



Aprende Machine Learning

antes de que sea demasiado tarde

GENERAL

Interpretación de Modelos de Machine Learning

🕒 abril 30 by Na8

Descifrar las decisiones tomadas por la máquina

La interpretación de las decisiones tomadas por nuestros algoritmos de Machine Learning pasa a un plano muy importante: para comprender el modelo y mejorarlo, **evitar «biases» (ó descubrirlos)**, para justificar nuestra confianza en el modelo y hasta legalmente pues es requerido por **leyes como la GDPR** -para decisiones delicadas como puede ser dar ó no un crédito a una persona-.

Si nuestro algoritmo tuviera que detectar enfermedades y suponiendo que logramos una tasa de aciertos del 90% ¿no te parecería lógico comprender cómo lo ha hecho? ¿es puro azar? ¿está teniendo en cuenta combinaciones de características que nosotros no contemplamos?

Si de pequeño eras curioso y querías saber cómo funcionaban las cosas: relojes, autos, ó hasta el mismísimo ordenador... serás un poco como yo... y... no siempre nos convence el concepto de «caja negra».

Abriendo la Caja negra

El concepto de caja negra a veces es muy beneficioso, en sistemas decimos «yo al método le tiro estos parámetros y me devuelve true ó false». Genial, con eso nos basta. Podemos trabajar en equipos distribuidos, intercambiar interfaces y listo. Podemos confiar en otras librerías ó paquetes «sin saber cómo lo hacen» pero que nos resuelven problemas. Y las encajamos como piezas de un puzzle.

Los algoritmos de Machine Learning, hasta ahora funcionaban muy de ese modo. Es decir, podemos hacer una red neuronal de 10 capas con 80 neuronas cada una, dropout, recurrencia y que nos dé unas buenas clasificaciones. Pero ¿qué pasa por dentro? ¿cómo hizo? ¿es magia?... esas oscuras épocas de incertidumbre deben acabar y deberemos tomar control de porqué se hacen las cosas como se hacen.

¿Que hay dentro de la caja negra?

Explainable Machine Learning

Interpretar el Modelos en Machine Learning es la habilidad de explicar su funcionamiento ó presentarlo de manera comprensible al humano.

¿Por qué es importante interpretar los modelos?

Imaginemos que nuestro algoritmo decidirá a qué empleado le daremos un ascenso, dadas sus características e historia en la empresa. Y luego de entrenar el modelo vemos que «aparentemente da buenos resultados» pero... todas las elecciones para puestos gerenciales son siempre para hombres y ninguna mujer.... mmmm.. sospechoso, ¿no?

Ese modelo «aprendió» que durante los últimos 10 años, los cargos gerenciales de esa empresa siempre fueron para hombres. Si ese algoritmo pasa a producción, estará discriminando a las mujeres e impidiendo su ascenso.

Entonces ¿Cómo hacemos para interpretar el modelo?

Respuesta corta: **con otro modelo** que ayude a los humanos a interpretar los procesos.

Hay que decir que modelos como «1 árbol de decisión pequeño» ó clasificación lineal, pueden llegar a interpretarse por su gráfica y/o fórmula (repito: si son sencillos). Sin embargo un **Random Forest** ó las **Redes Neuronales** son complejas y prácticamente imposibles de comprender <<de un vistazo>>.

Los beneficios de la «interpretabilidad de los modelos «son:

- Dar confiabilidad en los resultados.
- Ayudar en el Debugging.
- Informar a la Ingeniería de Características (Feature Engineer).
- Detectar necesidad de coleccionar nuevas muestras.
- Ayudar a una persona en la toma de decisiones.
- Mayor seguridad/robustez en el modelo obtenido.

Técnicas de Interpretación de modelos

Del análisis de los modelos podemos obtener:

Del análisis de los modelos podemos obtener:

- Características más importantes (features)
- Para una predicción en particular del modelo, el efecto que tuvo en ella cada característica
- Efecto de cada característica en el global de las predicciones del modelo

Veamos algunas de esas técnicas y qué librerías de Python nos brindan estas funcionalidades:

1- Permutation Importance

¿Cuales de las features piensa el modelo que son más importantes? ¿Qué Características tienen mayor impacto en las predicciones? Estos conceptos son conocidos como «Feature Importante» y «Permutation Importance» y nos sirven para calcular nuestras características de entrada al modelo. Nos sirve para poder ver cuando nuestro modelo está funcionando de manera contra-intuitiva y también para demostrar a terceros cuando el funcionamiento es correcto.

Para hacer Permutation Importante debemos primero entrenar un modelo y «encajarlo» (fit). Luego tomamos el set de validación y tomamos las features una por vez: por ejemplo, tomamos la primer columna de entrada y mezclamos todos sus valores entre sus filas (pero el resto de features se mantienen igual). Entonces hacemos predicción usando el mismo modelo entrenado y deberían empeorar los resultados. Si «desmejoran mucho» es que esa feature era muy importante. En cambio, si no afecta demasiado, tampoco variarán mucho las predicciones obtenidas y quiere decir que esa característica no es relevante. Y así lo hacemos con todas las características, desordenando de a una a la vez.

Podemos utilizar la librería [ELIS para Python](#) para visualizar la Permutation Importance

2- Partial Dependence Plots (PDP)

Los PDPs muestran el efecto marginal de una o dos características que tienen sobre la predicción dictada por un modelo. Los PDPs muestran cómo afectan las distintas características a las predicciones. El PDP puede mostrar la relación entre nuestra variable de salida y una ó dos características de entrada.

Lo que hacemos es tomar de a una sola fila, e ir variando los valores de una sola de las

features (que queremos investigar) contra un modelo YA entrenado. Entonces veremos en que intervalos esa característica afecta a los resultados del modelo.

Lo podemos hacer hasta con 2 variables a la vez usando «2D Partial Plots» y visualizarlo.

Para esto podemos utilizar la librería **PDPBox**

3-SHAP Values (en predicciones individuales)

SHAP viene de «Shapley Additive exPlanation» y está **basado en la teoría de Juegos** para explicar cómo cada uno de los jugadores que intervienen en un «juego colaborativo» contribuyen en el éxito de la partida. Con esto podemos comprender una predicción y como impacta cada feature. Podemos decir que la interpretabilidad que nos ofrecen los valores SHAP es de las mejores.

De manera muy sencilla -e incompleta- de cómo se calculan estos valores podemos imaginar a una grupo de desarrolladores, testers, arquitectos y managers (features) que trabajan en conjunto («juegan»/colaboran) para crear un Sistema de Software y queremos saber cuánto contribuyó cada uno de ellos en su producción. Lo que haremos es ir intercalando a los participantes en diversos «orden de aparición» ABCD, ABDC, ADBC, etc. e ir midiendo la <<**contribución marginal**>> de cada participante cada vez. Con ello sacar el promedio de cada uno y tendremos los valores Shapley que nos indican cuánto contribuyo cada jugador a conseguir el resultado obtenido.

Supongamos que tenemos que explicar a una persona por qué se ha rechazado su solicitud de un crédito -esto es, una única predicción, y no «el accuracy global» del modelo- los valores SHAP nos muestran cuales características que alimentan al modelo <<empujan>> a la denegación (ó aceptación) esa petición en concreto.

Utilizamos la librería **SHAP** para python para obtener estos valores.

4- Usos avanzados de Shap (comprensión global)

Si recopilamos muchos valores Shap podremos tener una mejor comprensión del modelo en su conjunto. De allí aparecen las gráficas «Shap Summary Plot» y «Shap Dependence Contribution Plot».

Shap Summary Plot

Calculando los Shap Values de cada muestra, podemos obtener esta Visualización que nos muestra cuales características son las más importantes y el rango de valores donde afecta al set de datos.

Shap Dependence Contribution Plot

Esta gráfica es similar a la de los PDPs (vistos en el punto 2) pero nos dan mucho mayor detalle.

No puedo dejar de mencionar a una gran [librería para ML Interpretability llamada LIME \(Local Interpretable Model Explanation\)](#) y que nos ofrece comprensión a humanos para modelos de NLP (destacando visualmente palabras en el texto) y para imágenes clasificadas por una CNN (mostrando las áreas en donde «mira» la red).

También mencionar otra [Librería Python llamada Skater](#) -es de Oracle- y aunque aún está en desarrollo, provee de buenas herramientas.

Conclusión

La importancia de la interpretabilidad de los modelos de Machine Learning es crucial para poder justificar y comprender las predicciones y/o resultados obtenidos y hasta [legalmente](#). Es curioso que necesitemos «modelos que expliquen como funcionan los modelos» para poder «bajar» a entendimiento humano la complejidad de lo que ocurre en nuestras máquinas de aprendizaje. Finalmente, aplicando diversos métodos, Permutation Importance, los PDP y los Shap Values logramos obtener transparencia en nuestro desarrollo y un panorama claro sobre cómo funciona nuestro engranaje para obtener los resultados.

Suscribe al Blog

Recibe los nuevos artículos sobre Aprendizaje Automático, teoría y el código Python

Email:

ENVIAR

NOTA: algunos usuarios reportaron que el email de confirmación a la suscripción entraron en la carpeta SPAM. Te sugiero que revises y recomendando agregar el remitente a tus contactos. Gracias!

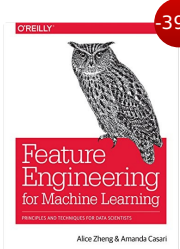
Recursos Adicionales

Te recomiendo sobre todo y para pasar al Código este curso completo en Kaggle: [Machine Learning Explainability](#)

Algunos artículos y videos sobre Interpretación de Modelos (en inglés)

- Libro muy bueno! [Interpretable Machine Learning](#)
- artículo [Hands-on Machine Learning Model Interpretation](#)
- (artículo) [Interpretable Machine Learning](#)
- Video (1:30hs) [Open the black box: an intro to model interpretability](#)
- Video (1:30hs) [Interpretable ML for Computer Vision](#)
- [One Method to Rule Them All: Shap Values](#)
- [Derisking ML and Artificial Intelligence](#)
- [The Mythos of Model Interpretability](#)


Producto disponible en Amazon.es



Feature Engineering for M...

Precio: **EUR 35,97**

Precio recomendado: EUR 58,80



Comparte el artículo:



Relacionado



12 Consejos útiles para aplicar Machine Learning

7 pasos del Machine Learning para construir tu máquina
Describiré los 7 pasos genéricos que debes seguir para construir tu propia Inteligencia Artificial con Machine Learning. Paso 1: Colectar Datos Dada la problemática que



Principales Algoritmos usados en Machine Learning

[aprender](#)[Aprendizaje Automático](#)[consejos](#)[Definición](#)[ejemplo](#)[Gráficas](#)[Modelos](#)[Python](#)[12 CONSEJOS ÚTILES PARA APLICAR MACHINE LEARNING](#)[CLASIFICACIÓN CON DATOS DESBALANCEADOS](#)

4 comments



Bernardo Meléndez Álvarez · mayo 3

Gracias por el artículo y lo enlaces recomendados.

Responder



Na8 · mayo 4

Me alegro que te haya gustado! Saludos

Responder



Enrique Codert · junio 16

Muy buen artículo. Muchas gracias. No conocía las librerías que has comentado y que seguro utilizaré. 😊

Responder



Na8 · junio 19

Hola Enrique, la verdad que es bastante novedoso todo el tema de interpretación de modelos y de hecho están apareciendo unas

interpretación de modelos, y de hecho están apareciendo unas nuevas bastante interesantes, intentaré actualizar este artículo pronto con nuevos recursos. Saludos y gracias por escribir

Responder

Deja un comentario

Introduce aquí tu comentario...

Visita nuestra Guía de Aprendizaje

Buscar

Search ...

Contacto

Suscripción

Recibe los artículos de Aprende Machine Learning en tu casilla de correo. Cada 15 días y sin Spam!

Email:

ENVIAR

Proudly powered by WordPress | Theme: Eighties by Justin Kopepasah.

Obtén un [navegador compatible](#) para conseguir un reto reCAPTCHA.

¿Por qué tengo que hacer esto?