PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Engenharia de Dados

André Vieira de Lima

ENGENHARIA DE DADOS COM PROCESSAMENTO EM STREAMING PARA ANÁLISE DE SENTIMENTO EM AVALIAÇÕES DE VOOS

Rio de Janeiro
2023

André Vieira de Lima

ENGENHARIA DE DADOS COM PROCESSAMENTO EM STREAMING PARA ANÁLISE DE SENTIMENTO EM AVALIAÇÕES DE VOOS

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Engenharia de Dados como requisito parcial à obtenção do título de especialista.

Rio de Janeiro
2023

SUMÁRIO

1. Introdução

1.1. Contextualização

O trabalho de conclusão do curso apresenta a utilização de técnicas e ferramentas de engenharia de dados para processamento em *streaming* de avaliações de voos postadas por clientes de companhias aéreas. As mensagens de texto recepcionadas, são prontamente submetidas à uma estrutura arquitetural de *Big Data*, que auxilia no rápido *processamento de linguagem natural*, interpretando o conteúdo através de técnicas de *Machine Learning*. O resultado é disponibilizando para as partes interessadas em vários formatos de dados e em poucos segundos.

1.2. O problema Proposto

Como uma forma lúdica de descrever o problema, assim como a solução proposta: a ilusória empresa brasileira Mandíbula Analytics está disponibilizando um serviço inovador para a avaliação da experiência do usuário em voos realizados por todo o mundo. Produto desenvolvido em uma moderna plataforma de *Big Data*, que realiza *análise de sentimentos* em *posts* de passageiros. A classificação da experiência da viagem do consumidor é rapidamente disponibilizada para as companhias áreas em curtos tempos de processamento.

A performática ferramenta realiza o processamento de dados utilizando recursos de inteligência artificial em *streaming* de dados. O quê, sem sombra de dúvidas, agrega grande valor às companhias aéreas, permitindo um rápido contato com o consumidor no interesse de coletar detalhes após a utilização de um serviço de voo, e possibilitar o rápido feedback aos seus clientes.

A inovadora Mandíbula Analytics emergiu há pouco tempo no mercado brasileiro, e expandiu rapidamente suas atividades no mercado mundial. Sua carteira abrange empresas aéreas de todo o globo, sempre

focada em *Big Data* e Inteligência Artificial. Como fórmula de sucesso: uso de engenharia de dados com alta tecnologia que alavancam a qualidade de produtos e serviços dos clientes.

1.3. Objetivos

Construção de um fluxo de processamento contínuo de mensagens postadas em diversos idiomas, onde as informações são coletadas através de APIS, tratadas e armazenadas em bancos performáticos para análise e consulta. Tais dados são trafegados em *serviços de mensageria* de baixa latência, processados com recursos de *Machine Learning* e disponibilizados para o cliente em distintos formatos de armazenamento, além de funcionalidade construída em ferramenta de *Data Visualization*.

2. Modelagem Conceitual e Definição das Tecnologias/Ferramentas/Arquitetura

2.1 Dataset Utilizado

O primeiro módulo da aplicação foi construído para a coleta de dados alvo da análise. Tais informações foram extraídas da Kaggle (Airline Passenger Reviews) em junho de 2023. Link de acesso do dataset original:

https://www.kaggle.com/datasets/malharkhatu/airline-passenger-reviews

No interesse em adequar os dados ao desafio do projeto, algumas estruturas e informações foram modificadas, adequando o dataset para o objetivo proposto. Esse dataframe adaptado que foi utilizado, pode ser recuperado no repositório Git do próprio projeto:

https://github.com/fxmuld3r/puc_airlines_reviews_sentiment_analysis/tree/main/puc_airlines_reviews_sentiment_analysis/mock/data

Estrutura do dataset Airlines Reviews:

Nome da coluna/campo	Descrição	Tipo
col_index	Número que identifica a avaliação do usuário	número
departure_city	Cidade ou aeroporto de partida do passageiro	texto
airline	Nome da companhia aérea	texto
customer_review_original_language	Avaliação da experiência de viagem do cliente descrita em seu próprio idioma	texto
flight_date	Data do voo	texto

Tabela 1 - Dicionário de Dados de Avaliações de Experiência do Cliente

2.2 Modelagem Conceitual

O modelo conceitual concentra-se no mais alto nível de abstração e não leva em conta o banco de dados em si, mas a forma como as estruturas serão criadas para armazenar os dados.

Para o projeto não foi vislumbrado relacionamentos entre tabelas, mas apenas o uso da tabela principal de avaliações de voos de clientes para práticas de técnicas analíticas, *Machine Learning* e fluxo armazenamento contínuo de dados em *near real time*.

Estrutura do dataset Airlines Reviews Sentiment Analysis:

Nome da coluna/campo	**Descrição**	**Tipo**
col_index	Número que identifica a avaliação do usuário	número
departure_city	Cidade ou aeroporto de partida do passageiro	texto
airline	Nome da companhia aérea	texto
customer_review_original_language	Avaliação da experiência de viagem do cliente descrita em seu próprio idioma	texto
text_translated	Texto traduzido para o inglês referente à avaliação de experiência de viagem do cliente	texto
polarity	Grau de positividade ou negatividade no texto de avaliação de experiência de viagem do cliente	número
subjectivity	Grau de subjetividade no texto de avaliação de experiência de viagem do cliente	número
sentiment_result	Classificação do resultado de análise de sentimento no texto de avaliação de experiência de viagem do cliente	texto
flight_date	Data do voo	texto

Tabela 2 - Dicionário de Dados de Análise de Sentimentos em Avaliações de Experiência do Cliente

2.3 Stack Tecnológico

Conjunto de ferramentas e técnicas usadas para a solução:

- **Apache Airflow:** ferramenta de código aberto, escrita em Python e desenvolvida pela Apache Foundation. Seu objetivo é orquestrar pipelines de tarefas agendadas por meio de arquivos Python com instruções de sequenciamento definidas chamados DAGs;
- **Apache Kafka:** plataforma *open source* de processamento de *streams* desenvolvida pela Apache Software Foundation, escrita em Scala e Java. O projeto tem como objetivo fornecer uma plataforma unificada, de alta capacidade e baixa latência para tratamento de dados em tempo real. Sua camada de armazenamento é, essencialmente, uma fila de mensagens de *publishers/subscribers* maciçamente escalável projetada como um log de transações distribuído, tornando-o altamente valioso para infra estruturas corporativas que processam transmissão de dados;
- **Apache Parquet:** formato de armazenamento de dados orientado a colunas gratuito e de código aberto no ecossistema Apache Hadoop. É semelhante ao RCFile e ORC, os outros formatos de arquivo de armazenamento colunar no Hadoop, e é compatível com a maioria das estruturas de processamento de dados do Hadoop;
- **Apache Pinot:** ferramenta para armazenamento de dados distribuído, de código aberto e orientado a colunas. Pinot foi projetado para executar consultas OLAP com baixa latência. É adequado em contextos onde análises rápidas, como agregações, são necessárias em dados imutáveis, possivelmente, com ingestão de dados em tempo real;
- **Apache Spark:** framework de código fonte aberto para computação distribuída. Trata-se de um mecanismo de análise unificado para processamento de dados em grande escala com módulos integrados para SQL, streaming, machine learning e processamento de gráficos.

Ele provê uma interface para programação de clusters com paralelismo e tolerância a falhas;

- **Apache Spark Streaming:** extensão da API principal do Spark que dá suporte ao processamento de dados em tempo real com tolerância a falhas, alto desempenho e de forma escalável. Ele oferece suporte para cargas de trabalho em batch e streaming;
- **Apache Superset**: aplicativo de software de código aberto para exploração e visualização de dados capaz de lidar com dados em escala de petabytes;
- **Apache ZooKeeper**: servidor de código aberto para coordenação distribuída altamente confiável de aplicativos em nuvem. Ele é essencialmente um serviço para sistemas distribuídos que oferece um armazenamento hierárquico de valores-chave, que é usado para fornecer um serviço de configuração distribuída, serviço de sincronização e registro de nomenclatura para grandes sistemas distribuídos;
- **Docker**: conjunto de produtos de plataforma como serviço que usam virtualização de nível de sistema operacional para entregar software em pacotes chamados contêineres. Os contêineres são isolados uns dos outros e agrupam seus próprios softwares, bibliotecas e arquivos de configuração;
- **Flask**: micro *framework open source* escrito em Python para o desenvolvimento de Web APIs. Ele fornece as ferramentas básicas para criar aplicações web, como gerenciamento de rotas e requisições HTTP, sem fornecer muitos recursos adicionais ou configurações complexas;
- **MongoDB:** software de banco de dados orientado a documentos livre, de código aberto e multiplataforma, escrito na linguagem C++. Classificado como um programa de banco de dados NoSQL, o MongoDB usa documentos semelhantes a JSON com esquemas;
- **Python:** linguagem de programação de alto nível, interpretada de script, imperativa, orientada a objetos, funcional, de tipagem dinâmica e forte;

- **TextBlob:** biblioteca Python para processamento de dados textuais. Ele fornece uma API simples para mergulhar em tarefas comuns de *processamento de linguagem natural* (PNL), como marcação de classe gramatical, extração de sintagmas nominais, análise de sentimento, classificação, tradução e muito mais.

2.4 Arquitetura de Big Data

Para facilitar o entendimento da arquitetura proposta, a mesma está descrita nesse artigo, levando em consideração cinco temas:

- Identificação da origem dos dados;
- Obtenção dos dados;
- Armazenamento dos dados;
- Tratamento dos dados;
- Utilização da informação obtida

As seguintes características da arquitetura também são propostas para o projeto:

- Escalabilidade: uso de Docker para alguns módulos no interesse de aumentar capacidade de processar e armazenar dados de acordo com aumento de demanda;
- Tolerância a falhas: disponibilidade do sistema deve ser garantida, mesmo que ocorram falhas em alguns dos equipamentos utilizados;
- Dados distribuídos: os dados são armazenados entre diferentes máquinas, evitando assim o problema de armazenamento de grandes volumes;

- Processamento distribuído: o processamento de dados é realizado entre diferentes máquinas para melhorar os tempos de execução e fornecer escalabilidade ao sistema;
- Localidade dos dados: os dados e os processos que irão traduzi-los devem estar próximos para evitar transmissões de rede que adicionam latências e aumentam os tempos de execução;
- Análise e visualização: vem em primeiro lugar. Aqui os dados são exibidos para exploração e análise usando técnicas estatísticas, algoritmos de análise preditiva, aprendizado de máquina, etc.
- Governança de dados: concentra-se na integração, governança e segurança de dados. É necessário escolher os dados adequados que permitirão um processamento eficiente, com a qualidade exigida e protegê-los adequadamente, minimizando os riscos de segurança.
- Armazenamento e processamento: seu foco está no armazenamento dos dados obtidos e seu processamento eficaz de acordo com as necessidades que temos.

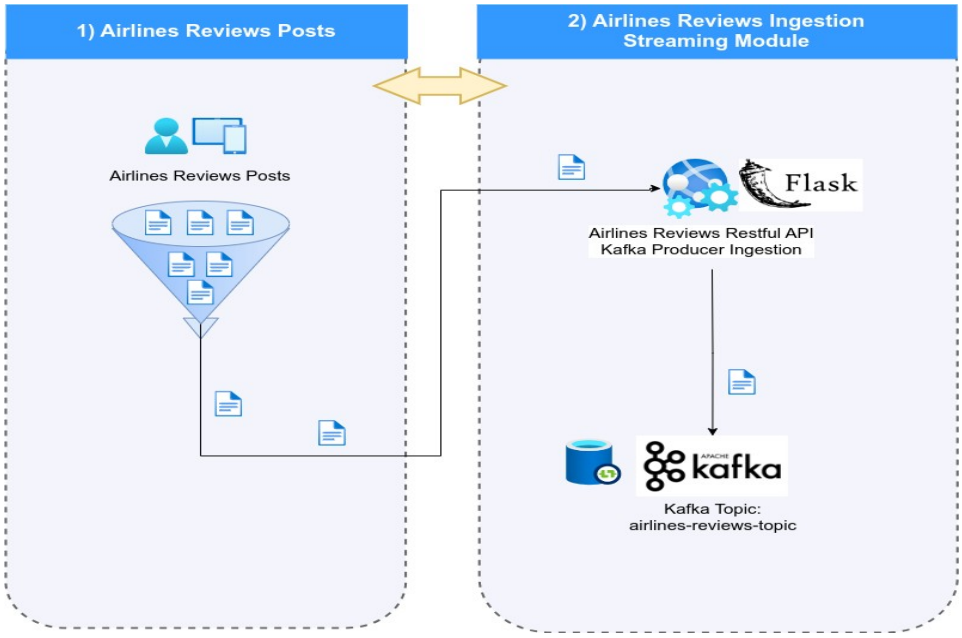
3. Ingestão de Dados

Para simular o processo de postagem de avaliações de experiência em voos de usuários, utilizamos um arquivo (formato CSV) de massa de dados contendo aproximadamente 50.000 *reviews* em 12 idiomas distintos. Mensagens de texto de cunho positivo ou negativo, narrando a experiência em voos de passageiros em 10 companhias aéreas clientes de nossa fictícia empresa.

Projetando um mundo onde passageiros reportam seu nível de satisfação com grande frequência, uma API Flask de *mock de dados* foi construída para carregar sequencialmente mensagens do arquivo de

massa de dados de testes. As mesmas são transmitidas (10 postagens em intervalos de um segundo) para outra API Flask, já no contexto principal de nossa aplicação, que consome as avaliações de voos. O mock é importante para o exercitar o comportamento do usuário no processo de ingestão de dados, e avaliar a capacidade de nossa aplicação, já que neste momento não temos a figura real dos passageiros realizando a publicação de *posts*.

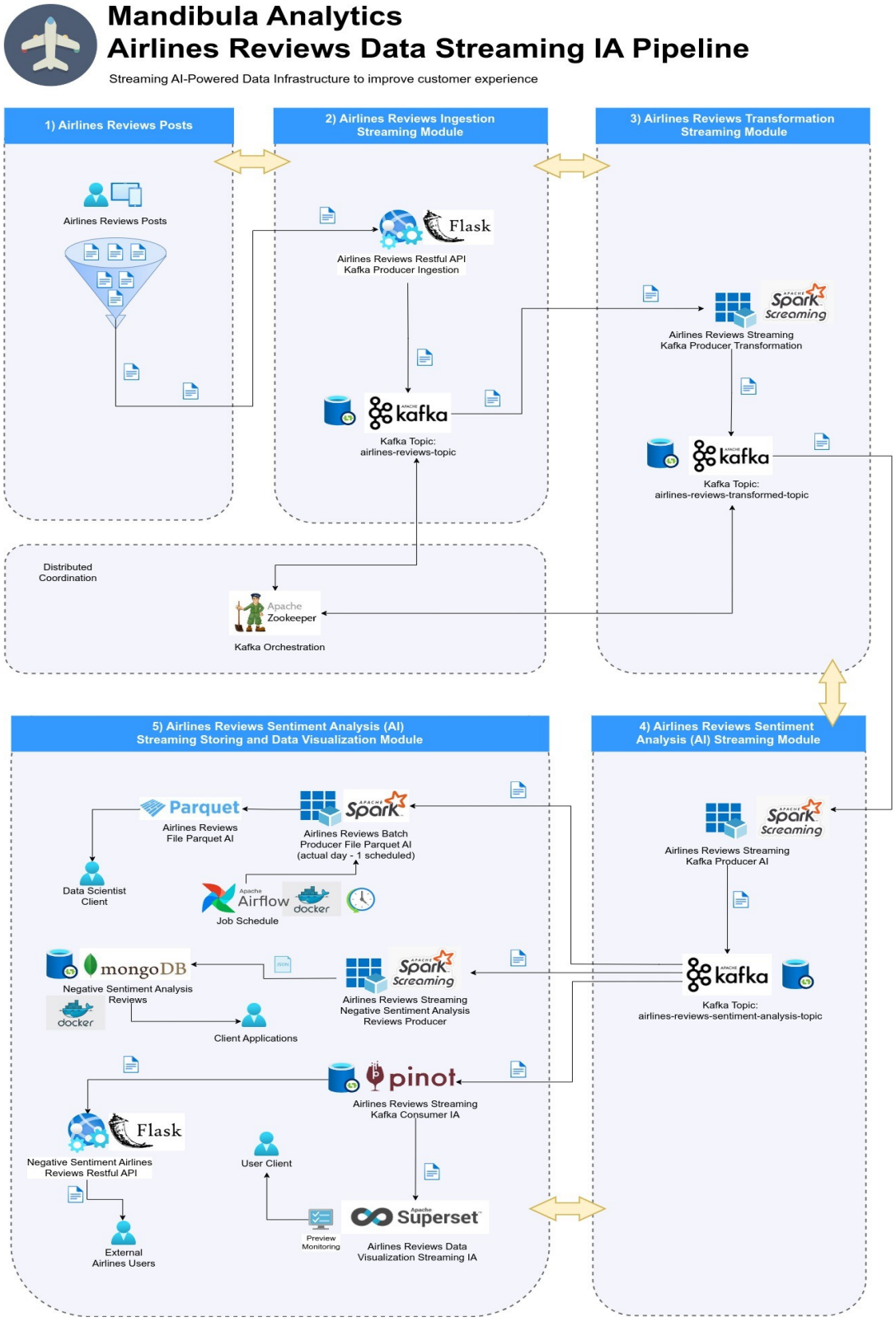
Por sua vez, a API de ingestão armazena as mensagens de forma crua (formato JSON) em um tópico Kafka (airlines-reviews-topic), que retém o conteúdo por vinte quatro horas até serem consumidas por outra funcionalidade da aplicação. Um desafio tranquilo em relação à capacidade das ferramentas utilizadas.



Fluxo de Ingestão

4. Orquestração de dados

Nesta seção, temos o desenho da arquitetura e a descrição detalhada de toda a pipeline de dados da aplicação.



Infográfico: Pipeline de Dados

As etapas de processamento são descritas de forma à ilustrar as principais funcionalidades, tecnologias, ambiente, fluxo de dados, consumo e armazenamento.

4.1 – Ingestão de Dados

Conforme narrado na seção 3 desse artefato (itens 1 e 2 do infográfico), os dados simulados de avaliações de voos (arquivo CSV) são carregados por uma API Flask Mock que envia as mensagens para serem consumidas por uma API Flask da aplicação. Essa última, armazena as mensagens (formato JSON) em um tópico Kafka (airlines-reviews-topic) orquestrado pelo Apache Zookeper e preparado para armazenar os dados originais dos usuários.

4.2 – Transformação de Dados

Nesta etapa (item 3 do infográfico) temos um módulo Spark Streaming que realiza a leitura e processa em NRT (near real time) as mensagens armazenadas no tópico Kafka de ingestão de dados (airlines-reviews-topic). As avaliações de voos são submetidas em fluxo contínuo aos seguintes processos de transformação de texto:

- Conversão do texto para letras minúsculas;
- Remoção de excessos de espaços entre as palavras;
- Remoção de quebra de linhas de parágrafos;
- Conversão do texto para o formato UTF-8;
- Remoção de URLs nos textos;
- Tradução das mensagens dos idiomas originais (português, espanhol, italiano, francês, árabe, alemão, chinês, turco, indiano,

holandês, japonês e grego) para o idioma inglês através da API pública do Google Translate.

Após o processamento de texto, as mensagens limpas e transformadas são armazenadas em um tópico Kafka (airlines-reviews-transformed-topic) orquestrado pelo Apache Zookeper (informações guardadas em novos campos de forma à manter os dados originais).

4.3 – Análise de Sentimentos de Avaliações de Voos

Nesta etapa (item 4 do infográfico) temos um módulo Spark Streaming que realiza a leitura e processa em NRT (near real time) as mensagens armazenadas no tópico Kafka (airlines-reviews-transformed-topic) de dados transformados. As avaliações de voos são submetidas em fluxo contínuo à um módulo de *machine learning* que realiza a análise de sentimentos dos textos publicados.

A biblioteca Text Blob, popular em aprendizado de máquina, avalia a polaridade e subjetividade dos posts dos usuários. Recurso de processamento de linguagem natural que avalia os textos de forma à classificar se o conteúdo possui informações negativas ou positivas dentro de um determinado contexto.

Cumprindo o objetivo principal da aplicação, as classificações de passageiros são incluídas em novos campos que representam pontuações de polaridade e subjetividade (escala de -10 à 10). Conteúdo que pode ser utilizado para entendimento da experiência do usuário, e planejamento de tomadas de ação para melhoria de serviços e retenção dos clientes.

Na sequência, o módulo Spark Streaming armazena (formato JSON) as mensagens com avaliação de sentimento em um tópico Kafka (airlines-reviews-sentiment-analysis-topic) orquestrado pelo Apache Zookeper.

4.4 – Disponibilização do Resultado de Processamento de Avaliações de Voos

Nesta etapa (item 5 do infográfico) temos uma miscelânea de funcionalidades que armazenam e disponibilizam os dados processados para usuários com diferentes papéis ou para aplicações externas.

4.4.1 – Disponibilização do Resultado Processamento de Avaliações de Voos para Cientistas de Dados

Nesta etapa (item 5 do infográfico) temos um módulo Spark que realiza a leitura de mensagens do tópico Kafka (airlines-reviews-sentiment-analysis-topic) com dados de análise de sentimentos e disponibiliza o resultado do processamento para usuários de companhias aéreas.

As avaliações de voos processadas pela nossa aplicação, são extraídas e fornecidas (arquivo no formato Parquet) diariamente para cientistas de dados dos clientes. Insumo útil para avaliação personalizada de comportamento dos passageiros.

O *processamento batch* é agendado e executado uma vez ao dia através do Apache Airflow. Tecnologia instanciada em recursos de nuvem através do Apache Docker.

4.4.2 – Armazenamento de Avaliações Negativas de Voos em Banco de Dados NOSQL

Nesta etapa (item 5 do infográfico) temos um módulo Spark Streaming que realiza a leitura (com filtros de resultados) de mensagens do tópico Kafka (airlines-reviews-sentiment-analysis-topic) e armazena em banco de dados NoSql para uso de outras aplicações.

As avaliações negativas de voos processadas pela nossa aplicação, são extraídas em fluxo contínuo, e armazenadas (formato JSON) em tabela banco de dados MongoDB. Insumo útil para consumo de outros sistemas internos ou das companhias aéreas.

O MongoDB é instanciado através do Apache Docker, tecnologia com recursos de cloud computing, bastante útil para virtualização de software.

4.4.3 – Armazenamento de Avaliações de Voos em Banco de Dados OLAP de Baixa Latência

Nesta etapa (item 5 do infográfico) temos Apache Pinot configurado para consumo de mensagens do tópico Kafka (airlines-reviews-sentiment-analysis-topic) e armazenamento interno.

As avaliações de voos processadas pela nossa aplicação, são lidas em fluxo contínuo, armazenadas e disponibilizadas em NRT (near real time) através de recursos OLAP para consulta de baixa latência.

O conteúdo do tópico Kafka podem ser lidos e contabilizados através de instruções SQL de forma extremamente performática. Recurso útil para acompanhamento de processamento e consumo de dados através de APIs internas do Pinot. Dados que podem ser recuperados por desenvolvedores, cientistas de dados, DevOps ou aplicações diversas.

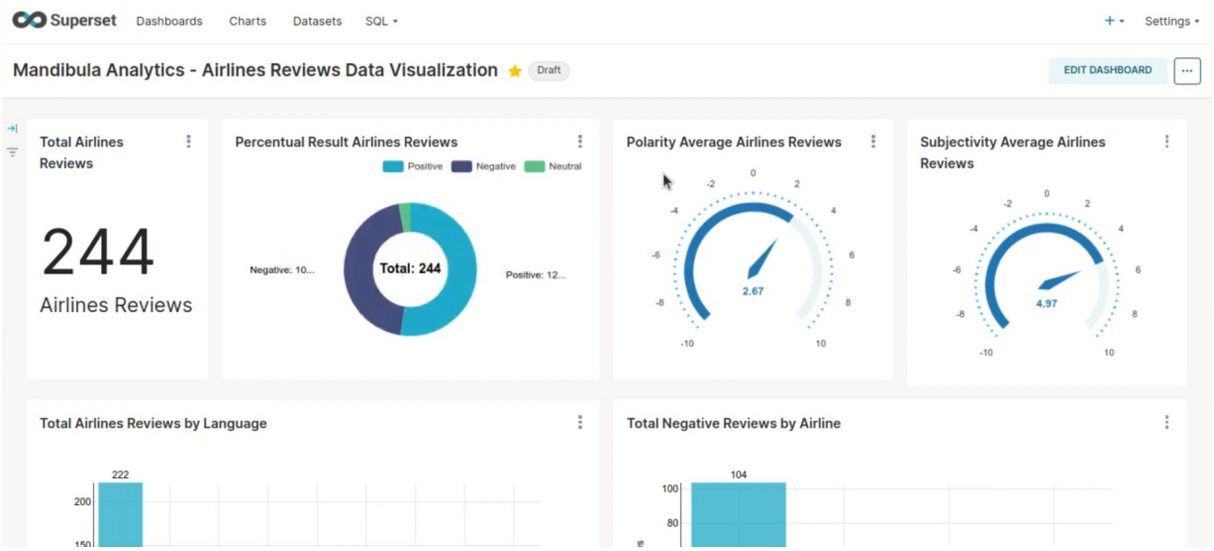
4.4.4 – Disponibilização de Avaliações de Voos para Companhias Aéreas

Nesta etapa (item 5 do infográfico) temos uma API Flask desenvolvida para consumo de mensagens do Apache Pinot e a disponibilização de dados (formato JSON) para as companhias aéreas. As avaliações de voos processadas pela nossa aplicação, são recuperadas do Pinot de forma performática em consultas de baixa latência.

5. Visualização de dados

Nesta etapa (item 5 do infográfico da seção anterior) temos um Dashboad que fornece dados para o acompanhamento do processamento do resultado de análise de sentimentos.

As avaliações de voos processadas pela nossa aplicação, são consumidas em fluxo contínuo do Apache Pinot e disponibilizadas em diversos gráficos do Apache Superset. Esse último, ferramenta low code extremamente útil para exploração e visualização de dados:



O Dashboard apresenta para o usuário final a quantidade de avaliações de voos, percentual de avaliações positivas e negativas, grau de polaridade das mensagens (positiva ou negativa), grau de subjetividade das mensagens (positiva ou negativa), total de avaliações por idioma, total de avaliações negativas por companhia aérea, histograma com notas de polaridade, total de avaliações negativas por idioma e as dez primeiras companhias aéreas com maior número de avaliações negativas.

O Apache Superset foi configurado (interface gráfica) para atualizar os dados de forma dinâmica à cada 10 segundos. Recurso que permite o acompanhamento do processamento de forma interativa.

6. Considerações Finais

A arquitetura foi preparada para um fluxo de processamento contínuo conforme premissas e necessidades da aplicação. Foram utilizadas tecnologias inovadoras e escaláveis, melhorando a qualidade de serviços contratados.

Dessa forma, toda a estrutura de engenharia comporta o processamento de um grande volume de dados, retornando informações em diversos formatos para usuários de diversos perfis, o quê representa grande valor ao cliente.

7. Links

Links públicos materiais utilizados no projeto:

Link para o vídeo (5 minutos) de apresentação do TCC:
https://www.youtube.com/watch?v=KXtxDYEkhag

Link para o repositório Git com o código fonte do projeto:
https://github.com/fxmuld3r/puc_airlines_reviews_sentiment_analysis

REFERÊNCIAS

SINGH, Ajit. **Processamento de linguagem natural com Python: Simplesmente em profundidade.** Babelcube Inc, 2021.

APÊNDICE

- **Análise de Sentimentos** – uso de processamento de linguagem natural, análise de texto, linguística computacional e biometria para identificar, extrair, quantificar e estudar sistematicamente estados afetivos e informações subjetivas;
- **Dasboard** – tipo de interface gráfica do usuário que geralmente fornece visualizações rápidas dos principais indicadores de desempenho relevantes para um objetivo ou processo de negócios específico;
- **Data Analysis** – processo de inspeção, limpeza, transformação e modelagem de dados com o objetivo de descobrir informações úteis, informar conclusões e apoiar a tomada de decisões;
- **Data Visualization** – expressão contemporânea da comunicação visual que consiste na representação visual de dados;
- **Kaggle** – plataforma de competição de ciência de dados e uma comunidade online de cientistas de dados e profissionais de aprendizado de máquina da Google LLC;
- **Machine Learning** – conceito ligado à inteligência artificial;
- **Near Real Time (NRT)** – processamento próximo ao tempo real;
- **Polaridade** – refere-se ao grau de positividade ou negatividade em um determinado texto. Na PNL, a análise de polaridade é usada para determinar o sentimento de um texto, seja ele positivo, negativo ou neutro.
- **Parquet** – um formato de arquivo em coluna com otimizações para acelerar as consultas;
- **Processamento de Linguagem Natural (PLN) –** subárea de inteligência artificial que estuda os problemas da geração e compreensão automática de línguas humanas naturais;
- **Stack Tecnológico** – conjunto de tecnologias;
- **Streaming** – fluxo contínuo;
- **UTF-8** - tipo de codificação binária de comprimento variável que pode representar qualquer caractere universal padrão do Unicode, sendo também compatível com o ASCII.