**Estructura de Datos**

**Conferencia Octogatos 2022**

**Nombre**: Christian Santamaria

**Fecha**: 20 de septiembre de 2022

1. **Hacia una Inteligencia Artificial responsable.**

Los ponentes de la charla fueron Carla Mamani Chávez y Luis Beltrán, ambos Microsoft MVP. Hoy en día, gracias a los nuevos avances tecnológicos, sobre todo en cuanto a hardware, es realmente divertido e interesante crear diversos algoritmos de Inteligencia Artificial; sin embargo, aún no se tiene asimilado que estas prácticas deben ir de la mano con ciertas consideraciones éticas.

Una encuesta realizada por Tech Vision expuso el nivel de confianza que tienen los usuarios hacia compañías que implementan Inteligencia Artificial en sus aplicaciones/productos/servicios: 35% confían en el buen uso de esta tecnología a pesar de no tener certeza de cómo realmente funciona, el restante 65% no lo hace.

La IA responsable es una práctica enforcada en la evaluación, desarrollo e implementación de sistemas de IA de manera segura, confiable y ética, lo cual conlleva tomar decisiones y hacerse cargo de sus consecuencias. En este contexto, se cuenta con 6 principios importantes:

1. Equidad. Considerar la posibilidad de que un modelo de ML (*Machine Learning*) podría incluir in BIAS (sesgo) que genere desigualdad o discriminación hacia cierto sector de usuarios; ante ello, siempre se debe considerar la mayor cantidad de grupos sociales.
2. Transparencia. Se habla de *“Feature Importance”* tanto global como local, esto quiere decir, identificar cuales son las características más importantes que determinan una predicción en un algoritmo de ML.
3. Fiabilidad. Los modelos deben ser robustos y seguros para que sean confiables, no deben deteriorarse con el tiempo. Encontrarse en constante entrenamiento con datos actualizados.
4. Privacidad. No vulnerar los datos privados de individuos testeados.
5. Inclusión. Reconocimiento de las múltiples necesidades de los seres humanos, producto de ser una raza con tanta diversidad.
6. Responsabilidad. Ser conscientes que toda decisión tomada conlleva un proceso de rendición de cuentas.
7. **Construyendo ML OpenSource.**

El ponente de esta charla fue Omar Sanseviero. Se habló de un poco de historia acerca de algoritmos de ML abiertos y compartidos al público: hace tres años, una compañía llamada OpenAI lanzó GPT2, un algoritmo que intentaba predecir la siguiente palabra a la hora de escribir texto. Ahora, se tiene GPT3, algoritmo capaz de crear páginas web sencillas y escribir líneas de código a partir de instrucciones, sin embargo, el algoritmo no fue compartido por preocupación a la posibilidad de desarrollo de aplicaciones maliciosas, algo perfectamente entendible. Lo negativo de dicha decisión es que se imposibilitó la investigación a detalle de cómo se entrenaba el modelo y descubrir posibles sesgos.

Por su parte, Google lanzó su modelo BERT 101, considerado State of the Art, no obstante, para entrenarlo se necesitaba de muchos recursos y poder computacional, por lo que solo universidades y centros de investigación con gran capacidad adquisitiva podían realizar pruebas con él.

Teniendo en cuenta las consideraciones previas, nace *Hugging Face,* una plataforma, *GitHub-based,* donde se pueden compartir modelos de ML de forma OpenSource. Dado que uno de los grandes problemas para poder trabajar con modelos era la baja potencia computacional, se ideó publicar y compartir algoritmos que usen Tranfer Learning, es decir que se puede aprovechar lo que ya se aprendió en un modelo base y ocuparlo para entrenar otro; además, son necesarias las *Model Cards* que funcionan a manera de documentación: explican los BIAS, parámetros, hiper parámetros usados, reportan métricas; todo esto facilita la reproducibilidad. De igual forma, existe *Datasets Hub,* repositorios que permiten utilizar bases de datos inmensas de entrenamiento con una sola línea de código; incluso, Python cuenta con librerías que permiten el manejo de esta gran cantidad de datos a pesar de no tener una RAM con suficiente procesamiento.

Otro problema que antes existía era la dificultad para compartir resultados de un algoritmo de ML entrenado. Antes, múltiples personas alrededor del mundo que trabajaban sobre sus modelos deseaban compartir resultados a la comunidad, pero hacerlo era extremadamente tedioso. La mejor opción era compartirlo a través de una aplicación web, para ello se necesitaban conocimientos de Flask, HTML, CSS y más, lo cual desanimaba a los desarrolladores. Hoy en día, se cuenta con una librería denominada *gradio* que facilita la creación de interfaces interactivas donde se pueden probar modelos de ML entrenados.

Nos encontramos en un punto de cambio en el Machine Learning: antes, solo aquellos que tenían conocimientos especializados y gran poder adquisitivo podían entrenar modelos de ML, ahora, cualquier persona con acceso a un browser puede hacerlo.

1. **Accesibilidad WEB.**

En la vida cotidiana, generalmente, es muy obvio darnos cuenta de que existen lugares, productos o servicios que no son accesibles para todas las personas, por ejemplo, sitios a donde se puede acceder solo por escaleras/gradas. Esto no es tan evidente en el mundo digital.

Las discapacidades que evitan la accesibilidad web pueden clasificarse realizándonos dos preguntas:

¿Cuánto tiempo dura la incapacidad?

1. Permanente
2. Temporal
3. Situacional

¿Qué parte del cuerpo involucra?

1. Sensitivo
2. Visual
3. Auditivo
4. Motor

Actualmente, según WHO, se estima que existen 1.300 millones de personas con discapacidad visual, los desarrolladores no pueden ignorar esto ya que se trata de nuestros consumidores/clientes finales.

Existen técnicas (algunas automatizadas) que permiten probar la accesibilidad de un sitio web.

1. Testear código durante el desarrollo de la aplicación. (@axe\_core/react permite ejecutar pruebas de accesibilidad en aplicaciones React).
2. Uso de herramientas de desarrollo de los navegadores web (Google Chrome). Inspeccionar elementos HTML y aplicar las pruebas AA y AAA, destinadas para personas con discapacidad visual.
3. Pruebas Unitarias con jest-axe que ayudan a resaltar vulnerabilidades del código de acuerdo con reglas predefinidas sobre accesibilidad (coloraciones, posiciones de botones, etc.)
4. Uso de la herramienta LightHouse. Permite generar informes detallados sobre la accesibilidad de un sitio web únicamente con proporcionarle el URL.
5. Existen otros softwares que facilitan realizar pruebas y simulaciones: Wave, Accessibility Insights, Emulador de deficiencias visuales de Firefox.

Si se desea automatizar estos procesos, es posible mediante GitHub Actions, scripts que ejecutan las pruebas mencionadas en tiempo real y retornan retroalimentación a considerar.

Por los problemas de accesibilidad es que han nacido herramientas realmente útiles como los lectores de pantalla: Apple Voice Over, Microsoft Narrator, Linux Orca.