Estrategias de Entrenamiento

Orlando Ramos Flores

Contenido I

- Configuración de conjuntos de entrenamiento
  - Conjuntos de entrenamiento, desarrollo y prueba
  - Tamaño de los conjuntos de entrenamiento
  - Tips
- Regularización
  - Norma L2 (Weight decay)
  - Norma L1
  - Dropout
- Outliers
  - Definición
  - ¿Cómo se introducen el los datasets?
  - Tipos de valores atípicos
  - ¿Cómo detectar valores atípicos?
  - ¿Qué hacer con los outliers detectados?
  - ECOD

Contenido II

- Bias & Variance
  - Bias & Variance
  - Bias & Variance Trade-off
  - Comparación con la tasa de error óptima
  - Técnicas para reducir Bias
  - Técnicas para reducir Variance

Configuración de conjuntos de entrenamiento

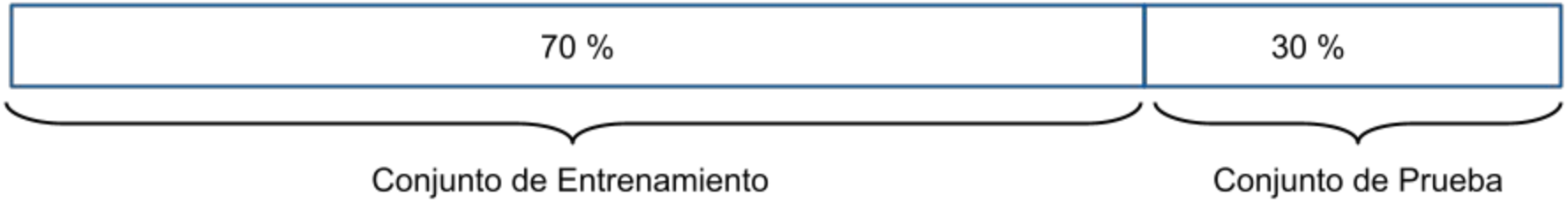
Conjuntos de entrenamiento, desarrollo y prueba

- **Entrenamiento**.
  - El conjunto que se emplea para ejecutar el algoritmo de aprendizaje.
  - Permite obtener los pesos de la red en cuestión.
  - Ajuste de parámetros.
- **Desarrollo**.
  - Se utiliza para ajustar parámetros, seleccionar características y tomar otras decisiones con respecto al algoritmo de aprendizaje.
  - A veces también llamado conjunto de validación cruzada de exclusión.
  - Permite comparar el desempeño de distintas arquitecturas de red
- **Prueba**.
  - Se emplea para evaluar el rendimiento del algoritmo, pero no para tomar ninguna decisión con respecto a qué algoritmo de aprendizaje o parámetros usar.
  - El mejor modelo se debe probar en este conjunto.

Reference: Andrew, N. (2020). Khát Khao Học Máy (Machine Learning Yearning).

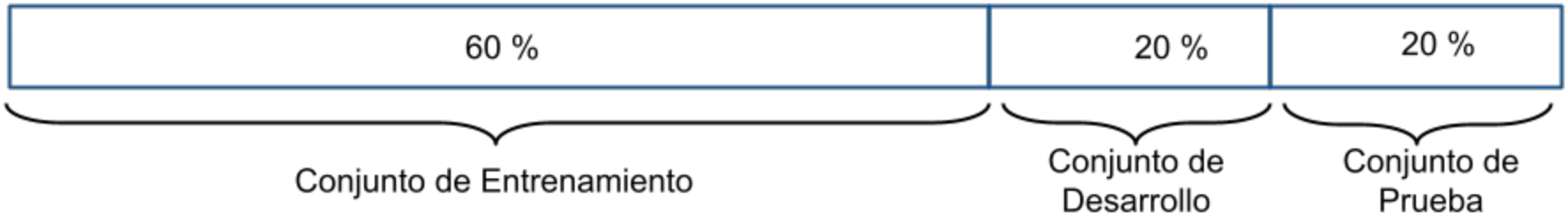
Tamaño de los conjuntos de entrenamiento

Si el conjunto de datos es pequeño, digamos de 100 a 10,000 muestras, generalmente se utiliza una división de 70/30. 70% para el conjunto de entrenamiento y 30% para el conjunto de prueba.



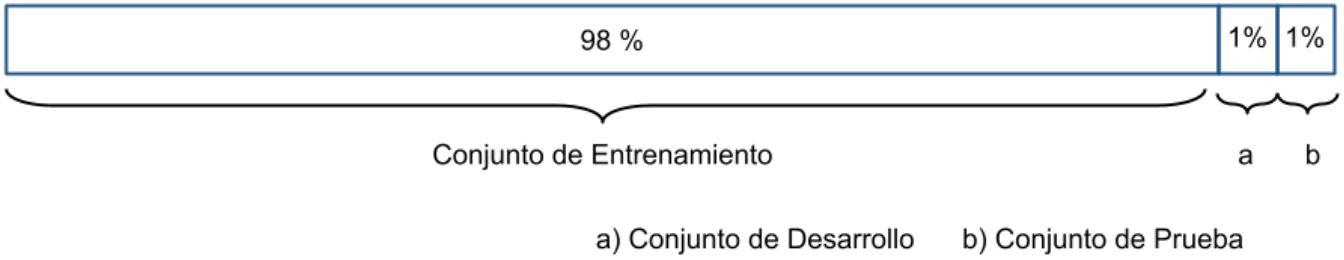
Tamaño de los conjuntos de entrenamiento

O bien utilizar una división de 60/20/20. Dónde, 60% para el conjunto de entrenamiento, 20% para el conjunto de desarrollo y 20% para el conjunto de prueba.



Tamaño de los conjuntos de entrenamiento

Actualmente en la era “big data”, hay una gran cantidad de datos disponibles. Entonces, digamos que el conjunto de datos tiene un millón de muestras. Podría ser bastante razonable configurar los conjuntos de datos como 98/1/1. 98% para el conjunto de entrenamiento, 1% para el conjunto de desarrollo y 1% para el conjunto de prueba.



Tips

- El algoritmo funciona bien en el conjunto de desarrollo pero no en el conjunto de prueba. Si sus conjuntos de desarrollo y prueba provienen de la misma distribución, entonces: el conjunto de desarrollo se ha sobreajustado.
  - La solución podría ser obtener más datos para del conjunto de desarrollo.
- Establecer una métrica de evaluación de un solo número para así poder optimizar el modelo
  - Podría ser la exactitud (accuracy)
  - Otras podrían ser la precisión (precision) y el recuerdo (recall), sin embargo sería mejor escoger una de las formas de combinarlas para obtener solo un valor. Por ejemplo, tomar el promedio de la precisión y el recuerdo. Otro ejemplo sería calcular la métrica F1.

Reference: Andrew, N. (2020). Khát Khao Học Máy (Machine Learning Yearning).

Tips

- Elegir el conjuntos de desarrollo y prueba de una distribución que refleje qué datos se esperan obtener en el futuro y en los que se desea hacerlo bien. Estos conjuntos pueden ser de una distribución distinta al conjunto de datos de entrenamiento
- Elija conjuntos de desarrollo y prueba de la misma distribución si es posible
- El conjunto de desarrollo debe ser lo suficientemente grande como para detectar cambios significativos en la precisión del algoritmo, pero no necesariamente mucho más grande.
- El conjunto de prueba debe ser lo suficientemente grande como para brindar una estimación confiable del rendimiento final del sistema.

Reference: Andrew, N. (2020). Khát Khao Học Máy (Machine Learning Yearning).

Regularización

Regularización

Muchos enfoques de regularización se basan en limitar la capacidad de los modelos, como las redes neuronales, la regresión lineal o la regresión logística, agregando una penalización de norma de parámetro **Ω(θ)** a la función de costo (objetivo) **J**. Denotamos la función objetivo regularizada por **J_hat**:

$$\hat{J}(\theta, X, y) = J(\theta; X, y) + \alpha\Omega(\theta)$$

Donde $\alpha\epsilon[0, \infty)$ es un hiper parámetro que pondera la contribución relativa del término de penalización estándar, **Ω**, en relación con la función objetivo estándar **J**.

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Norma L2

- La penalización de la norma del parámetro L2 o L^2 se conoce comúnmente como caída de peso (weight decay).
- En otras comunidades académicas, la regularización de L2 también se conoce como regresión de cresta o regularización de Tikhonov.
- La regularización L2 regularization fue formulada primero por el matemático sovietico Andrey Tikhonov en 1943
- La idea de la regularización L2 es utilizar la norma euclidiana para el término de regularización.

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Norma L2

- La norma L2 de un vector $\mathrm{x}=(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n)$ es simplemente: $\sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$
  La norma L2 de un vector x se denota $\mathrm{L2}(\mathrm{x})$, o más comúnmente por $\|\mathrm{x}\|_2$.
- Así que ahora se tiene:

$$\hat{J}(\theta, X, y) = J(\theta; X, y) + ||w||_2$$

- En la comunidad de Machine Learning, en lugar de $\|\mathrm{x}\|_2$ se usa el cuadrado de la norma $\mathrm{L2}$, es de $(||w||_2)^2 = ||w||_2^2$ que se puede ver como $\sum_w w^2$.

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Norma L2

- También se agrega un hiper parámetro para poder ajustar la cantidad de regularización que queremos usar (llamado parámetro de regularización o tasa de regularización, y denotado por λ), y dividirlo por el tamaño del lote utilizado (para tener en cuenta el hecho de que queremos que sea proporcional).

$$\hat{J}(\theta, X, y) = J(\theta; X, y) + \frac{\lambda}{n}||w||_2^2 = J(\theta; X, y) + \frac{\lambda}{n}\Sigma_w w^2$$

- Pytorch aplica el “weight decay” tanto a los pesos como al sesgo.

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Norma L2

- La intuición es que durante el procedimiento de aprendizaje, se preferirán pesos más pequeños, pero se considerarán pesos más grandes si la disminución general del error es significativa. Esto explica por qué se llama 'decadencia de peso'.
- La elección de λ determina cuánto se preferirán los pesos pequeños (cuando λ es grande, la preferencia por los pesos pequeños será grande).

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Norma L1

- La regularización L1, “lazo” o “eliminación de ruido de búsqueda básica”, fue propuesta por primera vez por Robert Tibshirani en 1996
- L1 usa el valor absoluto en lugar de los cuadrados como en L2:

$$\hat{J}(\theta, X, y) = J(\theta; X, y) + \frac{\lambda}{n}||w||_1 = J(\theta; X, y) + \frac{\lambda}{n}\Sigma_w|w$$

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Norma L1

- Para la mayoría de los problemas de predicción y clasificación L2 es mejor.
- Sin embargo, L1 es superior para aquellos problemas muchos datos irrelevantes.
    - Estos pueden ser datos muy ruidosos o características que no son informativas, pero también pueden ser datos escasos (donde la mayoría de las características son irrelevantes porque faltan)
- La regularización L1 hará que muchos más pesos sean ligeramente más pequeños, lo que generalmente da como resultado que muchos pesos se acerquen a 0.

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

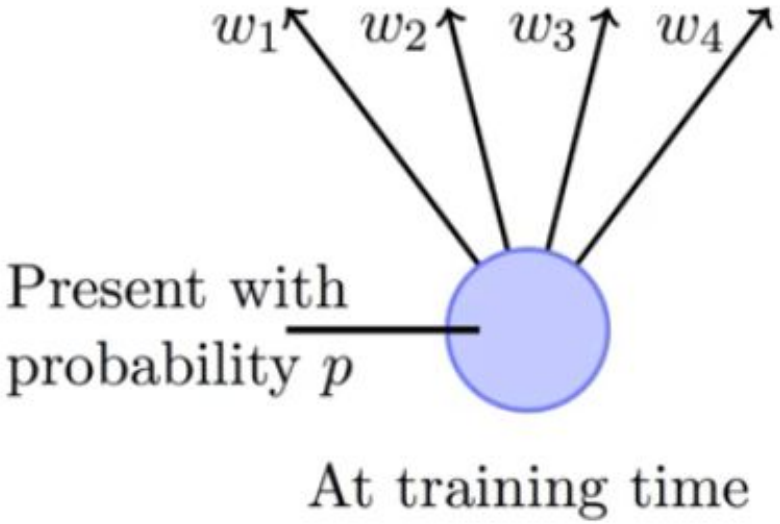
Dropout

- Hemos optado por definir la regularización como la adición de un término de regularización a la función de costo y, de acuerdo con esta definición, el dropout no es regularización, pero reduce la brecha entre el error de entrenamiento y el error de prueba, y en consecuencia, reduce el sobreajuste.
- Se podría definir la regularización como cualquier técnica que reduzca esta brecha, entonces el dropout sería una técnica de regularización.

Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Dropout

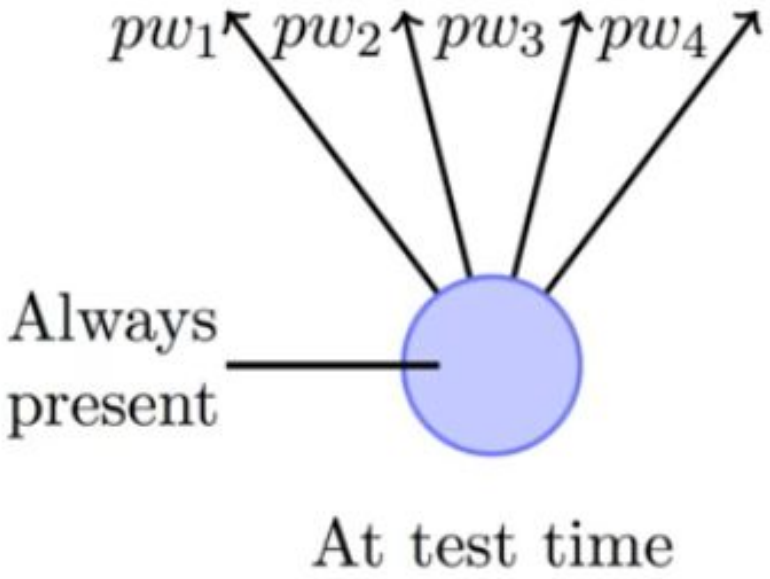
- En cada iteración de entrenamiento, a cada nodo de la red se le asocia una probabilidad ***p*** para permanecer en la red o para desactivarlo (dropout) fuera de la red con probabilidad ***1-p***.
- Eso significa que los pesos asociados con los nodos se actualizan solo ***p*** fracción de veces porque los nodos están activos solo ***p*** veces durante el entrenamiento.



Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Dropout

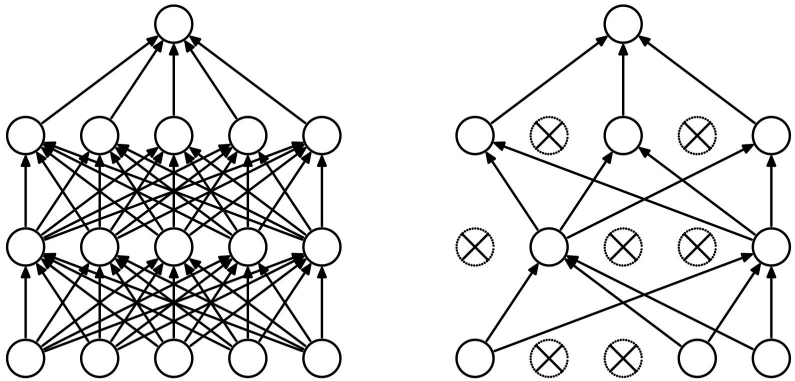
- Durante el test, consideramos la red neuronal original con todas las activaciones presentes y escalamos la salida de cada nodo por un valor p. Ya que cada nodo se activa en p veces únicas.



Reference: Skansi, S. (2018). *Introduction to Deep Learning: from logical calculus to artificial intelligence*. Springer.

Dropout

- El dropout puede reducir drásticamente el sobreajuste y es muy utilizado. Intuitivamente, la razón por la que dropout funciona bien es que evita la coadaptación compleja de las unidades ocultas.



(a) Standard Neural Net (b) After applying dropout.

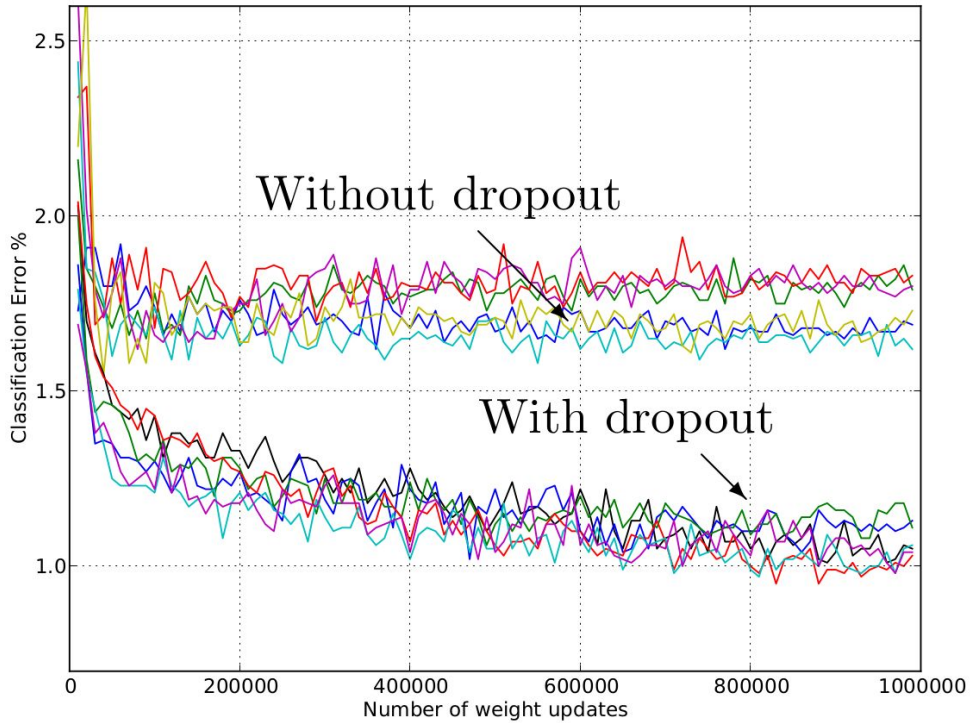
Reference: Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.

Dropout

- Desactiva las neuronas aleatoriamente en cada paso de entrenamiento.
- En lugar de entrenar los datos en la red original, entrenamos los datos en la red con nodos eliminados.
- En la siguiente iteración del paso de entrenamiento, las neuronas ocultas se desactivan por cambios del dropout debido a su comportamiento probabilístico.
- De esta forma, aplicando dropout, es decir, desactivando ciertos nodos individuales al azar durante el entrenamiento, podemos simular un conjunto de redes neuronales con diferentes arquitecturas.

Reference: Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.

Dropout



Reference: Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.

Regularización

Un problema central en el aprendizaje automático es cómo hacer un algoritmo que funcione bien no solo en los datos de entrenamiento, sino también en la predicción de nuevas entradas. Muchas estrategias utilizadas en el aprendizaje automático están diseñadas explícitamente para reducir el error de prueba, posiblemente a expensas de un mayor error de entrenamiento. Estas estrategias se conocen colectivamente como regularización.

Regularización definida como: “cualquier modificación que hacemos a un algoritmo de aprendizaje que pretende reducir su error de generalización pero no su error de entrenamiento”.

Reference: Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.

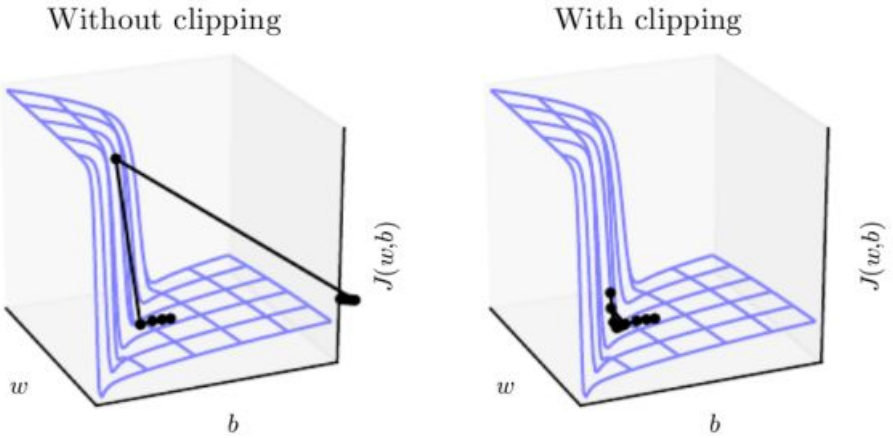
Recorte de Degradado (Gradient Clipping)

- El recorte de gradiente es una técnica para evitar la explosión de gradientes en redes muy profundas, generalmente en redes neuronales recurrentes.
- Hay muchas formas de calcular el recorte de gradiente, pero una común es cambiar la escala de los gradientes para que su norma sea, como máximo, un valor particular.
- Con el recorte de degradado, se introduce un umbral de degradado predeterminado y, a continuación, las normas de degradado que superan este umbral se reducen para que coincidan con la norma.
- Esto evita que cualquier gradiente tenga una norma mayor que el umbral y, por lo tanto, los gradientes se recortan.
- Hay un sesgo introducido en los valores resultantes del degradado, pero el recorte de degradado puede mantener las cosas estables.

Gradient Clipping Definition | DeepAI

Recorte de Degradado (Gradient Clipping)

- Gradient Clipping es un método en el que la derivada del error se cambia o se recorta a un umbral durante la propagación hacia atrás a través de la red y se utilizan los gradientes recortados para actualizar los pesos.
- Al volver a escalar la derivada del error, las actualizaciones de los pesos también se volverán a escalar, lo que disminuirá drásticamente la probabilidad de un desbordamiento o subdesbordamiento.



Understanding Gradient Clipping - neptune.ai

Outliers
(Datos Atípicos)

Outliers

- En estadística, un valor atípico (outlier) es un punto de datos que difiere significativamente de otras observaciones.
- Un valor atípico puede deberse a la variabilidad en la medición o puede indicar un error experimental; estos últimos a veces se excluyen del conjunto de datos.
- Un valor atípico es algo que es diferente o que está separado de la multitud.

Outliers: ¿Cómo se introducen en los conjuntos de datos?

Las causas más comunes de valores atípicos en un conjunto de datos son:

- **Errores de entrada de datos**. Los errores humanos, como los errores causados durante la recopilación, el registro o la entrada de datos, pueden causar valores atípicos en los datos.
- **Error de medición** (errores del instrumento). Es la fuente más común de valores atípicos. Esto se produce cuando el instrumento de medición utilizado resulta defectuoso.
- **Errores experimentales**. Extracción de datos o errores de planificación y/o ejecución de experimentos.
- **Intencional**. Son valores atípicos ficticios hechos para probar los métodos de detección.

Outliers: ¿Cómo se introducen en los conjuntos de datos?

- **Errores de procesamiento de datos**. Manipulación de datos o mutaciones no deseadas del conjunto de datos. Cuando en un proyecto de aprendizaje automático o ciencia de datos, normalmente extraemos datos de diferentes fuentes, esta extracción también puede conducir a errores o valores atípicos.
- **Errores de muestreo**. Cuando se selecciona una muestra que no representa las poblaciones reales, los datos se convertirán en valores atípicos. Ejemplo: seleccionar la altura solo de atletas y se incluyen a los jugadores de baloncesto, esto se convierte en valores atípicos.
- **Valor atípico natural** (No es un error, novedades en los datos). Cuando un valor atípico no es artificial (debido a un error), es un valor atípico natural. La mayoría de los datos del mundo real pertenecen a esta categoría.

Outliers: Tipos de valores atípicos

- **Valores atípicos univariados**. Hay un valor extremo en una sola variable o atributo entre los diferentes atributos o variables.

- **Valores atípicos multivariados**. Hay múltiples valores extremos o inusuales en al menos dos o más atributos o variables.

Outliers: ¿Cómo detectar valores atípicos?

- Visualización
  - Histograma
  - Distribución
  - Box and Whisker Plot
  - Scatter Plot
- Hypothesis Testing
- Z-score
- Z-score Robust (Median absolute deviation)
- I.Q.R (InterQuartile Range)
- DBSCAN Clustering
- Linear Regression Models (PCA, LMS)

Outliers: ¿Qué hacer con los outliers detectados?

Existen algunas técnicas para tratar con los outliers detectados

- Eliminación de muestras
- Transformar los valores
- Imputation
- Tratar por separado

Outliers and Various methods of Detection

Eliminación de muestras

- A veces, es mejor eliminar por completo las muestras del conjunto de datos para evitar que distorsionen el análisis.
- Se eliminan los outliers si se debe a un *error de entrada de datos*, un *error de procesamiento de datos* o si las muestras de los outliers son muy pequeñas.
- También se puede usar el recorte en ambos extremos para eliminar los outliers.
- Pero eliminar la muestra no es una buena idea cuando se tiene un conjunto de datos pequeño.

Outliers and Various methods of Detection

Transformar los valores

- La transformación de variables también puede eliminar valores atípicos.
- Estos valores transformados reducen la variación causada por los valores extremos.
  - Escalar
  - Transformación Log
  - Normalización de raíz cúbica
  - Transformación Box-Cox
- Estas técnicas convierten los valores del conjunto de datos en valores más pequeños.
- Si los datos tienen muchos valores extremos o están sesgados, este método ayuda a que sus datos sean normales.
- Sin embargo estas técnicas no siempre dan los mejores resultados.
- No hay pérdida de datos de estos métodos.
- En todos estos métodos, la transformación de Box Cox da el mejor resultado.

Outliers and Various methods of Detection

Imputación

- Al igual que la imputación de valores faltantes, también se pueden imputar valores atípicos.
- Se puede usar la media, la mediana y el valor cero en este método.
- En la imputación no hay pérdida de datos.
- Aquí la mediana es apropiada porque no se ve afectada por los valores atípicos.

Imputación
Outliers and Various methods of Detection

Tratar por separado

- Si hay un número significativo de valores atípicos y el conjunto de datos es pequeño, se deben tratar por separado en el modelo estadístico.
- Uno de los enfoques es tratar a ambos grupos como dos grupos diferentes y construir un modelo individual para ambos grupos, y luego combinar el resultado.
- Sin embargo, esta técnica es tediosa cuando el conjunto de datos es grande.

Outliers and Various methods of Detection

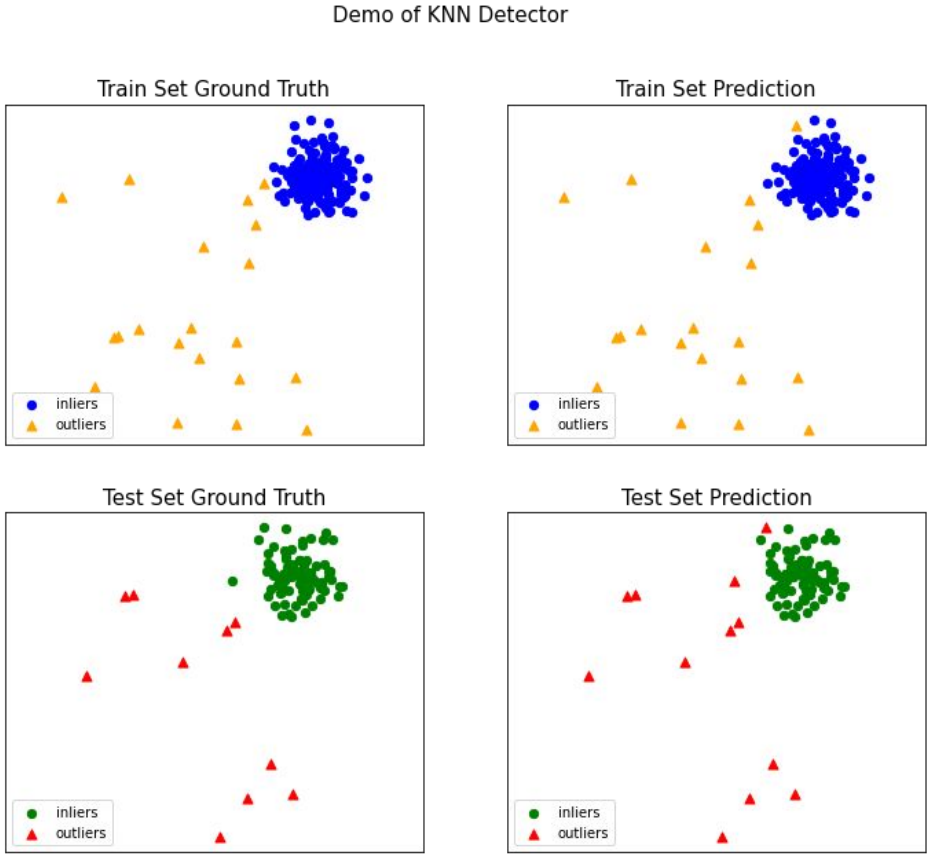
Outliers. ECOD

- Es un algoritmo simple pero efectivo llamado ECOD (Empirical-Cumulative-distribution-based Outlier Detection), que se inspira en el hecho de que los valores atípicos son a menudo los "eventos raros" que aparecen en las colas de una distribución.
- En pocas palabras, ECOD primero estima la distribución subyacente de los datos de entrada de forma no paramétrica mediante el cálculo de la distribución acumulativa empírica por dimensión de los datos.
- ECOD luego usa estas distribuciones empíricas para estimar las probabilidades de cola por dimensión para cada punto de datos.
- Finalmente, ECOD calcula una puntuación atípica de cada punto de datos al agregar las probabilidades de cola estimadas en todas las dimensiones.

ECOD: Unsupervised Outlier Detection Using Empirical Cumulative Distribution Functions

ECOD

- Este algoritmo está implementado en Python Outlier Detection (PyOD).
- Es el conjunto de herramientas de Python más completo y escalable para detectar objetos periféricos en datos multivariados.
- Incluye más de 40 algoritmos de detección, desde LOF clásico (SIGMOD 2000) hasta el último ECOD (TKDE 2022).



ECOD: Unsupervised Outlier Detection Using Empirical Cumulative Distribution Functions

Bias & Variance

Bias, Variance

- Hay dos fuentes principales de error en el aprendizaje automático: el sesgo (bias) y la varianza.
- Si la tasa de error en el conjunto de entrenamiento es del 15% (85% de precisión), pero el objetivo es del 5% de error (95 % de precisión), entonces el primer problema a resolver es mejorar el rendimiento del algoritmo en el conjunto de entrenamiento.
- El rendimiento del conjunto de desarrollo/prueba suele ser peor que el rendimiento del conjunto de entrenamiento.
    - Entonces, si se obtiene un 85% de precisión en las muestras que el algoritmo ha visto, no hay forma de que obtenga un 95% de precisión en muestras que el algoritmo ni siquiera ha visto.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Bias, Variance: Ejemplo

Supongamos que un algoritmo tiene un error del 16% (84% de precisión) en el conjunto de desarrollo. Hay que dividir el error del 16% en dos componentes:

- Primero, la tasa de error del algoritmo en el conjunto de entrenamiento. En este ejemplo, es del 15% (85% de precisión). Esto informalmente se conoce como el **bias** (sesgo) del algoritmo.
- En segundo lugar, qué tan peor funciona el algoritmo en el conjunto de desarrollo (o prueba) en relación con el conjunto de entrenamiento. En este ejemplo, lo hace un 1% (16% - 15% = 1%) peor en el conjunto de desarrollo en relación con el conjunto de entrenamiento. Informalmente esto se conoce como la **varianza** del algoritmo.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Bias & Variance: Ejemplo 1

Considerar un clasificador “ideal” (tal como un humano) que podría lograr un desempeño casi perfecto en la tarea. Supongamos que el algoritmo funciona de la siguiente manera:

- Training error = 1%
- Development error = 11%

Bias = 1%

Variance = 11% - 1% = 10%

Por lo tanto, tiene una **alta varianza**. El clasificador tiene un error de entrenamiento muy bajo, pero no se puede generalizar el conjunto de desarrollo. Esto también se llama sobreajuste (**overfitting**).

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Bias & Variance: Ejemplo 2

Considerar lo siguiente:

- Training error = 15%
- Development error = 16%

Bias = 15%

Variance = 16% - 15% = 1%

Este clasificador se ajusta mal al conjunto de entrenamiento con un error del 15%, pero su error en el conjunto de desarrollo es apenas mayor que el error de entrenamiento. Este clasificador, por lo tanto, tiene un **alto sesgo**, pero una baja varianza. Decimos que este algoritmo es desajustado (**underfitting**).

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Bias & Variance: Ejemplo 3

Considerar lo siguiente:

- Training error = 15%
- Development error = 30%

Bias = 15%

Variance = 30% - 15% = 15%

Este clasificador tiene un **alto sesgo** y una **alta varianza**: lo está haciendo mal en el conjunto de entrenamiento y, por lo tanto, tiene un **alto sesgo**, y su rendimiento en el conjunto de desarrollo es aún peor, por lo que también tiene una **alta varianza**. La terminología de sobreajuste/desajuste es difícil de aplicar aquí ya que el clasificador está sobreajustado y desajustando simultáneamente.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Bias & Variance: Ejemplo 4

Considerar lo siguiente:

- Training error = 0.5%
- Development error = 1%

Bias = 0.5%

Variance = 1% - 0.5% = 0.5%

Este clasificador está funcionando bien, ya que tiene un **sesgo bajo** y una **varianza baja**.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Comparación con la tasa de error óptima

Considerar que se desarrolla un modelo de ML para reconocimiento de voz con 14% de los clips de audio con demasiado ruido de fondo, o son tan ininteligibles que ni siquiera un ser humano puede reconocer lo que se dijo.

En este caso, incluso el sistema de reconocimiento de voz más "óptimo" podría tener un error de alrededor del 14%.

Supongamos que en este problema de reconocimiento de voz, el algoritmo logra:

- Training error = 15%
- Dev error = 30%

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Comparación con la tasa de error óptima

- Este ejemplo es similar al Ejemplo 3, que también tenía un error de entrenamiento del 15% y un error de desarrollo del 30%.
- Si la **tasa de error óptima** es ~0%, entonces un error de entrenamiento del 15% deja mucho margen de mejora.
- Esto sugiere que los cambios para reducir el sesgo podrían ser fructíferos.
- Pero si la **tasa de error óptima** es del 14%, entonces el mismo rendimiento del conjunto de entrenamiento nos dice que hay poco margen de mejora en el sesgo del clasificador.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Comparación con la tasa de error óptima

Continuando con el ejemplo anterior de reconocimiento de voz, el error total del conjunto de desarrollo es del 30%. Se puede desglosar de la siguiente manera (se puede aplicar un análisis similar al error del conjunto de prueba):

- **Tasa de error óptima** ("bias inevitable"). Ocurre cuando un sistema o modelo de ML tiene cierto (14%) de error que no puede mejorarse en el mundo real.
- **Bias evitable**. Esto se calcula como la diferencia entre el error de entrenamiento y la tasa de error óptima (15% - 14% = 1%).
- **Bias** = Tasa de error óptima ("bias inevitable") + bias evitable. (14% + 1% = 15%)
- **Variance**. La diferencia entre el error de desarrollo y el error de entrenamiento. (30% - 15% = 15%)

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Comparación con la tasa de error óptima

- Si el resultado de obtener el **bias evitable** es un número negativo, esto significa que se está sobreajustando el conjunto de entrenamiento y el algoritmo ha memorizado demasiado el conjunto de entrenamiento.
- Para solucionar este hay que centrarse en los métodos de reducción de la varianza en lugar de otros métodos de reducción del sesgo.
- El **bias evitable** refleja que tan peor se desempeña el algoritmo en el conjunto de entrenamiento respecto al "clasificador óptimo".

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Comparación con la tasa de error óptima

- El concepto de varianza sigue siendo el mismo que antes.
- En teoría, siempre se puede reducir la varianza a casi cero entrenando en un conjunto de entrenamiento masivo.
- Por lo tanto, toda variación es "evitable" con un conjunto de datos lo suficientemente grande, por lo que no existe la "variación inevitable".

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Comparación con la tasa de error óptima

Considerar un ejemplo más donde la **tasa de error óptima** es de 14% y se tiene:

- Training error = 15%
- Dev error = 16%

Bias evitable = 15% - 14% = 1%

Variance = 16% - 15% = 1%

Por lo tanto, el algoritmo ya funciona bien, con poco margen de mejora. Es solo un 2% peor que la tasa de error óptima.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Bias evitable y Varianza

- Si se tiene un **alto bias evitable**, se puede aumentar el tamaño del modelo (por ejemplo, aumentar el tamaño de la red neuronal agregando capas/neuronas, o probar con diferentes arquitecturas neuronales).
  - Aumentar el tamaño del modelo generalmente reduce el sesgo, pero también puede aumentar la varianza y el riesgo de sobreajuste.
  - Sin embargo, este problema de sobreajuste generalmente surge sólo cuando no está utilizando la regularización.
  - Si incluye un método de regularización bien diseñado, por lo general puede aumentar de forma segura el tamaño del modelo sin aumentar el sobreajuste.
  - La única razón para evitar el uso de un modelo más grande es el mayor costo computacional.
- Si se tiene una **variación alta**, agregar datos al conjunto de entrenamiento.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

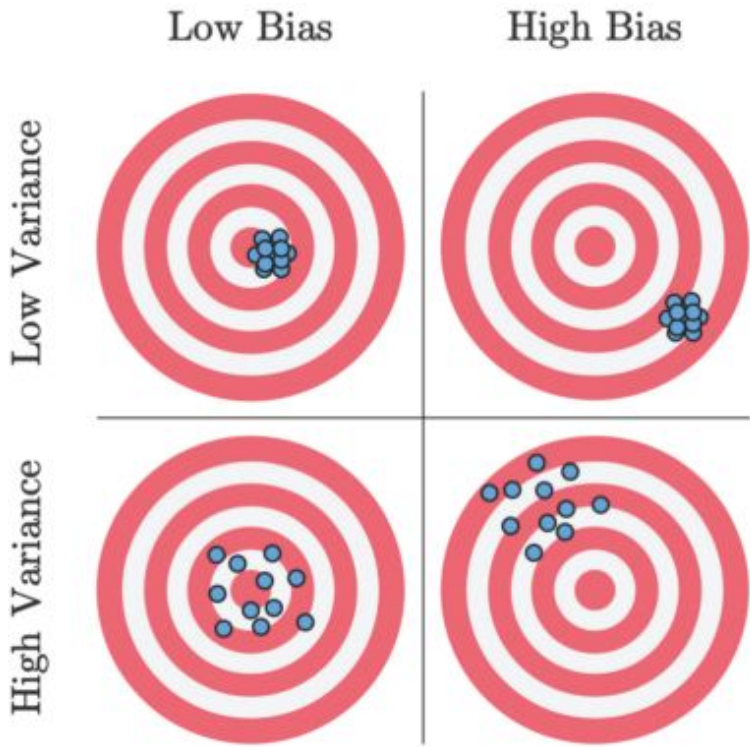
Bias, Variance

High Bias

- Modelo demasiado simplificado.
- Debajo del ajuste (underfitting).
- Alto error en los datos de prueba y entrenamiento.

High Variance

- Modelo demasiado complejo.
- Demasiado ajustado (overfitting).
- Bajo error en los datos del entrenamiento.
- Alto error en los datos de prueba.
- Comienza a modelar el ruido en la entrada.



Técnicas para reducir un Bias evitable alto

- **Aumentar el tamaño del modelo** (como la cantidad de neuronas/capas): esta técnica reduce el sesgo, ya que debería permitir ajustarse mejor al conjunto de entrenamiento. Si se encuentra que esto aumenta la varianza, utilizar la regularización, que normalmente eliminará el aumento de la varianza.
- **Modificar las características de entrada en función de la perspectiva del análisis de errores**: si el análisis de errores inspira a crear características adicionales que ayuden al algoritmo a eliminar una categoría particular de errores. Estas nuevas funciones podrían ayudar tanto con el sesgo como con la varianza. En teoría, agregar más funciones podría aumentar la varianza; pero si se encuentra que este es el caso, utilizar la regularización, que normalmente eliminará el aumento de la varianza.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Técnicas para reducir un Bias evitable alto

- **Reducir o eliminar la regularización** (la regularización L2, regularización L1, dropout): esto reducirá el sesgo evitable, pero aumentará la varianza.
- **Modificar la arquitectura del modelo** (como la arquitectura de la red neuronal) para que sea más adecuada para el problema: esta técnica puede afectar tanto al sesgo como a la varianza.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Técnicas para reducir una Variance alta

- **Agregar más datos de entrenamiento**: esta es la forma más simple y confiable de abordar la varianza, siempre que se tenga acceso a una cantidad significativamente mayor de datos y suficiente poder computacional para procesar los datos.
- **Agregar regularización** (regularización L2, regularización L1, dropout): esta técnica reduce la varianza pero aumenta el sesgo.
- **Agregar early stopping** (es decir, detener el descenso del gradiente antes, en base al conjunto de desarrollo): esta técnica reduce la varianza pero aumenta el sesgo. La detención anticipada se parece mucho a los métodos de regularización y algunos autores la llaman técnica de regularización.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Técnicas para reducir una Variance alta

- **Selección de características para disminuir el número/tipo de características de entrada**: esta técnica podría ayudar con los problemas de varianza, pero también podría aumentar el sesgo.
  - Es poco probable que reducir ligeramente la cantidad de características (por ejemplo, pasar de 1000 funciones a 900) tenga un gran efecto sobre el sesgo.
  - Reducirlo significativamente (por ejemplo, pasar de 1000 funciones a 100, una reducción de 10 veces) es más probable que tenga un efecto significativo, siempre que no excluya demasiadas características útiles.
  - En el aprendizaje profundo moderno, cuando los datos son abundantes, ha habido un cambio en la selección de características, y ahora es más probable que le demos todas las características que tenemos al algoritmo y dejemos que el algoritmo decida cuáles usar en función de los datos.
  - Pero cuando el conjunto de entrenamiento es pequeño, la selección de características puede ser muy útil.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Técnicas para reducir una Variance alta

- **Disminuir el tamaño del modelo** (como el número de neuronas/capas): usar con precaución.
  - Esta técnica podría disminuir la varianza y posiblemente aumentar el sesgo.
  - Sin embargo, Andrew Ng no recomienda esta técnica para abordar la varianza.
  - Agregar regularización generalmente brinda un mejor rendimiento de clasificación.
  - La ventaja de reducir el tamaño del modelo es reducir su costo computacional y, por lo tanto, acelerar la rapidez con la que puede entrenar modelos.
  - Si es útil acelerar el entrenamiento del modelo, entonces considerar disminuir el tamaño del modelo.
  - Pero si el objetivo es reducir la varianza y no se tiene preocupación respecto al costo computacional, se puede considerar el agregar la regularización en su lugar.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.

Técnicas para reducir una Variance alta

- **Disminuir el tamaño del modelo** (como el número de neuronas/capas): usar con precaución.
  - Esta técnica podría disminuir la varianza y posiblemente aumentar el sesgo.
  - Sin embargo, Andrew Ng no recomienda esta técnica para abordar la varianza.
  - Agregar regularización generalmente brinda un mejor rendimiento de clasificación.
  - La ventaja de reducir el tamaño del modelo es reducir su costo computacional y, por lo tanto, acelerar la rapidez con la que puede entrenar modelos.
  - Si es útil acelerar el entrenamiento del modelo, entonces considerar disminuir el tamaño del modelo.
  - Pero si el objetivo es reducir la varianza y no se tiene preocupación respecto al costo computacional, se puede considerar el agregar la regularización en su lugar.
- **Modificar la arquitectura del modelo** (como la arquitectura de la red neuronal) para que sea más adecuada al problema: esta técnica puede afectar tanto al sesgo como a la varianza.

Ng, A. (2017). Machine learning yearning. URL: https://www.deeplearning.ai/programs/ (96), 139.