Aprendizaje Automático – Guía de Ejercicios [*]

Departamento de Computación – FCEyN
Universidad de Buenos Aires

Primer cuatrimestre 2024
Versión: 5 de abril de 2024

Cambios respecto a la versión anterior:

- Agregados dos items en el Ejercicio 3.7
- Agregada las secciones 4 (selección, validación, evaluación) y 5 (métricas)

1. Repaso probabilidad y estadística

Para resolver los siguientes ejercicios recomendamos leer el **Capítulo 6** del libro Mathematics for Machine Learning de Deisenroth, Faisal y Soon Ong. `https://mml-book.github.io/book/mml-book.pdf`

Ejercicio 1.1. Explique por qué los siguientes eventos son independientes de a pares pero no independientes entre todos. Dadas 2 monedas,

(a) la primera moneda es cara;

(b) la segunda moneda cara;

(c) las dos monedas son iguales.

Ejercicio 1.2. Demostrar el teorema de Probabilidad Total: dados una partición $\{A_i\}_{i=1}^n$ del espacio muestral tal que $P(A_i) > 0$ para todo i, y un evento B:

$$P(B) = \sum_{i=1}^{n} P(B \mid A_i) \cdot P(A_i)$$

Ejercicio 1.3.

(a) Sugerencia, mirar este video: `https://www.youtube.com/watch?v=HZGCoVF3YvM&t=57s`(3blue1brown Bayes)

(b) Demostrar el teorema de Bayes: dados dos eventos A y B tal que $P(B) > 0$,

$$P(A \mid B) = \frac{P(B \mid A) \cdot P(A)}{P(B)}$$

(c) Un local vende dos marcas de televisores, A y B. El 40 % de los televisores vendidos son de la marca A y un 30 % de ellos tienen un defecto. Por otro lado, el 20 % de los televisores vendidos son de la marca B y el 10 % tienen un defecto. Si un televisor tiene un defecto, ¿cuál es la probabilidad de que sea de la marca A?

(d) Supongamos que tienes dos máquinas, A y B, que producen tornillos. La longitud de los tornillos producidos por cada máquina sigue una distribución normal. Se sabe que la máquina A produce tornillos con una longitud media de 10 cm y una desviación estándar de 0.5 cm, mientras que la máquina B produce tornillos con una longitud media de 10.5 cm y una desviación estándar de 0.7 cm.

Ahora, supongamos que se selecciona un tornillo al azar de la producción combinada de ambas máquinas y se encuentra que tiene una longitud de 10.2 cm. ¿Cuál es la probabilidad de que este tornillo provenga de la máquina B?

Ejercicio 1.4.

[*] Algunos ejercicios fueron adaptados de los libros "Machine Learning", de Tom Mitchell (McGraw-Hill, 1997); "Pattern Recognition and Machine Learning", de Christopher Bishop (Springer, 2006); y "An Introduction to Statistical Learning", de James, Witten, Hastie & Tibshirani (Springer, 2015).

(a) Explicar con tus palabras qué es la media y qué es el desvío estándar.

(b) En una fábrica de producción de caramelos, se mide la longitud de los caramelos producidos. Si la longitud media es de $5cm$ y un desvío de $0{,}05cm^2$. Siendo que tener un caramelo de más de 5.05 cm o menos de 4.95 cm se considera defectuoso, ¿qué significa esto en términos de la calidad de los caramelos producidos por esa fabrica?. ¿Hicieron alguna suposición sobre la distribución?

Ejercicio 1.5.
Has realizado un experimento en el que lanzaste una moneda 10 veces y la secuencia observada de resultados fue HHHTHTTHHT.

1. Suponiendo que la moneda es justa (es decir, la probabilidad de que salga cara (P(H)) = 0,5 y la probabilidad de que salga cruz (P(T)) = 0,5), calcule la probabilidad de observar la secuencia dada.
2. Es más probable que la moneda esté sesgada hacia la cara en 70 % ó que la moneda este balanceada dados los datos observados. (HHHTHTTHHT).
3. Imaginen ahora que queremos estimar la carga de la moneda lo mejor posible dados los datos de la tirada. Plantear los pasos a seguir para encontrar este valor. Tip: están calculando máxima verosimilitud.
4. Calcularla :)
5. Sugerencia, ver: `https://www.youtube.com/watch?v=Dn6b9fCIUpM&t=195s`) (statquest)

2. Introducción

Ejercicio 2.1. Revisar y completar el notebook `notebook_1_herramientas.ipynb`.

Ejercicio 2.2. Describir para los siguientes problemas si se trata de aprendizaje supervisado o aprendizaje no supervisado. Especificar qué medida de performance y de un ejemplo de una base de datos que permita encarar el problema.

(a) detección de discurso de odio en tweets;

(b) recomendación de películas;

(c) diagnóstico de tumores por imágenes;

(d) autocompletar textos;

(e) segmentación comercial de clientes;

(f) autenticación biométrica (ej: huellas dactilares);

(g) detección de fraude en tarjetas de crédito.

Ejercicio 2.3. Determinar para los siguientes problemas de aprendizaje supervisado si se trata de problemas de clasificación o de regresión. Para cada caso, indique un ejemplo de instancia (el valor de sus atributos) junto a una etiqueta posible especificando el tipo de cada valor.

(a) Dado un tweet, determinar si habla en contra o a favor de un candidato presidencial.

(b) Predecir cuánto gastará una empresa en luz el próximo semestre.

(c) Dado un tweet, predecir la probabilidad de que hable en contra o a favor de un candidato.

(d) Predecir a qué distancia de la facultad vive una persona.

(e) Predecir si se gastará más o menos que $50.000 por mes de luz el próximo semestre. ¿Qué responderían si en la base de datos tenemos etiquetas reales? ¿Y qué si tuvieramos etiquetas categoricas (sí, no)?

(f) Predecir la probabilidad de que se gaste más o menos que $50.000 por mes de luz el próximo semestre.

(g) Predecir la nota que tendrá un alumno en un examen cuya nota puede ser $0, 1, 2, \ldots, 10$

(h) Predecir la nota que tendrá un alumno en un examen cuya nota puede ser “A”, “R” o “I”.

(i) Predecir dónde vive una persona.

(j) Predecir la próxima palabra a autocompletar dadas las oraciones anteriores.

(k) Predecir el valor que tomará el dolar en los próximos diez días.

Ejercicio 2.4. Sea un problema de clasificación en el cual cada instancia tiene 2 atributos numéricos (coordenadas x e y) y pertenece a una de dos clases posibles (blanco o negro).
Se tienen tres tipos de hipótesis ilustrados en la Figura que representan (a) rectas, (b) líneas verticales (hasta 30 líneas), (c) elipses (tantas como se quiera).
Para cada uno de ellos, se pide:

- Describir el espacio de hipótesis H;
- Identificar los parámetros de la hipótesis (el conjunto de valores que permiten describir una hipótesis en concreto, θ).[1]

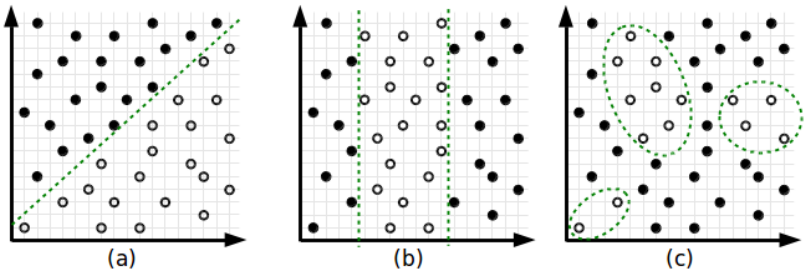


Figura 1: Tipos de Hipótesis

Ejercicio 2.5. Completar el notebook `notebook_2_titanic.ipynb` .

[1] "Machine Learning", de Tom Mitchell (McGraw-Hill, 1997);

3. Árboles de decisión

Cielo	Temperatura	Humedad	Viento	¿Salgo? (clase a predecir)
Sol	Calor	Alta	Débil	No
Sol	Calor	Alta	Fuerte	No
Nublado	Calor	Alta	Débil	Sí
Lluvia	Templado	Alta	Débil	Sí
Lluvia	Frío	Normal	Débil	Sí
Lluvia	Frío	Normal	Fuerte	No
Nublado	Frío	Normal	Fuerte	Sí
Sol	Templado	Alta	Débil	No
Sol	Frío	Normal	Débil	Sí
Lluvia	Templado	Normal	Débil	Sí
Sol	Templado	Normal	Fuerte	Sí
Nublado	Templado	Alta	Fuerte	Sí
Nublado	Calor	Normal	Débil	Sí
Lluvia	Templado	Alta	Fuerte	No

Tabla 1: Salgo a caminar

Ejercicio 3.1. Hacer en papel y lápiz un árbol de decisión correspondiente a entrenar con los datos de la Tabla Salgo a caminar. Utilizar el criterio "Gini Gain" para calcular el feature que mejor separa cada decisión. Armar luego tres ejemplos posibles de instancias nuevas (no existentes en la tabla) y usar el árbol para predecir la clase de salida. Calcular además la importancia de cada atributo (decrecimiento total en la impureza Gini).

Ejercicio 3.2. ¿Cómo cambiaría el árbol si restringiéramos su altura a 2 niveles?

(a) ¿Cuál sería el resultado de las predicciones del ejercicio anterior?

(b) ¿Algunas de las instancias existentes en la tabla, serían clasificadas incorrectamente?

Ejercicio 3.3. ¿Quién es quién?
El quién es quién es un juego (buscar en Google) en el que hay que adivinar el nombre del personaje del rival. Para ello se hacen preguntas que son respondidas por sí o por no.

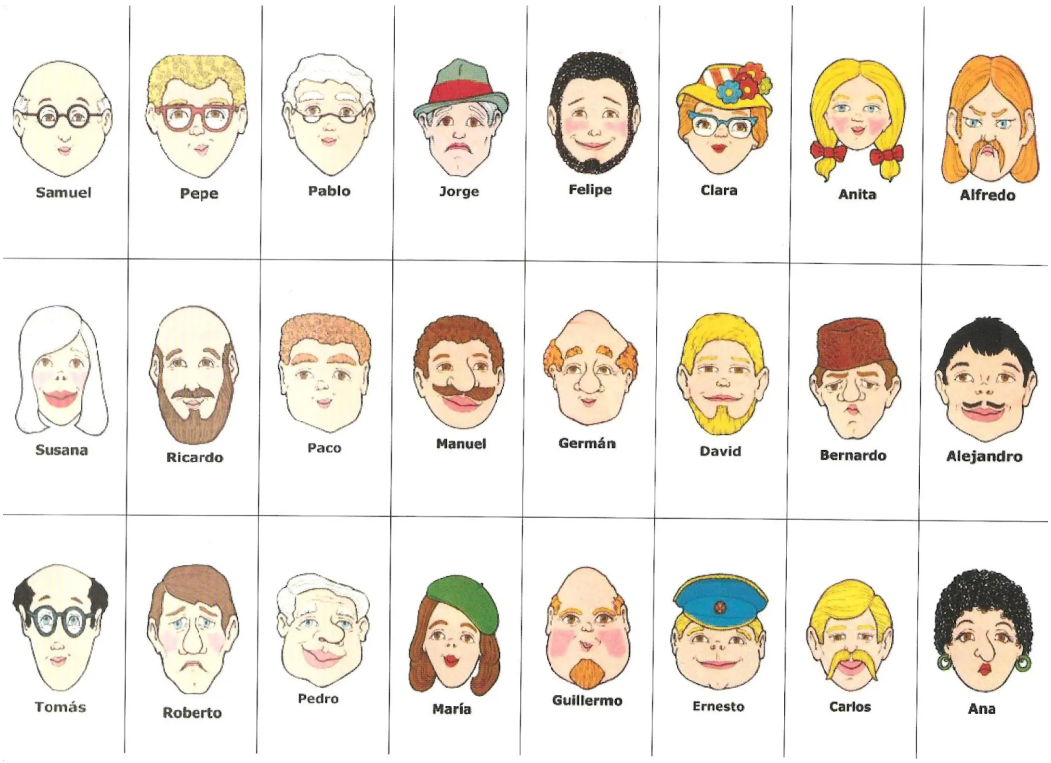


Figura 2: Juego de ¿quién es quién?

Tomemos la libertad de jugar únicamente a adivinar el sexo biológico (masculino/femenino).

(a) Si preguntamos si el personaje es calvo... ¿Cuál es la ganancia gini al separar por este atributo?

(b) Ordenar por valor de ganancia gini qué conviene como primera pregunta: i) ¿Tiene calvicie? ii) ¿Tiene cabello rubio? iii) ¿Tiene vello facial? (bigote y/o barba) iv) ¿Tiene sombrero?

Ejercicio 3.4. En la Figura Cortes en el espacio de atributos puede verse diversas regiones en el espacio de atributos.

- Determinar cuáles de ellas pueden haber sido generadas por árboles de decisión.
- Para las que lo sean, mostrar un árbol que hubiese generado estas regiones (suponer ejes x_1 y x_2)

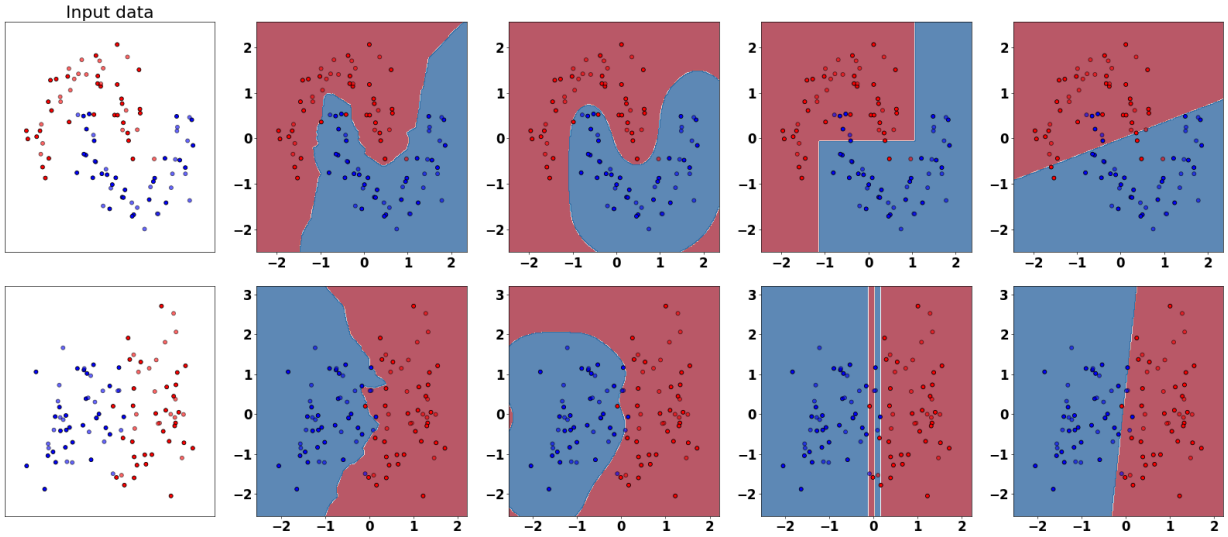
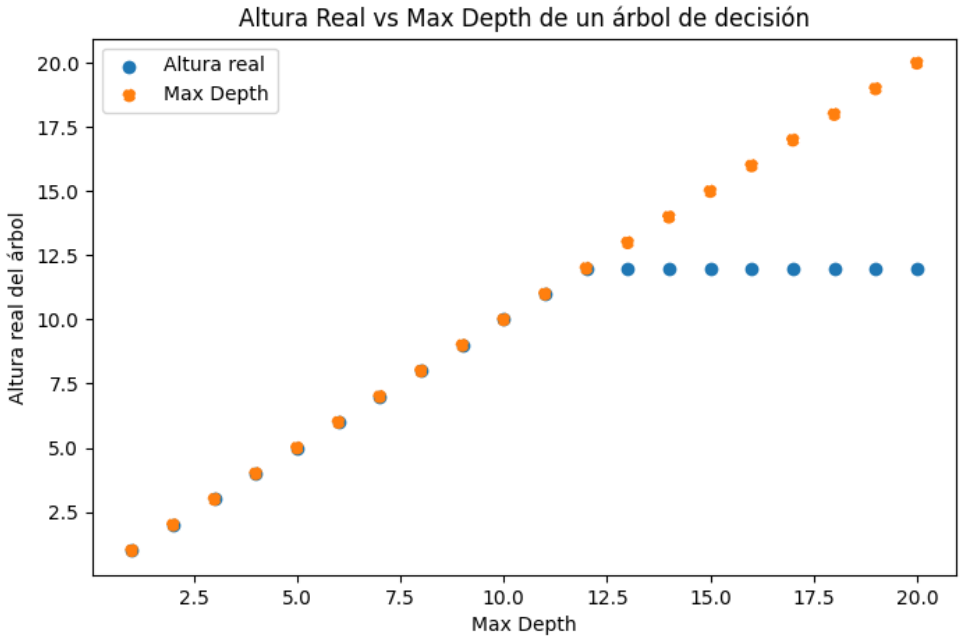


Figura 3: Cortes en el espacio de atributos

Ejercicio 3.5. Se entrenan 20 árboles distintos variando el parámetro max_{depth} desde 1 hasta 20 inclusive. Graficamos, para cada uno de estos árboles, el valor de max_{depth} y el valor de la altura real de estos. Responder:



- ¿Por qué sucede que, a partir de max_depth igual a 12, la altura real del árbol es constante?
- Dado n instancias, ¿cuál seria el valor máximo de max_depth que tiene sentido utilizar?

Ejercicio 3.6. Dado el algoritmo de construcción de árboles visto en clase para atributos continuos:
Sea S una muestra de instancias con atributos A. Para construir un árbol de decisión ejecutamos:

(1) Mientras no se cumpla un criterio de detención:

   (I) Creamos `nodo_actual`, un nodo que representa una decisión de corte.

   (II) Elegimos el par $a \in A$, $c \in R$ entre los posibles pares $\langle$ atributo, corte$\rangle$, que mejor divida a S para `nodo_actual` según $\Delta M(S, \langle a, c \rangle)$
   (ΔM es una función que devuelve cuánto gano si divido a S en dos utilizando $\langle a, c \rangle$ y según la medida M. Por ejemplo ΔM podría ser `GananciaDeInformación`).

   (III) Crear dos hijos de `nodo_actual`.

   (IV) Dividir las instancias de S en los nuevos nodos, según $\langle a, c \rangle$:

$$S_{\leq} \leftarrow \{x | x \in S \wedge x[a] \leq c\}$$
$$S_{>} \leftarrow \{x | x \in S \wedge x[a] > c\}$$

   (V) Repetir para cada hijo.

(2) El valor asignado a cada región resultante (a cada hoja) será el de la clase mayoritaria de las instancias que pertenezcan a esta región.

Se pide:

(a) Escribir el pseudocódigo (puede ser similar a python) para el paso (b) (elegir el mejor par a, c entre los posibles pares). Es decir, definir `mejor_corte`$(S, A, \Delta M)$, en donde `S` representa la muestra, `A` un conjunto de atributos y ΔM la función que computa la ganancia de una división. Puede suponer dadas funciones tales como `S[a]` que devuelve la columna de valores para el atributo a. Es importante que esta función no evalúe dos veces cortes que devuelven exactamente las mismas regiones para un mismo atributo.

(b) Introducir los cambios necesarios en el algoritmo general (y si fuera necesario, en la función que definieron para `mejor_corte`) que permita medir la importancia de cada atributo. La importancia deberá ser un valor numérico entre 0 y 1.

Ejercicio 3.7. Determinar cuáles de las siguientes son afirmaciones verdaderas:

(a) El objetivo de construir un árbol de decisión es crear el árbol de menor tamaño posible en el cual las hojas contengan valores de una sóla clase.

(b) Los algoritmos de construcción vistos (CART, ID3, etc) exploran todos los posibles árboles y se quedan con el que mejor separa a las instancias.

(c) La pureza describe qué tan cerca está un nodo de contener instancias de una sola clase.

(d) Un atributo puede aparecer sólo una vez en cada rama del árbol (llamamos rama a un camino directo desde una hoja hasta la raíz).

(e) Un par (atributo, corte) puede aparecer sólo una vez en cada rama del árbol (llamamos rama a un camino directo desde una hoja hasta la raíz).

(f) Para cada nueva instancia, un árbol permite predecir la clase a la que pertence. Por otra parte, para predecir **la probabilidad** de pertencer a una clase u otra, es necesario modificar el algoritmo de creación de árboles.

(g) Un árbol de decisión, con criterios de corte suficientemente laxos, puede siempre conseguir 100 % de aciertos en los datos de entrenamiento.

(h) Un árbol de decisión, con criterios de corte suficientemente laxos, puede siempre conseguir 100 % de aciertos en los datos de entrenamiento siempre y cuando no haya contradicciones entre las etiquetas de instancias iguales.

Ejercicio 3.8. Resolver el notebook `notebook_3_arboles_de_decision_sklearn.ipynb`.

Ejercicio 3.9. Preguntas conceptuales para discutir:

(a) ¿Cuál el sesgo inductivo del algoritmo que construye el árbol de decisión?

(b) ¿Qué sucede cuando dos atributos empatan en ganancia de información? ¿Esta decisión es parte del sesgo inductivo?

(c) ¿Cómo se comporta la ganancia de información en comparación a impureza Gini cuando se comparan atributos con gran cantidad de valores distintos? Por ejemplo, si el atributo x_1 tiene dos valores posibles (true y false) y el atributo x_2 tiene 40 valores distintos, ¿es justo usar ganancia de información para elegir entre ellos? ¿Qué desventajas tiene? ¿Cómo se podría mitigar?

(d) Dado un atributo a continuo en la cual queremos encontrar el mejor corte c según alguno de los criterios considerados (Gini o Entropy), ¿cuál es la complejidad de peor caso si se asume que tenemos n instancias en nuestro conjunto de datos?

Ejercicio 3.10. Completar el notebook `notebook_4_implementacion_arbol.ipynb`. Este notebook contiene una implementación parcial de un algoritmo de creación de árboles de decisión.

Ejercicio 3.11. Dado el algoritmo de construcción de árboles para atributos continuos (ver algoritmo en el ejercicio 3.6) ¿Qué cambios son necesarios para que funcione para problemas de regresión? Para pensar:

- ¿Pueden las medidas vistas en clase (CER, Gini, Entropía) ser utilizadas en este caso?
- ¿Qué cambiarían en el criterio de detención?
- ¿Cómo cambiarían el valor asignado a los nodos hojas?

4. Búsqueda de hiperparámetros, validación cruzada y Evaluación

Ejercicio 4.1.
Sea GRID_SEARCH(X : LIST⟨TUPLE⟨ALGORITMO, HYPERS⟩⟩, D : DATA, M : METRICA) : TUPLE⟨ALGORITMO, HYPERS, FLOAT⟩ la función que ejecuta una búsqueda exhaustiva para encontrar la mejor configuración entre una lista de posibilidades y retorna el valor esperado para dicha combinación.
Detalles de los parámetros:

- X es una lista de TUPLE⟨A, HS⟩ : TUPLE⟨ALGORITMO, HYPERS⟩, en donde A es el nombre del algoritmo y HS representa un diccionario de hiperparámetros a valores.
  Por ej, un elemento de la lista puede ser $\langle arbol\_de\_decision, \{altura\_max : 10, medida : gini, atributos : todos\}\rangle$
  otro puede ser $\langle arbol\_de\_decision, \{altura\_max : 5, medida : entropy, atributos : todos\}\rangle$
- D, un dataset de $n \times m$ (instancias $\times$ atributos)
- M, una métrica (por ejemplo, ACCURACY)

(a) Escribir el pseudocódigo de la función. Para ello, suponga ya implementada la función CROSS_VAL(A : ALGORITMO, HS : HYPERS, D : DATA, M : METRICA) : FLOAT que devuelve el resultado de ejecutar validación cruzada para un algoritmo A e hiper-parámetros HS sobre la base de datos D y usando la métrica M. Retorna el valor obtenido.

(b) Escribir el pseudocódigo de la función CROSS_VAL(A : ALGORITMO, HS : HYPERS, D : DATA, M : METRICA) : FLOAT. ¿Qué tipo de validación cruzada elegiste para tu pseudocódigo?

(c) Definir ahora la función RANDOMIZE_SEARCH(..., N : INT) : TUPLE⟨ALGORITMO, HYPERS, FLOAT⟩ con los mismos parámetros que GRID_SEARCH más N, un parámetro que indica la cantidad de intentos a probar. Suponer para esta función que H : HYPERS es un diccionario en donde los valores también pueden ser distribuciones probabilísticas a las que se las puede muestrear con la función SAMPLE(D: DISTRIBUCIÓN) : FLOAT (suponer que SAMPLE(CONSTANTE) = CONSTANTE. Ej, $\langle arbol\_de\_decision, \{altura\_max : Binomial(n = 20, p = 0{,}5), medida : entropy, atributos : todos\}\rangle$

Ejercicio 4.2. Validación cruzada. Verdadero o Falso. Justificar

(a) Hacer validación cruzada evita el sobreajuste (overfitting) de los modelos sobre los datos.

(b) Hacer validación cruzada ayuda a obtener estimaciones más realistas de la performance de un modelo sobre nuevos datos que al hacerlo sobre los mismos datos de entrenamiento.

(c) En K-fold cross validation, conviene que K se acerque a N. De esta manera el resultado será lo más realista posible ya que se tiende a generar N modelos independientes. El problema es que hay que entrenar demasiados modelos.

(d) Evaluar un modelo sobre el conjunto de evaluación (control) resultará en un valor siempre peor o igual al conseguido en desarrollo.

(e) Una vez elegido el mejor modelo durante desarrollo, es conveniente hacer k-fold cross validation nuevamente, pero ahora incluyendo también el conjunto de evaluación.

(f) Luego de seleccionar la mejor configuración, es ideal volver a correr el algoritmo pero esta vez utilizando todos los datos de desarrollo para luego evaluarlo en los datos de evaluación.

Ejercicio 4.3. Consideremos K-fold cross validation con las siguientes modificaciones:

- Para cada instancia x_i dentro de un conjunto de datos $\mathcal{D}$ se tiene un mapeo G que le asigna un único grupo $g \in \mathcal{G}$ tal que $G(x_i) = g$ (pueden pensarlo como un DICC(INSTANCIA, GRUPO)).
- Al dividir $\mathcal{D}$ en los k folds, se asegura que cada grupo g este contenido únicamente en un fold.

(a) Construir el algoritmo GROUP_K_CROSS_VAL(A : ALGORITMO, HS : HYPERS, D : DATA, M : MÉTRICA, G: GRUPOS, K: INT) : FLOAT que devuelve el resultado de evaluar el algoritmo dado con los hiperparámetros dados y algún valor de k respetando los grupos dados.

(b) ¿En que escenarios podría tener sentido utilizar este procedimiento? Desarrolle.

(c) ¿Qué cambiaría en su implementación si hubiese más folds que grupos?

Ejercicio 4.4. Explicar las posibles causas por las cuales un modelo entrenado y evaluado de la siguientes maneras puede funcionar peor que lo esperado al ser llevado a producción.

(a) Entrenado y evaluado sobre datos de entrenamiento.

(b) Seleccionado entre muchas posibilidades sobre datos de desarrollo (utilizando grid o random search junto a cross-validation).

(c) Seleccionado entre muchas posibilidades sobre datos de desarrollo (utilizando grid o random search junto a cross-validation) y evaluado en datos held-out.

Ejercicio 4.5. Preguntas para desarrollar.

(a) En la clase teórica vimos que al hacer cross validation los datos no siempre deben separarse al azar, ¿por qué?. Pensar ejemplos de al menos dos situaciones en las cuales no sea conveniente.

(b) ¿Por qué se deberían usar una sola vez los datos held-out?

(c) ¿Qué sería conveniente hacer si luego de un muy buen resultado en desarrollo, encuentro un pésimo resultado en evaluación?

Ejercicio 4.6. La técnica de SOBREMUESTREO (oversampling) consiste en crear copias de instancias al azar (a veces agregando mínimas modificaciones en alguno de sus atributos) y asignarle la etiqueta original.

Este proceso se utiliza mucho cuando las clases están muy desbalanceadas y no es suficiente la cantidad de instancias de alguna clase para entrenar un clasificador. En estos casos es normal, por ejemplo, sobremuestrear hasta lograr la misma cantidad de instancias para cada clase.

Por ejemplo, dado el problema de clasificar entre GATOS, PERROS y CONEJOS, y dado que la cantidad de instancias es 100, 2000 y 20 respectivamente, generamos un nuevo dataset de 2000, 2000, y 2000 instancias, en donde las instancias agregadas son copias de instancias de la clase correspondiente elegidas al azar y a las que se le agrega un poco de ruido a sus columnas numéricas.

Describir los pros y contras de aplicar la técnica en los siguientes pasos del proceso. Justificar en términos de posibilidades de sub o sobreestimar la performance de nuestros modelos y concluir cuál es la opción más segura:

(I) Antes de partir en desarrollo - evaluación

(II) Antes de partir en Folds en desarrollo.

(III) Luego de seleccionar los folds que se utilizarán para entrenar el modelo dentro de las iteraciones de K-fold cross val. Aplicarlo sólo a los datos de entrenamiento.

(IV) Luego de seleccionar los folds que se utilizarán para entrenar el modelo dentro de las iteraciones de K-fold cross val. Aplicarlo tanto a los datos de entrenamiento como a los de validación.

Ejercicio 4.7. Ordenar las siguientes acciones para evitar problemas (indentar si hay loops y especificar sobre qué datos se haría)

(a) Mandar reporte final a toda la empresa y submitear paper.

(b) Seleccionar la mejor configuración, entrenar con los datos completos.

(c) Partir los datos en desarrollo - evaluación

(d) Correr un clasificador super simple (lo más sencillo que les suela funcionar) y ver si tiene confianza alta para etiquetas incorrectas.

(e) Entrenar un modelo usando una configuración, testear su performance en datos que no fueron utilizados para entrenar. Guardar el resultado.

(f) Partir en 10 folds.

(g) Correr un proceso que lista los atributos según qué tanto se correlacionan con las etiquetas y conservar sólo el top10 para entrenar el modelo.

(h) Definir la métrica a utilizar (ej: accuracy pesada por clase)

(i) Mirar / escuchar / leer instancias del dataset para pensar buenos atributos.

(j) No me dio tan bien como esperaba. Empiezo de nuevo repensando algunas cosas.

(k) Armar una grilla de configuraciones a probar

(l) Después de correr lo anterior me di cuenta que podría haber probado tal modelo / tal hiperparámetro.

(m) Agregar a las configuraciones y repetir.

(n) Evaluar el modelo entrenado con los datos completos.

(ñ) Plotear la distribución de las etiquetas.

Ejercicio 4.8. Completar el notebook `notebook_seleccion_modelos.ipynb`

5. Métricas

Ejercicio 5.1. Matriz de Confusión

(a) En un problema de clasificación binaria, ¿a qué se denomina clase positiva y a qué clase negativa? Si nuestro problema consiste en clasificar spam vs. no-spam, ¿cuál es la clase positiva? Si nuestro problema es clasificar imágenes de perros vs. gatos, ¿cuál es la clase positiva?

(b) Explicar con tus palabras la definición de *verdadero positivo*, *verdadero negativo*, *falso positivo* y *falso negativo*.

(c) Completar la Primera Parte del notebook `notebook_metricas.ipynb`. El Test 1 debería pasar.

(d) ¿Por qué podría un falso positivo ser considerado más (o menos) importante que un falso negativo? Dar un ejemplo en donde es más grave tener falsos negativos que falsos positivos.

Ejercicio 5.2. Métricas de umbral fijo

(a) Explicar con tus palabras la definición de *accuracy*, *precision* y *recall*.

(b) Completar la Segunda Parte del notebook `notebook_metricas.ipynb`. El Test 2 debería pasar.

(c) ¿Por qué es un problema medir *accuracy* de un clasificador para compararlo con otro? Dar un ejemplo en donde sería engañoso utilizar esta comparación.

(d) Demuestre que F_β puede ser reescrito como $F_\beta = \dfrac{(1+\beta^2)\cdot \mathrm{TP}}{(1+\beta^2)\cdot \mathrm{TP} + \beta^2 \cdot \mathrm{FN} + \mathrm{FP}}$

(e) Demuestre que la métrica recall vista como función del umbral de decisión, es una función monótona decreciente ($recall_{M,D}(\mu_1) \leq recall_{M,D}(\mu_2)$ si $\mu_1 > \mu_2$)

(f) Muestre cómo la métrica precision vista como función del umbral de decisión, **no necesariamente** es una función monótona creciente ($precision_{M,D}(\mu_1) \not\leq precision_{M,D}(\mu_2)$ si $\mu_1 < \mu_2$)

Ejercicio 5.3. Considerar la Figura Umbral de clasificación. En esta figura se ven instancias ordenadas según la probabilidad detectada por un clasificador (entre 0 y 1). Además, se encuentran marcados cuatro umbrales de decisión.

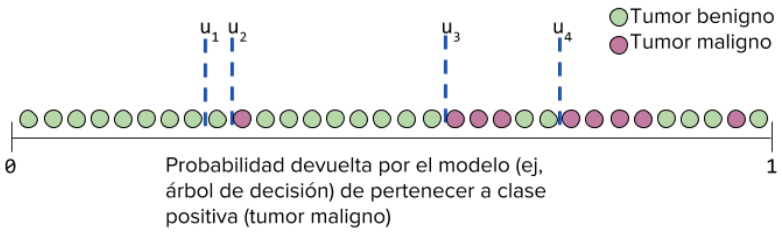


Figura 4: Umbral de clasificación

(a) Calcular las tablas de confusión resultantes para cada uno de los cuatro umbrales de decisión. Recordar que si la probabilidad está por debajo del umbral, la instancia será clasificada como perteneciente a la clase negativa; si está por encima, como clase positiva.

(b) ¿Cuál es el mejor umbral?

(c) Calcular la curva ROC para dicha clasificación.

Ejercicio 5.4.

(a) Escribir el pseudocódigo de la siguiente función:
CURVA-PR(LABELS : LIST⟨BOOL⟩, PROBAS: LIST⟨FLOAT⟩) : LIST⟨TUPLE⟨UMBRAL, VALORPREC, VALORREC⟩⟩
que devuelve los valores de precisión y recall junto a cada umbral explorado.

(b) ¿Qué cambios son necesarios para que esta función devuelva **también** los valores necesarios para construir una curva ROC?

Ejercicio 5.5. Verdadero o Falso (justificar)

(a) El AUC-ROC es invariante de escala. Mide qué tan bien se clasifican las predicciones, en lugar de sus valores absolutos.

(b) En caso que el ordenamiento de las instancias sea el mismo, AUC-ROC va poder distinguir un clasificador muy confiado (instancias clasificadas como positiva tienen scores muy altos, instancias clasificadas como negativas scores muy bajos) de uno poco confiado (scores menos extremos).

(c) AUC-ROC es invariante de umbral de clasificación. Mide la calidad de las predicciones del modelo independientemente del umbral de clasificación elegido.

(d) En los casos en los que existen grandes disparidades en el costo de los falsos negativos frente a los falsos positivos, puede ser fundamental minimizar un tipo de error de clasificación. Por ejemplo, al detectar spam, es probable que desee priorizar la minimización de los falsos positivos (incluso si eso resulta en un aumento significativo de los falsos negativos). AUC-ROC no es la mejor métrica para este tipo de optimización.

Ejercicio 5.6. Sea A un clasificador que tiene un F_1 de 0.80 (con un umbral de clasificación de 0.5), y sea B un clasificador que tiene un F_1 0.70 (también con un umbral de clasificación de 0.5). Sin embargo al cambiar el umbral a $0{,}4$ obtenemos F_1 de 0.76 y 0.80 respectivamente.

(a) Explicar por qué puede suceder este fenómeno dando un ejemplo aproximado.

(b) ¿Podemos concluir algo sobre el AUC Prec-Recall de estos dos modelos?

Ejercicio 5.7. Verdadero o Falso (justificar)

(a) Tanto *recall* como *precision* no toman en cuenta qué tan bien el modelo maneja los casos negativos.

(b) Un modelo que no produce falsos positivos tiene *precision* $= 1.0$.

(c) Un modelo que no produce falsos negativos tiene *recall* $= 1.0$.

(d) Si un clasificador devuelve probabilidades, la matriz de confusión se construye de manera ponderada según la probabilidad de cada clase.

(e) Si un clasificador devuelve probabilidades, hay muchas matrices de confusión asociadas dependiendo del umbral de clasificación.

(f) Si un clasificador devuelve probabilidades, hay infinitas matrices de confusión asociadas dependiendo del umbral de clasificación.

(g) Aumentar el umbral de clasificación produce que la *precision* siempre suba.

(h) Aumentar el umbral de clasificación produce que el *recall* baje o se mantenga igual.

(i) La métrica *precision* es parte fundamental del cálculo de la *curva ROC*.

Ejercicio 5.8. ¿Binaria o 2 clases?

Sean A y B clasificadores que distinguen entre imágenes de perros e imágenes de gatos. Al medir la performance del clasificador (utilizando F_1 para evitar los problemas de utilizar *accuracy*) y "gato" como clase positiva, obtenemos $F_1(A) = 0{,}9$, $F_1(B) = 0{,}8$. ¿Podemos concluir que el clasificador A es mejor que el clasificador B para este problema?
Resolver los siguientes ítems para poder responder a la pregunta:

(a) Al calcular F_1 utilizando "gato" como clase positiva, ¿importa qué ocurre con los perros que fueron clasificados correctamente? Revisar la Tercera Parte del notebook `notebook_metricas.ipynb` y decidir cuál clasificador funciona mejor, basándose en las métricas obtenidas. Observar el cambio que ocurre al intercambiar cuál es la clase positiva.

(b) ¿Para qué sirve el parámetro average en la función `f1_score` de la librería sklearn?

(c) ¿Qué sucede si la cantidad de instancias sobre las que fueron testeados es distinta? ¿Cómo se ve afectada la métrica F_1 al cambiar los True Negatives? Correr la Cuarta Parte del notebook `notebook_metricas.ipynb`. El gráfico muestra cómo varía la métrica F_1 al aumentar la cantidad de True Negatives (observar que estamos cambiando la cantidad de instancias sobre las que testeamos). ¿Qué se puede concluir de este experimento?

Ejercicio 5.9. Recordemos la métrica de `accuracy` para un problema de clasificación, para simplificar solo consideraremos el caso binario, sea y el vector de etiquetas verdadera, $\hat{y}$ las respectivas predicciones, y N la cantidad total de muestras ($y, \hat{y} \in \{0,1\}^N$):

$$\texttt{accuracy}(y, \hat{y}) = \frac{1}{N} \sum_{i=1}^{N} 1(\hat{y}^{(i)} = y^{(i)})$$

Donde $1(x)$ es la función indicadora (vale 1 si el argumento es verdadero, 0 caso contrario). Vamos a introducir una pequeña modificación en como contamos los aciertos para cada clase, supongamos que p es la proporción de la clase minoritaria (positiva):

$$\texttt{balanced\_accuracy}(y, \hat{y}) = \frac{1}{N} \left(\frac{1}{2p} \sum_{i=1}^{N} 1(\hat{y}^{(i)} = y^{(i)} = 1) + \frac{1}{2(1-p)} \sum_{i=1}^{N} 1(\hat{y}^{(i)} = y^{(i)} = 0) \right)$$

(a) ¿Qué rango de valores posibles tiene esta métrica? ¿Cuánto vale esta métrica si tenemos un clasificador constante que predice siempre la clase mayoritaria $\hat{y} = 0$? ¿Y si predice siempre la clase minoritaria?

(b) Escribir `balanced_accuracy` en términos de TP, FP, TN y FN.

(c) Mostrar que en el caso binario, esta métrica es equivalente al promedio aritmético entre *sensitividad* (true positive rate) y la *especificidad* (true negative rate).