Qué es overfitting y underfitting y cómo solucionarlo

Las principales causas al obtener malos resultados en Machine Learning son el overfitting o el underfitting de los datos. Cuando entrenamos nuestro modelo intentamos “hacer encajar” -fit en inglés- los datos de entrada entre ellos y con la salida. Tal vez se pueda traducir overfitting como “sobreajuste” y underfitting como “subajuste” y hacen referencia al fallo de nuestro modelo al generalizar -encajar- el conocimiento que pretendemos que adquieran. Lo explicaré a continuación con un ejemplo.

Generalización del Conocimiento

Como si se tratase de un ser humano, las máquinas de aprendizaje deberán ser capaces de generalizar conceptos. Supongamos que vemos un perro Labrador por primera vez en la vida y nos dicen “eso es un perro”. Luego nos enseñan un Caniche y nos preguntan: ¿eso es un perro? Diremos “No”, pues no se parece en nada a lo que aprendimos anteriormente. Ahora imaginemos que nuestro tutor nos muestra un libro con fotos de 10 razas de perros distintas. Cuando veamos una raza de perro que desconocíamos seguramente seremos capaces de reconocer al cuadrúpedo canino al tiempo de poder discernir en que un gato no es un perro, aunque sea peludo y tenga 4 patas.

Cuando entrenamos nuestros modelos computacionales con un conjunto de datos de entrada estamos haciendo que el algoritmo sea capaz de generalizar un concepto para que al consultarle por un nuevo conjunto de datos desconocido éste sea capaz de sintetizarlo, comprenderlo y devolvernos un resultado fiable dada su capacidad de generalización.

El problema de la Máquina al Generalizar

Si nuestros datos de entrenamiento son muy pocos nuestra máquina no será capaz de generalizar el conocimiento y estará incurriendo en underfitting. Este es el caso en el que le enseñamos sólo una raza de perros y pretendemos que pueda reconocer a otras 10 razas de perros distintas. El algoritmo no será capaz de darnos un resultado bueno por falta de “materia prima” para hacer sólido su conocimiento. También es ejemplo de “subajuste” cuando la máquina reconoce todo lo que “ve” como un perro, tanto una foto de un gato o un coche.

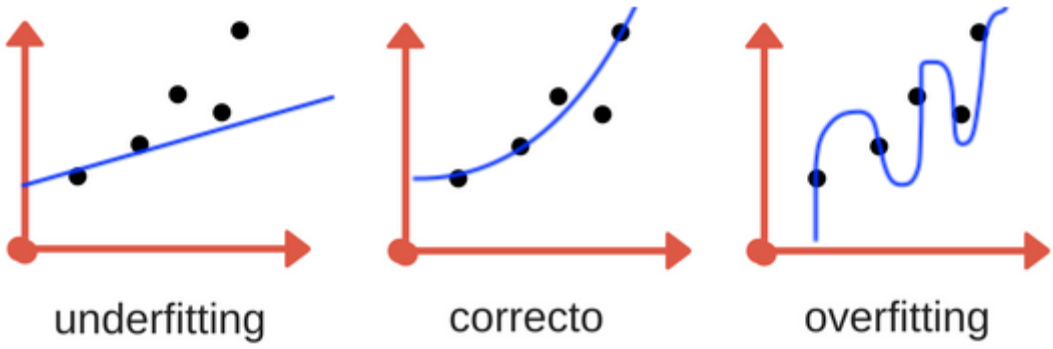
Por el contrario, si entrenamos a nuestra máquina con 10 razas de perros sólo de color marrón de manera rigurosa y luego enseñamos una foto de un perro blanco, nuestro modelo no podrá reconocerlo cómo perro por no cumplir exactamente con las características que aprendió (el color forzosamente debía ser marrón). Aquí se trata de un problema de overfitting.

Tanto el problema del ajuste “por debajo” como “por encima” de los datos son malos porque no permiten que nuestra máquina generalice el conocimiento y no nos darán buenas predicciones (o clasificación, o agrupación, etc.)

Overfitting en Machine Learning

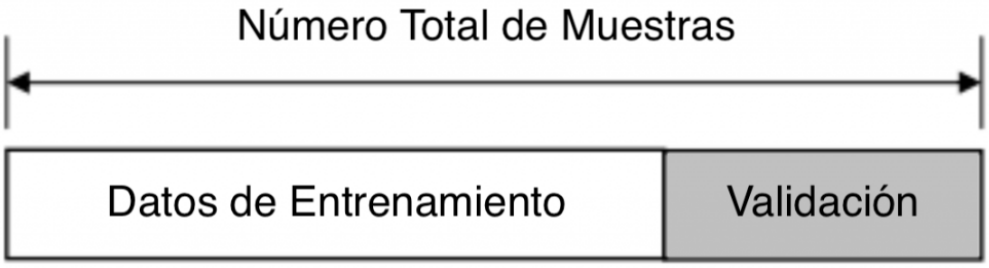
Es muy común que al comenzar a aprender machine learning caigamos en el problema del Overfitting. Lo que ocurrirá es que nuestra máquina sólo se ajustará a aprender los casos particulares que le enseñamos y será incapaz de reconocer nuevos datos de entrada. En nuestro conjunto de datos de entrada muchas veces introducimos muestras atípicas (ó anomalas) o con "ruido/distorción" en alguna de sus dimensiones, o muestras que pueden no ser del todo representativas. Cuando "sobre-entrenamos" nuestro modelo y caemos en el overfitting, nuestro algoritmo estará considerando como válidos sólo los datos idénticos a los de nuestro conjunto de entrenamiento –incluidos sus defectos– y siendo incapaz de distinguir entradas buenas como fiables si se salen un poco de los rangos ya prestablecidos.

El equilibrio del Aprendizaje



Deberemos encontrar un punto medio en el aprendizaje de nuestro modelo en el que no estemos incurriendo en underfitting y tampoco en overfitting. A veces esto puede resultar una tarea muy difícil.

Para reconocer este problema deberemos subvididir nuestro conjunto de datos de entrada para entrenamiento en dos: uno para entrenamiento y otro para la Test que el modelo no conocerá de antemano. Esta división se suele hacer del 80% para entrenar y 20%. El conjunto de Test deberá tener muestras diversas en lo posible y una cantidad de muestras suficiente para poder comprobar los resultados una vez entrenado el modelo.



Cuando entrenamos nuestro modelo solemos parametrizar y limitar el algoritmo, por ejemplo la cantidad de iteraciones que tendrá o un valor de "tasa de aprendizaje" (learning-rate) por iteración y muchos otros. Para lograr que nuestro modelo dé buenos resultados iremos revisando y contrastando nuestro entrenamiento con el conjunto de Test y su tasa de errores, utilizando más o menos iteraciones, etc. hasta dar con buenas predicciones y sin tener los problemas de over-under-fitting.

Prevenir el Sobreajuste de datos

Para intentar que estos problemas nos afecten lo menos posible, podemos llevar a cabo diversas acciones.

Cantidad mínima de muestras tanto para entrenar el modelo como para validarlo.

Clases variadas y equilibradas en cantidad: En caso de aprendizaje supervisado y suponiendo que tenemos que clasificar diversas clases o categorías, es importante que los datos de entrenamiento estén balanceados. Supongamos que tenemos que diferenciar entre manzanas, peras y bananas, debemos tener muchas fotos de las 3 frutas y en cantidades similares. Si tenemos muy pocas fotos de peras, esto afectará en el aprendizaje de nuestro algoritmo para identificar esa fruta.

Conjunto de Test de datos. Siempre subdividir nuestro conjunto de datos y mantener una porción del mismo "oculto" a nuestra máquina entrenada. Esto nos permitirá obtener una valoración de aciertos/fallos real del modelo y también nos permitirá detectar fácilmente efectos del overfitting /underfitting.

Parameter Tunning o Ajuste de Parámetros: deberemos experimentar sobre todo dando más/menos "tiempo/iteraciones" al entrenamiento y su aprendizaje hasta encontrar el equilibrio.

Cantidad excesiva de Dimensiones (features), con muchas variantes distintas, sin suficientes muestras. A veces conviene eliminar o reducir la cantidad de características que utilizaremos para entrenar el modelo. Una herramienta útil para hacerlo es PCA.

Quiero notar que si nuestro modelo es una red neuronal artificial –deep learning-, podemos caer en overfitting si usamos capas ocultas en exceso, ya que haríamos que el modelo memorice las posibles salidas, en vez de ser flexible y adecuar las activaciones a las entradas nuevas.

Si el modelo entrenado con el conjunto de train tiene un 90% de aciertos y con el conjunto de test tiene un porcentaje muy bajo, esto señala claramente un problema de overfitting.

Si en el conjunto de Test sólo se acierta un tipo de clase (por ejemplo "peras") o el único resultado que se obtiene es siempre el mismo valor será que se produjo un problema de underfitting.