Tarea 3: Clasificación

Profesor: Felipe Tobar
Auxiliares: José Díaz, Diego Garrido, Jou-Hui Ho, Luis Muñoz
Consultas: Diego Garrido.
Fecha entrega: 10/7/2019
Formato entrega: En grupos máximo dos integrantes entregue un informe en formato PDF con una **extensión máxima de 2 páginas solo las P1 y P2**. Este informe debe presentar y analizar sus resultados y detallar la metodología utilizada. Adicionalmente, usted debe entregar un Jupyter Notebook con los códigos que creó para resolver la tarea.

P1. Clasificador multiclase (1.0 pto)
En clases se vio que es posible extender la formulación de la clasificación lineal binaria para $k > 2$ clases utilizando k funciones lineales de la forma

$$y_k = a^\top x + b,$$

y luego asignar la clase a una muestra x mediante $k^\star = \arg\max_k y_k$. Muestre que este clasificador multiclase es coherente a diferencia de los métodos *one-versus-all* y *one-versus-one*. En particular, muestre que usando este clasificador propuesto:

- No quedan regiones del espacio de entrada sin clasificar. (**0.25 ptos**)

  Solución

  La función $argmax_k y_k$ consiste en evaluar k funciones lineales, como toda función lineal está bien definida para todo $\mathbb{R}^n$, luego el máximo de un conjunto finito de valores reales siempre existe, aunque puede no ser único, por ejemplo, puede darse que $a_i^T x + b_i = a_j^T x + b_j$ para algún x .

- No existen regiones del espacio de entrada asignadas a más de una clase. (**0.25 ptos**)

  Solución

  La región del espacio asociada a una clase se puede representar como un poliedro, donde un poliedro corresponde a una intersección finita de semiespacios ($\{x \in \mathbb{R}^n | a^T x \leq b\}$) y es un conjunto convexo[1]. Formalmente la región del espacio asociada a la clase i se puede escribir como sigue:

$$\begin{aligned}
P_i &= \{x \in \mathbb{R}^n | a_1^T x + b_1 \leq a_i^T x + b_i, \ldots, a_k^T + b_k \leq a_i^T x + b_i\} \\
P_i &= \{x \in \mathbb{R}^n | (a_1 - a_i)^T x \leq b_i - b_1, \ldots, (a_k - a_i)^T \leq b_i - b_k\} \\
P_i &= \{x \in \mathbb{R}^n | \hat{a}_1^T x \leq \hat{b}_1, \ldots, \hat{a}_k^T \leq \hat{b}_k\}
\end{aligned}$$

  Si un punto del espacio pertenece a más de una clase, por ejemplo, a la clase i y j este punto cumple $(a_j - a_i)^T x = b_i - b_j$, de hecho esta región corresponde al hiperplano que separa i de j, ya que en el espacio $(a_j - a_i)^T x < b_i - b_j$ no pueden haber puntos de la clase j y en el espacio $(a_j - a_i)^T x > b_i - b_j$ no pueden haber puntos de la clase i. En la porción del hiperplano que cumple $a_1 x^T + b_1 \leq \min\{a_i^T x + b_i, a_j^T x +$

[1] intersección finita de conjuntos convexos es convexa

$b_j\}, \ldots, a_k x^T + b_k \leq \min\{a_i^T x + b_i, a_i^T x + b_i\}$ (semiplano) la asignación no es única, es decir, la observación pertenece a ambas clases. Para evitar la asignación a múltiples clases podemos definir que la región $\hat{a}_j^T x \leq \hat{b}_j$ se asocia a la clase i y la región $\hat{a}_j^T x > \hat{b}_j$ a la clase j, donde $\hat{a}_j = a_j - a_i$ y $\hat{b}_j = b_i - b_j$.

Información adicional (no evaluado): El interior de un poliedro es la región en donde todas las desigualdades se satisfacen de forma estricta, por otro lado en el borde se tiene al menos una desigualdad satisfecha con igualdad. Que la asignación puede no ser única solo en el borde significa que no existe *overlapping* entre el interior de los poliedros, es decir, si tomamos dos puntos del interior de una misma clase se tiene que estos puntos pertenecen únicamente a esta clase (ya que todas las desigualdades son estrictas) y el segmento que une a ambos puntos también (por convexidad), cosa que no ocurre en los métodos *one-versus-all* o *one-versus-one*. Sea x_1, x_2 puntos del interior de P_i, sea $\bar{x} = \lambda x_1 + (1-\lambda)x_2$ con $\lambda \in (0,1)$, luego $\forall j \in [k] \setminus \{i\}$ se tiene:

$$(a_j - a_i)^T \bar{x} = \lambda(a_j - a_i)^T x_1 + (1-\lambda)(a_j - a_i)^T x_2 < \lambda(b_i - b_j) + (1-\lambda)(b_i - b_j) = (b_i - b_j)$$

, es decir, $a_j^T x + b_j < a_i^T x + b_i \; \forall j \in [k] \setminus \{i\}$, por ende argmax$_k y_k$ es único e igual a i.

- Eligiendo $k = 2$, este criterio colapsa al presentado para el caso binario. (**0.25 ptos**)

  Solución

  Si eligimos $k = 2$ nos queda $P_1 = \{x \in \mathbb{R} | (a_2 - a_1)^T + b_2 \leq b_1 - b_2\}$ y $P_2 = \{x \in \mathbb{R} | (a_2 - a_1)^T + b_2 \geq b_1 - b_2\}$, luego $P_1 \cap P_2 = \{x \in \mathbb{R}^n | (a_2 - a_1)^T x = b_1 - b_2\}$, obteniéndose el hiperplano separador.

- Muestre cómo es la región de decisión y el subconjunto correspondiente a cada clase. (**0.25 ptos**)

  Solución

  En la figura 1 se muestra para $k = 3$ como luce la región de cada una de las clases, en donde se observa que cada región es un poliedro y la intersección solo puede ocurrir en sus bordes.

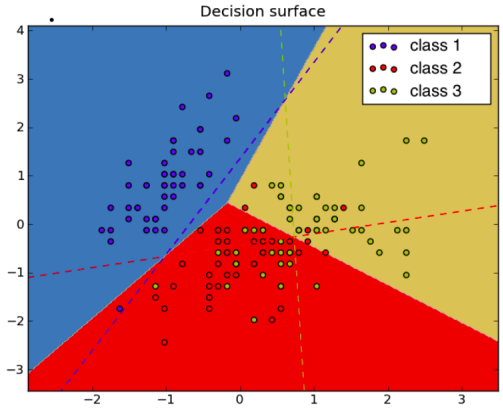

Fig. 1: División del espacio para el clasificador discriminante con $k = 3$. La línea punteada azul es y_1, la línea punteda roja es y_2 y la línea punteada amarilla es y_3

P2. Detección de barcos en imágenes satelitales (4 ptos) **Solución**

El objetivo de esta pregunta es ayudar a abordar la difícil tarea de detectar la ubicación de grandes barcos usando imágenes satelitales. La importancia de esta tarea es múltiple, en particular, la detección automática de barcos podría facilitar el monitoreo de los niveles de actividad del puerto y el análisis de la cadena de suministro. El dataset disponible consiste de imágenes satelitales de barcos recopiladas en las áreas de la bahía de San Francisco y la bahía de San Pedro de California. Este incluye 4000 imágenes 80x80 RGB, con la etiqueta "ship" (1) y "no-ship" (0). Más detalles sobre cómo cargar los datos y como extraer vectores de características a partir de una imagen en el notebook **tutorial.ipynb**. **Observación:** en el informe solo debe responder las preguntas planteadas, no adjunte código y debe ser lo más breve posible en los puntos a) y c).

a) (0.5 ptos.) Cargue los datos y extraiga los vectores de características (vea **tutorial.ipynb**), pues las siguientes partes de esta pregunta serán todas en base a las características, no a las *entradas crudas*. Divida los datos de forma aleatoria en conjuntos de entrenamiento (70%), validación (15%) y testeo (15%).

   Observación: Le puede ser útil la función *train_test_split* de *sklearn*.

   Solución

   Al particionar los datos en conjunto de entrenamiento, validación y testeo debemos asegurarnos que la distribución de los datos sea similar en ambos conjuntos, sino el modelo no podrá generalizar correctamente a observaciones que provienen de una distribución completamente distinta a la que generó los datos de entrenamiento. Una forma sencilla de hacer esto es seguir los siguientes pasos:

   - Barajar los datos, los datos pudieron ser recolectados siguiendo un patrón, por lo la distribución de las primeras 100 observaciones puede ser completamente diferente a las siguientes 100 observaciones, de hecho, en la tarea las primeras imágenes son de barcos y luego le siguen las que no tienen barcos (**0.2 ptos**).
   - Una vez barajados los datos hacer muestro aleatorio independiente, esto garantizará que la distribución de los tres conjuntos sea más o menos similar (**0.2 ptos**).
   - Para garantizar reproducibilidad de los resultados es necesario fijar una semilla aleatoria (**0.1 ptos**).

b) (1.5 ptos.) Implemente la regresión logística bajo el método del descenso del gradiente estocástico (ver sección 4.5.2 del apunte). Su implementación debe incluir dos hiperparámetros: la tasa de aprendizaje η, y el número de épocas de entrenamiento e, donde una época corresponde a una pasada por todo el conjunto de entrenamiento (pues recuerde que el gradiente estocástico solo usa una observación a la vez). Grafique la log-verosimilitud normalizada por el número de observaciones (eje y) versus el número de épocas (eje x), tanto para el conjunto de entrenamiento como el de validación para distintas tasas de aprendizaje, para ello considere $\eta \in \{0.01, 0.05, 0.1\}$ y épocas $e \in \{1, \ldots, 100\}$. Luego, en base a los gráficos responda la siguiente pregunta: ¿Cuál es la configuración óptima de η y *épocas*? **Justifique su elección y explique qué pasa con tazas de aprendizaje muy grandes o pequeñas**.**Observaciones:** La clase anterior debe ser escrita usando el stack de Python y Numpy, no está permitido el uso de algún solver para optimizar la log-verosimilitud. El archivo **tutorial.ipynb** cuenta con una plantilla a modo de guía para su implementación.

   Solución

La probabilidad de "ship" viene dada por $P(y_i = 1) = \frac{1}{1+e^{-w^T x_i}} = \sigma_i$ y la probabilidad de "no-ship" viene dada por $P(y_i = 0) = 1 - P(y_i = 1)$ (puede ser al revés, depende de como se halla modelado) (**0.5 ptos método fit**). Luego la asignación binaria del método *predict* asigna a la clase "ship" si $P(y_i = 1) \geq 0.5$ y a la clase "no-ship" si no (**0.1 ptos método predict**) (el umbral de corte se puede tunear en un conjunto de validación). La log-verosimilitud $l(w) = \sum_{i=1}^{N} y_i log(\sigma_i) + (1 - y_i) log(1 - \sigma_i)$ (**0.2 ptos método log_verosimilitud**).

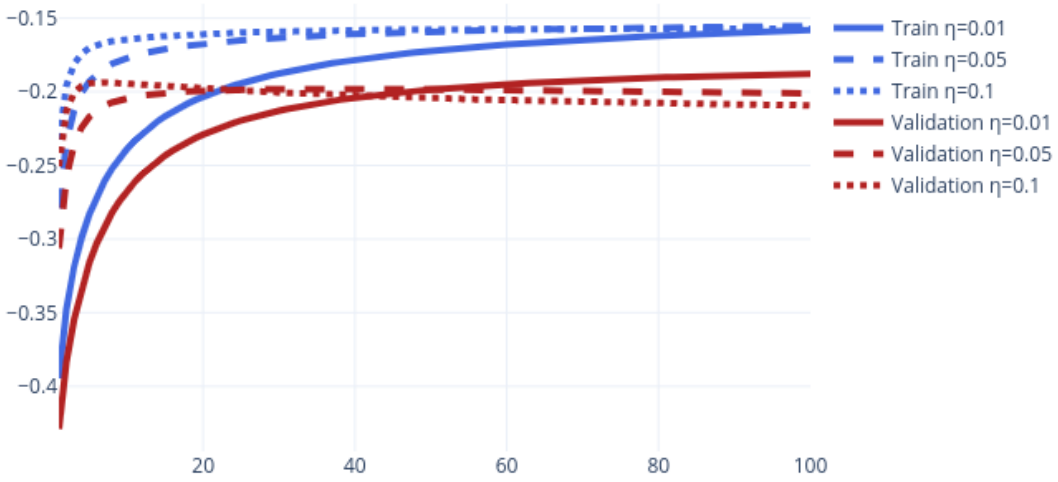

Fig. 2: log-verosimilitud de la regresión logística tanto en el conjunto de entrenamiento como en el conjunto de test bajo el algoritmo de gradiente estocástico.

El objetivo es encontrar la combinación de hiperparámetros con mejor desempeño en el conjunto de validación (generalización), es decir, con mayor log-likelihood, en este caso particular fue (η, *epochs*) = (0.01, 100) (**0.2 ptos**). En la Figura 2 se puede observar que para $\eta \in \{0.05, 0.1\}$ la log-likelihood alcanza un máximo en el conjunto de validación, tras este máximo a más épocas de entrenamiento el desempeño en validación comienza a empeorar y el de entrenamiento tiende a mejorar, observandosé un claro overfitting (de hecho a SGD siempre mejora la log-verosimilitud en entrenamiento, por lo que si el número de épocas es infinito w explota) (**0.2 ptos**). El efecto de η en el entrenamiento es la velocidad de convergencia, en este caso a mayor η más rápidamente alcanza un máximo, en el caso de $\eta = 0.01$ que es el valor se tiene que en las 100 épocas aún no alcanza un máximo, por lo que podría seguir mejorando si aumentamos la cantidad de épocas de entrenamiento. En conclusión, pequeñas tasas de aprendizaje requieren más épocas de entrenamiento dado que los pesos de una época a otra cambian ligeramente (también es más fácil quedar atrapado en óptimos locales, aunque no en este caso, ya que la función objetivo es convexa), en cambio tasas grandes de aprendizaje convergen más rápido pero pueden tener problemas de divergencia (**0.3 ptos**). En la práctica se usan métodos de gradiente estocástico que adaptan la tasa de aprendizaje, donde la tasa de aprendizaje decrece en función del número de épocas/batchs de entrenamiento.

c) (1.0) Implemente el discriminante lineal de Fisher. En la sección 4.3 del apunte se describe cómo obtener a, con lo que luego podemos definir un umbral b para asignar un x a $\mathcal{C}_1$ si $a^T x \geq -b$ y a $\mathcal{C}_2$ en caso contrario. Para efectos de esta pregunta considere:

$$-b = a^\top \frac{1}{2}(\mu_1 + \mu_2) = \frac{1}{2}(m_1 + m_2)$$

.

Observaciones: La clase anterior debe ser escrita usando el stack de Python y Numpy. El archivo **tutorial.ipynb** cuenta con una plantilla a modo de guía para su implementación.

Solución

Una observación se asigna a la clase "ship" (1) si $a^T x \geq -b$ y la clase "no-ship" (2) si no, con $a = S_W^{-1}(\mu_1 - \mu_2)$, con $S_W = \sum_{n \in \mathcal{C}_1}(x_n - \mu_1)(x_n - \mu_1)^T + \sum_{n \in \mathcal{C}_2}(x_n - \mu_2)(x_n - \mu_2)^T$. Distribución del puntaje: **0.8 ptos método fit** y **0.2 ptos método predict**.

d) (1.0 pts.) ¿Cuál modelo (regresión logística o discriminante de Fisher) tiene mejor desempeño en el conjunto de testeo? Para responder esta pregunta proponga y justifique una métrica para evaluar el desempeño de un método general de clasificación, luego, evalúe ambos clasificadores usados y compárelos. ¿Por qué cree usted que es necesario comparar ambos modelos en el conjunto de testeo en vez de el conjunto de entrenamiento o validación?

Observaciones: para el caso de la regresión logística considere la mejor configuración del punto b) y a la hora de escoger la métrica tenga en cuenta que las clases están **desbalanceadas**, i.e., la cantidad de elementos por clase es distinta.

Solución:

Observación en la elección de la métrica: Debe proponerse una métrica adecuada para el dominio del problema que es discreto, si se proponen métricas para problemas de regresión (como error cuadrático medio) dar 0 ptos. Además, la métrica debe permitir comparar ambos modelos, en este caso la verosimilitud no sería valida (a menos que se aplica función sigmoide a FLDA), dar 0 ptos en este caso. Si se proponen métricas que no consideran el desbalanceo de clases como *accuracy* dar la mitad del puntaje y si no se justifica la métrica propuesta descontar 0.2 ptos.

Una forma de minorizar el efecto de clases desbalanceadas es construir métricas que le dan más importancia la clase minoritaria, puesto que suele ser la clase que más nos interesa. En este caso, una muy buena idea sería trabajar con matrices de utilidad, cuantificando el beneficio/costo de predecir bien/mal tanto la clase "ship" como "no-ship". En general, cuantificar esos costos y beneficios es una tarea difícil, por lo que se suele hacer es trabajar con las tasas de acertividad (*recall*) y precisión (*precision*) de las clases o mezclas de estas, por ejemplo, *macro recall*, que es el promedio simple entre la tasa de acertividad de todas las clases, esto significa que dado el desbalanceo presente un acierto de la clase "ship" equivale a 3 aciertos de la clase "no-ship". En este caso el mejor modelo fue FLDA con un macro recall del 91% por sobre la regresión logística que obtuvo un 85% (**0.5 ptos**).

El objetivo de toda tarea predictiva es desarrollar modelos con alta capacidad de generalización. Para medir esta capacidad hacemos una simulación, la cual consiste de seleccionar una muestra de nuestra datos, la cual asumimos que no sabemos su etiqueta y será el conjunto donde mediremos la verdadera capacidad predictiva de nuestros modelos de acuerdo a la métrica escogida. Este conjunto es distinto al de entrenamiento por que en este conjunto calibramos los parámetros de los modelos y al de validación ya que en este calibramos los hiperparámetros de uno de nuestros modelos. En el conjunto no debemos calibrar nada de nuestros modelos, ni siquiera sus hiperparámetros, pues en una correcta simulación no deberíamos conocer de antemano la etiqueta, así de esta manera podemos medir la verdadera capacidad predictiva de los modelos (**0.5 ptos**).

P3. Proyecto curso (1 pto)

Esta parte debe ser respondida por **un único** integrante del grupo, donde debe indicar los demás integrantes del grupo. Se habilitará una tarea aparte para su entrega en formato PDF cuya **extensión máxima es media página**.

a) (0.5 ptos) Enmarque su proyecto en algún tipo de aprendizaje (clasificación, regresión, clustering, aprendizaje, reforzado, etc.). Identifique según el caso el rol de sus variables, por ejemplo, si su proyecto es de clasificación, describa su variable independiente, sus características y sus clases. Si su proyecto no parece caber evidentemente en ninguno de estos casos, explíquelo en detalle y propongo cómo lo abordará desde el punto de Aprendizaje de Máquinas.

b) (0.5 ptos) Proponga al menos 2 métodos para resolver su problema, fundamentando brevemente su elección.