Churn Modelling: A Perspective on Mixture-Based Clustering of Customer Behaviours

Victor (Sheng) Wang

University of Oxford

Supervisors: Andrew Mellor, Junaid Mubeen

A technical report for

InFoMM CDT Mini-Project 1

in partnership with

Whizz Education

Trinity 2018

Contents

1 Introduction

Retained customers in general create higher revenues than new customers do, and making a sell to a new customer can cost up to 5 times more depending on the business [5]. Therefore, many companies form the Customer Relationship Management (CRM) team with a focus on customer retention strategies. A crucial step is then to identify high risk customers who are intending to discontinue their usage of the services. This assessment is better known as *churn prediction*.

Our project aims to perform the churn prediction task based on investigation of customers's behavioural clusterings, and formulate the processes into a scalable pipeline which can be easily reused, updated and extended for many applications. In particular, we apply the pipeline to analyse pupil subscribers' data for Whizz Education (referred to as "Whizz"). Whizz provides online virtual tutorial service, Math-Whizz, which pupils can access by purchasing subscriptions. We have investigated whether behavioural-based data analytics can be used to help detect potential subscription cancellations.

The pipeline starts from representing pupils' behaviours by numerical data, also known as *feature* extraction. The structured features' data are fed into a mixture model that decodes features' distribution as a weighted combination of simple distributions. Since simple distribution is assumed to be generated by an unobserved *state*, we are interested in uncovering the states as well as their associated emitted feature distributions, also known as *clusters*. The mixture model is deemed to be effective if the identified clusters of pupils exhibit distinguishable proportions of churn, or *churn rates*. Assessing churn rate of the fitted mixture model results in a trained model that establishes a map between features to probability of churn. This enables the prediction of new coming pupils' churn probabilities by feeding their feature data into the trained model.

We show that clusters inferred from behavioural data are characterised by non-trivially different churn rate. Moreover, there is recognisable pattern observed in cluster sizes. This verifies the effectiveness of the mixture model approach in uncovering what level of churn risk the pupils are at. We further investigate how features impact churn probability and examine that the mixture model can result in very minimal overfitting issues in prediction practices. We also infer the temporal transitional probabilities of states by studying on the dynamics of behavioural changes.

The report is structured as follows. We start by elaborating the modelling framework and pipeline in section 2. Next in section 3 we describe the feature data preparation and pre-processing techniques used to adapt Whizz's data for better modelling performance. Then we explain the details of clustering analysis and model evaluation in section 4. Finally, we carry out prediction task and also assess overfitting issues, as detailed in section 5. We conclude the project with deliverables and future directions in section 6.

2 Modelling Customer Behaviours

Customers' behaviour evolves over time as they respond to business offering and adjust to their own demand. We would like to represent the various behaviour dynamics of a collection of customers by panel data, and employ the Markov chain model to describe these data. In addition, we use the mixture model to find Markov states, each of which defines a partition of the behaviours observable within a single time interval.

2.1 Representing Behaviours by Features

A *feature* is an individual measurable property of a behaviour being observed, and choosing informative features is crucial for effective clustering. For example, to measure how often the pupil uses Whizz online tutorial, we can define the time spent or number of visits within a month as the feature. For each customer, we define multiple features to capture his behaviours of various aspects within a time interval. Suppose we are interested in studying bahaviours of n customers in T discrete consecutive time periods, and define m features, then we denote the feature data as a sequence:

$$\{\mathbf{X}_1,\ \mathbf{X}_2,\ \ldots,\ \mathbf{X}_t,\ \ldots,\ \mathbf{X}_T\}\,, \tag{1}$$

in which the t-th element $\mathbf{X}_t = (x_{ij}^t) \in \mathbb{R}^{m\times n}$ is a matrix for all $t = 1, 2, \ldots, T$. In addition, we denote the features of customer j at the t-th time interval by the j-th column of $\mathbf{X}_t$ by $\mathbf{x}_{tj} = [x_{1j}^t\ x_{2j}^t\ \cdots\ x_{mj}^t]^\top$.

2.2 Customer Journeys and Markov Chain

Customer journeys reflect their feature dynamics over time, denoted as (1). One practical challenge of computing such sequence of matrices is to resolve the inconsistency present in journeys of different customers. The inconsistency refers to the problem that the time intervals for different customers being alive in the services are not aligned, so that their features are not comparable. Moreover, it is highly likely to have significant missing information within specific time intervals for customers who have not yet entered the service or have already churned. To resolve the inconsistency, we align and aggregate customers' features by *customer month* rather than calendar month. Doing so enables the effective modelling using Markov chain.

2.2.1 Customer Month

Splitting pupils' behaviours into monthly time periods makes most sense provided the business settings at Whizz. Pupils subscribe to access Whizz products on a 1-month contract, and make the choice to leave the service at the end of each subscription. If no action is taken, a renewal will be made by default.

Due to the inconsistency present in journeys of different customers, we align their features by switching the reference from calendar month to customer month. This is illustrated by an example in Figure 2.1. It provides much cleaner and more sensible data for modelling task.

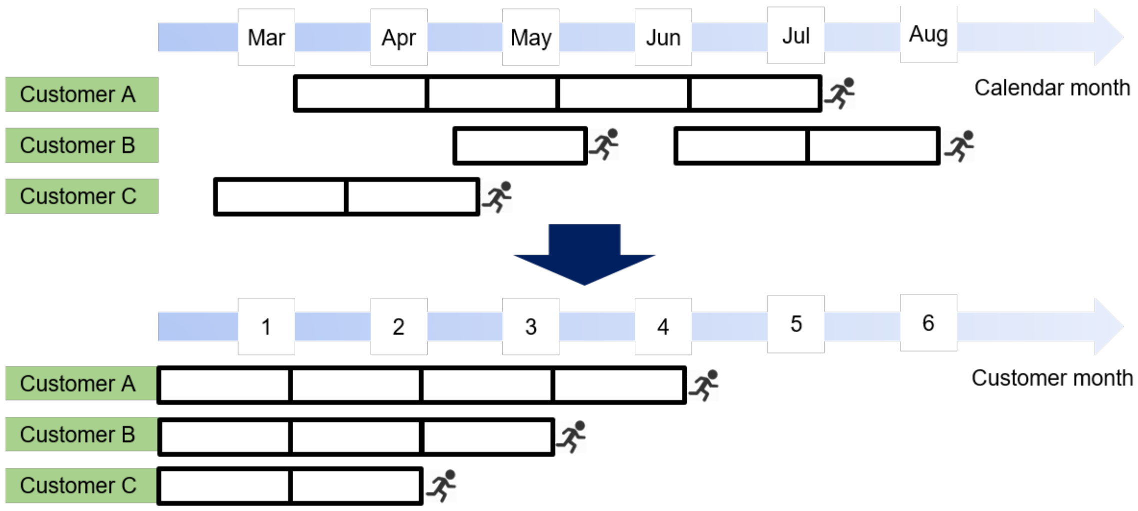

Figure 2.1: Change reference from calendar month to customer month. Customer A, B and C have very different journeys in the sense of subscription start and end dates. Each block represents customer's monthly features. Under calendar month reference, we have to choose studying months from March to August to cover all activities. This choice results in irregular temporal distribution of missing information for all 3 customers. After changing the reference to customer month, features are aligned by customer month and therefore comparable. Moreover, the missing information only occurs after the customer churns. It can also handle discontinuous subscriptions like the case of customer B.

2.2.2 Markov Chain - Dynamic Model for Behavioural Changes

We assume that customers with intentions to churn exhibit different behaviours than others do. Behaviours are different distributionally, and generated from a finite number of *states*. Then the discrete time behaviour of each customer results in a chain of states over time. This formulates into a discrete-time Markov chain with states transiting over time, where each state emits distinguishable behaviour distribution from others. An example is given in Figure 2.2. In brief, a Markov chain is a stochastic process characterised by *transition* and *emission* probabilities.

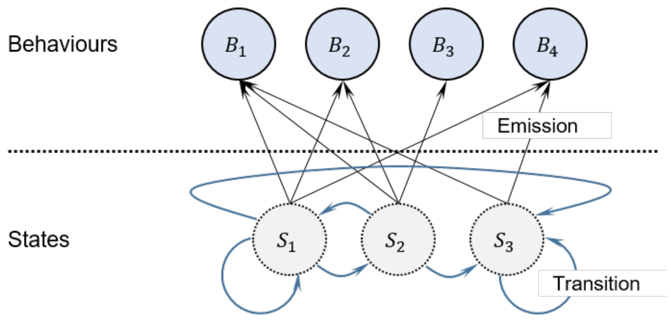

Figure 2.2: Behaviours emitted from Markov states. States S_1, S_2 and S_3 generate differently distributed behaviours. For example, S_1 generates $\{B_1, B_2, B_4\}$ while S_3 produces $\{B_1, B_4\}$. Even if the two states can generate the same set of behaviours, the emission probabilities can be different, thus still resulting in different behaviour distributions. States transit between each other over time stochastically.

Transition Consider the behavioural journey of customer j, which is represented by a sequence of feature data $\{\mathbf{x}_{tj}\}_{t=1}^{T}$. At time interval t, feature $\mathbf{x}_{tj}$ is an instance from a distribution generated by a state. Let's denote the time sequence of generative states as $\{s_t(j)\}_{t=1}^{T}$. In a Markov chain setting the sequences are defined stochastically, with the next state being conditionally dependent on the present state, but not any further previous history. This is known as *Markov property.*

If we define a finite set of states by $\mathcal{S} = \{S_q\}_{q=1}^{Q}$, then $s_t(\cdot)$ is a map from $\mathbb{N}$ to $\mathcal{S}$. Due to the Markov property, successive states are linked together with the (conditional) *transition probability matrix* defined as:

$$A_t = (a_{pq}^t) \in [0,1]^{Q\times Q}, \tag{2a}$$

with

$$a_{pq}^t = \mathbb{P}\left(s_{t+1} = S_p | s_t = S_q\right). \tag{2b}$$

If we assume time-homogeneity or *stantionarity* of the Markov chain, then we can remove the time subscript from the notations because the parameters describing all probabilistic transitions are themselves constant.

Emission Recall that feature data $\{\mathbf{x}_{tj}\}_{t=1}^{T}$ observed are instances from some distribution. We introduce a generic notation for observed sequence of features, $\{\mathbf{o}_t\}_{t=1}^{T}$, where $\mathbf{o}_t = [o_{t1}\ o_{t2}\ \cdots\ o_{tm}]^\top$. The sequence of observations are generated in the following way:

1. At each time step, the system generates a state s_t according to the state-to-state transition probability matrix A_t.

2. Once the state s_t has been generated, the system generates a cluster c_t according to a state-to-cluster emission probability distribution $\pi(s_t, c_t)$. Suppose we define a finite collection of clusters $\mathcal{C} = \{C_k\}_{k=1}^{K}$, then we denote:

$$\pi_{qk} = \pi(S_q, C_k) = \mathbb{P}\left(c_t = C_k | s_t = S_q\right). \tag{3}$$

3. Once the cluster c_t have been determined, an observation vector $\mathbf{o}_t$ is produced probabilistically according to some cluster-specific distribution $\phi(\mathbf{o}_t|\theta(s_t, c_t))$, where $\theta(s_t, c_t)$ denotes the distribution parameters. We write,

$$\phi(\mathbf{o}_t|\theta_{qk}) = \phi(\mathbf{o}_t|\theta(S_q, C_k)) = \mathbb{P}\left(\mathbf{o}_t | s_t = S_q, c_t = C_k\right). \tag{4}$$

Given the generative process described above, we can now model the state-to-observation emission probability by a mixture of densities:

$$\begin{aligned} b_q(\mathbf{o}_t) &= \mathbb{P}\left(\mathbf{o}_t | s_t = S_q\right) && \text{(5a)}\\ &= \sum_{k=1}^{K} \mathbb{P}\left(c_t = C_k | s_t = S_q\right) \mathbb{P}\left(\mathbf{o}_t | s_t = S_q, c_t = C_k\right) && \text{(5b)}\\ &= \sum_{k=1}^{K} \pi_{qk}\phi(\mathbf{o}_t|\theta_{qk}). && \text{(5c)} \end{aligned}$$

2.2.3 Decoding the Markov Chain

The problem is how to estimate the transition probabilities and parameters in the emission term, A_t and (π, θ) from the observations $\mathbf{X}_t$. Our strategy is to split this decoding process into two separate steps, where we first make inference on emission and then uncover the temporal structure configured by transition.

Decoding Emission Ideally from the perspective of practical application of the model, we wish to encode a finite number of states representing different levels of churn risk. As a consequence, the prediction task is to find out the state that the pupil belongs to and therefore assigning the associated risk label. Hence, we choose not to infer states purely from observed behavioural data, but define states by also incorporating churn outcome information. At high level, we take two steps to estimate the parameters in the emission term:

1. We look at the behavioural distribution without conditioning on state, namely,

$$b(\mathbf{o}_t) = \mathbb{P}(\mathbf{o}_t) = \sum_{k=1}^{K} \mathbb{P}(c_t = C_k)\mathbb{P}(\mathbf{o}_t | c_t = C_k) = \sum_{k=1}^{K} \pi_k \phi(\mathbf{o}_t | \theta_k). \tag{6}$$

   Then we estimate $\{\pi_k, \theta_k\}_{k=1}^K$ from observed feature data $\mathbf{X}_t$ with pre-defined multivariate kernel density $\phi(\cdot)$. This will be elaborated in section 2.3.

2. The previous step gives not only the weights π and cluster density $\phi(\cdot|\theta)$, but also a consequential cluster assignment of all customers. If we know the churn outcome for all customers, then we can calculate the proportion of churned customers, or churn rate, within each cluster. Thereafter, we can form states by grouping together clusters of similar level of churn rate.

   Formally, we define the set of customers who have been assigned into cluster C_k as

$$\mathcal{N}_k^t = \{j : \ j \text{ is assigned into } C_k \text{ at time } t\}. \tag{7}$$

   Meanwhile, we define the churn outcome information by a set

$$\mathcal{N}_{\text{churn}}^t = \{j : \ j \text{ churns at } t+1\}, \tag{8}$$

   so that the cluster churn rate is defined as

$$\lambda_k^t = \frac{|\mathcal{N}_k^t \cap \mathcal{N}_{\text{cancel}}^t|}{|\mathcal{N}_k^t|}. \tag{9}$$

   Once the churn rates of all K clusters, $\{\lambda_k^t\}_{k=1}^K$, are computed, we group them and form Q states. Typically Q is much smaller than K. We denote the set of clusters consisting state S_q as,

$$\mathcal{K}_q^t = \{k : \ C_k \text{ is emitted from } S_q \text{ at time } t\}. \tag{10}$$

Afterwards, we can revisit the calculation for the state-to-observation emission probability $b_q(\mathbf{o}_t)$. Note that by the way we define state, we have

$$\pi_{qk} = \mathbb{P}\left(c_t = C_k | s_t = S_q\right) = \frac{|\mathcal{N}_k^t|}{\sum_{l \in \mathcal{K}_q^t} |\mathcal{N}_l^t|} \mathbb{1}_{\{k \in \mathcal{K}_q^t\}}, \tag{11}$$

where $\mathbb{1}_{\{\cdot\}}$ is the indicator function. Then

$$b_q(\mathbf{o}_t) = \sum_{k=1}^{K} \pi_{qk} \phi(\mathbf{o}_t; \theta_{qk}) = \sum_{k \in \mathcal{K}_q^t} \frac{|\mathcal{N}_k^t|}{\sum_{l \in \mathcal{K}_q^t} |\mathcal{N}_l^t|} \phi(\mathbf{o}_t | \theta_k). \tag{12}$$

Decoding Transition We define the set of customers who transit from S_p at t to S_q at $t+1$ as

$$\mathcal{Q}_{q \to p}^t = \left\{ j : \; s_t(j) = S_q, \; s_{t+1}(j) = S_p \right\}. \tag{13}$$

We assume that customers' behaviours are i.i.d. samples from the generating process, then the maximum likelihood estimate of the transition probability is

$$\hat{a}_{pq}^t = \frac{|\mathcal{Q}_{q \to p}^t|}{\sum_{l=1}^{Q} |\mathcal{Q}_{q \to l}^t|}. \tag{14}$$

If the Markov chain is assumed to be stationary, then we estimate

$$\hat{a}_{pq} = \frac{\sum_{t=1}^{T} |\mathcal{Q}_{q \to p}^t|}{\sum_{t=1}^{T} \sum_{l=1}^{Q} |\mathcal{Q}_{q \to l}^t|}. \tag{15}$$

2.3 Probabilistic Clustering Using Mixture Model

A critical step of the state-cluster-observation Markov chain is the mixture model that describes the probabilistic assignment of observations to clusters. Practical calibration of the mixture model faces many choices such as the kernel density $\phi(\cdot)$, the number of clusters K, etc. A fundamental model setting is however the choice of frequentist or Bayes approach to estimate model parameters. We choose specifically the *Dirichlet Process Mixture* setting which has two important features:

1. It is Baysian and treats θ as a random variable, of which distributions will be updated from a prior as observed data coming in.

2. The Dirichlet process (DP) is used as a nonparametric prior resulting in that the number of clusters is random and grows as new data are observed.

The benefits of this model choice are massive. It does not view the observed data as infinitely available as independent replicates like frequentists, so that it does not worry about the unobserved data and can be updated with new data coming in. Moreover, it infers the number of clusters from observed data, and opens the opportunities of finding new clusters as more data are observed.

2.3.1 Dirichelet Process Mixture Model

Definitions A Dirichelet Process (DP) is a distribution of a random measure. Let G_0 be a base distribution (measure) for our cluster density parameter $\theta \in \Theta$, a measurable space, and let α be a positive, real-valued scalar. A random measure G is then distributed according to *Dirichelet Process* with scaling parameter α and base measure G_0:

$$G \sim \mathrm{DP}(\cdot|G_0, \alpha), \tag{16a}$$

if for all $K \in \mathbb{N}$, and all $\{\Theta_1, \dots, \Theta_K\}$ finite partitions of Θ:

$$(G(\theta_1), \dots, G(\theta_K)) \sim \mathrm{Dir}\left(\alpha G_0(\theta_1), \dots, \alpha G_0(\theta_K)\right), \tag{16b}$$

where $\mathrm{Dir}(\cdot)$ denotes the *Dirichelet distribution.* The Dirichelet distribution is a distribution of the standard $K-1$ simplex. Let $\boldsymbol{\pi} = \{\pi_k\}_{k=1}^K$ with $\sum_{k=1}^K \pi_k = 1$ and $\forall k : \pi_k \geq 0$, and let $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_K)$ with $\alpha_1, \dots, \alpha_K \geq 0$. Then

$$\mathbb{P}(\boldsymbol{\pi}|\boldsymbol{\alpha}) = \mathrm{Dir}(\alpha_1, \dots, \alpha_K) = \frac{1}{\mathrm{Beta}(\boldsymbol{\alpha})} \prod_{k=1}^K \pi_k^{\alpha_k - 1} = \frac{\Gamma\left(\sum_k \alpha_k\right)}{\prod_k \Gamma(\alpha_k)} \prod_{k=1}^K \pi_k^{\alpha_k - 1}, \tag{16c}$$

where $\mathrm{Beta}(\cdot)$ is the beta function, $\Gamma(\cdot)$ is the gamma function.

Clustering Effect We use DP as a prior to distribution of cluster parameter θ:

$$\theta|G \sim G(\cdot) \quad \text{and} \quad G \sim \mathrm{DP}(\cdot|G_0, \alpha). \tag{17}$$

This model exhibits a "clustering effect" which enables us to infer number of clusters from data rather than pre-defining it. Suppose we independently draw n random values $\theta^{(j)}$ from G under the model (17), then Blackwell and MacQueen's urn representation theorem [1] states that, marginalising out the random measure G, the joint distribution of the collection of variables $\{\theta^{(1)}, \dots, \theta^{(n)}\}$ exhibits a clustering effect:

$$\mathbb{P}\left(\theta^{(j)}|\theta^{(1)}, \dots, \theta^{(j-1)}\right) \propto \alpha G_0(\theta^{(j)}) + \sum_{l=1}^{j-1} \delta_{\theta^{(l)}}(\theta^{(j)}), \tag{18}$$

where $\delta_{\theta^{(l)}}(\cdot)$ is a Dirac delta at $\theta^{(l)}$. Thus the variables $\{\theta^{(1)}, \dots, \theta^{(n)}\}$ are randomly partitioned according to which variables are equal to the same value. Moreover, let $\{\theta_1, \dots, \theta_K\}$ denote the distinct values of the drawn samples $\{\theta^{(1)}, \dots, \theta^{(j-1)}\}$, let $\{\kappa_1, \dots, \kappa_{j-1}\}$ be the assignment variables such that $\theta^{(l)} = \theta_{\kappa_l}$. Then,

$$\mathbb{P}\left(\theta^{(j)}|\theta^{(1)}, \dots, \theta^{(j-1)}\right) \propto \frac{\alpha}{j-1+\alpha} G_0(\theta^{(j)}) + \sum_{k=1}^K \frac{|\{l : \kappa_l = k\}|}{j-1+\alpha} \delta_{\theta^{(l)}}(\theta^{(j)}). \tag{19}$$

This implies that the j-th draw has a probability of $(j-1)/(j-1+\alpha)$ to be exactly the same as some previously drawn value. This forms a natural clustering effect. In addition, with a probability of $\alpha/(j-1+\alpha)$ a new cluster will be produced with the new distinct value θ_{K+1}. Hence the number of clusters is allowed to grow as new data are observed.

Dirichelet Process Mixture Model Given the clustering effect exhibited in DP, we define the *Dirichelet process mixture model* as:

$$\mathbf{o}|\theta \sim \phi(\cdot|\theta) \quad \text{and} \quad \theta|G \sim G(\cdot) \quad \text{and} \quad G \sim \mathrm{DP}(\cdot|G_0, \alpha), \tag{20}$$

recall that $\mathbf{o}$ is the observed feature vector, we remove the time subscript t since the mixture model holds for all time steps.

2.3.2 Generative Process with Stick-Breaking Representation

The definition of DP stated in (16) does not provide useful information of generating a DP in practice. Sethuraman [4] proposes the *stick-breaking representation* to explicitly construct a DP. Suppose there are two infinite sets of independent random variables, $\beta_k \sim \mathrm{Beta}(1, \alpha)$ and $\theta_k \sim G_0$, $\forall k \in \{1, 2, \dots\}$. The stick-breaking representation of G is:

$$G(\cdot) = \sum_{k=1}^{\infty} \pi_k(\boldsymbol{\beta})\delta_{\theta_k}(\cdot), \tag{21a}$$

where

$$\pi_k(\boldsymbol{\beta}) = \beta_k \prod_{l=1}^{k-1}(1 - \beta_l). \tag{21b}$$

Note that by construction $\sum_{k=1}^{\infty} \pi_k(\boldsymbol{\beta}) = 1$. In the DP mixture model, $\boldsymbol{\pi}(\boldsymbol{\beta}) = \{\pi_k(\boldsymbol{\beta})\}_{k=1}^{\infty}$ gives the mixing proportions of mixture components represented by atoms $\{\theta_k\}_{k=1}^{\infty}$. By far, we can describe the feature data generative process as follows:

1. Draw an infinite collection of $\beta_k|\alpha \sim \mathrm{Beta}(1, \alpha)$, $\forall k \in \{1, 2, \dots\}$.
2. Draw an infinite collection of $\theta_k|G_0 \sim G_0$, $\forall k \in \{1, 2, \dots\}$.
3. For j-th data point, $j = 1, \dots, n$:
   (a) Draw $z^{(j)}|\boldsymbol{\beta} \sim \mathrm{Mult}(\boldsymbol{\pi}(\boldsymbol{\beta}))$;
   (b) Draw $\mathbf{o}^{(j)}|z^{(j)} \sim \phi(\mathbf{o}|\theta_{z^{(j)}})$.

2.3.3 Inference

We use the variational inference for the Dirichelet process mixture model presented by Blei and Jordan [2]. In practice Dirichlet Process inference algorithm is approximated and uses a truncated distribution with a fixed maximum number of components, say $K_{\max}$, of β and θ. Nevertheless, the number of components actually used K almost always depends on the data, and that $K \leq K_{\max}$.

2.3.4 Predictive Density

Based on model setting (20), model configuration $(\phi(\cdot), \alpha, G_0)$ and observed feature data points $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, we are able to make inference on the posterior distribution $\mathbb{P}(\theta|\mathbf{x}_1, \dots, \mathbf{x}_n, \alpha, G_0)$. Afterwards, we can compute the predictive density:

$$\mathbb{P}(\mathbf{x}|\mathbf{x}_1, \dots, \mathbf{x}_n, \alpha, G_0) = \int \phi(\mathbf{x}|\theta)\mathbb{P}(\theta|\mathbf{x}_1, \dots, \mathbf{x}_n, \alpha, G_0)\mathrm{d}\theta. \tag{22}$$

2.4 Modelling Pipeline

a table show each step, input and output

Feature extraction

Features distributional modelling

Fitting mixture model

Analytic (churn rate calculation, clusters interpretation)

3 Data Description and Pre-processing

3.1 Whizz Database

Imbalanced data - the reason we don't mak them balanced is because we do not want to twist numerical value of churn rate. For example, if we balance the data set perfectly, then the population churn rate will become 50%. After we do the clustering task, it is not straightforward to transform back the cluster churn rate for the balanced data set to that for the original imbalanced data set.

3.2 Feature Extraction

A table showing the definition of features.

3.3 Data Transformation

We denote the feature data by a matrix $\mathbf{X} = (x_{ij}) \in \mathbb{R}^{m\times n}$, which describes n observed values for each of the m features. In addition, we denote the i-th row of $\mathbf{X}$ by $\mathbf{x}_{i,\mathrm{R}} = [x_{i1}\ x_{i2}\ \cdots\ x_{in}]$, and the j-th column by $\mathbf{x}_j = [x_{1j}\ x_{2j}\ \cdots\ x_{mj}]^\top$.

In general, learning algorithms benefit from standardisation of the data set. We have employed 3 transformations in sequence to make our data more suitable for mixture model learning task. The transformations are performed for each feature separately, resulting in different sets of transformation parameters for different features. To be specific, for i-th feature we describe the transformations as following.

- Linear transformation
  The linear transformation is applied either to ensure all data to be positive for eligibility of applying the following power transformation, or to adapt to distributional modelling choice. It has the format:

$$\mathbf{x}'_{i,\mathrm{R}} = a_i\mathbf{1} + b_i\mathbf{x}_{i,\mathrm{R}}, \tag{23}$$

  where a_i and b_i are constants and $\mathbf{1} \in \mathbb{R}^{1\times n}$ is the row vector of all ones.

- Box-Cox power transformation
  The Box-Cox power transformation is used to modify the distributional shape

of a set of data to be more normally distributed so that tests and confidence limits that require normality can be appropriately used. It has the format:

$$x'_{ij} = \begin{cases} \frac{x_{ij}^{\lambda_i}-1}{\lambda_i} & \text{if } \lambda_i \neq 0, \\ \ln x_{ij} & \text{if } \lambda_i = 0, \end{cases} \tag{24}$$

for $j = 1,\ 2,\ \dots, m$. In Box-Cox transformation, λ_i is estimated by maximizing the likelihood function [reference].

- Standardisation
  We standardise data for different features by scaling them into the same range to ensure the robustness to very small standard deviations of features. We choose to scale all features into range $[1, 100]$. If we denote the maximum and minimum values of observed feature i as $x_i^{\max}$ and $x_i^{\min}$ respectively, then the standardisation is a linear transformation such that,

$$\mathbf{x}'_{i,\mathrm{R}} = \mathbf{1} + \frac{100-1}{x_i^{\max} - x_i^{\min}} \left(\mathbf{x}_{i,\mathrm{R}} - x_i^{\min} \mathbf{1} \right). \tag{25}$$

Table of transformation parameters

Exhibitions of pre and post transformation

4 Clustering Analysis

4.1 Distributional Modelling of Features

empirical distributional properties require simply Gaussian mixtures
correlation analysis enables to separate a few features to be fitted independently

4.2 Fitting Baysian Gaussian Mixture Model

brief description of EM-algorithm and Baysian inference

4.3 Assessing Churn Probability

(start of the supervised learning, because of the usage of churn label)
compute cluster churn rate
interpret churn probability of individual pupil

4.4 Feature Impact

4.5 Markov States Temporal Transition Analysis

4.5.1 Defining Markov States

4.5.2 Transitional Analysis

calculate transition probability

visualise transition (matrix + Sankey plot) [3]

5 Churn Probability Prediction

5.1 Prediction Workflow

5.2 Evaluating Overfitting

6 Conclusion

References

[1] D. Blackwell and J. MacQueen. Ferguson distributions via Polya urn schemes. *The Annals of Statistics*, 1:353–355, 1973.

[2] D. M. Blei and M. I. Jordan. Variational inference for dirichlet process mixtures. *Bayesian Anal.*, 1(1):121–143, 03 2006.

[3] P. Rothenbuehler, J. Runge, F. Garcin, and B. Faltings. Hidden markov models for churn prediction. In *2015 SAI Intelligent Systems Conference (IntelliSys)*, pages 723–730, Nov 2015.

[4] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[5] S. F. Slater and J. C. Narver. Intelligence generation and superior customer value. *Journal of the Academy of Marketing Science*, 28(1):120, Dec 2000.