

Projeto Livre:

Avaliação do perfil nas redes sociais de estudantes
e associados da UTFPR utilizando Análise Associativa

Daniel A. P. de Castro

danielcastro@alunos.utfpr.edu.br - (41) 99777-1037

Gustavo F. Armênio

gustavofardoarmenio@alunos.utfpr.edu.br - (49) 99810-7140

Resumo

Este documento se refere ao projeto livre da disciplina de Introdução à Modelagem e Aprendizado, com a finalidade de apresentar uma aplicação dos algoritmos de Descoberta de Associações para o escopo da interação entre os estudantes da UTFPR com as redes sociais, no que tange a motivação referente ao impacto das influências políticas e ideológicas dos estudantes. O trabalho contou com inúmeras ferramentas de coleta e processamento de dados, além de bibliotecas de aprendizado em máquina para ajudar no desenvolvimento. No fim, houve resultados precisos os quais puderam determinar estatisticamente as correlações entre simpatizantes de cursos com entidades influentes no *Instagram*.

Palavras-chave — Associações, Influências, Redes Sociais, Estudantes, UTFPR

1 Introdução

Apesar da importância da Universidade Tecnológica Federal do Paraná para inovações científicas e tecnológicas no Brasil, é perceptível uma disparidade de ideais culturais, políticos e sociais dos estudantes que a compõe quando comparada com outras instituições federais. Para fins de exemplo, observou-se recentemente a falta de engajamento na manifestação de reivindicação por ambulatórios médicos nas universidades [1]. Isso motivou o questionamento referente ao quão politizados são os alunos da UTFPR, além de procurar perfis de redes sociais que podem influenciar seus devidos comportamentos, já que estão vinculados com suas bases ideológicas e socioculturais [2].

Para tanto, foi utilizada uma ferramenta de Análise Associativa de Aprendizado Não-Supervisionado com os dados dos perfis seguidos por quem segue a UTFPR no *Instagram*. Foram coletados dados com a ajuda da biblioteca *instaloader* [3] do *Python*. Os dados se organizaram utilizando *pandas* [4] a fim de manipular com facilidade listas e arquivos *csv*. Para executar as ferramentas de Análise Associativa foi importado o Framework *mlxtend* [5], o qual contém os algoritmos *apriori*, *fpgrowth* e *hmine*. Os próprios algoritmos auxiliaram na extração de filtros os quais eliminam dados de ruído, além de que foram selecionados os perfis associados à UTFPR para avaliação de cada.

2 Métodos

Tal como previamente mencionado, o projeto contou com a disposição de inúmeras bibliotecas em *Python* para auxílio nas fases desde a coleta de dados até o pós processamento. O Diagrama de classes em UML da Figura 1 ilustra o projeto do sistema com suas respectivas funcionalidades.

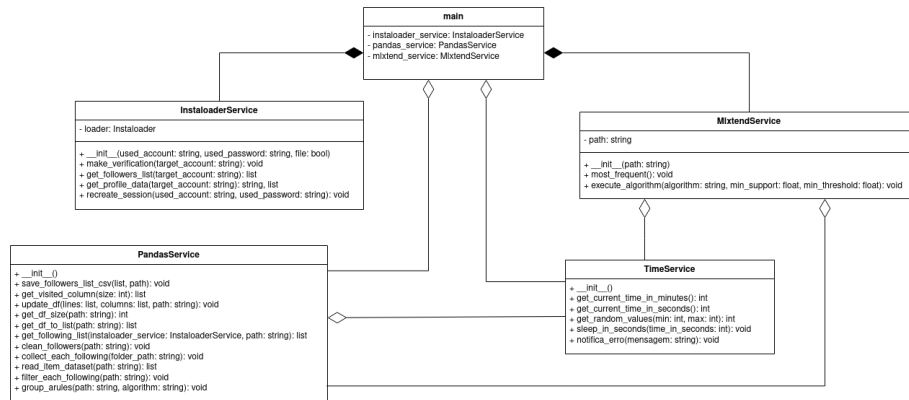


Figura 1: Diagrama de Classes UML do projeto.

2.1 Instaloader

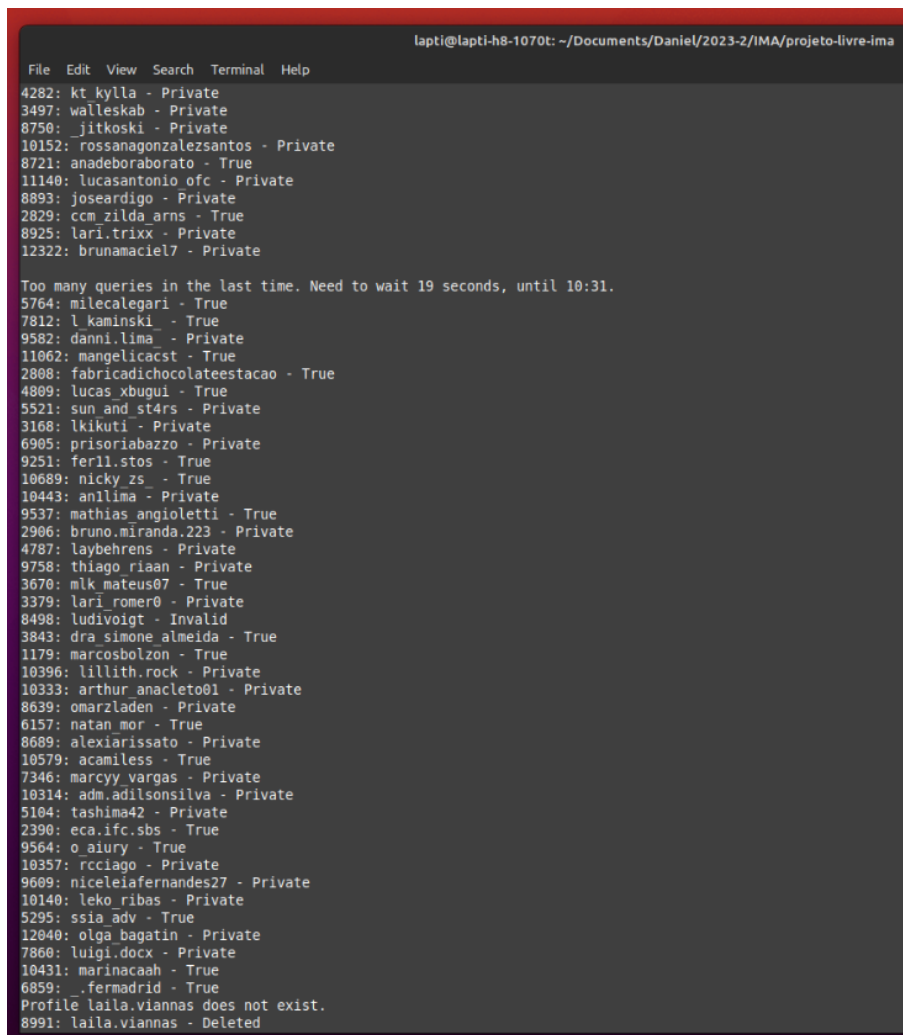
A biblioteca do *instaloader* providencia ferramentas de análise de perfil por meio de funções que rodeiam carregar dados de uma conta-alvo. Um dos requisitos para uso dessas funcionalidades é estar com uma sessão aberta no *Instagram*. Um dos problemas encontrados durante esse processo foi o sistema de detecção de bots do próprio *Instagram* o qual derruba perfis com atividades suspeitas. Por essa razão, foram testadas e adotadas diversas estratégias para garantir a coleta mais eficiente possível de dados. No final das contas, descobriu-se que o melhor método era criar vários perfis falsos no *Instagram* e ir alternando entre eles para acessar o *instaloader* à medida que as ações levantassem suspeitas.

Considerando que não há possibilidade de coletar dados de perfis privados e que existem perfis com número grande de usuários seguidos, fez-se necessário filtrar quais dos dados eram privados ou inválidos para a análise. Para tanto, dispõe-se listar os perfis já conferidos com um rótulo que descrevia "True", "Private" e "Invalid". Dos 12484 perfis que seguem a UTFPR, foram visitados 3324 (cerca de 25% so total), sendo 1279 (cerca de 10% do total) perfis com dados válidos coletáveis (sendo os demais privados ou de quantidade de usuários que seguem maior que 2000).

2.2 Pré-processamento

Uma vez coletada a lista de quem segue o campus Curitiba da UTFPR, a biblioteca *pandas* gerou um *dataframe* para salvar em um arquivo *csv* por meio do método *save_followers_list_csv*. Com o método *get_following_list* foram iterados aleatoriamente os perfis pertencentes aos salvos, de modo que se verificassem um por um, salvando os dados (cada perfil em uma planilha da pasta *data_files/each_following*) e atribuindo

uma marcação, tal como mostra na Figura 2. Para gerar valores aleatórios e notificar mensagens de erro eventuais, implementou-se na classe `TimeService` ferramentas diversas que auxiliassem tal processo.



```
lapti@lapti-h8-1070t: ~/Documents/Daniel/2023-2/IMA/projeto-livre-ima
File Edit View Search Terminal Help
4282: kt.kylla - Private
3497: wallekabb - Private
8750: jtkoski - Private
10152: rossanagonzalezsantos - Private
8721: anadeboraborato - True
11140: lucasantonio.ofc - Private
8893: joseardigo - Private
2829: ccm.zilda.arns - True
8925: lari.trixx - Private
12322: brunamaciel7 - Private

Too many queries in the last time. Need to wait 19 seconds, until 10:31.
5764: milecalegari - True
7812: l.kaminski - True
9582: dannilima - Private
11062: mangelicacst - True
2808: fabricadichocolateestacao - True
4809: lucas.xbugui - True
5521: sun.and.st4rs - Private
3168: lkikuti - Private
6905: prisoriabazzo - Private
9251: ferll.stos - True
10689: nicky.zs - True
10443: anllima - Private
9537: mathias.angioletti - True
2906: bruno.miranda.223 - Private
4787: laybehrens - Private
9758: thiago.riaan - Private
3670: mlk.mateus07 - True
3379: lari.romer0 - Private
8498: ludivoigt - Invalid
3843: dra.simone.almeida - True
1179: marcosbolzon - True
10396: lillith.rock - Private
10333: arthur.anacleto01 - Private
8639: omarzladen - Private
6157: natan.mor - True
8689: alexiarissato - Private
10579: acamiless - True
7346: marcy.vargas - Private
10314: adm.adilsonsilva - Private
5104: tashima42 - Private
2390: eca.ifc.sbs - True
9564: o.aiury - True
10357: rcciago - Private
9609: niceleiafernandes27 - Private
10140: leko.ribas - Private
5295: ssia.adv - True
12040: olga.bagatin - Private
7860: luigi.docx - Private
10431: marinacaah - True
6859: .fermadrid - True
Profile laila.viannas does not exist.
8991: laila.viannas - Deleted
```

Figura 2: Rótulos de perfis encontrados aleatoriamente.

Após a extração do valor de 1279 instâncias, executou-se o método da classe *Pandas* `collect_each_following` com o qual salva os itens presentes em cada arquivo em apenas um arquivo de Dataset (*all_profiles.csv*). A partir disso, já foi utilizada a biblioteca do *mlxtend* para filtrar os itens mais frequentes desse Dataset (gerando o arquivo *frequent.csv*), e remover os que não pertenciam a esse grupo com o método `filter_each_following`, o qual gera de saída o arquivo *all_filtered_profiles.csv*. A escolha de itens frequentes utilizou um suporte mínimo de 1% e o algoritmo *hmine* devido a sua maior eficiência (que será discutida mais adiante).

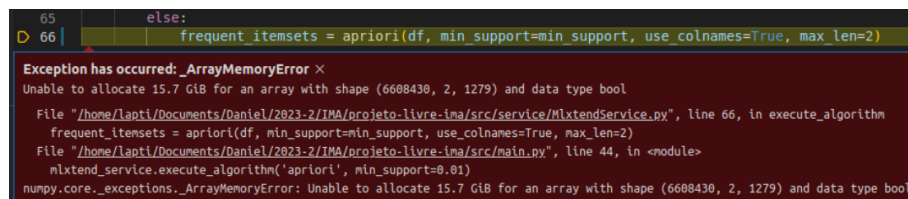
2.3 Algoritmos

Os Algoritmos de Descoberta de Associações possuem como princípio canônico encontrar os subconjuntos de N elementos mais frequentes em um Dataset de itens. Para tanto, é realizado um procedimento complexo que determina e contabiliza combinações limitadas por um valor mínimo de Suporte e Confidência. O Suporte se refere à quantidade de vezes que é encontrado um subconjunto em relação à quantidade total de instâncias do Dataset. Já a Confidência de um item em um subconjunto diz respeito ao tanto de vezes que esse item aparece quando os demais desse subconjunto são instanciados. Com isso, é possível determinar Regras de Associação (RA) formando uma tabela de Antecedente e Consequente de um item, conforme o tamanho especificado. No caso da análise em questão, é conveniente utilizar o tamanho de dois elementos para as RA, já que é feita uma avaliação entre pares de perfis, e não necessariamente subconjuntos com mais que dois elementos.

No código em questão, a classe `MlxtendService` desempenha um papel genérico para os três algoritmos. Ao executar o método `execute_algorithm` é passado de parâmetro o algoritmo selecionado (sendo o *Apriori* de *default* e conforme o algoritmo são salvas as Regras de Associação em um *csv* contendo o seu nome.

2.3.1 Apriori

O primeiro algoritmo a ser desenvolvido para determinar itens frequentes foi o *Apriori*. Sua abordagem é "de baixo para cima", começando com subconjuntos de 1 elemento a serem contabilizados conforme o suporte mínimo fornecido, e escalonando o tamanho do subconjunto com a inserção de uma unidade de elemento por vez. Sua complexidade computacional é de ordem exponencial, já que a quantidade de combinações de subconjuntos possíveis dentro de um conjunto de itens é de $O(2^n)$. Ao executar o algoritmo para os dados pré-processados com um suporte de 1%, houve problema de sobrecarga de memória, vide Figura 3 fazendo com que só se pudesse executar com valores de suporte acima de 2%. [6]



```
65         else:
66             frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True, max_len=2)

Exception has occurred: _ArrayMemoryError ×
Unable to allocate 15.7 GiB for an array with shape (6688430, 2, 1279) and data type bool
File ~/home/lapti/Documents/Daniel/2023-2/IJA/projeto-livre-ima/src/service/MlxtendService.py, line 66, in execute_algorithm
    frequent_itemsets = apriori(df, min_support=min_support, use_colnames=True, max_len=2)
File ~/home/lapti/Documents/Daniel/2023-2/IJA/projeto-livre-ima/src/main.py, line 44, in <module>
    mlxtend_service.execute_algorithm('apriori', min_support=0.01)
numpy.core._exceptions._ArrayMemoryError: Unable to allocate 15.7 GiB for an array with shape (6688430, 2, 1279) and data type bool
```

Figura 3: Erro de sobrecarga de memória Apriori.

2.3.2 Fpgrowth

A partir do *Apriori* houve a intenção de aprimorar o valor assintótico de complexidade das operações, motivando a criação de algoritmos como o *Fpgrowth*, o qual adapta a lógica do *Apriori* ao representar os dados em uma estrutura de árvore. Por meio disso, é otimizada a busca de elementos em subconjuntos devido à disposição em nós.

2.3.3 Hmine

Já o algoritmo *Hmine* utiliza estruturas de *hyper-links* (*H-struct*), as quais acabam sendo mais eficientes ainda do que as árvores do *Fpgrowth*. Na Figura 4 há um comparativo do tempo de execução dos dois algoritmos (usando a classe *TimeService*). [7]

```
Iniciando fpgrowth
Algoritmo fpgrowth finalizado. Tempo total: 86.487 segundos.
Iniciando hmine
Algoritmo hmine finalizado. Tempo total: 64.523 segundos.
lapti@lapti-h8-1070t:~/Documents/Daniel/2023-2/IMA/projeto-livre-ima$
```

Figura 4: Comparação entre tempo do *Fpgrowth* e *Hmine*.

2.4 Pós Processamento

Após o salvamento das regras de associação em seus respectivos arquivos *csv*, a classe *PandasService* utiliza o último método de filtro para agrupar as regras conforme os perfis de entidades universitárias pré-selecionados. Esse método (*group_arules*) seleciona em uma lista as regras que utilizam o perfil-alvo e elimina todos os outros perfis do grupo para que não exista ruído de interrelações dessas entidades, visto que já é de se esperar haver associações entre perfis dentro do mesmo escopo. Assim, gerou-se de resultado final todos os arquivos *csv* dentro da pasta *grouping_arules*, de modo que os dados estejam prontos para análise.

3 Análise

A análise de dados permitiu concluir que ao todo havia regras de associações envolvendo 7 figuras políticas relevantes (com suas respectivas quantidades de regras): lulaoficial (PT) - Presidente da República - 18 RA; caroldartora13 (PT) - Deputada Federal PR - 8 RA; hilton_erika (PSOL) - Deputada Federal SP - 5 RA; renatofreitasumdenos (PT) - Deputado Estadual PR - 2 RA; guilhermeboulos.oficial (PSOL) - Deputado Federal SP - 1 RA; rafaelgrecaooficial (PSD) - Prefeito de Curitiba - 1 RA; jairmessiasbolsonaro (PL) - Ex-Presidente da República - 1 RA.

Outras seleções específicas foram realizadas, como grupos de perfis de engajamento político na universidade: abelhas.utfpr - 4 RA; econsultoriautfpr - empresa júnior de consultoria ecológica - 3 RA; utfpr sustentavel - sustentabilidade - 6 RA; biotiba - projetos de biodiversidade - 6 RA; saudenautfpr - luta por ambulatorios - 1 RA; mobilizautfpr - mobilização de alunos - 4 RA; tecsolutfpr - economia solidária - 1 RA; uneoficial - movimento estudantil - 5 RA; uniperifa - periferia e movimento estudantil - 1 RA.

Avaliando os perfis de maneira generalizada, pôde-se perceber que a maior parte das Regras de Associações com perfis de caráter político envolviam entidades simpatizantes com cursos de humanas (Design, Letras e Comunicação Organizacional). Além do mais, traçando um comparativo da métrica de confiança entre as Atléticas, também pôde-se encontrar maior percentual das Atléticas de humanas (AAADec) do que da atlética de engenharias (Avalanche).

4 Considerações Finais

O projeto obteve resultados relevantes para elaboração de explicações associativas entre simpatizantes de perfis específicos do Instagram. Contudo, seria ideal que todos os dados de seguidores da UTFPR pudessem ter sido coletados, incluindo os perfis privados e inválidos, pois aumentaria a qualidade do Dataset a ser trabalhado. Além do mais, muitas expectativas referentes a determinadas correlações puderam ser cumpridas, o que evidencia o fato de existir uma maneira exata de comprovar esses critérios com o uso dos algoritmos. O código-fonte de todo o processo de desenvolvimento está disponível em um repositório próprio do *github* [8].

Referências

- [1] Redação Bem Paraná. Após morte de estudante, protesto é marcado e UTFPR Curitiba dá explicações. Disponível em: <https://www.bemparana.com.br/noticias/parana/apos-morte-de-estudante-protesto-e-marcado-e-utfpr-curitiba-da-explicacoes/>, 2023.
- [2] Robson Bonin. Para jovens, rede social e fake news vão influenciar o voto neste ano. Disponível em: <https://veja.abril.com.br/coluna/radar/para-jovens-rede-social-e-fake-news-vao-influenciar-o-voto-neste-ano>, 2022.
- [3] Instaloader Contributors. instaloader. Disponível em: <https://instaloader.github.io/>, 2021.
- [4] Pandas Contributors. pandas. Disponível em: <https://pandas.pydata.org/>, 2021.
- [5] Mlxtend Contributors. mlxtend. Disponível em: <https://rasbt.github.io/mlxtend/>, 2021.
- [6] Software Testing Help. Frequent Pattern (FP) Growth Algorithm In Data Mining. Disponível em: <https://www.softwaretestinghelp.com/fp-growth-algorithm-data-mining/>, 2023.
- [7] Jian Pei, Jiawei Han, Hongjun Lu, Shojiro Nishio, Shiwei Tang, and Dongqing Yang. H-mine: Fast and space-preserving frequent pattern mining in large databases. *IEEE Transactions*, 39:593–605, 03 2007.
- [8] Daniel A. P. Castro and Gustavo F. Amênio. Github projeto livre. Disponível em: <https://github.com/daniapc/projeto-livre-ima>, 2023.