Alzheimer:

un enfoque desde la Ciencia de Datos

Priscila Rodríguez

Onofre Pouplana

Índice

1. Motivación y descripción del problema

El Alzheimer es una de las principales causas de demencia, afectando sobre todo a las personas mayores y causando un deterioro cognitivo progresivo. Detectarla a tiempo y con precisión es clave para tratarla y obtener mejores resultados para los pacientes.

Los métodos tradicionales de diagnóstico, como las neuroimágenes y el análisis del líquido cefalorraquídeo, suelen ser invasivos, prolongados y caros. Pero gracias a los avances en inteligencia artificial y aprendizaje automático, ahora tenemos alternativas prometedoras que son menos invasivas, más rápidas y económicas.

A medida que evolucionan las herramientas de modelado de datos será cada vez más fácil hacer un diagnóstico prematuro para la detección de la enfermedad.

2. Objetivo inicial deseado

El objetivo del proyecto es construir un modelo predictivo que permita identificar la enfermedad de Alzheimer con alta precisión poniendo énfasis en minimizar los falsos-negativos (FN). Número de casos que la prueba declara negativos y que en realidad son positivos. Por ejemplo, decirle a un paciente que no tiene Alzheimer cuando realmente si lo tiene. Tiene unas consecuencias negativas al no tratar al paciente lo antes posible.

Para ello nos centraremos en optimizar nuestros modelos para conseguir tener un Recall y F1-Score cercanos a 1.

3. Origen de los datos

El conjunto de datos utilizado para esta investigación se obtuvo de la plataforma de código abierto Kaggle. https://www.kaggle.com/datasets/rabieelkharoua/alzheimers-disease-dataset

```
@misc{rabie_el_kharoua_2024,
title={Alzheimer's Disease Dataset},
url={https://www.kaggle.com/dsv/8668279},
DOI={10.34740/KAGGLE/DSV/8668279},
publisher={Kaggle},
author={Rabie El Kharoua},
year={2024}}
```

El conjunto de datos incluye 35 variables que proporcionan información relacionada con la enfermedad tales como: detalles demográficos, factores del estilo de vida, historial médico, mediciones clínicas, evaluaciones cognitivas y funcionales, síntomas y un diagnóstico de la enfermedad de Alzheimer.

3.1 Definición de los datos:

El dataset incluye 2149 muestras con 35 variables:

Identificación paciente:

- **PatientID:** Número de identificación para cada paciente.

Detalles 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).

Factores de 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

4. Limpieza de datos

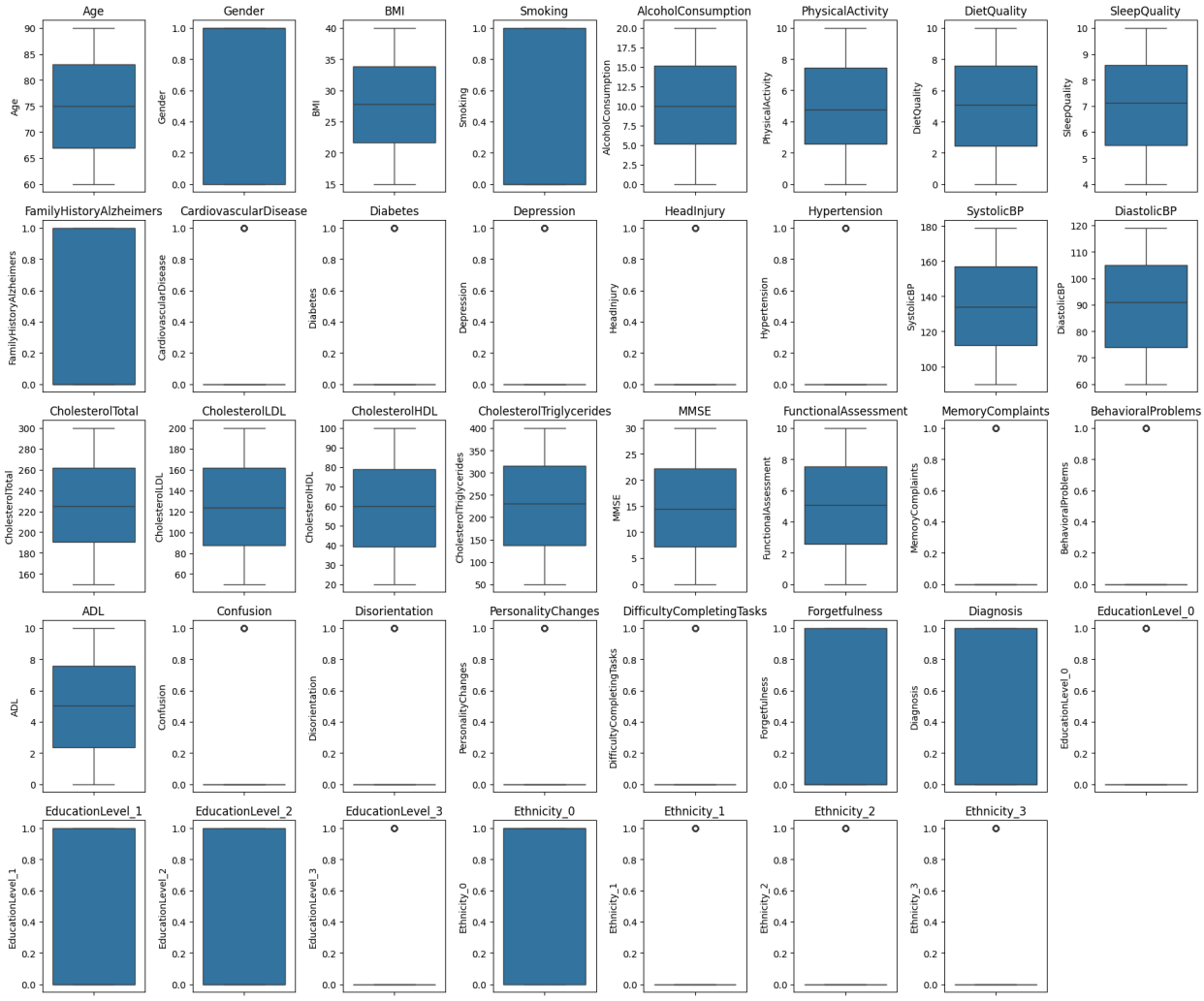
4.1 Duplicados

No se encontraron duplicados en el dataset.

4.2 Valores faltantes (missings)

No se detectaron valores faltantes.

4.3 Valores atípicos (outliers)



Graficando los datos con boxplots no se detectaron outliers significativos.

Ampliamos el cálculo basado en la desviación estándar respecto a la media y el análisis a partir de los cuantiles.

4.4 Descarte de variables no significativas

Se descartaron dos variables sin valor predictivo en una fase temprana:

- 'PatientID'
- 'DoctorInCharge'

Las variables se eliminaron haciendo un **drop**.

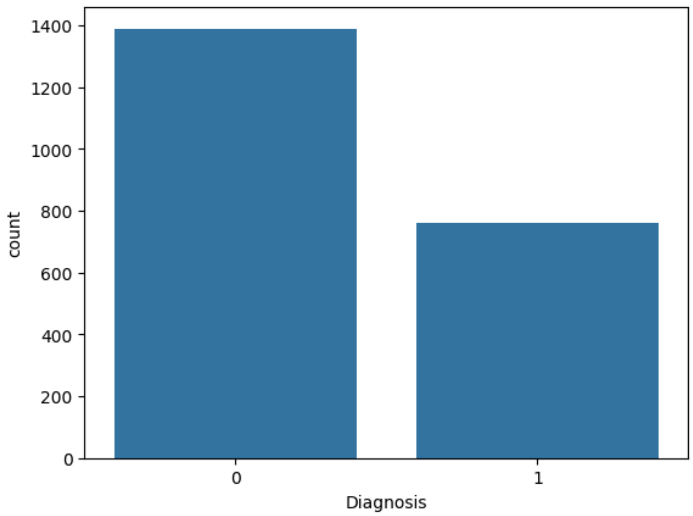
Concluimos que el dataset no requiere de ninguna limpieza específica más.

5. Exploración de datos

5.1 Identificación de variable objetivo

La variable objetivo identificada es **Diagnosis**, que representa si un paciente ha sido diagnosticado con Alzheimer (1 para sí, 0 para no).

Distribución variable objetivo:



La distribución muestra un desbalance entre clases, con más pacientes no diagnosticados que diagnosticados. Para determinar la magnitud del desequilibrio calculamos los siguientes Índices:

- Índice de Desequilibrio : 1.83.
- Índice de Gini : 0.46.

Nos planteamos reajustar el desequilibrio en el dataset en pasos posteriores.

5.2 Clasificación de variables:

Variables numéricas:

- 'PatientID'
- 'Age'
- 'BMI'
- 'AlcoholConsumption'
- 'PhysicalActivity'
- 'DietQuality'
- 'SleepQuality'
- 'SystolicBP'
- 'DiastolicBP',
- 'CholesterolTotal',
- 'CholesterolLDL',
- 'CholesterolHDL'
- 'CholesterolTriglycerides'
- 'MMSE'
- 'FunctionalAssessment'
- 'ADL'

Variables categóricas:

- 'DoctorInCharge'
- ‘Ethnicity’
- ‘EducationLevel’

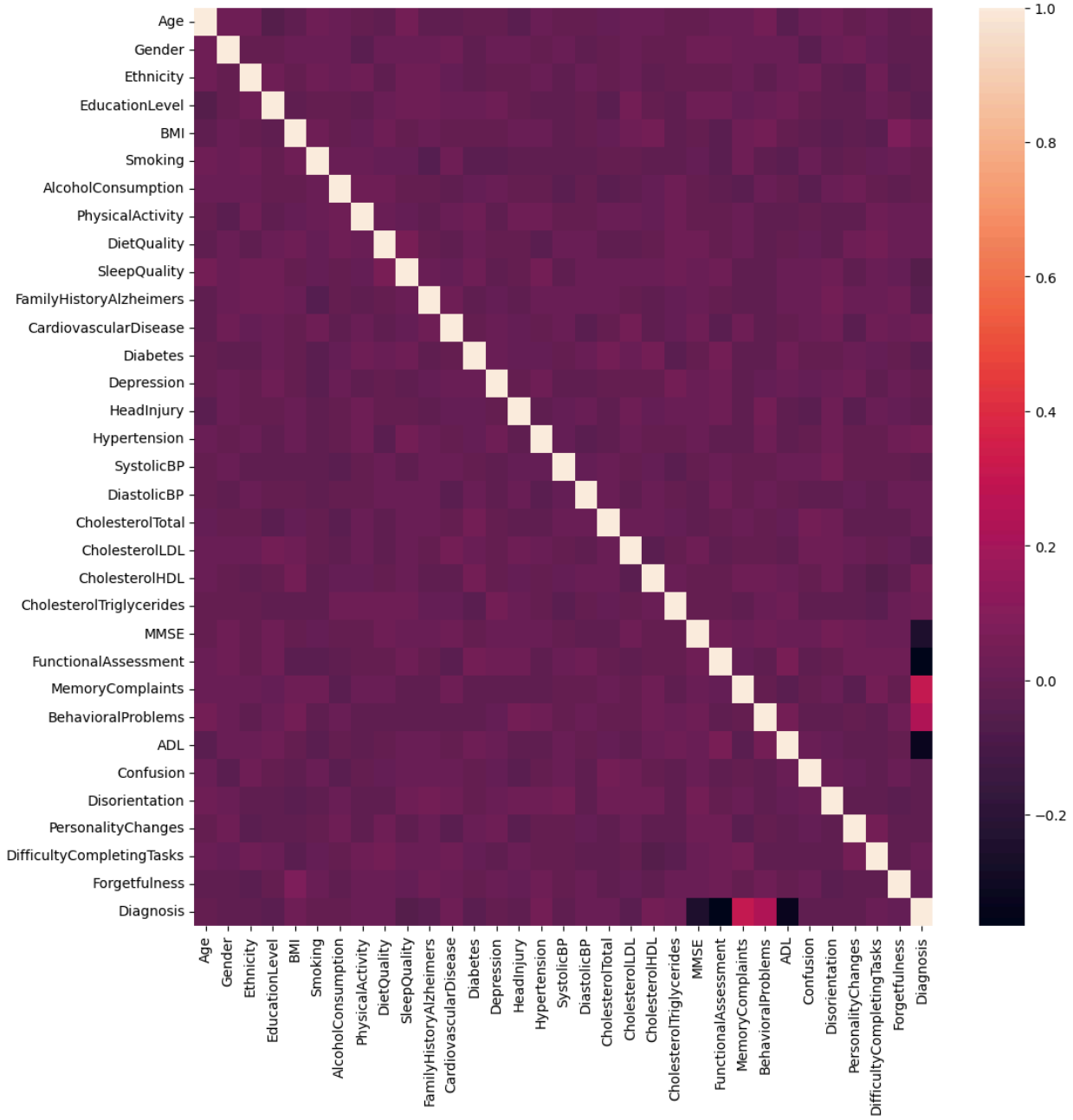
Variables binarias:

- 'Gender',
- 'Smoking'
- 'FamilyHistoryAlzheimers'
- 'CardiovascularDisease'
- 'Diabetes'
- 'Depression'
- 'HeadInjury'
- 'Hypertension'
- 'MemoryComplaints'
- 'BehavioralProblems'
- 'Confusion'
- ’Disorientation'
- 'PersonalityChanges'
- 'DifficultyCompletingTasks'
- 'Forgetfulness'
- 'Diagnosis'

Ejemplo de variables de las 5 primeras muestras:

	0	1	2	3	4
PatientID	4751	4752	4753	4754	4755
Age	73	89	73	74	89
Gender	0	0	0	1	0
Ethnicity	0	0	3	0	0
EducationLevel	2	0	1	1	0
BMI	22.927749	26.827681	17.795882	33.800817	20.716974
Smoking	0	0	0	1	0
AlcoholConsumption	13.297218	4.542524	19.555085	12.209266	18.454356
PhysicalActivity	6.327112	7.619885	7.844988	8.428001	6.310461
DietQuality	1.347214	0.518767	1.826335	7.435604	0.795498
SleepQuality	9.025679	7.151293	9.673574	8.392554	5.597238
FamilyHistoryAlzheimers	0	0	1	0	0
CardiovascularDisease	0	0	0	0	0
Diabetes	1	0	0	0	0
Depression	1	0	0	0	0
HeadInjury	0	0	0	0	0
Hypertension	0	0	0	0	0
SystolicBP	142	115	99	118	94
DiastolicBP	72	64	116	115	117
CholesterolTotal	242.36684	231.162595	284.181858	159.58224	237.602184
CholesterolLDL	56.150897	193.407996	153.322762	65.366637	92.8697
CholesterolHDL	33.682563	79.028477	69.772292	68.457491	56.874305
CholesterolTriglycerides	162.189143	294.630909	83.638324	277.577358	291.19878
MMSE	21.463532	20.613267	7.356249	13.991127	13.517609
FunctionalAssessment	6.518877	7.118696	5.895077	8.965106	6.045039
MemoryComplaints	0	0	0	0	0
BehavioralProblems	0	0	0	1	0
ADL	1.725883	2.592424	7.119548	6.481226	0.014691
Confusion	0	0	0	0	0
Disorientation	0	0	1	0	0
PersonalityChanges	0	0	0	0	1
DifficultyCompletingTasks	1	0	1	0	1
Forgetfulness	0	1	0	0	0
Diagnosis	0	0	0	0	0
DoctorInCharge	XXXConfid	XXXConfid	XXXConfid	XXXConfid	XXXConfid

5.3 Matriz de correlación

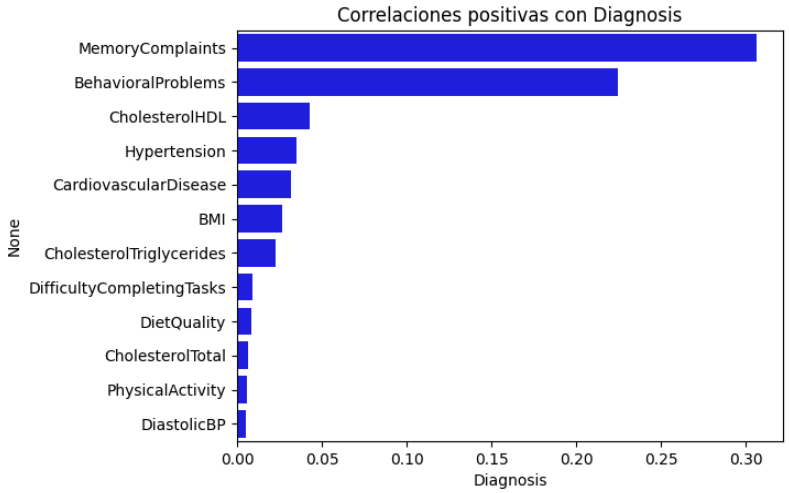


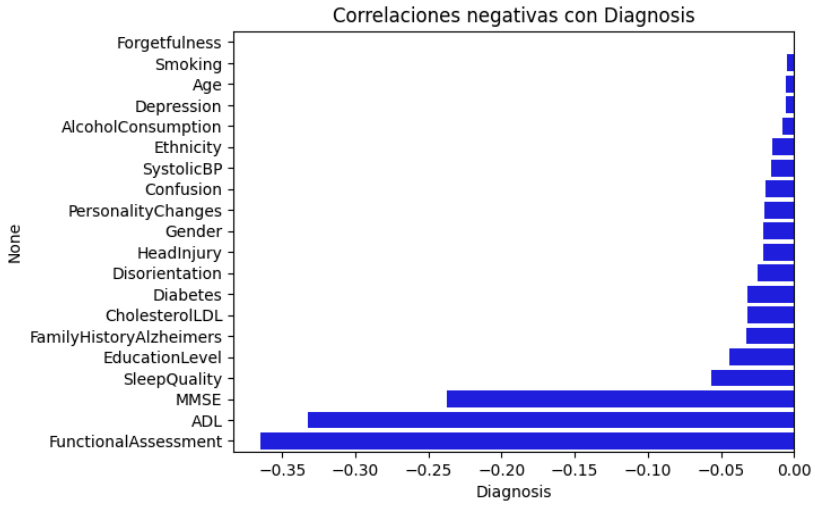
La matriz de correlación ayudó a identificar la relación entre variables numéricas, destacando algunas correlaciones fuertes.

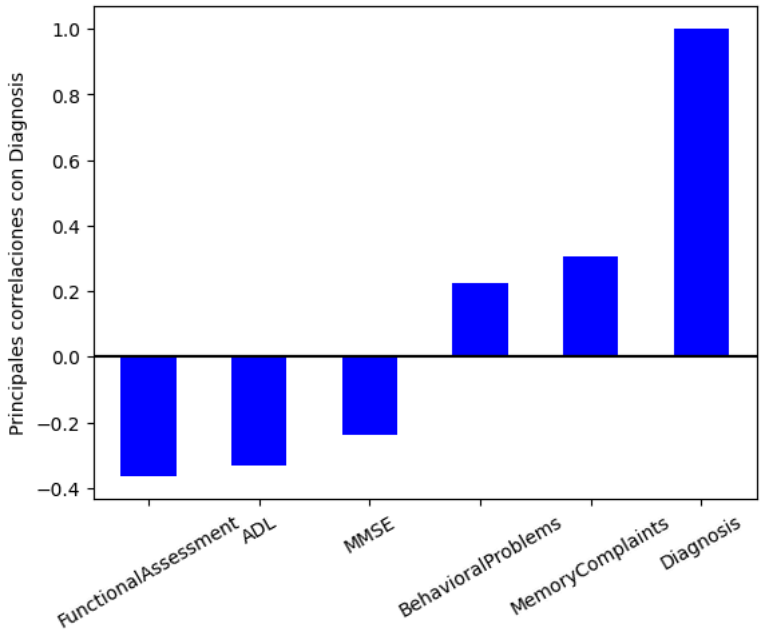
Las áreas más claras y más oscuras fuera de la diagonal indican correlaciones fuertes entre variables específicas. Algunos detalles significativos:

- Parece que hay una correlación significativa entre diferentes tipos de colesterol (Total, LDL, HDL, Triglycerides), lo cual es lógico debido a la naturaleza de estas mediciones.
- La variable ‘Diagnosis’ parece tener una relación interesante con algunas variables como ‘MemoryComplaints’, ‘Confusion’, ‘Disorientation’, ‘PersonalityChanges’, ‘DifficultyCompletingTasks’, y ‘Forgetfulness’. Esto sugiere que estos síntomas pueden ser predictores importantes para el diagnóstico de la enfermedad de Alzheimer.
- ‘MMSE’ y ‘FunctionalAssessment’ podrían estar correlacionados con la variable ‘Diagnosis’, indicando que estas evaluaciones son cruciales para identificar la presencia de la enfermedad.

5.4 Análisis de Correlación de variables predictoras VS variable objetivo



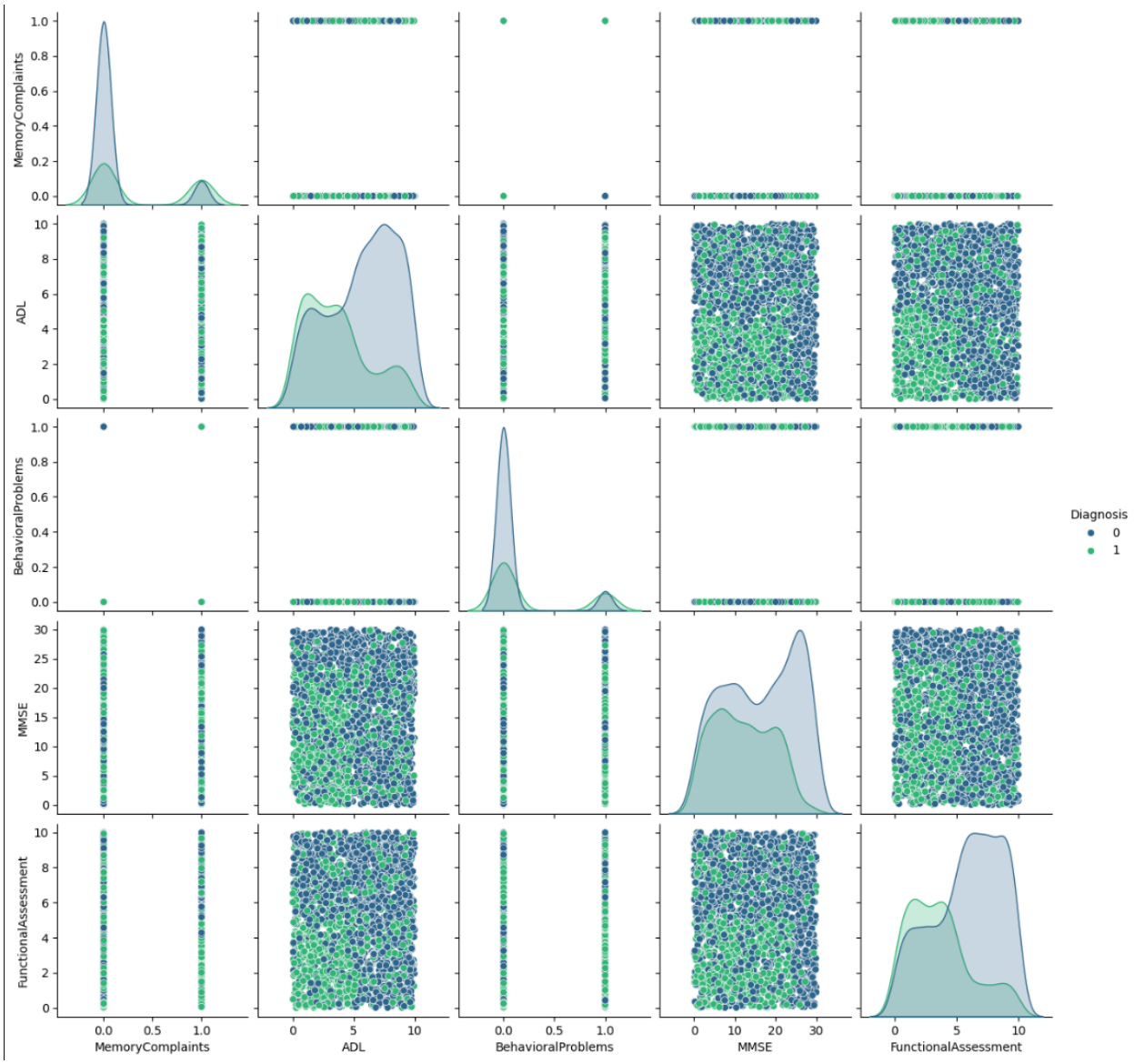




Identificamos las variables con mayor correlación como las mejores predictoras:

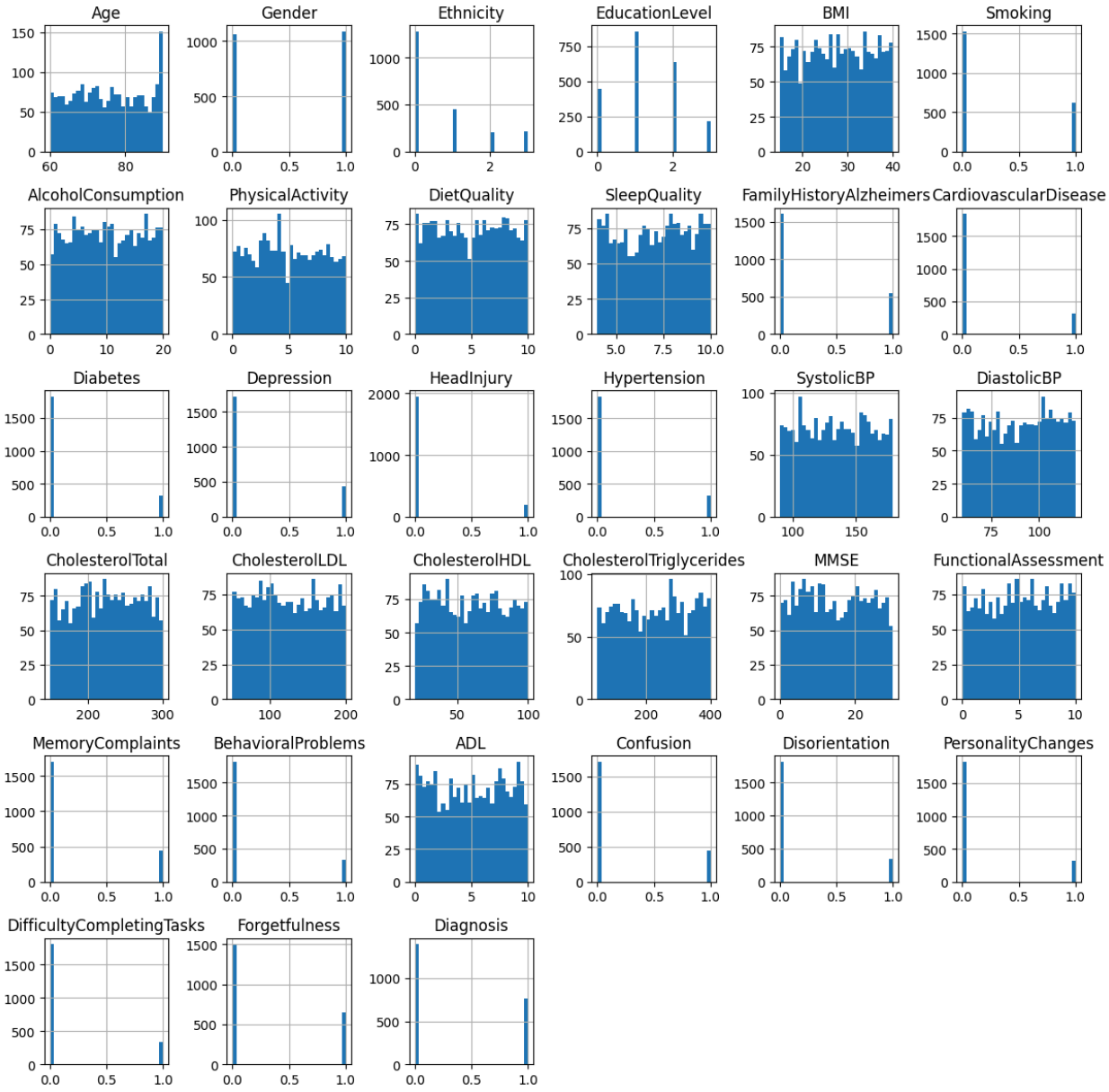
- ‘Functional Assessment’
- ‘Memory Complaints’
- ‘ADL’
- ‘Behavioral Problems’
- ‘MMSE’

5.5 Pair Plot de las variables principales



El pair plot confirma visualmente que las variables detectadas tiene patrones de distribución distintos a la distribución de la variable objetivo.

5.6 Análisis distribución de valores



Los histogramas mostraron las distribuciones de variables con bastante uniformidad, aunque la variable **Age** no tenía una distribución normal, no se ajustó ya que una distribución asimétrica negativa es esperable en ese tipo de pacientes.

5.7 Valores estadísticos

	count	mean	std	min	25%	50%	75%	max
PatientID	2149.0	5825.000000	620.507185	4751.000000	5288.000000	5825.000000	6362.000000	6899.000000
Age	2149.0	74.908795	8.990221	60.000000	67.000000	75.000000	83.000000	90.000000
Gender	2149.0	0.506282	0.500077	0.000000	0.000000	1.000000	1.000000	1.000000
Ethnicity	2149.0	0.697534	0.996128	0.000000	0.000000	0.000000	1.000000	3.000000
EducationLevel	2149.0	1.286645	0.904527	0.000000	1.000000	1.000000	2.000000	3.000000
BMI	2149.0	27.655697	7.217438	15.008851	21.611408	27.823924	33.869778	39.992767
Smoking	2149.0	0.288506	0.453173	0.000000	0.000000	0.000000	1.000000	1.000000
AlcoholConsumption	2149.0	10.039442	5.757910	0.002003	5.139810	9.934412	15.157931	19.989293
PhysicalActivity	2149.0	4.920202	2.857191	0.003616	2.570626	4.766424	7.427899	9.987429
DietQuality	2149.0	4.993138	2.909055	0.009385	2.458455	5.076087	7.558625	9.998346
SleepQuality	2149.0	7.051081	1.763573	4.002629	5.482997	7.115646	8.562521	9.999840
FamilyHistoryAlzheimers	2149.0	0.252210	0.434382	0.000000	0.000000	0.000000	1.000000	1.000000
CardiovascularDisease	2149.0	0.144253	0.351428	0.000000	0.000000	0.000000	0.000000	1.000000
Diabetes	2149.0	0.150768	0.357906	0.000000	0.000000	0.000000	0.000000	1.000000
Depression	2149.0	0.200558	0.400511	0.000000	0.000000	0.000000	0.000000	1.000000
HeadInjury	2149.0	0.092601	0.289940	0.000000	0.000000	0.000000	0.000000	1.000000
Hypertension	2149.0	0.148906	0.356079	0.000000	0.000000	0.000000	0.000000	1.000000
SystolicBP	2149.0	134.264774	25.949352	90.000000	112.000000	134.000000	157.000000	179.000000
DiastolicBP	2149.0	89.847836	17.592496	60.000000	74.000000	91.000000	105.000000	119.000000
CholesterolTotal	2149.0	225.197519	42.542233	150.093316	190.252963	225.086430	262.031657	299.993352
CholesterolLDL	2149.0	124.335944	43.366584	50.230707	87.195798	123.342593	161.733733	199.965665
CholesterolHDL	2149.0	59.463533	23.139174	20.003434	39.095698	59.768237	78.939050	99.980324
CholesterolTriglycerides	2149.0	228.281496	101.986721	50.407194	137.583222	230.301983	314.839046	399.941862
MMSE	2149.0	14.755132	8.613151	0.005312	7.167602	14.441660	22.161028	29.991381
FunctionalAssessment	2149.0	5.080055	2.892743	0.000460	2.566281	5.094439	7.546981	9.996467
MemoryComplaints	2149.0	0.208004	0.405974	0.000000	0.000000	0.000000	0.000000	1.000000
BehavioralProblems	2149.0	0.156817	0.363713	0.000000	0.000000	0.000000	0.000000	1.000000
ADL	2149.0	4.982958	2.949775	0.001288	2.342836	5.038973	7.581490	9.999747
Confusion	2149.0	0.205212	0.403950	0.000000	0.000000	0.000000	0.000000	1.000000
Disorientation	2149.0	0.158213	0.365026	0.000000	0.000000	0.000000	0.000000	1.000000
PersonalityChanges	2149.0	0.150768	0.357906	0.000000	0.000000	0.000000	0.000000	1.000000
DifficultyCompletingTasks	2149.0	0.158678	0.365461	0.000000	0.000000	0.000000	0.000000	1.000000
Forgetfulness	2149.0	0.301536	0.459032	0.000000	0.000000	0.000000	1.000000	1.000000
Diagnosis	2149.0	0.353653	0.478214	0.000000	0.000000	0.000000	1.000000	1.000000

Se calcularon estadísticas como media, mediana y desviación estándar para cada variable, proporcionando un resumen del dataset.

6. Feature Engineering

6.1 Transformación de variables categóricas (binarización)

‘Ethnicity’ :

- 'Ethnicity_0'
- 'Ethnicity_1'
- 'Ethnicity_2'
- 'Ethnicity_3'

‘**EducationLevel’:**

- 'EducationLevel_0'
- 'EducationLevel_1'
- 'EducationLevel_2'
- 'EducationLevel_3'

Las variables categóricas se transformaron en binarias utilizando **`get_dummies`.**

Persistimos el dataset como Clean_dataset.

6.2 Generación de Dataset Balanceado

Para poder compensar el desbalanceo en la variable principal, probamos dos estrategias :

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

El resultado final se persistió como Balanced_dataset y Smote_dataset respectivamente.

Undersampling:

```
from sklearn.utils import resample

df_0 = df_2[df_2['Diagnosis'] == 0]
df_1 = df_2[df_2['Diagnosis'] == 1]
df_0_downsampled = resample(df_0,
                            replace=False,
                            n_samples=len(df_1),
                            random_state=42)
df_balanced = pd.concat([df_0_downsampled, df_1]
```

7. Algoritmos ML y entrenamiento

7.1 Segmentación train/test:

```
X = df_balanced.drop(['Diagnosis'], axis=1)  # Variables predictoras
y = df_balanced['Diagnosis']                 # Variable objetivo

# Dividir el DataFrame en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Para cada uno de nuestros tres dataset seleccionamos un 80% de las muestras para entrenamiento (Train) y un 20% para prueba (Test), y escalamos con StandardScaler.

7.2 Evaluación de Modelos de Clasificación

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

Se probaron varios modelos, incluyendo KNN, SVM y árboles de decisión, para determinar el mejor rendimiento con los distintos datasets.

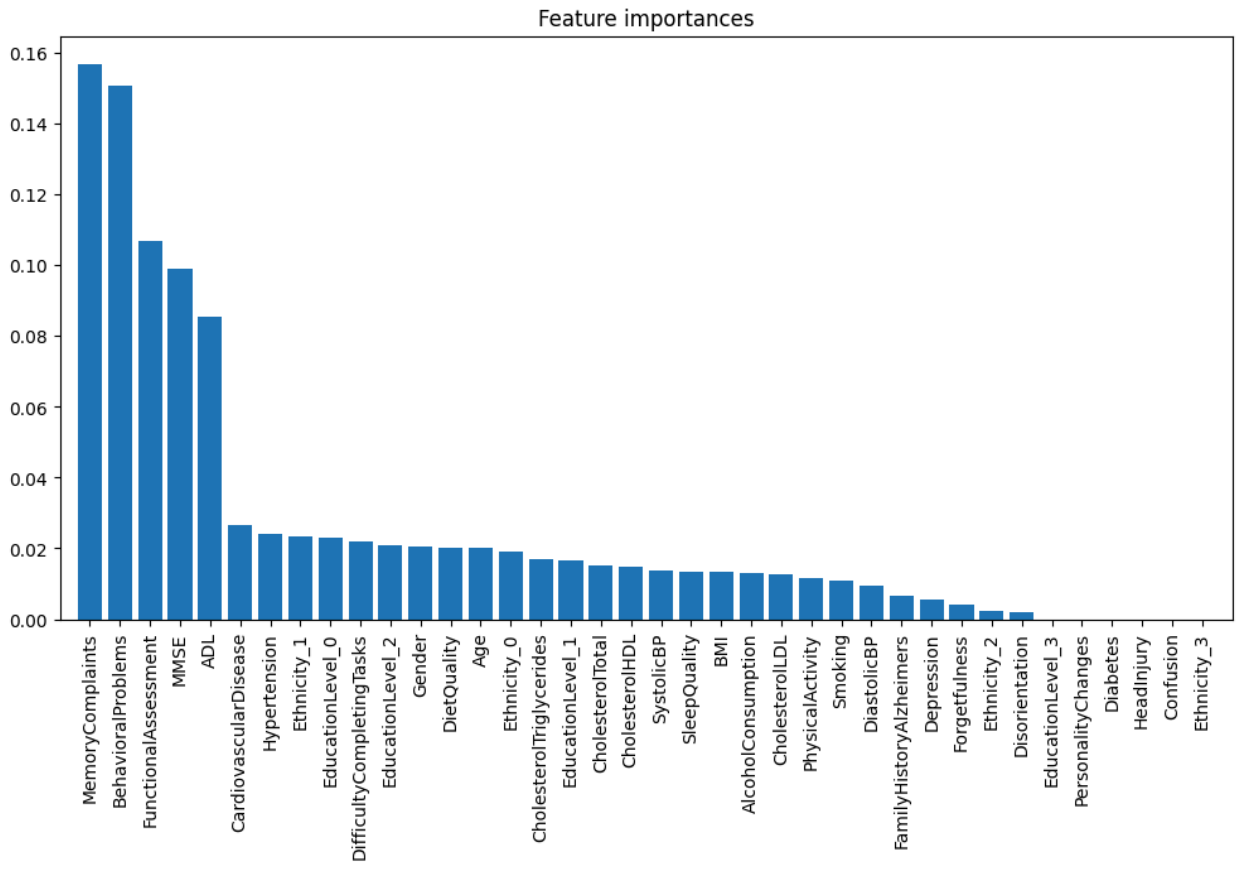
Conclusión:

- XGBoost y Random Forest son los modelos más consistentes y con mejor rendimiento, especialmente en el dataset Balanceado.
- Por ello, seguiremos profundizando en estos dos modelos para intentar mejorar su rendimiento a través de la selección de variables y el ajuste de sus hiperparámetros.

7.3 Optimización Modelo Random Forest

7.3.1 Selección de Variables por Importancia

Se evaluó la importancia de las variables utilizando BalancedRandomForestClassifier sobre el dataset Balanced:



A partir de los resultados, generamos dos datasets adicionales que contienen únicamente las 10 y 5 mejores características del dataset (balanced_top10_RF y balanced_top5_RF)

7.3.2 Ajuste de Hiperparametros

Busamos los mejores hiperparametros del modelo Random Forest usando su mejor dataset (TOP5) con la siguiente configuración:

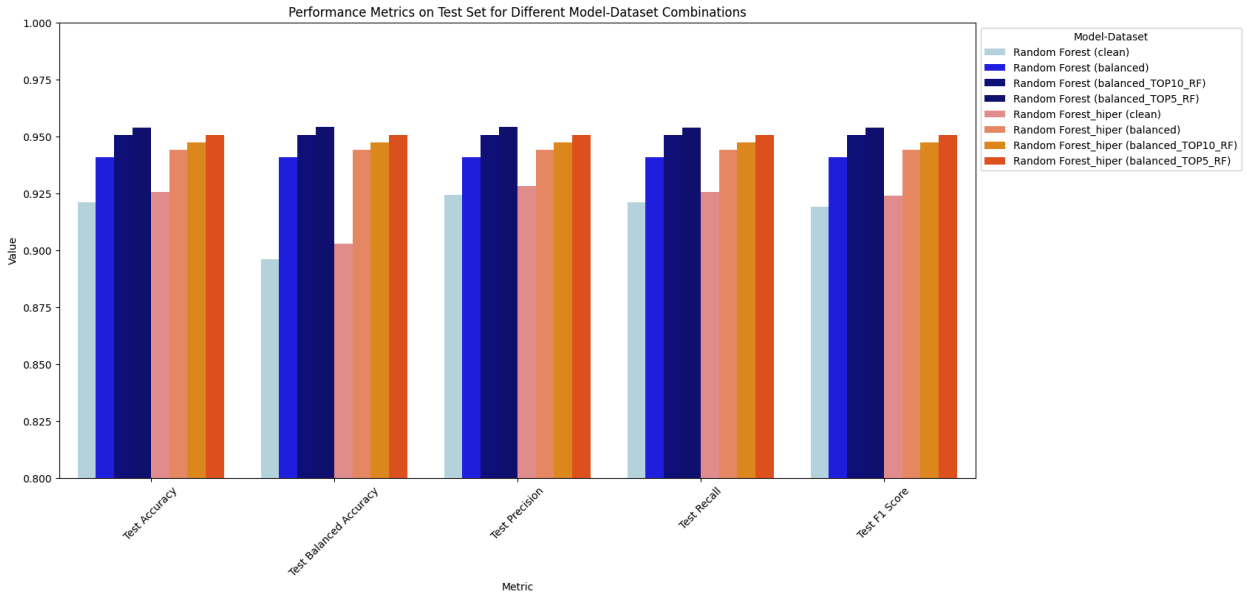
```
param_dist = {
    'n_estimators': [50, 100, 200, 300, 500],
    'max_depth': [None, 10, 20, 30, 40, 50],
    'min_samples_split': [2, 3],
    'min_samples_leaf': [1, 2, 3, 4, 5],
    'max_features': ['sqrt', 'log2'],
    'bootstrap': [True, False],
    'criterion': ['gini', 'entropy']
}

# Crear el objeto RandomForestClassifier
rf_clf = RandomForestClassifier(random_state=42)

# Crear el objeto RandomizedSearchCV
random_search = RandomizedSearchCV(estimator=rf_clf,
param_distributions=param_dist,
                                   n_iter=300, scoring='recall', cv=5,
                                   n_jobs=-1, verbose=1, random_state=42)
```

Los resultados obtenidos se guardan como Random Forest Hiper:

```
'Random Forest_hiper': RandomForestClassifier(bootstrap=False, max_depth=10,
max_features="log2", min_samples_leaf=2, min_samples_split=2, n_estimators=500,
criterion='gini', random_state=42)
```

7.3.3 Resultados

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



Conclusión:

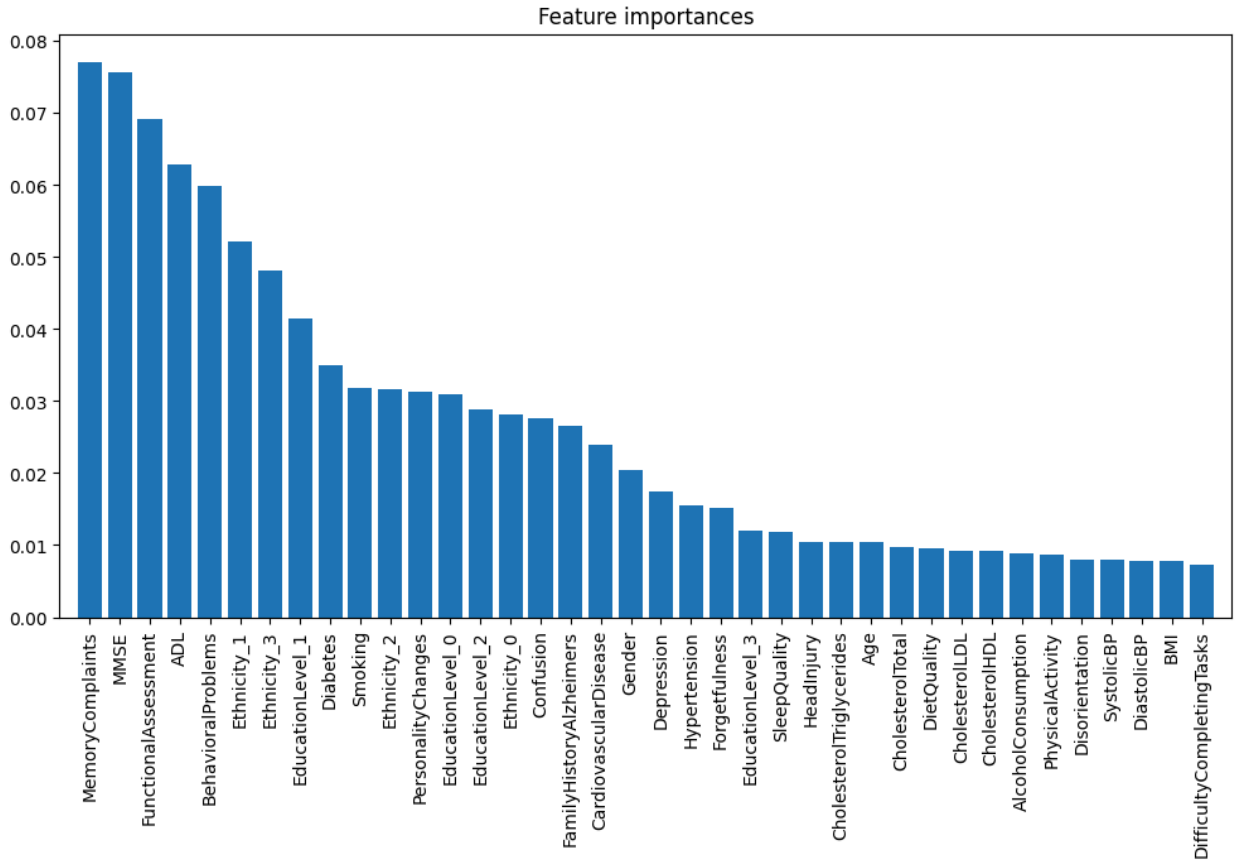
Mejor Modelo: **Random Forest (`balanced_TOP5_RF`)**

- Ofrece el mejor rendimiento en todas las métricas en el conjunto de prueba, mostrando las mejores precisiones, recalls y F1 Scores.
- El rendimiento en el conjunto de entrenamiento sigue siendo muy alto, lo que sugiere que el modelo está bien ajustado y generaliza eficazmente a nuevos datos.

7.4 Optimización Modelo XGBoost

7.4.1 Selección de Variables por Importancia

Usamos el XGBClassifier con el Balanced dataset para identificar las TOP10 y TOP5 variables



A partir de los resultados, generamos dos datasets adicionales que contienen únicamente las 10 y 5 mejores características del dataset (balanced_top10_XGB y balanced_top5_XGB)

7.4.2 Ajuste de hiperparámetros

Se realizó una búsqueda de hiperparámetros para optimizar el modelo con RandomizedSearchCV sobre el dataset balanced con la siguiente configuración:

```
param_dist = {

    'n_estimators': [20, 30, 50, 100],

    'learning_rate': [0.01, 0.02, 0.05, 0.1],

    'max_depth': [18, 20, 30, 50],

    'subsample': [0.7, 0.8, 0.9, 1.0],

    'colsample_bytree': [0.7, 0.8, 0.9, 1.0],

    'gamma': [0.1, 0.2, 0.3, 0.4]}

xgb_clf = xgb.XGBClassifier(random_state=42)

random_search = RandomizedSearchCV(estimator=xgb_clf, param_distributions=param_dist,

                                   n_iter=300, scoring='recall', cv=5,

                                   n_jobs=-1, verbose=1, random_state=42)
```

Persistimos el modelo como XGBoost hiper:

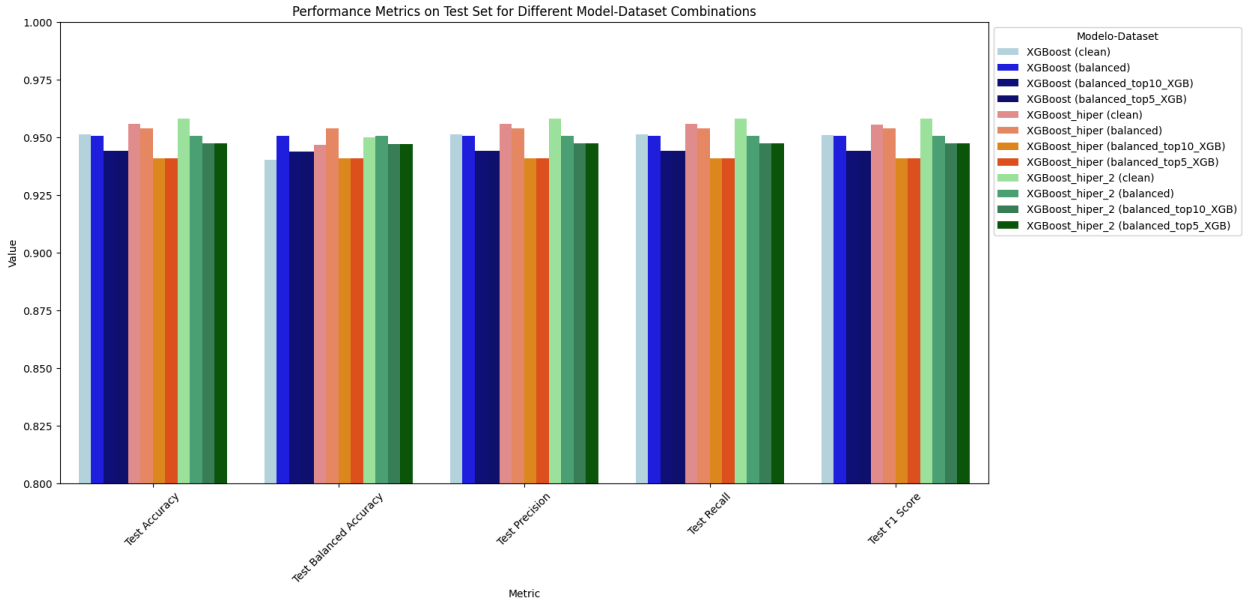
```
'XGBoost_hiper': XGBClassifier(
subsample=1.0, n_estimators=50, max_depth=18, learning_rate=0.1,
gamma=0.1,colsample_bytree=0.9, random_state=42)
```

Tras probar de nuevo el modelo con todos los datasets detectamos que el modelo en el dataset clean se comporta de forma significativamente mejor, por lo que generamos otra busqueda de hiperparametros sobre el dataset clean, y persistimos los resultados como XGBoost_hiper_2 con los siguientes parametros:

```
'XGBoost_hiper_2': XGBClassifier(
subsample=0.8, n_estimators=30, max_depth=30, learning_rate=0.05, gamma=0.1,
 colsample_bytree=1.0, random_state=42)
```

7.4.3 Resultados

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



Conclusión:

Mejor Modelo: **`XGBoost_hiper_2 (clean)`**

- Las métricas de prueba son ligeramente superiores y muestran mejor generalización.
- Las métricas de entrenamiento y prueba están más equilibradas, indicando un menor sobreajuste.
- Trabajar con datos clean evita la complejidad adicional del balanceo de datos que no muestra una mejora significativa en el rendimiento.

8. Selección de modelo

Comparamos los resultados de los dos modelos y seleccionamos los dos mejores para una validación final cruzada.

8.1 Comparativa final

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
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_top10_XGB	1.0000	0.9441	1.0000	0.9439	1.0000	0.9441	1.0000	0.9441	1.0000	0.9441
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_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_top10_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_top5_XGB	0.9482	0.9474	0.9480	0.9471	0.9487	0.9474	0.9482	0.9474	0.9482	0.9474

8.2 Validación cruzada

Realizamos una validación cruzada de los dos mejores modelos con 10 folds, cada uno con su respectivo dataset:

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

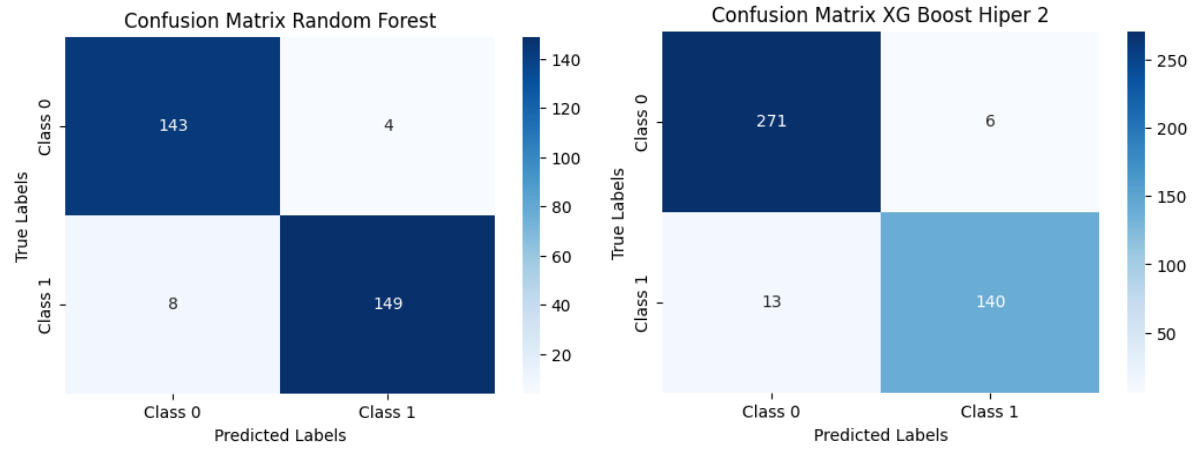
9. Conclusiones

9.1 Modelo seleccionado

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.



9.2 Próximos pasos

El modelo seleccionado muestra un grado de precisión y recall notable. Los siguientes pasos incluyen la validación adicional del modelo con otros datasets, preferiblemente con menor desbalance y la optimización de hiperparametros con GridSearchCV.