University of Minho
School of Engineering

Dados e Aprendizagem Automática

Trabalho Prático - Grupo 22

- Lucas Oliveira PG57886
- José Neiva PG53977
- Pedro Parpot PG47560
- Rafael Gomes PG56000

[1] Universidade do Minho, 4710-057 Braga, Portugal

Índice

1 Introdução

Este relatório tem como objetivo apresentar o desenvolvimento do trabalho de **Aprendizagem Automática** no âmbito da unidade curricular de Dados e Aprendizagem Automática, do Mestrado em engenharia Informática, da Universidade do Minho.

Este trabalho consiste na concepção e otimização de modelos de ***Machine Learning*** para a previsão da progressão de défices cognitivos leves (***MCI***) para a doença de Alzheimer (***AD***), utilizando técnicas avançadas de análise de imagens médicas. Sendo que o *dataset* foi extraído de uma iniciativa da *Alzheimer's Disease Neuroimaging Initiative* (***ADNI***), contendo exames de ressonância magnética (***MRI***) de pacientes em diferentes estados cognitivos.

O objetivo principal passa por explorar e validar a relevância do hipocampo na previsão da evolução para Alzheimer, bem como desenhar modelos preditivos otimizados para alcançar resultados robustos, contribuindo assim para a investigação no diagnóstico precoce de doenças neurodegenerativas.

2 Dataset

O *dataset* trabalhado pelo grupo foi o fornecido pela equipa docente, o **Alzheimer's Disease Neuroimaging Initiative (ADNI)**, que possui 305 linhas e 2181 colunas. Os dados deste *dataset* foram obtidos através de exames de ressonância magnética (*MRI*) do cérebro, focando-se em características radiómicas extraídas de diferentes regiões cerebrais. O objetivo principal é analisar os dados presentes e verificar de que forma o diagnóstico de evolução de condições como o **MCI** (*Mild Cognitive Impairment*) para *Alzheimer* (atributo objetivo) varia consoante os restantes atributos. Em que o *dataset* possui os seguintes atributos:

- **ID:** identificador único;
- **Image:** caminho para a localização do ficheiro;
- **Mask:** caminho para a localização da máscara correspondente a cada imagem;
- **diagnostics_Versions_(...):** indica a versão das bibliotecas usadas;
- **diagnostics_Configuration_(...):** indica configurações específicas do ambiente e parâmetros utilizados durante o processamento;
- **diagnostics_Image_(...):** indica estatísticas e propriedades técnicas da imagem principal;
- **diagnostics_Mask_(...):** indica informações relacionadas à máscara associada a cada imagem;
- **original_(...):** sem qualquer filtro aplicado, usa a imagem tal como foi adquirida;
- **wavelet_(...):** aplica decomposições, gerando oito imagens por nível;
- **log-sigma-(...) (Laplacian of Gaussian):** realça mudanças nos níveis de cinzento, enfatizando as arestas. O parâmetro "sigma" controla a granularidade do realce;
- **square_(...):** eleva ao quadrado a intensidade dos *pixels* e mapeia novamente os valores para a gama original, destacando áreas de alta intensidade;
- **squareroot_(...):** calcula a raiz quadrada das intensidades absolutas dos *pixels*, reduzindo a influência de valores extremos;
- **logarithm_(...):** aplica o logaritmo às intensidades absolutas. Este filtro reduz a diferença entre intensidades altas e baixas;
- **exponential_(...):** aplica a função exponencial às intensidades, ampliando diferenças entre intensidades menores;
- **gradient_(...):** calcula a magnitude do gradiente local, realçando as transições rápidas de intensidade;
- **lbp-2D_(...) (LocalBinaryPattern2D):** gera o padrão binário local (*LBP*) em *2D*, analisando texturas baseadas na relação entre um *pixel* e os seus vizinhos;
- **lbp-3D_(...) (LocalBinaryPattern3D):** versão em *3D* do *LBP*, usando harmónicos esféricos (funções harmónicas) para extrair texturas volumétricas;
- **Sex:** representa o género do participante em estudo, caso seja feminino (0) ou masculino (1);
- **Age:** idade do participante no momento de recolha dos dados;
- **Transition:** indica a transição do estado clínico do participante entre os exames.

3 Análise de Dados

De modo a que possamos identificar qual o melhor tratamento a aplicar nos dados, procedemos à seguinte análise.

3.1 Compreensão do dados

Inicialmente, realizámos uma análise exploratória para identificar a presença de valores em falta (*missing values*), atributos com valores constantes em todas as entradas, entradas duplicadas, colunas irrelevantes para o problema, bem como a distinção entre dados numéricos e categóricos. Chegando assim às seguintes conclusões:

- **Missing Values**: Não foram encontrados.
- **Atributos com o mesmo Valor**: Existem 159 atributos.
- **Entradas Duplicadas**: Não foram encontradas.
- **Atributos Categóricos e Numéricos**: Existem 2161 atributos numéricos e 20 atributos categóricos.
- **Atributos Irrelevantes**: Verificamos que os todos os atributos categóricos do dataset, menos o atributo *Transition* que é o que estamos a tentar prever, eram referentes a versões, tamanhos da imagem e configurações, o que são irrelevantes para o problema que estamos a estudar, portanto, achamos melhor excluir esses atributos.
- **Valores *Outliers*:** Foi calculado o **Intervalo Interquartil (*IQR*)** para identificar *outliers* em cada coluna do *dataset*. O limite inferior e superior foram determinados como $Q1 - 1.5 * IQR$ e $Q3 + 1.5 * IQR$, respetivamente. Sendo assim considerados *outliers*, todos os valores fora destes limites calculados.

3.2 Visualização dos Dados

De seguida, fomos verificar como estava a distribuição do número de condições do Transition no *dataset*, para isso realizamos este gráfico de barras e circular.

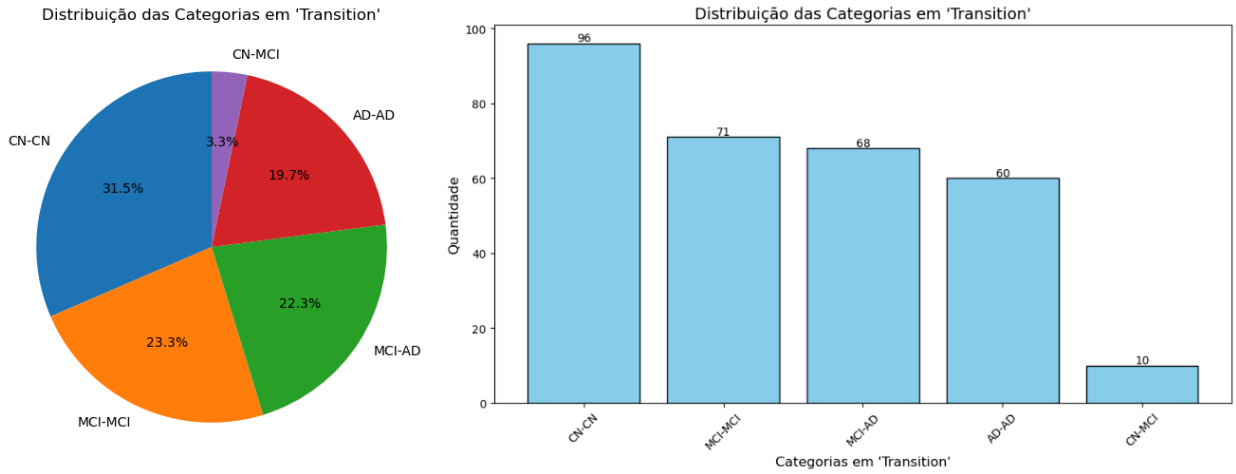

Figura 1. Gráficos da distribuição do atributo *Transition* em relação ao *dataset*.

Como é possível observar nos gráficos, a transição **CN-MCI** é a menos representada em comparação com as restantes. Esta disparidade pode dificultar a capacidade dos nossos modelos em prever essa transição, uma vez que a quantidade limitada de informação disponível é significativamente menor em relação às transições mais frequentes.

Por fim, podemos observar a relação entre ***Sex*** e ***Transition*** na Figura 2, que possui algumas variações na distribuição de homens e mulheres em diferentes categorias. Em transições como *CN-CN*, a proporção entre homens e mulheres é relativamente equilibrada, enquanto em outras, como *MCI-AD* e *MCI-MCI*, observa-se uma predominância de homens. Por outro lado, na categoria *CN-MCI*, o número de mulheres é ligeiramente superior ao de homens. Estas diferenças na distribuição por género podem indicar padrões relevantes associados às transições analisadas, merecendo atenção em estudos futuros.

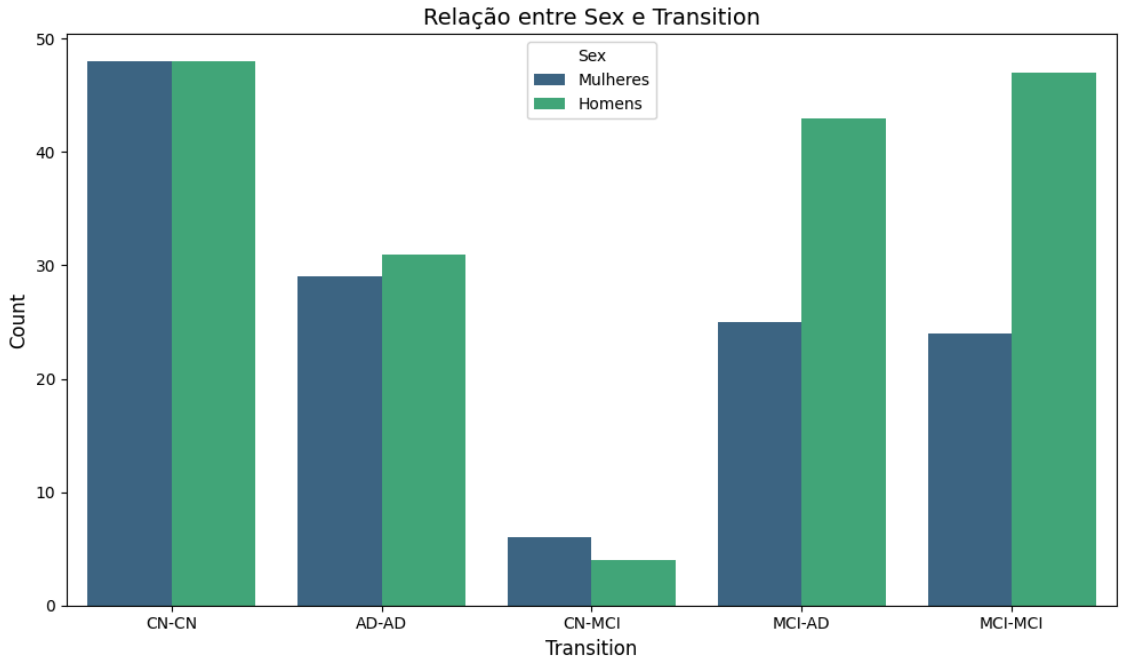

Figura 2. Gráfico da distribuição do atributo *Sex* pelas classes do atributo *Transition*.

4 Tratamento dos Dados

As tratamentos de dados que estão detalhados nesta secção foram aplicados de forma igual aos *datasets* do Hippocampus e ao *dataset* do Occipital Lobe para serem usados nos testes de previsão.

4.1 Remoção das Colunas com o Mesmo Valor em Todas as Entradas

A remoção das colunas com o mesmo valor em todas as entradas foi realizada, uma vez que estas não acrescentavam qualquer valor informativo ao *dataset*. Colunas deste tipo não fornecem variabilidade ou distinção entre as observações, sendo irrelevantes para análises ou modelos de previsão, apenas aumentaria a dimensão do *dataset* de forma desnecessária.

4.2 Remoção de Colunas com Entradas Alfanuméricas

Foram removidas todas as colunas que possuíam dados alfanuméricos com exceção da coluna ***Transition***, pois não forneciam dados relevantes para a solução do problema e incapacitavam a execução dos modelos de previsão de dados.

4.3 Conversão da Coluna *Age*

A coluna ***Age*** foi convertida do tipo *float* para o tipo *int*, dado que a precisão decimal é irrelevante para esta variável. Como os valores representam idades, faz mais sentido trabalhar com números inteiros, simplificando a análise.

4.4 Conversão da Coluna *Transition*

A coluna ***Transition*** foi convertida para valores numéricos, passando a ser representada da seguinte maneira:

- CN-CN $\rightarrow$ 1
- AD-AD $\rightarrow$ 2
- CN-MCI $\rightarrow$ 3
- MCI-AD $\rightarrow$ 4
- MCI-MCI $\rightarrow$ 5

Com o objetivo de facilitar o processamento e a previsão desta variável. Esta conversão simplifica a manipulação dos dados, especialmente em modelos de *machine learning*, que operam de forma mais eficiente com valores numéricos.

4.5 Verificação e Remoção de *Outliers*

A verificação e remoção de ***Outliers*** foi realizada para garantir que o *dataset* estivesse livre de valores extremos que pudessem distorcer os resultados das análises e previsões.

5 Modelação de Dados

Para a modelação de dados, testámos diversos tipos de modelos, incluindo algoritmos de *Clustering*, *Bagging*, *Gradient Boosting*, *SVM*, entre outros. Após várias iterações e testes, os modelos selecionados foram escolhidos com base no seu desempenho e relevância para o problema em estudo.

5.1 Estratégias de Modelação

Para desenvolver modelos de previsão de dados foram escolhidos os seguintes algoritmos lecionados nas aulas:

- ***Random Forest Classifier***
- ***MLP***
- ***XGBoost***

5.1.1 *Random Forest Classifier*

O ***Random Forest Classifier*** é um modelo de aprendizagem supervisionado baseado em árvores de decisão, que utiliza o *ensemble learning* para melhorar a precisão e reduzir o risco de *overfitting*. Consiste na combinação de várias árvores de decisão treinadas em subconjuntos aleatórios dos dados, destacando-se pela sua robustez, versatilidade e eficácia para lidar com dados ruidosos e desequilibrados.

Na nossa implementação, o modelo foi otimizado utilizando o método de busca em ***Grid*** (***GridSearchCV***), com o objetivo de identificar os melhores parâmetros que maximizassem o *Macro F1-Score*, a métrica principal definida pela competição. Após várias iterações e ajustes, os seguintes parâmetros foram selecionados:

Parâmetros	**Valores**	**Descrição**
n_estimators	*50*	Representa o nº de árvores na floresta.
bootstrap	*False*	Define se as amostras para construir cada árvore são extraídas com reposição.
class_weight	*balanced*	Ajusta o peso atribuído a cada classe.
max_depth	*10*	Limita a profundidade máxima de cada árvore.
max_features	*log2*	Determina o nº de características consideradas em cada divisão
min_samples_leaf	*2*	Especifica o nº mínimo de amostras necessárias para formar uma folha.
min_samples_split	*20*	Define o nº de amostras necessárias para dividir um nó.

Após o treino do modelo com os parâmetros selecionados, este foi utilizado para realizar previsões no conjunto de teste. Embora a sua performance geral não tenha sido das melhores, destacou-se por ser o único modelo capaz de prever, durante o treino, a classe minoritária do dataset, algo que os restantes modelos não conseguiram alcançar.

5.1.2 *MLP*

O ***MLP*** (*Multilayer Perceptron*) é um modelo de rede neural *feedforward* que consiste em múltiplas camadas densas interligadas e que utiliza funções de ativação não lineares para capturar padrões complexos nos dados. Este modelo é amplamente utilizado em problemas de classificação multiclasse devido à sua flexibilidade e capacidade de modelar relações não lineares nos dados.

Sendo assim, para maximizar o desempenho do modelo, utilizamos o método de seleção em *GridSearch* para identificar as melhores combinações de parâmetros. Essa abordagem sistemática permitiu explorar diferentes configurações do modelo e selecionar os parâmetros mais adequados para otimizar sua performance.

Parâmetros	**Valores**	**Descrição**
hidden_sizes	*[64,32]*	Define a arquitetura da rede neural, com o número de neurônios em cada camada oculta.
lr	*0.001*	Especifica a taxa de aprendizagem utilizada pelo otimizador.
dropout_rate	*0.3*	Indica a proporção de conexões descartadas para prevenir *overfitting*.
weight_decay	*0.00001*	Aplica regularização *L2* para penalizar pesos excessivos e reduzir *overfitting*.

Após o treino do modelo com os parâmetros selecionados, concluímos que apesar de ser um modelo capaz de lidar com problemas complexos não se mostrou o mais adequado para a resolução do problema em causa.

5.1.3 *XGBoost*

O **XGBoost** (eXtreme Gradient Boosting) é um modelo de árvores de decisão que utiliza a técnica de gradient boosting para melhorar a precisão preditiva e, destacasse pela capacidade de lidar com grandes volumes de dados, fornecer alta precisão preditiva e incorporar regularização para evitar *overfitting* o que, para o caso que tivemos a explorar, foi essencial principalmente pelo tamanho do *dataset*.

Na nosso implementação com o modelo **XGBoost** usamos uma classificação multiclasse, usando o algoritmo XGBClassifier com o objetivo de prever cinco classes e, tomando partido, da função de perda `multi:softprob`. O modelo foi otimizado utilizando a validação cruzada estratificada com cinco divisões e uma busca em ***Grid*** **(GridSearchCV)** para explorar hiperparâmetros como taxa de aprendizado, profundidade máxima das árvores e parâmetros de regularização. Para isso usamos os seguintes valores para Gridsearch, após várias iterações de exploração:

Parâmetros	**Valores**[*]	**Descrição**
learning_rate	*0 -* ***0.01*** *- 0.005*	O tamanho do passo para atualização dos pesos durante o treino, controla quanto o modelo aprende em cada iteração
n_estimators	*400 -* ***800*** *- 1000*	O número de árvores no modelo ensemble
max_depth	*2 -* ***5.0*** *- 12.0*	A profundidade máxima de cada árvore, determinando a complexidade do modelo e o risco de *overfitting*
gamma	*0 -* ***0.1*** *- 12.0*	Um parâmetro de regularização que controla a redução mínima de perda necessária para dividir um nó
min_child_weight	*0 -* ***1*** *- 12.0*	A soma mínima dos pesos das instâncias para necessária um nó filho, afetando o crescimento da árvore

[*] A coluna "Valores" representa o range de valores usados para os parâmetros na exploração deste modelo, a **negrito** encontra-se o valor usado em cada parâmetro para o melhor resultado no *score* privado.

Parâmetros	Valores†	Descrição
colsample_bytree	*0 - **1** - 12.0*	A fração de características de amostra para cada árvore, auxilia a seleção de features e reduz o *overfitting*
reg_alpha	*0 - 12.0*	Termo de regularização L1 nos pesos, incentivando a "*sparcity*" e reduzindo a complexidade do modelo.
reg_lambda	*0 - 12.0*	Termo de regularização L2 nos pesos, ajudando a controlar o *overfitting* ao penalizar pesos grandes

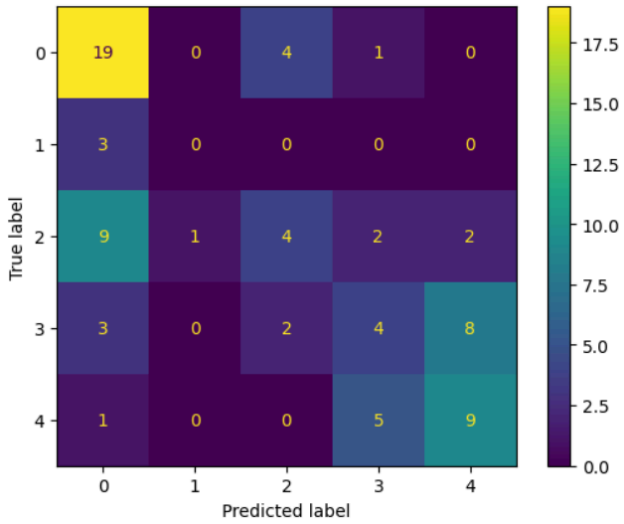

Figura 3. Matrix de confusão gerada para XGBoost.

Após o treino, o melhor modelo foi identificado e utilizado para realizar previsões no conjunto de teste. Este modelo foi o que apresentou o melhor resultados com o dataset disponibilizado, tanto no *private score*, como no *public score*.

5.1.4 *Overfitting*

Uma das grandes preocupações no desenvolvimento dos modelos foi mitigar o risco de *overfitting*, que acontece, quando o modelo aprende muito bem os padrões do conjunto de treino, mas não generaliza bem para novos dados. Desse modo, vários métodos foram usados para calcular e salvaguardar o *overfit* nos vários modelos, neste relatório e para efeito de demonstração, iremos destacar o que fizemos para esse efeito nos modelos:

1. **Análise**: Começamos por analisar e tentar mitigar o *overfit* criando métricas dos resultados de cada *run* e, para isso, usamos a *flag* `return_train_score=True` na *GridSearch* obtendo os seguintes resultados:

†A coluna "Valores" representa o range de valores usados para os parâmetros na exploração deste modelo, a **negrito** encontra-se o valor usado em cada parâmetro para o melhor resultado no *score* privado.

mean_train_score	mean_test_score
1	0.323658
0.995034	0.335403
...	...
0.706912	0.310258
0.704182	0.317886

que representam, respetivamente:

- **mean_train_score**: Mostra a média dos *scores* de avaliação do modelo no conjunto de treino para a combinação de hiperparâmetros usados.
- **mean_test_score**: Exibe a média dos *scores* de avaliação no conjunto de teste para a mesma combinação de hiperparâmetros.

Para além disso também geramos uma tabela com a informação da curva de aprendizagem, que mostra uma maior precisão do treino e da validação conforme o número de exemplos de treino aumenta:

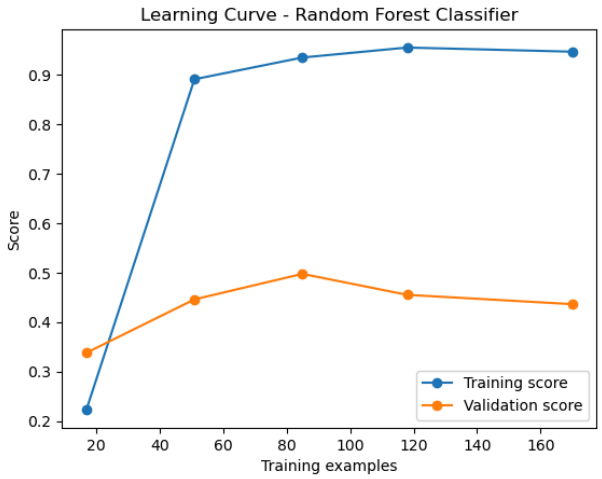

Figura 4. Gráfico gerado com *training score* e *validation score* do modelo Random Forest Classifier

Sabendo que *training scores* muito altos ou quase perfeitos sugerem *overfitting*, a partir destes resultados podemos concluir que há a indicação de *overfitting*, para além disso, uma grande discrepância entre os valores de validação/teste em relação aos de treino também demonstram a possiblidade de *overfitting*, o que é o caso.

2. **Mitigar**: Com base nas análises desempenhadas, para cada modelo tentámos usar vários técnicas de mitigação de Overfit como, por exemplo, para o XGBoost com Grid Search tentámos usar as seguintes técnicas:

   - Reduzir o parâmetro *max_depth*, uma menor profundidade limita a complexidade de cada árvore individual e força o modelo a aprender padrões mais gerais

- Aumentar a regularização do *reg_alpha* e *reg_lambda*, para forçar restrições mais rigorosas no crescimento das árvores, e controlar a redução mínima necessária na função de perda para fazer uma nova divisão.
- Introduzir os parâmetros *subsample*, que controla a fração de amostras usadas em cada árvore e *colsample_bytree*, que controla a fração de features usadas por árvore. A combinação desses parâmetros adiciona aleatoriedade ao processo que ajuda a prevenir que o modelo memorize os dados de treinamento.

No entanto, é importante de **notar**, que não conseguimos alcançar uma boa redução de overfit e manter um bom *score* de previsão com o uso destas técnicas sobre várias iterações, visto que, a redução de overfit não era substancial em relação á perda do *score* e, desse modo, estas modificações não foram usadas nos modelos finais.

6 Resultados

Tabela 1. Resultados do modelo Random Forest Classifier

Class	Precision	Recall	F1-Score	Support
0	0.47	0.79	0.59	24
1	1.00	0.17	0.29	6
2	0.43	0.12	0.19	24
3	0.27	0.32	0.29	19
4	0.36	0.42	0.39	19
accuracy			0.40	92
macro avg	0.51	0.36	0.35	92
weighted avg	0.43	0.40	0.37	92

Tabela 2. Resultados do modelo XGBoost

Class	Precision	Recall	F1-Score	Support
0	0.54	0.79	0.64	24
1	0.00	0.00	0.00	3
2	0.40	0.22	0.29	18
3	0.33	0.24	0.28	17
4	0.47	0.60	0.53	15
accuracy			0.46	77
macro avg	0.35	0.37	0.35	77
weighted avg	0.43	0.47	0.43	77

Tabela 3. Resultados do modelo MLP

Class	Precision	Recall	F1-Score	Support
0	0.54	0.72	0.62	29
1	0.00	0.00	0.00	3
2	0.20	0.10	0.13	21
3	0.21	0.19	0.20	21
4	0.48	0.61	0.54	18
accuracy			0.41	92
macro avg	0.29	0.32	0.30	92
weighted avg	0.36	0.41	0.37	92

Tabela 4. Resultados do modelo XGBoost com o dataset Occipital

Class	Precision	Recall	F1-Score	Support
0	0.36	0.31	0.33	96
1	0.00	0.00	0.00	10
2	0.35	0.13	0.19	71
3	0.28	0.28	0.28	68
4	0.16	0.33	0.22	60
accuracy			0.26	305
macro avg	0.23	0.21	0.20	305
weighted avg	0.29	0.26	0.25	305

Como demonstram os resultados, obtidos com `classification_report` apresentados nas tabelas acima, o modelo que alcançou o melhor desempenho foi o **XGBoost**. Observa-se também que os modelos **Random Forest Classifier** (RFC) e **MLP** apresentam métricas semelhantes, sendo a principal diferença, já mencionada anteriormente, o facto de o *RFC* ter conseguido prever a classe minoritária, ao contrário do *MLP*. Além disso, comparando os resultados das tabelas 2 e 4, onde foram usados os conjuntos de dados do **Hipocampo**

e do **Occipital**, respetivamente, nota-se uma diferença nas métricas obtidas, permitindo concluir que os dados do *Hipocampo* contribuem mais eficazmente para a previsão dos padrões de transição de Alzheimer em comparação com os do *Occipital*.

7 Conclusão e Trabalho Futuro

Neste trabalho, exploramos diferentes modelos de *machine learning* incluindo *Random Forest*, *MLP*, e *XGBoost*, destacando o *Random Forest* pois foi o único dos modelos a conseguir prever a classe minoritária e o *XGBoost* que apresentou a melhor previsão dos dados.

Relativamente à nossa performance na competição do *Kaggle*, apesar de não termos alcançado uma classificação ideal, consideramos que foi positiva. Quando foram reveladas as classificações da competição privada, verificámos que a discrepância entre posições não foi significativa, o que reforça a ideia de que o nosso trabalho, de forma geral, foi bem-sucedido nos diversos aspetos avaliados na competição.

Para trabalho futuro, será essencial focar em estratégias mais eficazes para combater o *overfitting*, garantindo que os modelos generalizem melhor para novos dados. Além disso, é necessário aprofundar a análise e a aplicação de técnicas específicas para lidar com classes minoritárias, como métodos de *oversampling* mais avançados ou ajustes nas métricas, de modo a garantir que todas as classes sejam bem representadas e previstas pelos modelos. Essas melhorias podem contribuir significativamente para a eficiência e previsibilidade dos modelos desenvolvidos.