UNIVERSIDADE FEDERAL DE SÃO CARLOS

JOÃO VITOR AVERALDO ANTUNES - 813979

PEDRO ENRICO BARCHI NOGUEIRA - 813099

RAFAEL MORI PINHEIRO - 813851

Entrega Final

Apresentação dos Resultados Obtidos

SOROCABA

2025

SUMÁRIO

1. INTRODUÇÃO

O futebol moderno é uma indústria multibilionária, onde dados são um ativo estratégico para clubes, investidores e mídia. Dessa forma, decisões sobre contratações, táticas de jogo e desenvolvimento de atletas são cada vez mais orientadas por análises detalhadas. No entanto, a grande variedade e o volume massivo de dados – estatísticas de desempenho, valores de mercado, histórico de transferências, eventos de jogo – representam um desafio computacional significativo.

Este projeto prático propõe a construção de um pipeline de processamento de dados para analisar o ecossistema do futebol profissional. Assim sendo, utilizando um grande conjunto de dados do portal Transfermarkt, investigaremos padrões complexos relacionados à valorização de jogadores, ao impacto tático de escalações e à trajetória de carreira dos atletas. Por fim, para lidar com a escala e a complexidade dos dados, empregaremos uma arquitetura combinando o poder do processamento distribuído do Apache Spark com a flexibilidade de bancos de dados NoSQL, utilizando MongoDB, além de uma análise estrutural acerca das relações entre os jogadores utilizando Neo4j.

2. OBJETIVOS DO PROJETO

A princípio, o objetivo geral é desenvolver e implementar uma solução que envolva todas as etapas de ingestão, processamento, armazenamento e análise de dados de futebol em larga escala. Consequentemente, os resultados obtidos permitirão extrair insights valiosos que hoje são de difícil obtenção através de métodos tradicionais.

Desse modo, os objetivos centrais do nosso trabalho são divididos em três eixos de análise, que guiarão toda a implementação técnica:

2.1. Eixo 1: Evolução do Valor de Mercado e Impacto das Transferências

- Analisar a correlação entre a idade, o desempenho em campo (gols, assistências, minutos jogados) e a curva de valorização de um jogador.

- Identificar quais ligas ou clubes têm maior impacto na inflação do valor de mercado dos atletas.
- Calcular o Retorno sobre o Investimento (ROI) para clubes em transferências específicas, comparando o custo de aquisição com a evolução do valor e o desempenho subsequente do jogador.

2.2. Eixo 2: Impacto das Escalações e Substituições no Desempenho do Time

- Avaliar a eficácia de diferentes formações táticas (inferidas a partir das posições dos jogadores em campo) em termos de probabilidade de vitória, gols marcados e sofridos.
- Analisar o impacto das substituições: em que momento do jogo e em quais posições as substituições geram um impacto mais positivo
- Comparar a contribuição ofensiva (gols, assistências) de jogadores titulares versus jogadores que entram como substitutos.

2.3. Eixo 3: Análise da Carreira e Padrões de Trajetória de Jogadores

- Mapear e visualizar a trajetória de carreira completa de um jogador, identificando os clubes pelos quais passou e os períodos correspondentes.
- Identificar padrões de transferência comuns.
- Descobrir "clusters" de relacionamento entre clubes, ou seja, quais clubes mais negociam jogadores entre si, formando uma rede de transferências.

3. FONTE DE DADOS

Para atingir os objetivos propostos, utilizamos um único, porém bem populado, conjunto de dados disponível publicamente no Kaggle: **Player Scores from Transfermarkt.**

Este dataset contém múltiplos arquivos CSV inter-relacionados que cobrem mais de 30.000 jogadores, 400 clubes, 60.000 jogos e, crucialmente, 400.000 registros históricos de valor de mercado e 1.200.000 registros de participações em jogos.

4. TECNOLOGIAS ESCOLHIDAS E JUSTIFICATIVAS

Para atender aos requisitos do projeto e aos nossos objetivos analíticos, selecionamos uma arquitetura que combina o processamento dos dados com dois modelos de banco de dados NoSQL, o MongoDB e o Neo4j, utilizando o Apache Spark para o pré-processamento dos dados. Desse modo, a seguir estão justificadas as tecnologias citadas, assim como seus papéis na arquitetura do projeto:

4.1. Apache Spark

- **ETL:** O dataset é composto por diversos arquivos CSV. Utilizaremos o Spark (com PySpark) para ler, limpar, padronizar e unificar esses arquivos de forma eficiente. Portanto, operações como a conversão de tipos de dados, tratamento de valores nulos e a criação de novos atributos serão realizadas em paralelo sobre todo o conjunto de dados.
- **Agregações e Cálculos Complexos:** O Spark é capaz de realizar os cálculos pesados antes de carregar os dados nos bancos de dados de consulta. Por exemplo, para cada jogador, podemos usar o Spark para pré-calcular estatísticas de carreira (total de gols, média de minutos por jogo, desvio padrão do valor de mercado) que seriam computacionalmente caras para calcular em tempo real.
- **Joins em Larga Escala:** A necessidade de cruzar informações entre jogadores, seus valores de mercado, suas aparições em jogos e os

detalhes desses jogos (players + player_valuations + appearances + games) exige *joins* sobre um grande volume de dados. Logo, realizar essas operações em memória com a performance do Spark é mais eficiente do que sobrecarregar os bancos NoSQL com essa tarefa.

4.2. MongoDB

- **Modelagem do Perfil do Jogador:** Em um único documento no MongoDB, podemos armazenar todas as informações relevantes de um atleta: dados pessoais, estatísticas de carreira e, sua série temporal de valores de mercado e um histórico de suas principais atuações. Criando um perfil completo e autocontido, otimizado para consultas que pedem "todas as informações sobre o jogador X", eliminando a necessidade de *joins* no momento da leitura.
- **Estrutura de Jogos:** De forma similar, cada jogo pode ser um documento contendo seus dados principais (data, resultado, competição) e listas de eventos ou jogadores participantes.
- **Flexibilidade de Esquema:** Baseando-se nos pontos anteriormente apresentados, a flexibilidade de esquema torna-se um ponto crucial para a estruturação da informação, afinal precisamos da união da informação de maneira simplificada e direta, facilitando o processo de recuperação da mesma.

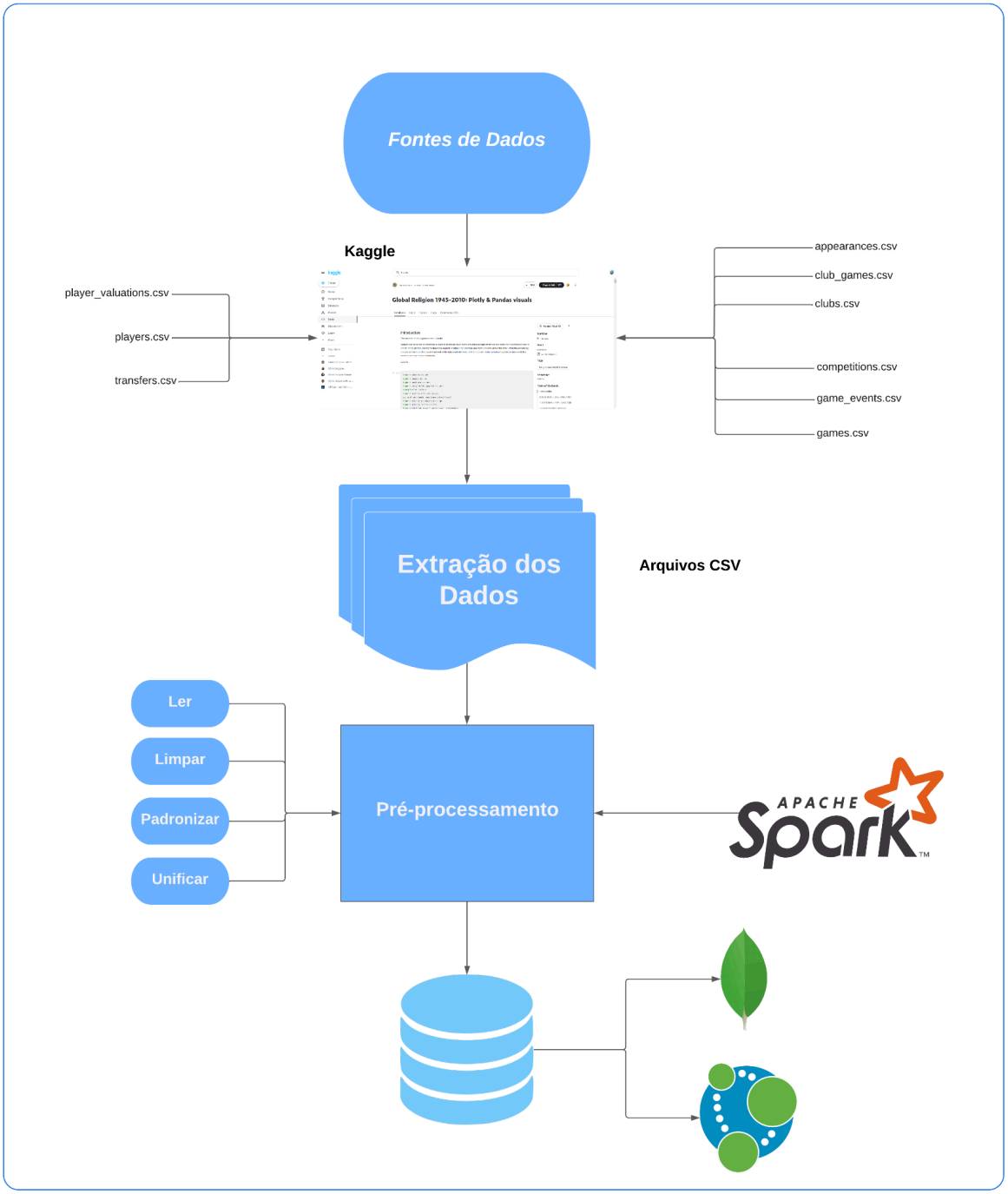
4.3. Neo4j

- **Análise de Trajetória e Transferências:** Podemos analisar a carreira de um jogador como um grafo de forma natural. Assim sendo, as entidades podem representar os nós (Jogador, Clube, País) e as conexões destes, podem ser vistas como as arestas (*jogou_por*, *foi_transferido_de_para*).
- **Análise de Padrões e Redes:** O modelo de grafos é mais adequado para encontrar padrões de relacionamentos entre entidades.

5. ARQUITETURA E FLUXOGRAMA DE DADOS

O fluxo de dados em nossa aplicação será linear e bem definido, garantindo que cada tecnologia seja usada em seu ponto de melhor atuação.

5.1. Fluxograma de Dados



5.2. Descrição do Fluxo

5.2.1. Ingestão e Processamento (Apache Spark)

- Os arquivos CSV do Kaggle são lidos pelo Apache Spark.
- O Spark executa o processo de ETL: limpeza de dados, padronização, junção das diferentes tabelas.
- São calculadas agregações complexas, como estatísticas de carreira por jogador.

5.2.2. Carga nos Bancos de Dados (Spark Connectors)

- Para o MongoDB: Os dados processados são modelados como documentos JSON. Por exemplo, para cada jogador, um documento único pode ser criado contendo seus dados e um array com seu histórico de valores. Esses documentos são carregados em coleções (players, games) usando o Spark Connector for MongoDB.
- Para o Neo4j: A partir dos mesmos dados processados, são criados os nós e as arestas que serão carregados no Neo4j usando o Spark Connector for Neo4j.

5.2.3. Análise e Consulta (MongoDB e Neo4j)

- As análises dos eixos 1 e 2 serão realizadas principalmente com consultas ao MongoDB, que é otimizado para recuperar documentos completos (perfis de jogadores, dados de jogos).
- As análises do eixo 3, focadas em relacionamentos e trajetórias, serão feitas com consultas Cypher no Neo4j.

5.3. Arquitetura Medalhão

O projeto foi estruturado seguindo o paradigma da **Arquitetura Medalhão** (conceito criado pelo Databricks), uma abordagem padrão da indústria para organizar e refinar dados em um pipeline:

- **Camada Bronze:** Contém os dados brutos, ingeridos diretamente dos arquivos CSV, sem modificações.
- **Camada Silver:** Uma versão limpa, validada, padronizada e enriquecida dos dados. É nesta camada que a maior parte da lógica de transformação de dados é aplicada. Os dados desta camada foram persistidos em formato **Delta Lake** para garantir atomicidade, consistência e performance.
- **Camada Gold:** Representada pelos bancos de dados NoSQL (MongoDB e Neo4j), onde os dados são modelados e agregados para responder a perguntas de negócio específicas e otimizar as consultas analíticas.

6. IMPLEMENTAÇÃO TÉCNICA DO PIPELINE

6.1. Configuração do Ambiente

O projeto exigiu uma configuração específica, incluindo JDK 17, Apache Spark 4.0.0 e Python 3.11. As dependências, como pyspark, delta-spark, pymongo e neo4j, foram gerenciadas em um ambiente virtual. A conexão com os bancos de dados foi estabelecida via conectores JAR específicos, carregados manualmente na sessão Spark para garantir consistência.

6.2. Camada Bronze: Ingestão de Dados Brutos

A primeira etapa do pipeline consistiu na leitura dos arquivos CSV de origem. Uma prática fundamental adotada foi a definição explícita dos schemas para cada tabela. Isso evitou o uso da onerosa operação inferSchema do Spark, garantindo maior performance na leitura e prevenindo erros de tipagem de dados nas etapas subsequentes.

6.3. Análise Exploratória e Diagnóstico dos Dados

Após a ingestão, uma Análise Exploratória de Dados (EDA) foi realizada. Com o auxílio de funções de visualização, foram identificados problemas críticos de qualidade de dados:

- **Valores Nulos:** Colunas importantes, como `foot` e `position` em jogadores, e `attendance` em jogos, apresentavam um número significativo de valores nulos.
- **Inconsistência Referencial:** Foram encontrados milhares de "registros órfãos", como participações em jogos (`appearances`) que referenciavam `player_id` ou `game_id` inexistentes nas tabelas principais.
- **Dados Mal Formatados:** Colunas monetárias, como `total_market_value`, estavam em formato de texto (ex: €25.5m) e não podiam ser usadas em cálculos.

6.4. Camada Silver: Limpeza, Validação e Enriquecimento

Com base no diagnóstico da EDA, os dados da camada Bronze foram transformados na camada Silver. As principais operações foram:

- **Correção de Integridade:** Registros órfãos foram eliminados através de inner joins com conjuntos de IDs válidos extraídos das tabelas principais, garantindo que todos os relacionamentos fossem válidos.
- **Tratamento de Nulos:** Valores nulos em colunas categóricas foram preenchidos com "Unknown", enquanto em colunas numéricas (como gols e assistências) foram preenchidos com 0.
- **Engenharia de Atributos:**
    - Uma função `parse_euro` foi criada para converter valores monetários em texto para o tipo numérico double.
    - A coluna `age` foi calculada para cada jogador com base em sua data de nascimento.
    - Como a coluna `total_market_value` dos clubes se mostrou inutilizável, uma nova feature foi criada: o valor de mercado de

cada clube foi recalculado agregando a soma dos valores de mercado individuais de todos os jogadores do seu elenco.

- **Persistência em Delta Lake:** Ao final do processo, todas as tabelas limpas e enriquecidas foram salvas em formato Delta Lake, criando um checkpoint robusto e otimizado.

6.5. Modelagem e Carga nos Bancos de Dados NoSQL

- **Carga no MongoDB:** Foram criadas duas coleções. A coleção `players` armazena um documento para cada jogador, com seus dados e arrays aninhados de histórico de valorações e transferências, ideal para os Eixos 1 e 2. A coleção `games` armazena um documento completo por partida, com escalações e eventos, para as análises do Eixo 2.

- **Carga no Neo4j:** Para o Eixo 3, foram criados nós `:Player` e `:Club`. O modelo de grafo foi enriquecido com dois tipos de relacionamento para cada transferência (`:TRANSFERRED_TO` e `:TRANSFERRED_FROM`), permitindo a análise direcional completa da carreira dos atletas.

7. ANÁLISE DE RESULTADOS

7.1. **Resultados do Eixo 1:** Análise do Valor de Mercado

- A análise de correlação entre idade e valor de mercado revelou a clássica "curva de valorização", mostrando que o pico de valor dos jogadores no dataset ocorre, em média, entre os 25 e 28 anos, com jogadores de maior média de gols (representados por pontos maiores e mais escuros no gráfico) atingindo valores de mercado significativamente mais altos.

- A segunda análise buscou identificar quais clubes são mais eficazes em valorizar seus atletas, calculando a média do crescimento percentual do valor de mercado dos jogadores durante seu período em cada clube. O gráfico de barras resultante destaca os 10 clubes com o

maior impacto positivo. No topo do ranking, clubes como Manchester United, FC Bayern München e Brighton & Hove Albion se destacam com as maiores médias de valorização. Este resultado sugere que essas instituições são particularmente bem-sucedidas em duas áreas principais: identificar e contratar jovens talentos com baixo valor de mercado e alto potencial, e desenvolver esses atletas em seu sistema, resultando em uma expressiva valorização.

- O estudo de caso do Retorno sobre Investimento (ROI) para as transferências de Romelu Lukaku demonstrou a capacidade de calcular a valorização pós-compra. A análise indicou que, embora algumas transferências tenham gerado um ROI positivo, outras resultaram em desvalorização, fornecendo um insight prático para avaliação de contratações.

7.2. Resultados do Eixo 2: Análise de Impacto Tático (MongoDB)

- A análise de eficácia das formações táticas mostrou que, entre as mais utilizadas, a formação 4-2-3-1 apresentou a maior taxa de vitória para os times da casa.

- A investigação sobre substituições revelou que a janela de tempo com o maior número de alterações é entre os 61 e 75 minutos de jogo.

- A comparação de contribuição ofensiva confirmou que jogadores titulares (`starting_lineup`) marcam um volume de gols muito superior aos que entram como substitutos (`substitutes`), validando a importância da titularidade para a produção de gols.

7.3. Resultados do Eixo 3: Análise de Carreira e Redes de Transferência (Neo4j)

- O mapeamento da trajetória de carreira, exemplificado com Álvaro Morata, permitiu a criação de um grafo interativo que mostra visualmente todo o percurso do jogador, incluindo a cronologia e os valores de cada transferência, de forma clara e sequencial.

- A análise de padrões de transferência entre países revelou as principais "rotas" do futebol mundial. Rotas como França ➜ Inglaterra e Inglaterra ➜ França apareceram como as mais frequentes, destacando os fluxos de talentos consolidados.

- A descoberta de "clusters" de relacionamento entre clubes mostrou, através de um grafo de rede, quais clubes mais negociam entre si. Pares como Inter de Milão e Genoa, e Juventus e Genoa, emergiram como parceiros de negócio frequentes, com a espessura da aresta representando a intensidade da relação comercial.

8. CONCLUSÃO

Este projeto realizou a construção de um pipeline de dados de ponta a ponta, desde a ingestão de dados brutos até a geração de insights analíticos sobre o ecossistema do futebol. A arquitetura escolhida, combinando Apache Spark, Delta Lake, MongoDB e Neo4j, provou ser robusta e adequada para lidar com os desafios de volume e variedade dos dados, permitindo que cada tecnologia fosse utilizada em seu ponto de maior força.

Os objetivos definidos nos três eixos de análise foram cumpridos de forma que foi possível modelar a evolução do valor de mercado dos jogadores, analisar o impacto de decisões táticas e mapear as redes de relacionamento que formam o mercado de transferências.

O trabalho demonstra o potencial da aplicação de tecnologias de processamento massivo de dados para extrair conhecimento estratégico de um domínio complexo e de grande interesse público como o futebol.

9. FONTES E REFERÊNCIAS BIBLIOGRÁFICAS

- https://www.kaggle.com/datasets/davidcariboo/player-scores
- https://www.mongodb.com/docs/manual/
- https://www.mongodb.com/docs/spark-connector/current/
- https://neo4j.com/docs/
- https://neo4j.com/developer/spark/
- https://spark.apache.org/docs/latest/