Mineração de processos judiciais

V-Lab UFPE

Centro de Informática UFPE

SUMÁRIO

Introdução

Este documento tem como objetivo servir como um guia prático ao entendimento e uso do projeto elaborado em resposta ao desafio técnico JuMP para o cargo de cientista de dados.

O projeto consiste na aplicação de técnicas de mineração de processos na área judicial, manipulação e análise desses dados para geração de insghts e um melhor entendimento desse conjunto de processos, gerando, ao final, um modelo processual que melhor descreva as etapas seguidas durante cada caso.

A mineração de processos é uma técnica que utiliza algoritmos e ferramentas para extrair conhecimento a partir de logs de eventos, revelando o fluxo real de processos dentro de uma organização.

Em processos judiciais, essa abordagem é particularmente relevante, pois permite identificar gargalos, inconsistências e padrões que podem impactar a eficiência e a justiça das decisões.

A análise dos dados processuais através da mineração de processos possibilita uma compreensão mais profunda do comportamento dos processos, auxiliando na melhoria contínua do sistema judicial, na redução de atrasos e na promoção de maior transparência e previsibilidade nos resultados.

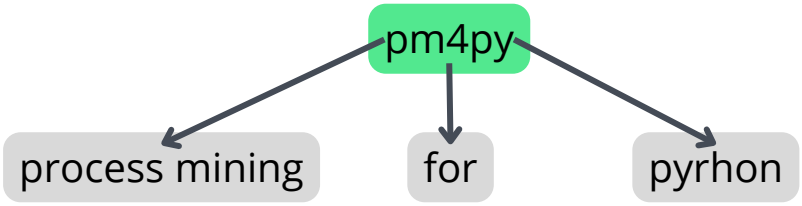
Linguagem e bibliotecas

Todo o código foi feito na linguagem python, por ser uma linguagem muito versátil, de fácil entendimento e com um grande número de bibliotecas.

As bibliotecas escolhidas visam um melhor uso de recursos dentro da proposta do projeto, já existe hoje muito material disponível para diversos tipos de tarefas, essas bibliotecas ajudam a tornar o código mais conciso e direto.

Para esse projeto utilizamos majoritariamente a biblioteca pandas para a análise e pre-processamento de dados, a escolha dessa biblioteca se explica pelo seu grande número de funções e por sua forte integração com a biblioteca pm4py.

A biblioteca pm4py, também muito utilizada no projeto, é uma biblioteca especializada em mineração de processos para o python, ela oferece diversas funções que ajudam na formatação do nosso dataset (event log) e criação do modelo processual.



Análise dos dados

Os datasets iniciais correspondem a dados de processos judiciais de duas unidades distintas, que aqui chamamos de unidade 1 e unidade 2. A quantidade e os tipos de campos das tabelas são identicos, oque facilita a análise comparativa.

Primeiro observou-se o tamanho de cada dataset, a unidade 1 possui X registros e unidade 2 possui Y registros. Por serem fontes de dados semelhantes os comportamentos dos dados também se assemelham.

Os números estão todos disponíveis no notebook para consulta e execução da análise. Para esse projeto, os campos de interesse foram actibity, complemento e documento.

Para melhor entendimento desses atributos e seu papel dentro do dataset, é necessário entendermos a entrutura de um event log.

processoID	activity	dataInicio	dataFinal
186510447	Distribuição	2022-10-24T23:46:31.769726000	2022-10-24T23:46:31.769726000
186510447	Audiência	2022-10-24T23:46:31.769726000	2022-10-24T23:46:32.593726000
186510447	Audiência	2022-12-14T12:58:53.104726000	2022-12-14T17:22:17.040726000
186510447	Expedição de documento	2022-12-14T17:22:17.040726000	2022-12-14T17:23:34.820726000
186919890	Expedição de documento	2022-10-21T17:40:19.013726000	2022-11-18T04:30:16.725726000

Um event log basicamente consiste de cases (ou traces), os cases são a tarefa final a ser executada, para um caso de um delivery, o seu case seria executar um pedido, o case é identificado por um id, que se repete durante o event log, isso se dá porque os cases são compostos por eventos.

Os eventos são as etapas que do case, cada uma dessas etapas são registradas como um registro no event log e levam os campos de id do case atrelado ao evento, a marcação de tempo e o nome do evento, esses são os atributos básicos mas podem haver outros.

No contexto do projeto, os event logs são os datasets da unidade 1 e unidade 2, os cases são os processos judiciais e os eventos são os movimentos, registrados na tabela como *activity*.

Outros tipos de tratamento dos dados foram feitos, como por exemplo, a retirada dos valores NaN nos campos documento e complemento, esses valores foram substituidos por strings vazias.

Unindo os campos complemento e documento

Para cumprir com os requisitos do projeto e com o formato padrão de um event log, foi necessário a criação de um novo campo na tabela: o campo criado foi o ‘org:resource’, esse campo tem o papel de fornecer um contexto maior para movimentos como Petição, já que possuem muitas ocorrências.

org:resource foi criado a partir da união dos campos complemento e documento, caso os dois campos existam, caso só um dos campos existam, só o campo existente entra no novo campo.

Retirada de movimentos insignificantes

Para uma melhor contrução do modelo processual, os movimentos "Publicação", "Decurso de Prazo", "Conclusão" e "Disponibilização no Diário da Justiça Eletrônico" foram retirados do dataset, isso se dá porque esses movimentos se repetem muitos e não trazem informação nova sobre o decorrer dos processos.

Para isso foi utilizada a função `filter_event_attribute_values` da biblioteca pm4py . Essa função filtra o event log baseado nos seguintes parâmetros:

log: event log / DataFrame do Pandas
attribute_key: chave do atributo
attribute to filter: atributo a ser filtrado
values: valores permitidos (ou proibidos)
level: especifica como o filtro deve ser aplicado ('case' filtra os casos onde ocorre pelo menos uma ocorrência, 'event' filtra os eventos, eventualmente cortando os casos)
retain: especifica se os valores devem ser mantidos ou removidos
case_id_key: atributo a ser usado como identificador do caso

Eventos documentados

Um modelo processual é crucial para visualizar, entender e otimizar o fluxo de atividades em processos, garantindo clareza, padronização e melhoria contínua.

Para aprimorar o modelo, movimentos repetitivos como "Publicação", "Decurso de Prazo", "Conclusão" e "Disponibilização no Diário da Justiça Eletrônico" foram removidos do dataset, pois não trazem novas informações relevantes, tornando o modelo mais focado e eficiente.

Outro requisito do projeto era o agrupamento e especificação de movimentos baseados em documentos e/ou complementos, tendo isso em vista, foi feita uma análise inicial para entender a quantidade e os eventos que possuiam o campo documento preenchido usando a função `filter_event_attribute_values`.

Dentre esses eventos foram observados os moviemntos mais frequêntes.

Agrupamento de movimentos similares

A tarefa de agrupamento de movimento similares consitia em, por meio da árvore hierárquica dada como um arquivo json, agrupar os movimentos Decisão, Despacho, Julgamento, Voto e Arquivamento com seus respectivos movimentos pais, a fim de evitar redundâncias e movimentos desnecessariamente específicos no nosso fluxograma.

Para essa tarefa de agrupamento foi utilizada como base a Tabela de Padronização de Unidades TPU do CNJ.

A abordagem utilizada foi a criação de um dicionário a partir dos movimentos e movimentoIDs presentes no nosso dataset e posteriormente a substituição dos ids pelos nomes dos movimentos na tabela de padronização (que contava somente com os ids dos movimentos) .

Com esse dicionário em mãos, foi criada uma função para encontrar o caminho de um dado nó (movimento) na árvore. a função `encontrar_caminho` tem como parâmetros: uma árvore, um nó alvo, e o caminho que é inicialmente vazio por default.

Como o uso da função pôde-se observar que todos os movimentos alvo estavam dentro do movimento de Magistrado.

A próxima etapa foi a alteração desses movimentos no dataset, foram alterados os campos *activity* e *movimentoID* para os valores 'Magistrado' e 1 respectivamente.

Especificação de movimentos

É fundamental fornecer um contexto mais claro sobre a execução de determinadas atividades, já que essas atividades podem ocorrer em diferentes momentos ao longo do processo judicial.

Para isso, foi analisado os documentos e complementos e suas ocorrências para cada um dos movimentos a serem especificados, são eles: Petição, Expedição de Documento, Mero Expediênte, Mandado e Audiência.

Após isso foi criada a função `discriminate_movement` para fazer a especificação (ou discriminação) dos movimentos, a função possui os parâmetros *eventLog, movimentId* e *complement* onde *complement* é do tipo booleando, pode ser verdadeiro ou falso.

Visualização do modelo processual

Um modelo processual é uma representação gráfica ou lógica das etapas e fluxos de um processo, como em um ambiente de negócios ou em processos judiciais. Ele é importante porque ajuda a visualizar como as atividades são sequenciadas, quem são os responsáveis e quais são as decisões críticas ao longo do processo, proporcionando clareza e transparência.

Para a criação do nosso modelo, utilizamos os event logs já formatados como parâmetros da função `heuristics_miner.apply_heu`

O resultado desse algoritmo de mineração heurística é aplicado na função `hn_vis.apply` que tem seu resultado aplicado na função: `hn_vis.view`

Assim, com um pouco de ajuste de parâmetros chegamos a nosso modelo processual final.

Lucas Nascimento Brandão
lnb@cin.ufpe.br

V-Lab UFPE

Centro de Informática UFPE