Dando explicación a modelos de caja negra

Antonio Peláez Moreno
dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla
Sevilla, España
antpelmor@alum.us.es

Francisco Fernández Mota
dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla
Sevilla, España
frafermot@alum.us.es

Repositorio: https://github.com/frafermot/IA-2022-2023-Interpretabilidad.git

El presente proyecto trata sobre modelos de caja negra y sus respectivas explicaciones. Para ello, se ha implementado un método LIME, se han elegido dos conjuntos de datos, y se han entrenado dos modelos de caja negra para cada uno. Además, se han diseñado una serie de métricas, que han sido aplicadas a estos modelos.

Como conlusión, es crucial entrenar los modelos con un gran número de muestras y considerar múltiples alternativas. Los resultados y pruebas varían según la aleatoriedad del sistema, las métricas seleccionadas y el método de entrenamiento. Para mejorar, es esencial profundizar en el conocimiento de las librerías, dominar Jupyter Notebook, solucionar errores y utilizar Python de manera más avanzada.

Palabras Clave—caja negra, métricas, LIME,…

I. INTRODUCCIÓN

La Inteligencia Artificial está a la órden del día. Es raro encontrar alguna empresa importante que no esté implementando de alguna forma estas nuevas tecnologías en su ámbito de trabajo. Sin embargo, este rápido avance ha probocado que, en ocasiones, los sistemas de IA actuales sean muy diferentes a los primeros que se idearon. No solo en eficiencia, rendimiento o potencia. En el caso de los modelos predictivos, este aumento de posibilidades y caracteristicas trae consigo un aumento de complejidad en los sistemas y modelos que se implementan. Es de aquí de donde nacen las Inteligencias Artificiales Explicables (XAI en inglés) [4]

Para dar una explicación a estos modelos, denominados de caja negra, surgen técnicas y métodos que nos ayudan a comprender mejor el funcinoamiento de los mismos. Nosotros nos centraremos en las técnicas agnósticas al modelo, es decir, aquellas que se pueden aplicar sea cual sea el modelo a explicar. En concreto el método LIME. Este método nos permite obtener un vector de pesos que indican la relevancia de cada atributo en la predicción realizada [5].

En este trabajo, se pretende implementar el método LIME para dar explicaciones a las predicciones dadas por 4 modelos de caja negra. Para ello, hemos seleccionado dos conjuntos de datos de la web UCI. Para cada uno de ellos, se ha entrenado un modelo. En concreto, nos hemos decantado por Random Forest y XGBoost. Además se han definido un conjunto de métricas para medir el cumplimiento de caracteristicas propias del método LIME.

El presente documento se estructura como sigue: Después de la presente Introducción, encontraremos unos Preliminares en los que se definen algunos conceptos y técnicas empleadas. A continuación de desarrolla la metodología seguida para implementar y diseñar todos los métodos y funciones, seguido de un análisis de los Resultados obtenidos. Por último, encontramos una Conclusion y la Bibliografía del proyecto.

II. PRELIMINARES

A. Métodos empleados

Para desarrollar este trabajo, se han empleado una serie de técnicas descritas a continuación:

- Tarea de Regresión: Es uno de los tipos de tareas resueltas por los modelos de aprendizaje automático. El resultado que se busca es un número, dentro de un conjunto infinito de posibles resultados [1]. En nuestro caso, se ha usado en el ámbito de aprendizaje supervisado.
- Tarea de Clasificación: Es otro de los tipos de tareas resueltas por modelos de aprendizaje automático. En su caso, el resultado es una clase, es decir, un valor definido de entre un número limitado de ellos [1]. En concreto, hemos usado una clasificación multiclase.
- Random Forest: Es un modelo de Inteligencia Artificial basado en la generación de árboles de decisión que se combinados. Funciona de forma que cada árbol se entrena con diferentes muestras, combinando después los resultados obtenidos para devolver una respuesta fiable [2]. En nuestro caso, hemos usado un Random Forest Regressor, que resuelve una tarea de regresión.
- XGBoost: El algoritmo XGBoost se basa en árboles de decisión. Cada árbol aprende a partir de árboles anteriores. Sirve para tareas de regresión y de clasificación [3].

III. METODOLOGÍA

Para la realización de este trabajo, hemos implementado el método LIME. Este método se basa en un modelo subrogado, en nuestro caso Regresión Ridge.

Los pasos que sigue el método son:

1) Generar muestras perturbadas: A partir de la muestra original, se genera una muestra perturbada. Esta perturbación consiste en elegir k atributos aleatoriamente y variarlos en un rango específico. En nuestro caso, el rango variaba entre el valor mínimo y máximo del atributo. Concretamente:

$$[v - min, max - v] \quad (1)$$

siendo v el valor del atributo

2) Calcular su representación: Teniendo la muestra perturbada, se calcula su representación, que consiste en un array de unos y ceros, de forma que el 1 representa la sí perturbación del atributo y el 0 lo contrario.

3) Calcular la distancia entre muestras: Como función de distancia entre la muestra original y la pertubada se ha utilizado la función coseno.

4) Sacar predicciones: Usando el modelo a explicar, se calculan las predicciones de las muestras perturbadas.

5) Calcular los pesos: Mediante el modelo Ridge entrenado con las predicciones obtendias, las representaciones anteriores y usando las distancias como ponderación, se calculan los pesos de los atributos. Estos pesos forman un vector de explicación. Cuanto más cercano a 0 sea el valor del peso, menos relevancia supone dicho atributo para la predicción. De manera contraria, cuanto más alto o elvado sea el peso, el atributo influirá positiva o negativamente a la predicción.

Podemos visualizar el método LIME con mayor claridad en el siguiente pseudocódigo:

Método LIME
Entrada:
- Número N de permutaciones a realizer
- Modelo f a explicar
- Número k de atributos a modificar
- Diccionario de rangos máximo y mínimo

Salidas:
- Vector de pesos

Algoritmo:
1. R <= {} representaciones
   W <= {} distancias
   Y' <= {} predicciones
2. For 1 to N do
   a. Seleccionar k atributos aleatoriamente
   b. x' = perturbar la muestra
   c. w = calcular distancia entre muestras
   d. r = calcular representación
   e. R < = R ∪ r
   f. W <= W ∪ w
   g. y' = predicción de modelo a explicar
   h. Y' = Y' ∪ y'
3. G = modelo entrenado con R, Y' y W
4. Devolver coeficientes de G

Pseudocódigo 1: Método LIME

Por otro lado, hemos entrenado dos modelos para cada conjunto de datos: 2 modelos Random Forest y 2 modelos XGBoost. Por un lado, para el primer conjunto de datos, cuyo objetivo es predecir un valor numérico, es decir, una tarea de regresión, se ha usado un RandomForestRegressor y, por su parte, el modelo XGBoost se ha configurado para que su objetivo sea la regresión, y se use como métrica de evaluación el error cuadrático medio.

Por otro lado, para el segundo conjunto de datos, cuyo objetivo es la clasificación multiclase, se ha usado un RandomForestClassifier, que clasifica entre las 10 clases posibles. Además, se ha configurado el modelo XGBoost para que su objetivo sea también la clasificación.

IV. RESULTADOS

Los resultados obtenidos tras aplicar las diferentes métricas a los 4 modelos entrenados se resumen en la siguiente **¡Error! No se encuentra el origen de la referencia.**.

Analizando los resultados obtenidos y comparándolos tras varias ejecuciones, cabe destacar lo siguiente:

- La métrica que calcula la estabilidad para los modelos Random Forest devuelve True en alguna de las ejecuciones y False en otras, por lo que depende de las muestras seleccionadas para su comprobación.
- La métrica que calcula la coherencia para los modelos entrenados con el fichero .csv Pen presentan un número menor ya que los errores son menores, esto quiere decir que las predicciones se aproximan mucho a las predicciones obtenidas una vez se eliminan las características no importantes. A diferencia del .csv Bicis, que los errores son mayores, por lo que las predicciones varían mucho tras eliminar los atributos menos importantes.
- La completitud para el modelo XGBoost del .csv Bicis es un valor negativo ya que el error de las explicaciones también lo es, esto quiere decir que al calcular la media, se han tenido en cuenta muchos valores negativos debido a las explicaciones actualizadas, que eran superiores a las iniciales. Cabe destacar que la métrica de la completitud para los 4 modelos presenta valores muy próximos a 0, por lo que el error de explicación es significativamente menor respecto al error de predicción.
- La congruencia para los modelos entrenados con el .csv Bicis presentan un valor muy superior a lo modelos entrenados con el .csv Pen ya que esta métrica también depende de la coherencia es por eso que los resultados obtenidos en ambas métricas son muy diferentes comparando los ficheros .csv.

Tabla. Resultado de las métricas

Métrica	**Modelo**			
	Random Forest - Bicis	***XGBoost - Bicis***	***Random Forest - Pen***	***XGBoost – Pen***
Identidad	True	True	True	True
Separabilidad	True	True	True	True
Estabilidad	False	---	False	---
Selectividad	7.83776	97.9385	0.99953	0.875
Coherencia	17.08414	18.361757	0.8632812	1.40625
Completitud	7.74e-16	-0.427196	0.0015143	0.0028263
Congruencia	43.96637	44.496502	1.9143933	2.146718

V. CONCLUSIONES

Tras este proyecto de interpretabilidad de modelos de caja negra, podemos concluir los siguientes aspectos.

Todo modelo es necesario que sea entrenado con el mayor número posible de muestras, ya que es necesario tener en cuenta el mayor número posible de alternativas, haciendo más difícil el desarrollo de una Inteligencia Artificial.

Tras analizar los resultados y haber realizado numerosas pruebas con los ficheros y el código, concluimos que estos varían de forma considerable dependiendo de la aleatoriedad del sistema, aplicado a las métricas, estas devuelven valores diferentes según se hayan escogido unas muestras u otras, tanto para entrenar el modelo, como para interpretarlo y ponerlo a prueba. También los resultados dependen mucho del método de entrenamiento, es decir, si se ha entrenado con Random Forest, XGBoost, redes neuronales, etc.

Finalmente como ideas de mejora y trabajo futuro, consideramos muy importante tener el mayor conocimiento posible acerca de las librerías usadas para el desarrollo de este proyecto, así como del uso de Jupyter Notebook y de identificar y solucionar errores. Además de conocer el lenguaje Python con más perfección y usarlo de forma más compleja.

Finalmente, se dedica la última sección para indicar las conclusiones obtenidas del trabajo. Se puede dedicar un párrafo para realizar un resumen sucinto del trabajo, con los experimentos y resultados. Seguidamente, uno o dos párrafos con conclusiones. Se suele dedicar un párrafo final con ideas de mejora y trabajo futuro.

REFERENCIAS

[1] Jose Martínez Heras, "¿Clasificación o Regresión?", https://www.iartificial.net/clasificacion-o-regresion/

[2] Jose Martínez Heras, "Random Forest (Bosque Aleatorio): combinando árboles", https://www.iartificial.net/random-forest-bosque-aleatorio/

[3] ArcGIS Pro, "Cómo funciona el algoritmo XGBoost", https://pro.arcgis.com/es/pro-app/latest/tool-reference/geoai/how-xgboost-works.htm

[4] IBM, "¿Qué es la IA explicable?", https://www.ibm.com/es-es/watson/explainable-ai

[5] Cristina Gil Marínez, "Interpretación de predicciones de modelos black box", https://rpubs.com/Cristina_Gil/interpretacion_predicciones_modelos_bb#:~:text=El%20m%C3%A9todo%20Local%20Interpretable%20Model,predicci%C3%B3n%20de%20una%20observaci%C3%B3n%20individual