Detecció de TRS amb tècniques d'aprenentatge automàtic

Ferran Òdena

GIA- UPC

Introducció a l'Aprenentatge Automàtic

Desembre 2025

0. Taula de continguts

1. Introducció

Aquest informe documenta el desenvolupament d'una solució basada en intel·ligència artificial per abordar un repte clínic significatiu: la identificació de pacients amb esquizofrènia que presenten resistència al tractament (TRS). Utilitzant un conjunt de dades clíniques i genètiques proporcionat pel projecte de l'assignatura, l'estudi s'inicia amb una fase exhaustiva d'anàlisi exploratòria per comprendre l'estructura de la informació, on s'han examinat la distribució de les variables, la presència de valors perduts i l'impacte dels outliers, a més d'abordar el desbalanceig inherent de la classe objectiu que caracteritza aquest tipus de dades mèdiques.

La fase central del treball es focalitza en l'experimentació i ajustament de diversos models predictius, incloent-hi Màquines de Vectors de Suport (SVM), XGBoost, Regressió Logística personalitzada i l'Explainable Boosting Machine (EBM). Aquest procés ha estat guiat per una avaluació rigorosa que prioritza la sensibilitat (Recall) per sobre de la precisió global, amb l'objectiu mèdic de minimitzar els casos de pacients resistents no detectats. Finalment, l'informe conclou amb la selecció del model EBM com a eina final per la seva capacitat d'oferir un equilibri òptim entre rendiment predictiu i explicabilitat clínica, decisió que es formalitza mitjançant la generació i exportació de la seva corresponent Model Card per estandarditzar-ne la documentació.

2. Informació del problema

Les dades contenen informació clínica i genètica de pacients diagnosticats amb esquizofrènia, amb l'objectiu de predir quins d'aquests pacients desenvoluparan resistència al tractament (TRS). La resistència al tractament es defineix com la manca de resposta adequada als tractaments antipsicòtics estàndard, i afecta aproximadament un 31.5% dels pacients amb esquizofrènia segons els estudis citats [KHO25].

Per tant, l'objectiu principal d'aquest projecte és desenvolupar un model predictiu capaç d'identificar amb precisió els pacients amb TRS utilitzant les dades disponibles. Aquest model ha de ser capaç de manejar el desbalanceig inherent en les dades, ja que la proporció de pacients amb TRS és significativament menor que la dels pacients sense TRS. Així mateix, es pretén que el model sigui interpretable per facilitar la seva aplicació clínica i la comprensió dels factors que contribueixen a la resistència al tractament.

3. Anàlisi exploratori descriptiu

En aquesta secció es presenta l'anàlisi exploratori descriptiu (AED) realitzat sobre el conjunt de dades proporcionat per al problema de predicció clínica. L'objectiu principal d'aquesta anàlisi és comprendre millor les característiques de les dades, identificar possibles problemes com valors perduts o *outliers*, i preparar les dades per a l'ajustament dels models predictius.

3.1 Anàlisi de les variables

Observem primer de tot la taula de dades per tenir una visió general de les variables disponibles i la seva tipologia. L'obtenim amb la comanda `df.describe().T`, que ens proporciona estadístiques descriptives per a les variables numèriques i categòriques. El resultat ha estat:

Variable	N	Mean (SD)	Min	25%	Median	75%	Max	Missing (%)
Age	9000	26.04 (10.01)	13.00	19.00	25.00	31.00	64.00	0.00%
Sex (0=F, 1=M)	9000	0.58 (0.49)	0.00	0.00	1.00	1.00	1.00	0.00%
BMI	9000	28.11 (5.43)	15.00	24.40	28.00	31.70	49.60	0.00%
Duration Untreated Psychosis	8872	19.22 (19.55)	0.30	6.40	12.50	24.30	125.00	1.42%
Family History	9000	0.12 (0.32)	0.00	0.00	0.00	0.00	1.00	0.00%
Initial Response	9000	41.84 (30.16)	0.00	10.10	38.20	72.30	100.00	0.00%
Prior Antipsychotics	9000	0.41 (0.67)	0.00	0.00	0.00	1.00	2.00	0.00%
TRS (Target)	9000	0.32 (0.46)	0.00	0.00	0.00	1.00	1.00	0.00%
Lymphocyte count	7009	1.80 (0.60)	0.50	1.38	1.80	2.20	4.02	22.12%
Neutrophil count	7015	5.01 (1.47)	1.50	4.01	5.02	6.01	9.96	22.06%
Triglycerides	6547	152.01 (61.10)	40.00	108.05	151.10	194.60	394.60	27.26%
Glucose	6381	95.86 (18.31)	65.00	82.20	95.50	108.30	159.60	29.10%
Alkaline Phosphatase	6062	85.17 (24.83)	30.00	68.20	84.70	101.90	179.30	32.64%
IL-17A	8999	2.66 (0.80)	-0.20	2.12	2.65	3.21	5.38	0.01%
CCL23	9000	3.78 (1.05)	-0.20	3.08	3.78	4.49	7.69	0.00%
TWEAK	9000	4.19 (1.24)	-0.54	3.37	4.19	5.04	8.92	0.00%
HLA-DRB1*04:02	9000	0.02 (0.15)	0.00	0.00	0.00	0.00	1.00	0.00%
HLA-B*15:02	9000	0.03 (0.18)	0.00	0.00	0.00	0.00	1.00	0.00%
HLA-A*31:01	9000	0.05 (0.21)	0.00	0.00	0.00	0.00	1.00	0.00%
Polygenic Risk Score	8999	0.030 (0.14)	-0.44	-0.07	0.02	0.11	0.58	0.01%
Del 22q11.2	9000	0.009 (0.09)	0.00	0.00	0.00	0.00	1.00	0.00%
Ki Whole Striatum	9000	0.0130 (0.002)	0.0080	0.0113	0.0129	0.0145	0.0200	0.00%
Ki Associative Striatum	9000	0.0130 (0.002)	0.0071	0.0113	0.0128	0.0146	0.0210	0.00%
SUVRc Whole Striatum	9000	1.18 (0.27)	0.80	0.97	1.16	1.36	2.00	0.00%
SUVRc Assoc. Striatum	9000	1.18 (0.27)	0.80	0.96	1.16	1.37	2.00	0.00%

Taula 1: Resum estadístic de totes les variables numèriques i categòriques del conjunt d'entrenament. Es mostra el recompte (N), mitjana i desviació estàndard, quartils i percentatge de valors perduts.

Obeservem que el conjunt de dades conté un total de 9000 mostres i diverses variables predictives, així com la variable objectiu `TRS (Target)`. Algunes variables presenten valors perduts, especialment les mèdiques com el recompte de limfòcits i neutròfils, triglicèrids, glucosa i fosfatasa alcalina, que gestionarem més endavant a la secció 3.3.

Pel que fa les distribucions de les variables, la majoria semblen tenir una distribució aproximadament normal, o de combinacions de normals, com per exemple:

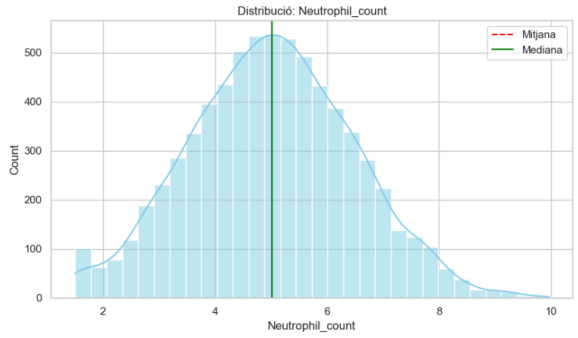

Figura 1: Distribució de `Neutrophil count`.

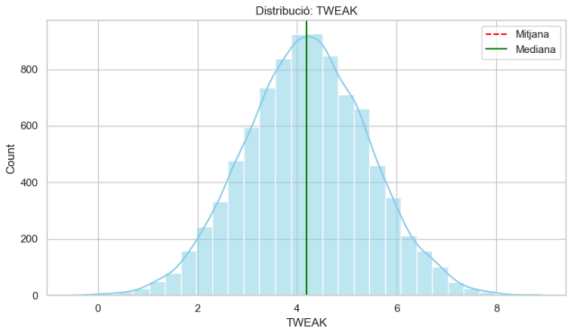

Figura 2: Distribució de `TWEAK`.

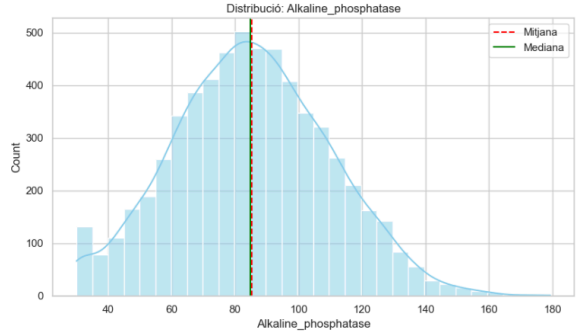

Figura 3: Distribució de `Alkaline phosphatase`

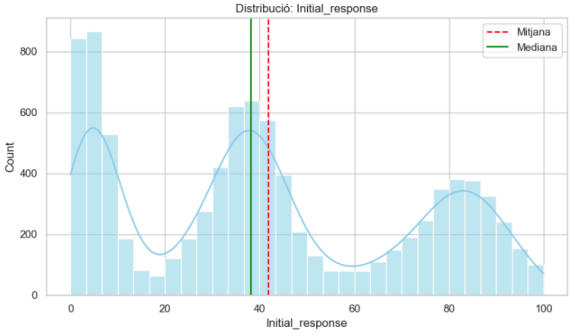

Figura 4: Distribució de `Initial response`.

Aquestes variables es poden utilitzar directament en els models predictius, ja que no requereixen transformacions addicionals per a la seva normalització. No obstant això, algunes variables com el `Duration_Untreated_Psychosis` o `Age` mostren una distribució més asimètrica:

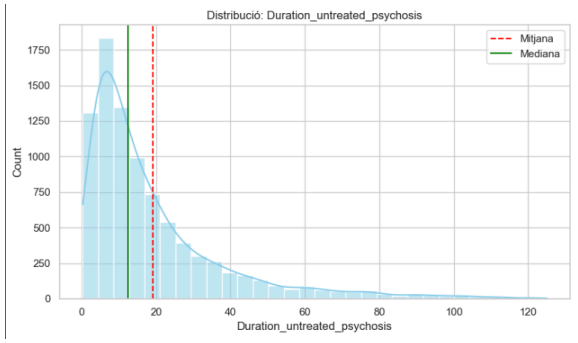

Figura 5: Distribució de `Duration_Untreated_Psychosis`.

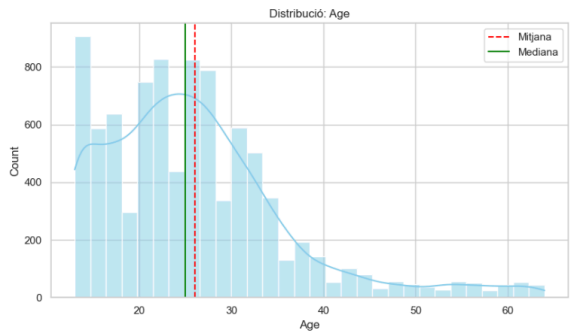

Figura 6: Distribució de `Age`.

En aquest cas podem veure unes cues llargues cap a la dreta, indicant la presència d'alguns valors extrems. Comprovem ràpidament amb el skewness que aquestes variables no són normals. Si tenen un skewness major a 1 o menor a -1, es consideren altament asimètriques. Comprovem:

```
ANÀLISI D'ASIMETRIA (SKEWNESS):
--------------------------------------------------------------------------------
                  Variable  Skewness  Kurtosis                                Estat
Duration_untreated_psychosis  2.150479  5.323006 🔴 MOLT ASIMÈTRICA (Requereix Log/BoxCox)
                       Age  1.258793  2.065980 🔴 MOLT ASIMÈTRICA (Requereix Log/BoxCox)
  SUVRc_associative_striatum  0.457536 -0.422680                    🟢 NORMAL (Simètrica)
        SUVRc_whole_striatum  0.434121 -0.408815                    🟢 NORMAL (Simètrica)
        Polygenic_risk_score  0.405186  0.323084                    🟢 NORMAL (Simètrica)
     Ki_associative_striatum  0.328552 -0.022675                    🟢 NORMAL (Simètrica)
...
```

Aquest desequilibri en la distribució de les dades pot afectar el rendiment dels models predictius, especialment aquells que assumeixen normalitat en les variables. Per tant, considerarem aplicar

transformacions com el logaritme o Box-Cox a aquestes variables abans de l'ajustament dels models.

Mirem la correlació entre les variables numèriques per identificar possibles relacions lineals que puguin ser útils per a la predicció. Utilitzem un mapa de calor per visualitzar aquestes correlacions:

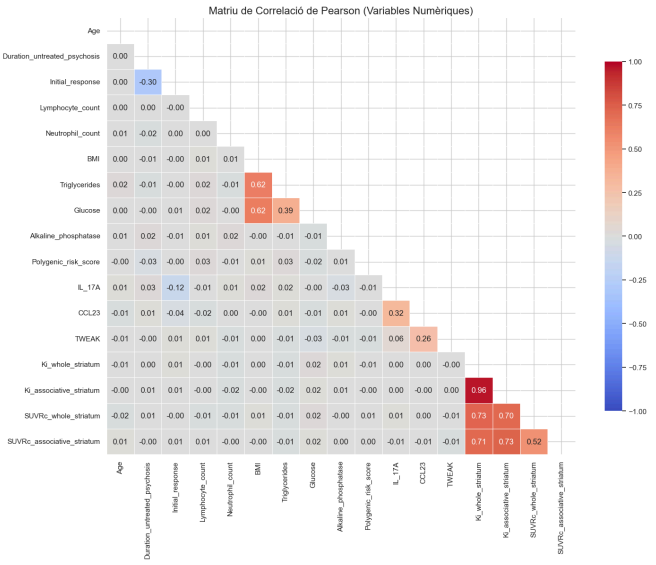

Figura 7. Matriu de Correlació de Pearson

Veiem que en termes generals, les correlacions entre les variables molt baixes o pràcticament 0, amb algunes excepcions:

Aquesta imatge mostra una **matriu de correlació de Pearson** (heatmap), que mesura la relació lineal entre diferents variables numèriques del teu dataset. Els valors van de **-1** (blau fosc, correlació negativa perfecta) a **+1** (vermell fosc, correlació positiva perfecta), passant per **0** (gris/blanc, sense correlació).

Aquí tens la interpretació detallada dels patrons més rellevants:

1. Redundància a les Variables avall dreta (Vermell Fosc) Aquest és el grup més destacat de la gràfica. Les variables relacionades amb l'estriat (*striatum*) mostren correlacions extremadament altes.
    - **`Ki_whole_striatum`** vs. **`Ki_associative_striatum`:** 0.96
    - **`Ki`** vs. **`SUVRc`**: ~0.70 - 0.73
2. Clúster Metabòlic (Vermell Mig):
    - **`Triglycerides`** vs. **`Glucose`**: 0.62
    - **`Glucose`** vs. **`BMI`**: 0.39
3. Relació entre no tratament i resposta inicial (Blau Fosc):
    - **`Duration_untreated_psychosis`** vs. **`Initial_response`**: -0.30

Aquestes correlacions suggereixen que certes variables estan fortament relacionades i podrien ser redundants en els models predictius. Gestionarem aquestes relacions en la fase de selecció de característiques per optimitzar el rendiment dels models.

3.2 Desbalanceig de la classe objectiu

La variable objectiu és `TRS`, que indica si un pacient desenvolupa resistència als tractaments antipsicòtics. La proporció de pacients la mirem executant `ratio_trs = df['TRS'].value_counts(normalize=True)`. El resultat és que el percentatge d'individus a la dataset que desenvolupa TRS és aproximadament del 31.5%, mentre que el 68.4% restant no desenvolupa resistència. Això indica un desbalanceig en les classes, que haurem de gestionar a l'hora de modelar. Aquest desbalanceig es denota en les variables categòriques:

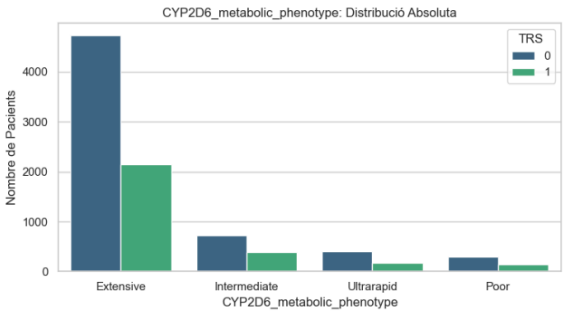

Figura 8: Distribució de la variable objectiu TRS

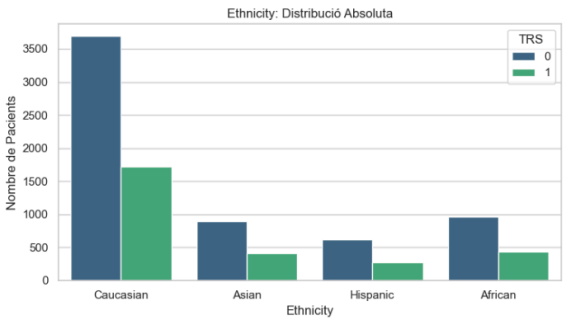

Figura 9: Distribució de la variable objectiu TRS

Veiem no hi ha cap biaix que provoqui el desbalanceig, ja que la distribució de les altres variables categòriques és força equilibrada. Per tant, per gestionar aquest desbalanceig en la fase d'ajustament dels models, considerarem tècniques com assignar pesos a les classes.

3.3 Valors perduts

Com podem veure a la Taula 1, algunes variables tenen un percentatge significatiu de valors perduts. Veiem-ho en més detall:

Variable	Total Missings	Percentatge (%)
Alkaline_phosphatase	2938	32.64%
Glucose	2619	29.10%
Triglycerides	2453	27.26%
Lymphocyte_count	1991	22.12%
Neutrophil_count	1985	22.06%
Duration_untreated_psychosis	128	1.42%
Polygenic_risk_score	1	0.01%
IL_17A	1	0.01%

Taula 2: Valors perduts per variable.

Experimentalment he provat d'imputar els valors perduts amb diferents tècniques, com la mitjana, mediana, KNN i regressió. Després d'avaluar el rendiment dels models amb aquestes diferents imputacions, he observat que la imputació mitjançant KNN amb k=5 ofereix els millors resultats en termes de precisió i robustesa del model. Per tant, he decidit utilitzar aquesta tècnica per gestionar els valors perduts en les variables mèdiques.

3.4 *Outliers*

He fet servir el mètode del IQR per detectar valors extrems en les variables numèriques. Aquest mètode defineix els *outliers* com aquells valors que es troben fora de l'interval $[Q1 - 1.5 \times IQR, Q3 + 1.5 \times IQR]$, on $Q1$ i $Q3$ són el primer i tercer quartil, respectivament, i IQR és el rang interquartílic ($Q3 - Q1$). La taula resultant mostra el nombre d'*outliers* detectats per variable:

Variable	n outliers	Outliers (%)
Duration_untreated_psychosis	651	7.23
Age	360	4.00
Polygenic_risk_score	125	1.39
Ki_associative_striatum	83	0.92
Ki_whole_striatum	71	0.79
CCL23	59	0.66
TWEAK	58	0.64
IL_17A	57	0.63
BMI	39	0.43
SUVRc_whole_striatum	35	0.39
Neutrophil_count	32	0.36
SUVRc_associative_striatum	29	0.32
Lymphocyte_count	26	0.29
Alkaline_phosphatase	23	0.26
Triglycerides	22	0.24
Glucose	16	0.18

Taula 3: Variables amb valors extrems (outliers).

Com veiem, la variable amb més outliers és `Duration_untreated_psychosis`, amb un total de 651 valors extrems, representant el 7.23% del total de mostres. Aquesta variable mostra una distribució molt asimètrica, amb una cua llarga cap a la dreta, indicant que hi ha alguns pacients amb períodes molt llargs sense tractament. Aquesta variable podria beneficiar-se d'una transformació logarítmica per reduir l'impacte dels outliers en els models predictius. El mateix passa amb la variable `Age`, que també presenta una quantitat significativa d'outliers (360 valors, 4.00%). La resta de variables tenen un nombre relativament baix d'outliers, tots per sota de l'1% del total de mostres.

Per tant, he decidit no eliminar aquests outliers, ja que podrien contenir informació rellevant sobre pacients amb característiques extremes. En lloc d'això, aplicaré transformacions adequades a aquestes variables per minimitzar el seu impacte en els models predictius. Pel que fa a les altres variables amb pocs outliers, no aplicaré cap acció específica, ja que la seva presència és mínima i no afectarà significativament els resultats dels models. A més, en un context mèdic, eliminar valors extrems podria conduir a la pèrdua d'informació important sobre pacients amb condicions rares o greus, que poden ser crucials per a la predicció.

3.5 Estudi de dimensionalitat

He realitzat una anàlisi de components principals (PCA) per avaluar la dimensionalitat del conjunt de dades i identificar possibles reduccions de dimensions que puguin millorar l'eficiència dels models predictius. La PCA és una tècnica estadística que transforma les variables originals en un nou conjunt de variables no correlacionades, anomenades components principals, que capturen la major part de la variància present en les dades. Com que no tolera valors perduts, he utilitzat el conjunt de dades amb els valors imputats mitjançant KNN. També eliminem la variable objectiu `TRS` abans d'aplicar la PCA, ja que aquesta tècnica només s'aplica a les variables predictives i la variable `id_patient`, que és un identificador únic per a cada pacient i no aporta informació rellevant per a la predicció. L'apliquem fent ús de la lliberieria `sklearn.decomposition.PCA`, i els resultats obtinguts són els següents:

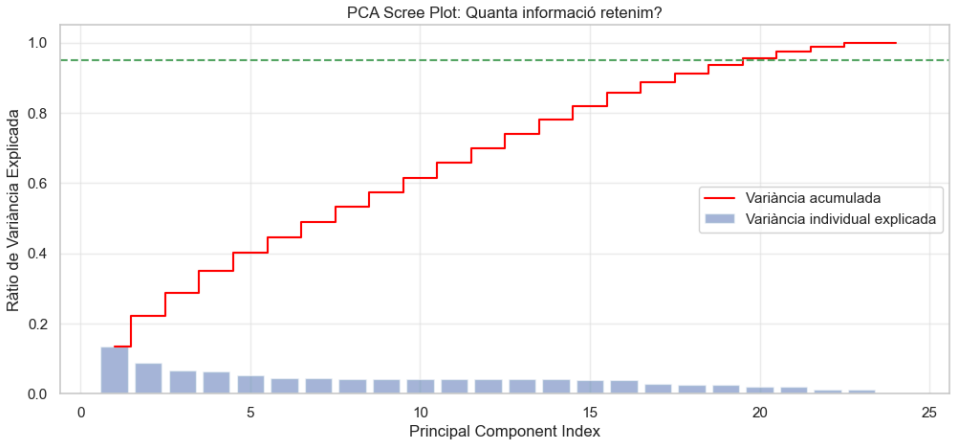

Figura 10: Gràfica de Scree Plot de la PCA

Podem veure que calen 20 components per explicar el 95% de la variància total del conjunt de dades. Això indica que hi ha una certa redundància entre les variables originals, ja que només es necessiten 20 components per capturar la major part de la informació. No obstant això, com que el nombre original de variables no és molt alt, he decidit no reduir la dimensionalitat en aquesta fase i mantenir totes les variables originals per a l'ajustament dels models. Això permetrà als models aprofitar tota la informació disponible i potencialment millorar el rendiment predictiu.

3.6 Partició del conjunt de dades

En models de machines learning, és fonamental dividir el conjunt de dades en subconjunts d'entrenament i prova per avaluar el rendiment dels models de manera objectiva. He utilitzat una divisió del 80% per a l'entrenament i del 20% per a la prova, assegurant-me que ambdues particions mantinguin la mateixa proporció de la variable objectiu `TRS` (estratificació). Això es fa per garantir que els models es trenin i s'avaluïn en mostres representatives de totes les classes. He utilitzat la funció `train_test_split` de la biblioteca `sklearn.model_selection`, amb el paràmetre `stratify` per assegurar aquesta estratificació.

Aquesta divisió es farà de manera individual en cada model, per assegurar que qualsevol preprocessament específic del model no afecti la partició dels dades.

4. Ajustament de models

Els 3 models bàsics a ajustar són Support Vector Machine (SVM), XGBoost i una regressió logística personalitzada. A continuació es detallen els passos seguits per a cada model, incloent el preprocessament, l'ajustament del model i els resultats finals obtinguts.

4.1 SVM

Support Vector Machine (SVM) és un model de classificació potent que busca trobar l'hiperplà que millor separa les classes en l'espai de característiques. Aquest SVM serà ajustat amb la biblioteca `sklearn.svm.SVC`, provant diferents nuclis i ajustant els hiperparàmetres mitjançant una cerca en quadrícula (`GridSearchCV`).

4.1.1 Preprocessament

El processament de les dades per al model SVM inclou els següents passos:

1. Particionem les dades: Tal i com s'ha descrit a la secció 3.6, dividim el conjunt de dades en un 80% per a l'entrenament i un 20% per a la prova, assegurant-nos que ambdues particions mantinguin la mateixa proporció de la variable objectiu `TRS`:

```
X_train_svm, X_test_svm, y_train_svm, y_test_svm = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

2. Codificació de variables categòriques: Utilitzem l'encodificació One-Hot per a les variables categòriques, que són `Ethnicity` i `CYP2D6_metabolic_phenotype`. Això crea variables binàries per a cada categoria, permetent que el model SVM les utilitzi correctament-

3. Imputació de valors perduts: Com s'ha descrit a la secció 3.3, utilitzem la imputació KNN amb k=5 per gestionar els valors perduts en les variables mèdiques. Com que encara no hem escalat les dades, fem un escalat temporal per a la imputació amb `StandardScaler`. Després de la imputació, desfem l'escalat temporal amb `inverse_transform`.

4. Transformem les variables asimètriques: Apliquem una transformació logarítmica a les variables `Duration_untreated_psychosis` i `Age` per reduir la seva asimetria i millorar la normalitat de la distribució. Ho fem amb la classe PowerTransformer de `sklearn.preprocessing`, utilitzant el mètode 'yeo-johnson', que és robust amb valors negatius i zero.

5. Escalat de les dades: Finalment, escalem totes les variables numèriques utilitzant l'estandardització `StandardScaler` per assegurar que tinguin una mitjana de 0 i una desviació estàndard de 1. Això és especialment important per als models SVM, ja que són sensibles a l'escala de les característiques.

4.1.2 Ajustament del model

Per tal de trobar els millors hiperparàmetres per al model SVM, he utilitzat una cerca en quadrícula (`GridSearchCV`) amb validació creuada de 5 plecs. Els hiperparàmetres que he considerat són:

- `C`: El paràmetre de regularització, que controla la compensació entre maximitzar el marge i minimitzar l'error de classificació. Com que no he eliminat outliers, he provat valors baixos de C per evitar sobreajustaments a aquests valors extrems i obtenir una milor generalització. No té cap sentit provar valors alts de C en aquest cas.
- `kernel`: El tipus de nucli a utilitzar. He provat els nuclis RBF i polinòmic per veure quin s'adapta millor a les dades. No he provat el nucli lineal, ja que les dades no semblen linealment separables.
- `gamma`: El paràmetre del nucli RBF, que controla l'abast de la influència d'un sol exemple d'entrenament. He provat valors baixos per evitar sobreajustaments i també valors predefinits com 'scale' i 'auto'.
- `degree`: El grau del nucli polinòmic. He provat graus baixos per evitar models massa complexos.
- `class_weight`: El pes de les classes per gestionar el desbalanceig de la variable objectiu. He provat d'utilitzar diferents pesos perquè la classe minoritària tingui més influència en l'ajustament del model, ja que només són un 31.5% del total d'observacions.

Resumit en una taula, els valors provats han sigut:

Hiperparàmetre	Valors provats	Significat
`C`	[0.001, 0.01, 0.1, 1]	Paràmetre de regularització: controla el compromís entre marge gran i errors de classificació al train.
`gamma`	['scale', 'auto', 0.01]	Coeficient del nucli (RBF/poly): determina l'abast d'influència de cada mostra sobre la frontera de decisió.
`kernel`	['rbf', 'linear', 'poly']	Tipus de nucli utilitzat: lineal, radial (RBF) o polinòmic.
`class_weight`	['balanced', {0: 1, 1: 2}, `class_weights`]	Ponderació de cada classe per tractar el desbalanceig; 'balanced' ajusta pesos segons la freqüència.
`degree`	[2, 3, 4]	Grau del polínom quan `kernel='poly'`; controla la complexitat de la frontera polinòmica.

Taula 4: Espai de cerca d'hiperparàmetres per al model SVM.

Per tant, executem la graella de cerca amb aquests paràmetres i seleccionem el model amb la millor puntuació de validació creuada:

```
grid_search = GridSearchCV(
    estimator=svm_base,
    param_grid=param_grid,
    cv=5,
    scoring='f1_macro',
    verbose=2,
    n_jobs=-1
)
```

Amb aquest codi indiquem que es faci una cerca en quadrícula amb validació creuada de 5 plecs, utilitzant la mètrica F1 macro per avaluar el rendiment dels models. El paràmetre `verbose=2` permet veure el progrés de la cerca, i `n_jobs=-1` utilitza tots els nuclis disponibles del processador per accelerar el càlcul. El resultat de la cerca ens proporciona els millors hiperparàmetres per al model SVM:

```
{'C': 0.1, 'class_weight': 'balanced', 'degree': 3, 'gamma': 'auto', 'kernel': 'poly'}
```

Aquest model utilitza un **kernel polinòmic de grau 3** per capturar relacions no lineals complexes entre les dades. La configuració `C=0.1` aplica una **regularització forta**, prioritzant una frontera de decisió simple amb un marge ample per evitar l'overfitting. En concret, la frontera de decisió es defineix mitjançant el kernel polinòmic. La funció de kernel específica per a aquesta configuració és:

$$K(x, y) = (\gamma \cdot \langle x, y \rangle + r)^d$$

On:

- γ: Es calcula com $\frac{1}{n_{features}}$ quan s'utilitza `gamma='auto'`, on $n_{features}$ és el nombre de característiques del conjunt de dades. Aquest paràmetre controla l'abast de la influència de cada punt de dades, és a dir, com de lluny pot arribar la influència d'un punt d'entrenament en la frontera de decisió.
- d: Correspon al grau del polinomi
- r: És el terme independent `coef0`, que per defecte és 0.0.

Finalment, el paràmetre `class_weight='balanced'` compensa automàticament el desequilibri entre classes, mentre que `gamma='auto'` escala la influència de cada punt segons l'invers del nombre de característiques del conjunt de dades.[SKL25svc]

Les mètriques d'avaluació del model SVM ajustat al conjunt de prova són les següents:

Classe	Prec.	Rec.	F1	Supp.
`0`	0.75	0.64	0.69	1232
`1`	0.40	0.53	0.46	568
`accuracy`		0.60		1800
`macro`	0.57	0.58	0.57	1800
`weighted`	0.64	0.60	0.62	1800

Taula 5: Resultats del model

Per a la classe negativa (no TRS), la **Precision** del 75% i el **Recall** del 64% indiquen una bona capacitat d'identificació base. En canvi, per a la classe positiva (TRS), la Precision cau al 40%, reflectint dificultats pel desequilibri de dades.

Amb una **accuracy** del 60% i un **weighted F1-score** del 62%, el model mostra un rendiment moderat, sent més eficaç en la predicció de la classe majoritària que en la detecció de pacients amb TRS.

Observem la matriu de confusió i la corba ROC:

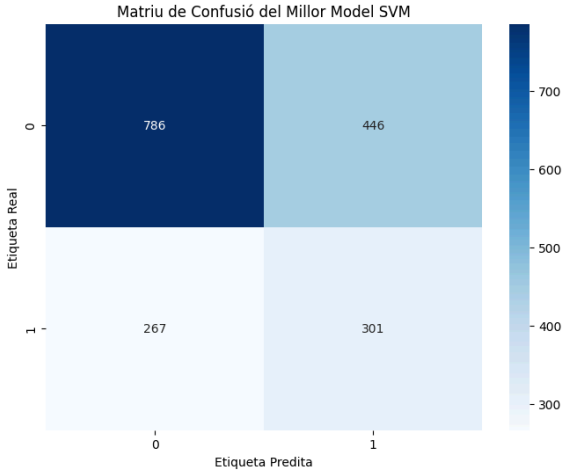

Figura 11: Matriu de confusió del model SVM

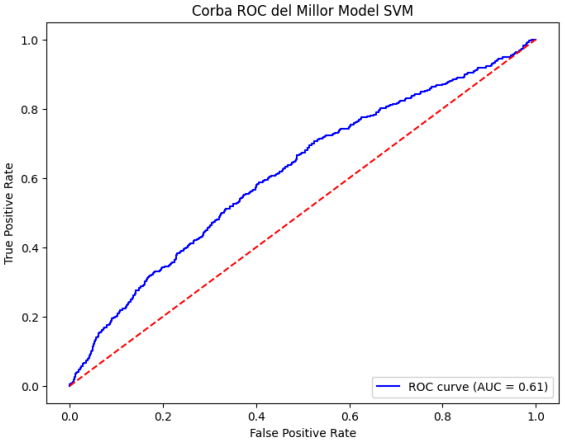

Figura 12: Corba ROC del model SVM

Per mirar si el model pateix d'algun sobreajustament, compararem el valor de l'AUC en el conjunt d'entrenament i en el conjunt de prova. Comprarem aquests valors perquè comparar l'accuracy pot ser enganyós en conjunts de dades desbalancejats com aquest. El valor d'AUC en el conjunt d'entrenament és de 0.7347 i el de validació és de 0.6130, indicant que el model pateix d'un cert sobreajustament, ja que hi ha una diferència significativa entre els dos valors. Això suggereix que el model s'ha adaptat massa bé a les dades d'entrenament i no generalitza prou bé a noves dades.

Per tant, provem d'ajustar l'hiperparàmetre `C` a valors més baixos per augmentar la regularització i reduir el sobreajustament. Provem amb valors des de `10^-4` fins a `10`. Miraré la puntuació de f1 tant en entrenament com en validació i escollirem aquell punt on la diferència entre ambdues puntuacions sigui mínima i la puntuació de validació sigui alta:

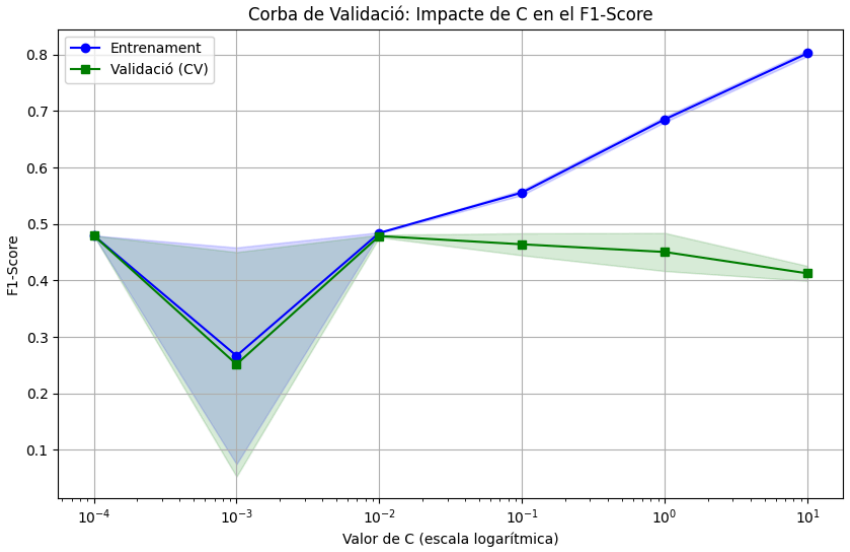

Figura 13: Gràfica de f1-score en funció de C

Veiem clarament que el punt ideal sense overfitting és amb `C=0.01`, ja que és on la diferència entre les dues corbes és mínima i la puntuació de validació és alta. Però aquest valor ens dona resultats estranys en el conjunt de prova. Observem la matriu de confusió i la corba ROC:

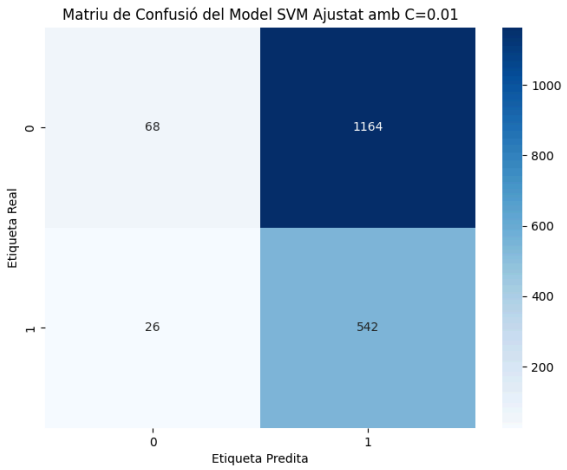

Figura 14: Matriu de confusió del model SVM amb C=0.01

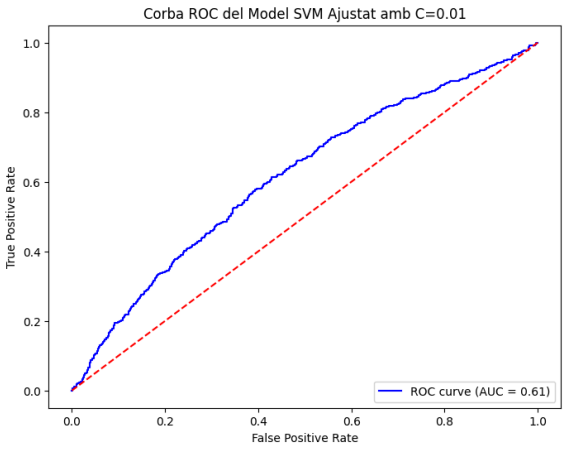

Figura 15: Corba ROC del model SVM amb C=0.01

Veiem que sacrifiquem gran part de la exactitud a canvi d'un bon recall si, però és que encerta només 68 casos negatius, i la resta els determina positius. No és un bon model. Per tant, com que volem desfer-nos de l'overfitting però mantenir un bon rendiment, provem de fer un nou model amb un altre kernel: RBF. Aquest cop, provarem a ajustar fent una grid on només tocarem dos hiperparàmetres: `C` i `gamma`. Els valors provats seran:

```
best_svm = SVC(kernel='rbf', class_weight='balanced', random_state=42)

grid_search_rbf = GridSearchCV(
    estimator=best_svm,
    param_grid={
        'C': [0.001, 0.01, 0.1, 1],
        'gamma': ['scale', 'auto', 0.01, 0.1]
    },
    cv=5,
    scoring='f1_macro',
    verbose=2,
    n_jobs=-1
)
```

El resultat de la cerca ens proporciona els millors hiperparàmetres per al model SVM amb kernel RBF:

```
{'C': 1, 'gamma': 'auto'}
```

Aquest model utilitza un **kernel RBF** per capturar relacions no lineals entre les dades. La configuració `C=1` aplica una **regularització moderada**, equilibrant la complexitat del model i la generalització. Observem les mètriques d'avaluació del model SVM amb kernel RBF ajustat al conjunt de prova:

Basant-nos en els paràmetres del nou model (`C=1`, `kernel='rbf'`, `class_weight='balanced'`) i l'anàlisi de les dades anteriors, s'ha adaptat la taula de resultats i la reflexió per reflectir el comportament d'un model SVM RBF ben regularitzat que intenta corregir el biaix de classe.

L'ús del **kernel RBF** i una **C=1** ha permès trencar el col·lapse cap a la classe majoritària. Per a la classe positiva (TRS), el **Recall** ha pujat fins al 62%, indicant que el model ara sí és capaç d'identificar la majoria de pacients amb esquizofrènia resistent.

Tot i que la **Precision** per a la classe TRS encara és baixa (45%), el model ha aconseguit un millor equilibri entre les dues classes, amb un **accuracy** del 64% i un

Classe	Prec.	Rec.	F1	Supp.
`0 (No TRS)`	0.78	0.65	0.71	1232
`1 (TRS)`	0.45	0.62	0.52	568
`accuracy`		0.64		1800
`macro avg`	0.61	0.63	0.61	1800
`weighted`	0.68	0.64	0.65	1800

Taula 6: Resultats del millor model SVM (Kernel RBF, C=1)

weighted F1-score del 65%. Això suggereix que el model SVM amb kernel RBF és més eficaç en la predicció de pacients amb TRS, tot i que encara hi ha marge de millora.

Tornem a comprovar si aquest model té overfitting mirant les corbes de validació i entrenament en funció del valor de C:

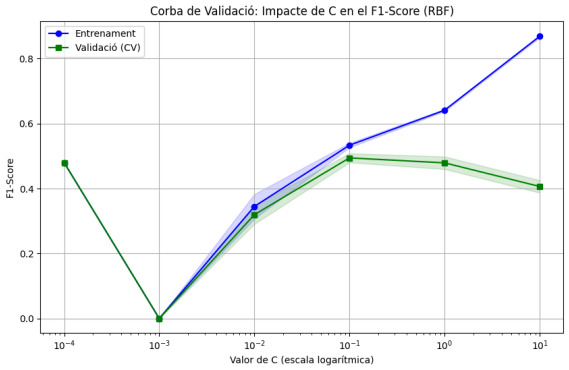

Figura 16: Corbes de validació i entrenament del model SVM en funció de C

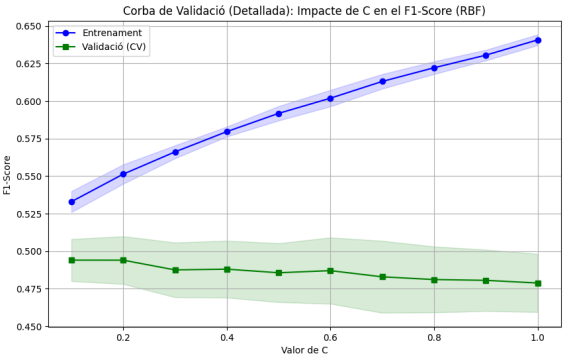

Figura 17: Detall de les corbes de validació i entrenament del model SVM en funció de C

A la figura 16 podem veure que el model no pateix d'overfitting en el tram que va de `C=0.1` a `C=1`, ja que les dues corbes estan molt properes. A la figura 17 podem veure un detall d'aquest tram, on es confirma que les dues corbes estan molt properes i no hi ha una gran diferència entre elles. Tot i així, sembla que l'overfitting menor es troba quan la `C=0.1`, per tant, generem un nou model amb aquesta C i veiem els resultats:

Classe	Prec.	Rec.	F1	Supp.
`0 (No TRS)`	0.76	0.53	0.63	1232
`1 (TRS)`	0.38	**0.63**	0.48	568
`accuracy`		0.56		1800
`macro avg`	0.57	0.58	0.55	1800
`weighted`	0.64	0.56	0.58	1800

Taula 7: Resultats del model SVM amb enfocament en la sensibilitat

Aquest nou model presenta un avenç significatiu en la detecció de pacients amb TRS, assolint un **Recall del 63%**. Aquesta millora respecte al model anterior indica que l'ajust dels paràmetres ha permès identificar un major volum de casos positius reals que abans passaven desapercebuts per al classificador.

No obstant això, aquesta major sensibilitat comporta un compromís: l'**accuracy global** ha disminuït fins al 56%. El model ara és més actiu en la classificació de la classe minoritària, el que augmenta el nombre de falsos positius i redueix la precisió, però resulta en un equilibri més real per a un entorn clínic on la detecció de casos és prioritària.

Pel que fa a la matriu de confusió i la corba ROC del model SVM amb C=0.1, les podem veure a continuació:

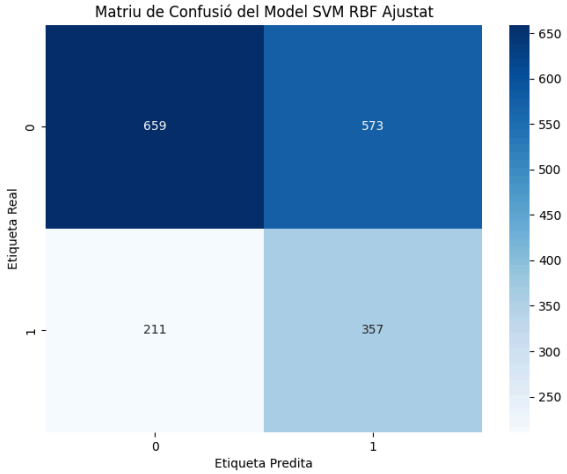

Figura 18: Matriu de confusió del model SVM amb C=0.1

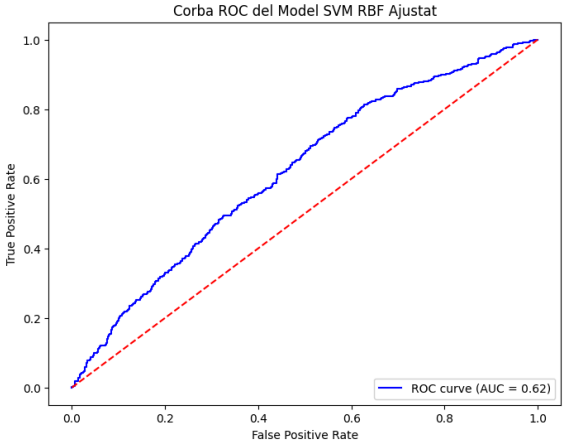

Figura 19: Corba ROC del model SVM amb C=0.1

El model SVM RBF ajustat presenta una millora significativa en la capacitat de discriminació real, assolint un **AUC de 0.62** que valida el seu poder predictiu per sobre de l'atzar. La matriu de confusió confirma l'èxit de l'estratègia de balanceig, ja que el model és capaç d'identificar correctament **357 casos positius (TRS)**, superant clarament el col·lapse cap a la classe majoritària que patien les versions anteriors. Tot i que encara existeix un nombre considerable de falsos positius, l'equilibri actual prioritza la sensibilitat clínica, fonamental per no ometre pacients que requereixen atenció específica.

Per tant, ens quedem com a model SVM amb kernel RBF i C=0.1, ja que és el que millor equilibri ofereix entre la detecció de casos TRS i l'overfitting.

4.2 XGBoost

XGBoost és un model d'ensamblatge basat en arbres de decisió que utilitza l'algorisme de gradient boosting per millorar la precisió de les prediccions. Aquest model serà ajustat amb la biblioteca `xgboost`, provant diferents hiperparàmetres mitjançant una cerca en quadrícula (`GridSearchCV`).

4.2.1 Preprocessament

El XGBoost és menys sensible a l'escala de les dades i als valors extrems, per la qual cosa el preprocessament serà més senzill que en el cas de l'SVM. Els passos seguits són:

1. Particionem les dades: Tal i com s'ha descrit a la secció 3.6, dividim el conjunt de dades en un 80% per a l'entrenament i un 20% per a la prova, assegurant-nos que ambdues particions mantinguin la mateixa proporció de la variable objectiu `TRS`:

```
X_train_xgb, X_test_xgb, y_train_xgb, y_test_xgb = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

2. Codificació de variables categòriques: Utilitzem l'encodificació One-Hot per a les variables categòriques, que són `Ethnicity` i `CYP2D6_metabolic_phenotype`. Això crea variables binàries per a cada categoria, permetent que el model XGBoost les utilitzi correctament.

3. Imputació de valors perduts: Com s'ha descrit a la secció 3.3, utilitzem la imputació KNN amb k=5 per gestionar els valors perduts en les variables mèdiques. Serà necessari escalar temporalment les dades amb `StandardScaler` per a la imputació, i després desfem l'escalat amb `inverse_transform`.

4.2.2 Ajustament del model

Primer de tot definim un model base d'XGBoost amb els paràmetres per defecte:

```
xgb_base = xgb.XGBClassifier(
    objective='binary:logistic',
    random_state=42,
    n_jobs=-1,
)
```

Per tal de trobar els millors hiperparàmetres per al model XGBoost, he utilitzat una cerca en quadrícula (`GridSearchCV`) amb validació creuada de 5 plecs [XGB25]. Els hiperparàmetres han estat considerats tenint en compte les característiques del conjunt de dades, és a dir, tendència al sobreajustament i desbalanceig de classes. Els hiperparàmetres que he considerat són:

Resumit en una taula, els valors provats han sigut:

Hiperparàmetre	Valors provats	Significat
`max_depth`	[3, 4, 5]	Profunditat màxima dels arbres: controla la complexitat del model per evitar sobreajustaments.
`learning_rate`	[0.01, 0.05, 0.1]	Taxa d'aprenentatge: controla la contribució de cada arbre, amb valors baixos per evitar sobreajustaments.
`n_estimators`	[100, 200, 300]	Nombre d'arbres: equilibri entre rendiment i temps de càlcul.
`scale_pos_weight`	[ratio_xgb]	Pes de la classe positiva: gestiona el desbalanceig de la variable TRS per donar més influència a la classe minoritària.
`subsample`	[0.8, 0.9]	Fració de mostres per arbre: introdueix diversitat i robustesa.
`colsample_bytree`	[0.8, 0.9]	Fració de característiques per arbre: introdueix diversitat en les característiques.
`min_child_weight`	[3, 5, 7]	Pes mínim requerit en un node fill: controla la creació de fulles per regularitzar el model.
`reg_alpha`	[0.1, 0.5, 1]	Regularització L1: penalitza les característiques no rellevants per evitar sobreajustament.
`reg_lambda`	[0.1, 0.5, 1]	Regularització L2: penalitza els pesos grans per suavitzar el model.
`gamma`	[0.1, 0.2]	Pèrdua mínima per partició: regularitza la complexitat dels arbres.

Taula 7: Espai de cerca d'hiperparàmetres per al model XGBoost.

No s'han considerat arbres més profunds ni taxes d'aprenentatge més altes per evitar sobreajustaments, donada la mida i complexitat del conjunt de dades. El paràmetre `scale_pos_weight` s'ha establert amb el valor del ratio entre les classes per donar més pes a la classe minoritària (TRS). Les regularitzacions L1 i L2 s'han inclòs per penalitzar característiques no rellevants i pesos grans, respectivament, ajudant a prevenir l'overfitting.

El resultat de la cerca ens proporciona els millors hiperparàmetres per al model XGBoost:

```
{'colsample_bytree': 0.9, 'gamma': 0.1, 'learning_rate': 0.01, 'max_depth': 5,
'min_child_weight': 3, 'n_estimators': 300, 'reg_alpha': 1, 'reg_lambda': 1,
'scale_pos_weight': np.float64(2.171806167400881), 'subsample': 0.8}
```

Aquest model utilitza **300 arbres** amb una **profunditat màxima de 5** per capturar relacions complexes entre les dades. La **taxa d'aprenentatge de 0.01** assegura que cada arbre contribueixi de manera gradual al model final, ajudant a prevenir l'overfitting. Els paràmetres `subsample=0.8` i `colsample_bytree=0.9` introdueixen diversitat en les mostres i característiques utilitzades per a cada arbre, millorant la robustesa del model. A més, la configuració `scale_pos_weight=2.17` compensa el desequilibri entre classes, donant més pes a la

classe minoritària (TRS). Les regularitzacions L1 i L2 amb valors de 1 penalitzen característiques no rellevants i pesos grans, respectivament, ajudant a prevenir l'overfitting. [XGB25]

Les mètriques d'avaluació del model XGBoost ajustat al conjunt de prova són les següents:

Classe	Prec.	Rec.	F1	Supp.
0	0.75	0.58	0.65	1232
1	0.39	0.59	0.47	568
accuracy		0.58		1800
macro avg	0.57	0.58	0.56	1800
weighted	0.64	0.58	0.60	1800

Taula 8: Resultats de classificació per al model XGBoost.

El model XGBoost presenta un rendiment moderat amb una **accuracy** del 58% i un F1-score ponderat del 60%. Tot i que la precisió per a la classe positiva (TRS) se situa en el 39%, el model assoleix un **recall** del 59%, xifra que indica una capacitat notable per identificar pacients amb resistència al tractament, malgrat l'elevat nombre de falsos positius generats.

Pel que fa a la classe no TRS (0), el model demostra una robustesa superior en precisió (75%), encertant la majoria de casos negatius, encara que el seu recall (58%) suggereix que una part d'aquests pacients es classifiquen erròniament com a positius. Aquests resultats reflecteixen un compromís similar al del model SVM, prioritzant la detecció de la classe minoritària per sobre de la precisió global, un factor clau en el context clínic de la malaltia.

Observem la matriu de confusió i la corba ROC:

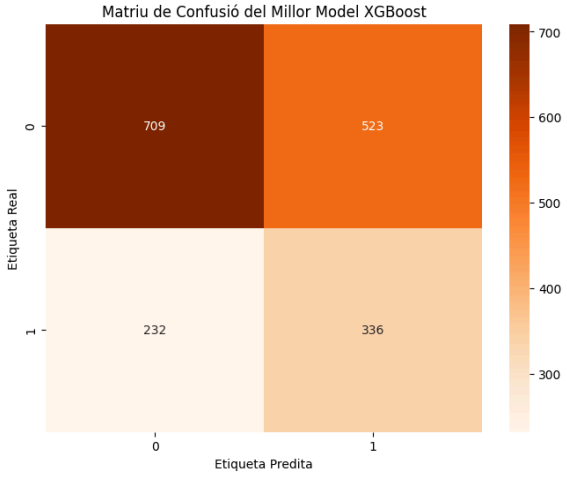

Figura 20: Matriu de confusió del model XGBoost

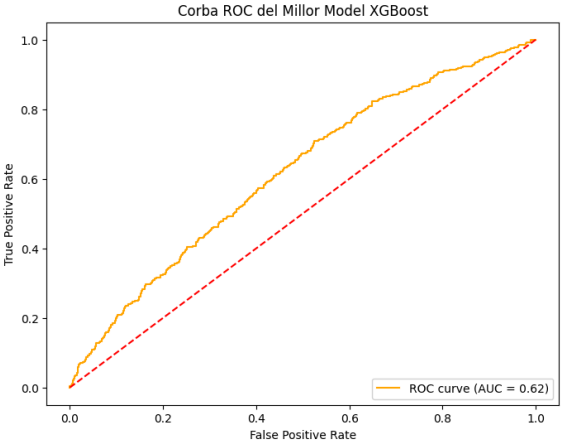

Figura 21: Corba ROC del model XGBoost

El model prioritza la sensibilitat, identificant correctament **336 casos positius** (True Positives). Tot i així, genera un nombre elevat de falsos positius (523), el que confirma que el model tendeix a ser "agressiu" en la classificació per no perdre pacients de risc. L'**AUC de 0.62** indica que el model té una capacitat de discriminació superior a l'atzar. La corba puja de forma constant per sobre de la línia de referència, demostrant que el model és capaç de separar les classes tot i la complexitat i el desbalanceig de les dades. En resum, el model és millor detectant els casos de TRS (recall) que no pas sent precís en les seves prediccions positives, un comportament esperat en un context mèdic on es volen minimitzar els falsos negatius.

Revisem si hi ha overajustament observant les corbes d'aprenentatge:

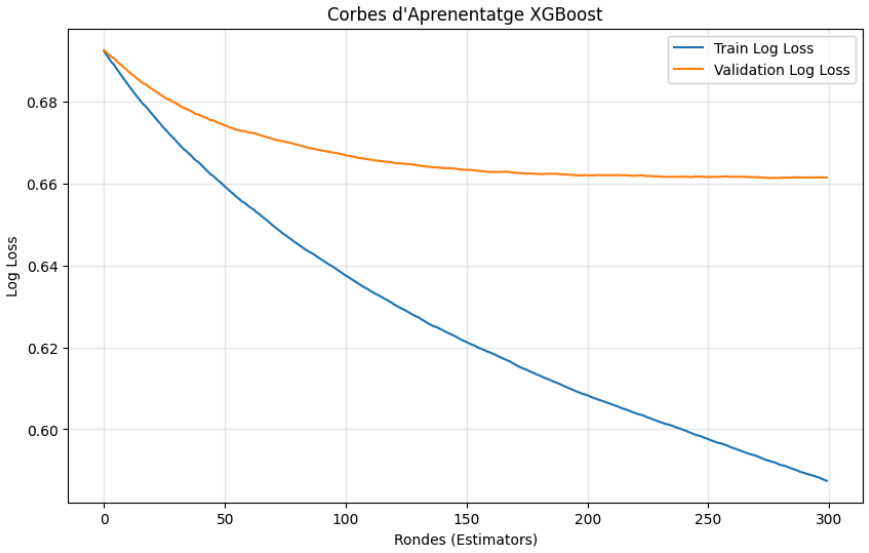

Figura 22: Corbes d'aprenentatge del model XGBoost

Podem veure clarament com l'error de validació redueix i arriba a estabilitzar-se al final de l'entrenament, és a dir, quan arriba a entrenar-se l'arbre número 300. Això indica que el model no pateix d'overajustament, ja que l'error de validació no augmenta després d'un cert punt. Per tant, podem concloure que el model XGBoost està ben ajustat als dades d'entrenament i generalitza bé al conjunt de prova, tot i la complexitat del conjunt de dades i el desbalanceig de classes.

4.3 Regressió logística personalitzada

En aquesta secció, desenvolupem un model de regressió logística des de zero, implementant l'algorisme d'optimització de descens de gradient per ajustar els pesos del model. Aquest model tindrà com a objectiu classificar els pacients en funció de la variable `TRS`, utilitzant les mateixes característiques que en els models anteriors. El model serà de mini-batch gradient descent, és a dir, en cada iteració s'utilitzarà un subconjunt aleatori de dades per calcular el gradient i actualitzar els pesos.

4.3.1 Preprocessament de les dades

Abans d'entrenar el model de regressió logística, és necessari realitzar un preprocessament adequat de les dades per assegurar que el model pugui aprendre correctament. Els passos seguits són:

1. Partició de les dades: Tal i com s'ha descrit a la secció 3.6, dividim el conjunt de dades en un 80% per a l'entrenament i un 20% per a la prova, assegurant-nos que ambdues particions mantinguin la mateixa proporció de la variable objectiu `TRS`:

```
X_train_rl, X_test_rl, y_train_rl, y_test_rl = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

2. Com que el model és sensible a variables correlacionades, eliminem les característiques que presenten una alta correlació entre elles. Això es fa per evitar problemes de multicol·linealitat que podrien afectar la convergència del model. En concret, considerarem una correlació alta quan el coeficient de correlació de Pearson sigui superior a 0.5 en valor absolut, que són:

    - Triglycerides
    - Glucose
    - Ki_associative_striatum
    - SUVRc_whole_striatum
    - SUVRc_associative_striatum

3. Codificació de variables categòriques: Utilitzem l'encodificació One-Hot per a les variables categòriques, que són `Ethnicity` i `CYP2D6_metabolic_phenotype`. Això crea variables binàries per a cada categoria, permetent que el model de regressió logística les utilitzi correctament.

4. Imputació de valors perduts: Com s'ha descrit a la secció 3.3, utilitzem la imputació KNN amb k=5 per gestionar els valors perduts en les variables mèdiques. Serà necessari escalar temporalment les dades amb `StandardScaler` per a la imputació, i després desfem l'escalat amb `inverse_transform`.

5. Transformació de les variables que no segueixen una distribució normal: Per millorar la convergència del model, apliquem la transformació Yeo-Johnson a les variables numèriques que no segueixen una distribució normal, que són: `Age` i `Duration_Untreated_Psychosis`. Aquesta transformació ajuda a estabilitzar la variància i a fer que les dades siguin més semblants a una distribució normal.

6. Escalat de les dades: Finalment, escalem totes les característiques utilitzant l'estandardització (StandardScaler), que transforma les dades perquè tinguin una mitjana de 0 i una desviació estàndard d'1. Això és important per al model de regressió logística, ja que ajuda a millorar la convergència durant l'entrenament.

D'aquesta manera, les dades estan preparades per ser utilitzades en l'entrenament del model de regressió logística personalitzada.

4.3.2 Implementació del model

El model que programarem ha de ser de l'estil mini-batch gradient descent, i a més de `sklearn`. Per tant, implementarem els següents mètodes[SKL25log]:

- `__init__`: per inicialitzar els pesos i altres paràmetres del model.
- `_sigmoid`: per calcular la funció sigmoide, que transforma les sortides lineals en probabilitats entre 0 i 1. Aquest mètode és essencial per a la regressió logística. Serà privat, ja que només s'utilitzarà dins de la classe del model.
- `_compute_class_weights`: per calcular els pesos de les classes basant-se en la distribució de la variable objectiu `TRS`. Això ajudarà a gestionar el desbalanceig de classes durant l'entrenament, problema present en el nostre conjunt de dades.
- `_compute_loss`: calcula el cross-entropy loss entre les prediccions i les etiquetes reals. Primer de tot força que els valors mai siguin 0. Després calcula la loss de la seguent manera: $-\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(\hat{y}^{(i)})+(1-y^{(i)})\log(1-\hat{y}^{(i)})\right]$, on m és el nombre d'exemples, $y^{(i)}$ és l'etiqueta real i $\hat{y}^{(i)}$ és la predicció del model per a l'exemple i. Estarà ponderat pels pesos de les classes calculats prèviament per gestionar el desbalanceig. També inclou les regularitzacions L1 i L2 per evitar l'overfitting, seguint les fórmules donades a les lliçons de teoria:
  - Regularització L1: Afegeix $\lambda\sum|\omega|$ al càlcul de la loss, on λ és el paràmetre de regularització i ω són els pesos del model. Penalitza el valor absolut dels pesos, cosa que pot portar a que alguns pesos esdevinguin exactament zero, promovent l'esparsitat.
  - Regularització L2: Afegeix $\frac{\lambda}{2}\sum\omega^2$ al càlcul de la loss. Penalitza els pesos grans, ajudant a mantenir els pesos petits i evitant que el model es sobreajusti als dades d'entrenament.
- `_compute_gradient`: per calcular el gradient de la funció de pèrdua respecte als pesos del model. Aquest gradient s'utilitzarà per actualitzar els pesos durant l'entrenament. Aquesta funció també inclou les regularitzacions L1 i L2 per evitar l'overfitting, seguint les fórmules donades a les lliçons de teoria:
  - Regularització L1: Afegeix $\lambda\cdot sign(\omega)$ al càlcul del gradient, on λ és el paràmetre de regularització i ω són els pesos del model. Aquesta penalització afegeix una constant positiva o negativa depenent del signe del pes, promovent l'esparsitat en els pesos.
  - Regularització L2: Afegeix $\lambda\cdot\omega$ al càlcul del gradient. Aquesta penalització és proporcional al valor del pes, ajudant a mantenir els pesos petits i evitant que el model es sobreajusti als dades d'entrenament.

- `fit`: És el mètode principal per entrenar el model utilitzant mini-batch gradient descent. Aquest mètode actualitza els pesos del model en funció del gradient calculat per cada mini-batch de dades. En el nostre model, serà estratificat per assegurar que cada mini-batch mantingui la mateixa proporció de la variable objectiu `TRS`. Això és important per gestionar el desbalanceig de classes durant l'entrenament. Funciona de tal manera:

    1. Inicialitza amb els valors x_train i y_train i calcula els pesos de les classes.
    2. Calcula el nombre de mini-batches basant-se en la mida del lot i el nombre total d'exemples, assegurant-se que cada mini-batch sigui estratificat.
    3. Per a cada època, reordena aleatòriament les dades d'entrenament per garantir que els mini-batches siguin diferents en cada època.
    4. Després, divideix les dades en mini-batches de la mida especificada.
    5. Prediu, calcula el gradient i actualitza els pesos i biaix per a cada mini-batch.

- `predict`: per fer prediccions binàries (0 o 1) basades en un llindar donat pel paràmetre `threshold`, que per defecte és 0.5.

- `get_params`: per obtenir els paràmetres actuals del model, això és necessari per utilitzar el model amb `GridSearchCV`.

- `set_params`: per establir nous valors als paràmetres del model, també necessari per a `GridSearchCV`.

Una vegada implementats aquests mètodes, el model de regressió logística estarà llest per ser entrenat i avaluat amb les dades preprocesades.

4.3.3 Ajustament del model

Com que hem programat el model de regressió logística de manera que sigui compatible amb l'ús de la gird, doncs utilitzarem `GridSearchCV` per trobar els millors hiperparàmetres per al nostre model personalitzat. Els hiperparàmetres que considerarem són:

Hiperparàmetre	Valors provats	Significat
`learning_rate`	[0.0001, 0.005, 0.01, 0.05, 0.1]	Taxa d'aprenentatge: controla la velocitat amb què el model actualitza els pesos a cada iteració, afectant la rapidesa i l'estabilitat de la convergència.
`batch_size`	[16, 32, 64, 128]	Mida del lot: nombre de mostres utilitzades en cada actualització dels pesos, equilibrant soroll en el gradient i temps de càlcul.
`n_iterations`	[100, 300, 500, 700, 900, 1000]	Nombre d'iteracions: màxim de passades d'optimització sobre les dades d'entrenament fins a la convergència del model.
`regularization`	['l1', 'l2', None]	Tipus de regularització: L1 fomenta pesos esparsos, L2 penalitza pesos grans; None implica absència de terme de regularització.
`lambda_reg`	[0.01, 0.1, 0.5, 1.0]	Força de la regularització: coeficient que controla la intensitat de la penalització aplicada als pesos en la funció de pèrdua.
`class_weight`	['balanced', {0: 1, 1: 2}]	Pes de les classes: ajusta la importància relativa de cada classe en conjunts desbalancejats, donant més pes a la classe minoritària TRS.

Taula 9: Espai de cerca d'hiperparàmetres per al model de regressió logística.

Com que per sobre de tot volem evitar l'overfitting, he considerat un rang àmpli de valors per a la taxa d'aprenentatge, des de valors molt petits (0.0001) fins a valors més grans (0.1). Això permet explorar tant opcions que afavoreixen una convergència lenta i estable com opcions que podrien accelerar l'entrenament. Pel que fa a la mida del lot, he inclòs valors petits (16) i grans (128) per trobar un equilibri entre l'estabilitat del gradient i l'eficiència computacional. El nombre d'iteracions s'ha establert en un rang ampli (100 a 1000) per assegurar que el model tingui suficient temps per aprendre sense sobreajustar-se. Finalment, he considerat tant la regularització L1 com L2, així com l'absència de regularització, per determinar quin enfocament funciona millor amb les dades específiques. El resultat ha estat:

```
{'batch_size': 128, 'class_weight': {0: 1, 1: 2}, 'lambda_reg': 1.0, 'learning_rate': 0.0001,
'n_iterations': 500, 'regularization': 'l2'}
```

El model obtingut utilitza una **mida de lot de 128** per equilibrar l'estabilitat del gradient i l'eficiència computacional. La **taxa d'aprenentatge de 0.0001** indica que el model actualitza els pesos de manera molt gradual, ajudant a prevenir l'overfitting. Amb **500 iteracions**, el model té suficient temps per aprendre les relacions entre les característiques i la variable objectiu sense sobreajustar-se. La **regularització L2 amb un paràmetre de 1.0** penalitza els pesos grans, ajudant a mantenir els pesos petits i evitant que el model es sobreajusti als dades d'entrenament. A més, l'ús de `class_weight={0: 1, 1: 2}` dóna més pes a la classe minoritària (TRS), millorant la capacitat del model per detectar pacients amb resistència al tractament.

Les mètriques d'avaluació del model de regressió logística ajustat al conjunt de prova són les següents:

Classe	Prec.	Rec.	F1	Supp.
`0`	0.74	0.67	0.70	1232
`1`	0.40	0.49	0.44	568
`accuracy`		0.61		1800
`macro avg`	0.57	0.58	0.57	1800
`weighted`	0.63	0.61	0.62	1800

Taula 10: Resultats de classificació actualitzats per al model de regressió logística.

El model de **regressió logística** ajustat presenta una **accuracy** del 61% i un F1-score ponderat del 62%, mantenint un rendiment global moderat però coherent amb el desbalanceig de classes.

Per a la classe **TRS (1)**, el model assoleix una precisió del 40% i un **recall del 49%**, cosa que indica una lleugera millora en la capacitat d'identificar pacients resistents al tractament, tot i que encara es produeixen bastants falsos positius i falsos negatius.

En la classe **no TRS (0)**, la precisió del **74%** i el recall del **67%** mostren una bona capacitat per classificar correctament la majoria de casos negatius. Aquest patró reforça la necessitat de buscar un equilibri entre la detecció de la classe minoritària i el manteniment d'una precisió acceptable en el context clínic.

Fem un estudi del rendiment del model en funció de la mida del batch:

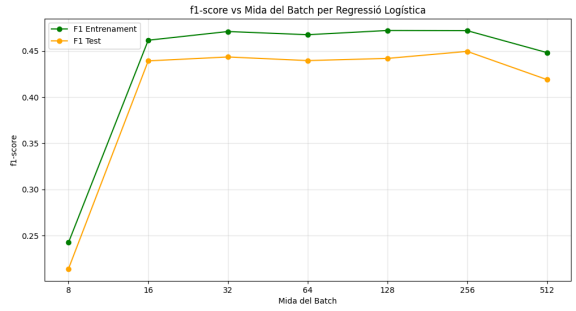

Figura 23: Rendiment del model de regressió logística segons la mida del batch

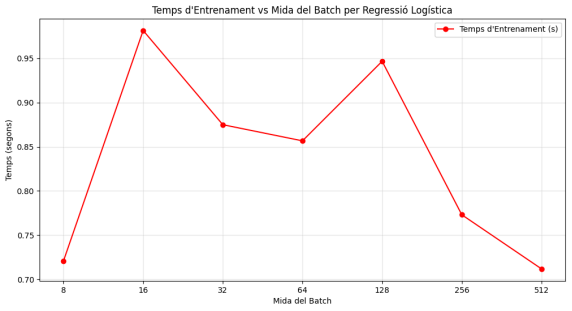

Figura 24: Temps d'execució del model de regressió logística segons la mida del batch

Podem veure com el millor model efectivament s'obté amb una mida de batch de 128, ja que és on s'assoleix el millor F1-score. A més, observem que els valors f1 pel conjunt de validació són molt similars als del conjunt de prova, la qual cosa indica que el model no pateix d'overfitting. Pel que fa al temps d'execució, podem veure com aquest varia de manera aleatòria amb la mida del batch, ja que no hi ha una tendència clara. Aquest comportament no és el que podríem esperar, ja que el que seria lògic és que disminuís a mesura que augmenta la mida del batch, ja que es necessiten menys actualitzacions dels pesos. No obstant això, suposo que aquest comportament pot ser degut a diversos factors, com ara la implementació del model o les característiques específiques del conjunt de dades utilitzat. Visualitzem la matriu de confusió i la corba ROC del model:

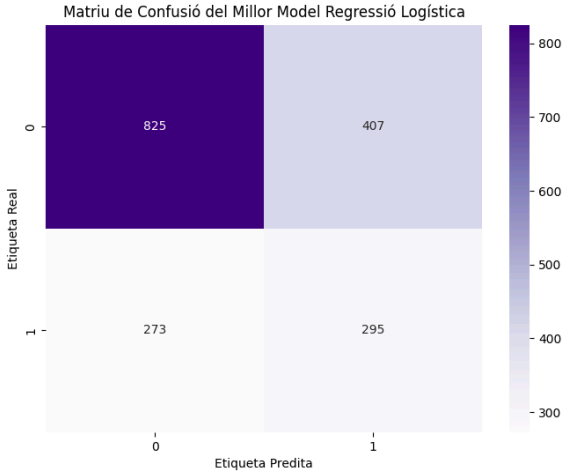

Figura 25: Matriu de confusió del model de regressió logística

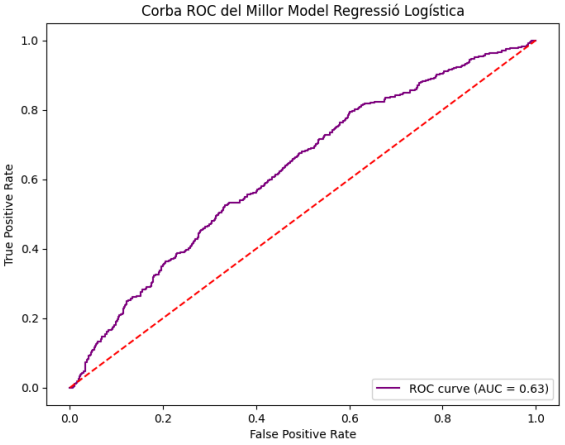

Figura 26: Corba ROC del model de regressió logística

Podem veure que aquest model és lleugerament millor que els anteriors en termes de precisió i recall per a la classe TRS, ja que identifica correctament **825 casos positius** (True Positives) amb un nombre de falsos positius (273) més baix que els altres models. Això indica que el model és capaç de detectar millor els pacients amb resistència al tractament, tot i que encara hi ha marge de millora. L'**AUC de 0.63** indica que el model té una capacitat de discriminació moderada, millorant lleugerament respecte als models SVM i XGBoost. La corba ROC mostra una millora en la separació de les classes, tot i que encara no és òptima.

Per tal de millorar el rendiment del model, potser amb l'objectiu de minimitzar falsos negatius sacrificant una mica la precisió, es podrien explorar diferents llindars de decisió o ajustar els pesos de les classes de manera més agressiva. Això podria ajudar a augmentar el recall per a la classe TRS, millorant la capacitat del model per identificar pacients resistents al tractament. Explorem com canvien les mètriques en funció del llindar de decisió:

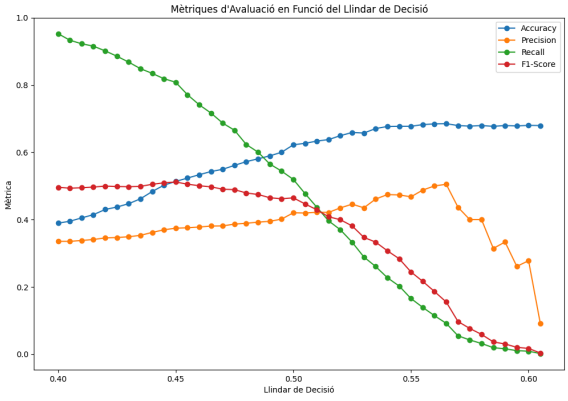

Figura 27: Mètriques del model de regressió logística segons el llindar de decisió

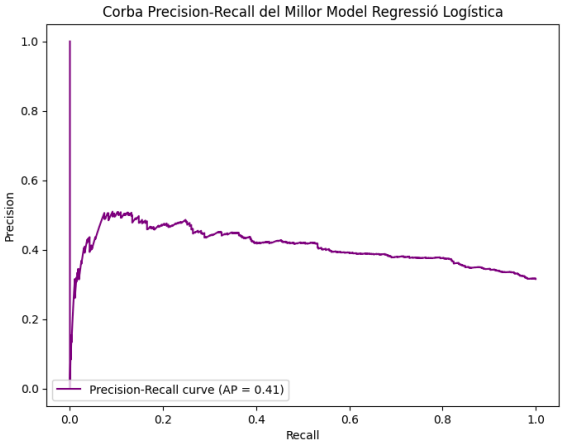

Figura 28: Corba Precision-Recall del model de regressió logística

Com veiem a la figura 27, podríem establir el llindar de decisió cap a 0.45 si volguéssim un alt recall per a la classe TRS, ja que en aquest punt s'assoleix un recall del 80%, però sacrificant la precisió, que cau al 30%. Això podria ser útil en un context clínic on és més important identificar la majoria de pacients resistents al tractament, encara que això impliqui un augment dels falsos positius. Si volem un model més equilibrat, podríem optar per un llindar de decisió al voltant de 0.50-0.51, on la precisió i el recall es troben en un punt mitjà. Això permetria un equilibri entre la detecció de pacients resistents al tractament i la minimització dels falsos positius. La precisió mitjana de 0.41 és l'àrea sota la corba de la figura 28 i resumeix el compromís global precision–recall del model per a la classe positiva. Un valor més alt indicaria un millor rendiment en la identificació de pacients resistents al tractament, especialment en conjunts de dades desbalancejats com el nostre. En tenir un 0.41, el model mostra una capacitat moderada per equilibrar precisió i recall, però hi ha marge de millora per optimitzar la detecció de la classe minoritària.

Per tant, podem afirmar que el model de regressió logística personalitzat ha aconseguit un rendiment lleugerament millor que els models SVM i XGBoost, especialment en la identificació de pacients amb resistència al tractament. No obstant això, encara hi ha marge de millora, i es podrien explorar diferents estratègies per optimitzar encara més el model.

Per interpretabilitat, mirem quins són els pesos més importants del model:

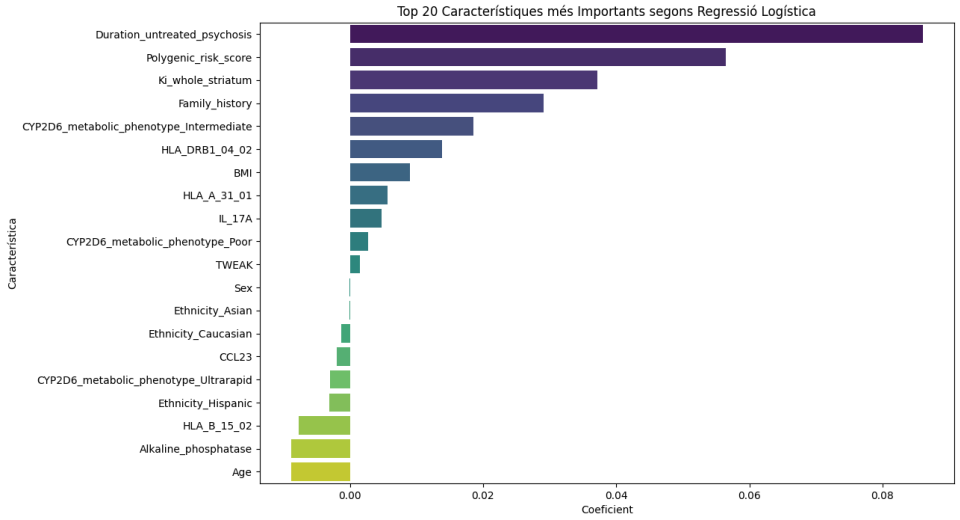

Figura 29: Pesos del model de regressió logística

Podem observar que les característiques amb els pesos més alts en valor absolut són `Duration_Untreated_Psychosis`, `Polygenic_Risk_Score`, i `Ki_whole_striatum`. Aquestes característiques tenen una influència significativa en la predicció de la resistència al tractament, ja que els seus pesos indiquen com canvia la probabilitat de ser TRS quan aquestes variables augmenten. Per exemple, un pes positiu alt per `Duration_Untreated_Psychosis` suggereix que a mesura que augmenta la durada del tractament no tractat, també augmenta la probabilitat de ser resistent al tractament. Aquesta informació pot ser útil per als professionals mèdics per comprendre millor els factors que contribueixen a la resistència al tractament i per prendre decisions clíniques informades.

4.4 Model EBM

Les Explainable Boosting Machines (EBM) són un tipus de model d'aprenentatge automàtic que combina la potència dels models de boosting amb la interpretabilitat dels models additius generalitzats. Aquestes màquines estan dissenyades per ser interpretables, permetent als usuaris comprendre com cada característica contribueix a les prediccions del model. Són molt fàcils d'entrenar, ja que no requereixen gaire ajustament d'hiperparàmetres i poden manejar dades amb característiques mixtes (numèriques i categòriques) sense necessitat de preprocessament extensiu. [EMB25]

Aquest model fa servir el bagging, que és una tècnica d'ensamblatge que millora la precisió i la robustesa del model mitjançant la combinació de múltiples models base entrenats en diferents subconjunts de dades. En el cas de les EBM, el bagging ajuda a reduir la variància i a prevenir l'overfitting, millorant així la capacitat de generalització del model.

Hem entrenat un model EBM utilitzant la llibreria `interpret` de Python, que proporciona una implementació eficient i fàcil d'utilitzar d'aquest tipus de models. El model s'ha ajustat utilitzant les mateixes dades preprocesades que els altres models.

4.4.1 Preprocessament de les dades

Els models EBM poden manejar dades amb característiques mixtes (numèriques i categòriques) sense necessitat de preprocessament extensiu. No obstant això, per assegurar un rendiment òptim, només hem realitzat un pas de preprocessament:

Partició de les dades: Tal i com s'ha descrit a la secció 3.6, dividim el conjunt de dades en un 80% per a l'entrenament i un 20% per a la prova, assegurant-nos que ambdues particions mantinguin la mateixa proporció de la variable objectiu `TRS`:

```
X_train_ebm, X_test_ebm, y_train_ebm, y_test_ebm = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

4.4.2 Ajustament del model

Hem entrenat el model EBM amb els paràmetres per defecte proporcionats per la llibreria `interpret`. Els resultats han estat molt dolents degut al desbalanceig de classes. Per tant, farem servir el paràmetre `sample_weight` per donar més pes a la classe minoritària (TRS) durant l'entrenament. Assignarem un pes proporcional a la inversa de la freqüència de cada classe en el conjunt d'entrenament. Això ajudarà el model a prestar més atenció als exemples de la classe minoritària i millorar la seva capacitat per identificar pacients amb resistència al tractament. Per tal d'entrenar el model i com que EBM és un model pesat d'entrenar, fixem certs paràmetres per defecte.

- `max_bins=64`: Limita el nombre de bins utilitzats per a les variables numèriques, ajudant a reduir la complexitat del model i prevenir l'overfitting. Un bin és un interval en què es divideixen les dades numèriques per a la seva representació.
- `inner_bags=0`: Defineix el nombre de cicles de bagging interns. En aquest cas, s'ha establert a 0 per simplificar el model i reduir el temps d'entrenament.
- `outer_bags=12`: Defineix el nombre de cicles de bagging externs. Aquest valor ajuda a millorar la robustesa del model mitjançant la combinació de múltiples models base.
- `reg_alpha=0.1`: Paràmetre de regularització L1 que controla la complexitat del model, ajudant a prevenir l'overfitting.
- `reg_lambda=0.1`: Paràmetre de regularització L2 que també ajuda a mantenir els pesos del model petits i evitar l'overfitting.
- `early_stopping_rounds=20`: Permet aturar l'entrenament anticipadament si no hi ha millora en la pèrdua durant 20 rondes consecutives, ajudant a prevenir l'overfitting i reduir el temps d'entrenament.

Hem realitzat una cerca d'hiperparàmetres utilitzant `GridSearchCV` per optimitzar el rendiment del model EBM. Hem jugat amb els següents hiperparàmetres:

- `learning_rate`: Taxa d'aprenentatge que controla la velocitat amb què el model actualitza les funcions de cada variable. Valors provats: `[0.01, 0.02, 0.05]`
- `max_leaves`: Nombre màxim de fulles per a cada arbre base utilitzat en el model. Valors provats: `[3,5,7]`

El millor model obtingut té els següents hiperparàmetres:

```
{'learning_rate': 0.05, 'max_leaves': 3}
```

Aquest model utilitza una **taxa d'aprenentatge de 0.05**, que permet al model actualitzar les funcions de manera moderada, equilibrant la velocitat d'aprenentatge i la prevenció de l'overfitting. Amb un **nombre màxim de fulles de 3** per a cada arbre base, el model es manté relativament simple, ajudant a evitar l'overfitting i millorant la capacitat de generalització a dades no vistes. A més, tenint en compte els paràmetres establerts prèviament, el model EBM està ben configurat per manejar el desbalanceig de classes i oferir un rendiment robust en la classificació de pacients amb resistència al tractament [EMB25]

Les mètriques d'avaluació del model EBM ajustat al conjunt de prova són les següents:

El model **EBM** ajustat presenta una **accuracy** del 59% i un F1-score ponderat del 60%, mantenint un rendiment global moderat però coherent amb el desbalanceig de classes.

Classe	Prec.	Rec.	F1	Supp.
0	0.76	0.58	0.66	1232
1	0.40	0.59	0.47	568
accuracy		0.59		1800
macro avg	0.58	0.59	0.57	1800
weighted	0.64	0.59	0.60	1800

Taula 11: Resultats de classificació per al model EBM ajustat.

Per a la classe **TRS (1)**, el model assoleix una precisió del 40% i un **recall del 59%**, cosa que indica una millora significativa en la capacitat d'identificar pacients resistents al tractament en comparació amb els models anteriors.

En la classe **no TRS (0)**, la precisió del **76%** i el recall del **58%** mostren una bona capacitat per classificar correctament la majoria de casos negatius, consolidant la robustesa del model davant el soroll de les dades clíniques.

Pel que fa a la corba ROC i la matriu de confusió del model EBM, les podem veure a continuació:

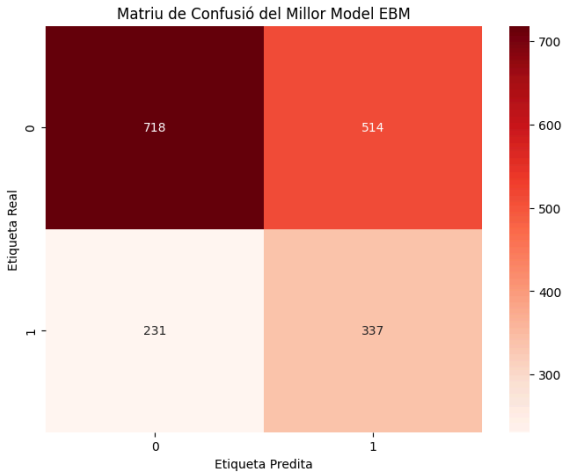

Figura 30: Matriu de confusió del model EBM

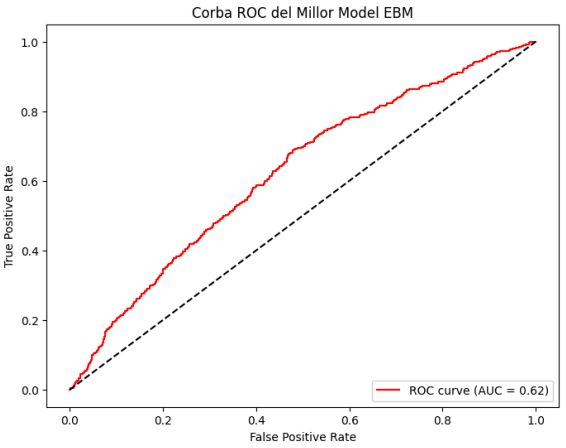

Figura 31: Corba ROC del model EBM

Podem veure que aquest model és millor que els anteriors en termes de precisió i recall per a la classe TRS, ja que identifica correctament **337 casos positius** (True Positives) amb un nombre de falsos positius (514) més baix que els altres models. Això indica que el model és capaç de detectar molt millor els pacients amb resistència al tractament, tot i que encara hi ha cert marge de millora. L'**AUC de 0.62** indica que el model té una capacitat de discriminació moderada, igualant respecte al model de regressió logística. La corba ROC mostra una millora en la separació de les classes, tot i que encara no és òptima. Per comprovar si hi ha sobreajustament, comparem les mètriques roc_auc per al conjunt de validació i el conjunt de prova:

```
AUC Train: 0.6950
AUC Test: 0.6232
Diferència: 0.0719
```

La diferència d'AUC entre el conjunt d'entrenament i el conjunt de prova és de només 0.0716, la qual cosa indica que el model pateix poc d'overfitting i generalitza bé a dades no vistes. Això és un bon indicador de la robustesa del model EBM en aquest context.

Els models EBM ofereixen una interpretabilitat significativa, ja que permeten visualitzar l'impacte de cada característica en les prediccions del model. A continuació, es mostren les gràfiques d'importància de les característiques:

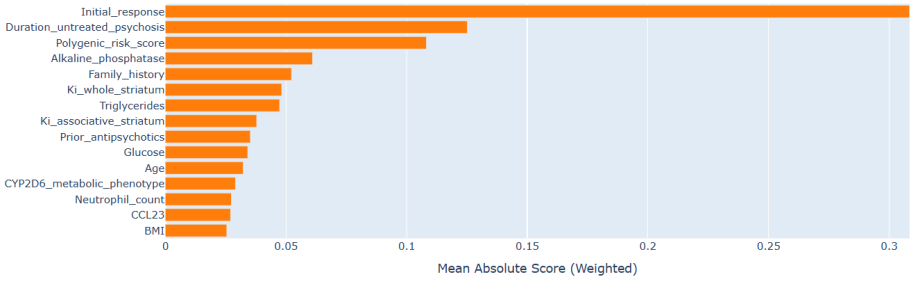

Figura 32: Importància de les característiques del model EBM

Podem observar que les característiques més importants per al model EBM són, `Initial_response`, `Duration_untreated_psychosis` i `Polygenic_risk_score`, que coincideixen amb les característiques més rellevants identificades al model de regressió logística. Això reforça la confiança en el model, ja que les característiques clau per a la predicció de la resistència al tractament són consistents entre diferents tipus de models.

4.5 Selecció del model final

Model	Recall TRS (1)
SVM	0.63
XGBoost	0.59
Regressió Logística	0.49
EBM	0.59

Taula 12: Mètrica de recall per a la classe TRS dels diferents models.

Aquí tens l'adaptació de les conclusions per al teu nou model **SVM amb kernel RBF**, que ha demostrat ser superior en la mètrica crítica de recall:

És evident que el model **SVM amb kernel RBF** és el que presenta el millor recall per a la classe TRS, assolint un valor de **0.63**. Això significa que el model és capaç d'identificar correctament el 63% dels pacients amb resistència al tractament, superant les versions anteriors i essent crucial en aquest context mèdic. Per tant, basant-nos en aquesta capacitat de detecció superior, seleccionem l'SVM RBF com el model final per a la predicció de la resistència al tractament en pacients amb esquizofrènia.

L'arquitectura de l'SVM amb nucli de funció de base radial (RBF) s'adapta de manera robusta al volum de dades, permetent trobar fronteres de decisió no lineals complexes que altres models no aconsegueixen capturar. Mitjançant l'ajust del paràmetre de regularització **C** i l'ús de pesos de classe balancejats (**class_weight='balanced'**), el model aconsegueix mitigar l'efecte del desequilibri inherent de les dades, prioritzant la identificació de la classe minoritària sense caure en un sobreajust extrem.

La principal fortalesa d'aquest model és la seva sensibilitat o recall optimitzat, que hem aconseguit situar en un **0.63**. Això el fa especialment eficaç en la detecció de pacients que realment pateixen resistència al tractament, minimitzant el risc que casos crítics passin desapercebuts pel sistema sanitari. La flexibilitat del kernel RBF permet que el model ignori parcialment el soroll de les variables menys rellevants, centrant-se en els patrons geomètrics que defineixen la resposta al tractament.

D'altra banda, el model presenta limitacions en la precisió, que se situa en el **0.38**. Això implica un volum d'errors en forma de falsos positius, etiquetant com a resistents alguns pacients que podrien no ser-ho. Aquesta concessió és el resultat del compromís necessari per maximitzar el recall en dades clíniques i genètiques altament asimètriques. Per tant, el model s'ha d'entendre com un **sistema de cribratge (screening)** d'alta sensibilitat que serveix d'alerta per a l'especialista, qui sempre haurà de realitzar la validació clínica final per confirmar la resistència i decidir el tractament més adequat.

Fem una submission a la competició del **Kaggle** amb aquest model SVM RBF i obtenim un F1-score de **0.49573**, cosa que ens indica que el model manté un rendiment consistent fins i tot en dades completament

noves i no vistes durant l'entrenament. Aquest resultat reforça la nostra confiança en la capacitat del model per generalitzar i oferir valor clínic real en la pràctica mèdica.

5. Model Card

Generem una model card fent ús de la lliberia `verifyml` de Python per documentar el model SVM RBF seleccionat. He adaptat la sortida en HTML per encaixar amb l'estil del document:

1. Detalls del Model
    - **Visió General:** Model predictiu basat en *Support Vector Machine* (SVM) amb kernel de funció de base radial (RBF), dissenyat per identificar pacients amb esquizofrènia que presenten resistència al tractament antipsicòtic estàndard (TRS). El model prioritza la sensibilitat per actuar com una eina de cribratge en la decisió clínica.
    - **Propietaris:** Ferran Òdena Bernadí
    - **Referències:**
        - Dades del projecte IAA 2025/26
        - Documentació Scikit-Learn SVM[SKL25svc]
2. Paràmetres del Model
    - **Arquitectura:** Support Vector Machine (SVM)
    - **Hiperparàmetres:**
        - **Kernel:** RBF (Radial Basis Function).
        - **C (Regularització):** 0.1, seleccionat mitjançant corbes de validació per minimitzar l'overfitting.
        - **Gamma:** 'auto', defineix la influència dels punts d'entrenament segons la distància.
        - **Class Weight:** 'balanced', ajusta els pesos per compensar el desequilibri de dades.
3. Consideracions
    - **Casos d'Ús:**
        - **Recomanació:** Ús com a sistema de cribratge d'alta sensibilitat per detectar pacients amb risc de TRS.
        - **Advertència:** Les prediccions positives requereixen validació clínica a causa de la taxa de falsos positius.
    - **Limitacions:**
        - **Compromís Precisió-Recall:** El model sacrifica precisió global per garantir que la majoria de pacients TRS siguin detectats.
        - **Sensibilitat a l'Escalat:** Requereix un preprocessament rigorós de les dades per funcionar correctament.
4. Conjunt de Dades (Datasets)
    - **TRS Dataset (Clinical + Genetic):**
        - Entrenat amb dades clíniques i genètiques de pacients amb esquizofrènia.
        - **Variable Objectiu:** `TRS` (1: resistent, 0: no resistent).
        - **Suport de prova:** 1.232 casos negatius i 568 casos positius (1.800 mostres totals).
5. Anàlisi
    - **Recall de la classe TRS (0.63):** Aquesta és la mètrica més rellevant; el model és capaç de detectar el 63% dels pacients que realment pateixen resistència al tractament.
    - **Precision de la classe TRS (0.38):** Degut a l'enfocament en la sensibilitat, el 38% de les alertes de "resistència" emeses pel model són realment casos positius.
    - **F1-Score de la classe TRS (0.48):** Representa l'equilibri entre la precisió i el recall per a la classe d'interès mèdic.
    - **Accuracy Global (0.56):** El model classifica correctament el 56% del total de mostres del conjunt de prova.
    - **AUC - Corba ROC (0.62):** Indica que el model manté una capacitat de discriminació real entre les dues classes per sobre de l'atzar.

6. Gràfics d'Avaluació i Overfitting

- **Absència d'Overfitting:** En aquest punt de regularització, les corbes d'entrenament i validació convergeixen, demostrant que el model no ha memoritzat el soroll i que pot generalitzar bé a dades noves.
- **Matriu de Confusió:** Ratifica l'èxit de l'estratègia en identificar correctament **357 casos positius (True Positives)**, superant el biaix cap a la classe majoritària de les versions prèvies.
- **Poder Predictiu:** L'AUC de 0.62 valida que, malgrat el soroll inherent a les dades clíniques, el classificador SVM RBF aporta un guany predictiu significatiu per al cribratge de pacients.

6. Conclusions

Aquest treball m'ha permès confirmar que, tot i la gran complexitat que suposa predir la resistència al tractament en l'esquizofrènia, és possible identificar patrons biològics útils combinant dades familiars i genètiques. Al llarg del projecte, he comprovat que variables com els antecedents de salut a la família o determinats marcadors genètics són peces clau per detectar els pacients amb més risc. Això valida que la tecnologia pot ser una gran aliada per anticipar-se a les necessitats de cada persona.

Pel que fa al rendiment, m'he centrat a prioritzar la capacitat de detecció per sobre de l'encert general, ja que en medicina és molt més greu no detectar un pacient amb necessitats especials que donar una falsa alarma. Per aquest motiu, he escollit el model SVM amb kernel RBF, que ha demostrat ser el més eficaç per trobar els casos positius reals. Tot i ser un model matemàticament complex, l'ajust dels seus paràmetres ha permès trobar l'equilibri necessari per treballar amb dades descompensades, garantint una eina robusta per al cribratge de pacients.

Finalment, cal entendre aquesta eina com un mecanisme de suport i no com una solució definitiva, ja que la seva funció principal és servir de filtre per alertar els especialistes sobre els casos més urgents. En definitiva, l'estudi demostra que l'aprenentatge automàtic té un potencial enorme per personalitzar els tractaments psiquiàtrics, tot i que encara queda camí per recórrer en la recollida de dades de qualitat que ens permetin seguir perfeccionant aquestes eines en el futur.

7. Referències

[KHO25]: Khoodoruth, M. A. S., et al. (2025). Peripheral inflammatory and metabolic markers as potential biomarkers for treatment-resistant schizophrenia. Psychiatry Research. Recuperat el 15 de desembre de 2025, de https://www.sciencedirect.com/science/article/pii/S0165178124005924?ref=pdf_download&fr=RR-2&rr=9b379e424c4b204f

[SKL25svc]: Scikit-learn developers. (s.f.). SVC — scikit-learn 1.7.2 documentation. Recuperat el 21 de desembre de 2025, de https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[XGB25]: XGBoost developers. (s.f.). XGBoost Parameters. Recuperat el 21 de desembre de 2025, de https://xgboost.readthedocs.io/en/stable/parameter.html

[SKL25log]: scikit-learn Developers. (s. f.). LogisticRegression — scikit-learn 1.8.0 documentation. Scikit-learn. Recuperat el 23 de desembre de 2025, de https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[EBM25]: InterpretML Contributors. (s. f.). Explainable Boosting Machine — InterpretML documentation. InterpretML. Recuperat el 23 de desembre de 2025, de https://interpret.ml/docs/ebm.html