Introdução

A mobilidade urbana é um dos principais problemas da sociedade atual, especialmente em grandes cidades. Assim sendo, a importância do papel desempenhado pelos aeroportos é maior do que nunca, assegurando a interligação entre cidades, países ou mesmo de continentes, impactando diretamente, tanto a economia, quanto a experiência dos passageiros. Problemas que levem a atrasos de voos, ou até mesmo cancelamentos, tais como atrasos em check-ins, sobrelotação dos aeroportos, condições climatéricas adversas, afetam não só os passageiros, mas também as companhias aéreas e aeroportos, quer de forma monetária, quer a sua reputação.

Compreender os fatores que contribuem para atrasos e cancelamentos de voos, as causas, e como estes influenciam os preços dos bilhetes, pode ajudar na procura de soluções mais eficientes.

**Objetivos do projeto**:

O projeto tem como objetivo analisar as possíveis causas dos atrasos e cancelamentos de voos e as suas possíveis causas com os preços dos mesmos.

* Determinar aeroportos com maior quantidade de atrasos e possíveis causas
* Determinar aeroportos com mais voos cancelados e possíveis causas
* Perceber quais as rotas mais utilizadas
* Analisar os preços dos bilhetes nos aeroportos mencionados
* Analisar tempo médio de atraso por aeroporto vs. distância média por origem

Metodologia

De forma a conseguir resultados mais consistentes, foram utilizados 2 datasets com dados de voos dos EUA. Um dos datasets contém dados de voos como, trimestre, origem do voo, destino, preço, etc., sendo estes dados do ano 2018. Como segunda fonte de dados, foi utilizado um dataset com detalhes de voos, atrasos e cancelamentos. Ambos os datasets foram encontrados na plataforma Kaggle.

No caso do dataset dos voos encontrámos um problema, com quase 10 milhões de linhas, o pySpark era capaz de ler o ficheiro CSV, porém quando pedíamos para fazer leituras de linhas específicas ou alterar certos dados, era devolvido um erro de falta de memória, este erro para além de parar a célula do notebook em questão também acabava por terminar a ligação criada inicialmente com o MongoDB. Por isso decidimos criar um script simples em Python para diminuir o tamanho desse dataset, primeiro seleciona apenas o terceiro trimestre de 2018 através da coluna “Quarter” e depois escolhe aleatoriamente 250 mil linhas, acabando por exportar para um novo ficheiro CSV.

Depois de feito este pequeno tratamento de dados acedemos à plataforma MongoDB Compass para importar os dados, utilizando a instância criada pelo container do Docker proporcionado para a realização do projeto. Dentro da aplicação criámos duas coleções uma para os voos outra para os atrasos e importámos o ficheiro correspondente a cada uma.

Já dentro do notebook, começamos por criar a ligação entre o pySpark e a instância de MongoDB hospedada no nosso computador, depois é só fazer a leitura das duas coleções para cada um dos dataframes que iremos analisar.

Após a leitura dos dados, foram selecionadas apenas as features consideradas úteis, em ambos os datasets, eliminando as restantes. No dataset dos atrasos, foi feita a seleção dos voos apenas de 2018, de forma a ir de encontro ao outro dataset. No caso do dataset dos voos, para além de apagar as colunas consideradas desnecessárias, normalizámos os títulos de todas as colunas para ficarem com letras minúsculas. Aplicámos filtros de limpeza em ambos os datasets, eliminando linhas duplicadas ou com valores nulos, acabando por verificar se resta algum. Por último, reiniciámos o index nos dois dataframes, isto tem de ser feito de forma diferente do que em Pandas, utilizando a função Window para tal.

No que toca à análise exploratória inicial, começámos por perceber quais os pontos mais interessantes para se estudar. Fazemos uma análise de atrasos, cancelamentos e preços de bilhetes, começando pela identificação dos aeroportos com mais atrasos. Primeiro, os dados de atrasos são agrupados por aeroporto, e a soma dos atrasos (arr\_del15) é calculada para cada um. Depois, os aeroportos são ordenados em ordem decrescente pelo total de atrasos, e os 5 principais aeroportos mais atrasados são selecionados. Com base nesses aeroportos, o dataframe de voos é filtrado para incluir apenas os voos com destino a eles, e calcula-se o preço médio das passagens para cada um desses destinos. Além disso, os dados de atrasos são agrupados novamente por aeroporto para calcular o total de voos, total de cancelamentos e a percentagem de voos cancelados. A partir disso, os 10 aeroportos com as maiores percentagens de cancelamentos são identificados. Por fim, o dataframe de voos é filtrado para incluir apenas os aeroportos mais cancelados, e calcula-se a média do preço das passagens para cada um desses destinos. Os resultados fornecem uma visão abrangente sobre atrasos, cancelamentos e preços médios de bilhetes nos aeroportos mais importantes.

Análise de resultados

Começámos por analisar os aeroportos com maiores quantidades de atrasos significativos, **voos atrasados em 15 minutos ou mais**.

Uma imagem com texto, captura de ecrã, diagrama, Gráfico

Descrição gerada automaticamente

Como é possível observar, os 5 aeroportos que mais atrasos significativos apresentam encontram-se todos em cidades importantes ou com grande afluência turística, sendo estes:

* ORD: Chicago
* ATL: Atlanta
* DFW: Dallas/Fort Worth
* EWR: Newark, New Jersey
* SFO: San Francisco

De seguida, verificámos o preço médio dos bilhetes de cada um destes aeroportos, verificando que independentemente da quantidade de atrasos elevada, devido à importância dos mesmos para o tráfego aéreo, os preços dos bilhetes se mantêm bastante altos.

Uma imagem com texto, Tipo de letra, captura de ecrã, número

Descrição gerada automaticamente

Tal como vai ser analisado no futuro, a grande afluência de passageiros e complexidade operacional, é um fator importante nos atrasos e cancelamentos de voos.

Foram analisados também as **percentagens de cancelamentos por aeroporto**:

Uma imagem com texto, captura de ecrã, Tipo de letra, file

Descrição gerada automaticamente

Analisando os resultados, é possível destacar o aeroporto OGS (Ogdensburg International Airport), apresentando uma percentagem de cancelamentos muito superior aos restantes aeroportos, ultrapassando os 14% de voos cancelados. Este destaque pode surgir devido às condições climatéricas, uma vez que Ogdensburg, Nova York, é conhecido por enfrentar condições climatéricas adversas. Esta percentagem de cancelamentos pode também estar relacionada com a possível baixa frequência de voos.

Os aeroportos PGV, HVN e OAJ, apesar de apresentarem percentagens menores, não de ser significativas, podendo estar também relacionado com o clima local adverso. Podem também tratar-se de aeroportos de menor porte e com menos afluência de passageiros e operações. Um fator importante são também as infraestruturas, uma vez que aeroportos com problemas técnicos recorrentes ou limitações de infraestruturais, podem levar a mais elevadas taxas de cancelamentos.

Os restantes aeroportos, apesar de ainda apresentarem taxas relativamente altas de cancelamentos, podem ser considerados casos menos severos.

Tal como anteriormente, ao analisar os **preços das passagens aéreas dos aeroportos com maiores percentagens de cancelamentos** (imagem abaixo), é possível confirmar que praticamente todos os bilhetes se mantêm bastante caros, com a exceção do aeroporto OGS, que apresenta um custo por passagem bastante baixo quando comparado com os restantes, podendo confirmar a baixa afluência de voos e a menor infraestrutura do aeroporto me questão.

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Comparamos também a **média de minutos de atraso por aeroporto com as distâncias médias dos voos**, chegando à conclusão de que o aeroporto que o aeroporto que apresenta o maior atraso médio de chegada é o HNL (Honolulu International Airport), apesar de possuir uma distância média semelhante aos demais.

Uma imagem com texto, Tipo de letra, captura de ecrã, número

Descrição gerada automaticamente

Tal como referido em análises anteriores, isto pode dever-se tanto a questões operacionais, quanto a condições climatéricas. Apesar de Honolulu ser considerado como tendo um clima estável, ventos tropicais constantes podem afetar bastante as operações.

Implementação Técnica

O projeto foi implementado num ambiente Docker através da imagem fornecida. Foi criado o container aplicando todos os parâmetros descritos no ficheiro “docker-compose.yml”, de forma a configurar o MongoDB e Jupyter Notebook no mesmo ambiente. Após a criação do container, foram importados os dados no MongoDB Compass, através da conexão anteriormente definida. Criámos duas coleções distintas dentro de uma database “projeto”, uma coleção para voos e outra para atrasos, importando os devidos dados para cada uma.

Uma imagem com texto, captura de ecrã, Tipo de letra, branco

Descrição gerada automaticamente

Após a implementação e configuração do ambiente, deparámo-nos com dificuldades na ligação entre o Pyspark (biblioteca usada para fazer a ligação entre notebook e mongodb) e o MongoDB, vindo mais tarde a perceber que a quantidade elevada de dados de voos era a causa. Por isso, tal como mencionado na metodologia, foi criado um script para “cortar” os dados, mantendo apenas 250000 linhas. Foi então refeita a importação destes dados no MongoDB e iniciada a análise, após garantir a conexão entre o notebook e a base de dados.

Conclusões e Trabalho Futuro

Concluímos assim que tanto a logística dos aeroportos, quanto condições climatéricas, podem ser consideradas as principais causas do falhas e incumprimentos do tráfego aéreo. Deparamo-nos com uma elevada taxa de cancelamentos no aeroporto OGS, que pode indicar tanto problemas a nível logístico, uma vez que se trata de um aeroporto de menor escala e com menor afluência, quanto condições climatéricas adversas, como mencionado anteriormente. A nível de atrasos muito significativos, foi o aeroporto ORD, em Chicago que demonstrou um maior número de ocorrências. Em termos de atraso médio, foi o aeroporto de Honolulu, que tal como mencionado anteriormente, pode ser devido a fatores tanto logísticos como climatéricos.

Como **trabalho futuro**, seria interessante procurar fazer analises mais detalhadas, com datasets com maior variedade temporal de dados, e com maior variedade de destinos e origens de voos. Seria também interessante explorar a sazonalidade, procurando entender quais as épocas do ano com maiores turbulências a nível de tráfego aéreo. A automatização e monitoramento em tempo real também seria uma opção a considerar, podendo integrar uma API de meteorologia e modelos preditivos, de forma a tentar compreender e prever atrasos e cancelamentos nos diferentes voos.

Apêndice Técnico

MongoDB, uma das tecnologias utilizadas, é um banco de dados NoSQL que tem como vantagem a flexibilidade e escalabilidade do mesmo. É também bastante eficaz nas consultas de dados, o que facilita no uso de análise de dados e em aplicações de big data, devido à sua fácil conexão com APIs, como Pyspark. MongoDB é bastante utilizado em análises em tempo real, como monitoramento de negócios e movimentos.

Pyspark, anteriormente referido, é uma interface Spark para python. Trata-se de uma ferramenta de processamento de dados distribuídos e é utilizada em big data, devido à sua computação de larga escala. É compatível com uma série vasta de formatos de dados e combina o processamento em memoria com armazenamento distribuído, o que contribui bastante para a velocidade e eficácia do sistema. Tal como estudado, é bastante utilizado em casos de processamento, tanto streaming quanto batch. É usado em grandes empresas como Netflix, Amazon, AirBnb ou até mesmo Spotify. O Pyspark tem também a vantagem de ser capaz de processar algoritmos de machine learning em larga escala.

As duas tecnologias mencionadas complementam-se no meio de big data, uma vez que cada uma implementa vantagens suportadas pela outra. Onde MongoDB trata do armazenamento e recuperação dos dados, Pyspark processa e analisa os mesmos.

Alguns dos comandos utilizados no projeto foram:

* docker-compose up –build (construir o container do projeto)
* python *script.py* (usado para correr scripts escritos)
* !pip install *biblioteca* (instalar bibliotecas necessárias)
* import *biblioteca* (importar bibliotecas usadas)