INSTITUTO FEDERAL DO CEARÁ (IFCE)

LABORATÓRIO DE INTELIGÊNCIA ARTIFICIAL (LIA)

Relatório preliminar para o sistema de recomendação de próxima funcionalidade (Problema 1)

Ajalmar Rocha, Amauri Holanda, Atila Negreiros
Henrique Leitão, Luan Sousa, Victor Prata

10 de agosto de 2018

1 Introdução

Neste relatório são apresentadas as atividades realizadas até o presente momento no contexto do projeto SIGA. O escopo do projeto SIGA inclui o desenvolvimento de soluções para três problemas distintos, sendo abordado neste relatório o problema relacionado à recomendação de próxima funcionalidade/tela/*frame* (denominado daqui para frente de **problema 1**).

A necessidade de recomendação de funcionalidades/telas surge, por exemplo, em sistemas computacionais nos quais a utilização de tais sistemas ocorre através da navegação entre páginas/*frames*. Tal recomendação pode ser descrita de forma simplificada pela necessidade de recomendar a próxima tela com maior probabilidade de ser acessada por usuário. Esta capacidade de recomendação é especialmente relevante em sistemas com uma quantidade elevada de telas, sendo que tais sistemas de recomendações podem oferecer um aumento considerável na produtividade e aprendizagem do sistema por parte dos seus usuários.

Além desta introdução, o restante deste relatório está organizado como descrito a seguir. Na seção 2 é definido o problema de interesse, incluindo sua formulação matemática; enquanto na seção 3 é feita uma revisão bibliográfica de problemas semelhantes encontrados na literatura. Já nas seções 4 e 5 são descritas, respectivamente, uma formulação matemática para o problema e uma solução inicial com base em cadeias de Markov. Na seção 6 são descritas as tecnologias utilizadas juntamente com a arquitetura proposta. Na seção 7 é apresentado os resultados preliminares da solução inicial proposta. Por fim, na seção 8, são feitas algumas considerações finais seguida das referências.

2 O problema de recomendação de funcionalidades/telas

De modo simples, o problema consiste em recomendar próximas funcionalidades/telas a usuários de um sistema/produto qualquer, de tal maneira que a recomendação deva ser feita levando-se em consideração o uso do sistemas pelos usuários. Ou seja, nesse contexto,a sequência de acesso/utilização das telas é relevante e deve ser utilizada no processo de predição. Alternativamente, o problema pode ainda ser interpretado como um problema de predição de série temporal, em que cada predição consiste em uma ou mais recomendações de telas do sistema/produto em questão.

De uma forma mais específica e conforme definição acordada durante a elaboração do projeto SIGA, as recomendações poderão ser feitas em três níveis diferentes: (*i*) a nível de usuário (*user*), (*ii*) a nível de empresa (*company*) e (*iii*) a nível de produto/sistema (*system*). Do exposto, tem-se recomendação específica para usuário, na qual se leva em consideração o histórico de utilização do sistema realizado por um certo usuário. Tem-se ainda recomendação por empresa, opção na qual o histórico de utilização refere-se as ações realizadas pelos diversos usuários de uma empresa específica. Por fim, tem-se a recomendação por produto, em que o histórico de utilização refere-se as ações realizadas por todos os usuários das diversas empresas que utilizam um determinado produto.

3 Revisão bibliográfica

O problema de recomendação de telas se difere dos sistemas de recomendações clássicos principalmente pela componente temporal/sequencial. Nesse contexto, [6] apresenta uma ótima revisão de modelos e arquiteturas propostas para sistemas de recomendação baseados em sequências (*sequence-aware recommender systems*).

Dentre as abordagens mais comuns, destacamos modelos probabilísticos baseados na cadeia de Markov, tais como MDP (Markov Decision Processes) [7]; Árvores de Contexto [2], que correspondem a cadeias de Markov de ordem variável; e Molelos de Markov Ocultos [4], em que variáveis latentes são introduzidas para capturar dependências de mais longa duração.

Além disso, redes neurais recorrentes (*recurrent neural nets*, RNN) têm também sido aplicadas ao problema de interesse. Dentre os trabalhos mais relevantes, destacamos os recentes trabalhos [3], [5] que utilizam RNN e vizinhos mais próximos para recomendações baseadas em sessão, bem como o trabalho [8] para predição de cliques.

De um ponto de vista probabilístico, podemos descrever o problema de interesse como segue.

4 Formulação matemática

Matematicamente, o problema de interesse consiste na predição do frame $f_t \in \{1, \ldots, N_f\}$ no instante de tempo t a partir de frames anteriormente observados $f_{t-1}, f_{t-2}, \ldots, f_1$ para um determinado usuário u. Na verdade, informação adicional sobre produtos e empresas também serão utilizadas, situação que será abordada adiante.

Antes de descrever a formulação do problema, precisamos definir a notação. A partir de agora neste documento, letras maiúsculas vão denotar variáveis aleatórias (VAs), enquanto letras minúsculas representam atribuições das variáveis aleatórias correspondentes. A função massa de probabilidade (probability mass function, PMF) é denotada por $p(\cdot)$. Assim, para uma variável aleatória discreta X, sua PMF é denotada por $p(x)$. A letra P maiúscula vai denotar a probabilidade de um evento, de modo que, para variáveis aleatórias discretas, $p(x) = P(X = x)$. Casos multivariados serão representados em negrito, e.g, conjunto de variáveis aleatórias são denotados por $\mathbf{X} = \{X_1, X_2, \ldots, X_n\}$, tal como suas atribuições $\mathbf{x}$.

Assuma que F_t representa uma VA discreta que denota o frame no instante t e que F_t assume um valor qualquer no conjunto $\mathcal{F} = \{1, \ldots, N_f\}$, em que N_f é o número total de

frames. De modo similar, assuma que U é uma VA representando usuários e assumindo valores em $\mathcal{U} = \{1, \ldots, N_u\}$. Considere ainda que os produtos/sistemas são representados por uma VA S, onde cada atribuição consiste em uma valor em $\mathcal{S} = \{1, \ldots, N_s\}$. Por fim, empresas são denotas por E, tal que cada atribuição $e \in \mathcal{E} = [1, \ldots, N_e]$

Nós estamos interessados em encontrar K recomendações, definidas por

$$\begin{aligned}
\hat{f}_t^{(1)} &= \arg\max_{f_t \in \mathcal{F}} P(F_t = f_t \mid F_{t-1} = f_{t-1}, \ldots, F_1 = f_1, U = u, S = s, E = e), \\
\hat{f}_t^{(2)} &= \arg\max_{f_t \in \mathcal{F} \setminus \{\hat{f}_t^{(1)}\}} P(F_t = f_t \mid F_{t-1} = f_{t-1}, \ldots, F_1 = f_1, U = u, S = s, E = e), \\
&\vdots \\
\hat{f}_t^{(K)} &= \arg\max_{f_t \in \mathcal{F} \setminus \{\hat{f}_t^{(1)}, \ldots, \hat{f}_t^{(K-1)}\}} P(F_t = f_t \mid F_{t-1} = f_{t-1}, \ldots, F_1 = f_1, U = u, S = s, E = e).
\end{aligned}$$

Observe que o valor K é uma parâmetro que o usuário pode definir. Ele representa o tamanho do *ranking* de recomendações. O valor $\hat{f}_t^{(1)}$ representa a primeira escolha mais provável para a tela no instante t; de modo similar, $\hat{f}_t^{(k)}$ denota a k-ésima opção no *ranking* de recomendações para a tela no instante t.

Observe que o termo $P(F_t = f_t \mid F_{t-1} = f_{t-1}, \ldots, F_1 = f_1, U = u, S = s, O = o)$, ou simplesmente $p(f_t \mid f_{t-1}, \ldots, f_1, u, s, e)$, denota a probabilidade do frame no instante t dado os valores dos frames em todos os instantes anteriores, o usuário, o sistema e a empresa. Essa especificação a nível de usuário consiste na recomendação mais granular possível. A partir do cálculo de probabilidades no nível mais granular, podemos então encontrar as probabilidades para quaisquer nível de especificação (seja do produto ou empresa) utilizando um procedimento simples chamado de *marginalização* — neste, os valores das variáveis que não são de interesses são eliminadas através de operações de somas.

Como exemplo, caso estejamos interessados em realizar a predição especificando somente o produto S e a empresa E, a predição $\hat{f}_t^{(1)}$ seria dada por

$$\begin{aligned}
\hat{f}_t^{(1)} &= \arg\max_{f_t \in \mathcal{F}} p(f_t \mid f_{t-1} \ldots, f_1, s, e), \\
&= \arg\max_{f_t \in \mathcal{F}} \sum_{u \in \mathcal{U}} p(f_t, u \mid f_{t-1}, \ldots, f_1, s, e) \\
&= \arg\max_{f_t \in \mathcal{F}} \sum_{u \in \mathcal{U}} p(f_t \mid f_{t-1}, \ldots, f_1, u, s, e) p(u \mid f_{t-1}, \ldots, f_1, s, e).
\end{aligned}$$

5 Solução inicial proposta: cadeias de Markov

A solução inicial empregada pela equipe SIGA/IFCE foi a utilização de cadeias de Markov de ordem 1. Mais especificamente, uma cadeia de Markov de tempo e espaço de estado discretos, em que cada estado representa uma possível tela do *framework*.

Formalmente, uma sequência de variáveis aleatórias $X_0, X_1, \ldots$ assumindo valores no espaço de estados $\{1, 2, \ldots, M\}$ é chamada uma cadeia de Markov se para todo $n \geq 0$,

$$P(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) = P(X_{n+1} = j | X_n = i). \tag{5.1}$$

O termo $P(X_{n+1} = j | X_n = i)$ denota a *probabilidade de transição* do estado i para o estado j. Além disso, nós assumimos que a cadeia é homogênea, ou seja, $P(X_{n+1} = j | X_n = i)$ é o mesmo para todo n.

Basicamente, a solução empregada utiliza a propriedade de Markov para simplificar o problema de recomendação de telas, através da consideração de que dado o histórico anterior de telas $F_{n-1}, \ldots, F_1$, somente a última variável, F_{n-1}, contribui na predição da tela no instante seguinte F_n.

Vale ainda ressaltar que o termo $P(X_{n+1} = j|X_n = i)$ pode ser genericamente representado por uma matriz, chamada *matriz de transição*. No caso de M estados possíveis, a matriz de transição $\mathbf{Q}$ é uma matriz $M \times M$ dada por $[q_{ij} = P(X_{n+1} = j|X_n = i)]$. Com isso, $\mathbf{Q}$ é uma matriz não-negativa cuja soma dos elementos de cada linha é igual a 1.

A aprendizagem da matriz de transição a partir de dados pode ser feita utilizando por exemplo o *príncipio da máxima verossimilhança*. Nesse caso, os parâmetros da cadeia, os valores de probabilidade $P(X_{n+1} = j|X_n = i)$, são dados por contagem, ou seja,

$$P(X_{n+1} = j|X_n = i) = \frac{\#[X_{n+1} = j, X_n = i]}{\#[X_n = i]}.$$

O termo $\#[X_{n+1} = j, X_n = i]$ representa a quantidade de vezes que a sequência $[j, i]$ foi encontrada nos dados, enquanto $\#[X_n = i]$ denota quantas vezes o estado i foi encontrado.

A aplicação de cadeias de Markov para o problema de recomendação de telas é direta. O termo $p(f_t \mid f_{t-1}, \ldots, f_1, u, s, e)$ é então interpretado pela probabilidade de transição $P(F_t = f_t \mid F_{t-1} = f_{t-1})$ cujo espaço de estados é o conjunto de telas $\mathcal{F} = \{1, \ldots, N_f\}$. Observe no entanto que deve ser criada uma matriz de transição para cada usuário, sistema e empresa, ou seja, teremos uma matriz de transição indexada por u, s e e — $\mathbf{Q}(u, s, e)$.

A seguir são descritas as tecnologias utilizadas para armazenamento dos dados no modulo de registo e *big data*, bem como para o sistema de recomendação como um todo.

6 Tecnologias Utilizadas

6.1 MongoDB

O MongoDB é um banco de dados NoSQL e *open source* que utiliza um modelo de dados orientado a documentos. Enquanto os bancos relacionais aplicam o conceito de tabelas e linhas, a arquitetura do MongoDB é baseada em coleções e documentos. Documentos compreendem conjuntos de pares {*chave*:*valor*} e são a unidade básica de dados no MongoDB. Coleções contêm conjuntos de documentos e funcionam como o equivalente as tabelas dos bancos de dados relacionais.

Dessa forma, o MongoDB tem a capacidade de armazenar dados em documentos flexíveis (em formato JSON), de modo que os campos podem variar de documento para documento e com a facilidade de que a estrutura de dados possa ser alterada facilmente ao longo do tempo. Além dessa flexibilidade, o MongoDB é um banco de dados altamente escalável, auto-fragmentável (*auto-sharding*), possibilita *queries* dinâmicas similares a linguagem SQL e mapeia os objetos de uma aplicação nos objetos de banco de dados.

Atualmente, os dados que estamos analisando é baseado na coleção previamente estabelecida (*FeatureTimeLine*) do banco de dados *SigaData* . Como base em discussão realizada em reunião ocorrida em 08/06/2018, construímos uma coleção com dados simulados (mocados) para avaliação dos modelos iniciais do problema 1. Os dados simulados estão relacionados a 30 telas, 10 usuários e 3 produtos, totalizando 651.666 documentos. Cada documento da coleção contém a seguinte estrutura:

- **_id**: identificador do documento gerado pelo próprio MongoDB. Cada documento na coleção possui um identificador único.
- **Cnpj**: número do Cnpj da empresa, de modo que os dados do documento pertencem/correspondem a uma empresa com respectivo Cnpj.
- **Product**: corresponde a um *software* com finalidade específica. Cada *software* contém um conjunto de telas.
- **User**: corresponde ao usuário de uma empresa (determinada pela chave **Cnpj**) que gerou os dados contidos no documento.

- **Machine**: corresponde ao nome da máquina em que um determinado usuário estava logado ao gerar os dados.
- **Feature**: refere-se a tela/*frame* em que um usuário realizou uma tarefa específica. Através da informação contida nessa chave e nas chaves **StartTime** e **EndTime**, pode-se construir modelos de recomendação para um usuário específico, produto e empresa.
- **StartTime**: corresponde a data em que uma tela começou a ser utilizada por um usuário específico, em uma máquina específica de uma determinada empresa.
- **EndTime**: corresponde a data em que um usuário saiu da tela.

6.2 Azure

Dentre as diversas opções que existem atualmente no mercado, uma que se mostra bem interessante é o Microsoft Azure. Fato reconhecido pela Gartner, que pelo quinto ano consecutivo elegeu o Azure como líder em Infraestrutura de Nuvem como Serviço (IaaS), Serviços de Armazenamento em Nuvem e Plataforma de Aplicativos como Serviço (PaaS) [1]. As vantagens oferecidas pelo Azure são muitas, com maior valor de destaque para:

- Alta disponibilidade: Ao contrário de outros provedores, a nuvem do Microsoft Azure oferece alta disponibilidade e redundância em data centers em escala global. Por esse motivo, o Azure pode oferecer um contrato de nível de serviço (SLA) de 99,95% (aproximadamente 4,38 horas de tempo de inatividade por ano), algo que a maioria das empresas não consegue atingir.
- Segurança: O Microsoft Azure tem um forte foco em segurança, seguindo o modelo de segurança padrão de Detectar, Avaliar, Diagnosticar, Estabilizar e Fechar. Juntamente com controles de segurança cibernética, este modelo permitiu que o Azure obtivesse várias certificações de conformidade, todas as quais estabelecem o Azure como líder em segurança de IaaS. Não apenas a plataforma é protegida, o usuário final também é coberto pelo Azure. Esse nível de proteção em vários níveis é essencial, pois as ameaças de segurança continuam se multiplicando diariamente no mundo todo, segmentando usuários finais e colocando em risco os dados da sua empresa. O Azure fornece serviços simples e fáceis de usar para maior proteção, como autenticação de vários fatores e requisitos de senha do aplicativo.
- Escalabilidade: O Microsoft Azure facilita a ampliação ou redução da capacidade de computação com nada mais que o clique de um botão. Com essa estrutura de escalabilidade, as empresas têm a flexibilidade de pagar apenas pelo que usam.
- Custo por Eficácia: Evidentemente, é essencial manter os orçamentos de TI em mente ao escolher um provedor de nuvem, e é por isso que a plataforma Microsoft Azure é tão atraente para muitas organizações. O preço pré-pago do Azure permite que as SMB gerenciem melhor seus orçamentos de TI, adquirindo apenas o quanto precisarem. Além disso, o ambiente de nuvem permite que as empresas iniciem aplicativos de clientes e aplicativos internos na nuvem, o que economiza custos de infraestrutura e reduz os encargos de hardware e manutenção no gerenciamento de TI interno.

6.3 Arquitetura do Sistema

A arquitetura escolhida para fornecimento do serviço de recomendação de telas pode ser visto na Figura 6.1.

Dentre as opções disponibilizadas pelo Azure para construir aplicações WEB, o escolhido foram os *Web Apps* pelas seguintes razões: (*i*) facilidade de implantação, além de fazê-lo com

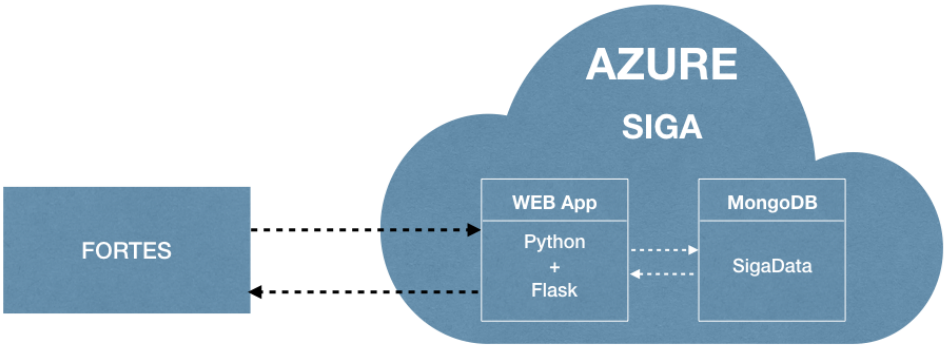

Figura 6.1: Arquitetura de comunicação entre as empresas do grupo FORTES e o *webservice* do SIGA, ao solicitar recomendações.

mais competência e velocidade que outros serviços em nuvem; (*ii*) capacidade de agregação de múltiplas aplicações (com o intuito de economizar recursos); (*iii*) auto-escalabilidade sem necessidade de reimplantação; (*iv*) configuração de tarefas agendadas através de *web jobs*; e (*v*) suporte para múltiplas linguagens de programação *open source*, tal como Python (muito utilizada atualmente para desenvolvimento de algoritmos de aprendizagem de máquianas).

Python é a linguagem de programação escolhida para o desenvolvimento da aplicação, tanto em sua parte acadêmica (modelos de aprendizagem de máquina) quanto em seu viés tecnológico (*webservice*). Entre as principais razões, pode-se citar: (*i*) facilidade de aprendizado; (*ii*) possui uma comunidade de programadores muito ativa; (*iii*) sintaxe organizada, resultando em facilidade de leitura/compreensão; e (*iv*) por ser uma das linguagens mais populares para desenvolvimento científico, contendo diversas bibliotecas dentre as mais usadas no mundo.

O desenvolvimento de uma aplicação web pode ter sua produtividade aumentada quando utiliza-se *frameworks*. Isso porque tais ferramentas possuem muitas tarefas intrínsecas já prontas para uso, reduzindo assim o tempo de desenvolvimento bem como permitindo um foco maior na aplicação. Flask é a escolha para o projeto SIGA. Por ser um *microframework*, possui um *core* simples mas extensível. Por consequência, é uma escolha que nos permite trabalhar com mais aprofundamento em otimizações que são dependentes do problema.

De uma forma geral, as requisições são feitas ao *webservice* hospedado no Azure, que busca na coleção de predições geradas pelo treinamento dos modelos e retorna o que foi solicitado. Exemplo: caso seja solicitado por um usuário as recomendações das 5 telas mais prováveis a partir de sua tela atual, o *webservice* retorna um conjunto de tuplas {(tela, probabilidade desta ser a próxima tela)}. Alternativamente, pode ser devolvido uma matriz de transição, para o caso do uso de cadeias de Markov, para o sistema hospedeiro do sistema de recomendação a fim de evitar múltiplas requisições ao Azure.

Uma tarefa de treinamento para um certo modelo deve ser agendada, por exemplo, às 02h00 todos os dias para fins de atualização do mesmo com as novas informações armazenadas no dia imediatamente anterior. Assim, esse procedimento visa coletar da base SigaData todos os dados do dia anterior e fará com que os modelos de aprendizagem retornem as telas mais prováveis para os três níveis de recomendação citados anteriormente (predições por usuário, com produto e empresa específicos; predições por empresa, abrangendo todos os usuários de empresa para um produto específico; e predições por produto, abrangendo todos os usuários de todas as empresas clientes do grupo FORTES).

7 Resultados Obtidos

7.1 Resultados - Macroentrega 1

O modelo de dados e a infraestrutura foram definidos e já são capazes de armazenar grande quantidade de dados (*big data*). Atualmente temos 8627440 de registros já coletados de dados

reais. Essa quantidade demonstra a capacidade do modelo em termos de armazenamento. A Figura 7.1 exibe exemplos de dados reais (em formato JSON) armazenados na coleção *FeatureTimeLine*.

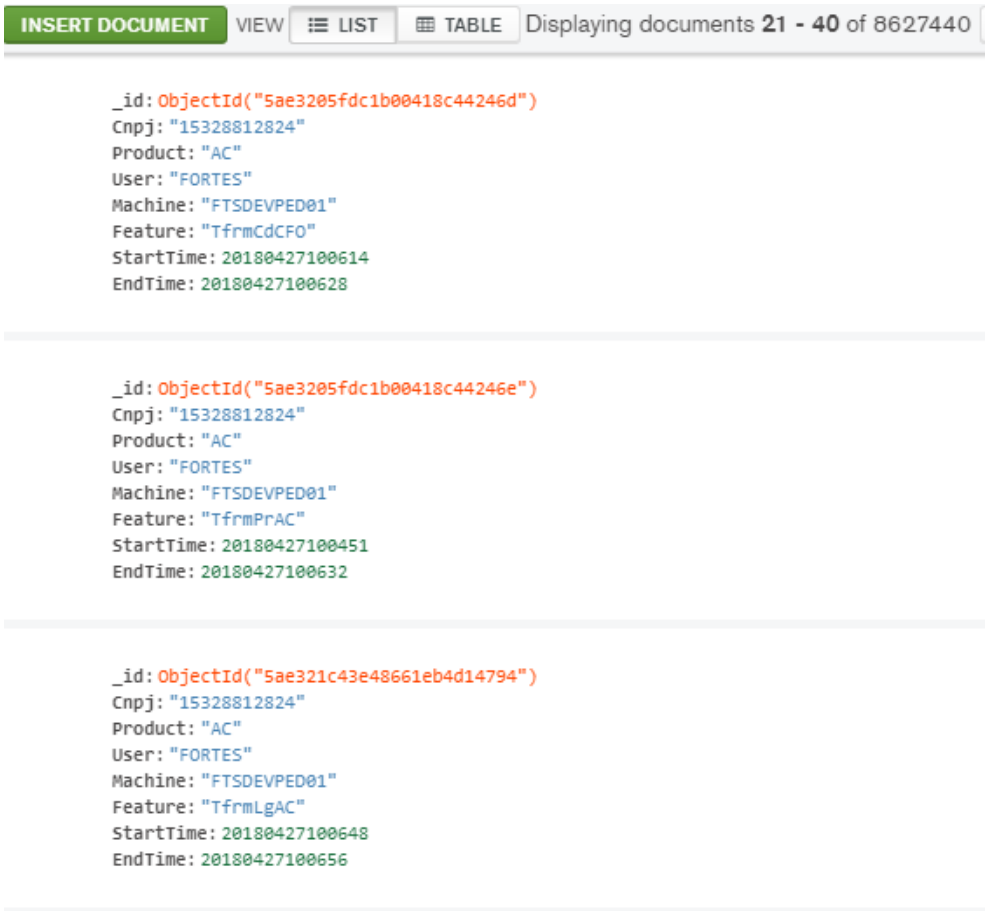

Figura 7.1: Exemplo de registros coletados na coleção *FeatureTimeLine* do banco *SigaData*.

7.2 Resultados preliminares - Macroentrega 2

A avaliação dos modelos baseados em Cadeias de Markov de ordem 1 foi realizada através dos dados simulados (mocados) com estrutura semelhante a coleção FeatureTimeLine do banco de dados SigaData.

Em um experimento com dados simulados de fluxos com padrões de acessos mais aleatórios, os modelos para recomendação da próxima tela de um usuário específico apresentaram uma taxa de acerto média de 88,78% (conforme Tabela 7.2) quando aplicado um *rank* de tamanho 5. O *rank* representa a quantidade de telas recomendadas pela sistema. Se um determinado usuário clicou em um tela sugerida pelo sistema de recomendação isso é reportado como um acerto, caso contrário, um erro. Para o *rank* mínimo, de tamanho 1, a média de acerto foi em torno de 45%.

Os modelos do sistema de recomendação da próxima tela para um produto específico resultaram em 17,83% de acerto para um *rank* de tamanho 5 e 2,72% para um *rank* de tamanho 1.

Em outro experimento com dados mocados representando um fluxo com padrões de acessos mais recorrente, obtemos valores expressivos para o sistema de recomendação. Com $rank = 5$, os modelos resultaram em taxa de acerto de 100% na recomendação da próxima tela para um usuário específico e 94,28% para recomendação de telas de um produto em específico da empresa. No *rank* mínimo, ou seja, de tamanho 1, o sistema de recomendação de telas para usuário obteve taxa de média em torno de 72% e para o produto em torno de 19%, conforme

Rank	Média usuário	Média produto
1	45,48%	2,72%
2	65,22%	6,19%
3	76,19%	10,13%
4	83,84%	12,09%
5	88,78 %	17,83%

Tabela 7.1: Modelo Cadeia de Markov de ordem 1 para base de dados com padrões de acessos mais aleatórios

pode ser visto na Tabela 7.2.

Rank	Média usuário	Média produto
1	71,95%	19,56%
2	91,49%	41,72%
3	97,09%	69,40%
4	99,44%	81,58%
5	100 %	94,28%

Tabela 7.2: Modelo Cadeia de Markov de ordem 1 para base de dados com padrões de acessos mais recorrente

A avaliação dos modelos baseados em Cadeias de Markov de ordem 2 foi realizada por meio dos dados mocados representando um fluxo com padrões de acessos aleatórios. Conforme a Tabela 7.3, os modelos com Cadeias de Markov de ordem 2 apresentaram resultados similares aos modelos de Cadeias de Markov de ordem 1 (Tabela 7.2). Como pode ser visto, os modelos para recomendação da próxima tela de um usuário específico apresentaram uma taxa de acerto média de 88,16% quando aplicado um *rank* de tamanho 5 e uma média de acerto de 50,27% para o *rank* de tamanho 1.

Os modelos com Cadeias de Markov de ordem 2 do sistema de recomendação da próxima tela para um produto específico resultaram em 14,41% de acerto para um *rank* de tamanho 5 e 12,17% para um *rank* de tamanho 1.

Rank	Média usuário	Média produto
1	50,27%	12,17%
2	66,44%	12,95%
3	76,54%	13,47%
4	83,37%	13,97%
5	88,16 %	14,41%

Tabela 7.3: Modelo Cadeia de Markov de ordem 2 para base de dados com padrões de acessos mais aleatórios

8 Considerações finais

O armazenamento de dados está sendo feito em nuvem Azure, sendo que a quantidade de informação já ultrapassa 8 milhões de registros (*big data*) relacionados ao uso de funcionalidades pelo usuário em um determinado sistema.

No tocante a modelagem do problema abordada atualmente que envolve a utilização de cadeias de Markov, consideramos que a mesma se faz bastante adequada e contém

características interessantes. Dentre elas vale ressaltar o processo de decisão, o qual pode ser realizado pelo uso de matrizes de transição (o que permite diminuir o número de requisições a Azure e uma consequente redução de custo).

Os resultados preliminares obtidos para dados simulados são bastante promissores. Vale ressaltar que simulações computacionais sobre os dados reais coletados das diversas empresas estão em processo de realização para avaliação dos modelos.

Referências

[1] Gartner. Gartner's magic quadrant for infrastructure as a service. `https://azure.microsoft.com/pt-br/resources/gartner-iaas-magic-quadrant/en-us/`, 2018. Online; accessed 12 July 2018.

[2] Q. He, D. Jiang, Z. Liao, S. C. H. Hoi, K. Chang, E. P. Lim, and H. Li. Web query recommendation via sequential query prediction. In *2009 IEEE 25th International Conference on Data Engineering*, pages 1443–1454, 2009.

[3] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. *CoRR*, 1511.06939, 2015.

[4] M. Hosseinzadeh Aghdam, N. Hariri, B. Mobasher, and R. Burke. Adapting recommendations to contextual changes using hierarchical hidden markov models. In *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, pages 241–244, 2015.

[5] D. Jannach and M. Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, pages 306–310, 2017.

[6] M. Quadrana, P. Cremonesi, and D. Jannach. Sequence-aware recommender systems. *CoRR*, 1802.08452, 2018.

[7] G. Shani, D. Heckerman, and R. I. Brafman. An mdp-based recommender system. *J. Mach. Learn. Res.*, 6:1265–1295, 2005.

[8] Y. Zhang, H. Dai, C. Xu, J. Feng, T. Wang, J. Bian, B. Wang, and T.-Y. Liu. Sequential click prediction for sponsored search with recurrent neural networks. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1369–1375, 2014.