­­­­Cuestionario 3

# Aprendizaje Automático – Julio A. Fresneda – 49215154F

## Ejercicio 1.

Tanto “bagging” como validación-cruzada cuando se aplican sobre una muestra de datos nos permiten dar una estimación del error de un modelo ajustado a partir de dicha muestra. Enuncie las diferencias y semejanzas entre ambas técnicas. Diga cuál de ellas considera que nos proporcionará una mejor estimación del error en cada caso concreto y por qué.

Bagging es una técnica que consiste en entrenar muchos modelos sobre diferentes puntos de una muestra, y a partir de estos modelos obtener un modelo promedio.

La validación cruzada es una técnica que consiste en dividir la muestra en K subconjuntos, obtener K modelos a partir de diferentes K-1 subconjuntos, y ponerlos a prueba con el subconjunto restante. Después se hace la media de los K valores obtenidos en el test.

Bagging y validación cruzada tienen algunas similitudes:

Ambas técnicas entrenan varios modelos con subconjuntos de las muestras, ambas usan los datos que no se han usado en el entrenamiento para test y ambas técnicas nos pueden servir para reducir el overfitting.

Sin embargo, tienen varias diferencias:

En bagging, de entre todos los modelos estimados, calculamos un modelo promedio, en cambio, en validación cruzada, de entre todos los modelos estimados, simplemente nos quedamos con el modelo que mejores resultados nos haya dado.

En bagging, los conjuntos de entrenamiento han sido obtenidos mediante bootstrapping (remuestreo de forma aleatoria y con reemplazamiento) mientras que los conjuntos de entrenamiento de validación cruzada se obtienen simplemente de dividir el conjunto inicial de entrenamiento en K subconjuntos.

La intención para la cual usamos cada técnica también es una gran diferencia. Bagging se usa como una forma de mejorar la generalización (y disminuir overfitting) al promediar muchos modelos entrenados con conjuntos extraídos con bootstrapping. La validación se usa típicamente como una forma de elegir empíricamente buenos hiperparámetros sin contaminar el conjunto de prueba, no necesariamente para corregir overfitting.

Bagging y validación cruzada no son excluyentes, si no complementarios. Se puede usar validación cruzada para varios modelos obtenidos con bagging, por ejemplo, para elegir el número de modelos usados para calcular el modelo promedio o la complejidad de estos modelos.

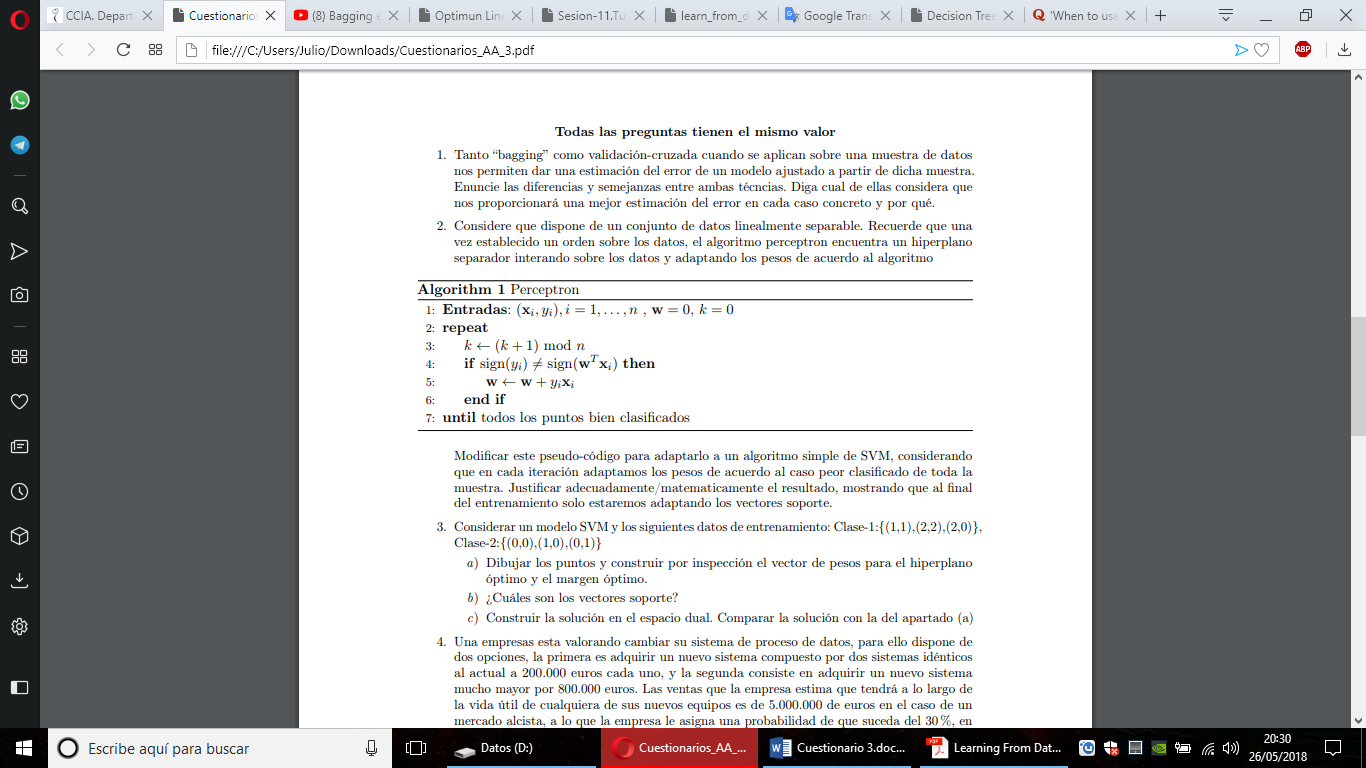
En cuanto en qué caso es mejor una técnica u otra, voy a exponer dos ejemplos.

Si, por ejemplo, tenemos una muestra de entrenamiento en la cual una parte tiene un ruido muy pronunciado (un input de entrenamiento que difiere mucho de la media del resto de inputs), con bagging vamos a tener un modelo promedio que, aunque ha disminuido la varianza, sigue teniendo en cuenta este ruido. En cambio con si usamos cross-validation nuestro modelo elegido no tiene por qué haber tenido en cuenta este ruido, de hecho con los hiperparámetros correctos nuestro modelo final no se verá afectado por el ruido.

En cambio, un ejemplo en el cual usamos bagging en vez de cross-validation, es en los árboles de decisión y random forest. Bagging tiene una ventaja respecto a cross-validation, y es que reduce la varianza, y cross-validation no. Esto nos es útil en los árboles de decisión, en los cuales buscamos baja varianza y bajo sesgo, cosa que conseguimos con bagging y no con cross-validation. Además podemos construir random forests sobre la idea de bagging, el cual es un predictor muy eficiente que construye árboles no correlacionados.

## Ejercicio 2.

Considere que dispone de un conjunto de datos linealmente separable. Recuerde que una vez establecido un orden sobre los datos, el algoritmo perceptron encuentra un hiperplano separador interando sobre los datos y adaptando los pesos de acuerdo al algoritmo:



Modificar este pseudo-código para adaptarlo a un algoritmo simple de SVM, considerando que en cada iteración adaptamos los pesos de acuerdo al caso peor clasificado de toda la muestra. Justificar adecuadamente/matematicamente el resultado, mostrando que al final del entrenamiento solo estaremos adaptando los vectores soporte.

## Ejercicio 3

Considerar un modelo SVM y los siguientes datos de entrenamiento: Clase-1:{(1,1),(2,2),(2,0)}, Clase-2:{(0,0),(1,0),(0,1)} a) Dibujar los puntos y construir por inspección el vector de pesos para el hiperplano óptimo y el margen óptimo. b) ¿Cuáles son los vectores soporte? c) Construir la solución en el espacio dual. Comparar la solución con la del apartado (a).

A)

Primero vamos a dibujar los puntos:

Por inspección, podemos ver claramente qué hiperplano separa de forma óptima:

De forma rápida podemos obtener los pesos. La función dibujada tiene la forma y= x + 1,5, ya que para x=0, y=1,5 y para y=0, x=1,5.

Puesto que todas las funciones lineales de este tipo son y=ax+b, y a = -w1/w2, tanto w1 como w2 posiblemente sean 1. Como b = -bias/w2, bias será igual a -1,5.

B)

Los vectores soporte son los vectores formados por los puntos que chocan con el margen del hiperplano, es decir, los que están exactamente en el margen mínimo, a una distancia de exactamente.

En este caso, hay dos vectores soporte: El vector soporte formado por los puntos (1,0) y (0,1) y el vector soporte formado por los puntos (1,1) y (2,0). Gráficamente el margen con los vectores de soporte se ve así:

C)

Vamos a buscar el vector de pesos w para el hiperplano óptimo.

El hiperplano que buscamos debe clasificar correctamente todos los puntos, por lo que se debe cumplir que para todo n. Por comodidad a la hora de hacer cálculos, vamos a considerar respecto a los pesos y la b (bias) de forma que la magnitud mínima de sea mayor o igual a 1. Esto se puede hacer porque siempre podemos reescalar los pesos y la bias sin obtener diferencia en el resultado.

Teniendo en cuenta lo anterior, podemos decir que nuestro hiperplano separa los datos si y solo si sus pesos y bias cumplen que:

Lo que buscamos para nuestro hiperplano es que la distancia con el punto más cercano sea máxima. La distancia la podemos calcular de la siguiente manera:

En el punto más cercano, . Por lo tanto, la distancia con el punto más cercano es

Como buscamos que la distancia del hiperplano con el punto más cercano sea máxima, buscamos maximizar , o lo que es lo mismo, minimizar

Debemos minimizar , teniendo en cuenta que se cumple la anterior condición, que se puede reescribir de la siguiente manera:

Por lo tanto, nuestro espacio dual es el siguiente:

Vamos a ver cómo afecta esta última condición a cada punto individualmente.

Simplificando:

Gracias a que antes reescalamos las magnitudes de forma que , hemos podido obtener el sistema de inecuaciones anterior.

Operando, podemos llegar a las siguientes inecuaciones:

Para , tenemos que , lo que nos lleva a que .

Entonces, , por lo que tenemos que .

Para simplificar vamos a dividir estos pesos y bias entre 2:

Si probamos a clasificar con estos pesos, vemos que clasifica correctamente para todos los puntos.

Por lo tanto, nuestro clasificador es .

Para dibujar el hiperplano, vamos a pasar estos pesos a dos dimensiones para obtener una función de la forma y=ax+b.

. Por lo tanto, y=x+1,5.

El hiperplano dibujado en el gráfico sería el siguiente.

La distancia óptima del hiperplano con el punto más cercano como hemos dicho es

## Ejercicio 4

Una empresa está valorando cambiar su sistema de proceso de datos, para ello dispone de dos opciones, la primera es adquirir un nuevo sistema compuesto por dos sistemas idénticos al actual a 200.000 euros cada uno, y la segunda consiste en adquirir un nuevo sistema mucho mayor por 800.000 euros. Las ventas que la empresa estima que tendrá a lo largo de la vida útil de cualquiera de sus nuevos equipos es de 5.000.000 de euros en el caso de un mercado alcista, a lo que la empresa le asigna una probabilidad de que suceda del 30 %, en caso contrario, las ventas esperadas son de 3.500.000 euros. Construir el árbol de decisiones y decir que opción es la más ventajosa para la empresa.

Qué equipo comprar

1x800.000€ 2x200.000€

Tendencia del mercado

Tendencia del mercado

Alcista (p 0.3): Ventas 5.000.000€

Alcista (p 0.3): Ventas 5.000.000€

No alcista (p 0.7): Ventas 3.500.000€

No alcista (p 0.7): Ventas 3.500.000€

Ganancia 4.600.000€

Ganancia 3.100.000€

Ganancia 4.200.000€

Ganancia 2.700.000€

Vamos a decidir qué opción es más ventajosa. Como las opciones usan probabilidad, vamos a usar la esperanza matemática.

Usando la esperanza matemática, el beneficio esperado de la opción de 1 equipo de 800.000€ cada uno es de:

2.700.000 x 0,7 + 4.200.000 x 0,3 = 3.150.000

En cambio, el beneficio esperado de la opción de 2 equipos de 200.000 euros es de:

4.600.000 x 0,3 + 3.100.000 x 0,7 = 3.550.000

Por lo tanto, nos conviene comprar 2 equipos de 200.000 euros cada uno, ya que obtendremos más beneficio.

## Ejercicio 5

¿Que algoritmos de aprendizaje no se afectan por la dimensionalidad del vector de características? Diga cuáles y por qué.

## Ejercicio 6

Considere la siguiente aproximación al aprendizaje. Mirando los datos, parece que los datos son linealmente separables, por tanto decidimos usar un simple perceptron y obtenemos un error de entrenamiento cero con los pesos óptimos encontrados. Ahora deseamos obtener algunas conclusiones sobre generalización, por tanto miramos el valor dvc de nuestro modelo y vemos que es d + 1. Usamos dicho valor de dvc para obtener una cota del error de test.

Argumente a favor o en contra de esta forma de proceder identificando los posible fallos si los hubiera y en su caso cual hubiera sido la forma correcta de actuación.

El primer error es mirar los datos. Nunca hay que mirar los datos. No podemos mirar los datos con los que vamos a trabajar porque podemos hacer suposiciones que aunque correspondan con la muestra, no tenemos la certeza de que correspondan con la realidad.

La forma correcta de actuar hubiera sido intentar separar los datos con varios modelos (perceptrón, regresión lineal para clasificación, etc) y quedarnos con el que mejor resultados nos haya dado, pero sin mirar los datos.

## Ejercicio 7

Discuta pros y contras de los clasificadores SVM y Random Forest (RF). Considera que SVM por su construcción a través de un problema de optimización debería ser un mejor clasificador que RF. Justificar las respuestas.

## Ejercicio 8

¿Cuál es a su criterio lo que permite a clasificadores como Random Forest basados en un conjunto de clasificadores simples aprender de forma más eficiente? ¿Cuales son las mejoras que introduce frente a los clasificadores simples? ¿Es Random Forest óptimo en algún sentido? Justifique con precisión las contestaciones.

## Ejercicio 9

En un experimento para determinar la distribución del tamaño de los peces en un lago, se decide echar una red para capturar una muestra representativa. Así se hace y se obtiene una muestra suficientemente grande de la que se pueden obtener conclusiones estadísticas sobre los peces del lago. Se obtiene la distribución de peces por tamaño y se entregan las conclusiones. Discuta si las conclusiones obtenidas servirán para el objetivo que se persigue e identifique si hay algo que lo impida.

Las conclusiones obtenidas pueden no ser válidas por que posiblemente los peces capturados no están independiente e idénticamente distribuidos. Cuando se echa la red para capturar la muestra representativa, la red captura la muestra de una pequeña parte del lago. No sabemos, si, por ejemplo, en el fondo del lago, donde la red no ha llegado, hay otro tipo de peces mucho más grandes que no se han capturado, por lo que no tenemos la certeza de que la muestra sea representativa del lago.

## Ejercicio 10

Identifique dos razones de peso por las que el ajuste de un modelo de red neuronal a un conjunto de datos puede fallar o equivalentemente obtener resultados muy pobres. Justifique la importancia de las razones expuestas.

Incorrecta inicialización de pesos:

Cuando usábamos la función de signo , la clasificación no era suave, era +1 o -1. En redes neuronales, en vez de usar la función de signo, usamos una aproximación suave a la función de signo que nos permita usar métodos analíticos en vez de puramente combinacionales para encontrar los pesos óptimos. Esta función, que suaviza la clasificación en +1 o -1, es la función tanh(·), la cual devuelve valores entre -1 y +1. La función tahn(·) es una función cuya representación gráfica es parecida a la sigmoide.

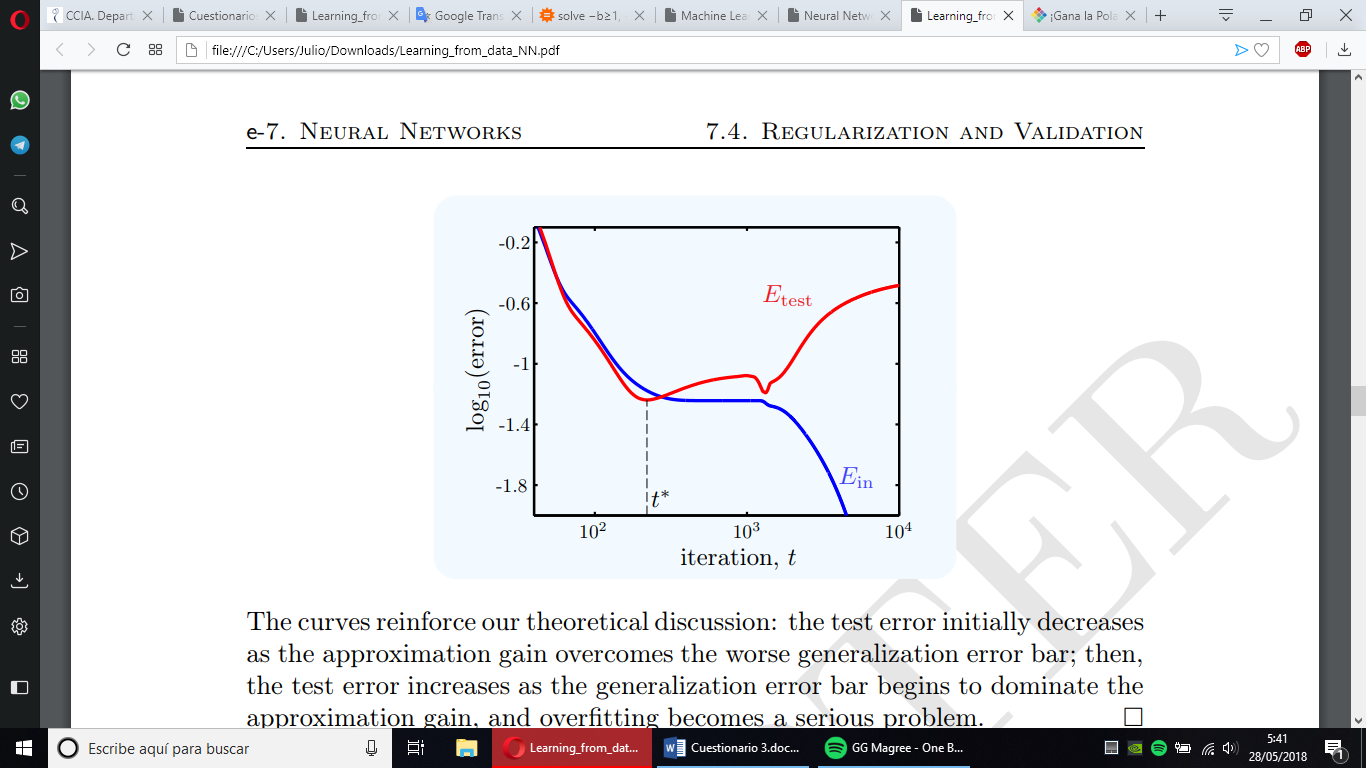
Si los pesos iniciales que elegimos son demasiado grandes, saturaremos la función que usamos en el algoritmo de aprendiaje, , la cual devolverá valores muy cercanos a +1 y -1, por lo tanto el gradiente que usamos en el algoritmo para obtener los pesos será cercano a 0 y el algoritmo no avanzará a ninguna parte. Esto es porque la función tanh(·) es extremadamente plana cuando su argumento es muy grande (en valor absoluto), por lo que de una iteración a otra apenas avanzaremos. De hecho, al ver que no mejoramos casi nada el error al pasar las iteraciones, podríamos pensar que hemos llegado a un mínimo, cuando no es así.

Es mejor inicializar los pesos con valores aleatorios muy pequeños de forma que . Sin embargo, también es un problema inicializar los pesos iniciales exactamente a 0, ya que el gradiente no realizará ningún movimiento.

El algoritmo realiza demasiadas iteraciones y no está regularizado:

Si realizamos demasiadas iteraciones y no hacemos “Early stopping”, ni usamos ningún otro sistema de regularización, el algoritmo ajustará demasiado bien a nuestros datos de entrenamiento y tendremos overfitting. Para una mejor generalización, es bueno realizar una parada prematura, “Early stopping”.

Aquí muestro un ejemplo de entrenamiento sin ninguna regularización, concretamente un entrenamiento de un modelo que distinga el número 1 del resto de números.



Como vemos, a partir de un cierto umbral de iteraciones, el Ein cae en picado (overfitting) y el Eout se dispara.

Es por esto que el entrenamiento de redes neuronales funciona mejor con Stochastic Gradient Descent que con Batch Gradient Descent, ya que el SGD no entrena con todos los puntos y obtiene un modelo más generalizado, con menos overfitting.

También hay que elegir bien el tamaño del set de train y test: Con un test demasiado grande habrá pocos datos en train para entrenar, y con un test demasiado pequeño el Eout no será representativo. Sobre saber cuándo parar, y sobre elegir el tamaño de los sets de train y test, se pueden averiguar usando Cross-Validation.