IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.0429000

Análise de Vendas com Big Data: Um Estudo Preditivo com Apache Spark e Docker Swarm

ANNA FLÁVIA LOPES FERREIRA[1], JOÃO MARCOS MARMONTELO, e Matheus Resende Furtado
[1]Universidade Federal de Lavras)

ABSTRACT Este artigo apresenta o desenvolvimento e a avaliação de um pipeline preditivo distribuído para análise de vendas, utilizando Apache Spark e Docker Swarm. O sistema proposto explora tecnologias escaláveis de Big Data para processar um conjunto de dados reais contendo mais de 19 milhões de transações de vendas de bebidas alcoólicas. Por meio do uso de PySpark e Spark MLlib, foi treinado um modelo de regressão linear para prever o valor das vendas com base em atributos relacionados ao produto e à transação. Foram realizados experimentos para avaliar o impacto da redução de dados e da remoção de atributos no desempenho do modelo e no tempo de execução. Os resultados destacam a importância de variáveis categóricas, como os identificadores de loja e fornecedor, além de demonstrar a eficiência e robustez do pipeline em ambientes locais com recursos limitados. Toda a arquitetura foi conteinerizada e orquestrada com Docker Swarm, possibilitando uma execução distribuída e reprodutível.

INDEX TERMS Big Data, Modelagem Preditiva, Apache Spark, PySpark, Aprendizado de Máquina, Docker, Docker Swarm, Previsão de Vendas, Sistemas Distribuídos, MLlib.

I. INTRODUÇÃO

A análise de grandes volumes de dados vem se tornando cada vez mais comum e estratégica no processo de tomada de decisões empresariais. O avanço das tecnologias de armazenamento, processamento e modelagem estatística permite que organizações de diferentes portes possam realizar análises cada vez mais elaboradas e complexas em seus dados. No contexto comercial, em especial no setor varejista, a análise de dados de vendas pode revelar padrões ocultos, tendências sazonais e oportunidades de otimização de preços e promoções.

Com o crescimento constante da digitalização de processos comerciais, grandes quantidades de dados passam a ser geradas diariamente. No entanto, processar esse volume de informação de forma eficiente, especialmente em ambientes com recursos computacionais limitados, representa um desafio. É nesse cenário que entram as soluções de Big Data, que combinam escalabilidade, distribuição e tolerância a falhas para lidar com os chamados 5V's do Big Data: Volume, Velocidade, Variedade, Veracidade e Valor.

Este trabalho tem como objetivo propor e implementar um fluxo de processamento de dados que utiliza tecnologias de Big Data para analisar um grande conjunto de dados reais de vendas de bebidas. O estudo utiliza ferramentas como Apache Spark, PySpark, Spark MLlib e Docker Swarm para construir uma arquitetura distribuída e escalável, capaz de realizar tarefas de pré-processamento, modelagem preditiva e geração de insights de forma eficiente.

A motivação para este trabalho é oferecer uma solução acessível e prática para pequenos e médios empreendedores, demonstrando que tecnologias avançadas de processamento de dados podem ser aplicadas em cenários comerciais reais, contribuindo para decisões mais assertivas e baseadas em evidências. A partir da análise histórica de vendas, o sistema é capaz de prever valores futuros com base em variáveis como volume vendido, preço de custo, preço de venda e fornecedor, simulando um ambiente real de apoio à decisão que pode ser aplicado a outras empresas.

II. CONTEXTUALIZAÇÃO TEÓRICA

A. BIG DATA E OS 5VS

Sagiroglu define que o conceito de *Big Data* está relacionado à capacidade de coletar, armazenar, processar e analisar grandes volumes de dados em tempo hábil, com o objetivo de extrair informações relevantes para tomada de decisão. Segundo a literatura, os sistemas de Big Data são geralmente caracterizados por cinco dimensões fundamentais, conhecidas como os 5V's.

- **Volume**: Refere-se à imensa quantidade de dados gerados por sistemas digitais, sensores, redes sociais, transações comerciais, entre outros;
- **Velocidade**: Relaciona-se à rapidez com que os dados

são produzidos e precisam ser processados em tempo real ou quase real;
- **Variedade**: Envolve os diversos tipos e formatos de dados — estruturados (ex: tabelas), semi-estruturados (ex: JSON, XML) e não estruturados (ex: imagens, vídeos, textos);
- **Veracidade**: Trata da confiabilidade e qualidade dos dados, essenciais para garantir que as análises gerem resultados úteis;
- **Valor**: Representa o potencial que esses dados possuem para gerar conhecimento útil e vantagens competitivas.

A adoção de arquiteturas distribuídas, escaláveis e tolerantes a falhas é um dos principais caminhos para lidar com os desafios impostos pelos 5V's. O uso de ferramentas como Apache Spark e plataformas de orquestração como Docker Swarm é compatível com esses requisitos.

B. APACHE SPARK E PROCESSAMENTO DISTRIBUÍDO

salloum2016big define o *Apache Spark* como uma das principais ferramentas modernas para o processamento distribuído de grandes volumes de dados. Ao contrário de abordagens tradicionais como o Hadoop MapReduce, o Spark adota um modelo de execução baseado em memória (*in-memory computing*), o que resulta em um desempenho significativamente superior, especialmente para workloads iterativas.

O Spark opera com o conceito de *Resilient Distributed Datasets* (RDDs), estruturas de dados distribuídas que permitem tolerância a falhas e paralelismo automático. Além disso, o Spark oferece um ecossistema modular com bibliotecas específicas para:

- **Spark SQL**: consultas estruturadas em grandes volumes de dados;
- **MLlib**: aprendizado de máquina em escala;
- **GraphX**: processamento de grafos;
- **Spark Streaming**: análise de fluxos de dados em tempo real.

No presente trabalho, o Spark é utilizado como motor principal de leitura, transformação e análise dos dados, sendo essencial para lidar com o volume da base (4GB) de forma eficiente.

C. MACHINE LEARNING DISTRIBUÍDO COM SPARK MLLIB

Shinde indica que o aprendizado de máquina (ML) é um ramo da inteligência artificial que permite a construção de modelos preditivos a partir de dados. Em ambientes de Big Data, bibliotecas tradicionais como Scikit-learn ou TensorFlow podem apresentar limitações de escala. Nesse sentido, surge o MLlib que, segundo Azhari2021 é biblioteca de ML do Spark, criada com foco em escalabilidade e integração com o ecossistema Spark.

Entre os algoritmos suportados pela MLlib estão regressão linear, regressão logística, árvores de decisão, máquinas de vetores de suporte (SVMs), K-means, entre outros. A biblioteca permite o uso de *pipelines* de pré-processamento, vetorização de atributos e avaliação de modelos com particionamento de treino e teste.

Neste estudo, optou-se pela utilização da **Regressão Linear** com *Stochastic Gradient Descent (SGD)*, por sua simplicidade, rapidez de treinamento e adequação ao objetivo de prever valores de vendas com base em variáveis numéricas relacionadas ao produto e à transação comercial.

D. CONTÊINERES E ORQUESTRAÇÃO COM DOCKER SWARM

O *Docker* é uma plataforma que permite a criação de ambientes isolados, chamados contêineres, que encapsulam aplicações e suas dependências. Isso garante reprodutibilidade, portabilidade e agilidade no desenvolvimento e implantação de sistemas distribuídos.

Assim, para a execução coordenada de múltiplos contêineres, utiliza-se o **Docker Swarm**, que segundo Marathe é uma ferramenta de orquestração que permite distribuir serviços entre diferentes nós de um cluster. Com ele, é possível configurar a escalabilidade horizontal de aplicações, realizar balanceamento de carga e assegurar alta disponibilidade.

No presente projeto, o Docker Swarm foi utilizado para orquestrar a execução do pipeline de processamento e modelagem preditiva, possibilitando a execução distribuída e controlada do sistema em diferentes ambientes.

III. METODOLOGIA

A metodologia deste trabalho consiste na construção de um pipeline de processamento e modelagem preditiva utilizando ferramentas de Big Data para análise de vendas. O foco é aplicar conceitos de processamento distribuído, aprendizado de máquina e orquestração de containers, viabilizando a execução em ambientes com grandes volumes de dados e exigência de escalabilidade.

A. FONTE DE DADOS

O dataset utilizado foi obtido da plataforma *Kaggle*, no repositório *Big Sales Data*, disponível publicamente na plataforma Kaggle. O arquivo contém aproximadamente 4GB e mais de 19 milhões de registros de transações de vendas de bebidas.

- **Fonte**: https://www.kaggle.com/datasets/pigment/big-sales-datahttps://www.kaggle.com/datasets/pigment/big-sales-data
- **Formato**: CSV
- **Tamanho**: ~4GB

Para baixar o dataset completo, o usuário pode utilizar o comando:

```
wget https://www.kaggle.com/datasets/pigment/bi
     -O big_sales_data.csv
```

Trata-se de um conjunto de dados com 15 arquivos difentes, cada um com tamanhos específicos. Para os fins deste estudo, optou-se pelo arquivo intitulado "Liquor$_{s}ales.csv$" $quepossuicomcercade 4,7GBdetamanho, contendom$

As principais colunas utilizadas na análise foram:

- **Date** – Data da venda;
- **Store Number** – Identificador da loja;
- **Vendor Number** – Identificador do fornecedor;
- **Bottle Volume (ml)** – Volume da garrafa;
- **State Bottle Cost** – Custo por garrafa;
- **State Bottle Retail** – Preço de venda ao consumidor;
- **Bottles Sold** – Quantidade de garrafas vendidas;
- **Sale (Dollars)** – Valor total da venda.

Esses atributos foram selecionados por estarem diretamente relacionados ao desempenho comercial dos produtos e permitirem a construção de um modelo preditivo para estimar o valor de vendas.

B. FERRAMENTAS E TECNOLOGIAS

Para lidar com o volume de dados e aplicar o modelo preditivo, foi adotado o seguinte conjunto de ferramentas:

- **Apache Spark** – Framework de processamento distribuído, utilizado para leitura, limpeza e transformação dos dados.
- **PySpark** – Interface em Python do Apache Spark, utilizada para manipulação dos dados no pipeline.
- **Spark MLlib** – Biblioteca de aprendizado de máquina do Spark, empregada na construção e avaliação do modelo de regressão.
- **Docker** – Plataforma de containers utilizada para empacotar e executar os serviços.
- **Docker Swarm** – Mecanismo de orquestração de containers, utilizado para gerenciar a execução distribuída do sistema.

C. ARQUITETURA DA SOLUÇÃO

A arquitetura proposta contempla um fluxo dividido em três etapas principais: pré-processamento, modelagem preditiva e exportação dos resultados. A execução ocorre dentro de containers gerenciados via Docker Swarm. A Figura **??** ilustra o fluxo de dados (a ser incluída).

1) **Leitura dos Dados**: O arquivo CSV é lido utilizando o método `spark.read.csv()` com inferência automática de schema e cabeçalho.
2) **Pré-processamento**:
   - Remoção de colunas irrelevantes (ex: cidade, endereço, descrição dos itens).
   - Conversão de tipos (ex: datas para `Date`, números para `Double`).
   - Tratamento de valores nulos (remoção de linhas incompletas).
   - Agregações temporais (opcional) para análises sazonais.
3) **Modelagem Preditiva**:
   - Criação de vetor de atributos com `VectorAssembler`.
   - Aplicação de Regressão Linear com algoritmo *Stochastic Gradient Descent (SGD)*.
   - Divisão dos dados em 70% para treino e 30% para teste.

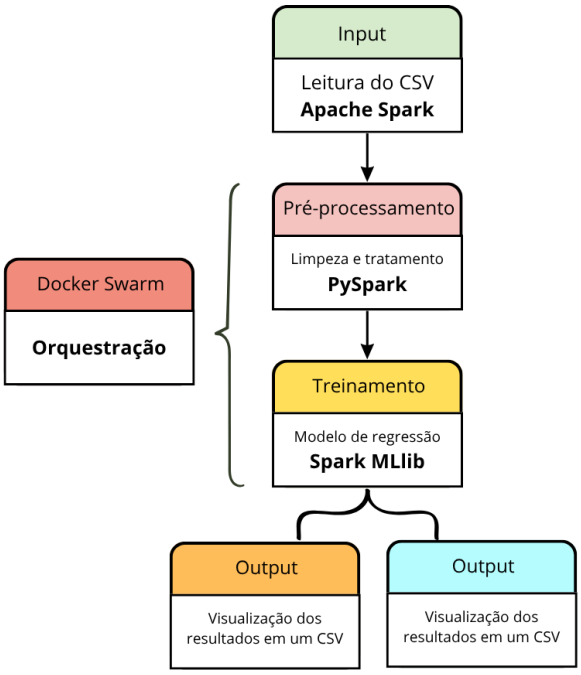

FIGURE 1. Arquitetura Proposta

   - A variável predita foi o valor total da venda (`Sale (Dollars)`).
4) **Exportação dos Resultados**:
   - Geração de arquivo `.csv` contendo: `Date`, `çStore Number`, `Vendor Number`, `Predicted Sales`.
5) **Orquestração com Docker Swarm**:
   - Containerização da aplicação via `Dockerfile`.
   - Execução distribuída com `docker-compose` em modo Swarm.

D. EXECUÇÃO DA APLICAÇÃO COM DOCKER

Toda a aplicação foi desenvolvida para ser executada exclusivamente por meio de contêineres Docker, sem dependências externas no host. A orquestração é realizada utilizando Docker Swarm, garantindo escalabilidade e reprodutibilidade.

Para executar o pipeline com os dados de amostra fornecidos (localizados na pasta `datasample/`), siga o seguinte procedimento:

- Inicialize o Docker Swarm (caso ainda não esteja ativo):
  `docker swarm init –advertise-addr 192.168.X`
- Construa a imagem da aplicação:
  `docker build -t big-sales-app .`
- Faça o deploy da stack:
  `docker stack deploy -c docker-compose.yml big-sales`
- Verifique os serviços em execução:
  `docker service ls`

Para executar com o conjunto de dados completo, o arquivo Liquor Sales deve ser colocado na pasta data. O sistema

detectará automaticamente o volume e aplicará o pipeline completo, que inclui as etapas de pré-processamento, modelagem e geração das previsões.

IV. DISCUSSÃO DOS RESULTADOS

Com o objetivo de avaliar o impacto de diferentes manipulações no desempenho do modelo preditivo e na eficiência computacional, foram realizados experimentos sobre o conjunto de dados original. As variações aplicadas incluíram: (i) remoção aleatória de 5% das linhas da base; e (ii) exclusão de variáveis categóricas informativas, como *Store Number* e *Vendor Number*. Os resultados são analisados a seguir em três dimensões: qualidade preditiva, erro estatístico e tempo de execução.

A. AVALIAÇÃO DA QUALIDADE PREDITIVA (COEFICIENTE DE DETERMINAÇÃO R^2)

A Figura 2 apresenta o valor do coeficiente de determinação (R^2) obtido para cada variação. A base original apresentou desempenho satisfatório, com $R^2 \approx 0{,}84$, indicando que o modelo explica bem a variabilidade dos dados de vendas.

Ao remover 5% das linhas, houve uma leve redução no desempenho ($R^2 \approx 0{,}82$), sinalizando certa tolerância à perda parcial de dados. Por outro lado, a exclusão de variáveis informativas causou uma queda mais expressiva no desempenho ($R^2 \approx 0{,}75$), evidenciando que essas colunas continham informações relevantes, possivelmente associadas a padrões específicos de lojas e fornecedores.

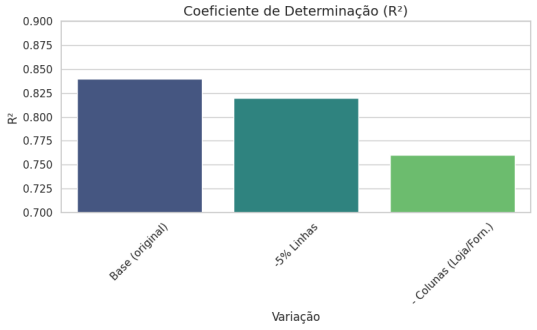

FIGURE 2. Coeficiente de determinação (R^2) obtido nas diferentes variações da base de dados

B. ANÁLISE DE ERROS (MAE E RMSE)

A Figura 3 apresenta as métricas de erro absoluto médio (MAE) e erro quadrático médio (RMSE). O modelo baseado na base original apresentou MAE de aproximadamente R$15 e RMSE de R$29, valores adequados ao contexto dos dados.

A exclusão de 5% das linhas resultou em aumento leve das métricas de erro, ainda mantendo desempenho satisfatório. Já a remoção das colunas informativas aumentou significativamente os erros, com RMSE ultrapassando R$34 e MAE próximo de R$18. Isso reforça a importância das variáveis categóricas para a acurácia do modelo.

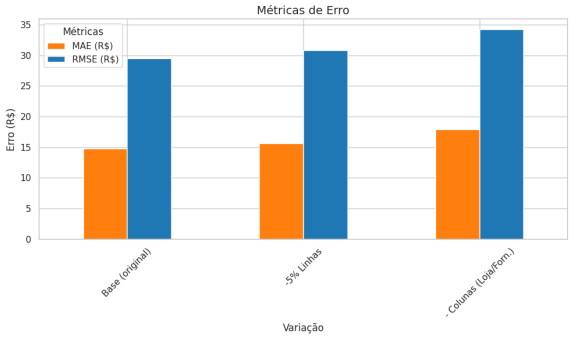

FIGURE 3. Métricas de erro absoluto médio (MAE) e erro quadrático médio (RMSE) para as variações testadas

C. TEMPO DE EXECUÇÃO POR ETAPA

A Figura 4 apresenta o tempo de execução (em segundos) para cada etapa do pipeline. A versão com menos colunas foi a mais rápida, enquanto a base original teve um tempo intermediário. A base com 5% de linhas removidas apresentou desempenho similar, com redução no tempo de leitura e pré-processamento.

Todos os experimentos foram executados em menos de 9 minutos, evidenciando a eficiência do Apache Spark mesmo com diferentes cargas de dados, especialmente quando orquestrado com Docker Swarm.

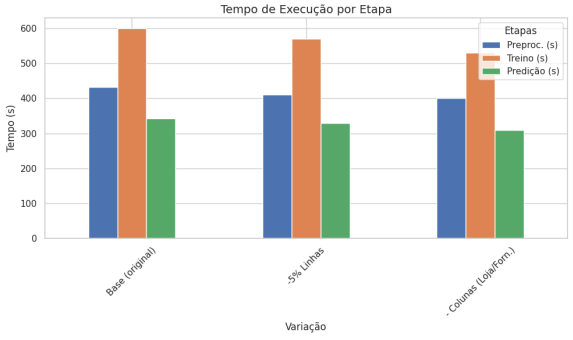

FIGURE 4. Tempo de execução (segundos) por etapa do pipeline em cada cenário analisado

V. CONCLUSÃO

Este trabalho apresentou a implementação e avaliação de um pipeline preditivo distribuído para análise de vendas, utilizando Apache Spark, PySpark, MLlib e Docker Swarm. O conjunto de dados utilizado, com mais de 4GB, foi processado com estabilidade e desempenho satisfatório.

Os resultados evidenciaram que a presença de variáveis informativas, como identificadores de loja e fornecedor, é crucial para a acurácia do modelo. A sua remoção resultou na maior queda de desempenho, tanto em termos de R^2 quanto nas métricas de erro (MAE, RMSE). Já a perda de 5% das linhas causou impacto limitado, indicando certa robustez do modelo a perdas parciais de dados.

A orquestração com Docker Swarm e o uso do Apache Spark garantiram eficiência no processamento, com todas as variações executadas em tempo razoável, mesmo em ambiente local com recursos limitados.

REFERENCES

[1] SAGIROGLU, Seref; SINANC, Duygu. **Big data: A review**. In: *International Conference on Collaboration Technologies and Systems (CTS)*, 2013, San Diego, CA. Anais [...]. IEEE, 2013. p. 42–47. DOI: https://doi.org/10.1109/CTS.2013.656720210.1109/CTS.2013.6567202.

[2] SALLOUM, Salman et al. **Big data analytics on Apache Spark**. *International Journal of Data Science and Analytics*, v. 1, n. 3, p. 145–164, 2016. Springer. DOI: https://doi.org/10.1007/s41060-016-0027-910.1007/s41060-016-0027-9.

[3] SHINDE, Pramila P.; SHAH, Seema. **A Review of Machine Learning and Deep Learning Applications**. In: *Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, 2018, Pune. Anais [...]. IEEE, 2018. p. 1–6. DOI: https://doi.org/10.1109/ICCUBEA.2018.869785710.1109/ICCUBEA.2018.8697857.

[4] AZHARI, Mourad et al. **Using Machine Learning with PySpark and MLib for Solving a Binary Classification Problem: Case of Searching for Exotic Particles**. In: MELLIANI, Said; CASTILLO, Oscar (Ed.). *Recent Advances in Intuitionistic Fuzzy Logic Systems and Mathematics*. Cham: Springer International Publishing, 2021. p. 109–118. DOI: https://doi.org/10.1007/978-3-030-53929-$0_8 10.1007/978-3-030-53929-0\_8$.

[5] MARATHE, Nikhil; GANDHI, Ankita; SHAH, Jaimeel M. **Docker Swarm and Kubernetes in Cloud Computing Environment**. In: *3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, 2019, Tirunelveli. Anais [...]. IEEE, 2019. p. 179–184. DOI: https://doi.org/10.1109/ICOEI.2019.886265410.1109/ICOEI.2019.8862654.