Tema 2 - Trabajo práctico

Inteligencia Artificial Explicable 2024/25

Juan Eizaguerri Serrano juan.eizaguerri@alumnos.upm.es
Daniel Hernández Martínez daniel.hernandezmar@alumnos.ump.es

Índice

1. Introducción

En esta práctica vamos a resolver un problema de clasificación de individuos según su riesgo en dos clases: bueno y malo. El objetivo es abarcar tres tipos de modelos de caja blanca: lógico, lineal y Generalized additive model y comparar sus resultados e interpretabilidad. Para ello hemos implementado un árbol de clasificación, una regresión logística y un explainable boosting machine (EBM). Contamos con tres conjuntos de datos: un conjunto de entrenamiento, uno de validación y otro de prueba.

Se comenzó haciendo un pequeño análisis de los datos para posteriormente realizar un breve estudio de los hiperparámetros y se creó un árbol con los datos obtenidos. Sobre este árbol, y aprovechando la naturaleza inherentemente explicativa, se procedió a realizar explicaciones tanto globales como locales y ha realizar las diversas métricas de evaluación correspondiente, observando el patrón de alta explicabilidad y baja capacidad predictiva propio de los árboles de decisión. Finalmente se transformó el árbol en reglas para estudiar si se favorecía la explicabilidad. A continuación se implementó e interpretó un modelo de regresión logística. El último modelo interpretable implementado fue un EBM, el cual fue el que mejor relación entre interpretabilidad y rendimiento ofreció. Por último se compararon los modelos creados con un Random Forest, con el objetivo de comprobar si un modelo de caja negra ofrecía mejores resultados y observamos que este no era el caso.

2. Metodología

El objetivo de este trabajo es el de desarrollar, entrenar y evaluar distintos clasificadores, teniendo en cuenta las métricas de confusión de los modelos frente a un conjunto de datos de validación, y discutiendo la interpretabilidad global y local de dichos modelos.

El conjunto de datos cuenta con un total de 24 variables, siendo la primera de ellas *RiskPerformance* la que se tratará de predecir. Esta variable toma los valores de cadena ‘Good’ o ‘Bad’, que pueden ser mapeados a los valores 1 y 0 respectivamente. El resto de variables toman valores enteros. Los datos están divididos en un conjunto de entrenamiento que cuenta con conjuntos de entrenamiento, uno de validación que se utiliza para el ajuste de hiperparámetros y otro de test con el que se realizan las evaluaciones finales para la comparación de modelos.

Observando la variable *RiskPerformance* se observa un balanceo de los datos aproximadamente equitativo en ambos conjuntos.

Partición	n_{total}	$\#n_{Good}$ (%)	$\#n_{Bad}$ (%)
Train	6459	3109 (0.48)	3350 (0.52)
Validation	2000	967 (0.48)	1033 (0.52)
Test	1888	873 (0.47)	1015 (0.53)

Calculando la matriz de correlación del conjunto de datos se observa un alto grado de correlación entre muchas de las variables, por lo que en muchos casos pueden aportar información similar a la hora de realizar predicciones.

No existen valores nulos en ninguna de las columnas del conjunto de datos por lo que no ha sido necesario eliminar ni modificar ningún registro.

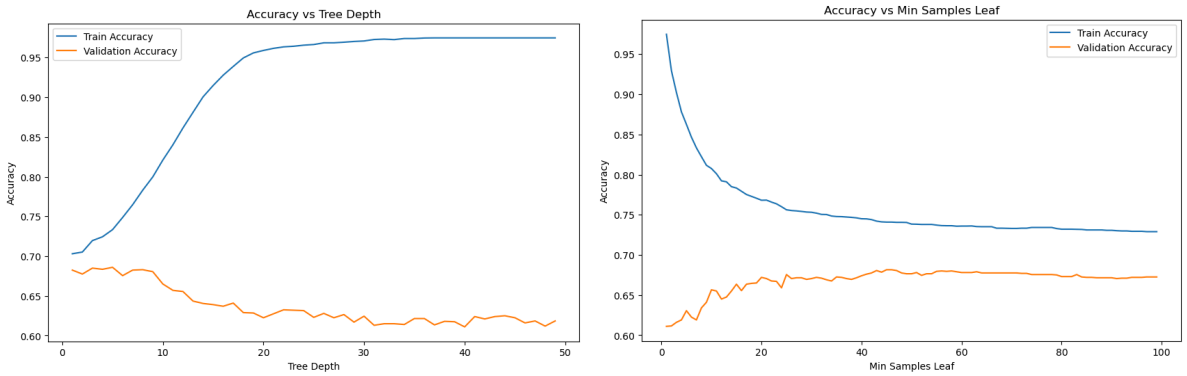
Dado que se van a utilizar modelos que usan descenso de gradiente durante su entrenamiento, se realiza una normalización estándar de los datos para evitar posibles problemas de desvanecimiento de gradiente.

Se entrenan distintos modelos de clasificadores utilizando los datos de entrenamiento y se evalúa su desempeño frente al conjunto de validación.

2.1. Modelo lógico

En primer lugar se utiliza un clasificador lógico, en concreto un árbol de decisión. Este clasificador tiene la ventaja de ser inherentemente explicable, además de requerir un grado de preparación de los datos muy bajo, pues permite el uso de datos categóricos y sin normalizar.

Con el objetivo de buscar el mejor clasificador de árbol de decisión posible, se calcula la precisión del modelo en función de los parámetros *max_depth*, que limita la profundidad máxima del árbol, y *min_samples_leaf*, que establece el número mínimo de muestras que pueden ser afectadas por una regla del árbol.

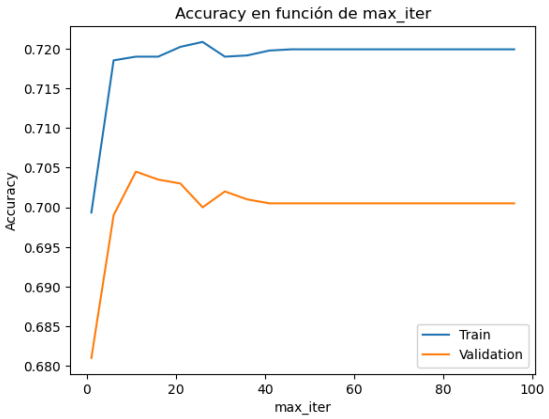


Se observa que a medida que aumenta la profundidad máxima del árbol puede producir un sobreajuste muy grande, ya que se crean reglas muy específicas haciendo que el modelo “memorice” los datos de entrenamiento, lo que tiene un efecto negativo a la hora de evaluar con datos que no había visto anteriormente. Por el contrario, el parámetro *min_samples_leaf* es una herramienta para combatir el sobreajuste, evitando las hojas ajustadas a datos específicos del conjunto de entrenamiento. El efecto de este parámetro deja de tener un efecto significativo alrededor de *min_samples_leaf*=40.

2.2. Modelo lineal

Se quiere entrenar y evaluar un modelo de regresión logística para resolver el problema de clasificación. En particular, se utiliza un modelo *LogisticRegression* de la librería *interpretml* con sus parámetros por defecto.

Se prueban distintos algoritmos de regresión, en concreto *newton-cg*, *lbfgs*, *liblinear*, *sag* y *saga*. Se obtienen con todos ellos los mismos resultados, con un *accuracy* de 0.72 sobre el conjunto de datos de entrenamiento y 0.70 sobre el de validación. Se utilizará *lbfgs* en el modelo final. Además, el algoritmo de entrenamiento converge tras 60 iteraciones, por lo que valores más altos del parámetro *max_iter* no tendrán ningún efecto en los resultados para este conjunto de datos de entrenamiento.



2.3. GAM

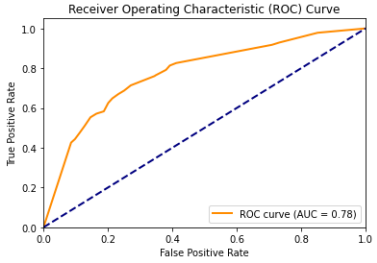
Para el modelo GAM (Generalized additive models), se ha decidido utilizar un Explainable Boosting Machine de la biblioteca interpretml, debido a la gran interpretabilidad de este modelo y su alta relación entre rendimiento e interpretabilidad. Una vez elegido el algoritmo, decidimos realizar un estudio de los hiperparámetros utilizando una búsqueda exhaustiva GridSearch. Para ello, consultamos los parámetros más importantes en la documentación de interpretml y escogimos analizar los 4 primeros, ya que debido a la exhaustividad del algoritmo GridSearch no podíamos analizar un grupo demasiado grande de parámetros. Tras la ejecución los mejores valores fueron los siguientes:inner_bag = 0, interaction = 10, max_bins = 102y smoothing_rounds = 50. Con estos hiperparámetros obtuvimos un accuracy del 73.34%. La ejecución de este algoritmo duró varias horas. Si se desease explorar un mayor número de hiperparámetros o un mayor rango del valor de los mismos, hay que tener en cuenta que el orden de coste GridSearch es exponencial. Esto implica que un aumento muy grande podría suponer que el algoritmo tarde muchas horas en ejecutarse e incluso días, lo que en nuestro caso no era factible. Otra opción podría haber sido utilizar un Random Search, de forma que puedas abarcar un mayor rango de hiperparámetros a costa de perder la exhaustividad. Aún así, decidimos conservar la exhaustividad utilizando GridSearch con un rango menor de hiperparámetros, aunque adecuado para esta práctica.

3. Resultados

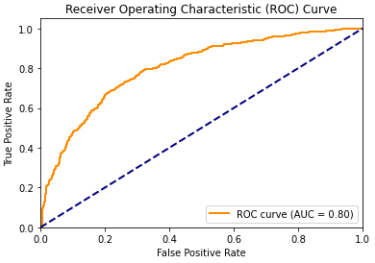
Clasificador	Mejores parámetros	ACC_{train}	ACC_{test}	AUC
Árbol de decisión	*max_depth = 5* *min_samples_leaf* = 100	0.7240	0.7255	0.7751
Regresión logística	*max_iter* = 100 *solver* = 'lbfgs'	0.7199	0.7336	0.8021
Explainable Boosting Machine	*inner_bag = 0* *interaction = 10* *max_bins=102* *smoothing_rounds = 50*	0.7436	0.7489	0.8215
Random Forest	*n_estimators=600* *max_features = log2* *max_depth = 100* *criterion = log_loss*	0.9748	0.7341	0.8161

Se realiza una evaluación del rendimiento del modelo entrenado utilizando los datos de entrenamiento y validación. La tabla anterior muestra un resumen de los resultados obtenidos en estas pruebas.Para el modelo de árbol de decisión se obtiene un *accuracy* del 73% en el conjunto de entrenamiento y 69% en el

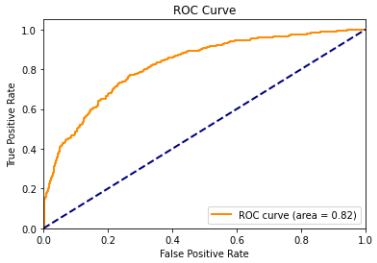
de test. Observando la curva ROC frente a datos de test se observa un desempeño significativamente mejor que el de un clasificador aleatorio.Utilizar un modelo lineal produce una ligera mejora en los resultados, obteniendo 72% *accuracy* en el conjunto de entrenamiento y 73% en el de test. La diferencia positiva entre los resultados de ambos conjuntos puede deberse a la partición aleatoria de datos, siendo este segundo conjunto más fácilmente clasificable por el modelo lineal. A la hora de utilizar un modelo más complejo, como es el caso del Explainable Boosting Machine, podemos observar como las métricas aumentan, obteniendo un 74% de accuracy en train y un 75% en test. En cambio, al utilizar un modelo de caja negra como es el Random Forest, los resultados no aumentan con respecto al EBM. A continuación se muestran las curvas ROC de los distintos modelos.



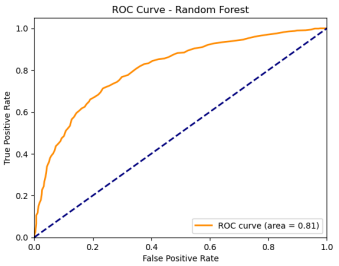
Árbol de clasificación



Regresión Logística



EBM

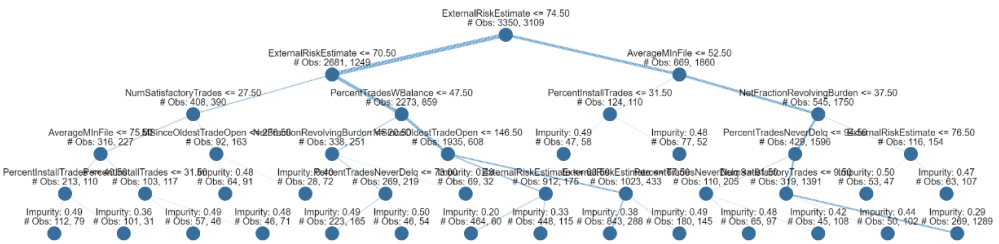


Random Forest

4. Modelos

4.1. Modelo lógico

4.1.1. Interpretación global



Analizando el árbol de podemos observar como la variable más importante del conjunto de datos para nuestro árbol es el estimado del riesgo externo (*ExternalRiskEstimate*), ya que es la que se utiliza en el nodo raíz. Esto quiere decir que es la variable que sirve para dividir el conjunto de datos de forma más efectiva. Cabe recalcar que esto sucede para nuestro árbol en concreto y no tendría por qué ser así para

todos los árboles que solucionen el problema. Podemos observar que al utilizar esta variable para dividir el conjunto de datos entre los individuos que presentan un estimado del riesgo mayor y menor a 74.50, se obtienen dos conjuntos bastante diferentes. Uno de ellos muestra un número mayor de individuos cuyo riesgo es bueno (clase 0) y el otro contiene un número mayor de individuos con riesgo alto (clase 1). Esa diferencia entre los individuos de ambos subconjuntos muestra como ExternalRiskEstimate es una variable capaz de discriminar de forma bastante efectiva entre ambas clases, lo que explica por qué es la primera que se utiliza para dividir el conjunto de datos. Además esta también es utilizada en la siguiente división realizada en el subárbol izquierdo, reforzando así la importancia de esta variable. Debido a esto, si queremos explicar las decisiones tomadas por nuestro árbol, podríamos afirmar de forma bastante segura que uno de los factores más importantes es el estimado del riesgo externo, pudiendo dar así una primera idea del por qué un individuo ha sido asignado a una clase.

Si analizamos los *features importances* de las variables, podemos observar que la variable que presenta un mayor valor es *ExternalRiskEstimate*, coincidiendo con la primera variable utilizada en el árbol. Aún así, cabe recalcar que el que una variable tenga un gran *feature importance* no implica que esta vaya a ser utilizada en todos los árboles. Puede que sea una de las más prioritarias en cierto árbol mientras que no sea utilizada en absoluto por otro árbol. En nuestro caso, todas las variables que presentan un *feature importance* positivo son utilizadas en el árbol, aunque esto no tiene por qué pasar en todos los árboles. Sin embargo, ninguna de las variables que presentan un valor de *feature importance* igual a 0 son utilizadas. Esta es una forma muy directa de poder mostrar qué variables no influyen en la decisión, pudiendo dar información sobre qué aspectos no se tienen en cuenta a la hora de decidir la clase de un individuo.

Si volvemos a la variable *ExternalRiskEstimate*, podemos observar que no solo es la primera que se utiliza, si no que además se usa otras 4 veces en nuestro árbol de profundidad 5. Esto explica por qué es la variable con mayor *feature importance*.

Si queremos seguir profundizando en qué variables son relevantes, podríamos destacar las siguientes utilizadas en el árbol, las cuales son *AverageMinFile*, NumSatisfactoryTrades, *PercentTradesWBalance*, PercentInstallTrades y NetFractionRevolvingBurden. Cuanto más bajemos en el árbol, menor influencia tendrán dichas variables en la decisión, por lo que observar las variables utilizadas en nodos muy inferiores no nos da información muy relevante a nivel global, aunque sí puede ser útil a nivel local, al tratar de explicar la decisión tomada para un individuo en concreto.

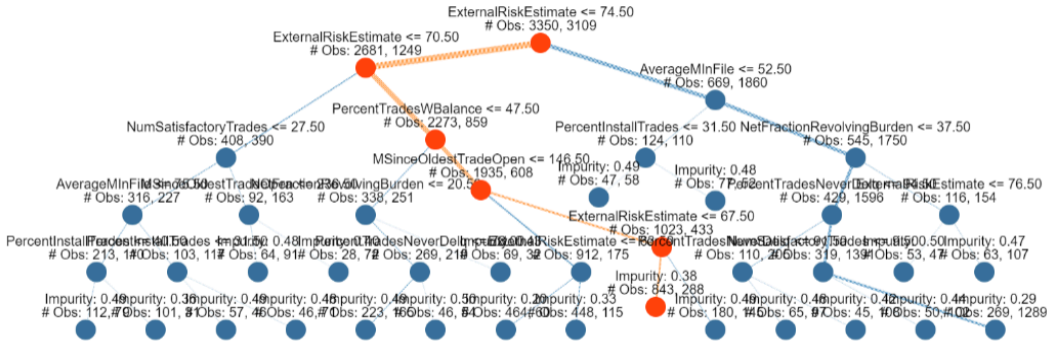
Si analizamos los nodos hoja, podemos observar que no existe ninguno que presente conjuntos completamente heterogéneos, dificultando así la fiabilidad del modelo. Aún así, si existen ciertos nodos que presentan una cantidad de individuos de una clase mucho mayor que de la otra, como es el caso del último nodo, con una proporción de individuos del 17% y 83%. Si algún individuo cae en ese nodo, podemos tener una gran seguridad de que será bien clasificado, mientras que si pertenece a un nodo con una proporción cercana al 50-50, como es el caso del sexto nodo hoja (46% y 54%), no podemos ofrecer una clasificación fiable.

Pese a que una profundidad de 5 pueda no parecer muy elevada, podemos observar como el árbol obtenido es bastante complejo, haciendo que sea difícil de interpretar. Debido a esto, una opción posible sería reducir la profundidad máxima. De esta forma podríamos perder capacidad predictora, pero favoreceríamos la interpretabilidad. Depende de cual sea el aspecto al que demos más importancia y la finalidad del modelo, que podemos considerar esta opción o descartarla completamente.

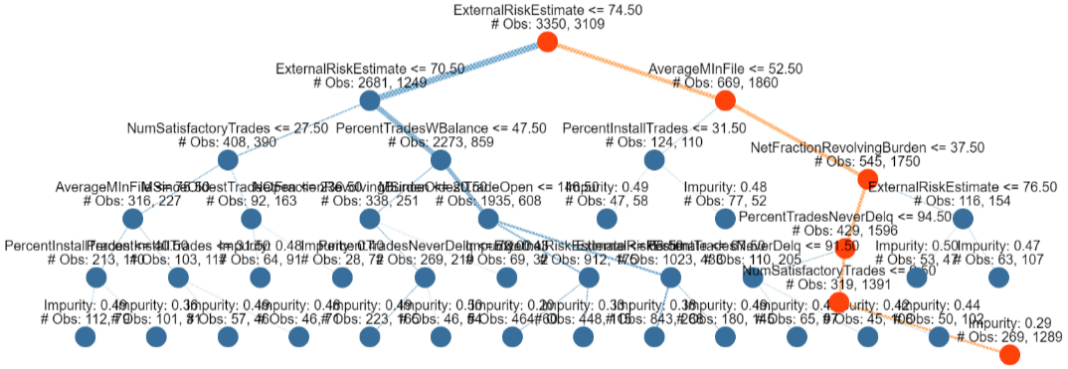
4.1.2. Interpretación local

En un árbol de decisión, es posible comprobar el razonamiento seguido por el modelo para llegar a la respuesta obtenida.

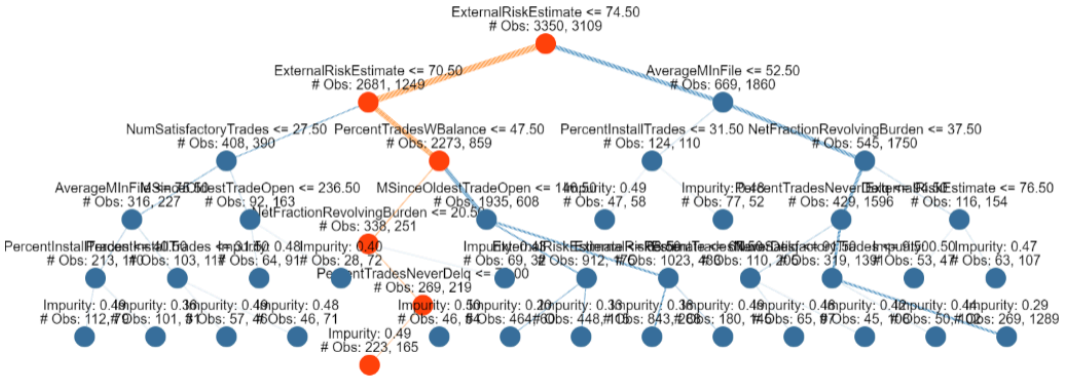
En la siguiente figura se muestra el camino del árbol seguido ante un ejemplo que ha sido clasificado correctamente como ‘Bad’, con una confianza de 0.745. Se observa que todos los nodos seguidos tienen un buen grado de acierto frente a los datos de entrenamiento.

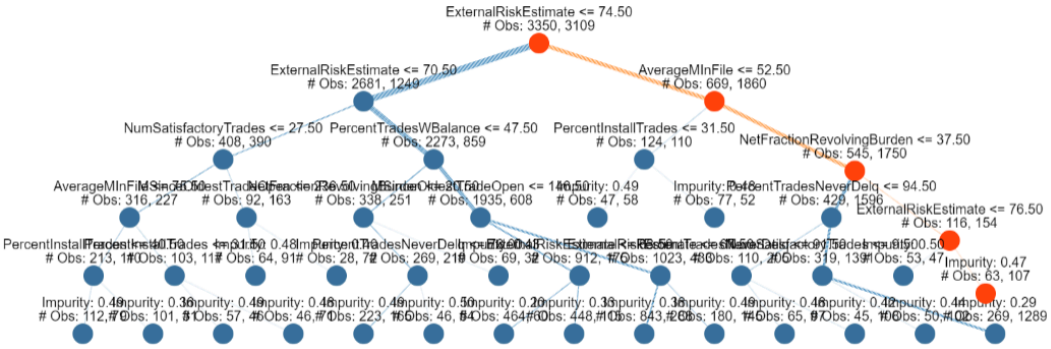


De forma similar, en la siguiente figura se muestra una clasificación correcta del valor ‘Good’ con una confianza todavía mayor, de 0.827.



Este método para interpretar los resultados es especialmente útil a la hora de comprobar por qué se ha equivocado el modelo en sus predicciones. A modo de ejemplo, a continuación se muestra un caso de falso positivo y otro de falso negativo.





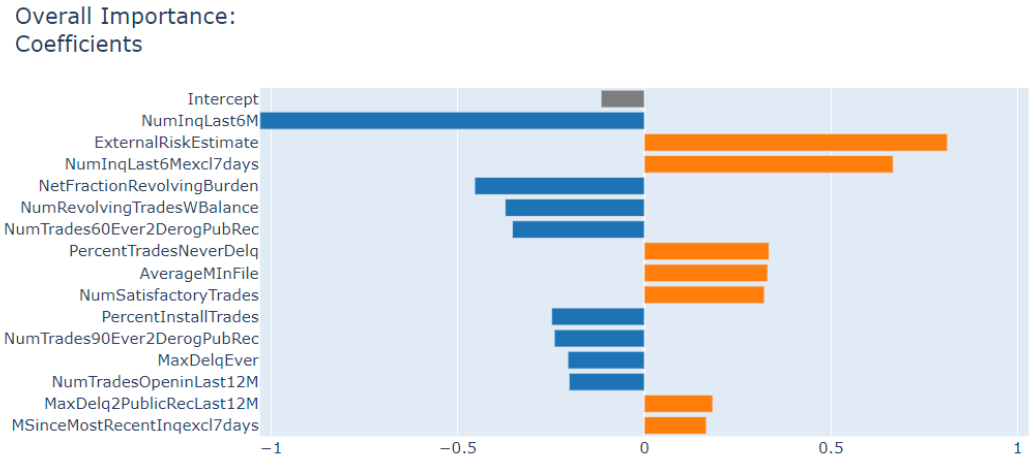
En primer lugar, cabe destacar que la confianza del modelo para estas predicciones es significativamente menor a la de las predicciones verdaderas (PrScore=0.575 para el primer ejemplo y PrScore=0.629 para el segundo). Además, ambas predicciones son incorrectas por motivos similares: La existencia de al menos una regla que no es suficientemente discriminante.

En el caso del falso positivo, la hoja a la que se llega ha tenido 223 aciertos frente a 165 fallos durante el entrenamiento, una proporción muy mejorable. Similarmente, en el falso negativo se llega a una hoja en la que se han observado 107 aciertos frente a 63 fallos.

Estos errores pueden ser solucionados aumentando la profundidad del árbol y con ello el número de reglas, para que el modelo se adapte mejor al conjunto de datos, pero como se ha visto en el apartado de metodología, esto no es recomendable ya que aumentaría significativamente el sobreajuste del modelo.

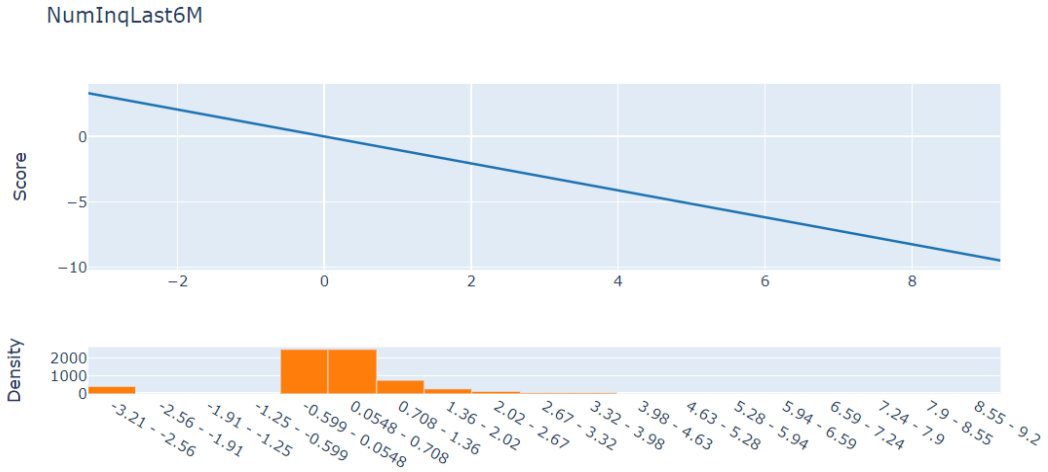
4.2. Modelo lineal

4.2.1. Interpretación global



Para conocer el impacto de cada una de las variables en los resultados se analizan sus coeficientes de importancia. Se puede observar que distintas variables pueden contribuir tanto positivamente como negativamente al resultado. Cabe destacar el alto valor negativo de la variable *NumInqLast6M*. Cualquier registro con un valor alto de esta variable será probablemente clasificado negativamente (clase “malo”). Lo contrario ocurre con variables como *ExternalRiskEstimate* y *NumInqLast6Mexcl7days*. Es interesante observar que las variables *NumInqLast6M* y *NumInqLast6Mexcl7days* mencionadas tienen una correlación de 0.99 y mantienen una relación casi lineal, por lo tanto, habitualmente, el coeficiente positivo de la segunda variable contrarrestará parcialmente al de la primera como se verá en las figuras del siguiente apartado. No parece haber variables con tan poca importancia como para ser consideradas despreciables.

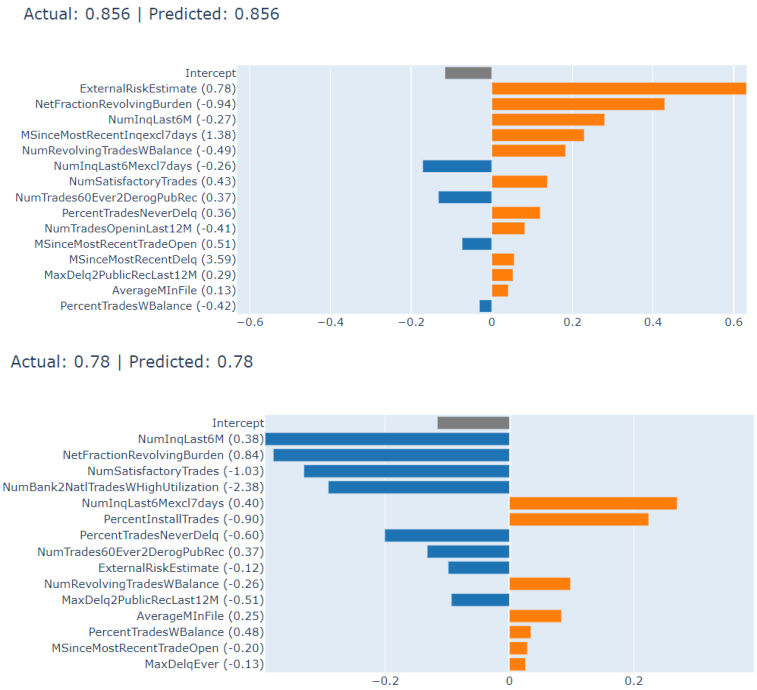
Es posible estudiar la influencia de las variables en la predicción en función de su valor. Observamos a modo de ejemplo la variable *NumInqLast6M*.



Por la naturaleza del modelo, el Score mantiene una relación lineal con el valor, trazando una recta con la inclinación correspondiente a su coeficiente de importancia, en este caso cercano a -1.

4.2.2. Interpretación local

Se puede calcular el aporte de cada una de las variables en la predicción de cada instancia multiplicando su valor por su coeficiente de importancia. A continuación se muestran dos ejemplos clasificados correctamente con un alto grado de confianza, siendo el primero un *true positive* y el segundo un *true negative*.

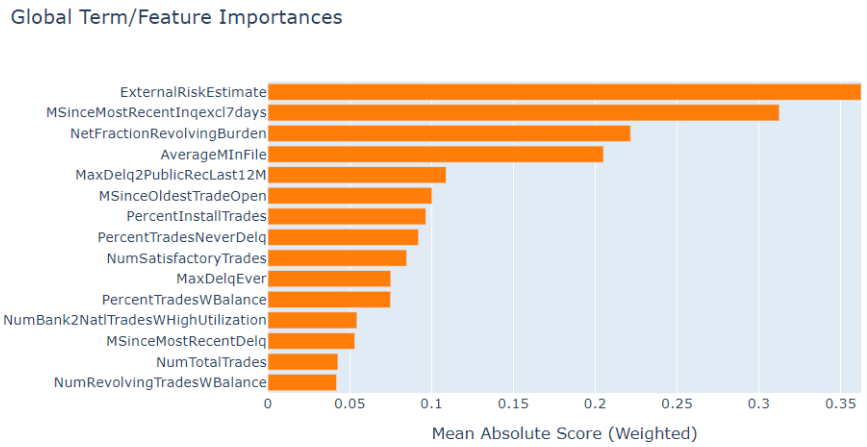


Las variables que más contribuyen al resultado son en muchas ocasiones las de coeficiente de importancia más alto observadas en la interpretación global. También es interesante ver que un valor negativo tiene un impacto positivo en la confianza de la clasificación para variables con coeficiente de importancia negativo y viceversa.

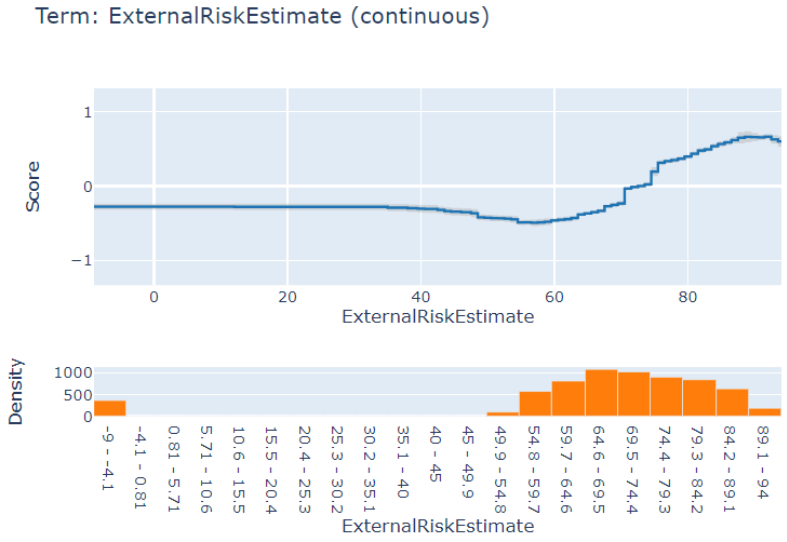
4.3. GAM

4.3.1. Interpretación global

Para obtener una interpretación global usaremos la función *explain_global*. En la sección Summary podemos observar las variables ordenadas según las Feature Importances, medidas utilizando Mean Absolute Score. De esta forma podemos hacernos una idea de las variables que más impacto tienen en las decisiones. Podemos observar cómo External Risk Estimate es la variable con mayor impacto, coincidiendo con los resultados obtenidos con los otros clasificadores, reforzando la importancia de esa variable. Otras con gran importancia son MSinceMostRecesntInquexcl7days, NetFractionRevolvingBurden y AverageMInFile. A partir de aquí, podemos observar una gran disminución en el feature importance de las variables, indicando una disminución en su importancia.

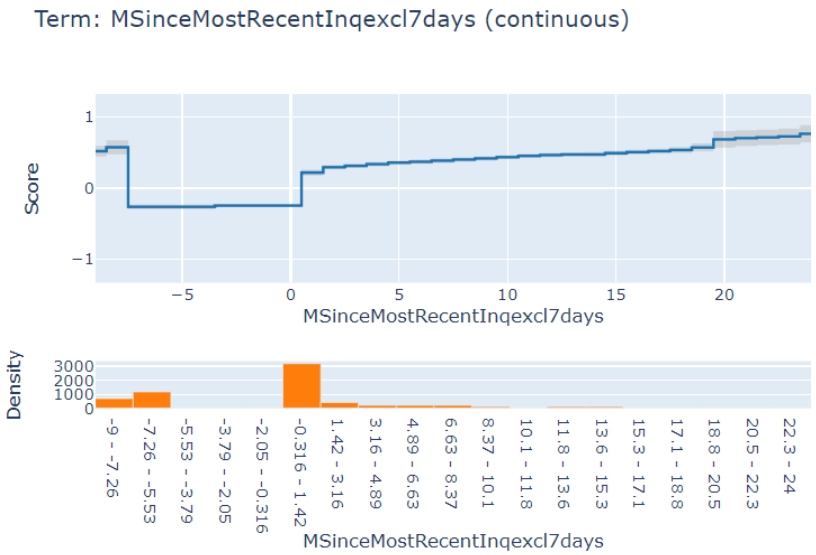


Si analizamos algunas de las variables podemos sacar varias conclusiones. Comenzaremos con la de mayor feature importance: ExternalRiskEstimate

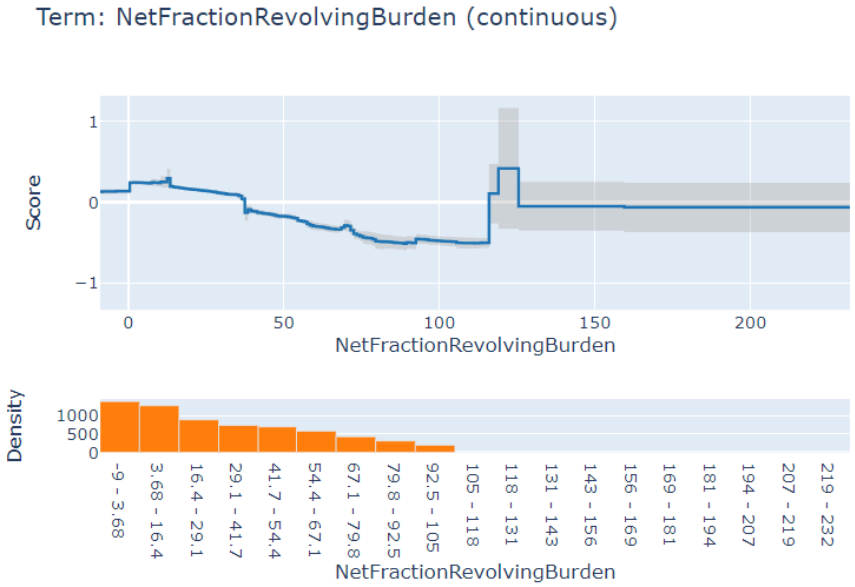


Podemos observar como ExternalRiskEstimate es muy efectiva para clasificar a los individuos. Entre -9 a 35 muestra un valor negativo constante y hasta 45 solo cambia ligeramente la pendiente. Esta parte de la gráfica coincide con el intervalo de menor densidad, por lo que ExternalRiskEstimate no es muy relevante a la hora de clasificar a un individuo cuando este toma valores dentro de este intervalo. A partir de 45, la gráfica sufre un cambio importante, coincidiendo con el intervalo de alta densidad, el cual se mantiene

hasta el valor máximo de ExternalRiskEstimate. Observando la curva podemos ver que la gráfica decrece hasta alcanzar un mínimo en 56, donde empieza a crecer, hasta alcanzar el valor 0 para valores de ExternalRiskEstimate de 70.5. Debido a esto y a las casi inexistentes barras de error, podemos predecir con bastante seguridad que para valores de ExternalRiskEstimate menores de 70.5, la clase de la variable objetivo del individuo será “malo”, clase que corresponde a los valores negativos de Score. Una vez la gráfica llega a cero en valores de ExternalRiskEstimate de 70.5, esta se mantiene prácticamente constante hasta 74.5. Para los individuos para los cuales ExternalRiskEstimate tome valores dentro de este intervalo o cercanos, podemos observar que esta variable no nos sirve para explicar la decisión que toma el algoritmo. En este intervalo serán otras variables las que sirvan para entender las decisiones tomadas por el modelo. En cambio, a partir de 74.5, la gráfica vuelve a crecer, hasta estabilizarse y decrecer ligeramente para los últimos valores de ExternalRiskEstimate. Debido a este comportamiento, podemos explicar que valores mayores de 74.5 para ExternalRiskEstimate provocan que la clase de la variable objetivo del individuo sea “bueno”, correspondiente a valores positivos de Score. Además, aunque en este intervalo las barras de error son ligeramente mayores, estas siguen siendo bastante bajas, reforzando así las interpretaciones obtenidas al analizar esta variable. Tras este análisis llegamos a la conclusión de que, para valores mayores de 45 (la inmensa mayoría) podemos interpretar lo siguiente: un ExternalRiskEstimate de entre 45 y 70.5 provoca un RiskPerformance “malo”, a partir de 74.5 un ExternalRiskEstimate “bueno” y entre 70.5 y 74.5 o valores cercanos no nos otorga información de la clase a la que pertenecen los individuos. Esto explica a su vez por qué esta variable es la más importante para tomar decisiones. Para el intervalo de RiskPerformance entre 70.5 y 74.5 y valores cercanos, analizar otras variables sería interesante. Para ello interpretaremos por encima las siguientes dos variables más importantes: MSinceMostRecentInqexcl7days y NetFractionRevolvingBurden

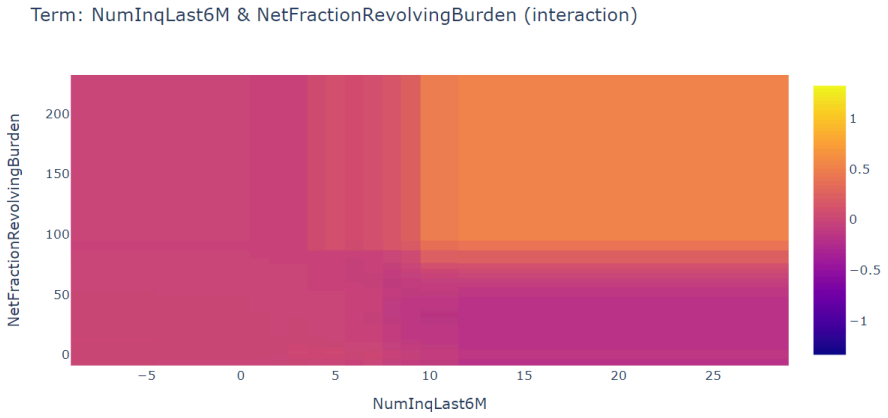


En el caso de MSinceMostRecentInqexcl7days, podemos observar que las distribuciones de los individuos están bastante desbalanceadas, aproximadamente la mitad se encuentran dentro del intervalo de [-0.316, 1.42]. Aun así, en este intervalo hay un cambio brusco en la predicción justo en el punto 0.5. Para valores de menores que este, la variable ayuda a hacer predicciones de clase “malo”, mientras que para mayores,influye en la clasificación de los individuos como “buenos”. Lo mismo, pero a la inversa, sucede en el intervalo también altamente poblado [-9, 5.53].



Comentando por encima la variable NetFractionRevolvingBurden, podemos observar como sus barras de error son bajas en el intervalo altamente poblado [-9, 105] y altas en el bajamente poblado a partir de 105. Por lo que esta variable sólo nos será útil para entender las decisiones tomadas para valores dentro de dicho intervalo, el cual presenta valores positivos (clase “bueno”) entre [-9, 37.5] y negativos (clase “malo”) entre [37.5, 105].

En cuanto a las interacciones, como hemos comentado previamente al hablar de los hiperparámetros seleccionados, el modelo ha sido creado con 10 interacciones. Al observar sus gráficos podemos ver que la mayoría de los mapas de calor de dichas interacciones no muestran ninguna información relevante, ya que muestra una distribución prácticamente uniforme de valores cercanos a 0. Aún así hay alguno que muestra información interpretable. Este es el caso de la interacción entre NumInqLast6M y NetFractionRevolvingBurden.

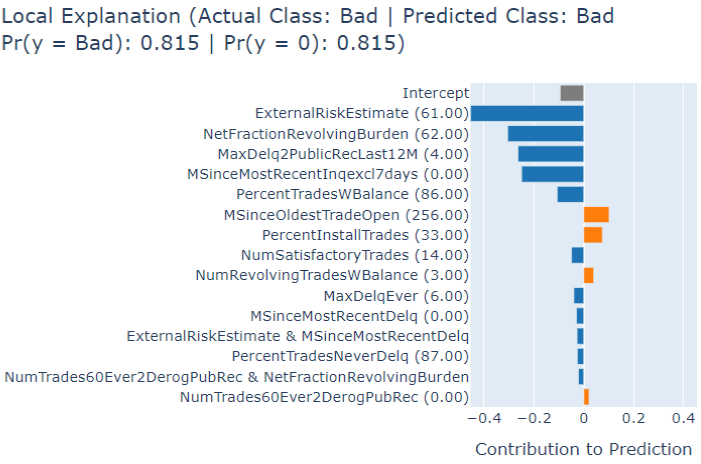


Al analizar este mapa de calor podemos observar dos patrones distintos. En primer lugar, podemos observar que cuando el Número de preguntas en los últimos 6 meses (NumInqLast6M) ha sido menor que 8 (aproximadamente) el Net Fraction Revolving Burden no tiene prácticamente impacto en la clasificación de los individuos. Simplemente los valores son ligeramente más negativos cuanto más aumenta el Net Fraction Revolving Burden (en la imagen es prácticamente inapreciable). En cambio el comportamiento cambia radicalmente para valores de NumInqLast6M mayores que 8 (aproximadamente). En este caso se puede observar como el valor de Net Fraction Revolving Burden influye claramente en la clasificación de los individuos. Se puede apreciar cómo para valores cercanos a cero de Net Fraction Revolving Burden, el

valor del mapa de calor es negativo, indicando individuos de clase “malo”. Pero según aumenta el Net Fraction Revolving Burden, el valor del mapa de calor crece, hasta convertirse en positivo a partir de 63.5, indicando mayor probabilidad de individuos “buenos”. Es decir, para valores de NumInqLast6M mayores que 8, la probabilidad de que un individuo sea “bueno” aumenta según aumenta el Net Fraction Revolving Burden. En cambio para valores de NumInqLast6M menores que 8 la probabilidad de que un individuo sea “malo” aumenta según aumenta el Net Fraction Revolving Burden, aunque tan ligeramente que puede ser despreciable y asumir que este no tiene impacto real en la clasificación.

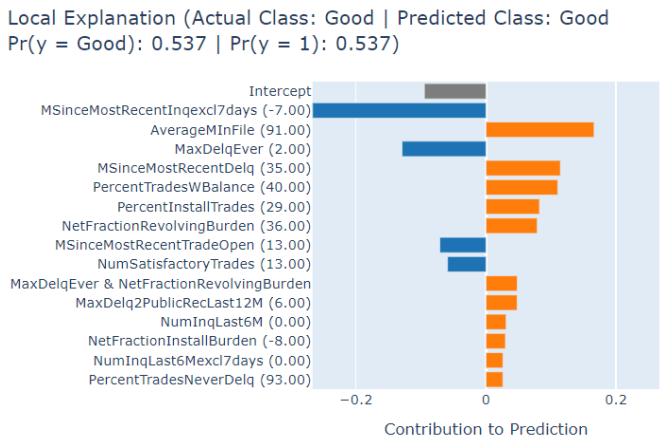
4.3.2. Interpretación local

Para la interpretación global usaremos la función *explain_local*. Primero interpretaremos a un individuo clasificado correctamente como “malo” con una probabilidad alta (0.815). Este es el individuo 1 del conjunto de test.



Podemos observar que la variable más importante es ExternalRiskEstimate seguida por NetfractionRevolvingBurden. Observando los valores que estas toman (61 y 62 respectivamente), y consultando las gráficas expuestas para estas variables en la sección de interpretación global, podemos observar que ambos valores corresponden a valores negativos del Score de las gráficas, el correspondiente a la clase “malo”. Esto explica por qué la decisión es tomada con tanta seguridad y nos permite interpretar que dichos valores de estas dos variables tienen un gran impacto en la clasificación del individuo. Además se observa en la gráfica de explicación local como las variables que más impacto tienen en la decisión, todas contribuyen con una predicción negativa, es decir, predicen la clase “malo”, lo cual explica la alta probabilidad de esta clasificación (0.815).

A continuación analizaremos al individuo 12 del conjunto de test, el cual ha sido correctamente clasificado como “bueno” pero con probabilidad muy baja (0.537).



Podemos observar que, esta vez, ExternalRiskEstimate no se encuentra entre las variables con más impacto. Esto se debe a que esta variable toma el valor 74, el cual se encuentra dentro del intervalo [70.5, 74.5], el cual mostramos en la sección de interpretación global que poseía valores de Score muy cercanos a 0, indicando que esta no servía para para explicar la clasificación. El valor tomado para la variable MSinceMostRecentInqexcl7days es de -7, el cual corresponde a la clase “malo”. Pero este individuo ha sido clasificado como “bueno”. Esto se puede entender al analizar la gráfica de la interpretación local. Aunque la variable con mayor impacto tenga una predicción negativa (clase “malo”), la suma de las contribuciones de cada variable da un número positivo (clase “bueno”), explicando por qué esta es la decisión final. Además, el que haya tanto contribuciones negativas y positivas y que su suma total sea cercana a 0, explica el por qué de la baja probabilidad de la clasificación (0.537), la cuál es casi prácticamente aleatoria.

5. Discusión

Debido a la interpretabilidad inherente a los árboles de decisión, ha sido bastante sencillo entender las decisiones que toma el modelo, tanto a nivel global como local. Aún así, esto trae consigo un aspecto negativo, el cual es la baja capacidad predictora de los árboles. Normalmente, la explicabilidad y capacidad predictiva de un modelo suelen ser inversamente proporcionales. Debido a ello, un punto muy importante a la hora de decidir qué modelo utilizar para resolver nuestro problema es saber el equilibrio que deseamos obtener entre estos dos aspectos. Como hemos indicado en el apartado de resultados, este modelo, sin realizar una búsqueda exhaustiva de hiperparámetros, ha obtenido una *accuracy* del 73% en el conjunto de entrenamiento y del 69% en el conjunto de prueba. Estos valores no son muy altos, contrastando completamente con los altos niveles de explicabilidad del modelo. Debido a ello, consideramos que sólo se debería plantear usar árboles de decisión para resolver nuestro problema si la principal prioridad es entender por qué se toman las decisiones y no tanto el resultado del modelo. Esto puede concordar con el problema que abordamos, ya que se basa en decidir si una persona tiene un alto o bajo riesgo. Si la decisión que tomamos puede influenciar la vida de una persona, es importante poder explicar por qué la hemos tomado. Por el contrario, tampoco sería correcto basarnos en los resultados producidos por un modelo con baja capacidad predictora.

Como sabemos, los árboles de decisión presentan ciertos problemas como la dificultad para encontrar relaciones aditivas entre las variables o que pequeños cambios en las variables pueden provocar grandes variaciones en las predicciones. Actualmente, no sabemos si estos problemas tienen un gran impacto en el modelo que hemos creado. Implementar distintos modelos que no presenten estos inconvenientes pueden ayudarnos a mejorar los resultados en caso de que estos tengan un gran impacto en el modelo final.

El árbol obtenido no es óptimo, esto puede influenciar el rendimiento del modelo. La utilización de un árbol óptimo puede llegar a igualar los resultados obtenidos por conjuntos de árboles, sin perder la explicabilidad que estos últimos no ofrecen. Debido a ello, podría ser interesante estudiar la implementación de un árbol óptimo para observar si mejora el rendimiento del modelo.

Si transformamos el árbol en reglas se podría realizar alguna simplificación. Un ejemplo es el primer subárbol de la izquierda. Esto es debido a que tanto en el nodo raíz como en el primer nodo del subárbol se evalúa la misma variable: ExternalRiskEstimate. Esto implica que juntando ambas reglas podríamos realizar una simplificación.

Por su parte, el modelo de regresión logística obtiene resultados algo mejores que el de árboles de decisión y permite explicar las clasificaciones realizadas a través de los coeficientes de importancia de las variables implicadas y la aportación de cada una de ellas al resultado final.

La diferencia de escalas entre las distintas variables puede ocasionar irregularidades en el entrenamiento, por lo que son estandarizadas. Esto hace que se pierdan las unidades de cada variable perdiendo interpretabilidad. Además, el modelo está limitado a fronteras lineales en sus variables, lo cual no ocurre habitualmente con datos reales. Se puede realizar una expansión de variables para superar este problema, pero una vez más, se pierde interpretabilidad en cuanto al significado de las variables resultantes y las unidades en las que se expresan.

En cuanto al modelo GAM explainable boosting machine hemos decidido no realizar una estandarización de los datos ya que en la regresión logística observamos que los resultados no aumentaron sustancialmente y en cambio se dificultó ampliamente la interpretabilidad. Aunque la poca mejoría en los resultados no tiene por qué trasladarse a este modelo, decidimos optar por la mejor interpretabilidad ya que era el objetivo principal de esta práctica.

Al final de la práctica hemos decidido implementar un modelo de caja negra para comparar su rendimiento con los modelos interpretables. Para ello hemos implementado un Random Forest. Hemos realizado un estudio de hiperparámetros,esta vez con el algoritmo de RandomSearch. Una vez entrenado y evaluado el modelo, nos ha sorprendido ver que el rendimiento ofrecido por el EBM ha sido ligeramente mayor que el del Random Forest. Esto ha demostrado el buen rendimiento del EBM a pesar de ofrecer gran interpretabilidad. Este gran balance entre rendimiento e interpretabilidad convierte al EBM en un modelo muy sólido para problemas en los que queremos modelos con buenos resultados y cuyas decisiones sean interpretables.

6. Conclusiones

En esta sección se comentarán las conclusiones obtenidas al comparar los distintos modelos respecto a su interpretabilidad y rendimiento. En cuanto al árbol de decisión, la evaluación del modelo muestra un amplio rango de mejora en las métricas de confusión, sin embargo, este no deja de ser un modelo atractivo por su facilidad de uso, robustez ante datos categóricos y no normalizados, y sobre todo, su interpretabilidad inherente, que permite observar fácilmente tanto las reglas que conforman el modelo como el razonamiento seguido para cada una de las predicciones realizadas. La regresión logística es una buena opción cuando los datos tienen relaciones lineales y es fácilmente interpretable, pero su funcionalidad se ve limitada ante fronteras de decisión no lineales, que no se pueden representar sin una pérdida de interpretabilidad. En cuanto al explainable boosting machine, muestra un gran compromiso entre interpretabilidad y rendimiento. Sus resultados son bastante satisfactorios, superando tanto a los otros dos modelos de caja negra como al random forest. Es probable que el random forest consiga superar al EBM con un estudio más profundo de hiperparámetros. Un posible trabajo futuro sería realizar un estudio más exhaustivo de sus hiperparámetros o probar otros tipos de modelos de caja negra para hacer una comparación más amplia de los resultados.

En cuanto a las conclusiones finales, consideramos que el modelo que mejor se adecúa a este problema es el Explainable Boosting Machine (GAM), debido a la gran relación que muestra entre explicabilidad y rendimiento para nuestro problema. Hemos llegado a obtener los mejores resultados (74.89% de accuracy en test) y, como hemos observado, sus resultados son fácilmente interpretables tanto global como localmente. En cambio, la regresión logística ha ofrecido peores resultados que el EBM y ha sido más difícil de interpretar, por lo que es el modelo que consideramos que menos se adecúa al problema. Por último, pese a que el árbol de decisión ha sido el modelo con peor rendimiento, es el que nos ofrece una interpretación más sencilla y amplia. Debido a ello, aunque consideramos al EBM como la mejor opción, un árbol podría ser interesante si consideramos la interpretabilidad nuestro objetivo principal a expensas del rendimiento y no quedamos satisfechos con la ofrecida por el EBM. A pesar de ello, esto no es el caso para nuestro problema.