INSTITUTO UNIVERSITÁRIO DE LISBOA

2º Semestre Ano Letivo 2022/2023

Unidade Curricular de Processamento de Big Data

Docentes João Pedro Oliveira e Adriano Lopes

Projeto

2º ano - LCD – CDB1

Marco Delgado Esperança, Nº 110451

Maria João Ferreira Lourenço, Nº 104716

Lisboa, 8 de abril de 2023

Índice

Introdução

O presente trabalho utilizou o dataset Flight Status Prediction (https://www.kaggle.com/datasets/robikscube/flight-delay-dataset-20182022) e seguiu os passos da pipeline utilizada para implementar projetos de ML, com o objetivo de prever os voos que teriam um atraso no mínimo de 15 minutos, ou seja, estudar a previsão dos mesmos e padrões relacionados com estes. Considerando que a identificação dos principais fatores que contribuem para a ocorrência de atrasos, pode ajudar as companhias aéreas a tomar medidas preventivas para reduzir a incidência de atrasos em voos futuros.

Deste modo, após limparmos o dataset, procedemos à correlação, selecionamos as variáveis que nos interessavam e realizamos um conjunto de estatísticas descritivas e gráficos, assim como, definimos conjuntos de treino e teste e, por fim, implementamos os algoritmos SVM e K-means.

Dataset e variáveis

O dataset utilizado - Flight Status Prediction - tem 4 GB e vários ficheiros contendo informações de voo distribuídas por um grande número de variáveis, que nos remetem para voos divergidos, cancelados e atrasados de 2018 a 2022. A informação anual dos voos pode ser encontrada nos ficheiros do tipo Combined_Flights_XXXX em formato csv e parquet. Os dados raw estão nos ficheiros do tipo Flights_XXXX_X.csv. Também existem dois ficheiros com a descrição detalhada de cada variável deste dataset. Por fim, temos o ficheiro Airlines.csv que tem as companhias aéreas e um código associado a cada uma, no entanto este não foi utilizado, pois em todos os ficheiros Combined_Flights_XXXX.csv tínhamos o nome da companhia aérea e não o seu código.

Não iremos proceder a uma descrição exaustiva das variáveis porque existem 61 variáveis diferentes, mas estão relacionadas com informações de voo – como origem, destino, companhia aérea, várias informações temporais, sobre os atrasos e seus tipos, sobre o avião que realizou o voo e aeroportos.

Importação

Antes de importarmos os ficheiros começamos por definir o tipo de cada variável para poupar tempo computacional, uma vez que de outra forma, o Apache Spark teria que correr todas as linhas do dataset para inferir o tipo, o que seria computacionalmente mais dispendioso. Assim, optamos por importar todos os ficheiros do tipo Combined_Flights_XXXX.csv de uma vez.

Verificação dos dados

Através do comando printSchema() verificamos se os tipos e as 61 colunas estavam de acordo com o que tínhamos definido em cima e pedimos para ver a primeira e última linhas, para percebermos o contexto dos dados.

Ao início tínhamos 29 193 782 linhas e não havia duplicados, mas após verificarmos que havia valores nulos e procedermos à remoção destes, ficamos apenas com 28 339 510 linhas. É de notar que removemos linhas e não colunas, porque a percentagem de valores omissos nas colunas em que esta situação se verificava era muito baixa e os valores omissos presentes em certas linhas poderiam interferir com o nosso modelo futuramente.
Relativamente à existência de valores únicos estes não são muito significativos, apresentando valores baixos, considerando o elevado volume de dados no dataset.

Correlação

Realizamos uma matriz de correlação com as variáveis numéricas. É de salientar que de vido ao elevado número de variáveis tivemos que arredondar os valores para uma casa decimal, se não seria impossível visualizar tantos valores. De seguida, fomos ver cada variável linha a linha e decidimos eliminar as variáveis que não iríamos usar nos próximos passos e com correlações entre si iguais ou abaixo de 0.5, uma vez que valores acima de 0.5 são indicadores de correlação fortes. Deste modo, eliminamos as variáveis *Flight_Number_Operating_Airline*, *DOT_ID_Operating_Airline*, *OriginCityMarketID*, *OriginStateFips*, *OriginWac*, *DestCityMarketID*, *DestStateFips*, *DestWac*, *TaxiOut*, *TaxiIn*, *CRSArrTime*, *ArrivalDelayGroups*, *DistanceGroup*, *DivAirportLandings*, o que fez com que ficássemos apenas com 47 das 61 colunas originais.

Outliers

Para verificar a existência de possíveis outliers fizemos uma breve análise descritiva apenas para as variáveis numéricas. O valor negativo das variáveis *DepDelay*, *AirTime*, *CRSElapsedTime*, *ActualElapsedTime*, *DepartureDelayGroups* e *ArrDelay*, apesar de existir era impossível em algumas delas considerando o contexto, por isso fomos à procura das linhas em que esta situação se verificava. Como isto só acontecia numa linha (para o *AirTime*), decidimos eliminá-la ficando apenas com 28 339 509 linhas.

Criação ficheiro parquet

Com vista às fases seguintes - análise exploratória, visualização, treino e teste e implementação dos modelos - criamos um ficheiro parquet com os dados limpos, o que nos permitiu realizar as próximas etapas de forma mais rápida e eficiente.

Análise exploratória e Visualização

Para realizar a análise exploratória utilizamos medidas descritivas e gráficos, que nos ajudaram a compreender melhor os dados. Começamos por explorar as variáveis

numéricas, ou seja, o número de linhas, a média, o desvio padrão, o mínimo e o máximo. Todas as variáveis têm 28 339 509 entradas. A média e o desvio padrão variam significativamente consoante a variável em causa. Há algumas variáveis com valores mínimos negativos, ainda que sejam apenas quatro e estão associados a voos que partiram antes do tempo. De seguida, realizamos uma diversidade de estatísticas descritivas e gráficos que pensamos serem interessantes, como por exemplo as variáveis de ordem temporal – *Year*, *Month*, *Quarter*, *DayofMonth*, *DayofWeek* -, as variáveis relacionadas com os atrasos – *ArrDel15* e *DepDel15* -, variável associada às companhias aéreas – *Airline* -, variáveis associadas às origens e destinos – *Origin* e *Dest* -, sobre os aeroportos e estados dos voos no total dos anos.

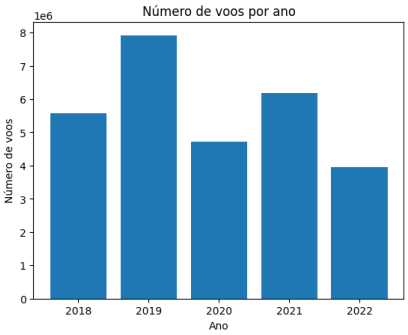

Figura 1-Gráfico número de voos por ano

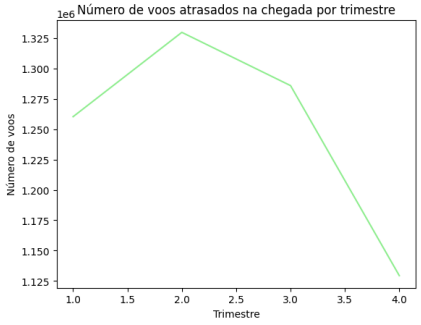

Figura 2-Time series voos atrasados na chegada por trimestre

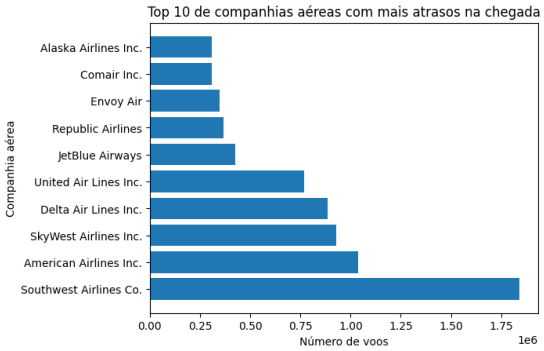

Figura 3-Top 10 de companhias aérea com mais atrasos na chegada

Na Figura 1, notamos um claro efeito da pandemia no número de voos, principalmente em 2020.

Na Figura 2 relativamente aos voos atrasados na chegada, nota-se um aumento crescente até ao segundo trimestre e um contínuo e progressivo decréscimo até ao quarto trimestre.

Na Figura 3, a companhia aérea com mais atrasos na chegada é a Southwest Airlines.

Treino e teste

Com vista à implementação de modelos que nos ajudassem a prever atrasos nos voos, começamos por criar uma subamostra com 15 milhões de linhas para que fosse possível correr localmente, considerando o elevado custo computacional que correr 28 339 509 linhas localmente traz.

Relativamente à criação de um conjunto de treino e teste utilizamos as percentagens recomendadas, ou seja, 80% treino - 2264861 linhas - e 20% teste - 566336 linhas – e

colocamos uma seed de 42 para que os resultados fossem sempre iguais. Estes conjuntos foram utilizados na implementação do modelo de aprendizagem supervisionada – SVM – e no modelo de aprendizagem não supervisionada – K-means.

Aprendizagem supervisionada – SVM

Aplicamos um modelo de classificação binária por SVM – Support Vector Machine – com o objetivo de prever se o atraso de um voo à chegada será pelo menos de 15 minutos, sendo o target *ArrDel15*, uma vez que este atraso pode ter um grande impacto no planeamento das companhias aéreas e passageiros, o que pode resultar em custos adicionais para ambos.

Procedemos à separação das colunas categóricas das não categóricas, para aplicar um pipeline usando o estimador de classificação LinearSVC e utilizando como hiperparâmetros: 10 como número máximo de iterações e 0.1 como parâmetro de regularização para prevenir o overfitting é possível identificar os principais fatores que contribuem para a ocorrência de atrasos e, consequentemente, ajudar as companhias aéreas a tomar medidas preventivas para reduzir a incidência de atrasos em voos futuros. A área abaixo da curva ROC é de 0.9778. Relativamente às métricas de classificação, obteve-se uma accuracy de 92,6%, precision de 0.798, recall de 0.777, specificity de 0.956 e F1-score de 0.788. No entanto, é importante destacar que a diferença entre a accuracy, a precision e recall sugere que as classes da variável alvo não estão balanceadas. Para lidar com essa questão, poderíamos ter utilizado uma amostra estratificada e a validação cruzada durante a análise, de modo a garantir uma distribuição equilibrada das classes na amostra de treino e teste. Desta forma, esta discrepância poderia ser possivelmente menor.

K-means

No contexto de dados de atrasos em voos, o modelo K-means, pode ser usado como uma ferramenta de aprendizagem não supervisionada para descobrir padrões e agrupamentos entre os voos com atrasos. Esses padrões podem ajudar as companhias aéreas e os reguladores a entender melhor as causas dos atrasos e a tomar medidas para melhorar o desempenho e a eficiência do sistema de transporte aéreo.

Sendo que o modelo K-means é especialmente adequado para esse tipo de análise, pois agrupa dados em clusters com base nas suas características semelhantes, o que pode ser útil para detetar padrões ocultos e relacionamentos entre os voos com atrasos.

Para esse efeito criamos dois modelos: num selecionamos três colunas específicas (*DepDelay*, *ArrDelay* e *Distance*) do dataframe df_sample para ajustar um modelo de clustering K-means. No outro analisamos outros fatores como: *Airline*, *Origin*, *Dest*, *CRSDepTime*, *DepDelayMinutes*, *DepDelay*, *ArrDelayMinutes*, *AirTime*, *Distance*, *Month*.

O melhor K para o modelo 1 é 5 e a Silhouete com distância euclidiana ao quadrado é de 0.698, logo o modelo tem uma coesão moderada com os dados iniciais.

O melhor K para o modelo 2 é 2 e a Silhouete com distância euclidiana ao quadrado é de 0.73, logo o modelo tem uma coesão moderada com os dados iniciais.

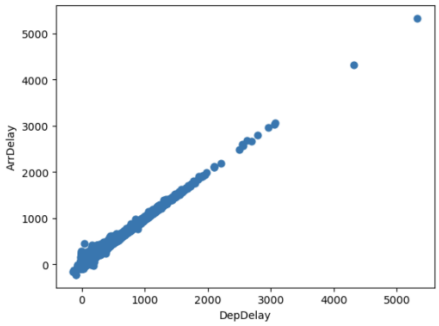

Figura 4-Gráfico com a relação entre ArrDelay e DepDelay

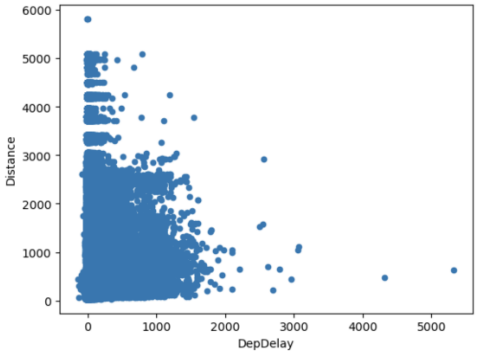

Figura 5-Gráfico com relação entre Distance e DepDelay

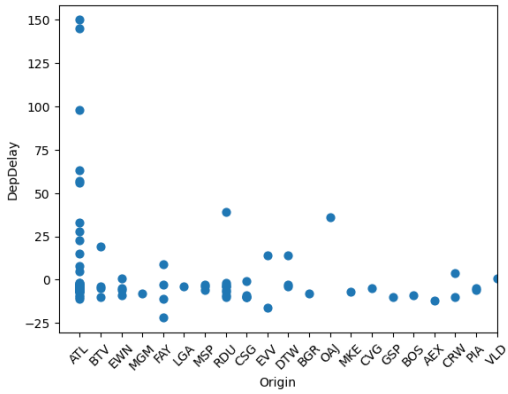

Figura 6-Gráfico com a relação entre DepDelay e Origin

Na Figura 4, verifica-se que existe uma relação, tal como se esperava, diretamente proporcional entre o atraso de partida (*DepDelay*) e atraso de chegada (*ArrDelay*).

No gráfico da Figura 5 podemos observar que há uma tendência para quanto maior for a distância da viagem, menor costumam ser os atrasos nas partidas. Além disso, verifica-se que a maioria dos atrasos não é muito significativo, até porque a viagem mais longa tem um atraso nulo.

Na Figura 6, existe um claro destaque para partidas atrasadas, principalmente no Aeroporto Internacional de Atlanta Hartsfield-Jackson, o mesmo se verifica para o gráfico análogo para as chegadas presente no notebook train_test.ipynb.

AWS

Para realizar a última parte do trabalho utilizámos o AWS, onde tivemos alguns problemas iniciais, como a importação dos ficheiros, que não funcionavam com a diretoria que utilizamos para correr localmente, mas que conseguimos ultrapassar ao percebermos que teríamos que guardar os ficheiros no S3, pois mesmo que o cluster fosse

terminado estes continuariam armazenados na cloud, permitindo uma utilização frequente e contínua, sem necessidade de voltar a importar os ficheiros.

De seguida, fomos comparar os tempos de execução dos modelos implementados utilizando o módulo time do Python, primeiramente com a subamostra que utilizamos para correr localmente e depois com o dataset completo. Os resultados obtidos ao correr localmente a subamostra estão na Tabela 1.

Subamostra local			
SVM			5.348 min
K-means	Modelo 1	Antes da escolha do melhor K	0.371 min
		Após a escolha do melhor K: K=5	0.887 min
	Modelo 2	Antes da escolha do melhor K	2.481 min
		Após a escolha do melhor K: K=2	0.107 min

Tabela 1-Resultados subamostra local

Os resultados temporais obtidos para a subamostra estão na Tabela 2.

Subamostra na AWS				
			2 instâncias	4 instâncias
SVM			5.249 min	5.335 min
K-means	Modelo 1	Antes da escolha do melhor K	1.107 min	1.190 min
		Após a escolha do melhor K: K=5	1.102 min	0.940 min
	Modelo 2	Antes da escolha do melhor K	2.629 min	2.649 min
		Após a escolha do melhor K: K=2	0.140 min	0.402 min

Tabela 2-Resultados da subamostra na AWS

Relativamente aos resultados da subamostra com duas instâncias Spark, o modelo SVM obteve 93% de accuracy, 0.98 de precision, 0.62 de recall, 0.997 de specificity e 0.76 de F1 Score.

Os resultados temporais obtidos com o dataset inteiro estão na Tabela 3.

Dataset inteiro na AWS			
			4 instâncias
SVM			41.952 min
K-means	Modelo 1	Antes da escolha do melhor K	1.245 min
		Após a escolha do melhor K: K=11	2.79 min
	Modelo 2	Antes da escolha do melhor K	1.402 min
		Após a escolha do melhor K: K=3	0.273 min

Tabela 3-Resultados dataset inteiro na AWS

Relativamente aos resultados do modelo SVM com 4 instâncias obtivemos accuracy de 0.93, 0.97 de precision, 0.64 de recall, 0.996 de specificity e 0.77 de F1 Score. O valor da

curva ROC é 0.966. O melhor valor de K para o modelo 1 do K-means é 11, o que neste caso implica que o modelo tem uma coesão moderada com os dados iniciais, devido à existência de Silhouette com a distância Euclidiana ao quadrado igual a 0.6538. Por sua vez, o melhor valor de K para o modelo 2 do K-means é 3 e como o valor da Silhouette com a distância Euclidiana ao quadrado é de 0.778, o modelo tem uma coesão elevada com os dados iniciais.

Comparando os diferentes resultados que se encontram na Tabela 1, 2 e 3, percebemos que relativamente ao SVM, o tempo de execução da subamostra na AWS e localmente não varia muito, mas que é mais alto quando se corre localmente. Quando corremos o dataset completo na AWS, o tempo aumentou cerca de 35 minutos, o que era expectável face ao elevado número de dados. Relativamente ao modelo 1 do K-means, com a subamostra o melhor tempo é conseguido localmente, no modelo 2 verifica-se a mesma situação.

Relativamente ao dataset inteiro, notou-se que os tempos de execução caíram consideravelmente no modelo 2, após a escolha do K.

Conclusão

Considerando os resultados alcançados pelos dois modelos, quer em AWS, quer localmente, podemos concluir que o modelo SVM tem uma boa capacidade de previsão, ainda que pudéssemos ter uma discrepância menor entre accuracy, precision e recall, coisa que melhoraríamos no futuro. Também poderíamos ajustar os hiperparâmetros do SVM para ver se conseguiríamos melhores resultados, assim como, utilizar outros modelos de classificação para a previsão.

Por outro lado, não consideramos a previsão do valor do FlightDelay, através de regressão e neste sentido, futuramente, conseguiríamos prevê-lo nos anos seguintes, usando redes neuronais LSTM.

Relativamente ao K-means tivemos bons resultados, mas podíamos explorar melhor os modelos que fizemos de forma a obter melhor coesão com os dados iniciais, assim como, explorar mais padrões relacionados com os atrasos.

Em suma, os resultados obtidos pelos modelos apresentados indicam que o uso de técnicas de Machine Learning pode ser uma abordagem eficaz para prever atrasos de voos, mas há oportunidade para melhorar os modelos utilizados, para fornecer previsões ainda mais precisas e confiáveis.