GENERAL

12 Consejos útiles para aplicar Machine Learning

abril 4 by Na8

Si vas por el buen camino hacia el aprendizaje del Machine Learning, la inteligencia artificial y la ciencia de datos, seguramente te hayas topado con trabas y obstáculos frecuentes. En este artículo repasaremos 12 útiles consejos para tener en cuenta a la hora de trabajar con los modelos del Aprendizaje Automático. Estos postulados surgen del paper *«A Few Useful Things to Know about Machine Learning»* escrito en 2012 por Pedro Domingos.

> No olvides seguir los 7 pasos del Machine Learning

Vamos al grano!

Con el objetivo de ilustrar mejor estos consejos, nos centraremos en la aplicación del Machine Learning de Clasificar, pero esto podría servir para otros usos.

Los 3 componentes del Aprendizaje Automático

Supongamos que tienes un problema al que crees que puedes aplicar ML. ¿Qué modelo usar? Deberá ser una combinación de estos 3 componentes: Representación, evaluación y optimización.

- **Representación**: Un *clasificador* deberá poder ser representado en un lenguaje formal que entienda el ordenador. Deberemos elegir entre los diversos algoritmos que sirven para resolver el problema. A este conjunto de «clasificadores aptos» se les llamará *«espacio de hipótesis del aprendiz»*. Ej: SVM, Regresión Logística, K-nearest neighbor, árboles de decisión, Redes Neuronales, etc.
- **Evaluación**: Se necesitará una función de evaluación para distinguir entre un buen clasificador ó uno malo. También es llamada *función objetivo ó scoring function*. Ejemplos son accuracy, likelihood, information gain, etc.
- **Optimización:** necesitamos un método de búsqueda entre los clasificadores para mejorar el resultado de la Evaluación. Su elección será clave. EJ: Descenso por gradiente, mínimos cuadrados, etc.
-

Lo más importante es lograr la Generalización de la máquina

El objetivo fundamental del Machine Learning es lograr la generalización del conocimiento más allá de las muestras tomadas en el conjunto de entrenamiento. Es es porque por más que entrenemos con muchísimos datos, es poco probable que se vuelvan a dar esos mismos casos al testear nuestra máquina. Por más que usemos «un millón de fotos de gatitos» para entrenar, siempre habrá fotos nuevas ó distintas que nuestro modelo desconoce. Entonces la magia ocurrirá si nuestro algoritmo es capaz de generalizar lo aprendido durante el entrenamiento y puede detectar «al gatito millón uno». Siempre es conveniente guardar una parte de nuestro set de datos para poder validar lo aprendido durante el entrenamiento, con muestras que la máquina nunca ha visto y comprobar que sigue dando un buen resultado.

Los datos solos, no alcanzan

Siendo la generalización el objetivo del ML trae como consecuencia que sólo con datos «no alcanza», no importa con cuántos contemos.

¿Entonces cómo podemos pretender que nuestras máquinas aprendan algo? Afortunadamente, en este proceso de inducción que realizamos con los algoritmos logramos llegar a ciertos niveles que nos dan buenos resultados.

El machine learning no hace magia. Sin datos no funcionará. Lo que hace es «*sacar más con poco*«. Y claro, cuantos más datos mejor. Pero no lo son todo. Deberemos combinar «conocimiento» con los datos.

El Overfitting tiene muchas caras

Si al entrenar nuestro modelo obtenemos resultados <<demasiado buenos>> con 100% aciertos y en el set de test apenas alcanzamos un 50% (ó menos!) es muy probable que nos hayamos topado con en «**el gran problema del ML»: el overfitting.**

Pero debemos saber que podemos «caer» en el overfitting de diversas maneras, a veces sin darnos cuenta. Dos de sus caras son:

- **Bias** (ó sesgo): es la tendencia a aprender -equivocadamente- algo falso.
- **Varianza**: es la tendencia a aprender algo random no relacionado con la realidad.

A veces puede ocurrir que un algoritmo -aprendiz- que «parece menos potente», obtenga

mejor resultado que uno «super poderoso» que cae en overfitting. A tener en cuenta 😉

Es difícil evitar el Overfitting, se puede utilizar Regularización ó la validación cruzada (cross validation), u otras técnicas pero ninguna nos asegura evitarlo del todo. Muchas veces ocurre que al querer corregir la varianza, caemos en Bias... y viceversa. Lograr evitar ambos en simultáneo es «el desafío» que tenemos.

A veces falla «la intuición» con muchas dimensiones

Cuando tenemos muchas features, por ejemplo 100 ó más, puede que alguno de nuestros algoritmos de aprendizaje «se vuelva loco»... es decir, que no logre generalizar ó que lo haga mal. A esto se le llamó **«la maldición de la dimensionalidad»** (Bellman, 1961). Esto dependerá también de la distribución de los datos de entrada, pero para entenderlo: con k-nearest neighbor es <<fácil visualizar en 2 ó 3 dimensiones>> los clusters. Pero al aumentar dimensiones puede ocurrir que para el algoritmo **todos los puntos sean vecinos cercanos** unos de otros, devolviéndonos resultados aleatorios.

Debemos tener esto en cuenta y si fuera el caso utilizar algún algoritmo de reducción de dimensiones – PCA, t-SNE- para aplacar el problema.

Que se cumplan supuestos teóricos no nos garantiza nada

Al aprender ML podemos leer en papers, ó en cursos y artículos algunas afirmaciones teóricas que intentan ayudarnos y guiarnos. Bueno, como el ML es un fenómeno muy complejo y **depende tanto de cada problema en particular**, las dimensiones y los datos de entrada (su distribución), los casos positivos/negativos que tengamos de las muestras y de tantas otras variables, es posible que **muchos de esos supuestos teóricos NO nos ayuden en nuestro «problema particular»**.

Esto puede sonar muy decepcionante, y en parte lo es. Pero esto debemos saberlo para contrastar nuestros datos y no confiar en que «como a Fulanito le dio así, a mi también me funcionará». Con sólo variar un parámetro, o un sólo dato de entrada de nuestro conjunto de entrenamiento, podemos obtener resultados completamente distintos. Estar alertas!!!

La clave está en la Ingeniería de

Características (Feature Engineer)

Seguramente pasemos mucho más tiempo seleccionando los features, transformando, preprocesando que el tiempo dedicado a preparar/ejecutar el algoritmo de Machine Learning. Muchas veces el desafío será si tenemos pocas dimensiones ser creativos y poder generar nuevas y útiles características, ó en caso de tener muchas poder seleccionar cuales serán realmente valiosas y cuales descartar.

Deberemos ser cuidadosos: si tenemos muchas features, podemos evaluarlas individualmente y pensar que algunas no aportan demasiado valor. Sin embargo, esas mismas características **puede que sean imprescindibles** si las consideramos *en combinación con otras*. Eh ahí nuestro ingenio y mucha prueba y error.

> Una gran herramienta es la Interpretación de Modelos por Importancia de Features, Los SHAP Values y más gráficas que comento en este nuevo artículo!

Más muestras superan a un algoritmo complejo

Conseguir más muestras para entrenamiento utilizando un algoritmo «simple», puede ser mejor que un algoritmo complejo que «tarde tanto en ejecutar y no termina nunca».

En ciencias de la computación, solíamos tener dos limitantes: **tiempo y recursos** cómo la memoria. Ahora, con el Machine Learning aparece una tercera: **los datos de entrenamiento**. Actualmente podemos encontrar cantidades masivas de datos y nuestro «cuello de botella» es el tiempo.

Si bien contar con «más y más datos» es bueno y hasta impulsa la creación de modelos (algoritmos) más complejos para aprovecharlos, se da una paradoja: en la práctica algoritmos «más simples» pueden obtener buenos resultados en tiempo razonable contra algoritmos complejos que tardan una eternidad.

Entonces el consejo es: **al afrontar un problema, empecemos probando con los modelos más sencillos a más complejos** (si hiciera falta!).

Ensamble de Modelos

Está bien aplicar un modelo para resolver el problema. Pero estudios han demostrado que hacer ensamble de modelos muchas veces mejora significativamente los resultados. Esto consiste en combinar más de un modelo (por ej. una red neuronal, K-nn, árboles, etc). Las tres técnicas más utilizadas son (ejemplo para clasificadores de «perros y gatos»):

- **Bagging:** alimentamos diversos modelos haciendo resamplig de las muestras y finalmente hacemos una votación (voting) con la clasificación obtenida (Supongamos que dos de tres modelos dicen «gato» y uno dice «perro», ganarían los felinos)
- **Boosting:** en este caso, utilizamos un modelo «potenciandolo» a detectar únicamente perros y otro a detectar sólo gatitos.
- **Stacking:** utilizaremos diversos modelos, apilados «uno detrás de otro», es decir, la salida del primero puede ser la entrada (ó un feature) del siguiente modelo y así sucesivamente. Para una famosa competición de Netflix, el ganador había encadenado más de 100 modelos!!

Simplicidad no implica precisión

Hubo un postulado algo confuso que parecía decir que al aplicar un modelo simple obteníamos la mayor precisión (frente a uno complejo). Sin embargo no hay que confundir: la simpleza no implica mayor precisión, esto se puede comprobar fácilmente con el punto anterior, pues al hacer ensamble de stacking de modelos vemos claramente que **no se deteriora el resultado** y en todo caso lo puede mejorar. Entonces podemos optar por modelos simples que «ya cuentan» con ventajas intrínsecas -probablemente en tiempo y coste- pero no necesariamente por la precisión del resultado.

Un problema «Representable», no implica que pueda resolverse con Aprendizaje Automático

Podemos tender a pensar que para cualquier problema que podamos representar podrá ser construida una máquina que lo resuelva. Sin embargo hay problemas que nunca podrán ser «aprendidos» por una máquina, pues por ejemplo no tenemos las muestras

suficientes para que generalice. En esos casos, podemos fácilmente encontrar la representación pero no lograremos dar con una solución en ML.

La correlación no implica causa

Para alimentar nuestras máquinas de Aprendizaje Automático utilizamos datos que son *«muestras observables»* donde **la variable predictiva no está bajo control** del algoritmo (en contraposición a *muestras experimentales*). Por esto es que las correlaciones que encontremos pueden ser interpretadas como señales de «responsabilidad» de **la causa** del problema. La realidad es que no necesariamente esa correlación implica la causa si no que son una buena pista para ponernos a investigar las conexiones que llevan a esos resultados.

Conclusión

Este gran paper de Pedro Domingos nos abre un poco los ojos sobre diversas trampas en las que podemos caer al trabajar en Machine Learning. Hace unos -pocos- años surgieron papers sobre Interpretación de Modelos de ML que comento en mi nuevo artículo y que dan luz a poder maniobrar ante situaciones problemáticas.

Moraleja en tiempos de IA

Personalmente creo que la «gran enseñanza» que nos deja es que debemos estar atentos, no confiarnos ni de datos, ni de algoritmos, ni de soluciones mágicas. Debemos ser muy profesionales y científicos, mucha prueba y error, validación y comprobación de los resultados. Y una vez hecho esto... volver a comprobar!

Descarga el paper original desde aquí

Suscripción: Nuevos artículos

Recibe los nuevos artículos sobre Aprendizaje Automático, teoría y práctica Python en tu casilla de correo!

Email: Your email address here

ENVIAR

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

Comparte el artículo:

Relacionado

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

Aprendizaje Profundo: una Guía rápida

¿Qué es Machine Learning? Una definición

consejos ejemplo Machine Learning Modelos overfitting

PRONÓSTICO DE VENTAS CON REDES NEURONALES – PARTE 2

INTERPRETACIÓN DE MODELOS DE MACHINE LEARNING

4 comments

loscalzojony · abril 4

Gran artículo!

te vendría bien realizar un podcast! vengo leyendo los articulos en inglés siempre, pero es una buena forma de «centrar» el conocimiento con el español.

Gracias por tomarte el tiempo!

Responder

Na8 · abril 5

Hola gracias por el comentario Jony, me parece una buena sugerencia, lo tendré en mente!

Responder

José · abril 7

Gracias por tomarte la molestia de hacer éste Blog! es de gran ayuda para los que estamos comenzando. Me suscribo igualmente a la sugerencia del podcast, si tienes tiempo por supuesto.

Responder

Na8 · abril 8

Hola José gracias por escribir y el apoyo al Blog! Ahora ya son dos los que piden un Podcast, por lo que lo sigo teniendo en cuenta para el futuro!.
Saludos y seguimos en contacto!

Responder

Deja un comentario

Introduce aquí tu comentario...

Visita nuestra Guía de Aprendizaje

Buscar

Search ...

Contacto

Contacto

Suscripción

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

Email: Your email address here

ENVIAR