Universidade do Minho
Escola de Engenharia

Trabalho Prático

Grupo 23

Dados e Aprendizagem Automática 2024/2025

Eduardo Cunha PG55939


João Magalhães PG55956

Jorge Rodrigues PG55966


Rodrigo Gomes PG56004

Setembro 2024, Ano Letivo 24/25

Conteúdo

Lista de Figuras

Introdução

O presente trabalho centra-se na previsão da progressão do Défice Cognitivo Ligeiro (DCL), ou em inglês, *Mild Cognitive Impairment* (MCI), para a doença Alzheimer, um desafio crítico na área da neurologia. O DCL caracteriza-se por um declínio cognitivo superior ao expectável para a idade e nível de educação do indivíduo, embora não interfira significativamente com as atividades diárias. Este estado representa uma fase transitória entre o envelhecimento normal e a doença de Alzheimer. O problema específico consiste em desenvolver modelos de Aprendizagem Automática capazes de prever, com base em características radiómicas extraídas de imagens de Ressonância Magnética (RM) cerebral, se um paciente com DCL irá progredir para Alzheimer ou manter-se estável.

O trabalho divide-se em duas tarefas principais.Uma tarefa de análise e validação, que visa confirmar a relevância do hipocampo na previsão da evolução das demências, e uma vertente competitiva, focada no desenvolvimento e otimização de modelos.

Este problema tem relevância clínica, dado que a identificação precoce dos casos de DCL com maior risco de progressão para Alzheimer pode permitir intervenções terapêuticas mais atempadas, contribuindo potencialmente para a redução do número de casos de demência a longo prazo.

1 Metodologia

A metodologia eleita para este projeto baseia-se no ciclo apresentado no decorrer da Unidade Curricular. Sendo assim, seguiremos uma sequência que inclua os seguintes pontos:

- **Definição do Problema**, etapa que envolve compreender e definir claramente o problema
- **Análise e Ingestão de Dados**, fase dedicada às fontes dos dados, assim como a sua leitura
- **Preparação dos Dados**, fase onde os dados passam por uma série de transformações para deixá-los no formato ideal para a modelação
- **Modelação**, etapa onde os modelos são treinados
- **Avaliação**, corresponde à fase onde se compara o desempenho dos modelos

2 Definição do Problema

A progressão do Défice Cognitivo Ligeiro (DCL) para a Doença de Alzheimer (DA) é um desafio crítico na área da neurologia, dado o impacto significativo desta doença na qualidade de vida dos pacientes e das suas famílias. Este trabalho centra-se na previsão dessa progressão utilizando modelos de aprendizagem automática baseados em características radiómicas extraídas de imagens de ressonância magnética (RM) cerebral.

O objetivo principal é desenvolver modelos capazes de prever se pacientes com DCL irão progredir para Alzheimer ou permanecer estáveis, explorando especificamente a relevância do hipocampo como região de interesse, em comparação com outras áreas como o lobo occipital. Este problema possui grande relevância clínica, pois a identificação precoce de pacientes com alto risco de progressão pode possibilitar intervenções terapêuticas que reduzam os casos de demência a longo prazo.

Para isso, serão analisados dois conjuntos de dados (hipocampo e lobo occipital), avaliados e preparados para modelagem. Além disso, será realizada uma tarefa competitiva de design e otimização de modelos preditivos, cuja eficácia será avaliada através de métricas como a F1 macro score.

3 Observação e Ingestão dos Dados

3.1 Fonte de Dados

Os Dados para o presente trabalho estão disponíveis no kaggle, e nesta mesma plataforma não está disponível referencia a hospital/clínica onde as Ressonâncias Magnéticas foram efetuadas. Ainda, foi disponibilizado pela

equipa docente um segundo conjunto de Dados de controlo para auxiliar na avaliação dos modelos. O *Dataset* consta com 305 entradas e com 2181 colunas, onde a maioria corresponde a valores numéricos. Dada a tecnicidade dos atributos (nomenclatura técnica de imagens médicas), assim como o seu volume, optamos por não perder tempo com análises individuais dos atributos.

3.2 Análise de Dados

A análise de dados é um dos procedimentos mais importantes para a modelação. Nesta etapa, procuramos perceber qual será o tratamento mais adequado para a construção eficiente dos modelos.

3.2.1 Compreensão dos Dados

Como primeiro passo na análise do *Dataset*, decidimos verificar a sua estrutura, conferindo os tipos de colunas, a existência de valores em falta, entradas duplicadas e colunas com valores constantes. Face a estes pontos, levantamos o seguinte:

- Não existem valores em falta
- Não existem entradas duplicadas
- Os valores categóricos são: ID, Image, Mask, diagnostics_Versions_PyRadiomics, ... Transition, num total de 20 colunas, onde a coluna Transition é o nosso *target*
- Existem 159 colunas com valores constantes

3.2.2 Visualização dos Dados

A forma mais expressiva de interpretar padrões nos dados é através de gráficos. Desta forma, começamos por visualizar a distribuição das entradas face ao nosso *target*.

Visualização da Distribuição das entradas pela Target

Segue abaixo um esquema com a distribuição dos dados; como se pode verificar, estão bastante desequilibrados.

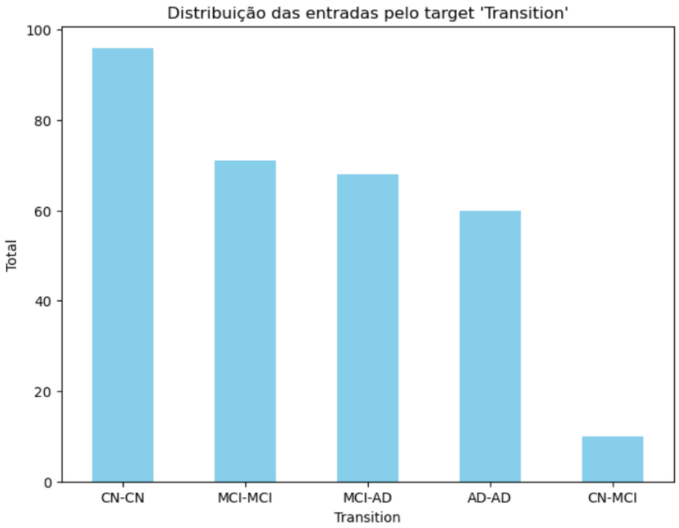

Figura 1: Distribuição dos Dados

Visualização da Correlação dos Dados

Apresenta-se, de seguida, uma amostra da correlação dos dados. Estas amostras foram obtidas através da aplicação de certos filtros às features. Devido ao elevado número inicial de features, foi necessário selecionar e apresentar apenas a correlação de um subconjunto representativo, em vez de incluir todas de forma simultânea.

As seguintes imagens foram geradas com a seleção das *features* geradas apartir dos filtros LBP e Wavelet:

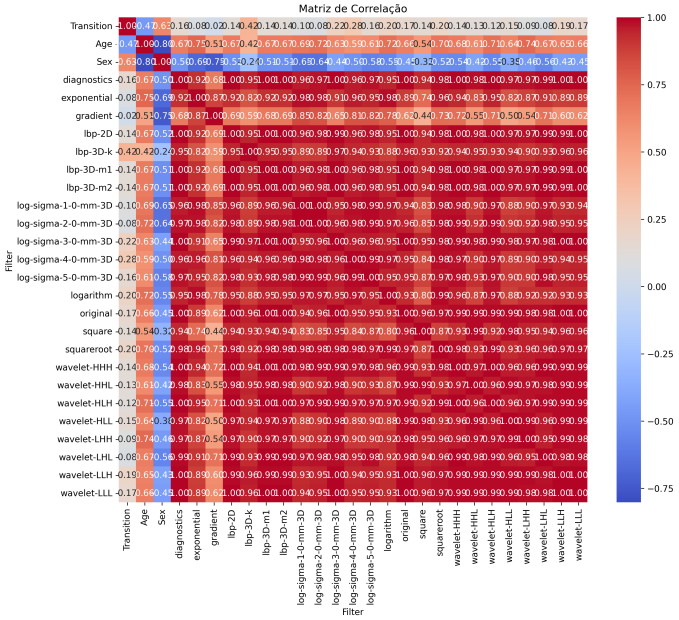

Figura 2: Heatmap de correlação com a aplicação do filtro LBP.

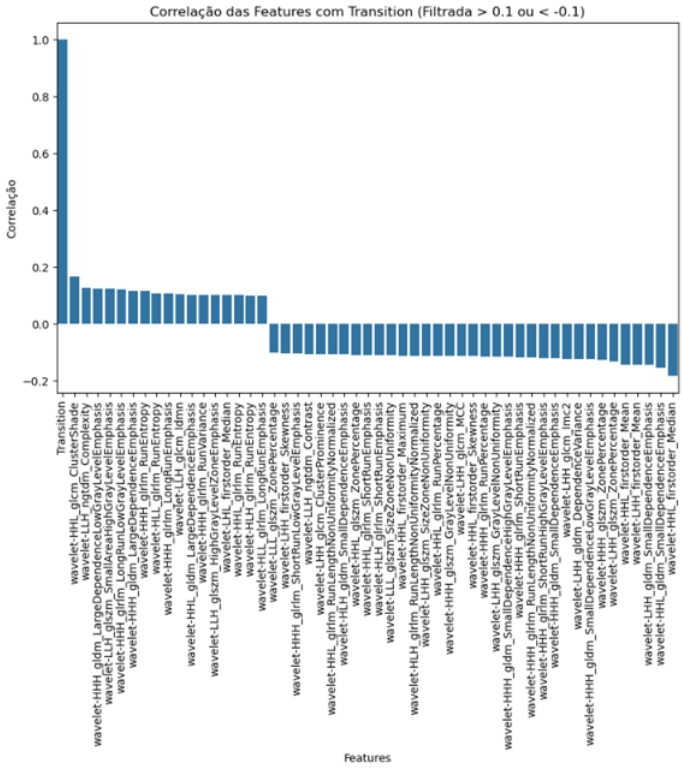

Figura 3: Gráfico de correlação com a aplicação do filtro Wavelet.

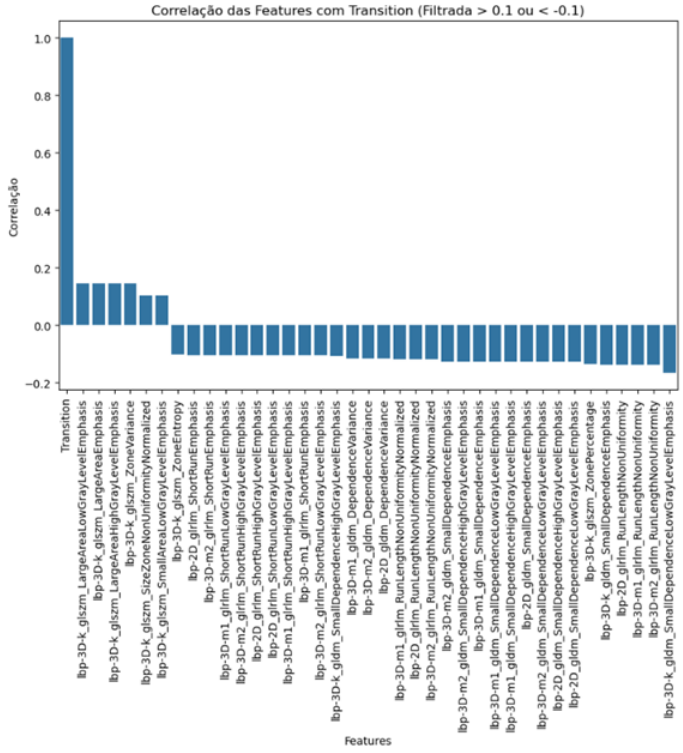

Figura 4: Gráfico de correlação com a aplicação do filtro LBP.

Das figuras observámos que, para a grande maioria das features, a correlação com a *feature* alvo é muito baixa, sendo menor que 0.3, e também que é percetível que haja elevada correlação entre *features* geradas a partir do mesmo filtro.

4 Preparação de Dados

4.1 Remoção de colunas não numéricas

Com a analise das colunas não numéricas, reparamos que estes atributos eram meramente identificadores, como a coluna *ID*, referencias para as imagens, como a coluna *Image*, ou até mesmo dados das configurações utilizadas, tal como as colunas *diagnostics_Versions_Numpy* e *diagnostics_Imageoriginal_Hash*

4.2 Remoção de colunas com valores constantes

Como referido, existem diversas colunas que apresentam o mesmo valor para todas as entradas. Desta forma, sabemos que não há conhecimento útil a ser gerado, pelo que prosseguir para a sua remoção é sensato. Desta forma, esperamos melhorar os tempos de treino dos modelos.

4.3 Normalização de Dados

Observamos que, dentro dos dados numéricos, existem escalas com variedades muito diferentes. Para melhorar o desempenho dos modelos, optamos por aplicar técnicnas de normalização.

4.3.1 MinMax

Esta técnica é a mais simples, consiste em dividir todos os valores de um determinado atributo pelo seu respetivo máximo. É especialmente útil quando não é conhecido o tipo de distribuição do atributo.

4.3.2 Standardization (Z-Score Normalization)

Esta técnica utiliza a média e o desvio padrão para transformar os dados numa distribuição normal. Geralmente eficiente quando as distribuições são consistentes.

4.4 Remoção de Outliers

A remoção de *outliers* é uma tarefa que deve ser sempre considerada, uma vez que esses valores atípicos podem influenciar significativamente o desempenho dos modelos. *Outliers* são observações que se distanciam significativamente do padrão geral dos dados e podem surgir por diversas razões, como erros de medição, variações naturais

ou condições experimentais anómalas. Dada o elevado número de colunas, a observação através de *BoxPlots* não nos foi possível, sendo que optamos por utilizar a técnica *Z-Score*.

4.5 Seleção das *K* melhores Colunas

Decidimos também escolher as *K* melhores *features*, isto porque o número de colunas dos dados é bastante elevado, o que será pesado para a parte da modelação. Para este efeito, testamos diversas técnicas de redução de *feature*, tais como: PCA, onde consideramos uma divergência de 95% ; Recursive Feature Elimination, onde um conjunto de *features* é selecionado por eliminação de *features* menos importantes; ANOVA, que se serve de cálculos estatísticos para a reduzir o conjunto de dados.

4.6 Balanceamento dos Dados

É importante salientar que verificámos um grande desbalanceamento nos dados, o que levava à generalização de vários modelos para o caso mais predominante no conjunto de dados. Testámos técnicas como o SMOTE para balanceamento dos dados e até mesmo a criação sintética de outros, mas isso resultou numa redução significativa nos resultados obtidos. Como acreditamos que os dados de teste privados estão igualmente desbalanceados, decidimos deixar de lado a preparação com SMOTE.

4.7 Preparações Utilizadas

Para a fase de modelação, combinamos estas técnicas de modo a maximizar as capacidades dos modelos.

Preparação 1
Remoção de colunas não numéricas
Normalização MinMax

Tabela 1: Preparação 1

Preparação 2
Remoção de colunas não numéricas
Remoção de colunas constantes
Remoção de outliers
Normalização Z-Score

Tabela 2: Preparação 2

Preparação 3
Remoção de colunas não numéricas
Remoção de colunas constantes
Normalização MinMax

Tabela 3: Preparação 3

Preparação 4
Remoção de colunas não numéricas
Remoção de colunas constantes
PCA
Normalização MinMax

Tabela 4: Preparação 4

Preparação 5
Remoção de colunas não numéricas
Remoção de colunas constantes
Normalização MinMax
Eliminação de Features com Cross-Validation

Tabela 5: Preparação 5

Preparação 6
Remoção de colunas não numéricas
Remoção de colunas constantes
Normalização MinMax
Eliminação de Features com Cross-Validation
SMOTE

Tabela 6: Preparação 6

Preparação 7
Remoção de colunas não numéricas
Remoção de colunas constantes
Feature Selection com ANOVA
Normalização MinMax

Tabela 7: Preparação 7

5 Modelação

5.1 Método de Trabalho

O nosso método de trabalho seguiu uma abordagem iterativa e sistemática com o objetivo de identificar o modelo mais adequado para o problema em questão. O processo consistiu nos seguintes passos:

1. **Seleção de Algoritmo:** Iniciámos por escolher um algoritmo de aprendizagem adequado ao problema, considerando as suas características e potencial de desempenho.

2. **Otimização de Hiperparâmetros:** Realizámos (*Grid Search*) combinada com (*Cross-Validation*) sobre todos os dados. O objetivo deste passo foi otimizar os hiperparâmetros do modelo para alcançar um desempenho equilibrado e robusto.

3. **Avaliação do Modelo:** Após identificar o melhor conjunto de hiperparâmetros, procedemos à avaliação do modelo, analisando métricas como o F1-score macro e gerando a matriz de confusão para obter uma visão detalhada sobre o desempenho por classe.

4. **Submissão no Kaggle (Opcional):** O modelo foi submetido na plataforma Kaggle para obter os resultados na competição.

5. **Registo de Resultados:** Os resultados obtidos foram registados numa folha de cálculo (*Excel*), permitindo uma análise comparativa entre diferentes abordagens e configurações.

6. **Revisão e Iteração:** Com base nos resultados, ajustámos o nosso processo, voltando ao passo inicial para explorar novos algoritmos ou refinar os modelos existentes.

É importante salientar que o objetivo principal nunca foi obter o maior valor possível, quer localmente, quer no Kaggle, pois isso poderia resultar em overfitting. Pelo contrário, o foco foi sempre reduzir a discrepância entre os valores obtidos localmente e os valores apresentados no Kaggle, garantindo uma generalização sólida do modelo.

Segue-se um esquema que ilustra o nosso método de trabalho:

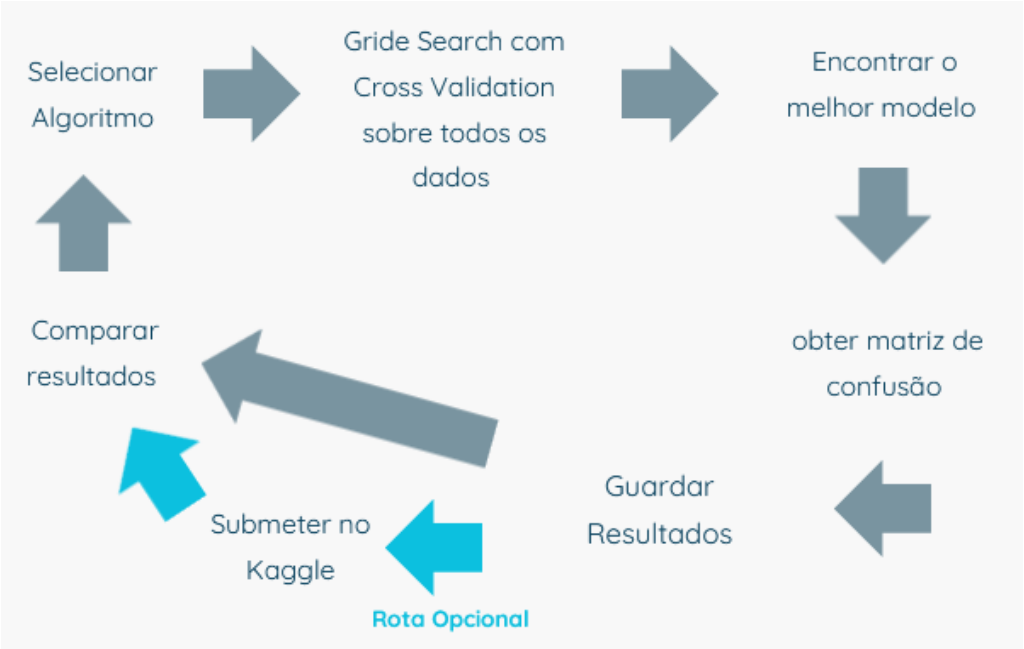

Figura 5: Método de Trabalho

É importante salientar que foram testadas muitas mais combinações do que as apresentadas nas tabelas que irão ser apresentadas de seguida. Contudo, apenas as combinações mais relevantes foram incluídas neste relatório.

5.2 Ensemblers

Os métodos de ensemble combinam múltiplos modelos para melhorar o desempenho preditivo em comparação com modelos individuais. Eles exploram a diversidade e a complementaridade das previsões de diferentes algoritmos, reduzindo o risco de overfitting e aumentando a robustez. A seguir, discutimos as principais abordagens de ensembles utilizadas neste trabalho.

5.2.1 Bagging

O *bagging* é uma técnica de ensemble que combina modelos treinados em subconjuntos diferentes dos dados, gerados por amostragem com reposição. A agregação final é feita através de uma votação (para classificação).

Preparação	n_estimators	F1-macro	Kaggle	Diff
1	100	0.335723023	/	/
2	100	0.327265027	/	/
3	100	0.325427311	/	/
4	100	0.260252105	/	/
5	50	0.373371477	0.3251	0.048271
6	300	0.708276411	/	/
7	300	0.335320281	/	/

Tabela 8: Resultados de Bagging com diferentes valores de n_estimators

De todas as técnicas exploradas, observamos que os resultados, por comparação, não foram os melhores. Este fenómeno era já esperado, visto que de todas as técnicas escolhidas, o Bagging é a mais simples. Ainda assim, a diferença comparativamente aos restantes modelos foi pequena, variando apenas algumas décimas. Atribuimos esta pequena diferença à falta de diversidade que alguns algoritmos, como as *Random Forest* conseguem captar.

5.2.2 Random Forest

O *Random Forest* é um método de *ensembling* baseado em árvores de decisão que utiliza o conceito de bagging para criar um conjunto de árvores independentes. Cada árvore é treinada num subconjunto diferente dos dados e com uma seleção aleatória de atributos, introduzindo maior diversidade entre os modelos. A decisão final é obtida por meio da votação das árvores (no caso de classificação).

Tabela 9: Resultados Random Forest

Prep	n_estimators	max_depth	criterion	max_features	F1-macro	Kaggle	Diff
1	100	5	entropy	log2	0.352970724	0.39298	0.040009276
2	300	5	entropy	None	0.3365891168	/	/
3	100	20	entropy	log2	0.336931552	0.31972	0.017211552
4	500	10	entropy	sqrt	0.277658465	/	/
5	50	20	gini	None	0.358217787	0.3504	0.007817787
6	500	20	entropy	sqrt	0.726276622	0.29079	0.435486622
7	300	20	entropy	sqrt	0.364177667	0.38099	0.024372333

Esta técnica foi das mais bens sucedidas, não só entre *ensemblers*, como também entre as diferentes preparações. Isto deve se a características como o número máximo de *features* considerados pelos estimadores, o que retém um impacto considerável num *dataset* como o nosso, assim como outros mecanismos de controlo de profundidade, que são especialmente úteis para lidar com problemas de *overfitting*. O modelo que consideramos um dos candidatos a melhor generalização foi concebido com a preparação 5, com os hiperparâmetros especificados na tabela 9.

5.2.3 Boosting

O *boosting* é uma técnica iterativa de ensemble que constrói modelos sequenciais, onde cada modelo posterior tenta corrigir os erros dos modelos anteriores. Algoritmos como *AdaBoost*, *Gradient Boosting* e *XGBoost* são exemplos populares dessa abordagem. O *boosting* é conhecido por seu alto desempenho em competições e aplicações práticas, especialmente em cenários com dados complexos e desbalanceados.

Tabela 10: Resultados Gradient Boosting

Prep	n_estimators	max_depth	learning_r	max_features	F1-macro	Kaggle	Diff
1	100	5	0.1	sqrt	0,36091994	0.2730	0,087909946
2	50	5	0.1	sqrt	0,337514053	/	/
3	50	5	0.3	none	0,34531770	0.40015	0,05483229
4	100	5	0.1	none	0,30322347	/	/
5	100	10	0.3	sqrt	0,3417128	0,09364276	/
6	100	10	0.3	sqrt	0,7353971	/	/
7	100	20	0.3	sqrt	0,3343648	/	/

Tabela 11: Resultados XGBoosting

Preparação	n_estimators	max_depth	learning_rate	F1-macro	Kaggle	Diff
1	50	5	0.1	0.34849889	0.30851	0.03998800
2	300	5	0.1	0.32311066	/	/
3	50	5	0.1	0.34849809	0.31212	0.03637800
4	50	5	0.1	0.28788948	/	/
5	50	5	0.3	0.32751658	/	/
6	300	5	0.3	0.70815849	/	/
7	100	0	0.3	0.33275179	/	/

Os resultados obtidos indicam que as técnicas de *boosting* apresentaram desempenhos competitivos em relação ao Random Forest na maioria dos casos, embora existam exceções em que este foi superior. Entre as abordagens de *boosting*, o Gradient Boosting mostrou-se mais eficaz do que o XGBoost para a resolução do problema em questão.Este fenómeno pode ser atribuído a uma configuração mais adequada dos hiperparâmetros no caso do Gradient Boosting, enquanto que, no XGBoost, a regularização possivelmente não foi ajustada de maneira ideal, limitando o seu desempenho.

5.2.4 Stacking

O *stacking* é uma técnica de ensemble que combina modelos base (primeira camada) com um modelo meta (segunda camada) para fazer previsões finais. Esta abordagem permite aproveitar as forças individuais de cada modelo base, enquanto o modelo meta aprende a combinar as previsões para melhorar a performance geral. O

stacking é particularmente eficaz em cenários complexos, onde diferentes algoritmos capturam diferentes aspetos dos dados.

Preparação	Meta model	Modelos	F1-macro	Kaggle	Diff
1	RandomForest	Rf, GB, SVM	0.324699587	0.34652	0.021820413
3	RandomForest	Rf, GB, SVM	0.305552525	/	/
4	RandomForest	Rf, GB, SVM	0.333704187	0.25241	0.081294187
5	RandomForest	Rf, GB, SVM	0.293556556	0.31103	0.017473444
6	RandomForest	Rf, GB, SVM	0.756487516	/	/
7	RandomForest	Rf, GB, SVM	0,332924057	0.3323	0.000624057

Tabela 12: Resultados de Stacking

Os resultados obtidos para a técnica de stacking ficaram aquém das expectativas. A inclusão do SVM não trouxe a diferenciação esperada, e mesmo com a diversificação do *meta model*, tanto em termos de algoritmo quanto de configuração, os resultados não se destacaram. Acreditamos que a principal limitação esteja na elevada complexidade dos modelos base, o que dificultou o desempenho colaborativo. A combinação utilizada não foi eficaz para lidar com a complexidade do problema, resultando em desempenhos abaixo do esperado.

5.2.5 Max Voting

O método de *max voting* é uma técnica simples de ensemble em que a previsão final é determinada pela classe mais votada entre os modelos base.

Preparação	Modelos	Pesos	F1-macro	Kaggle	Diff
1	GB, SVM, RF	[2,1,3]	0.338322464	/	/
3	GB, SVM, RF	[3,1,2]	0.345316348	0.27619	0.069126
4	GB, SVM, RF	[5,3,1]	0.303223475	/	/
5	GB, SVM, RF	[5,3,1]	0.357627429	0.25334	0.104287
6	GB, SVM, RF	[3,1,2]	0.7490927	0.3492	0.399893
7	GB, SVM, RF	[3,3,1]	0.342865921	/	/

Tabela 13: Results of Voting with Cross Validation

Os resultados obtidos com o método de Max Voting foram superiores aos do stacking em diversos casos. Isso pode ser atribuído à simplicidade da abordagem, que evita a complexidade adicional de treinar um *meta model*, o que pode ser uma vantagem quando os *ensemblers* não são tão compatíveis/combináveis. O Max Voting aproveita diretamente os pontos fortes dos modelos base, priorizando a classe mais votada sem a necessidade de ajustes adicionais. Esse comportamento parece ter permitido um melhor equilíbrio entre os modelos, resultando em desempenhos mais consistentes, especialmente nas configurações menos complexas.

5.3 SVM

O *Support Vector Machine* (SVM) é um método de que procura o hiperplano que melhor separa as classes no espaço dos dados. A técnica utiliza margens máximas para garantir maior generalização e pode ser aplicada tanto em problemas lineares quanto não lineares, através do uso de *kernels*, que projetam os dados em espaços dimensionais maiores.

De acordo com o que foi lecionado nas aulas teóricas, os SVMs demonstram uma excelente capacidade de generalização em casos com muitas features, o que explica a sua utilização na maioria dos ensembles anteriores e despertou a nossa curiosidade em testá-los como um modelo simples.

Preparação	C	kernel	gamma	F1-macro	Kaggle	Diff
1	1	linear	scale	0.314632233	/	/
3	1	rbf	scale	0.297462045	/	/
4	1	rbf	scale	0.320738023	/	/
5	100	sigmoid	scale	0.340245549	0.27993	0.060316
6	100	rbf	scale	0.744010658	/	/
7	1	rbf	scale	0.313844853	/	/

Tabela 14: Resultados de SVM com diferentes parâmetros

Os resultados obtidos com SVM foram interessantes, apresentando desempenhos que, embora inferiores em alguns casos aos *ensemblers*, mostraram diferenças menores do que o esperado. Isso indica que o SVM, mesmo como um modelo simples, conseguiu capturar parte da estrutura dos dados e generalizar de maneira razoável. Achamos interessante que, dependendo da preparação, o *kernel* ótimo varia, o que ajudou a compreender a importância de preparar os dados.

5.4 Redes Neuronais

Estávamos bastante interessados no desenvolvimento de redes neuronais, apesar de estarmos conscientes, desde o início, das limitações do dataset disponível. Estas limitações condicionaram significativamente os resultados obtidos, uma vez que as redes neuronais necessitam de uma diversidade e quantidade significativa de dados para aprender de forma eficaz, algo que o nosso dataset não permitia.

Conforme esperado, os resultados alcançados não foram satisfatórios. Procurámos otimizar os hiperparâmetros de forma criteriosa e explorámos diversas abordagens de pré-processamento de dados. Adicionalmente, desenvolvemos dois tipos distintos de redes neuronais: uma baseada no modelo apresentado nas aulas, que seguia uma arquitetura de aprendizagem mais robusta, e outra mais simplificada, concebida para explorar um modelo de menor complexidade e evitar possíveis problemas de *overfitting*.

Apesar dos esforços, os resultados permaneceram aquém das expectativas. Observámos que os processos de redução de dimensionalidade (feature reduction) aplicados no pré-processamento, embora úteis em outros contextos, tenderam a piorar o desempenho do modelo, possivelmente devido à perda de informações relevantes para a aprendizagem e, por outro lado, às características intrínsecas das redes neuronais, que são concebidas para atribuir pesos quase nulos às features irrelevantes de forma eficiente ao longo do treino.

O melhor resultado obtido foi um F1-score máximo de 0,23 e uma classificação de 0,19557 no Kaggle, valores que confirmam as dificuldades associadas ao dataset e à complexidade do problema abordado.

5.5 Clustering

Relativamente ao clustering, sabíamos que esta não seria a técnica mais adequada, dado que o dataset já incluía a variável que pretendíamos prever, tornando-o mais apropriado para métodos não supervisionados. Ainda assim, decidimos testar técnicas de aprendizagem não supervisionada, para avaliar o seu desempenho. Para tal, construímos modelos utilizando K-Means, DBSCAN e Affinity Propagation. Infelizmente, os resultados foram extremamente decepcionantes, com o modelo DBSCAN a ser incapaz de realizar a clusterização do dataset de forma eficaz.

Entre os modelos testados, o K-Means foi o que apresentou resultados mais promissores, com um F1-score de 0,23 para $k = 5$. Este valor de k foi definido previamente, dado o conhecimento prévio da possibilidade de agrupar as respostas em cinco categorias distintas. No entanto, de acordo com a análise do "$elbowmethod$", o valor mais adequado seria $k = 2$, para o qual se obteve um F1-score de 0,17.

Relativamente ao DBSCAN, como já mencionado, não conseguiu realizar qualquer clusterização útil. Por outro lado, o modelo Affinity Propagation tentou criar 27 grupos diferentes, resultando em valores irrisórios de F1-score de 0,01.

Ainda, tentamos também utilizar algumas destas técnicas para *Over Sampling* e *Under Sampling*, mas não foi possível, dado que uma das categorias alvo foi considerada insuficientemente representada para esse efeito.

5.6 *Exploitation*

Decidimos avançar com *exploitation* para as preparações promissoras, com o objetivo de aproveitar ao máximo os modelos mais eficazes desenvolvidos na fase anterior, tentando agregar os seus valores da melhor forma possível. Além disso, introduzimos também algumas técnicas não exploradas anteriormente, pois queremos incluir uma análise mais diversificada, que não ficasse limitada pelos classificadores lecionados na Unidade Curricular.

Posto isto, começamos por selecionar as 2 preparações que consideramos mais relevantes. Escolhemos as preparações 3 e 5, visto que os resultados obtidos para os modelos concebidos sobre estes dados foram os que

consideramos melhor, em termos de f1_score e *overfitting*. Daqui, seguimos diversas abordagens, tais como:

5.6.1 Experimentar com novos hiperparâmetros

Até agora, utilizávamos uma *pool* limitada de valores para cada hiperparâmetro dos modelos testados. Decidimos expandir esta abordagem na tentativa de obter melhores resultados em teste. No entanto, percebemos que, a menos que optássemos por pequenas alterações, como somar 2 ou 3 estimadores em algoritmos como o Random Forest, os resultados não apresentavam mudanças significativas. Em muitos casos, acreditamos que estávamos caminhando para situações de *overfitting*, o que nos levou a descartar essa abordagem para a maioria dos modelos.

No caso específico do SVM, foi o único modelo que realmente beneficiou desta nova abordagem. Devido aos períodos de treino mais curtos, conseguimos testar uma maior variedade de combinações de hiperparâmetros, o que resultou em melhorias mais significativas na performance.

5.6.2 Novas técnicas

Adicionalmente, exploramos novas técnicas de *ensembling*, como o AdaBoost e o ExtraTreesClassifier. Os resultados foram semelhantes aos obtidos com os *ensemblers* que já estávamos a utilizar. Contudo, uma análise mais detalhada das matrizes de confusão revelou que o AdaBoost apresentava um comportamento distinto, com uma melhor performance na classe "MCI-AD", mas pior desempenho nas demais classes. Por outro lado, o ExtraTreesClassifier não trouxe resultados excecionalmente diferentes dos já analisados.

Também investigamos o uso de técnicas para dados desbalanceados, como o EasyEnsembleClassifier, o RUSBoostClassifier, e o BalancedRandomForestClassifier. Estes métodos mostravam-se promissores devido à sua capacidade de lidar com a desproporcionalidade entre classes. No entanto, o nosso dataset é, sobretudo, grande em número de colunas, mas pequeno em entradas. Esta limitação no número de amostras disponíveis parece ter comprometido a eficácia destas técnicas, que não apresentaram o desempenho esperado.

5.6.3 Stacking e Voting

A partir do pressuposto de que combinar modelos poderia aproveitar o melhor de cada um, experimentamos as técnicas de Stacking e Max Voting. Contudo, os resultados mostraram que essa abordagem nem sempre é eficaz para o nosso problema.

No caso do Stacking, a complexidade na seleção de um bom *meta model* foi um obstáculo significativo. Apesar dos esforços em diversificar a abordagem, os resultados ficaram aquém das expectativas. Essa limitação pode ser atribuída à dificuldade em capturar a complementaridade dos modelos no contexto do nosso dataset.

Já o Max Voting, apresentou resultados inconsistentes.Nas situações observados, os resultados ou pioraram, ou apenas um modelo base acabava dominante sobre os demais, o que anulava o benefício esperado da combinação

5.6.4 Over sampling e Under sampling

Durante o projeto, testamos diversas técnicas de over sampling, como SMOTE, SMOTENC, SMOTEN e ADASYN. Apesar da sua popularidade para lidar com datasets desbalanceados, os resultados em treino apresentaram alterações severas, principalmente devido à sub-representação da classe "CN-MCI". Essa classe gerou inúmeras entradas ruidosas quando o dataset foi ampliado.

No caso do under sampling, utilizamos métodos como ClusterCentroids e TomekLinks. Contudo, essas abordagens enfrentaram os mesmos problemas observados na secção "Novas técnicas". A redução do dataset para equilibrar as classes acabou removendo informações relevantes das classes minoritárias e maioritárias, o que comprometeu o desempenho final dos modelos

5.6.5 Considerações sobre *Exploitation*

Os resultados esperados não foram os melhores, ficando muito aquém das expectativas. Com isto, consideramos que a fase de exploração mais ampla foi o real sucesso na nossa abordagem, e que o nosso foco deveria ter sido ainda superior nessa mesma etapa e não tanto na abordagem mais gananciosa de *Exploitation*.

6 Avaliação

6.1 Comparação de Modelos

Conforme referido anteriormente, as métricas utilizadas para a comparação dos modelos foram o F1-score macro e as submissões realizadas no Kaggle. O F1-score macro revela-se uma opção particularmente adequada, pois oferece uma visão equilibrada do desempenho entre as diferentes classes, mesmo em cenários com desequilíbrios nos dados.

Para modelos promissores, utilizamos também o dataset de controlo para efetivar se, a generalização do modelo, é viável e baseada em fatores relacionados com o problema em mãos.

Verificou-se, uma clara superioridade dos modelos de ensemble em relação aos modelos de redes neuronais, de clusterização e aos modelos mais simples.

Os resultados obtidos indicam que o modelo "RandomForest com a preparação 5"apresentou o melhor desempenho em termos de F1-score macro, enquanto o modelo "GradientBoost com a preparação 3"demonstrou maior robustez nas submissões públicas do Kaggle.

6.2 Impacto das Técnicas de Preparação

Entre as diversas técnicas de preparação de dados testadas, consideramos que a técnica 3 e a 5 foram as mais eficazes, devido à prestação nos testes locais, considerando também o valor de referência no dataset público.

Deve-se ainda destacar a preparação 6, que gerava ótimos resultados localmente, com F1-scores a rondar os 0,70, devido à utilização do SMOTE, que criava exemplos sintéticos para o caso **CM-MCI**, resultando em previsões consistentemente corretas. No entanto, esses valores eram irreais, como confirmado pelas submissões no Kaggle, onde se observou uma redução drástica para valores à volta de 0,30 de F1-score.

6.3 Dificuldades

Os erros identificados durante o desenvolvimento dos modelos evidenciaram algumas limitações importantes:

- **Classes desbalanceadas**: O desequilíbrio das classes teve um impacto negativo no desempenho inicial de alguns modelos, especialmente nas classes minoritárias.
- **Overfitting**: Alguns modelos demonstraram dificuldades em generalizar para os dados de teste devido a um ajuste excessivo aos dados de treino.
- **Qualidade dos dados**: A presença de um volume reduzido de dados com um número elevado de atributos prejudicou o desempenho dos modelos.

6.4 Desempenho no Kaggle

O desempenho nas competições do Kaggle ficou aquém das expectativas, embora isso tenha sido uma tendência geral observada entre os participantes. Nos testes públicos, alcançámos a **29ª posição**, enquanto nos testes privados optámos pela seguinte submissão:

Prep	n_estimators	max_depth	criterion	max_features	F1-macro	Kaggle	Diff
5	50	20	gini	None	0.358217787	0.3504	0.007817787

Obtivemos a **26ª posição**, o que demonstra que o modelo manteve uma generalização relativamente boa, apesar das limitações.

Considerámos relevante destacar esta submissão, que obteve um score de 0,41 nos testes privados, o que teria garantido a sexta posição:

Prep	n_estimators	max_depth	learning_r	max_features	F1-macro	Kaggle	Diff
3	50	5	0.3	none	0,34531770	0.40015	0,05483229

Estas duas submissões foram desde sempre as que considerámos mais interessantes. Decidimos selecionar o modelo Random Forest como submissão final, dado que apresentou um desempenho significativamente superior no dataset de controlo (ainda que, mesmo assim, longe do ideal). Além disso, a Random Forest foi também a submissão com o resultado mais elevado e menor diferença entre os resultados obtidos localmente e no Kaggle.

7 Conclusão

Neste trabalho, abordámos o desafio de prever a progressão do Défice Cognitivo Ligeiro (DCL) para a Doença de Alzheimer (DA), recorrendo a modelos de aprendizagem automática baseados em características radiómicas extraídas de imagens de ressonância magnética.

Consideramos que os objetivos do projeto foram atingidos. Contudo, identificámos uma limitação significativa relacionada com o reduzido tamanho do dataset disponível. Com apenas 100 entradas, torna-se extremamente desafiante desenvolver modelos preditivos com desempenho robusto e capacidade de generalização. Esta limitação é especialmente evidente em modelos como as redes neuronais, que dependem de grandes volumes de dados para treinar eficazmente os seus parâmetros e captar padrões complexos.

Apesar desta restrição, implementámos diversas técnicas de preparação de dados e modelação para mitigar os impactos do tamanho reduzido do dataset. Estas técnicas incluem a normalização, a remoção de outliers e a seleção das melhores variáveis, o que contribuiu para a obtenção de resultados promissores. Ainda assim, reconhecemos que a ampliação do conjunto de dados seria essencial para melhorar a eficácia dos modelos desenvolvidos, especialmente no caso das redes neuronais.

Consideramos que o trabalho realizado fornece uma base sólida para futuras investigações, que deverão centrar-se na obtenção de conjuntos de dados mais abrangentes.