IV SEMINÁRIO INTERNACIONAL DE ESTATÍSTICA COM R
R & PYTHON E AS TENDÊNCIAS DE COLABORAÇÃO
NITERÓI, 21 A 23 DE MAIO DE 2019

ETL E MODELAGEM DE EXTRATOS DO BANCO DO BRASIL: SELENIUM + TIDYVERSE

Marcelo dos Santos Ventura[1], Bruno Morais Neves de Castro [2], João Pedro Amoêdo de Victor Coutinho [3]

Extratos bancários são extremamente importantes para acompanhar a saúde financeira de pessoas físicas e jurídicas. A partir dos dados financeiros gerados pelas movimentações bancárias dos usuários, aplicativos como GuiaBolso e Mobills consultam extratos para montar infográficos sobre compras, saldos mensais, de forma a prover orientação financeira. Como incremento de informação, a aplicação de aprendizado de máquina permite a apresentação de análises preditivas baseadas no comportamento observado, caracterizando-se como aspecto importante para tomadas de decisão.

A automação de rotinas manuais é fundamental para remover ou diminuir processos burocráticos, e obter foco somente em análises complexas, como predições de dados. Desta forma, o presente trabalho se propõe a realizar um fluxo de extração, transformação e carga de dados utilizando **Selenium** e **Pandas** com Python. Além disso, aplica aprendizado de máquina para realizar previsões de saldos mensais e diários como experimentação, utilizando R. De forma geral, a intenção é realizar um trabalho útil para monitoramento e previsão de dados financeiros em conjunto com R e Python.

Para possibilitar um fluxo contínuo de ações, muitos agendadores de tarefas como **CRON** e **IBM DataStage** são utilizados. O fluxo deste resumo simples será orquestrado por **Apache Airflow**. que tem como objetivo a execução de tarefas pré-agendadas. A grande vantagem em utilizar essa tecnologia está na sua aplicabilidade, configuração, e monitoramento através de um código Python e servidor web.

Objetivos

Os objetivos deste trabalho envolvem i) automatização e agendamento de execução de códigos, ii) ETL para dados de extratos do Banco do Brasil, iii) predição de novos saldos.

[1] Associação de Poupança e Empréstimo (POUPEX), marcelo.ventura@poupex.com.br

[2] Instituto Federal de Brasília (IFB), bruno.castro@estudante.ifb.edu.br

[3] Instituto de Educação Superior de Brasília (IESB), jp.advc@gmail.com

Material e Método

O Selenium Webdriver é uma ferramenta frequentemente utilizada para testes de aceitação (citação), e raspagem de dados, e torna possível executar ações pré-programadas em páginas web de forma automatizada. Com ele, foi possível simular o passo-a-passo que um usuário comum faria para acessar a página de extrato do banco para, posteriormente, extrair os dados da tabela de extrato a partir de seu referente identificador de tag HTML. A página é ilustrada na figura 1 a seguir:

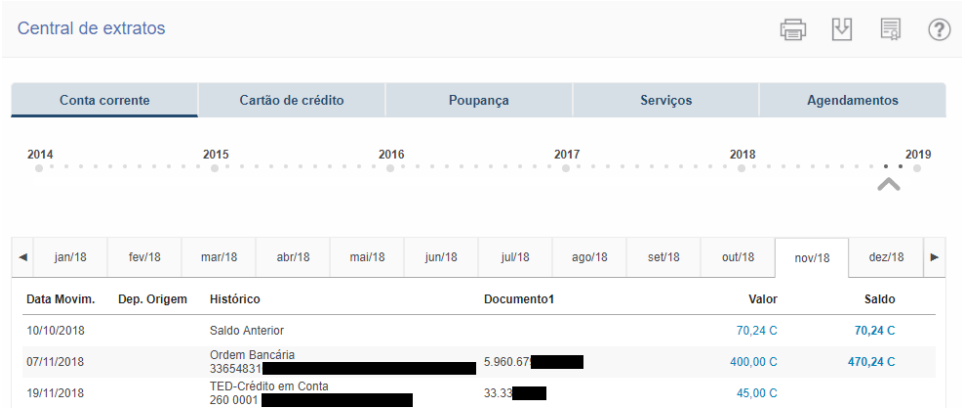

Figura 1 – Página “Central de extratos”

Fonte: Autores

Os arquivos baixados em formato de tabela foram lidos, manipulados e transformados com a biblioteca Pandas, conhecida pela alta performance em manipulação e análise de dataframes. O processo compreendeu os seguintes passos: (1) concatenação de todos os extratos coletados; (2) remoção das colunas "Dependencia Origem", "Data do Balancete" e "Unnamed: 6" devido à considerável quantidade de valores nulos; (3) ordenação crescente da coluna “Data”; (4) filtragem de dados do período informado; (5) definição da coluna “Número do documento” para tipo string e (6) redefinição dos índices. Duas funções contidas nos códigos utilizados são mostradas a seguir:

```
def salva_extrato(driver):
    driver.find_element_by_class_name('botaoToolBarSalvar').click()
    driver.find_element_by_class_name('csv').click()

def extrato_intervalo(driver,mes_procurado_final,diff_meses):
    mes_procurado_final -= diff_meses
    for i in range(diff_meses):
        driver.find_elements_by_class_name('ponto')[mes_procurado_final].click()
        scrap_pagina(salva_extrato,driver)
        mes_procurado_final += 1
```

Figura 2 – Código de simulação das interações do usuário

Fonte: Autores

Após a extração dos dados, os mesmos são consolidados em uma tabela ao ser combinado com o Pandas para tratamento consolidação de dados do extrato e escrita de arquivo no formato Comma-separated Values (CSV). Depois do processo de ETL, o pacote tidyverse é utilizado na leitura de dados, e transformação dos dados em somas de transações mensais, de forma a obter os saldos. Por fim, o pacote **astsa** auxilia na modelagem de dados utilizando o modelo ARIMA sazonal, com parâmetros p =0, d= 1, q= 1, P = 1 , D = 1, S = 12. A escolha de parâmetros foi manual, testando com alguns conjuntos de parâmetros com as funções **sarima**() , p-valor para coeficientes, e os critérios de seleção de modelos AIC e BIC2, Akaike (1973).

O Apache Airflow foi utilizado para monitoramento, organização e agendamento da execução de todos os scripts do processo descrito. Nele, temos o conceito de grafos acíclicos dirigidos (DAG) para representação dos fluxos de trabalho, que define um esquema lógico de relacionamentos e dependências para toda a coleção de tarefas. A Figura 3 exemplifica o fluxo utilizado no trabalho.

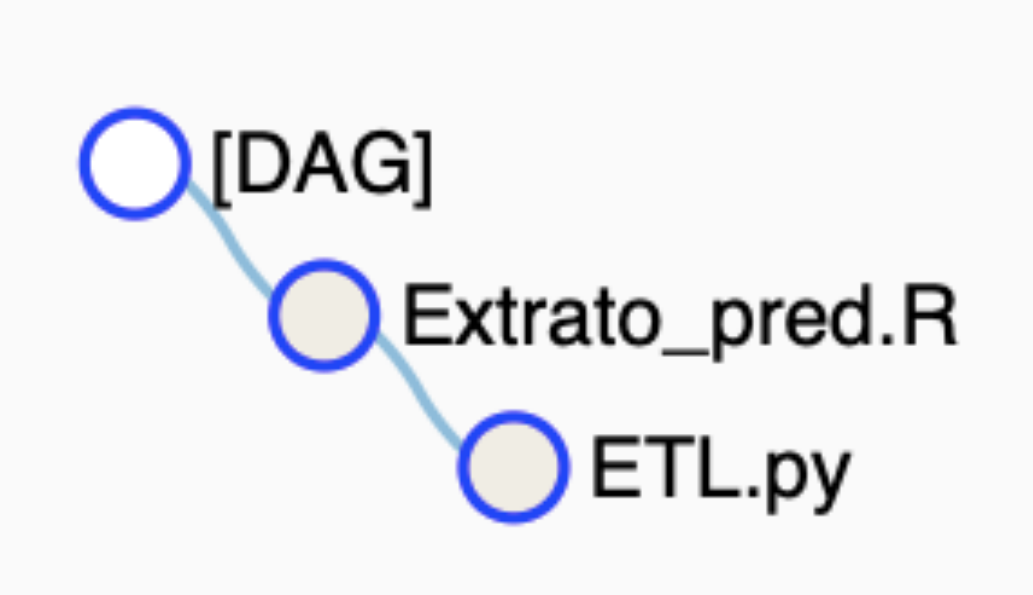

Figura 3 – DAG para AIRFLOW

Fonte: Autores

Primeiro, os extratos são coletados com o intervalo providenciado nos códigos python. Depois, o código R é executado para a leitura e modelagem de dados. A etapa de modelagem só pode ser automatizada depois da curadoria, e precisa ser revista mensalmente, tendo em vista o fluxo mensal de dados.

Resultados e Discussão

Primeiro, a execução do ETL com Selenium leva apenas 20 segundos para ser concluída para 1 ano. É um excelente tempo se comparado com o tempo que a tarefa levaria a ser realizada por um humano mecanicamente. A tabela 1 mostra os resultados do modelo Arima utilizado no modelo. Em conjunto com a figura 4, temos que os resultados são satisfatórios estatisticamente de acordo com os testes utilizados, e erros randomizados, o que caracteriza ausência de auto correlação. Portanto, os dados preditos com intervalos de confiança vide figura 5 são confiáveis.

Tabela 1 – Resultados de parametrizações do ARIMA

Parametros	**Taxa (%)**	
(0,1,1,1,1,0,S=1,P=12)	**Estimativa**	**p-valor**
mar1	-0.920	0.001
sar1	-0.236	0.284

Fonte: Autores, 2019

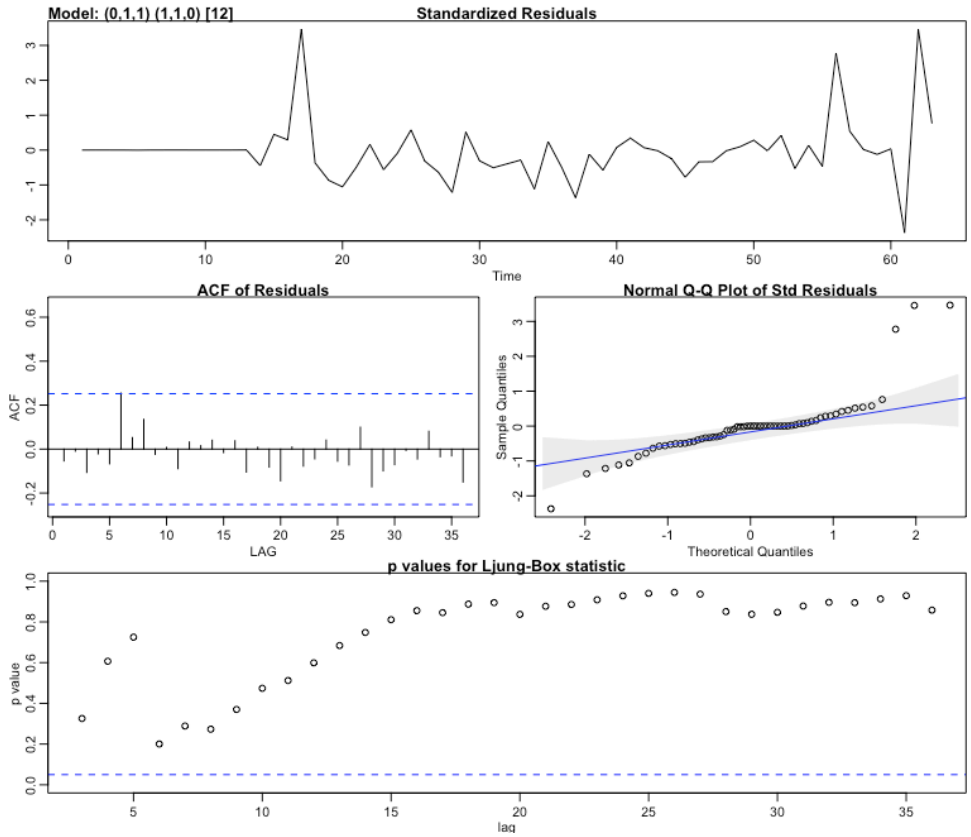

Figura 4 – Diagnósticos para modelo sugerido

Fonte: Autores

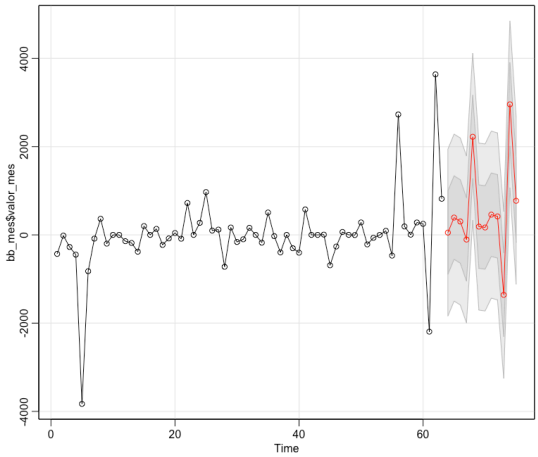

Figura 5 – Previsões para próximos 12 meses

Fonte: Autores

Conclusão

Dado o trabalho apresentado, nota-se que o potencial para automatizações de tarefas como downloads de extratos, em conjunto com técnicas estatísticas pode ser útil para usuários que busquem velocidade em obtenção de dados. Dado a curadoria em modelagens, o fluxo pode ser automatizado e revisto, de forma a orientar usuários financeiramente com predições que podem servir como termômetro e assim antever más decisões econômicas.

Referências

Airflow, https://airflow.apache.org/concepts.html

Akaike, H. Maximum likelihood identification of Gaussian autoregressive moving average models. *Biometrika*. 1973; **60**: 255- 265.

R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/

Selenium, https://www.seleniumhq.org

Stoffer (2017). astsa: Applied Statistical Time Series Analysis. R package version 1.8. https://CRAN.R-project.org/package=astsa

Wes McKinney (2010). Pandas: **Data Structures for Statistical Computing in Python**, Proceedings of the 9th Python in Science Conference, 51-56

Wickham (2017). tidyverse: Easily Install and Load the 'Tidyverse'. R package version 1.2.1. https://CRAN.R-project.org/package=tidyverse