AI for Medical Diagnosis

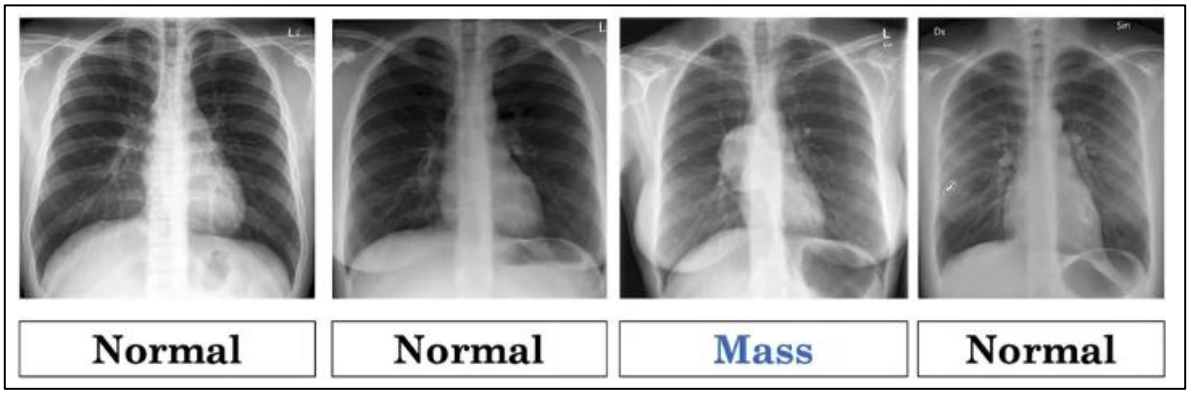
Week 1: Dataset problems

Se explica cómo manejar el desequilibrio de clases y pequeños conjuntos de datos. Como ejemplo se toma un conjunto de datos de radiografía de pecho (Chest X-Ray) para su interpretación (neumonía, cáncer …). Se explican lo siguiente:

PROBLEMAS	**1. Class Imbalance** (Desequilibrio del dataset)	**2. Multi-Task** (Detección múltiple de enfermedades)	**3. Dataset Size** (Pocas muestras en el dataset)
SOLUCIONES	Función de pérdida ponderada / Remuestreo de los datos	Modificación de la función de pérdida para múltiples etiquetas	Transferencia de aprendizaje / Data augmentation

Class Imbalance

En el mundo real la frecuencia de las radiografías de tórax existe más radiografías sanos que no sanos, por lo que no hay los mismos números de ejemplos. Esto produce un desequilibrio del conjunto de datos:



Esto crea un problema para el algoritmo de aprendizaje ya que se encuentra con más ejemplos de radiografías sanos que no sanos. Esto produce que el modelo prediga una probabilidad muy baja de radiografía con enfermedad para todo el mundo y no podrá identificar cuando un ejemplo tiene una enfermedad. Esto se ve mejor con un ejemplo:

Como función de pérdida utilizamos la siguiente:

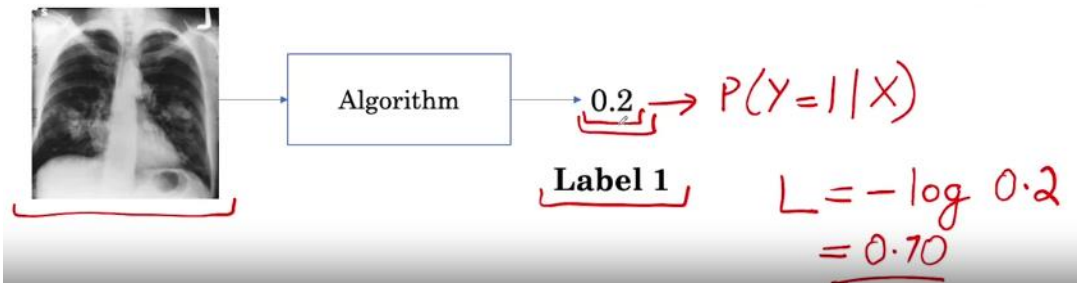
Binary cross-entropy loss

$$L(X,y) = \begin{cases} -\log P(Y=1|X) & \text{if } y=1 \\ -\log P(Y=0|X) & \text{if } y=0 \end{cases}$$

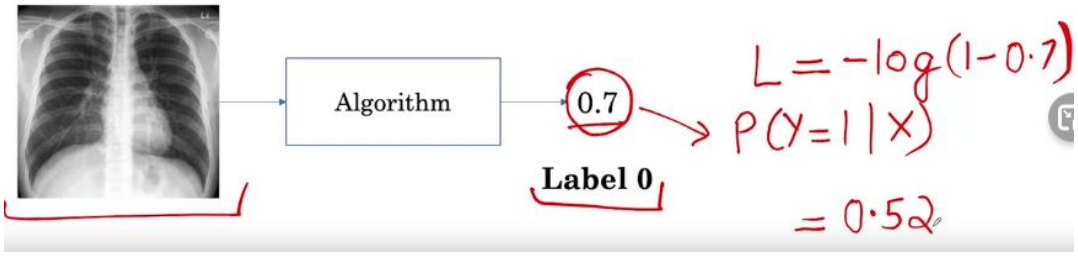
Cuando la etiqueta es 1 (es decir, que tiene una enfermedad)

Cuando la etiqueta es 0 (es decir, que no tiene una enfermedad)

Esta función mide el rendimiento de una clasificación cuya salida es entre 0 y 1. Vemos como se aplica para un ejemplo:



La radiografía SI tiene enfermedad



La radiografía NO tiene enfermedad

Vemos como se aplica para muchos ejemplos (Px es el id del paciente):

Examples	Prediction Probabilities	Loss
P1 Normal	0.5	0.3
P2 Normal	0.5	0.3
P3 Normal	0.5	0.3
P4 Mass	0.5	0.3
P5 Normal	0.5	0.3
P6 Normal	0.5	0.3
P7 Mass	0.5	0.3
P8 Normal	0.5	0.3

$-\log(1-0.5) = 0.3$

$-\log 0.5 = 0.3$

La contribución total a la pérdida de los ejemplos con enfermedad sale a 0.3 por 2, que es 0.6. Mientras que la pérdida total del ejemplo sanos sale a 0.3 veces los 6 ejemplos, que es 1.8.

Por lo que la mayor parte de la contribución a la pérdida proviene de los ejemplos sanos, en lugar de los ejemplos con enfermedad.

Entonces el algoritmo está optimizando para obtener los ejemplos sanos, y sin dar mucho peso relativo a los ejemplos con enfermedad. En la práctica, esto no produce un clasificador muy bueno.

Existen dos soluciones para este problema:

- **Función de pérdida ponderada**
  Modificar la función de pérdida, para ponderar las clases sanos y con enfermedad diferente. W_p será el peso que le asignemos a los ejemplos positivos (con enfermedad), y w_n a los ejemplos negativos (sanos).

$$L(X, y) = \begin{cases} w_p \times -\log P(Y = 1|X) & \text{if } y = 1 \\ w_n \times -\log P(Y = 0|X) & \text{if } y = 0 \end{cases}$$

  Por lo que ahora:

Examples		Loss
P1 Normal		2/8 × 0.3 = 0.075
P2 Normal		2/8 × 0.3 = 0.075
P3 Normal		2/8 × 0.3 = 0.075
P4 Mass	Total Loss From Mass Examples == 0.225 × 2 = 0.45 Total Loss From Normal Examples = 0.075 × 6 = 0.45	6/8 × 0.3 = 0.225
P5 Normal		2/8 × 0.3 = 0.075
P6 Normal		2/8 × 0.3 = 0.075
P7 Mass		6/8 × 0.3 = 0.225
P8 Normal		2/8 × 0.3 = 0.075

  En el caso general, el peso que pondremos en la clase positiva será la cantidad de ejemplos negativos sobre el número total de ejemplos.
  En nuestro caso, esto es 6 ejemplos normales sobre un total de ejemplos.
  El peso que pondremos en la clase negativa será el número de ejemplos positivos sobre el número total de ejemplos, que es 2/8.
  Con esta configuración de w_p y w_n, podemos tener sobre todos los ejemplos para las contribuciones de pérdidas de la clase positiva y negativa para ser la misma.

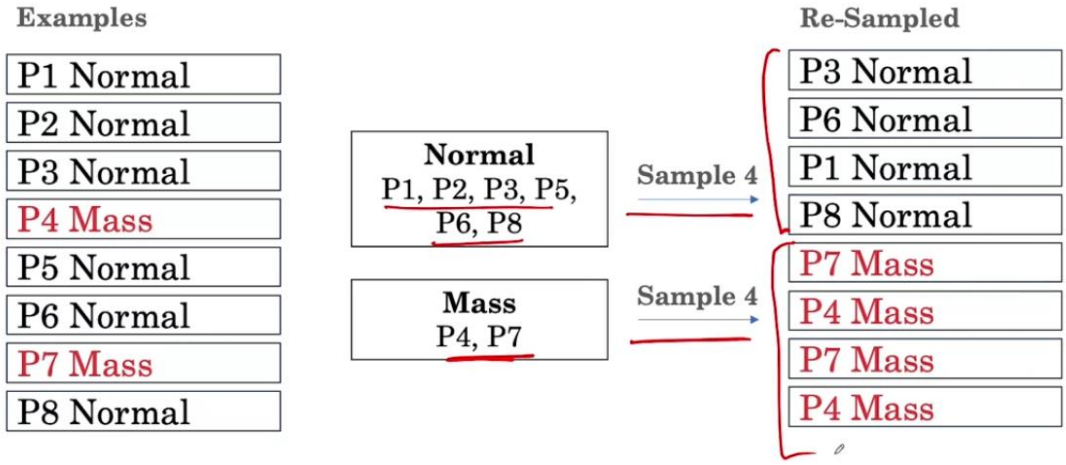
$$L(X, y) = \begin{cases} w_p \times -\log P(Y = 1|X) & \text{if } y = 1 \\ w_n \times -\log P(Y = 0|X) & \text{if } y = 0 \end{cases}$$

$$w_p = \frac{\text{num negative}}{\text{num total}} \qquad w_n = \frac{\text{num positive}}{\text{num total}}$$

Weighted Loss

- **Reemustreo de los datos**
  La idea básica aquí es volver a muestrear el conjunto de datos que tenemos el mismo número de ejemplos sanos y con enfermedades:

  Primero, agrupamos los sanos y los de enfermedad juntos.
  Observe que el grupo normal tiene seis ejemplos y los de enfermedad tiene dos ejemplos.
  Ahora de estos grupos, probaremos las imágenes de modo que haya un número igual de muestras positivas y negativas.



  Podemos hacer esto tomando muestras de la mitad de los ejemplos de la clase con enfermedad y la mitad de los ejemplos de la clase sanos.
  Tenga en cuenta que esto significa que es posible que no podamos incluir todos los ejemplos sanos en nuestra nueva muestra.
  Además, podemos tener más de una copia de los ejemplos con enfermedades en nuestro conjunto de datos re-muestreado.
  Hay muchas variaciones de este enfoque, como submuestrear la clase sano o sobre muestrear la clase con enfermedades, estos enfoques caen bajo la categoría de métodos de remuestreo.

Multi-Task

Sin embargo, tal vez podamos aprender a hacer todas las tareas usando un modelo. Una ventaja de esto es que podemos aprender características que son comunes a identificar más de una enfermedad, permitiéndonos utilizar nuestros datos existentes de manera más eficiente. Esta es la configuración del aprendizaje multitarea.

Entonces, en lugar de que los ejemplos tengan una sola etiqueta, ahora tienen una etiqueta para cada enfermedad en el ejemplo donde 0 denota la ausencia de esa enfermedad y 1 denota la presencia de esa enfermedad.

Examples (mass, pneumonia, edema)	Prediction Probabilities
P1 0, 1, 0	0.3, 0.1, 0.8
P2 0, 0, 1	0.1, 0.1, 0.8
P3 0, 1, 1	0.2, 0.2, 0.7
P4 1, 0, 1	0.6, 0.3, 0.8
P5 1, 1, 1	0.7, 0.7, 0.9
P6 1, 0, 0	0.8, 0.1, 0.2
P7 0, 1, 1	0.3, 0.9, 0.8
P8 0, 0, 0	0.1, 0.1, 0.2

Por ejemplo, para el primero, tenemos una ausencia de enfermedad de masa, presencia de neumonía y ausencia de otra enfermedad, edema.

El modelo ahora tendrá ahora tres salidas diferentes que denota la probabilidad de las tres enfermedades diferentes.

Para entrenar tal algoritmo, también tenemos que hacer la modificación. Modificamos la función de pérdida de modo que miremos el error asociado con cada enfermedad. Podemos representar nuestra nueva pérdida como la suma de las pérdidas por las múltiples enfermedades.

$$L(X, y_{\text{mass}}) + L(X, y_{\text{pneumonia}}) + L(X, y_{\text{edema}})$$

Examples (mass, pneumonia, edema)	Prediction Probabilities	Loss
P1 0, 1, 0	0.3, 0.1, 0.8	0.52 + 1.00 + 0.70
P2 0, 0, 1	0.1, 0.1, 0.8	0.05 + 0.05 + 0.10
P3 0, 1, 1	0.2, 0.2, 0.7	0.10 + 0.70 + 0.15
P4 1, 0, 1	0.6, 0.3, 0.8	0.22 + 0.52 + 0.10
P5 1, 1, 1	0.7, 0.7, 0.9	0.15 + 0.15 + 0.05
P6 1, 0, 0	0.8, 0.1, 0.2	0.10 + 0.05 + 0.10
P7 0, 1, 1	0.3, 0.9, 0.8	0.52 + 0.05 + 0.10
P8 0, 0, 0	0.1, 0.1, 0.2	0.05 + 0.05 + 0.10

Una consideración final es cómo podemos tener en cuenta el desequilibrio de clases en el entorno multitarea. Una vez más, podemos aplicar la pérdida ponderada que hemos cubierto anteriormente.

Esta vez, no solo tenemos un peso asociado solo con el etiquetas positivas y negativas, pero es por la etiqueta positiva asociado con esa clase en particular y la etiqueta negativa asociada con que tareas particulares tales que, para la masa, habrá un camino diferente a la clase positiva. que a la neumonía o al edema.

Multi-Task $L(X, y_{\text{mass}}) + L(X, y_{\text{pneumonia}}) + L(X, y_{\text{edema}})$

$$L(X, y_{\text{mass}}) = \begin{cases} -w_{\text{p,mass}} \log P(Y = 1|X) & \text{if } y = 1 \\ -w_{\text{n,mass}} \log P(Y = 0|X) & \text{if } y = 0 \end{cases}$$

Dataset Size

Para muchos problemas de imágenes médicas, la arquitectura de elección es la red neuronal convolucional, también llamado ConvNet o CNN.

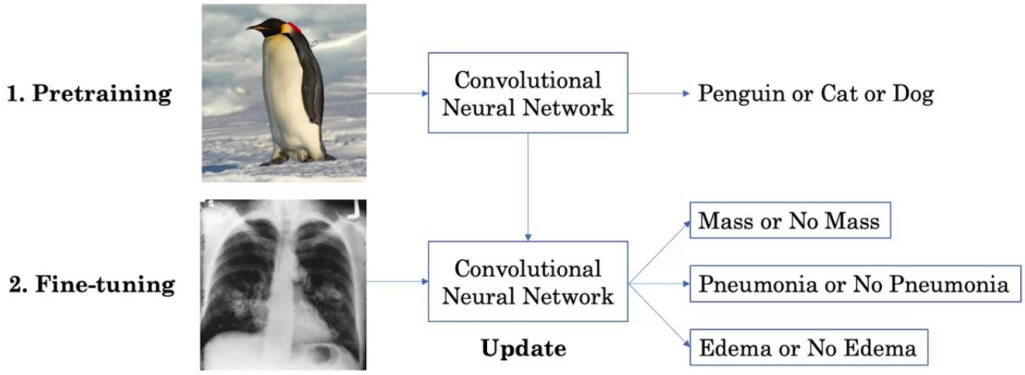
Están diseñados para procesar imágenes en 2D como rayos X. Varias arquitecturas de redes neuronales convolucionales, como Inception, ResNet, DenseNet, ResNeXt y EfficientNets se han propuesto y se muy popular en la clasificación de imágenes.

El desafío es que todas estas arquitecturas necesitan datos y se benefician de los millones de ejemplos que se encuentran en conjuntos de datos de clasificación de imágenes.

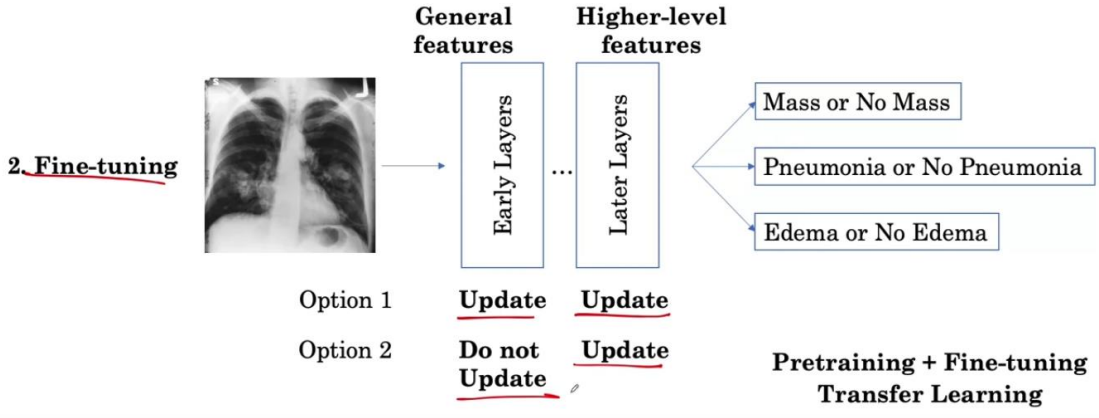
- **Entrenamiento previo y transferencia de aprendizaje**

Una solución es entrenar previamente la red. Aquí la idea es tener primero la red, mira imágenes naturales, y aprender a identificar objetos como pingüinos o gatos, o perros, utiliza esta red como punto de partida para aprender en la tarea de imágenes médicas copiando las características aprendidas.

Luego, la red puede capacitarse aún más para observar radiografías de tórax e identificar la presencia y ausencia de enfermedades. La idea de este proceso es que cuando estamos aprendiendo nuestra primera tarea de identificar gatos o perros, la red aprenderá características generales que le ayudará a aprender sobre la tarea médica.



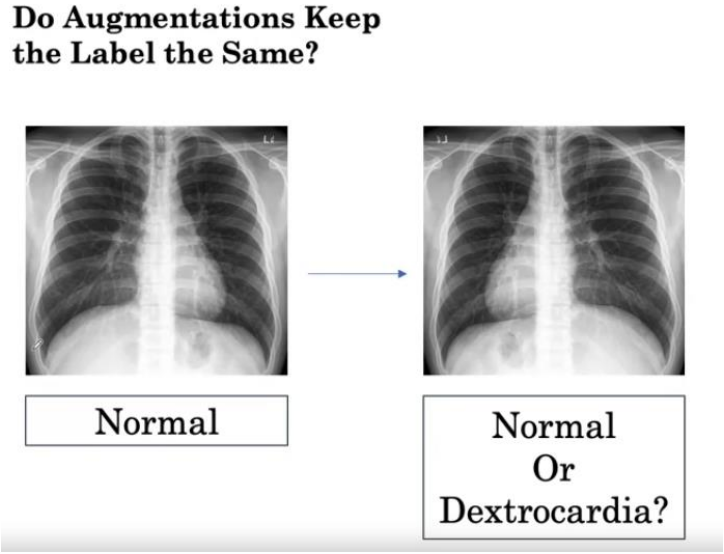
Un ejemplo de esto puede ser que las funciones que son útiles para identificar los bordes de un pingüino también son útiles para identificar bordes en un pulmón, que luego son útiles para identificar ciertas enfermedades. Luego, cuando transferimos este aprendizaje a nuestra nueva red, la red puede aprender la nueva tarea de interpretación de la radiografía de tórax con un mejor punto de partida.



Generalmente se entiende que las primeras capas de la red, es para capturar características de imagen de bajo nivel que son ampliamente generalizables, mientras que las capas posteriores capturan detalles que son más de alto nivel o más específico para una tarea.
Por ejemplo, la primera capa podría aprender sobre los bordes de un objeto, y esto podría ser útil para interpretación de la radiografía de tórax más tarde.

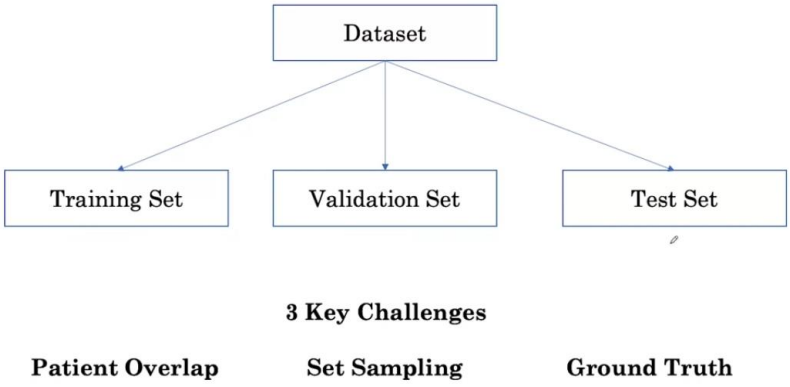
Pero las capas posteriores, podría aprender a identificar a la cabeza de un pingüino y puede que no sea útil para la interpretación de radiografías de tórax.
Entonces, cuando ajustamos la red en radiografías de tórax, en lugar de ajustar todas las funciones que hemos transferido, podemos congelar las funciones aprendidas por las capas poco profundas y simplemente ajuste las capas más profundas.
En la práctica, dos de las opciones de diseño más comunes son una, para afinar todas las capas, y dos, solo afine el último o la última capa y no afinar las capas anteriores.

- **Data augmentation**
  La idea es engañar la red en pensar que tenemos más ejemplos de formación que son eliminación de lo que realmente hacemos. Justo antes de pasar una imagen de rayos X a la red, podemos aplicarle una transformación.
  Tenemos varias opciones aquí: podemos rotarlo y pasarlo, o podemos traducirlo de lado y pasarlo, o podemos acercarnos, o podemos cambiar el brillo o el contraste, o aplicar una combinación de estas transformaciones.



  Una segunda opción de diseño es verificar que nuestra transformación mantiene la etiqueta igual. Por ejemplo, si estamos invertir lateralmente la radiografía de un paciente, esto significa dar la vuelta a la izquierda para la derecha y la derecha a la izquierda, entonces su corazón lo haría aparecen en el lado izquierdo de la imagen.
  Sin embargo, la etiqueta de sano ya no se mantendría porque esto es en realidad una rara afección cardíaca llamada dextrocardia en la que tu corazón está hacia el lado derecho de su pecho en lugar de hacia el lado izquierdo.
  Entonces esta no es una transformación que conserva la etiqueta. La clave aquí es que queremos que la red aprenda a reconocer imágenes con estas transformaciones que todavía tienen la misma etiqueta, no uno diferente.

Más allá de los rayos X, hay otros procedimientos útiles de aumento de datos para otras tareas. **La rotación y el volteo son útiles para algoritmos entrenados para detectar el cáncer de piel**, por ejemplo.



Patient Overlap: Digamos que un paciente viene dos veces para hacerse una radiografía, una en junio y otra en noviembre. En ambas ocasiones, llevan un collar cuando les toman las radiografías. Se toma una muestra de una de sus radiografías como parte del conjunto de entrenamiento y el otro como parte de la prueba. Entrenamos nuestro modelo y encontrar que predice correctamente. El problema es que es posible que el modelo realmente haya memorizado cuando vio al paciente con un collar.

Estos no son modelos hipotéticos de aprendizaje profundo puede memorizar involuntariamente datos de entrenamiento, y el modelo podría memorizar raras o aspectos de datos de entrenamiento únicos del paciente, como el collar, lo que podría ayudarlo a obtener la respuesta correcta al realizar la prueba en el mismo paciente. Esto conduciría a un rendimiento del conjunto de prueba demasiado optimista, donde pensaríamos que nuestro modelo es mejor de lo que realmente es.

Para abordar este problema en nuestro conjunto de datos, podemos asegurarnos de que las radiografías de un paciente solo se produzcan en uno de los conjuntos (entrenamiento, validación o prueba). Ahora, si el modelo memoriza el collar del paciente, no ayuda a lograr un mayor rendimiento en el equipo de prueba porque no ve al mismo paciente.

Set Sampling: es que cuando estamos al azar muestrear un conjunto de prueba de cientos de ejemplos del conjunto de datos, es posible que no muestreemos a ningún paciente que realmente tenga una enfermedad. Aquí, es posible que no muestreemos ningún ejemplo en el que la etiqueta con enfermedad sea 1. Por lo tanto, no tendríamos forma de probar el rendimiento del modelo en

estos casos positivos. Esto es especialmente un problema con los datos médicos donde ya tienen un pequeño conjunto de datos y no muchos ejemplos de cada enfermedad.

Una forma de abordar esto al crear conjuntos de prueba es muestrear un conjunto de prueba de manera que tengamos al menos un X% de ejemplos de nuestra clase minoritaria. Aquí, la clase minoritaria es simplemente la clase para que tenemos algunos ejemplos como aquí ejemplos donde está presente la masa. Una opción común de X es 50%.

Ground Truth: Una pregunta importante al probar un modelo es cómo determinamos la etiqueta correcta para un ejemplo. La etiqueta correcta se denomina más comúnmente 'Ground Truth' en el contexto de aprendizaje automático. En una radiografía de tórax, diferenciar entre algunas enfermedades puede resultar complejo. Podríamos hacer que un experto diga que esto es neumonía, otros expertos dicen que es otra enfermedad.

Entonces, ¿cómo determinamos la enfermedad en presencia de desacuerdo entre observadores?

Podemos utilizar el método de votación por consenso. La idea detrás de la votación por consenso es utilizar un grupo de expertos humanos, podemos hacer que tres radiólogos observen una radiografía de tórax y cada uno determina si hay neumonía presente o no. Si dos de los tres dicen que sí, entonces diríamos que la respuesta es sí. Alternativamente, podemos hacer que los tres radiólogos entren en una habitación y discutir su interpretación hasta que lleguen a una sola decisión. Una prueba más definitiva que se puede realizar es una tomografía computarizada.

Week 2: Evaluating models

Al calcular la precisión en un conjunto de prueba, miramos la proporción de los ejemplos totales que el modelo clasificó correctamente. Es decir, nº de ejemplos clasificados correctamente / nº total de ejemplos. Veamos un ejemplo, hay 2 modelos los cuales tienen como precisión 0.8:

Ground Truth	Model 2	Model 1
Normal	-	-
Normal	+	-
Normal	-	-
Normal	-	-
Normal	-	-
Disease	+	-
Normal	-	-
Disease	+	-
Normal	+	-
Normal	-	-

Aunque ambas tengan la misma precisión, tenemos la sensación de que el modelo 2 quizás haciendo algo más útil que modelo 1 porque es al menos intentando distinguir entre pacientes sanos y enfermos. Veamos la precisión con más detalle:

Accuracy

Interpretamos la precisión como probabilidad de ser correcto. Podemos descomponer esta probabilidad de ser correcto como la suma de dos probabilidades:

La probabilidad de que el modelo sea correcto y un paciente tiene la enfermedad + la probabilidad de que el modelo es correcto y el paciente sano.

$$\text{Accuracy} = P(\text{correct})$$
$$\text{Accuracy} = P(\text{correct} \cap \text{disease}) + P(\text{correct} \cap \text{normal})$$

La ley de la probabilidad condicional nos permite expandir aún más esto.

La ley de probabilidad condicional dice que la probabilidad de A y B es la probabilidad de A dado B multiplicado por la probabilidad de B.

$$\text{Accuracy} = P(\text{correct})$$
$$\text{Accuracy} = P(\text{correct} \cap \text{disease}) + P(\text{correct} \cap \text{normal})$$

Using $P(A \cap B) = P(A \mid B)\,P(B)$

$$\text{Accuracy} = P(\text{correct} \mid \text{disease})P(\text{disease}) + P(\text{correct} \mid \text{normal})P(\text{normal})$$
$$\text{Accuracy} = P(+ \mid \text{disease})P(\text{disease}) + P(- \mid \text{normal})P(\text{normal})$$

Entonces podemos reemplazar esto por la probabilidad de que predecir positivo dado que un paciente tiene la enfermedad.

Sensibilidad

De manera similar, la probabilidad de acertar cuando el paciente es sano significa que predijimos negativo, para que podamos reemplazar esto por la probabilidad de predecir negativo dado un paciente es sano.

Especificidad

Sensitivity & Specificity

Vemos que a partir de la precisión se deriva otras métricas de evaluación muy importantes como sensibilidad y especificidad.

$P(+ \mid \text{disease})$	$P(- \mid \text{normal})$
If a patient has the disease, what is the probability that the model predicts positive?	If a patient is normal, what is the probability that the model predicts negative?
Sensitivity	**Specificity**

La sensibilidad es la probabilidad de que el modelo clasifique un paciente como teniendo la enfermedad dado que tienen la enfermedad.

La especificidad es la probabilidad de que el modelo clasifique un paciente como sano dado que son sanos. El resto de la fórmula:

$$\text{Accuracy} = \text{Sensitivity} \times \underbrace{P(\text{disease})}_{\text{Prevalence}} + \text{Specificity} \times \underbrace{P(\text{normal})}_{1 - P(\text{disease})}$$

La probabilidad de que un paciente tenga una enfermedad en una población se llama **prevalencia**.

La probabilidad de ser sano es simplemente uno menos la probabilidad de tener una enfermedad o uno menos la prevalencia.

Por lo que podemos reescribir la precisión que permite encontrar cualquiera de estas cantidades dadas las otras tres cantidades:

$$\text{Accuracy} = \text{Sensitivity} \times \text{prevalence} + \text{Specificity} \times (1 - \text{prevalence})$$

Veamos un ejemplo:

Ground Truth	Model
Normal	-
Normal	-
Disease	+
Normal	-
Normal	-
Disease	-
Normal	-
Disease	+
Normal	+
Normal	-

Sensitivity

$$P(+ \mid \text{disease})$$

$$\frac{\#(+ \text{ and disease})}{\#(\text{disease})} = \frac{2}{3} = 0.67$$

Specificity

$$P(- \mid \text{normal})$$

$$\frac{\#(- \text{ and normal})}{\#(\text{normal})} = \frac{6}{7} = 0.86$$

La sensibilidad se puede calcular como la fracción de ejemplos de enfermedades que también son positivos y la especificidad se puede calcular como la fracción de ejemplos normales que también

Prevalence

$$P(\text{disease})$$

$$\frac{\#(\text{disease})}{\#(\text{total})} = \frac{3}{10} = 0.3$$

Accuracy

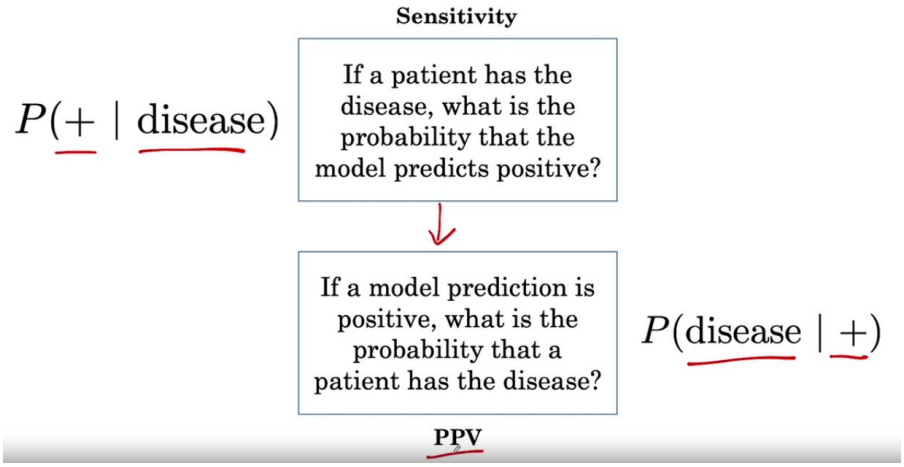
$$\text{Sensitivity} \times \text{prevalence} + \text{Specificity} \times (1 - \text{prevalence})$$

$$= 0.67 \times 0.3 + 0.86 \times 0.7 = 0.8$$

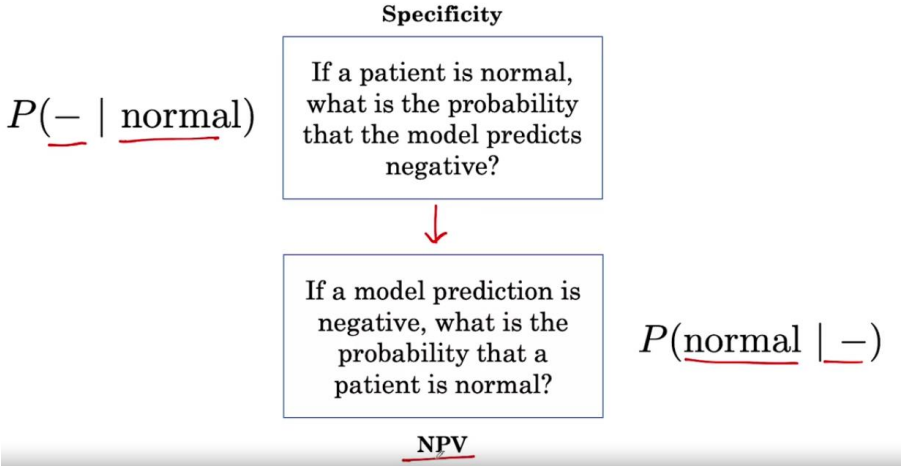
son negativos. Calculamos la prevalencia y finalmente, a partir de estas tres métricas calculamos la precisión del modelo.

PPV & NPV

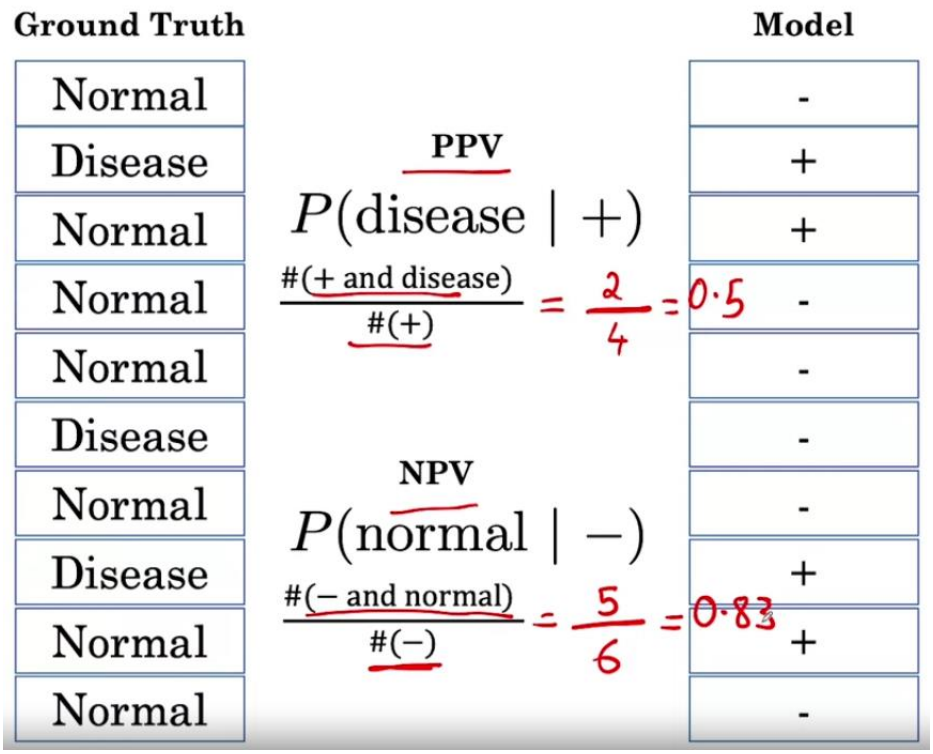
La sensibilidad nos dice, dado que sabemos que un paciente tiene una enfermedad, ¿Cuál es la probabilidad de que el modelo prediga positivo? En la clínica, el médico que utiliza un modelo de IA puede estar interesado en una pregunta diferente. Y eso se da, el modelo predice positivo en un paciente, ¿Cuál es la probabilidad de que realmente tengan la enfermedad? Esto se llama **valor predictivo positivo** o PPV del modelo.



Del mismo modo, mientras que la especificidad pregunta, ¿Cuál es la probabilidad de que el modelo prediga negativo, dado que un paciente es normal? El médico puede estar interesado en conocer la probabilidad de que un paciente sea normal, dado que la predicción del modelo es negativa. Esto se llama **valor predictivo negativo** o NPV de un modelo.



Veamos esto en el ejemplo:



El PPV se puede calcular como la fracción de ejemplos positivos que también son enfermedad. El NPV se puede calcular como la fracción de ejemplos negativos que también son normales. Veamos como ambos se relacionan:

Confusion matrix

Para ver su relación, usaremos la matriz de confusión. La matriz de confusión se puede utilizar para observar el rendimiento de un clasificador en forma de tabla. Cada columna de la matriz representa el número de predicciones de cada clase, mientras que cada fila representa a las instancias en la clase real., o sea en términos prácticos nos permite ver qué tipos de aciertos y errores está teniendo nuestro modelo a la hora de pasar por el proceso de aprendizaje con los datos.

Ground Truth	Model
Normal	-
Disease	+
Normal	+
Normal	-
Normal	-
Disease	-
Normal	-
Disease	+
Normal	+
Normal	-

GT	Model Output: +	Model Output: -
Disease	2	1
Normal	2	5

GT	Model Output: +	Model Output: -
Disease	True Positive (TP)	False Negative (FN)
Normal	False Positive (FP)	True Negative (TN)

Hemos visto las fórmulas para los cálculos de cada de estas métricas en términos de recuentos:

GT	Model Output: +	Model Output: -
Disease	True Positive (TP)	False Negative (FN)
Normal	False Positive (FP)	True Negative (TN)

$$\frac{\#(+ \text{ and disease})}{\#(\text{disease})} = \text{Sensitivit}$$

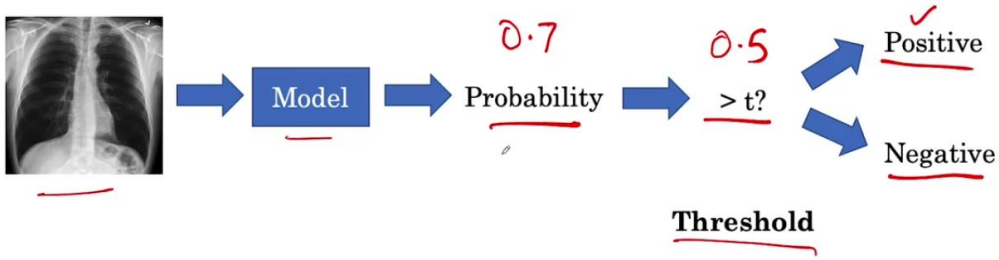
$$\frac{\#(- \text{ and normal})}{\#(\text{normal})} = \text{Specificity}$$

$$\frac{\#(+ \text{ and disease})}{\#(+)} = \text{PPV}$$

$$\frac{\#(- \text{ and normal})}{\#(-)} = NPV$$

ROC Curve

Veremos cómo la curva ROC nos permite trazar visualmente la sensibilidad de un modelo frente a la especificidad del modelo en diferentes umbrales de decisión.



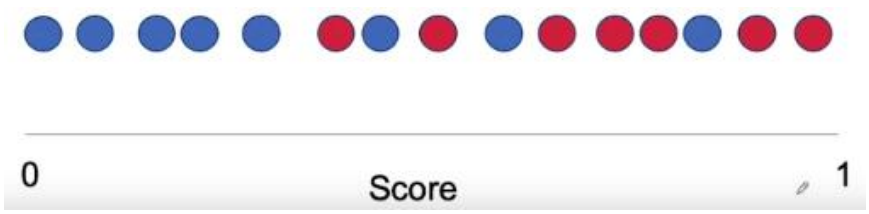
La salida se puede transformar en un diagnóstico utilizando un umbral. Cuando la probabilidad está por encima de un umbral, interpretamos esto como positivo o diciendo que el paciente tiene la enfermedad. Cuando la probabilidad está por debajo del umbral, lo interpretamos como negativo o diciendo que el paciente no tiene la enfermedad.

Nuestra elección de umbral afecta las métricas que hemos analizado hasta ahora. Por ejemplo, si tuviéramos un umbral t de 0 luego clasificaríamos todo como positivo. Y entonces nuestra sensibilidad sería una mientras que nuestra especificidad sería cero. Del mismo modo, si hubiéramos elegido un umbral de uno, clasificaríamos todo como negativo, por lo que nuestra especificidad sería uno mientras que nuestra sensibilidad sería cero.

Veamos cómo afecta la elección del umbral a las métricas mencionadas: digamos que tenemos un conjunto de prueba de 15 radiografías de tórax, que analizamos nuestro modelo para obtener una probabilidad de salida o una puntuación para cada uno de ellos:

X-Ray	Output Probability (Score)
1	0.30
2	0.42
3	0.78
...	...
15	0.98

Podemos graficar estas 15 puntuaciones de salida en una recta numérica entre cero y uno.

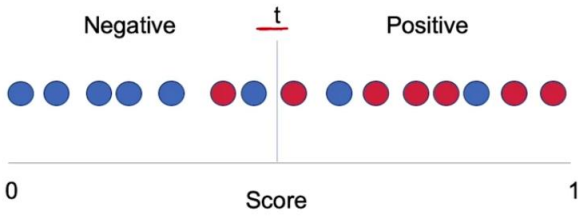


Aquí los ejemplos de enfermedad son rojos y los sanos es azul. Podemos elegir un umbral pequeño t, que pone todo a la derecha del umbral mientras clasificar positivo y todo a la izquierda del umbral, clasificamos como negativos.

$P(+ \mid \text{disease})$ $P(- \mid \text{normal})$

Sensitivity **Specificity**

$\frac{6}{7} = 0.85$ $\frac{6}{8} = 0.75$

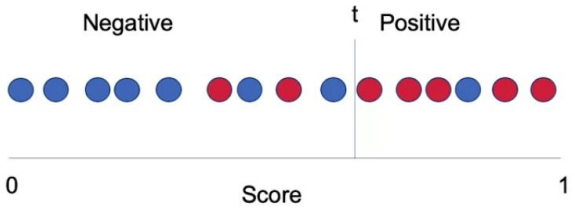


Si escogemos otro umbral:

$P(+ \mid \text{disease})$ $P(- \mid \text{normal})$

Sensitivity **Specificity**

$\frac{5}{7} = 0.71$ $\frac{7}{8} = 0.88$



Tenga en cuenta que la sensibilidad ha bajado, nuestro numerador ha caído, y la especificidad ha subido, nuestro numerador ha aumentado porque ahora estamos clasificando más correctamente pacientes sanos y más incorrectamente clasificando pacientes con enfermedad.

Por ejemplo, en un hospital hay 50.000 pacientes queremos saber la precisión del modelo, si pudiéramos, nuestro modelo da una precisión de 0.78. Sin embargo, esto es inviable ya que es muy difícil calcular para toda una población (denotada por *p*) por lo que se coger una muestra, por ejemplo, p = 100 pacientes, y el modelo nos da una precisión de 0.80.

Confidence intervals

Los intervalos de confianza nos permiten decir que, usando nuestra muestra, tenemos un 95 por ciento de confianza en que la precisión de la población *p* está en el intervalo 0.72 (llamado *lower bound)* y 0.88 (llamado *upper bound*).

0.80 (95% CI 0.72, 0.88)

Interpretation of 95% confidence interval

With 95% confidence, p is in the interval [0.72, 0.88]

Mis-interpretation

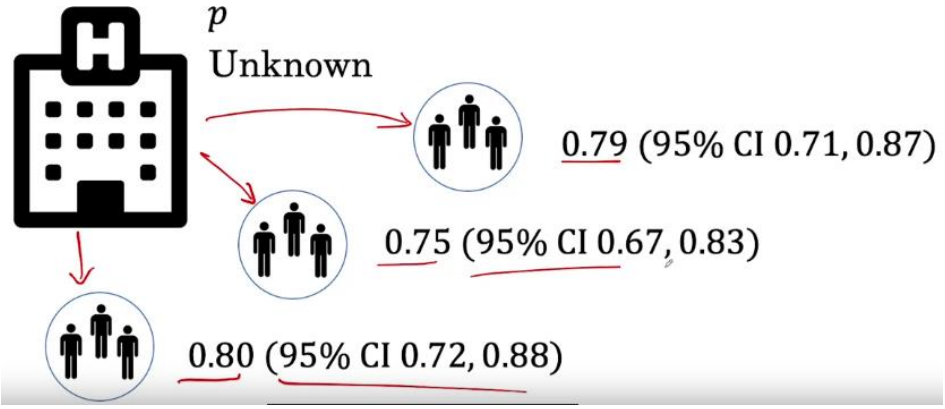
✗ There is a 95% probability that p lies within the interval [0.72, 0.88] ✗

Mis-interpretation

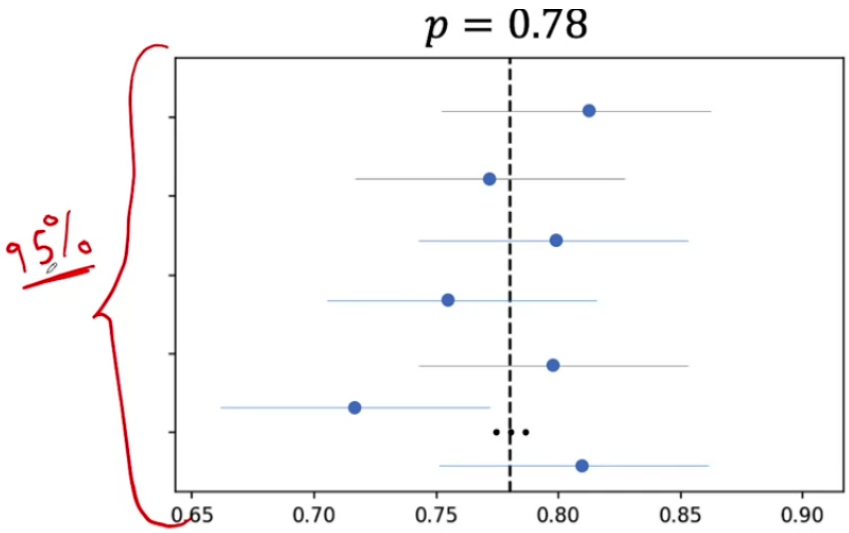
✗ 95% of the sample accuracies lie within the interval [0.72, 0.88] ✗

- 95 por ciento de confianza no dice que haya una probabilidad del 95 por ciento de que *p* se encuentre dentro del intervalo.
- Tampoco dice que el 95 por ciento de las precisiones de la muestra se encuentran dentro de este intervalo.

Digamos que pudimos muestrear repetidamente en 100 pacientes de la población varias veces. Cada vez que obtenemos una muestra diferente, por lo que una precisión de muestra diferente para cada una, y calculando los intervalos de confianza.



Si representamos las muestras de la siguiente manera: donde los puntos son las precisiones y las líneas los intervalos de confianza siendo la línea discontinua la precisión de la población *p.*

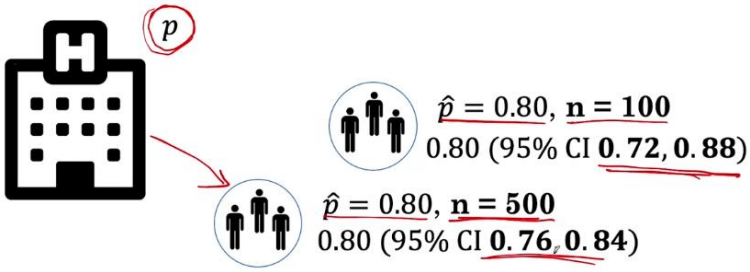


Vemos que en todas las muestras (menos en una), la precisión de p se encuentra dentro del intervalo de confianza.

Por tanto, la interpretación de 95 por ciento de confianza es que, en muestreo repetido, este método produce intervalos que incluyen la precisión de la población en aproximadamente el 95 por ciento de las muestras.

Sin embargo, en la práctica, el modelo solo se entrena a una muestra por lo que nuestro intervalo de confianza puede contener o no *p* (precisión de la población de 50.000) pero podemos estar 95% que lo estará.

El tamaño de la muestra afecta a lo cerca que están los números de los intervalos. Digamos que extrajimos otra muestra de la población, pero esta vez con 500 pacientes. Esto es 5 veces más grande que nuestra muestra anterior. Podemos esperar que tengamos una mejor estimación de la precisión de la población utilizando la muestra más grande.

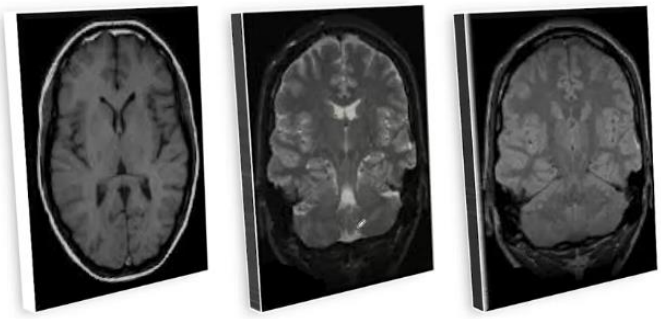


Podemos ver que a pesar de que el modelo obtiene una precisión de 0.8 en ambas muestras, observe que los intervalos de confianza son más estrictos para la muestra más grande y ancha para la muestra más pequeña.

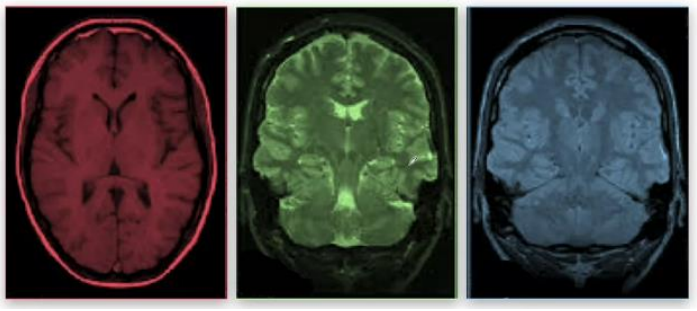
Por tanto, una muestra más grande nos da una mejor estimación de esta precisión de la población porque estos números están más cerca.

Week 3: Medical Image Segmentation

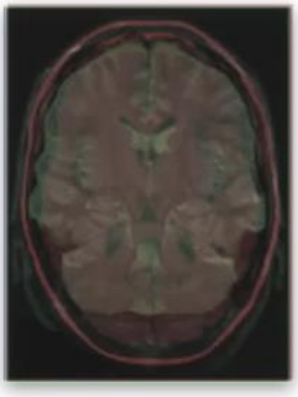
Un ejemplo de resonancia magnética estar formado por múltiples secuencias, y esto consistirá en múltiples volúmenes 3D. Veremos cómo podemos combinar estos múltiples volúmenes 3D en un volumen 3D.



Para ello cogemos una parte del cerebro visto en tres secuencias de resonancia magnética diferentes. La idea clave de que lo haremos utilizar para combinar la información de diferentes secuencias es para trátelos como canales diferentes.

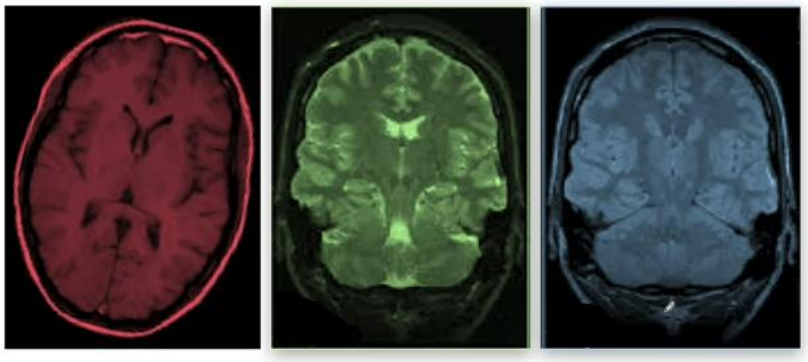


Una vez que cada secuencia es representada con el canal diferente, lo que hacemos ahora es combinar las secuencias juntas para producir una imagen, que es la combinación de todas las secuencias.

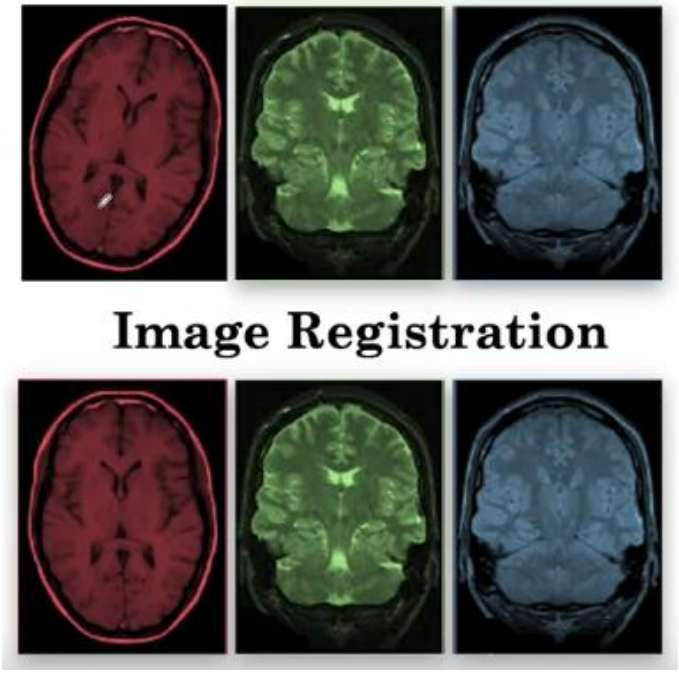


Combine Channels

Un problema con combinar estas secuencias es que pueden no estar alineados entre sí.



Un enfoque de preprocesamiento que se usa a menudo arreglar esto se llama registro de imágenes.

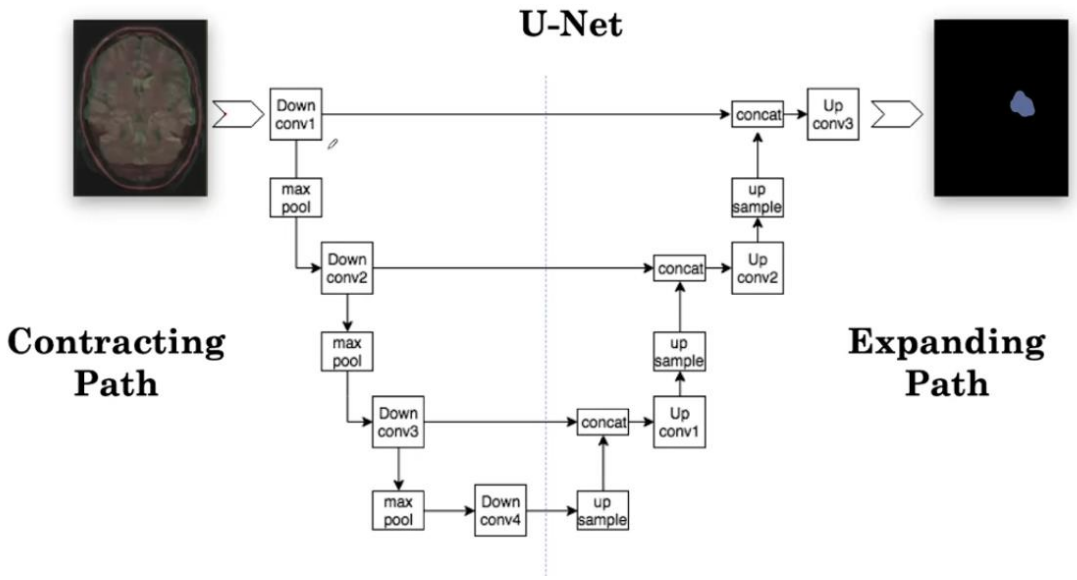


La idea básica con el registro de imágenes es transformar las imágenes para que están alineados o registrados entre sí.

Segmentation Architectures

Para imágenes 2D:

Una de las arquitecturas más populares para la segmentación ha sido la U-Net. La arquitectura U-Net debe su nombre a una forma de U. La U-Net consta de dos caminos: un camino de contracción, y un camino en expansión:



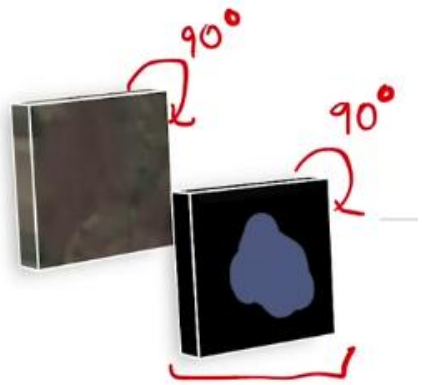
La **ruta de contracción** es una red convolucional típica como se usa en la clasificación de imágenes. Consiste en la aplicación repetida de operaciones de convolución y agrupación. La operación de convolución aquí se llama *convolución hacia abajo*. La clave aquí es que, en la ruta de contracción, nuestros mapas de características se vuelven espacialmente más pequeños, por eso se llama contracción.

El **camino de expansión** de alguna manera está haciendo lo contrario de la ruta de contracción. Está tomando nuestros pequeños mapas de características a través de una serie de muestreo ascendente y pasos de convolución ascendente para volver al tamaño original de la imagen. También concatena las representaciones de muestra ascendente en cada paso con los correspondientes mapas de características en la vía de contracción.

Finalmente, en el último paso, la arquitectura genera la probabilidad de tumor por cada píxel de la imagen. La arquitectura U-Net se puede entrenar en pares de entrada y salida de cortes 2D en el enfoque 2D.

Data augmentation for segmentation

Hablemos de una técnica que podemos aplicar al entrenamiento de dicho modelo, aumento de datos (*data augmentation*).



Una diferencia clave con el aumento de datos durante la segmentación ahora tenemos una **salida de segmentación**. Entonces, cuando giramos una imagen de entrada 90 grados para producir una entrada transformada. También necesitamos rotar las segmentaciones de salida en 90 grados. para obtener nuestra segmentación de salida transformada.

La segunda diferencia es que ahora tenemos volúmenes 3D en lugar de imágenes 2D. Entonces, las transformaciones deben aplicarse a todo el volumen 3D.

Lo último que debemos analizar es la función de pérdida. En nuestra función de pérdida, queremos poder especificar el error deberíamos asignar un ejemplo, dada la predicción del modelo y la etiqueta verdadera. Por ejemplo:

P			**G**			
p_1 0.1	p_2 0.1	p_3 0.1	g_1 0	g_2 0	g_3 0	1 (Tumor)
p_4 0.8	p_5 0.9	p_6 0.9	g_4 0	g_5 1	g_6 1	0 (Normal Brain Tissue)
p_7 0.1	p_8 0.4	p_9 0.1	g_7 0	g_8 1	g_9 0	

Aquí P representa la salida del modelo de segmentación en 9 píxeles. En cada ubicación, tenemos la probabilidad predicha de tumor. G especifica la etiqueta de verdad en cada una de estas ubicaciones de píxeles. Tres de los nueve píxeles son tumores representados como 1, y los seis restantes son tejido cerebral normal representado como 0.

i	p	g
1	0.1	0
2	0.1	0
3	0.1	0
4	0.8	0
5	0.9	1
6	0.9	1
7	0.1	0
8	0.4	1
9	0.1	0

Representando P y G en esta tabla nos permitirá comprender más claramente la función de pérdida:

La pérdida *Soft Dice* es una función de pérdida popular para los modelos de segmentación. La ventaja es que funciona bien en presencia de datos desequilibrados.

Esto es especialmente importante en nuestra tarea de segmentación de tumores cerebrales, cuando una fracción muy pequeña del cerebro sean regiones tumorales.

Queremos que el numerador sea lo mayor posible y el denominador el menor posible para obtener un valor de pérdida pequeña. Veamos esto con el ejemplo:

i	p	g	$p_i g_i$	p_i^2	g_i^2
1	0.1	0	0	0.01	0
2	0.1	0	0	0.01	0
3	0.1	0	0	0.01	0
4	0.8	0	0	0.64	0
5	0.9	1	0.9	0.81	1
6	0.9	1	0.9	0.81	1
7	0.1	0	0	0.01	0
8	0.4	1	0.4	0.16	1
9	0.1	0	0	0.01	0
			2.2	**2.47**	**3**

Soft Dice Loss

$$L(P, G) = 1 - \frac{2\sum_i^n p_i g_i}{\sum_i^n p_i^2 + \sum_i^n g_i^2}$$

$$= 1 - \frac{2 \times 2.2}{2.47 + 3}$$

$$= 1 - \frac{4.4}{5.47}$$

$$\approx 0.2$$