V^T|\theta)}$$

In this expression the numerator term gives us the probability of transition from $\omega_i(t\text{-}1)$ to $\omega_j(t)$ for a particular sequence V^T. While the denominator term gives the probability of getting the sequence V^T in all possible ways.

2. Initially we should choose the values of a_{ij} and b_{jk} arbitrarily.
3. Expected number of transitions from $\omega_i(t\text{-}1)$ to $\omega_j(t)$ at any time in the sequence V^T is

given by $\sum_{t=1}^{T}\gamma_{ij}(t)$.

4. Total expected number of transitions from ω_i is given by

$$\sum_{t=1}^{T}\sum_{k}\gamma_{ij}(t)$$

5. The estimated value of a_{ij} is $\hat{a}_{ij}$ = (expected number of transitions from state i to state j) divided by the (expected number of transitions from state i)

$$a_{ij} = \frac{\sum_{t=1}^{T}\gamma_{ij}(t)}{\sum_{t=1}^{T}\sum_{k}\gamma_{ij}(t)}$$

6. The estimated value of b_{jk} is $\hat{b}_{jk}$ = (expected number of times in state j and observing symbol v_k) divided by (expected number of times in state j).

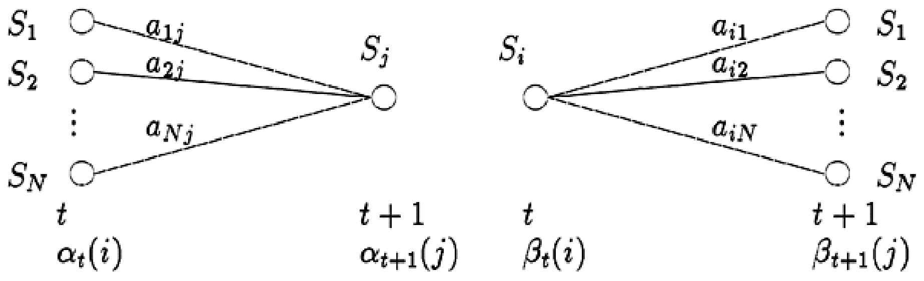
$$b_{ij} = \frac{\sum_{t=1}^{T}\sum_{l}\gamma_{jl}(t)}{\sum_{t=1}^{T}\sum_{l}\gamma_{jl}(t)}$$

only for those states where $v(t)=v_k$

7. Now we need to use the estimated values $\hat{a}_{ij}$ and $\hat{b}_{jk}$ to calculate $\alpha_i(t-1)$, $\beta_j(t-1)$

and $\gamma_{jl}(t-1)$.

8. Again, we will use these values to re-estimate the parameters $\hat{a}_{ij}$ and b_{jk}. This process should be repeated until convergence or at least when the values are below a specified threshold.



(a) (b)

Although in principle the forward-backward algorithm can do completely unsupervised learning of the A and B parameters, in practice the initial conditions are very important. For this reason, the algorithm is often given extra information. For example, for speech recognition, in practice the HMM structure is often set by hand, and only the emission (B) and (non-zero) A transition probabilities are trained from a set of observation sequence.

5. USE OF HMM FOR CREDIT CARD FRAUD DETECTION:

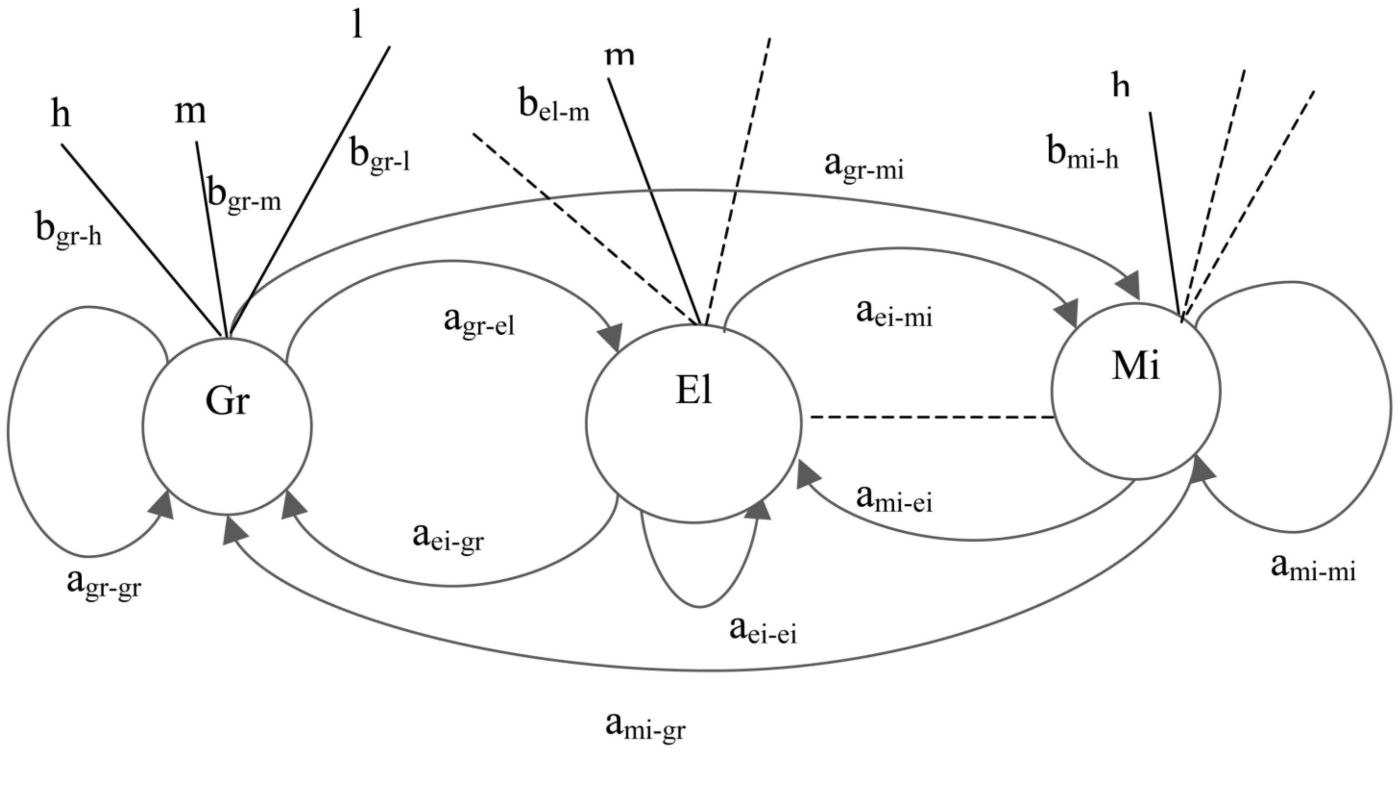
An FDS runs at a credit card issuing bank. Each incoming transaction is submitted to the FDS for verification. FDS receives the card details and the value of purchase to verify whether the transaction is genuine or not. The types of goods that are bought in that transaction are not known to the FDS.

To map the credit card transaction processing operation in terms of an HMM, we start by first deciding the observation symbols in our model.

Observable States: We quantize the purchase values which are known to FDS into M price ranges V1; V2; . . . VM, forming the observation symbols at the issuing bank. The actual price range for each symbol is configurable based on the spending habit of individual cardholders. These price ranges can be determined dynamically by applying a clustering algorithm on the values of each cardholder's transactions.

Hidden States: Cardholder makes purchases depending on his need for procuring different types of items over a period of time. This, in turn, generates a sequence of transaction amounts. Each individual transaction amount usually depends on the corresponding type of purchase. Hence, we consider the transition in the type of purchase as state transition in our model. The type of each purchase is linked to the line of business of the corresponding merchant. This **information about the merchant's line of business is not known to the issuing bank running the FDS**. Thus, the type of purchase of the cardholder is hidden from the FDS. The set of all possible types of purchase and, equivalently, the set of all possible lines of business of merchants forms the set of hidden states of the HMM.

It should be noted at this stage that the line of business of the **merchant is known to the acquiring bank**, since this information is furnished at the time of registration of a merchant. Also, some merchants may be dealing in various types of commodities (For example, Wal-Mart, Big-Bazaar, Reliance Mart sells tens of thousands of different items). Such types of line of business are considered as Miscellaneous, and we do not attempt to determine the actual types of items purchased in these transactions. Any assumption about availability of this information with the issuing bank and, hence, with the FDS, is not practical and, therefore, would not have been valid.



After deciding the state and symbol representations, the next step is to determine the probability matrices A, B, and Π so that representation of the HMM is complete. These three model parameters are determined in a training phase using the Baum-Welch algorithm. The initial choice of parameters affects the performance of this algorithm and, hence, they should be chosen carefully.

In this fraud detection system, we consider three different spending profiles of the card holder which is depending upon price range, named high (h), medium (m) and low (l). In this set of symbols, we define V = {l, m, h} and M =3. After finalizing the state using clustering in train data, the next step is to determine different components of the HMM for that user, i.e. the probability matrices A, B and initial distribution π are required for the HMM to be known. These model parameters are determined in a training phase using the forward-backward algorithm. The initial choice of parameters affects the performance of this algorithm and, hence, it is necessary to choose all these parameters carefully, which is done by analysing cardholder's profile. We consider the special case of fully connected HMM in which every state of the model can be reached in a single step from every other state, as shown in Fig. Gr, El, Mi, etc., are names given to the states to denote purchase types like Groceries, Electronic items, and Miscellaneous purchases.

Spending Profile of Card holder:
The spending profile of a cardholder suggests his normal spending behaviour. Cardholders can be broadly categorized into three groups based on their spending habits, namely, high-spending (hs) group, medium-spending (ms) group, and low-spending (ls) group. Cardholders who belong to the hs group, normally use their credit cards for buying high-priced items. Similar definition applies to the other two categories also. Spending profiles of cardholders are determined at the end of the clustering step. Let p_i be the percentage of total number of transactions of the cardholder that belong to cluster with mean ci which is of the high values transactions, then cardholder is said to belong to in high-spending group. Thus, spending profile denotes the cluster number to which most of the transactions of the cardholder belong. These categories help in choosing the initial parameters while training.

Model Parameters estimation and Training:
We use Baum-Welch algorithm to estimate the HMM parameters for each cardholder. The algorithm starts with an initial estimate of HMM parameters A, B, and π, and converges to the nearest local maximum of the likelihood function. Initial state probability distribution is considered to be uniform, that is, if there are N states, then the initial probability of each state is 1/N. Initial guess of transition and observation probability distributions can also be considered to be uniform. However, to make the initial guess of observation symbol probabilities more accurate, spending profile of the cardholder, as determined earlier, is taken into account. We make three sets of initial probability for observation symbol generation for three spending groups—ls, ms, and hs. Based on the cardholder's spending profile, we choose the corresponding set of initial observation probabilities. The initial estimate of symbol generation probabilities using this method leads to accurate learning of the model. Since there is no a priori knowledge about the state transition probabilities, we consider the initial guesses to be uniform.

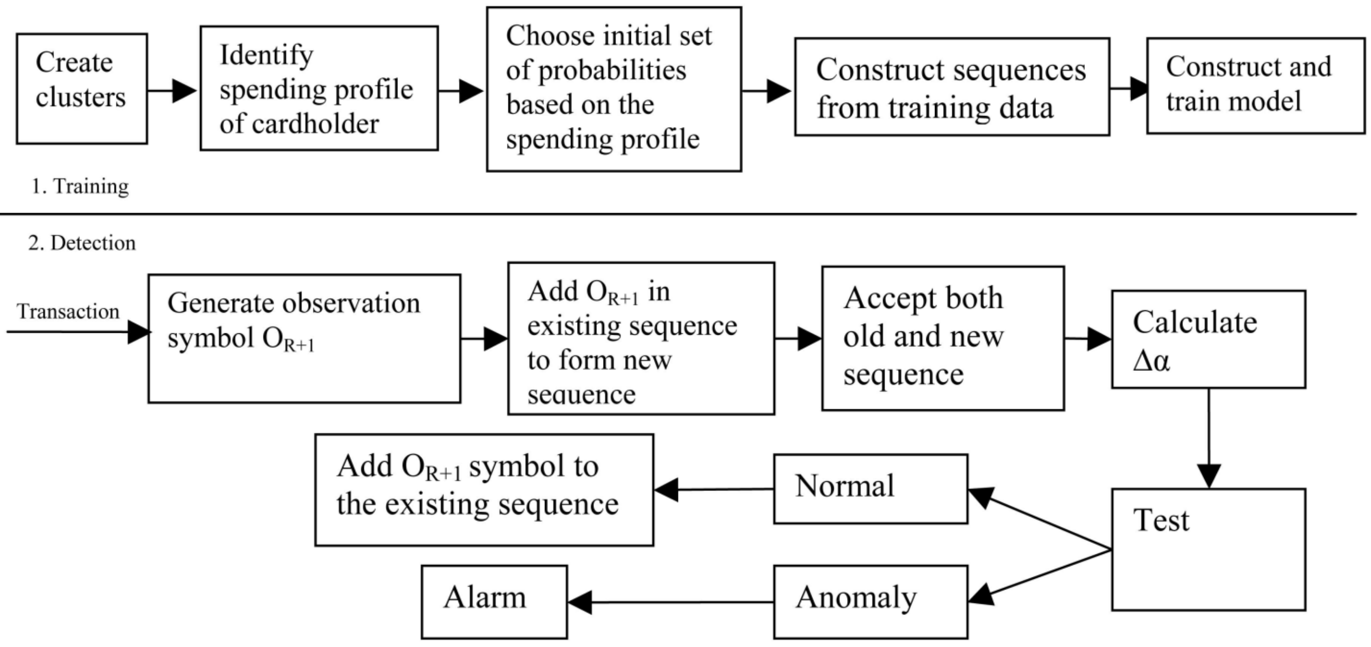
We now start training the HMM. The training algorithm has the following steps: 1) initialization of HMM parameters, 2)forward- backward procedure, for training the HMM and 3) forward procedure to predict the probability of upcoming observation to be in the same HMM. we convert the cardholder's transaction amount into observation symbols and form sequences out of them. At the end of the training phase, we get an HMM corresponding to each cardholder. Since this step is done offline, it does not affect the credit card transaction processing performance, which needs online response.

Fraud detection:
After the HMM parameters are learned, we take the symbols from a cardholder's training data and form an initial sequence of symbols. Let $O_1, O_2, \ldots O_r$ be one such sequence of length R. This recorded sequence is formed from the cardholder's transactions up to time t. We input this sequence to the HMM and compute the probability of acceptance by the HMM. Let the probability be α1, which can be written as follows:
$\boldsymbol{\alpha_1 = P(O_1, O_2, O_3, \ldots O_R \mid \lambda)}$
Let O_{R+1} be the symbol generated by a new transaction at time t+1. To form another sequence of length R, we drop O1 and append OR+1 in that sequence, generating $O_2, O_3, O_4 \ldots O_{R+1}$ as the new sequence. We input this new sequence to the HMM and calculate the probability of acceptance by the HMM. Let the new probability be α2
$\boldsymbol{\alpha_2 = P(O_2, O_3, O_4 \ldots O_{R+1} \mid \lambda)}$

If $\Delta \alpha = \alpha_1 - \alpha_2 > 0$, it means that the new sequence is accepted by the HMM with low probability, and it could be a fraud. The newly added transaction is determined to be fraudulent if the percentage change in the probability is above a threshold, that is,

$\Delta \alpha / \alpha_1 \geq$ Threshold.

The threshold value can be learned empirically, If O_{R+1} is malicious, the issuing bank does not approve the transaction, and the FDS discards the symbol. Otherwise, O_{R+1} is added in the sequence permanently, and the new sequence is used as the base sequence for determining the validity of the next transaction. The reason for including new no malicious symbols in the sequence is to capture the changing spending behaviour of a cardholder. Fig below shows the complete process flow of the proposed FDS. As shown in the figure, the FDS is divided into two parts—one is the training module, and the other is detection. Training phase is performed offline, whereas detection is an online process.


Create clusters
Identify spending profile of cardholder
Choose initial set of probabilities based on the spending profile
Construct sequences from training data
Construct and train model
1. Training
2. Detection
Transaction
Generate observation symbol O_{R+1}
Add O_{R+1} in existing sequence to form new sequence
Accept both old and new sequence
Calculate $\Delta\alpha$
Test
Normal
Anomaly
Add O_{R+1} symbol to the existing sequence
Alarm

6. Coded Example:

To understand this better, we simulated a dataset which can be viewed as person's history of credit card transactions. We made test set with 69 normal transactions and test set with 32 normal+ abnormal transaction. Each transaction has type of purchase and purchase amount available in data.

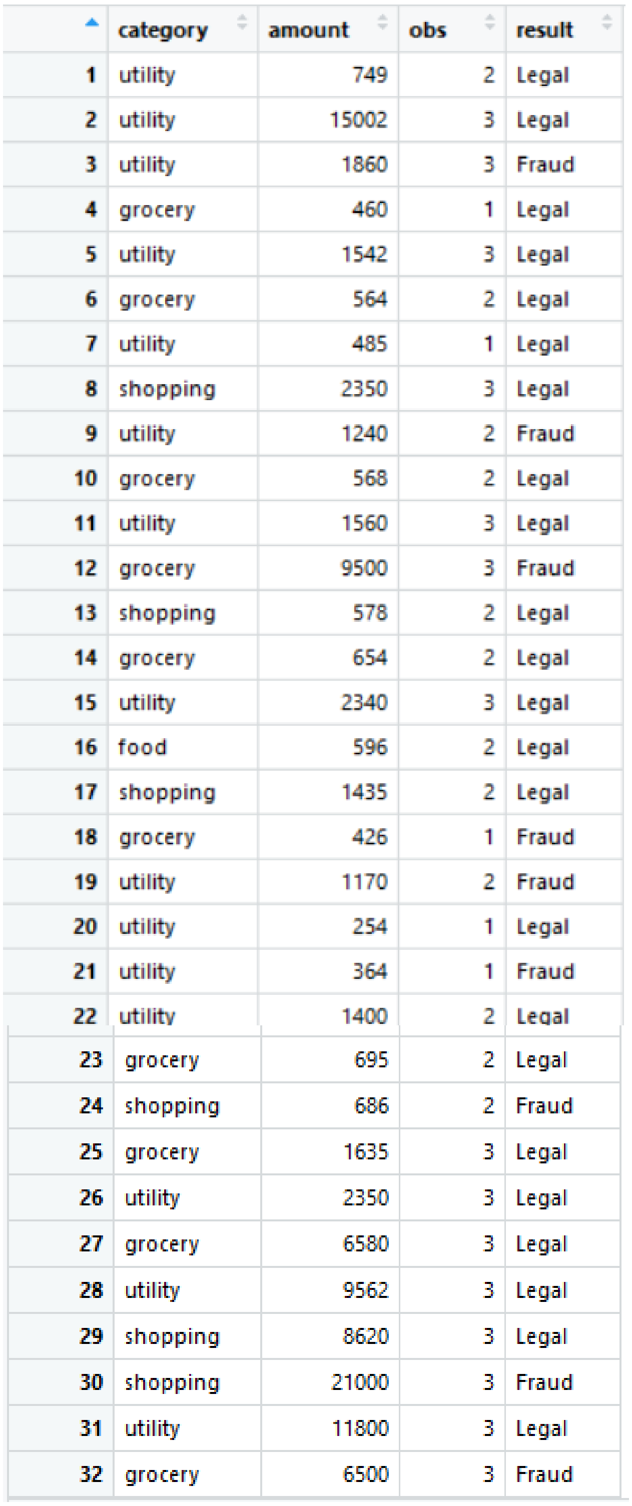
Approach:

First we clustered the transaction amounts into 3 clusters. (1, 2, 3 shows low, high, Medium). Now on the training set we trained Baum-Welch model with initial probability (4/7, 2/7, 1/7) of category 1,2 and 3 respectively. Baum-Welch algorithm is an iterative method which may converge to local minimum which totally depends on initial distribution matrices This is the reason we will first find spending profiles of that particular card user (ls, ms and hs) and then we will use initial parameters in accordance with that. (based on the other similar data and experience)

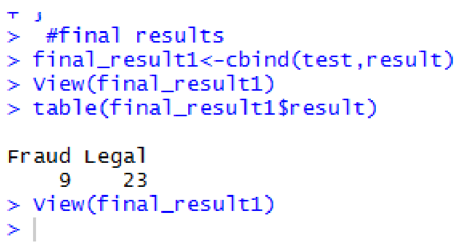
So after training the data on train data (estimating state transaction probability matrix and observation probability matrix), we went ahead with test dataset to predict whether each incoming transaction is fraud or not. As we discussed in theory part, suppose for incoming 70th transaction, (which is first transaction of test set) we predict **$\alpha_1 = P(O_1, O_2, O_3, \dots O_{69} | \lambda)$** and **$\alpha_2 = P(O_2, O_3, O_4, \dots O_{70} | \lambda)$** and compare **$|\Delta \alpha| / \alpha_1$** with threshold value of 10%. If the change is more than 10%, then we declare this transaction as fraud transaction.

Results:

Test Data Results:

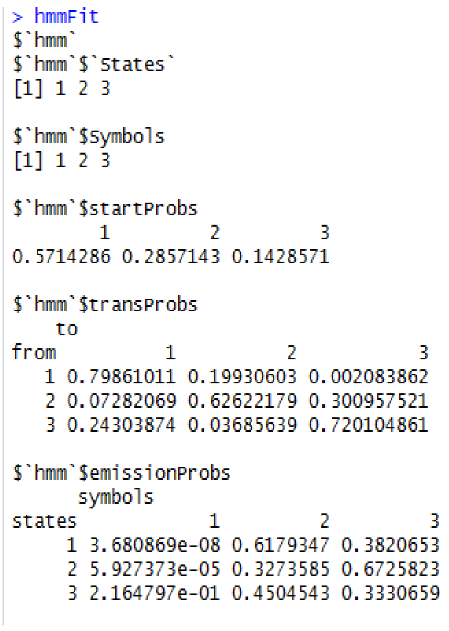


	category	amount	obs	result
1	utility	749	2	Legal
2	utility	15002	3	Legal
3	utility	1860	3	Fraud
4	grocery	460	1	Legal
5	utility	1542	3	Legal
6	grocery	564	2	Legal
7	utility	485	1	Legal
8	shopping	2350	3	Legal
9	utility	1240	2	Fraud
10	grocery	568	2	Legal
11	utility	1560	3	Legal
12	grocery	9500	3	Fraud
13	shopping	578	2	Legal
14	grocery	654	2	Legal
15	utility	2340	3	Legal
16	food	596	2	Legal
17	shopping	1435	2	Legal
18	grocery	426	1	Fraud
19	utility	1170	2	Fraud
20	utility	254	1	Legal
21	utility	364	1	Fraud
22	utility	1400	2	Legal
23	grocery	695	2	Legal
24	shopping	686	2	Fraud
25	grocery	1635	3	Legal
26	utility	2350	3	Legal
27	grocery	6580	3	Legal
28	utility	9562	3	Legal
29	shopping	8620	3	Legal
30	shopping	21000	3	Fraud
31	utility	11800	3	Legal
32	grocery	6500	3	Fraud



```
>  #final results
> final_result1<-cbind(test,result)
> View(final_result1)
> table(final_result1$result)

Fraud Legal
    9    23
> View(final_result1)
> 
```



```
> hmmFit
$`hmm`
$`hmm`$`States`
[1] 1 2 3

$`hmm`$Symbols
[1] 1 2 3

$`hmm`$startProbs
        1         2         3
0.5714286 0.2857143 0.1428571

$`hmm`$transProbs
    to
from          1          2           3
   1 0.79861011 0.19930603 0.002083862
   2 0.07282069 0.62622179 0.300957521
   3 0.24303874 0.03685639 0.720104861

$`hmm`$emissionProbs
      symbols
states            1         2         3
     1 3.680869e-08 0.6179347 0.3820653
     2 5.927373e-05 0.3273585 0.6725823
     3 2.164797e-01 0.4504543 0.3330659
```

Conclusion:

In this project, we learned, How Hidden Markov Models are useful to detect fraudulent online transaction through credit card. The proposed Fraud Detection System is also scalable for handling vast volumes of transactions data processing. It is very simple and practical system with genuine and fast result given the initial parameters are chosen carefully. It also doesn’t require labelled data for training purpose as in supervised algorithms. The Hidden Markov Model makes the processing of detection easy and tries to remove the complexity.

7. R Code:

```
#Loading data
train <- read.csv("C:/Users/DANISH/Desktop/credit1.csv")

test <- read.csv("C:/Users/DANISH/Desktop/test.csv")
# training the model
states = c(1,2,3)
symbols = c(1,2,3)

library(depmixS4)
library(HMM)
hmmInit = initHMM(states, symbols , c(4/7,2/7,1/7))
hmmFit = baumWelch(hmmInit, train$obs)

#Testing the Model
result<-c()


combined<-rbind(train,test)

hmm_model<-(hmmFit$hmm)

#If % changes in oberving the sequence is greater than
#specied threshold then it is a fraud tranaction

for(i in 1:32){
logForwardProbabilities = forward(hmm_model,c(as.character(combined$obs[i:(i+68)])))
probO1<-data.frame(exp(logForwardProbabilities))
##print(exp(logForwardProbabilities))

alpha1<-sum(probO1$X10)

logForwardProbabilities = forward(hmm_model,c(as.character(combined$obs[(i+1):(i+69)])))
probO2<-data.frame(exp(logForwardProbabilities))
##print(exp(logForwardProbabilities))

alpha2<-sum(probO2$X10)


# creating a vector to store results
{
if(((alpha2-alpha1)*100/alpha1)<10)
  result<-c(result,"Legal")
else
  result<-c(result,"Fraud")
}



}
 #final results
final_result1<-cbind(test,result)
```

9. Books and References:

- Pattern Recognition and Application by Prof. P.K. Biswas,(NPTEL Video Lecture series on pattern recognition (https://www.youtube.com/watch?v=cjlhpaDXihE)
- Probability Models by Sheldon Ross
- pattern recognition and machine learning Christopher M Bishop
- http://proceedings.mlr.press/v37/goernitz15.pdf
- https://hmmlearn.readthedocs.io/en/latest/tutorial.html#training-hmm-parameters-and-inferring-the-hidden-states
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.428.9363&rep=rep1&type=pdf
- http://www.ijarcst.com/doc/vol5issue1/divya.pdf
- https://www.ijert.org/research/a-predictive-approach-for-fraud-detection-using-hidden-markov-model-IJERTV2IS1304.pdf
- https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6493206
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.428.9363&rep=rep1&type=pdf