Hidden Markov Models for churn prediction

***Abstract*—Most companies favour the creation and nurturing of long-term relationships with customers because retaining customers is more profitable than acquiring new ones. Churn prediction is a predictive analytics technique to identify churning customers ahead of their departure and enable customer relationship managers to take action to keep them. This work evaluates the development of an expert system for churn prediction and prevention using a Hidden Markov model (HMM). A HMM is implemented on unique data from a mobile application and its predictive performance is compared to other algorithms that are commonly used for churn prediction: Logistic Regression, Neural Network and Support Vector Machine. Predictive performance of the HMM is not outperformed by the other algorithms. HMM has substantial advantages for use in expert systems though due to low storage and computational requirements and output of highly relevant customer motivational states. Generic session data of the mobile app is used to train and test the models which makes the system very easy to deploy and the findings applicable to the whole ecosystem of mobile apps distributed in Apple's App and Google's Play Store.**

Keywords*—*churn prediction; Hidden Markov Model; expert systems

I. Introduction

Nowadays, marketplaces in which companies evolve are often marked by a high competition and saturation. This leads companies to favour creation and nurturing of long-term relationships with customers. Indeed, the only source for new customers in a saturated and competitive market would be customers churning from competition.

Keeping existing customers is more profitable than attracting new ones, because new customers have a higher attrition rate [1]. Moreover, the cost to make a sell to a new customer compared to an additional sale to an existing customer is up to 5 times more [2] and retained customers create higher revenues and margin than new customers [3]. In addition, even an increase of only 1% in customer retention increases revenue significantly [4]. These findings are applicable to mobile applications, where customer acquisition costs are consistently increasing [5]. For these reasons, and due to the ever increasing availability of customer data, companies are choosing Customer Relationship Management (CRM) as their customer-centric marketing strategy.

In light of limited resources to address every customer individually, a technique to detect customers which need most care promises to be of great value to industry. These customers can be described as persons who are likely to stop interacting with the company, and hence churn. A part of CRM which focuses on retaining customers is churn prediction, i.e predicting if customers are about to depart. Once made available through an expert system, this information can be used by non-expert users. In the present case marketing and customer support teams can address targeted clients in order to increase their satisfaction and incentivise them to remain with the company. Concretely, this is mostly done by giving out gifts, offering significant discounts or - in the case of games - providing help to overcome a hurdle.

A closer look at the mobile app market shows that it is a highly competitive marketplace, with a large number of available apps in the App and Play Store. For the US and Apple's App Store alone, there are about 279'000 available games on October 2014 [6]. However, there is still room for new users as not every person has a smartphone yet [7], hence the market is not saturated. Nevertheless, the possibility of keeping existing customers and getting new ones at the same time can yield a key strategic advantage.

For this purpose, this research develops an expert system for churn prediction and prevention at one of the worlds leading mobile game developers. The system is built for a specific game in the companys portfolio using generic information, precisely the number of daily uses of the app by a customer. Hence, the system is not only transferrable to other games in the companys portfolio, but to apps in general. The expert system is developed using both commonly used algorithms, Logistic Regression, Neural Network and Support Vector Machine [8], [9], [10], [11], [12], and by exploring the application of Hidden Markov Models (HMMs). All approaches are benchmarked and advantages of using HMMs in building expert systems for churn prevention are outlined.

The paper is structured as follows. In Section II the target customers, the churning customers and the prediction problem are defined mathematically. Section III gives the theory necessary to understand the algorithms. In Section IV several experiments are performed with HMM that allow to model the intrinsic motivation of customers, the link between motivation and churn and the transitional behaviour between motivational states for customers. Furthermore, a new way to predict churn with HMM is introduced and benchmarked against commonly used algorithms. Finally, findings are summarised in Section V and an outlook for future work is given in Section VI.

II. Problem Formulation

In practice, the prediction model is applied once a day, in order to predict if the customers active in the last 24 hours are going to churn in the near future. This Section defines active customers, churning customers and finally the prediction problem itself.

A. Definition: Active Customers

The set of active customers is defined as in Definition II.1, based on which a set of about 246'000 customers that were all active on the same day is created. A customer is described by features as in Definition II.2, more specifically in this work only generic connection data of customers is used.

Each time a customer connects to the application (app), a "session start call" is stored in the database of the company. Data is represented as a weekly moving average, each feature represents the average number of connections over a week, these are called "activity features", and 14 of these are used to describe each customer.

Definition II.1. The set of active customers is defined as $A = \{P | S > 0, t_i \in [T - \Delta t, T]\}$. With P the customers of the app, S the number of session start calls[1] that are sent at timestamp t_i, with i = 1,2,...,n. T is the prediction time and $\Delta t = 24hours$.

Definition II.2. An active customer $a_i \in A$ with i = 1,2,...n, is defined by a set of m features with a vector $\boldsymbol{a}_i = (f_1, f_2, ..., f_m)$, where f_j is the value of the j^{th} feature, with j=1,2,...,m. Thereby, the set of active customers A can be written in matrix form as $\boldsymbol{A^T} = (\, \boldsymbol{a}_1^T \boldsymbol{a}_2^T ... \boldsymbol{a}_n^T \,)$.

B. Definition: churning customers

Depending on the relationship between the company and the customer, the definition of churn varies. For example, in the telecommunication industry a customer is often bound by a contract to the company, and hence churn can simply be defined as not pursuing this contract, i.e. not using the product anymore. However, this is not as simple in a freemium game environment, where an alternative definition has to be found.

Churn is seen as a lost in interest for a game, which is defined as not using the product for a defined period of time, after which most customers do not return. The threshold value for the absence time after which a customer is considered to be a churner is chosen to be 14 days, as in [13]. This is based on the hypothesis that the behaviour of customers is similar in the games studied in both works.

Furthermore, proactive retention actions require to identify churners before they leave the game. Based on this, a customer is considered to be a churner if he leaves the game in a time span of 6 days after prediction time. Thereafter, churn for customers in set A is defined as in Definition II.3, and illustrate it in Fig. 1.

Definition II.3. The set of churners in dataset A is defined as $C = \{A | t_{lastsession} \in [T, T + \Delta t]\}$. With A the set of active customers, $t_{lastsession}$ is the time at which a customer did his last session end call[2] in the app before starting a 14 days absence period, T the time at which the prediction is run, and $\Delta t = 6[days]$.

C. Binary Prediction Problem

Based on Definitions II.1, II.2 and II.3, a label "churn" is created for each element $a_i \in A$, where $churn = 1$ if $a_i \in C$ and $churn = 0$ if $a_i \notin C$.

The churn prediction problem can now be defined as a binary prediction problem, where the goal is to predict the label churn of each element $a_i \in A$, based on f_j the value of the customers' features, with j=1,2,...,m.

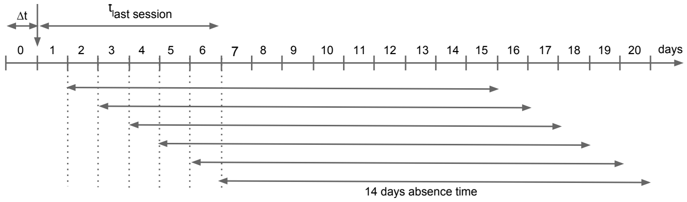

Fig. 1: Illustration of Definition II.3. Set A from II.1 includes customers from $\Delta t = 24hours$ before the prediction time. Churners in set A, are the customers for which $t_{lastsession}$ before starting a period of 14 days absence, is in the range defined between days 1 and 6. This range is used in order to enable proactive actions for churning customers.

III. Theory

This Section gives the necessary theory to understand the algorithms used in this work.

A. Hidden Markov Model

A Hidden Markov Model (HMM) is a statistical model that can be used to describe real-world processes with observable output signals. HMMs have an underlying stochastic process formed by a Markov chain that is not observable (hidden). For each hidden state a stochastic model creates observable output signals or observations, based on which hidden states can be estimated. The following theory is based on [14], [15].

The underlying Markov chain is a discrete-time random process, taking values in the state space $S = \{1, ..., N\}$, such that $S_t \in S$ at each time step $t = 1, 2, ..., T$. Furthermore, the process S_t is a first order Markov process if it has the Markov property from 1, for any $s_1, s_2, ..., s_{t+1} \in \{1, ..., N\}$. Which means that the state at any given time $t + 1$ depends only on the state at time t.

$$P(S_{t+1} = s_{t+1} | S_1 = s_1, S_2 = s_2, ..., S_t = s_t) = \\ P(S_{t+1} = s_{t+1} | S_t = s_t) \qquad (1)$$

The probability of a transition from state i at time t to state j at time $t + 1$, can be written $g_{ij} = P(S_{t+1} = j | S_t = i)$ with the properties $\sum_{j=1}^{N} g_{ij} = 1$ and $g_{ij} \geq 0$. This results in the transition matrix $\mathbf{G} = (g_{ij})$ of the Markov chain. In addition, the vector $\boldsymbol{\pi}$ of initial state probabilities $\pi_i = P(S_1 = i)$ is defined, with π_i corresponding to the probability of starting in state i at time $t = 1$.

Now that the hidden process is defined, the stochastic models that create observations in each hidden state are described. Assume that only a variable X_t can be observed, which at time t generates an observable output signal from a real-world process o_t, that is related to the state S_t but is not the state itself. The conditional distribution of the observed variable X_t given the hidden state S_t is the emission distribution. Hence, for every state in S, an emission distribution b_i as in 2 is defined, which can be written as a vector B of emission distributions b_i. Thus, a HMM is described by a

[1] A "session start" call is sent by the application, when a customer connects to the app.

[2] A "session end" call is sent by the application, when a customer disconnects from the app.

triple $\theta = (\boldsymbol{\pi}, \boldsymbol{G}, \boldsymbol{B})$, and is called a N-state HMM with N the number of states of the underlying Markov chain.

$$b_i(o_t) = P(X_t = o_t | S_t = i) \tag{2}$$

Three types of problems can be solved with a HMM:

1) Evaluation: given the observation sequence $\boldsymbol{O} = (o_1, o_2, ..., o_T)$ and the model $\theta = (\boldsymbol{\pi}, \boldsymbol{G}, \boldsymbol{B})$, the probability of the observation sequence given the model $P(\boldsymbol{O}|\theta)$ can be computed.
2) Decoding: given the observation sequence $\boldsymbol{O} = (o_1, o_2, ..., o_T)$, an optimal state sequence $\boldsymbol{S} = (s_1, s_2, ..s_T)$ can be found.
3) Estimation: given an initial guess of parameters $\theta = (\boldsymbol{\pi}, \boldsymbol{G}, \boldsymbol{B})$, the model parameters can be adjusted to maximise $P(\boldsymbol{O}|\theta)$.

Evaluation is done using the Forward procedure, decoding is done with the Viterbi algorithm which looks for a maximum likelihood state assignment, and estimation is performed with the Expectation-Maximisation (EM) algorithm that looks for parameters that give maximum likelihood. For a detailed explanation of these algorithms the reader is refered to [16], and for more information about HMM in general to [14], [17]. The HMM models are trained with the R package mhsmm described in [15].

B. Logistic Regression

The algorithm is shown in 3, it can take input values in the range $x \in]-\infty, +\infty[$ and has an output limited in the range $F(x) \in [0, 1]$. The input is a linear combination of explanatory variables and regression coefficients: where β_j with $j = 0, 1, ..., m$ are the regression coefficients in vector $\boldsymbol{\beta}$ and x_i with $i = 0, 1, ..., m$ are the explanatory variables in vector $\boldsymbol{x}$. The output can be seen as a probability, but by fixing a cutoff value it can be used for binary classification, e.g. class label ”churn” for an output higher than the cutoff value, and ”no-churn” for an output smaller than the cutoff value. Experiments are performed with the implementation of Logistic Regression from the software Weka [19]. For more details on Logistic Regression, refer to [18].

$$F(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\beta x}}} \tag{3}$$

C. Neural Network

Neural Networks are mathematical models that are organised in layers, composed by interconnected nodes, that can transmit information and learn patterns. The general process is that data is presented to the network through the input layer. Then, the input layer communicates to one or more hidden layers, in which the processing is done with a system of weighted connections and activation functions. The hidden layers are then linked to output nodes where the results are shown. Non-linear links between nodes are created with non-linear activation functions, which are commonly chosen as sigmoid functions as in 3. For this work a fully connected feedforward Neural Network is created, and trained by back-propagation combined with gradient descent.

Advantages of the algorithm are that it can create complex non-linear relations between features, and that it has proven its performance in a large range of applications. Nevertheless, the computational cost is high and it is difficult to interpret results, i.e. the output can not be linked directly to causes. In addition, the large number of parameters that needs to be set empirically, leads to difficulties in obtaining a combination that produces the highest performance. Another drawback concerns the optimisation method by gradient descent, which might get stuck in a local minima. Experiments are performed with the implementation of Neural Networks from the software Weka [19]. For more details on Neural Networks, refer to [20], [21].

D. Support Vector Machine

Support Vector Machine (SVM) are created to solve binary classification problems [22], [23]. SVM looks for an optimal hyperplane, that maximises the margin between two classes, which works well for linearly separable data. This hyperplane is built only on a reduced set of vectors in the dataset that are called support vectors. However, in order to classify non-linearly separable data, the input space can be transformed via a non-linear mapping into a higher dimensional feature space, by using what is called the kernel trick. The radial basis function (RBF) is chosen as kernel, because it has only one parameter γ to fix and it has a bounded output.

Kim et al [24] sum up advantages of SVM: there is only one free parameter in addition to the kernel parameter to be chosen, which is C the upper bound on training errors also named cost parameter. The SVM problem can be written as a linearly constrained quadratic problem, it's solution is therefore unique, optimal and global. Finally, SVM have good generalisation performance, because they are built on the structural risk minimisation (SRM) principle. The experiments for SVM are run with the R package e1071 [25], which is based on libSVM [26]. For more details about SVM the tutorial by Burges is recommended [27].

IV. Experiments

This Section describes the experiments performed to get an expert system for churn prediction.

A. Motivation Modelling

First, the motivation of a customer for using a product is modelled, because churn is defined as a lost in interest in the product. It is assumed that activity can describe motivation, hereby a low activity would represent a low motivation, and a high activity a high motivation.

Modelling motivation of customers with HMM is done by solving the estimation problem with the EM algorithm on time-series of activity. The activity time-series is a sequence of observations $\boldsymbol{O} = (o_1, o_2, ..., o_{T_1})$ with $T_1 = 14$ with each point representing a weekly moving average of activity.

For a customer $p_i \in P$, with P a set of n customers and i = 1,2,...n, a sequence of T_1 observations is defined with a vector $\boldsymbol{O}_i = (o_1, o_2, ..., o_{T_1})$, where o_t is the observation at

TABLE I. AIC SCORES FOR A RANGE OF 2 TO 10 STATES ON SET $\boldsymbol{A}$.

Number of states	*Log-Likelihood*	*Parameters*	*AIC*
2	-9098603	6	18197218
3	-7939013	12	15878050
4	-7190565	20	14381170
5	-6583783	30	13167626
6	-6147067	42	12294218
7	-5819358	56	11638828
8	-5557336	72	11114816
9	-5285993	90	10572166
10	-5117016	110	10234252

the t^{th} time step. Thereby, the customers in set P are written in matrix form as $\boldsymbol{P}' = (\boldsymbol{O}_1, \boldsymbol{O}_2, \ldots, \boldsymbol{O}_n)'$. Additionally, the time-series of observations correspond to the T_1 time steps before the churn prediction time T from Definition 1. For this experiment, the set of active customers $\boldsymbol{A}$ is used.

Further assumptions are:

- Markov property: activity on time $t+1$, depends only the state and observation emitted at time t.
- Observation sequences of customers are mutually independent.
- Each state of the HMM corresponds to a level of motivation of the customer
- A HMM gives the stochastic behaviour for an average customer and all sequences of observations are created by the same HMM,
- The emission distribution b_i of state i, is a Gaussian $b_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ with μ_i the mean, σ_i^2 the variance of the distribution and i=1,2,...,N.

HMMs with 2 to 10 states are fit to the observation sequences. The goodness of fit is measured with the Akaike Information Criterion (AIC) calculated as in 4, with p the number of parameters to estimate in the HMM, and log(L) the log-likelihood of the model explaining the set of sequences. The term p is a penalty term that increases with increasing N and the log-likelihood is a measure of fit of the model that decreases with increasing number of states N. However, as the number of sequences in the set under study is very large, the log-likelihood is significantly larger than the penalty term. Hence, the model is mostly chosen on the fit with the observation sequences. AIC scores are found in Table I, where the HMM with the highest number of states has the best AIC score.

$$AIC = 2p - 2log(L) \tag{4}$$

B. Motivation and Churn

Now that HMMs with states representing motivation of customers are built, a closer look is given to the link between motivation and churn. For this, the sequences from set $\boldsymbol{A}$ are decoded with the Viterbi algorithm on HMMs from the previous Section. Thereafter, each observation of each sequence has an assigned most likely state for each HMM. The last observation in each sequence, corresponds to the time step just before the churn labels are assigned, hence it can be seen in which states churners are before leaving.

The number of customers and the corresponding percentage of churn are shown for each state and HMM in Table II. The parameters of the corresponding emission distributions b_i are in Table III. It can be seen that for each N there is one state that has a significantly higher percentage of churn than other states, ranging from 6.0% at $N = 2$ to 47.0% at $N = 10$. Furthermore, the size of the state with most churn decreases from 153'000 to 4'000 customers. Thereby, the targeting of churners gets more precise with more states. In addition, there is a second state that gets a higher percentage of churn when N increases, ranging from 1.4% at $N = 2$ to 14.6% at $N = 10$. The emission distributions of the states with most churn correspond to the states with the lowest activity, with $\mu_1 = 3.0$ and $\sigma_1^2 = 3.8$ at $N = 2$ and with $\mu_1 = 0.1$ and $\sigma_1^2 = 0.0$ at $N = 10$. Based on these results, it can be said that motivation is related to churn, and that it can be captured by using a HMM on activity time-series.

C. Transition Analysis

This experiment looks at the transitional behaviour of customers, in order to see if previous states of motivation influence the percentage of churn in the current state.

Table IV shows the transition matrix for $N = 10$ of set $\boldsymbol{A}$. For each state the most probable transition is to stay in the same state. The next most probable transitions are to move to the nearest state in terms of motivation, either one state up or down. In addition, state 1 which has the highest percentage of churn, can only be joined from state 2, and this with a 2.4% probability. The fact that a customer can only move to adjacent states, and that the percentage of churn is higher in lower states, suggests that the churn event is a stepwise process, where a customer gradually decreases his motivation until he churns.

Based on this analysis, it is investigated if there is a higher probability to churn if a player did a transition from observation T_1-1 to T_1, which represents the time closest to the possible churn event. Table V shows the transition analysis for $N = 2, 3, ..., 10$, for each N the percentage of churn among customers that moved up, down or stayed in a state is given for each state, e.g. a customer that moved up from state 1 to state 2, is represented in state 2 under "up". A significant difference is noticed for $N = 9$ and $N = 10$ with respectively an increase of 3.6 percentage points (pp), and 3.7 pp for customers that stay in the lowest activity state twice in a row. However, the behaviour stays similar for other states.

D. Churn Prediction

Based on the finding that motivation is related to churn from Section IV-B, a method to predict churn with HMM is developed.

Predictions are made on a HMM with $N = 10$, because this N has the highest percentage of churn in its lowest activity states. The evaluation is performed with a 10 fold cross-validation, where a HMM is trained on the training set of each train/test set combination. Then, the sequences from the training set are decoded with the created HMM, which allows

TABLE II. FOR A HMM WITH N STATES ON SET $\boldsymbol{A}$, THE NUMBER OF CUSTOMERS (IN THOUSANDS) IN EACH STATE AND THE CORRESPONDING PERCENTAGE OF CHURN IN BRACKETS.

	N states of HMM								
State	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	153 (6.0)	95 (8.4)	51 (13.3)	25 (22.0)	18 (26.6)	13 (32.2)	9 (38.7)	4 (46.8)	4 (47.0)
2	93 (1.4)	98 (1.7)	80 (2.5)	67 (3.8)	54 (4.8)	46 (6.4)	38 (8.7)	30 (13.3)	25 (14.6)
3	-	53 (1.4)	75 (1.5)	66 (1.9)	53 (2.3)	44 (2.7)	39 (3.1)	37 (3.9)	34 (4.3)
4	-	-	40 (1.4)	59 (1.4)	51 (1.6)	43 (1.8)	36 (2.2)	35 (2.5)	31 (2.7)
5	-	-	-	31 (1.3)	45 (1.3)	43 (1.4)	38 (1.6)	32 (1.9)	28 (2.0)
6	-	-	-	-	24 (1.4)	37 (1.4)	37 (1.4)	33 (1.5)	29 (1.7)
7	-	-	-	-	-	20 (1.4)	32 (1.3)	32 (1.3)	29 (1.5)
8	-	-	-	-	-	-	18 (1.4)	28 (1.4)	28 (1.3)
9	-	-	-	-	-	-	-	15 (1.4)	25 (1.4)
10	-	-	-	-	-	-	-	-	14 (1.4)

TABLE III. FOR A HMM WITH N STATES ON SET $\boldsymbol{A}$, THE MEAN μ_i AND THE VARIANCE σ_i^2 IN BRACKETS FOR EMISSION DISTRIBUTION b_i, WITH I=1,2,...,N.

	N states of HMM								
State	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	3.0 (3.8)	1.9 (1.5)	1.1 (0.7)	0.5 (0.3)	0.4 (0.2)	0.3 (0.1)	0.2 (0.1)	0.1 (0.0)	0.1 (0.0)
2	12.4 (50.5)	6.2 (3.1)	3.7 (1.0)	2.6 (0.5)	2.1 (0.4)	1.8 (0.3)	1.5 (0.3)	1.1 (0.2)	1.1 (0.2)
3	-	15.8 (59.3)	7.8 (3.1)	5.1 (1.0)	4.0 (0.5)	3.3 (0.3)	2.8 (0.2)	2.4 (0.2)	2.2 (0.1)
4	-	-	17.6 (64.9)	9.2 (3.2)	6.5 (1.0)	5.1 (0.5)	4.2 (0.3)	3.7 (0.2)	3.3 (0.2)
5	-	-	-	19.3 (70.5)	10.5 (3.3)	7.5 (1.0)	5.9 (0.4)	5.0 (0.3)	4.4 (0.2)
6	-	-	-	-	20.9 (76.1)	11.5 (3.4)	8.2 (1.0)	6.7 (0.5)	5.7 (0.3)
7	-	-	-	-	-	22.1 (80.5)	12.2 (3.5)	9.0 (1.0)	7.3 (0.4)
8	-	-	-	-	-	-	23.1 (84.0)	13.1 (3.7)	9.7 (1.0)
9	-	-	-	-	-	-	-	24.2 (88.3)	13.8 (3.8)
10	-	-	-	-	-	-	-	-	25.0 (91.6)

TABLE IV. TRANSITION MATRIX $\boldsymbol{G}$ FOR HMM WITH $N = 10$ FOR SET $\boldsymbol{A}$, EACH VALUE REPRESENTS A VALUE g_{ij} IN THE TRANSITION MATRIX.

	State									
State	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1	0.87	0.10	0.01	0	0	0	0	0	0	0
2	0.02	0.86	0.10	0.01	0	0	0	0	0	0
3	0	0.05	0.83	0.12	0.01	0	0	0	0	0
4	0	0	0.10	0.76	0.13	0.01	0	0	0	0
5	0	0	0	0.12	0.73	0.13	0	0	0	0
6	0	0	0	0.01	0.13	0.75	0.12	0	0	0
7	0	0	0	0	0	0.11	0.79	0.09	0	0
8	0	0	0	0	0	0	0.09	0.85	0.05	0
9	0	0	0	0	0	0	0	0.06	0.92	0.02
10	0	0	0	0	0	0	0	0	0.04	0.96

TABLE V. FOR HMMS WITH $N = 2$ TO $N = 10$ FOR SET $\boldsymbol{A}$, THE PERCENTAGE OF CHURN FOR CUSTOMERS THAT MOVED UP, DOWN, OR STAYED IN A STATE ARE SHOWN FOR EACH STATE.

	State									
Transition	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
up	-	1.4								
down	5.9	-								
stay	6.0	1.4								
up	-	1.7	1.4							
down	8.4	1.7	-							
stay	8.5	1.7	1.4							
up	-	2.5	1.5	1.4						
down	13.4	2.4	1.5	-						
stay	13.0	2.5	1.4	1.4						
up	-	3.9	1.9	1.4	1.3					
down	22.0	3.7	1.9	1.6	-					
stay	21.8	3.9	2.0	1.3	1.3					
up	-	4.9	2.2	1.6	1.3	1.4				
down	26.7	4.8	2.2	1.5	1.7	-				
stay	25.7	4.9	2.4	1.6	1.2	1.0				
up	-	6.4	2.8	1.8	1.4	1.3	1.4			
down	32.3	6.4	2.6	1.9	1.5	1.7	-			
stay	31.7	6.5	2.8	1.8	1.2	1.2	1.0			
up	-	8.6	3.5	2.1	1.7	1.4	1.3	1.4		
down	38.7	8.7	3.0	2.2	1.5	1.6	1.5	-		
stay	38.4	8.9	3.1	2.3	1.8	1.2	1.3	1.2		
up	-	13.0	3.8	2.6	1.8	1.6	1.3	1.3	1.4	
down	46.6	13.4	3.9	2.3	1.8	1.4	1.5	1.6	-	
stay	50.2	13.2	3.9	2.7	2.1	1.4	1.0	1.4	1.4	
up	-	13.9	4.5	2.8	1.9	1.7	1.5	1.3	1.3	1.4
down	46.8	14.6	4.3	2.7	2.1	1.6	1.6	1.5	1.7	-
stay	50.5	14.9	4.4	2.7	2.1	1.5	1.4	1.0	1.3	1.3

to conduct the churn percentage analysis in each state as in Section IV-B.

Next, the state with most churn is declared to be a ”churn state”, which means that if a customer is in that state at observation T_1, he is predicted to be a churner. Next, sequences from the test set are decoded with the HMM estimated based on the training set, which gives motivational states to each observation in the test sequences. As the real churn labels of each sequence are known, it is possible to create a true positive rate (TPR) and false positive rate (FPR) for these predictions.

ROC curves are built on TPR and FPR and are used to evaluate the performance of the algorithms. For algorithms as Logistic Regression and Neural Network that have an output in form of a probability, the ROC curve is created based on different thresholds for which a prediction is said to be of one class or another. SVM's predictions are transformed to a probabilistic output by fitting a logistic distribution using maximum likelihood to the decision values, refer to [25], [26] for more details. Finally, for predictions with HMM, several TPR and FPR pairs are created by assigning the ”churn state” label to different combinations of states. After the prediction where only the state with most churn is declared as ”churn state”, the two states with most churn are declared as ”churn states”, and make churn predictions based on this. Then, the three states with most churn are declared as ”churn states”, and so on.

TABLE VI. AUC VALUES WITH 95% CI FOR GRID SEARCH ON NEURAL NETWORK FOR PARAMETERS MOMENTUM M AND LEARNING RATE μ WITH A 5 FOLD CROSS-VALIDATION ON THE SET OF ACTIVE CUSTOMERS.

	μ								
m	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*
0.1	78.8	78.9	78.9	78.9	79.0	79.0	79.0	79.0	79.1
0.2	78.8	78.9	78.9	78.9	79.0	79.0	79.0	79.1	79.1
0.3	78.9	78.9	78.9	79.0	79.0	79.0	79.1	79.1	79.1
0.4	78.9	78.9	78.9	79.0	79.0	79.1	79.1	79.1	79.1
0.5	78.9	78.9	79.0	79.0	79.1	79.1	79.1	79.1	79.1
0.6	78.9	78.9	79.0	79.1	79.1	79.1	79.1	79.1	79.1
0.7	78.9	79.0	79.1	79.1	79.1	79.1	79.1	79.0	79.0
0.8	79.0	79.1	79.1	79.1	79.0	79.0	78.2	78.2	78.7
0.9	79.1	79.1	77.9	78.2	78.2	76.7	76.7	75.5	76.4

E. Classic churn prediction methods on activity

In this Section, parameter tuning for Neural Network and SVM is performed on the set $\boldsymbol{A}$ used for HMM. These techniques do not require a specific order for the data, and dimensionality reduction can increase robustness and performance. Therefore, a Principal Components Analysis (PCA) that keeps 95% of the variance is applied before tuning parameters. For more information about PCA refer to [28]. ROC curves are used to compare performance, but a more convenient representation for parameter selection is to use the Area Under the ROC Curve (AUC), which gives the general performance of the algorithm for a set of parameters.

1) Neural Network: The number of nodes a in the hidden layer is selected as the average between the nodes in the input and output layers. This number of nodes has shown best results in [9] and a similar value has shown best results in [13]. The training time is set to 500 epochs.

Remaining parameters are the momentum $m \in [0, 1]$ and learning rate $\mu \in [0, 1]$, which are used to update weights between nodes during training. A grid search with a 5 fold cross-validation is conducted and parameters are selected based on the average AUC values with the 95% CI achieved during cross-validation. Table VI shows that several parameter combinations result in not statistically significant different average performance on a 95% confidence level. The parameter pair is chosen among the highest AUC values of 79.1 and is $m = 0.8$ and $\mu = 0.3$.

2) SVM: The upper bound on training error C and kernel parameter γ are selected based on a 5-fold cross-validation on exponential sequences of C and γ (i.e., $C = 2^{-5}, 2^{-3}, ..., 2^{13}; \gamma = 2^{-15}, 2^{-13}, ..., 2^{3}$), as recommended by Hsu et al. [29]. Parameters are chosen based on the performance of average AUC value and its 95% CI during cross-validation. In interest of time, a 15% sample of the set of players is used for grid-search. In addition, the features of the dataset are scaled to the range [0,1] before applying SVM. This avoids attributes with higher values dominating those with smaller, as well as numerical difficulties during calculation [29].

Table VII shows that several parameter combinations result in not statistically significant different average performance on a 95% confidence level, therefore the parameter pair with the highest average AUC of 79.7 is chosen $C = 2^7$ and $\gamma = 2^{-13}$.

TABLE VII. AUC VALUES WITH 95% CI FOR GRID SEARCH ON SVM FOR COST C AND KERNEL PARAMETER γ WITH A 5 FOLD CROSS-VALIDATION ON A 15% SAMPLE OF THE SET OF ACTIVE CUSTOMERS.

	γ									
C	2^{-15}	2^{-13}	2^{-11}	2^{-9}	2^{-7}	2^{-5}	2^{-3}	2^{-1}	2^{1}	2^{3}
2^{-5}	72.0	70.1	67.1	67.0	69.5	73.7	76.2	77.3	76.4	79.3
2^{-3}	72.0	70.1	66.7	66.3	69.8	75.6	75.7	76.7	77.3	79.3
2^{-1}	72.0	69.7	65.4	65.6	73.7	76.6	76.9	77.0	78.0	79.3
2^{1}	71.9	66.5	66.2	61.2	75.0	73.2	75.3	76.5	78.4	79.4
2^{3}	70.5	67.3	63.1	69.8	76.4	78.5	77.2	77.3	78.6	79.5
2^{5}	79.4	79.5	79.4	79.4	79.5	79.3	79.0	78.1	78.9	79.0
2^{7}	79.6	79.7	79.6	79.4	79.5	79.3	79.0	78.7	79.3	78.2
2^{9}	79.6	79.6	79.5	79.4	79.5	79.3	79.1	78.8	79.5	77.8
2^{11}	79.6	79.6	79.5	79.4	79.5	79.3	79.2	79.4	79.2	77.3
2^{13}	79.6	79.5	79.5	79.4	79.5	79.4	79.2	79.6	78.5	76.8

F. Benchmark

In this Section, churn prediction with HMM is benchmarked against Logistic Regression, Neural Network and SVM. All experiments are made with a 10 fold cross-validation, on the set of active customers $\boldsymbol{A}$.

Fig. 2 shows the ROC curves for these algorithms. Logistic Regression has the highest performance in the range 0 to 0.27 false positive rate, but for the range 0.27 to 1 SVM has the highest performance. A 10 states HMM has 10 operating points for predictions, the curve between these is only an approximation. Nevertheless, in its operating points HMM has the same performance as Neural Network and Logistic Regression, which have the highest performance in the low false positive rate. If more or different operating points are desired there is the possibility to use a HMM with more states $N > 10$.

In a practical implementation, the choice of algorithm is made based on a trade-off between false positive rate and true positive rate. However, low false positive rates are generally preferred, which makes HMM a valid choice for implementation.

V. Conclusion

This paper covers the results of a project aimed at building an effective expert system for churn prediction and prevention at one of the worlds leading mobile game developers. Churn prediction is operationalised as a daily prediction of churn for customers active in the last 24 hours. A mathematical definition is formulated for active customers in the last 24 hours, which constitutes the target audience for prediction. Then, the churn event of customers is defined by including a time margin that gives the possibility for proactive reactions to the event. Finally, the churn prediction problem is defined as a binary prediction problem.

The new approach to churn prediction developed in this work, is to model the intrinsic motivation of customers to use the product with HMM. A clear link between motivational states and churn is found, with up to 47% churn in the lowest motivational state. Furthermore, the previous state influences the current state of motivation, which results in a rise of up to 3.7 percentage points in churn present in a state. This link is used to define specific motivational states as churn states. When a customer is in such a motivational state,

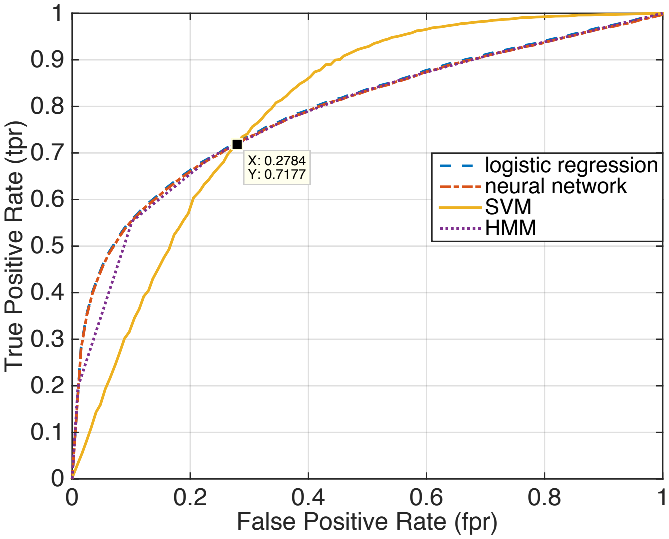

Fig. 2. ROC curves for Logistic Regression, Neural Network, SVM and HMM.

he is considered to be a churner, which allows to make binary predictions with HMM. HMM performs similarly to Logistic Regression, Neural Network and SVM. It however has substantial advantages for use in expert systems due to low storage and computational requirements and output of highly relevant customer motivational states. It allows to create an intelligent system that clusters customers at each step in time. This gives an intuitive output that can be readily acted upon by non-expert users. Customers in the lowest motivational state can receive strongest incentives which can then be adapted stepwise for higher motivational states. The implementation of the system is highly effective in that it only uses a small set of generic data. This further allows to adapt the system very easily for other mobile applications on the various app stores.

VI. Outlook

This work has shown that an expert system for churn prediction can be built with a HMM and that its performance combined with its other advantages make it a good choice for implementation. Nevertheless, there are still some limitations and open questions which can give directions for future research. Limitations concern the constraining assumptions made on the model itself. First, there is the Markov property that limits the influence of previous states on the current state. Second, the emission distribution is assumed to be Gaussian. A possible way of addressing the limitation of the Markov property is to build a higher order HMM, which extends the influence of previous states on the current state. This could lead to a better understanding of the transitional behaviour of customers. The second limitation concerning the Gaussian emission distribution can be addressed by experimenting with different distributions, e.g. an exponential distribution. This could lead to a model that better fits the data and the underlying process.

An open question is about how HMM is best applied to make predictions. In this paper, a HMM models the motivational states of all customers. Predictions are made by declaring motivational states as churn or non-churn states. This is based on the observation that there is a high amount of churning customers in certain states of motivation. A different approach could be to model churning and non-churning customers separately. Thereby, there would be two HMMs, one for each group of customers. As seen in Section III-A, one of the problems that can be solved with a HMM is the evaluation problem. This states that given an observation sequence and a HMM model, the probability of the observation sequence given the model can be computed. Having two HMMs, one for each group of customers, one could compute the probability for a customer to be generated by each HMM. Thereby, the HMM that has a higher probability to have generated the customer would assign that customer to its group. Hence, predictions are made by assigning a customer to the model that best describes him. In addition to classification, this method might give more insights into the behaviour of each group of customers, because the motivational states of an HMM would correspond to a specific group. However, there would be only one operational point for classification.

A different approach for classification consists of modelling only one group of customers. The previous method used two group specific HMMs. Predictions were made by comparing the probabilities for a customer of being generated by each of these two models. Instead, if a HMM models only churners or only non-churners, predictions could be made by looking at how likely it is that a customer is generated by this specific HMM. A threshold on this probability could then separate the customers into churners and non-churners. For example, if a HMM models only non-churning customers, it is likely that non-churning customers have a higher probability of being generated by this model than churning customers. This method allows to have more operational points, because it depends on the threshold on the probability and not on the number of states in the model as in the current implementation. Furthermore, in practice it is often the case that datasets are imbalanced and that data describing churning customers is rare. Thereby, this method could be an advantage, because only data describing non-churning customers could be used to make predictions.

Finally, this work has shown that the transitional behaviour of a customer can influence his probability to churn. However, this information has not been used to make predictions yet. Therefore, a way to improve performance might be to add this information to the prediction.

Acknowledgment

The authors would like to thank the gaming company for making this project possible.

References

[1] W. J. Reinartz and V. Kumar, "The impact of customer relationship characteristics on profitable lifetime duration," *Journal of marketing*, vol. 67, no. 1, pp. 77–99, 2003.

[2] S. F. Slater and J. C. Narver, "Intelligence generation and superior customer value," *Journal of the academy of marketing science*, vol. 28, no. 1, pp. 120–127, 2000.

[3] F. F. Reichheld and W. E. Sasser Jr, "Zero defections: quality comes to services." *Harvard business review*, vol. 68, no. 5, pp. 105–111, 1989.

[4] D. Van den Poel and B. Lariviere, "Customer attrition analysis for financial services using proportional hazard models," *European journal of operational research*, vol. 157, no. 1, pp. 196–217, 2004.

[5] S. Reyburn, "January 2015 cpi data: See how average mobile game cpi bids have evolved y-o-y," January 2015. [Online]. Available: http://blog.chartboost.com/mobile-game-cpi-january-2015/

[6] PocketGamer, "Count of active applications in the app store," October 2014. [Online]. Available: http://www.pocketgamer.biz/metrics/app-store/app-count/

[7] C. Lagane, "Les ventes de smartphones soutenues par les marches emergents," October 2014. [Online]. Available: http://www.silicon.fr/les-ventes-smartphones-soutenues-les-marches-emergents-96380.html

[8] F. Hadiji, R. Sifa, A. Drachen, C. Thurau, K. Kersting, and C. Bauckhage, "Predicting player churn in the wild," in *Computational Intelligence and Games (CIG), 2014 IEEE Conference on*. IEEE, 2014, pp. 1–8.

[9] B. Q. Huang, T.-M. Kechadi, B. Buckley, G. Kiernan, E. Keogh, and T. Rashid, "A new feature set with new window techniques for customer churn prediction in land-line telecommunications," *Expert Systems with Applications*, vol. 37, no. 5, pp. 3657–3665, 2010.

[10] B. Huang, M. T. Kechadi, and B. Buckley, "Customer churn prediction in telecommunications," *Expert Systems with Applications*, vol. 39, no. 1, pp. 1414–1425, 2012.

[11] K. Coussement and D. Van den Poel, "Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques," *Expert systems with applications*, vol. 34, no. 1, pp. 313–327, 2008.

[12] V. L. Miguéis, D. Van den Poel, A. S. Camanho, and J. Falcão e Cunha, "Modeling partial customer churn: On the value of first product-category purchase sequences," *Expert systems with applications*, vol. 39, no. 12, pp. 11 250–11 256, 2012.

[13] J. Runge, P. Gao, F. Garcin, and B. Faltings, "Churn prediction for high-value players in casual social games," in *Computational Intelligence and Games (CIG), 2014 IEEE Conference on*. IEEE, 2014, pp. 1–8.

[14] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[15] J. O'Connell, S. Højsgaard *et al.*, "Hidden semi markov models for multiple observation sequences: The mhsmm package for r," *Journal of Statistical Software*, vol. 39, no. 4, pp. 1–22, 2011.

[16] D. Ramage, "Hidden markov models fundamentals," *Lecture Notes. http://cs229. stanford. edu/section/cs229-hmm.pdf*, 2007.

[17] W. Zucchini and I. L. MacDonald, *Hidden Markov models for time series: an introduction using R*. CRC Press, 2009.

[18] D. Hosmer and S. Lemeshow, *Applied Logistic Regression*, ser. Applied Logistic Regression. Wiley, 2004. [Online]. Available: http://books.google.ie/books?id=Po0RLQ7USIMC

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.

[20] M. T. Hagan, H. B. Demuth, M. H. Beale *et al.*, *Neural network design*. Pws Boston, 1996, vol. 1.

[21] S. Haykin, *Neural Networks and Learning Machines*, ser. Neural networks and learning machines. Prentice Hall, 2009, no. v. 10. [Online]. Available: http://books.google.com.au/books?id=K7P36lKzI_QC

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[23] V. N. Vapnik and V. Vapnik, *Statistical learning theory*. Wiley New York, 1998, vol. 2.

[24] S. Kim, K.-s. Shin, and K. Park, "An application of support vector machines for customer churn analysis: Credit card case," in *Advances in Natural Computation*. Springer, 2005, pp. 636–647.

[25] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch, *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2014, r package version 1.6-4. [Online]. Available: http://CRAN.R-project.org/package=e1071

[26] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

[27] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[28] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2005.

[29] C. wei Hsu, C. chung Chang, and C. jen Lin, "A practical guide to support vector classification," 2010.