Alzheimer:

un enfoque desde la Ciencia de Datos

La enfermedad de Alzheimer es una de las principales causas de demencia, afectando principalmente a personas mayores y causando un deterioro cognitivo progresivo.

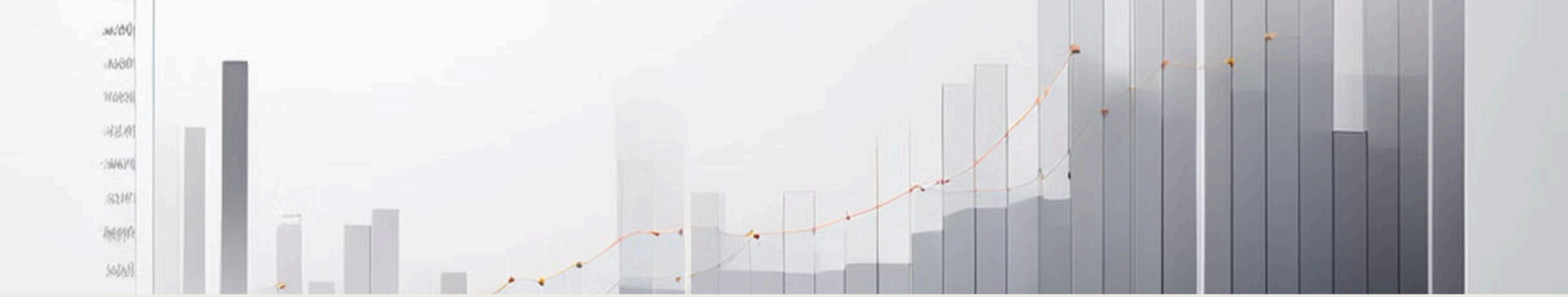
Este proyecto tiene como objetivo construir un modelo predictivo para identificar la enfermedad de Alzheimer con alta precisión, enfocándose en minimizar los falsos negativos .

Usando un dataset de la plataforma Kaggle, exploraremos técnicas de aprendizaje automático para desarrollar una herramienta de diagnóstico menos invasiva, más rápida y económica que los métodos tradicionales.

Descripción de los Datos

El dataset incluye 2149 muestras con 35 variables:

Tipo de Datos	Descripción variable
ID Paciente	**PatientID:** Número de identificación para cada paciente.
Demográficos	**Age:** Edad del paciente. **Gender:** Género del paciente (1 para hombre, 0 para mujer). **Ethnicity:** Origen étnico del paciente (0: caucásico, 1: afroamericano, 2: asiático, 3: Otro) **EducationLevel:** Nivel de educación del paciente (0: ninguno, 1: escuela secundaria, 2: licenciatura, 3: superior)
Estilo de Vida	**BMI:** Índice Masa corporal. **Smoking:** Fumador (1 para fumador, 0 para no fumador) **AlcoholConsumption:** Cantidad de consumo de alcohol **PhysicalActivity:** Nivel de actividad física. **DietQuality:** Calidad de la dieta. **SleepQuality:** Calidad del sueño.
Historial Médico	**FamilyHistoryAlzheimers**: Historia familiar de la enfermedad de Alzheimer (1 para sí, 0 para no). **CardiovascularDisease**: Presencia de enfermedad cardiovascular (1 para sí, 0 para no). **Diabetes**: Presencia de diabetes (1 para sí, 0 para no). **Depression**: Presencia de depresión (1 para sí, 0 para no). **HeadInjury**: Historial de lesiones en la cabeza (1 para sí, 0 para no). **Hypertension**: Presencia de hipertensión (1 para sí, 0 para no).
Mediciones Clínicas	**SystolicBP**: Presión arterial sistólica. **DiastolicBP**: Presión arterial diastólica. **CholesterolTotal**: Nivel de colesterol total. **CholesterolLDL**: Nivel de colesterol LDL. **CholesterolHDL**: Nivel de colesterol HDL. **CholesterolTriglycerides**: Nivel de triglicéridos.
Evaluaciones Cognitivas y Funcionales	**MMSE**: Puntuación del Mini-Examen del Estado Mental. **FunctionalAssessment**: Puntuación de la evaluación funcional. **MemoryComplaints**: Quejas sobre la memoria (1 para sí, 0 para no). **BehavioralProblems**: Presencia de problemas de conducta (1 para sí, 0 para no). **ADL**: Puntuación de las actividades de la vida diaria.
Síntomas	**Confusion**: Presencia de confusión (1 para sí, 0 para no). **Disorientation**: Presencia de desorientación (1 para sí, 0 para no). **PersonalityChanges**: Presencia de cambios de personalidad (1 para sí, 0 para no). **DifficultyCompletingTasks**: Dificultad para completar tareas (1 para sí, 0 para no). **Forgetfulness**: Presencia de olvido (1 para sí, 0 para no).
Información de Diagnóstico	**Diagnosis**: Diagnóstico de la enfermedad de Alzheimer (1 para sí, 0 para no). **DoctorInCharge**: Información confidencial, valor establecido en "XXXConfid" para todos los pacientes.



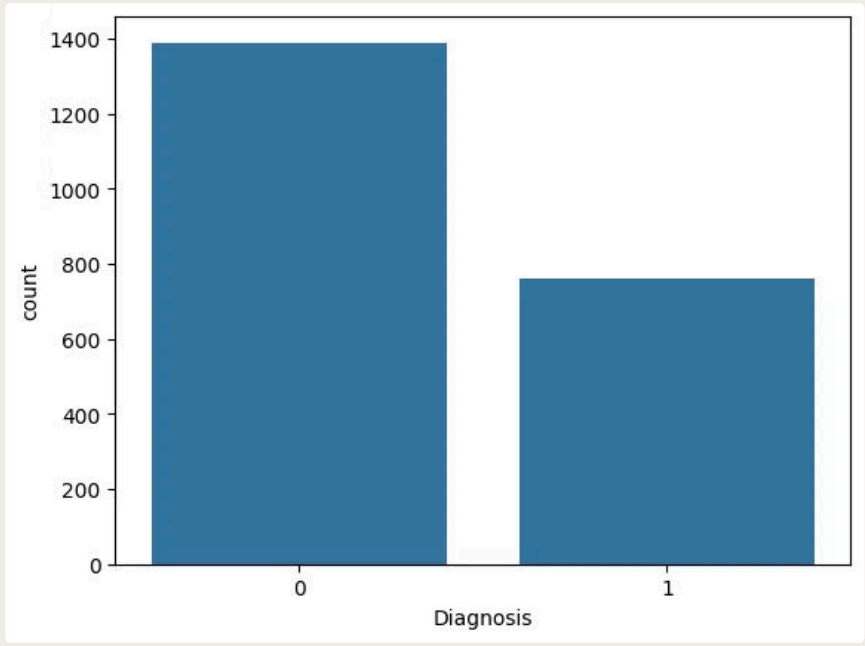
Exploración y Limpieza de Datos

1 Identificación de Variable Objetivo

Se identificó '**Diagnosis**' como la variable objetivo

2 Análisis de Distribución

Se detectó un desbalance entre clases, con más pacientes no diagnosticados que diagnosticados.



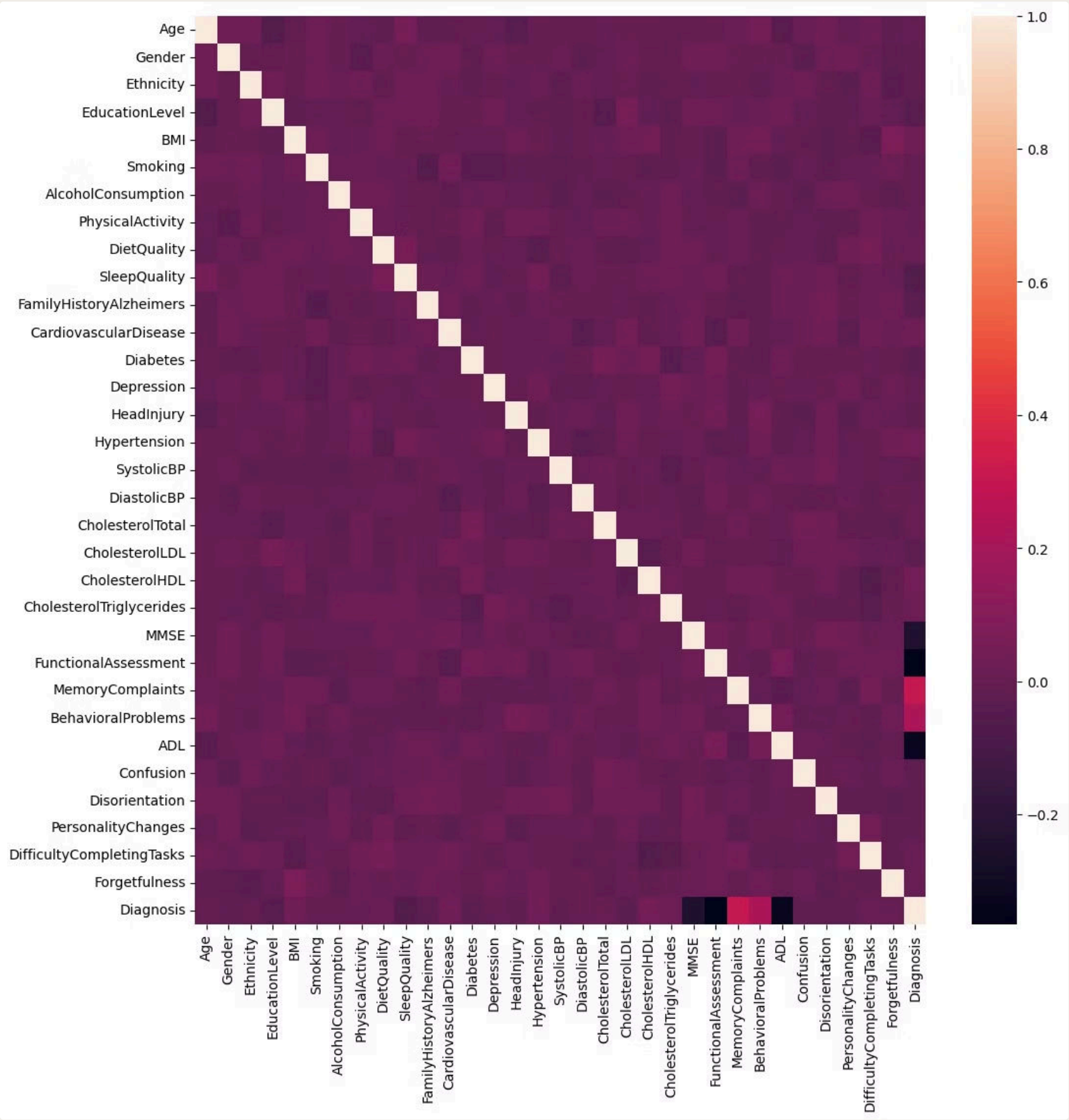
3 Limpieza de Datos

No se encontraron duplicados ni valores faltantes. Se descartaron variables sin valor predictivo como '**PatientID**' y '**DoctorInCharge**'.

Análisis de Correlación

Matriz de correlación

Se identificaron 'FunctionalAssessment', 'MemoryComplaints', 'ADL', 'BehavioralProblems' y 'MMSE' como las variables con mayor correlación con nuestra variable objetivo **'Diagnosis'**

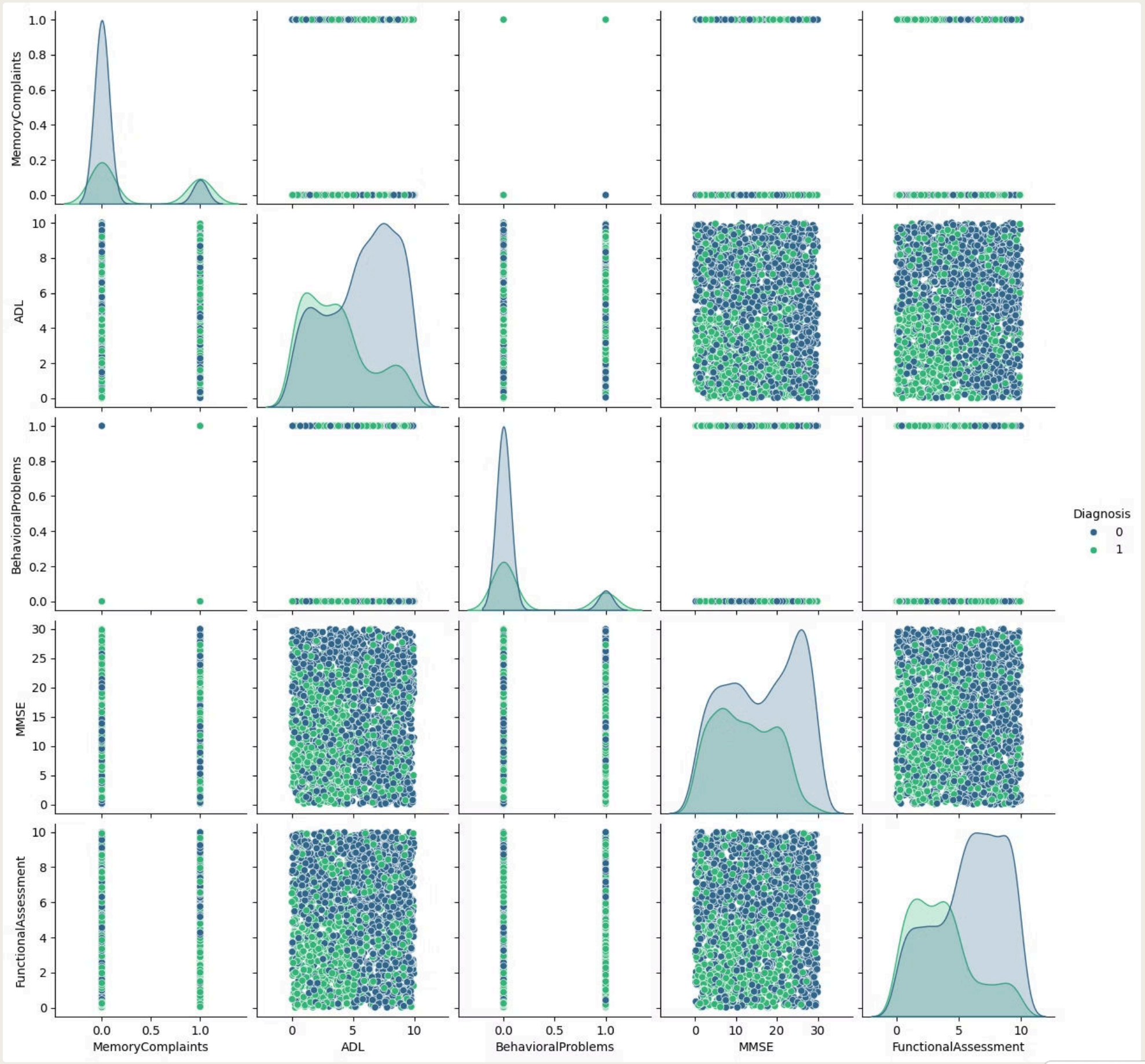


Análisis de Correlación

Gráfico de dispersión (Pair Plot) de las variables más significativas

Pairplot proporciona una visión de cómo se distribuyen y relacionan las distintas variables del conjunto de datos con respecto a la variable objetivo - diagnóstico.

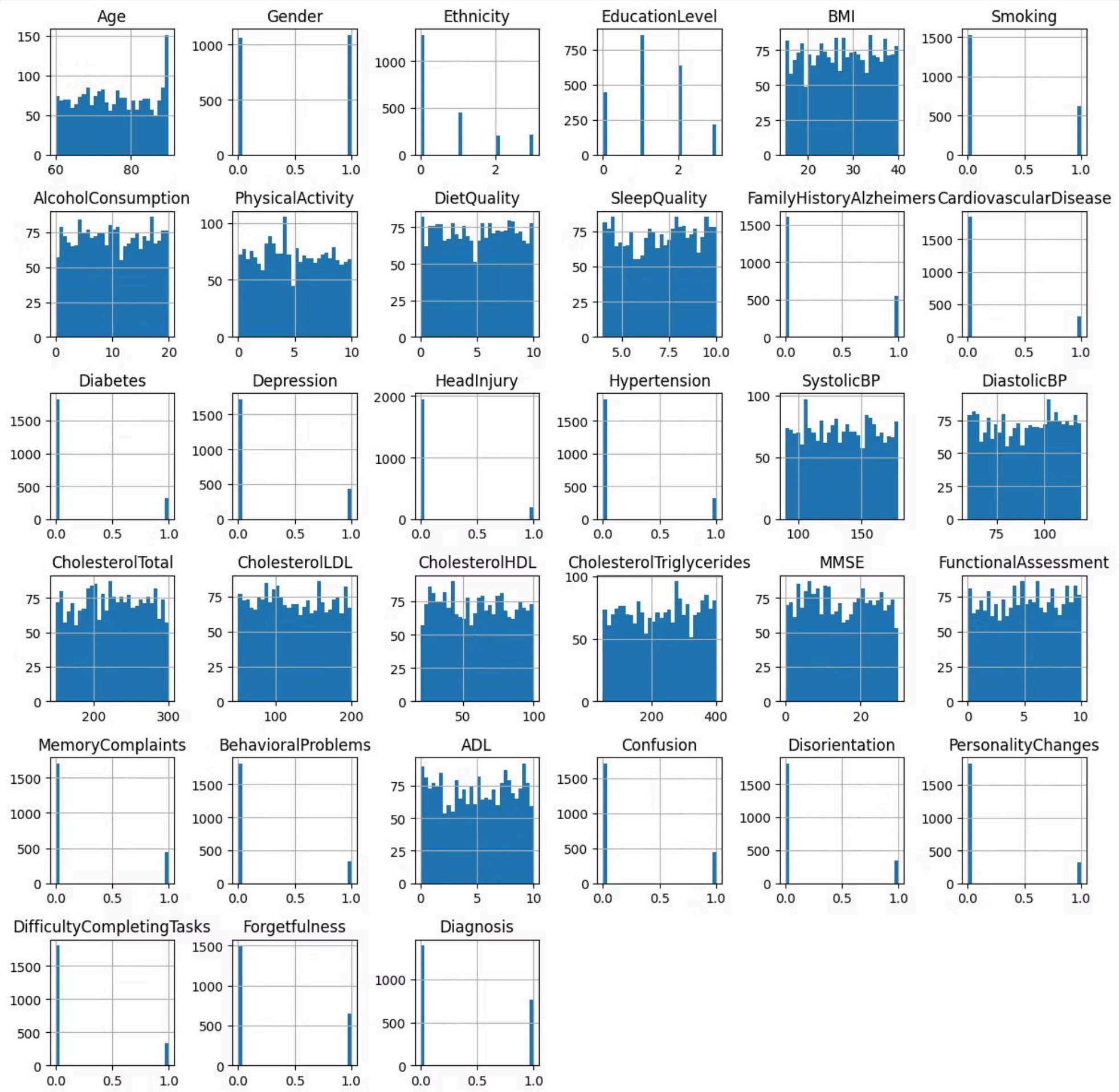
Las variables muestran patrones distintivos que pueden ser útiles para predecir el diagnóstico.



Distribución de Variables y Outliers

Histograma para todas las variables

Los histogramas muestran las distribuciones de variables con bastante uniformidad, aunque la variable **Age** tiene una distribución asimétrica negativa que es esperable en ese tipo de pacientes.



Feature Engineering

1

Transformación de Variables Categóricas

Se utilizó la técnica de 'get_dummies' para binarizar variables categóricas como 'Ethnicity' y 'EducationLevel'.

2

Balanceo del Dataset

Para poder compensar el desbalanceo en la variable objetivo - Diagnósis, probamos dos estrategias :

- **Undersampling** del dataset usando resample
- **Oversampling** del dataset usando SMOTE

3

Creación de nuevos Datasets

1. **Clean**: con todas las muestras menos las variables **PatientID**' y '**DoctorInCharge**'
2. **Balanced**: un subset de Clean con Undersampling
3. **SMOTE**: un subset de Clean con Oversampling

Algoritmos ML y entrenamiento

1

Segmentación Train/Test

Se dividió los datasets en 80% para entrenamiento (Train) y 20% para prueba (Test).

2

Escalado de datos

Se escalaron los datos con **StandardScaler.**

3

Selección de Modelos

Se usaron varios modelos de clasificación incluyendo: Logistic Regression, Decision Tree, Random Forest SVM, KNN, XGBoost y Naive Bayes.

4

Selección de Métricas de Evaluación

Se utilizaron Accuracy, Balance Accuracy, Precision, Recall y F1-Score poniendo especial énfasis en Recall, también conocido como **sensibilidad** o **tasa de verdaderos positivos. I**dentifica las instancias correctamente clasificadas como positivas.

5

Evaluación de Modelos

Model	Dataset	Train Accuracy	Test Accuracy	Train Balanced Accuracy	Test Balanced Accuracy	Train Precision	Test Precision	Train Recall	Test Recall	Train F1 Score	Test F1 Score
Logistic Regression	CLEAN	0.8668	0.8381	0.8667	0.8383	0.8673	0.8385	0.8668	0.8381	0.8667	0.8381
	BALAN	0.8372	0.8289	0.8371	0.8285	0.8372	0.8289	0.8372	0.8289	0.8372	0.8289
	SMOTE	0.8668	0.8381	0.8667	0.8383	0.8673	0.8385	0.8668	0.8381	0.8667	0.8381
Decision Tree	CLEAN	1	0.8255	1	0.826	1	0.8267	1	0.8255	1	0.8255
	BALAN	1	0.9046	1	0.9042	1	0.9047	1	0.9046	1	0.9046
	SMOTE	1	0.8219	1	0.8222	1	0.8224	1	0.8219	1	0.8219
Random Forest	CLEAN	1	0.9083	1	0.9092	1	0.9129	1	0.9083	1	0.9081
	BALAN	1	**0.9441**	1	**0.9441**	1	**0.9441**	1	**0.9441**	1	**0.9441**
	SMOTE	1	0.9047	1	0.9057	1	0.9106	1	0.9047	1	0.9044
Support Vector Machine	CLEAN	0.955	0.8471	0.955	0.8474	0.9552	0.8476	0.955	0.8471	0.955	0.8471
	BALAN	0.9424	0.8388	0.9424	0.8385	0.9425	0.8388	0.9424	0.8388	0.9424	0.8388
	SMOTE	0.955	0.8471	0.955	0.8474	0.9552	0.8476	0.955	0.8471	0.955	0.8471
K-Nearest_Neighbors	CLEAN	0.856	0.7572	0.8562	0.7555	0.861	0.7647	0.856	0.7572	0.8555	0.7549
	BALAN	0.8125	0.6711	0.8123	0.6729	0.8133	0.6755	0.8125	0.6711	0.8123	0.6703
	SMOTE	0.856	0.7572	0.8562	0.7555	0.861	0.7647	0.856	0.7572	0.8555	0.7549
XGBoost	CLEAN	1	0.9191	1	0.9197	1	0.9211	1	0.9191	1	0.919
	BALAN	1	**0.9507**	1	**0.9505**	1	**0.9507**	1	**0.9507**	1	**0.9507**
	SMOTE	1	0.9191	1	0.9197	1	0.9211	1	0.9191	1	0.919
Naive Bayes	CLEAN	0.784	0.7878	0.784	0.7877	0.784	0.7878	0.784	0.7878	0.784	0.7878
	BALAN	0.8051	0.7862	0.8052	0.7871	0.8054	0.7882	0.8051	0.7862	0.8051	0.7862
	SMOTE	0.784	0.7878	0.784	0.7877	0.784	0.7878	0.784	0.7878	0.784	0.7878

Conclusiones

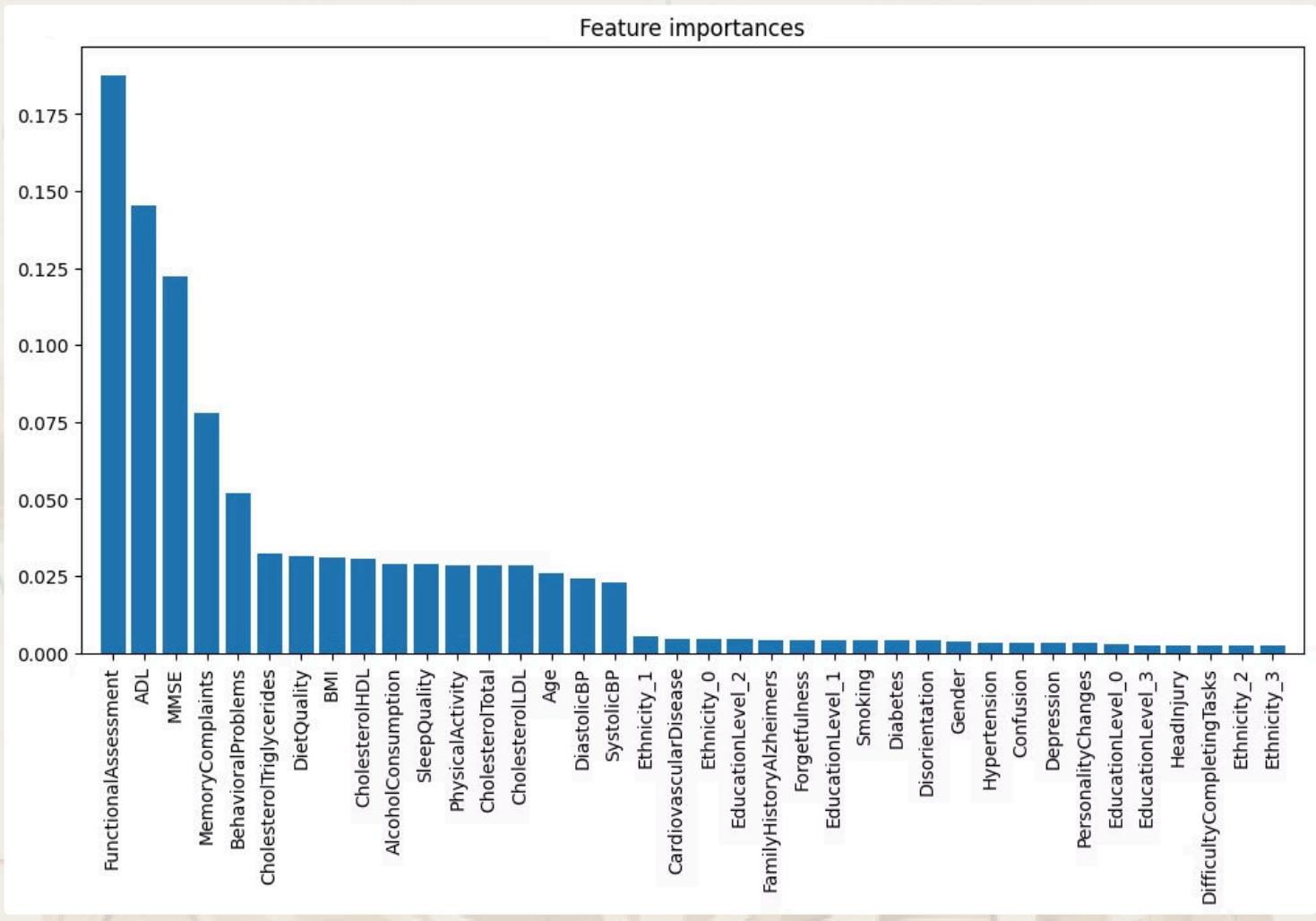
XGBoost y **Random Forest** mostraron el mejor rendimiento, especialmente en el dataset balanceado.

Los siguientes pasos consistirán en optimizar los dos modelos y determinar cual es el modelo final.

Optimización del Modelo Random Forest

1 Selección de Variables

Se evaluó la importancia de las variables utilizando **BalancedRandomForestClassifier**, generando dos datasets con las 10 y 5 mejores variables.



2 Ajuste de Hiperparámetros

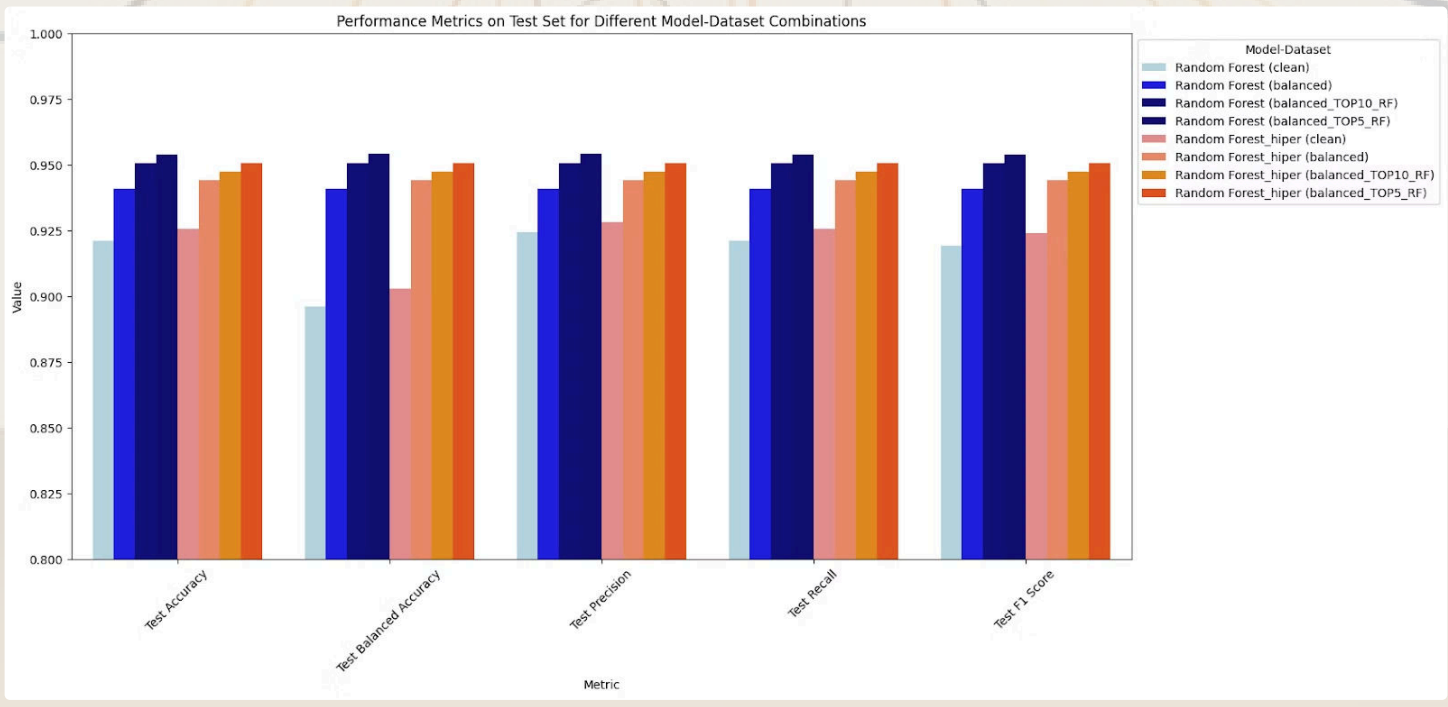
Se realizó una búsqueda de hiperparámetros con **RandomizedSearchCV** (descartamos **GridSearch** por su coste computacional)

El modelo con los mejores hiperparámetros obtenidos se guardan como **'Random_ForestHiper'.**

3 Resultados

El mejor modelo **Random Forest** con dataset ***balanced_TOP5_RF*** logró un Recall y F1 Score de 0.9540 .

Modelo	Dataset	Train Accuracy	Test Accuracy	Train Balanced Accuracy	Test Balanced Accuracy	Train Precision	Test Precision	Train Recall	Test Recall	Train F1 Score	Test F1 Score
Random Forest	clean	1.0000	0.9209	1.0000	0.8962	1.0000	0.9241	1.0000	0.9209	1.0000	0.9192
Random Forest	balanced	1.0000	0.9408	1.0000	0.9407	1.0000	0.9408	1.0000	0.9408	1.0000	0.9408
Random Forest	balanced_ TOP10_RF	0.9992	0.9507	0.9992	0.9507	0.9992	0.9507	0.9992	0.9507	0.9992	0.9507
Random Forest	**balanced_ TOP5_RF**	0.9992	**0.9539**	0.9992	**0.9543**	0.9992	**0.9543**	0.9992	**0.9539**	0.9992	**0.9540**
Random Forest_hiper	clean	0.9843	0.9256	0.9793	0.9027	0.9844	0.9282	0.9843	0.9256	0.9842	0.9241
Random Forest_hiper	balanced	0.9918	0.9441	0.9917	0.9441	0.9918	0.9441	0.9918	0.9441	0.9918	0.9441
Random Forest_hiper	balanced_ TOP10_RF	0.9868	0.9474	0.9868	0.9473	0.9870	0.9474	0.9868	0.9474	0.9868	0.9474
Random Forest_hiper	balanced_ TOP5_RF	0.9753	0.9507	0.9752	0.9507	0.9757	0.9507	0.9753	0.9507	0.9753	0.9507

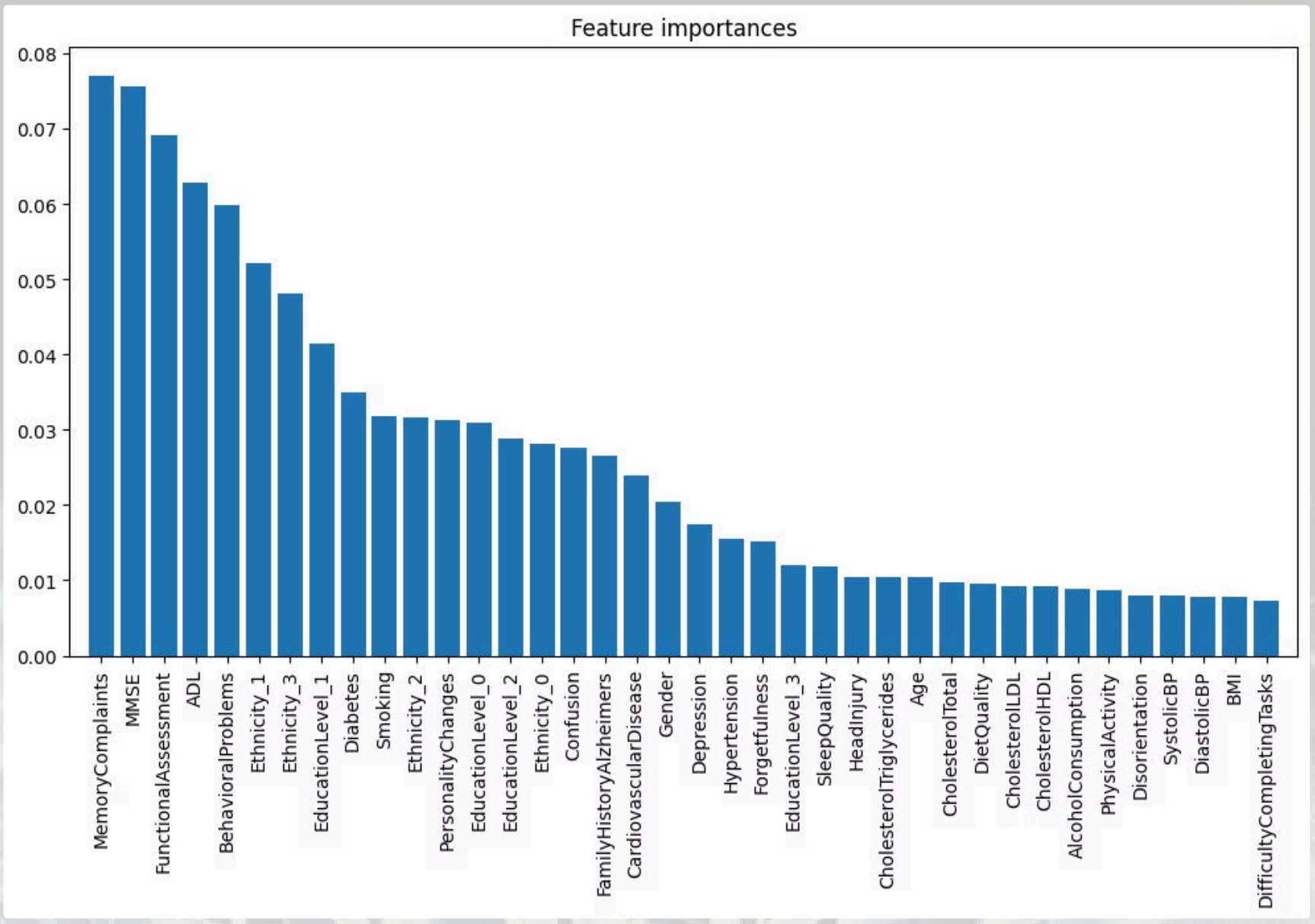


Optimización del Modelo XGBoost

1

Selección de Variables

Se evaluó la importancia de las variables utilizando **XGBClassifier**, generando dos datasets con las 10 y 5 mejores variables.



2

Ajuste de Hiperparámetros

Se utilizó **RandomizedSearchCV** para buscar los mejores hiperparámetros.

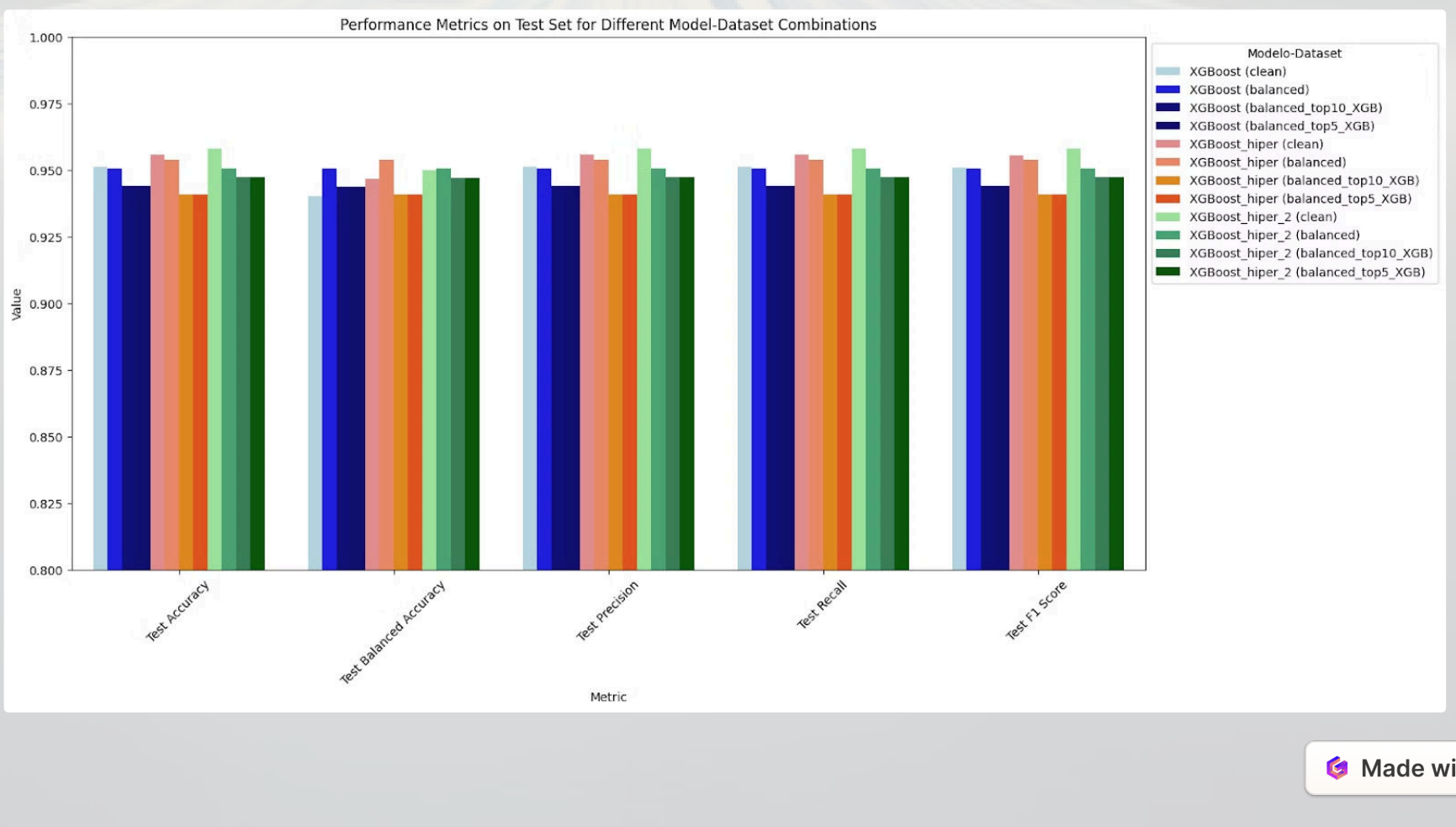
El mejor modelo se guardó como **'XGBoost_hiper'.** Durante las pruebas detectamos unos valores con buenos resultados en el dataset CLEAN y hiperparametrizamos otro modelo como **'XGBoost_hiper2'.**

3

Resultados

El mejor modelo **XGBoost_hiper2** con dataset ***clean*** alcanzó un F1 Score de 0.9579 en el conjunto de prueba.

Modelo	Dataset	Train Accuracy	Test Accuracy	Train Balanced Accuracy	Test Balanced Accuracy	Train Precision	Test Precision	Train Recall	Test Recall	Train F1 Score	Test F1 Score
XGBoost	clean	1.0000	0.9512	1.0000	0.9402	1.0000	0.9514	1.0000	0.9512	1.0000	0.9508
XGBoost	balanced	1.0000	0.9507	1.0000	0.9505	1.0000	0.9507	1.0000	0.9507	1.0000	0.9507
XGBoost	Df_balanced top5_XGB	1.0000	0.9441	1.0000	0.9439	1.0000	0.9441	1.0000	0.9441	1.0000	0.9441
XGBoost	Df_balanced top10_XGB	1.0000	0.9441	1.0000	0.9439	1.0000	0.9441	1.0000	0.9441	1.0000	0.9441
XGBoost_hiper	clean	0.9994	0.9558	0.9996	0.9467	0.9994	0.9559	0.9994	0.9558	0.9994	0.9556
XGBoost_hiper	balanced	0.9992	0.9539	0.9992	**0.9539**	0.9992	0.9539	0.9992	0.9539	0.9992	0.9539
XGBoost_hiper	Df_balanced top10_XGB	0.9762	0.9408	0.9761	0.9409	0.9763	0.9409	0.9762	0.9408	0.9761	0.9408
XGBoost_hiper	Df_balanced top5_XGB	0.9762	0.9408	0.9761	0.9409	0.9763	0.9409	0.9762	0.9408	0.9761	0.9408
XGBoost_hiper_2	**clean**	0.9616	**0.9581**	0.9542	0.9500	0.9616	**0.9582**	0.9616	**0.9581**	0.9615	**0.9579**
XGBoost_hiper_2	balanced	0.9597	0.9507	0.9596	0.9505	0.9599	0.9507	0.9597	0.9507	0.9597	0.9507
XGBoost_hiper_2	Df_balanced top5_XGB	0.9482	0.9474	0.9480	0.9471	0.9487	0.9474	0.9482	0.9474	0.9482	0.9474
XGBoost_hiper_2	Df_balanced top10_XGB	0.9482	0.9474	0.9480	0.9471	0.9487	0.9474	0.9482	0.9474	0.9482	0.9474



Validación y selección Final del Modelo

Realizamos una validación cruzada para ver como se comportan los modelos en otros escenarios de entrenamiento.

De los dos mejores modelos Random Forest (Balanced TOP 5) y XGBoost Hiper 2 (clean).

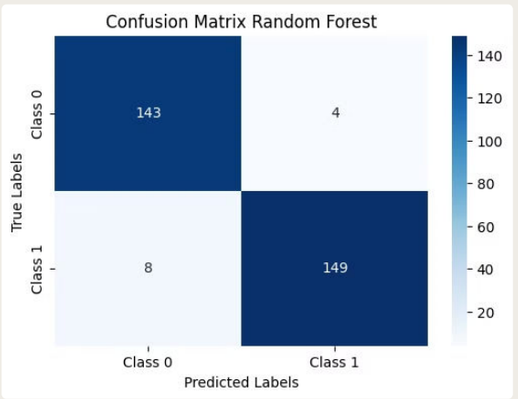
Modelo	Fold	Accuracy	Balanced Accuracy	Precision	Recall	F1 Score
Random Forest (Balanced TOP 5)	1	0.9508	0.9487	0.9512	0.9508	0.9507
	2	0.8934	0.8957	0.9026	0.8934	0.8931
	3	0.9508	0.9518	0.9514	0.9508	0.9509
	4	0.9344	0.9344	0.9344	0.9344	0.9344
	5	0.9344	0.9344	0.9363	0.9344	0.9344
	6	0.9180	0.9180	0.9185	0.9180	0.9180
	7	0.9339	0.9338	0.9339	0.9339	0.9339
	8	0.9669	0.9668	0.9674	0.9669	0.9669
	9	0.9421	0.9379	0.9448	0.9421	0.9418
	10	0.9504	0.9512	0.9510	0.9504	0.9504
	Summary	0.9375	0.9373	0.9392	0.9375	0.9375
	Test	**0.9605**	**0.9609**	**0.9609**	**0.9605**	**0.9605**
XGBoost Hiper 2 (clean)	1	0.9477	0.9360	0.9475	0.9477	0.9475
	2	0.9767	0.9783	0.9772	0.9767	0.9768
	3	0.9477	0.9318	0.9518	0.9477	0.9468
	4	0.9477	0.9474	0.9490	0.9477	0.9480
	5	0.9419	0.9275	0.9470	0.9419	0.9409
	6	0.9593	0.9506	0.9595	0.9593	0.9591
	7	0.9593	0.9484	0.9591	0.9593	0.9592
	8	0.9651	0.9532	0.9657	0.9651	0.9648
	9	0.8895	0.8860	0.8911	0.8895	0.8900
	10	0.9474	0.9395	0.9484	0.9474	0.9470
	Summary	**0.9482**	**0.9399**	**0.9496**	**0.9482**	**0.9480**
	Test	0.9558	0.9467	0.9559	0.9558	0.9556

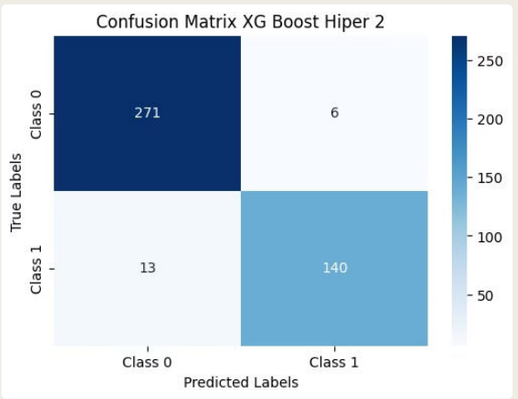
Resultados y Conclusiones

Ambos modelos **Random Forest** y **XGBoost Hiper 2** muestran un rendimiento muy alto en términos de Precisión, Recall, y F1 Score tanto en Cross-Validation como en el conjunto de prueba.

XGBoost Hiper 2 muestra ligeramente mejores resultados en la validación cruzada, pero Random Forest tiene un desempeño superior en el conjunto de prueba.

El **Random Forest** mantiene una precisión y recall más altos en el conjunto de prueba en comparación con XGBoost Hiper 2 y una matriz de confusión mejor.





Futuros Pasos

El modelo desarrollado ofrece una alternativa prometedora.

Los siguientes pasos incluyen la validación adicional del modelo con otros datasets, preferiblemente con menor desbalance.

La optimización de hiperparámetros con **GridSearchCV**.