Inteligencia Artificial eXplicable (XAI)

Marta Caro Martínez

¿Es la Inteligencia Artificial útil hoy en día?

¿Qué es XAI?

- eXplainable Artificial Intelligence (XAI)

La XAI es el campo de la IA que tiene como objetivo principal proponer un conjunto de técnicas que consigan hacer más entendibles para los usuarios los modelos de inteligencia artificial

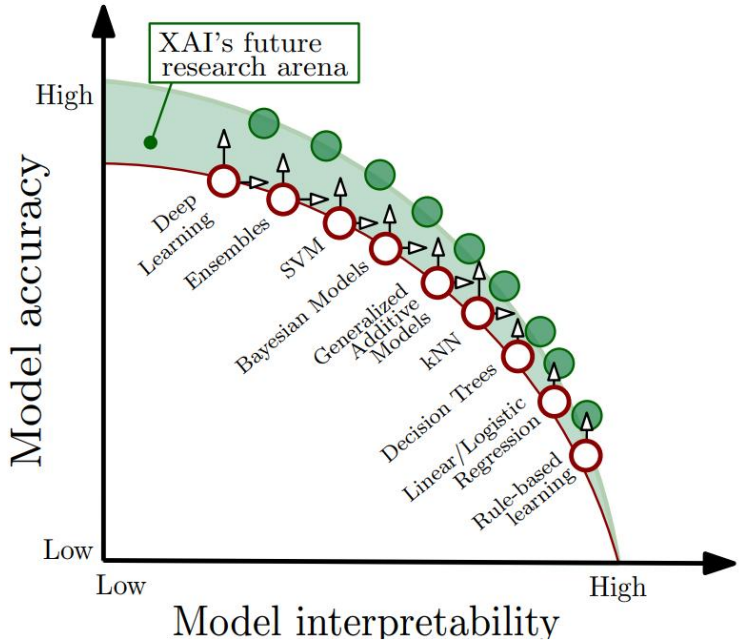
¿Qué es la explicabilidad?

Diferencias entre los conceptos:

- Transparencia
- Interpretabilidad
- **Explicabilidad**

Importancia de la XAI

- La XAI es actualmente un campo muy en auge y de importancia para el desarrollo de la IA
- Se pueden definir ciertas características que es necesario conocer para diseñar sistemas de explicaciones de calidad
- Ya existen explicadores que podemos usar y aplicar en modelos de IA



Fuente: Arrieta, A. B., et al. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible

Características de los sistemas de explicación

- Desde el punto de vista de la **motivación**
- Desde el punto de vista de las **restricciones/necesidades del problema de IA**
- Desde el punto de vista de los **atributos del explicador**
- Desde el punto de vista de **cómo se muestra la explicación**
- Desde el punto de vista de la **implementación**

Características de los sistemas de explicación

- Desde el punto de vista de la **motivación**
- Desde el punto de vista de las restricciones/necesidades del problema de IA
- Desde el punto de vista de los atributos del explicador
- Desde el punto de vista de cómo se muestra la explicación
- Desde el punto de vista de la implementación

Sistemas de explicación: Motivación

- Desde el punto de vista de los **usuarios** que usan el sistema
- Desde el punto de vista de los **objetivos** del sistema

Sistemas de explicación: Motivación

Desde el punto de vista de los **usuarios** que usan el sistema:

- Usuarios objetivo (único o grupo)
- Stakeholders
- Desarrolladores
- Organismos reguladores

Sistemas de explicación: Motivación

Desde el punto de vista de los **objetivos del sistema**:

- Efectividad
- Eficiencia
- Confianza
- Escrutinio
- Persuasión
- Satisfacción
- Transparencia
- Educación
- Debugging

Características de los sistemas de explicación

- Desde el punto de vista de la motivación
- Desde el punto de vista de las **restricciones/necesidades del problema de IA**
- Desde el punto de vista de los atributos del explicador
- Desde el punto de vista de cómo se muestra la explicación
- Desde el punto de vista de la implementación

Sistemas de explicación: problema a explicar

- Desde el punto de vista del **modelo de IA** que queremos explicar
- Desde el punto de vista de la **tarea de IA** que se quiere resolver
- Desde el punto de vista del **tipo de datos disponibles**

Características de los sistemas de explicación

- Desde el punto de vista de la motivación
- Desde el punto de vista de las restricciones/necesidades del problema de IA
- Desde el punto de vista de los **atributos del explicador**
- Desde el punto de vista de cómo se muestra la explicación
- Desde el punto de vista de la implementación

Sistemas de explicación: atributos

- **Scope**:

  Local Global Cohort

- **Portabilidad**:

  model-agnostic model-specific model class-specific

- **Concurrencia**:

  ante-hoc (white box) post-hoc (black-box)

Características de los sistemas de explicación

- Desde el punto de vista de la **motivación**
- Desde el punto de vista de las **restricciones/necesidades del problema de IA**
- Desde el punto de vista de los **atributos del explicador**
- Desde el punto de vista de **cómo se muestra la explicación**
- Desde el punto de vista de la **implementación**

Sistemas de explicación: presentación

- Desde el punto de vista del **tipo de output**:

  imágenes, texto, gráficas, sonido, realidad aumentada…

- Desde el punto de vista del **tipo de explicación** a mostrar**:**

  Factual, semi-factual, counterfactual

  Feature Importance

  Example-based

  Trace-based

  …

Sistemas de explicación: tipos de explicación

- **Factual**:

No se ha aceptado su préstamo porque no gana suficiente dinero

- **Semi-factual**:

Aunque tuvieras el doble de tu sueldo actual, te habrían denegado el préstamo

- **Counterfactual**:

Si solicitases una cantidad ligeramente inferior, te habrían aceptado

Sistemas de explicación: tipos de explicación

- **Feature Importance**:

No se ha concedido el préstamo porque su sueldo es bajo, lo cual contribuye al 60% en la no concesión del préstamo, y además, su contrato de trabajo es temporal, lo que contribuye en un 40%

Sistemas de explicación: tipos de explicación

- **Example-Based o Case-Based**:

No se ha aceptado su préstamo porque otra persona con su perfil que optó al préstamo pidió esta cantidad, y también se le rechazó

Sistemas de explicación: tipos de explicación

- **Trace-Based**:

La IA primero ha determinado que tu sueldo es bajo porque no llega a la media de sueldos de otras personas que sí pueden devolver el préstamo. Después ha determinado que su contrato de trabajo es temporal y, de acuerdo a sus datos anteriores, ha determinado que con este tipo de contrato un 56% de las personas no pueden devolver el préstamo.

Características de los sistemas de explicación

- Desde el punto de vista de la **motivación**
- Desde el punto de vista de las **restricciones/necesidades del problema de IA**
- Desde el punto de vista de los **atributos del explicador**
- Desde el punto de vista de **cómo se muestra la explicación**
- Desde el punto de vista de la **implementación**

Sistemas de explicación: implementación

- Desde el punto de vista del **explainer:**

Personalizado	Integrated Gradients
SHAP	GradCam
LIME	Nearest Neighbours
Anchors	Counterfactuals (imágenes)
DiCE	…

- Desde el punto de vista del **backend**:

  PyTorch, Sklearn, TensorFlow…

SHAP (SHapley Additive exPlanations)

- Está basado en **teoría de juegos cooperativos**, en la idea de Shapley
- Se calcula la **contribución de cada atributo** a la predicción
  - Se calculan todas las posibles permutaciones entre los distintos valores de los atributos
  - Para cada permutación (el valor modificado), se calculan **contribuciones marginales**:
    - se calcula la diferencia entre la predicción del modelo con ese valor y sin ese valor (se quita ese atributo)
  - **Valor Shapley de cada atributo: promedio ponderado**
    - se hace una media de las diferencias para ese atributo
    - los atributos con diferencias mayores → tendrán más relevancia
    - también damos más peso a los atributos que tienen más valores: el peso es el número de valores del atributo

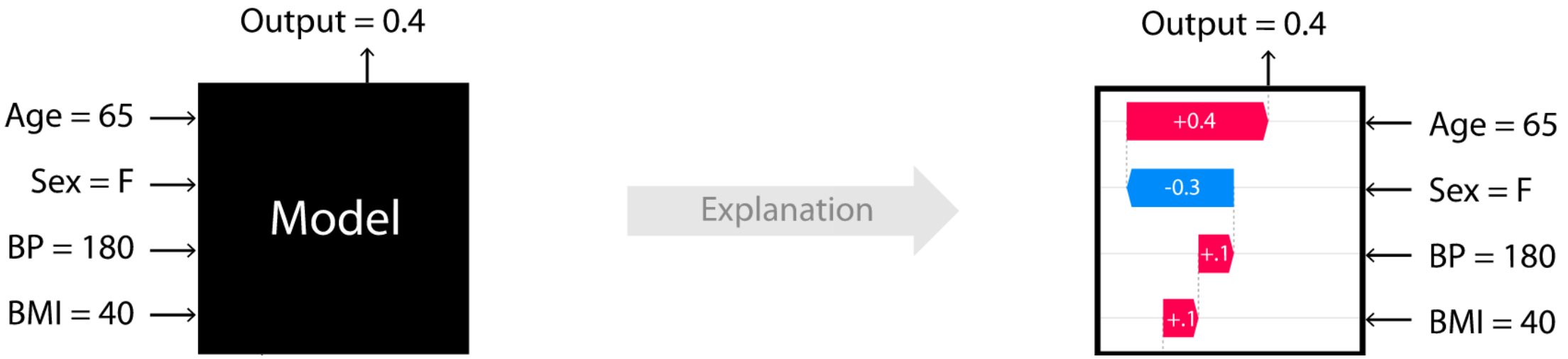
SHAP

- Devuelve **feature-based explanations**
- Puede devolver explicaciones **locales** o **globales** (estas miden el impacto promedio)
- Son **post-hoc**
- Pueden ser **model-specific o model-agnostic**
- En general muestran las explicaciones a través de **gráficos**
- Se puede aplicar sobre **machine learning** y **redes neuronales** para realizar tareas de **regresión y clasificación**.

SHAP

- Se pueden implementar con la librería **shap** para Python (se requiere usar **TensorFlow** para los modelos a explicar)
- Hay tres tipos de implementación:
  - **TreeExplainer**: model-specific para árboles de decisión
  - **DeepExplainer**: model-specific para Deep learning
  - **KernelExplainer**: model-agnostic

SHAP



Fuente: GitHub

LIME (Local Interpretable Model-agnostic Explanations)

- LIME manipula los datos de entrada y **determina cómo contribuye ese cambio en la predicción**
  - Se escoge la instancia a explicar y **se modifican sus atributos** aleatoriamente, obteniendo nuevas instancias
  - Se pasa cada cambio al modelo a explicar, y se obtienen nuevas predicciones
  - A cada instancia perturbada, se le asigna un peso:
    - Usando una métrica de similitud (normalmente distancia euclídea)
    - Es la comparación de las instancias perturbadas con las originales
    - Las instancias más parecidas a la original tendrán más peso: porque queremos enfocarnos en instancias parecidas a la instancia a explicar

LIME (Local Interpretable Model-agnostic Explanations)

- Se entrena un modelo de IA interpretable (generalmente **regresión lineal**):
  - Usamos las instancias perturbadas con sus predicciones y pesos
  - Función de coste a minimizar en la regresión → $\mathcal{L}(g) = \sum_i w(x_i)(f(x_i) - g(x_i))^2$
    - $f(x_i)$: predicción del modelo original
    - $g(x_i)$: predicción del modelo interpretable
    - $w(x_i)$: peso de cada instancia perturbada

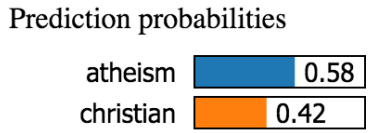
- **Explicación final: interpretación del modelo** (coeficientes de la regresión indican el peso de las características)

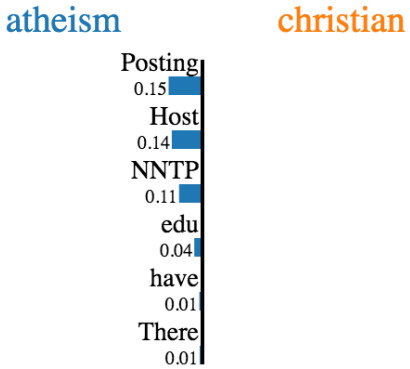
LIME

- Para explicar **modelos de aprendizaje automático y redes neuronales**
- Se puede aplicar sobre modelos que usen **imágenes, texto o datos tabulares**
- Se usa para explicar **clasificación y regresión**
- Es **post-hoc** y **model-agnostic** y produce explicaciones **locales**
- Devuelve **feature-based** explanations
- Tiene una librería específica (lime para Python)

LIME

Fuente: GitHub





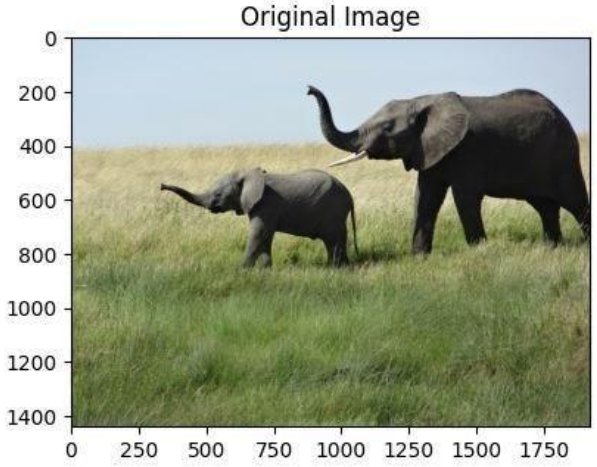
Texto →

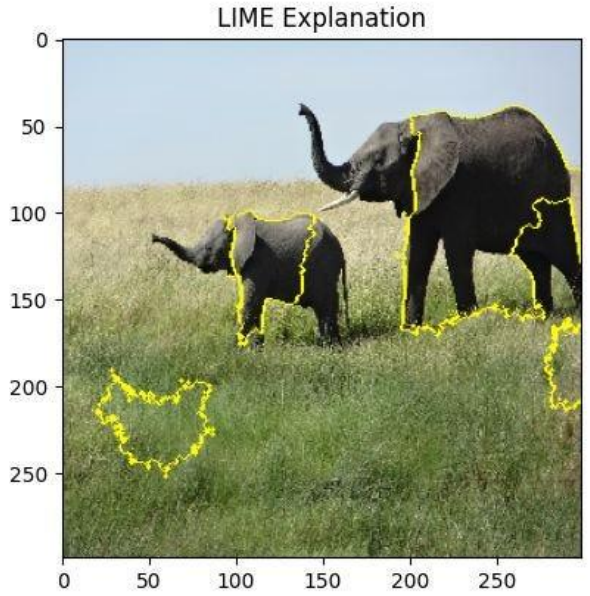
Text with highlighted words

From: johnchad@triton.unm.edu (jchadwic)
Subject: Another request for Darwin Fish
Organization: University of New Mexico, Albuquerque
Lines: 11
NNTP-Posting-Host: triton.unm.edu

Hello Gang,

There have been some notes recently asking where to obtain the DARWIN fish.
This is the same question I have and I have not seen an answer on the
net. If anyone has a contact please post on the net or email me.





← Imágenes

Anchors (High Precision Model-Agnostic Explanations)

- Generan reglas que cubren las condiciones de una predicción (se llaman **anchors**)
  - Estas reglas se pueden obtener de distintas formas (algoritmos de búsqueda para obtener las distintas combinaciones, reglas de asociación…)
- Ejemplo para autorización de préstamo:

Edad: 42

Ingresos: 2300€

Pagos mensuales: realizados

Predicción del préstamo: aprobado

Reglas:
Si realiza los pagos mensuales → aprobado
Si sus ingresos son > 2000 → Aprobado
Si la edad > 35 → Aprobado
….

Anchors (High Precision Model-Agnostic Explanations)

- Después, se comprueba la validez de las reglas → comprobando la precisión
  - Se modifican los atributos de las instancias del dataset

Edad: 37

Ingresos: 2100€

Pagos mensuales: realizados

Predicción del préstamo: aprobado

Edad: 34

Ingresos: 500€

Pagos mensuales: realizados

Predicción del préstamo: rechazado

- Evaluar la regla: si la regla sigue dando la misma predicción sobre las instancias modificadas, entonces es que es una buena regla
  - Si en la segunda instancia del ejemplo anterior hubiera sido “aprobado” y no “rechazado”, las reglas de la diapositiva anterior no funcionan → malas reglas

Anchors (High Precision Model-Agnostic Explanations)

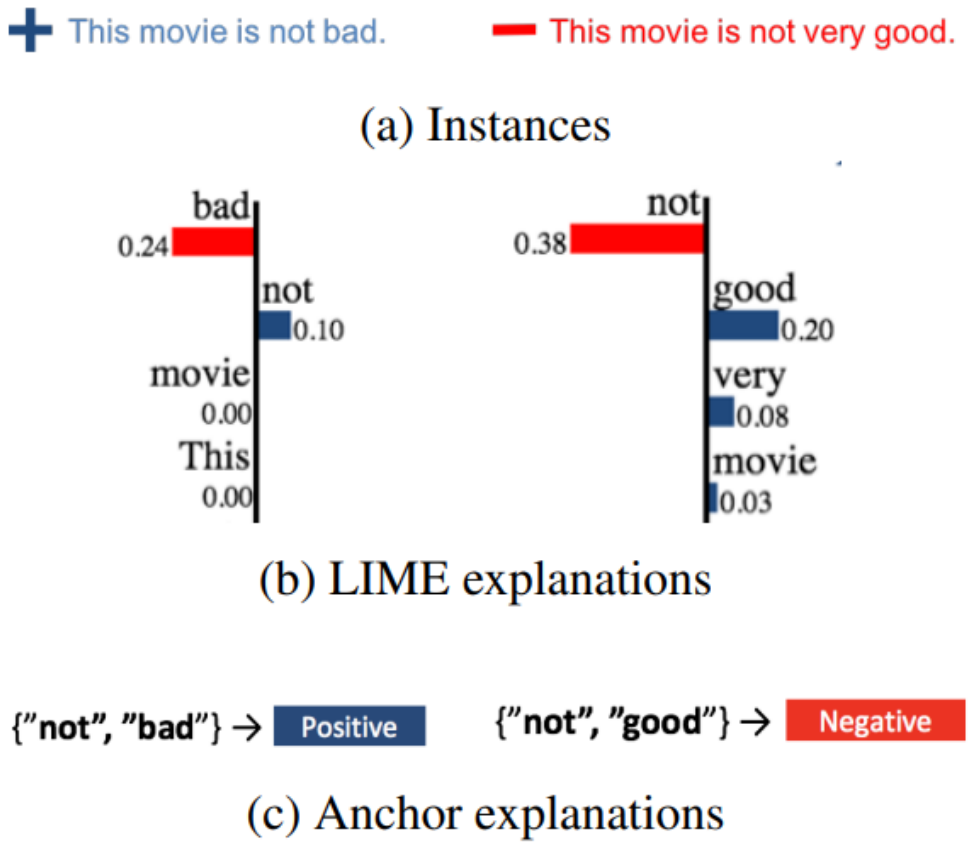
- Medimos las reglas en base a 2 métricas:
  - **Precisión**: La proporción de instancias perturbadas donde la regla predice correctamente.
  - **Cobertura**: Cuántas instancias del dataset cumplen la regla.

- Seleccionamos las reglas con la mejor precisión y cobertura
- **Explicación: interpretación de las reglas**

Anchors

- Las características de Anchors son las **mismas que las de LIME**
- Ambas técnicas las desarrollaron las mismas personas
- Diferencia con LIME: el tipo de explicación
  - Anchors → **Anchor explanation (agrupamiento de características)**
- Tiene su propia librería (anchors para Python)

Anchors



Fuente: GitHub

ALE (Accumulated Local Effects)

Sample Data			
age	bmi	heart_disease	P of stroke
2	12	0	20
3	15	0	21
6	11	0	20
22	24	0	30
24	21	0	31
27	24	0	29
45	23	0	40
43	25	0	41
47	25	0	45
66	30	1	93
68	28	1	88
63	29	1	95

- Se utiliza para comprobar el **efecto de cada característica en el comportamiento del modelo**
  - Se calculan los efectos locales acumulados para cada atributo
  - Por ejemplo: para la edad en este dataset

ALE

Sample Data			
age	bmi	heart_disease	P of stroke
2	12	0	20
3	15	0	21
6	11	0	20
[illegible]	[illegible]	[illegible]	[illegible]
24	21	0	31
27	24	0	29
45	23	0	40
43	25	0	41
47	25	0	45
66	30	1	93
68	28	1	88
63	29	1	95

- Se crean intervalos para cada atributo. Por ejemplo: primer intervalo [2,6]

ALE

Sample Data			
2	12	0	**20**
3	15	0	**21**
6	11	0	**20**
24	21	0	**31**
27	24	0	**29**
45	23	0	**40**
43	25	0	**41**
47	25	0	**45**
66	30	1	**93**
68	28	1	**88**
63	29	1	**95**

- Para cada intervalo, se calcula la predicción sustituyendo el valor real del atributo por el valor más bajo y el valor más alto del intervalo, y se calcula la diferencia entre ellos:

Age 3			
Age interval 2-6 (Lower)			
age	bmi	heart_disease	P of stroke
2	12	0	20
2	15	0	22
2	11	0	21
		Average P	21

Age 3			
Age interval 2-6 (Upper)			
age	bmi	heart_disease	P of stroke
6	12	0	22
6	15	0	23
6	11	0	20
		Average P	22

Difference	
2	
1	
-1	
0.67	Average Diff.

ALE

- Hacemos esto para todos los intervalos y luego calculamos la **media acumulada de las diferencias**
- N es el número de instancias

Age 3 — Age interval 2-6 (Lower)

age	bmi	heart_disease	P of stroke
2	12	0	20
2	15	0	22
2	11	0	21
		Average P	21

Age 3 — Age interval 2-6 (Upper)

age	bmi	heart_disease	P of stroke
6	12	0	22
6	15	0	23
6	11	0	20
		Average P	22

Difference	
2	
1	
-1	
0.67	Average Diff.

Age 24 — Age interval 22-27 (Lower)

age	bmi	heart_disease	P of stroke
22	24	0	30
22	21	0	29
22	22	0	27
		Average P	29

Age 24 — Age interval 22-27 (Upper)

age	bmi	heart_disease	P of stroke
27	24	0	31
27	21	0	29
27	22	0	29
		Average P	30

Difference	
1	
0	
2	
1	Average Diff.

Age 45 — Age interval 43-47 (Lower)

age	bmi	heart_disease	P of stroke
43	23	0	40
43	25	0	42
43	25	0	44
		Average P	42

Age 45 — Age interval 43-47 (Upper)

age	bmi	heart_disease	P of stroke
47	23	0	42
47	25	0	44
47	25	0	45
		Average P	44

Difference	
2	
2	
1	
1.67	Average Diff.

Age 66 — Age interval 63-68 (Lower)

age	bmi	heart_disease	P of stroke
63	30	1	93
63	28	1	87
63	29	1	94
		Average P	91

Age 66 — Age interval 63-68 (Upper)

age	bmi	heart_disease	P of stroke
68	30	1	96
68	28	1	90
68	29	1	95
		Average P	94

Difference	
3	
3	
1	
2.33	Average Diff.

0,67 + 1 + 1,67 + 2,33 =

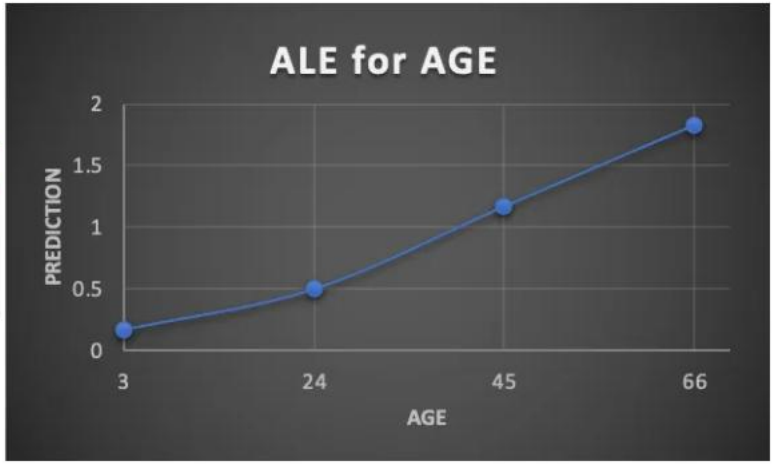
5.67	Accum Diff.
0.5	Accum Diff / N

*Average Prediction is rounded to the nearest integer value

ALE

- Para cada intervalo, calculamos la media del valor del atributo (X), medimos su diferencia media (Y) con la diferencia promedio acumulada (const) y devolvemos Y centrada
- La Y centrada es el **efecto acumulado para una característica**, y es lo que se dibuja en la **gráfica de explicación**

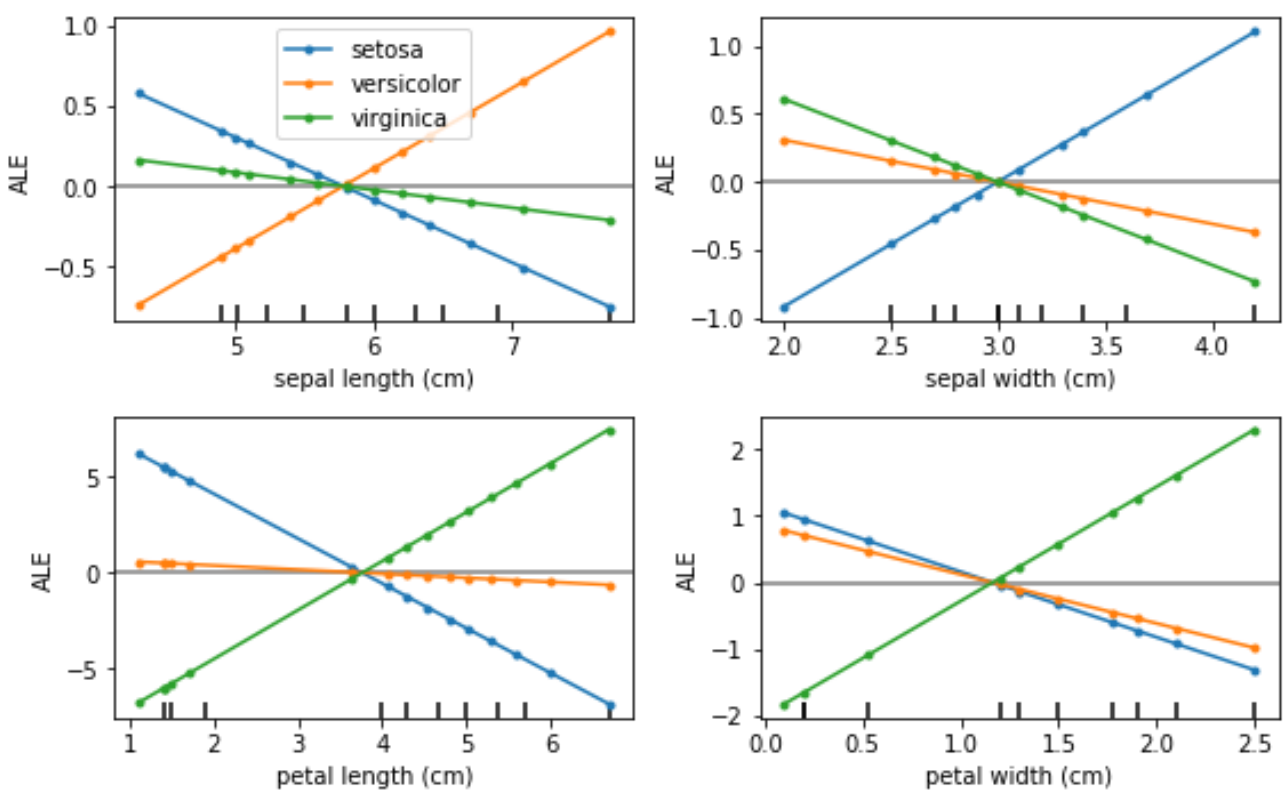
X(age)	Y(p̂) - const	centered Y(p̂)
3	0.67 - 0.5	0.17
24	1.00 - 0.5	0.5
45	1.67 - 0.5	1.17
66	2.33 - 0.5	1.83



ALE

- Es **post-hoc, global** y **model-agnostic**
- Se usa para **clasificación** y **regresión** para explicar técnicas de **aprendizaje automático** y **redes neuronales**
- Muestran las explicaciones a través de **gráficas**
- Funciona para datos **tabulares**
- Devuelve **feature-based explanations**
- Tiene sus propias librerías (alepython, pyale, alibiexplain…)

ALE



Fuente: GitHub

- **Si la pendiente es ascendente:**
  - Aumento del valor de la característica → aumenta la predicción del modelo
- **Si la pendiente es descendente:**
  - Descenso del valor de esa característica → tiende a reducir la predicción del modelo

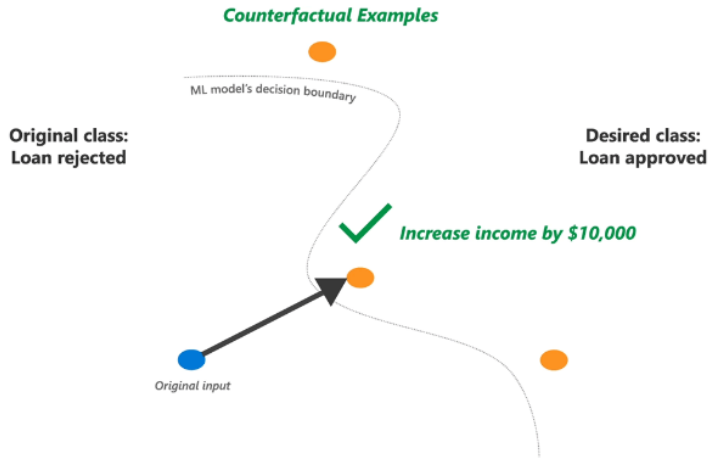
DiCE (Diverse Counterfactual Explanations)

- Genera explicaciones **counterfactuals** para datos tabulares
- Para una query, obtiene otras instancias similares a esa query, que tengan otra predicción distinta, y un atributo con un valor muy distinto
  - El programador normalmente determina el **atributo distinto** y un **rango** [X,Y] de posibles valores en ese atributo del counterfactual
    - Un determinado atributo en el counterfactual será como mucho Y y como mínimo X
  - La elección del atributo y el rango también lo puede decidir DiCE, si el programador no incluye esa información
  - **DiCE encuentra la instancia más similar a la query** con el atributo distinto dentro del rango determinado por el usuario
    - Primero se elige aleatoriamente la instancia
    - Se utiliza una **función de pérdida** para comparar la distancia entre la query y el posible counterfactual
    - Usa **descenso de gradiente** para optimizar el proceso
    - Gracias al descenso de gradiente, sabemos qué atributos modifican la predicción, con lo que ya podemos generar la explicación

DiCE

- Son **post-hoc**, y **model-agnostic** y general explicaciones **locales**
- Suelen mostrarse como una explicación en **tabla** ya que funciona para **datos tabulares**
- Se aplican sobre **tareas de clasificación**
- Sirven para explicar **machine learning** o **redes neuronales**
- Tiene su **propia librería** en Python (dice-ml) y requiere usar **TensorFlow, PyTorch** o **Sklearn** para los modelos a explicar

DiCE



- outcome = 0 (low income) → rejection
- outcome = 1 (high income) → approval

Query instance (original outcome : 0)

	age	workclass	education	marital_status	occupation	race	gender	hours_per_week	income
0	22.0	Private	HS-grad	Single	Service	White	Female	45.0	0.01904

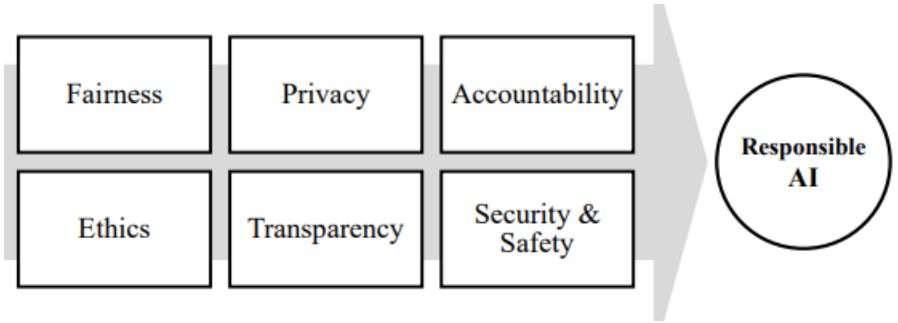
Diverse Counterfactual set (new outcome : 1)

	age	workclass	education	marital_status	occupation	race	gender	hours_per_week	income
0	70.0	-	Masters	-	White-Collar	-	-	51.0	0.534
1	-	Self-Employed	Doctorate	Married	-	-	-	-	0.861
2	47.0	-	-	Married	-	-	-	-	0.589
3	36.0	-	Prof-school	Married	-	-	-	62.0	0.937

Fuente: GitHub

Futuro de la XAI

- ¿Más características para definir sistemas XAI? ¿Menos?
- Necesidad de terminología estandarizada
- Métricas para evaluar XAI
- Contruir IA responsable



Fuente: Arrieta, A. B., et al. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible