6. Clasificadores

Ejercicio 6.1.

- Explica con tus palabras qué es el clasificador óptimo de Bayes.
- ¿Cuál es la razón para no construir siempre este clasificador en la práctica?
- Imaginemos que simulamos datos para clases Azul y Verde. Para la clase verde muestreamos de una distribución normal (μ=0, σ=2), y para la clase azul de una normal (μ=3, σ=1). Además supongamos que la probabilidad de la clase verde es 0.2. En resumen $X|Y = Verde \sim N(0, 2)$, $X|Y = Azul \sim N(3, 1)$, $P(Y = Verde) = 0.2$ ¿Cuál sería la clase asignada por el clasificador óptimo de Bayes para una instancia $\mathbf{x}^{(i)} = 2$?
- Explique cuál sería el proceso para dibujar las fronteras de decisión en una grilla (es decir, para instancias que viven en $\mathbb{R}^2$) conociendo las distribuciones de $X|Y = c$ y los priors $P(Y = c)$ para toda clase.

Ejercicio 6.2. Dibujar las fronteras de decisión, indicando la clase de predicción de cada región, para un ejemplo de modelo para clasificación binaria que:

I) sobreajusta (overfitting).

II) subajusta (underfitting).

III) es el resultado de aplicar un árbol de decisión de altura máxima 3 (raíz, hijos, nietos).

IV) es el resultado de aplicar K-vecinos más cercanos con K = n.

V) es el resultado de aplicar K-vecinos más cercanos con K = 1.

VI) es el resultado de aplicar LDA (tener en cuenta las probabilidades a priori).

Ejercicio 6.3. Se tienen instancias con sólo dos atributos: altura de una persona (medido en metros) y edad de la persona (medida en años). Se quiere saber si la persona es o no es basquetbolista profesional tomando en cuenta la experiencia de muchas personas.

(a) ¿Es buena idea utilizar el algoritmo de K-vecinos más cercanos con estos datos?

(b) ¿Suponiendo que se utiliza dicho modelo, será útil realizar alguna transformación a los datos previo a ejecutar el algoritmo? ¿Cuál? ¿Por qué?

Ejercicio 6.4. Determinar cuales de las siguientes distribuciones alcanzan por sí solas para decidir la clase de una instancia $x^{(t)}$ siguiendo la receta del clasificador óptimo de bayes (suponer clasificación binaria con clases "0" y "1").

(a) $P(Y = 1|X = x^{(t)})$

(b) $P(X = x^{(t)})$

(c) $P(X = x^{(t)}|Y = 1)$

(d) $P(X = x^{(t)}|Y = 1)$ y $P(X = x^{(t)}|Y = 0)$

(e) $P(X = x^{(t)}|Y = 1)$ y $P(Y = 0)$

(f) $P(Y = 1)$ y $P(Y = 0)$

Ejercicio 6.5.
Consideremos $\delta_c(x)$ la *función discriminante* para la clase c en un problema de clasificación multiclase con k clases y $x \in \mathbb{R}^p$ con $p = 1$:

$$\delta_c(x) = x \cdot \frac{\mu_c}{\sigma^2} - \frac{\mu_c^2}{2\sigma^2} + \log \pi_c \qquad \text{en donde } \pi_c = P(Y = c)$$

Bajo las suposiciones de LDA, se puede derivar

$$Pred(x^{(i)}) = \underset{c \in Clases}{\arg\max}\, P(Y{=}c|X{=}x^{(i)}) = \underset{c \in Clases}{\arg\max}\, \delta_c(x^{(i)})$$

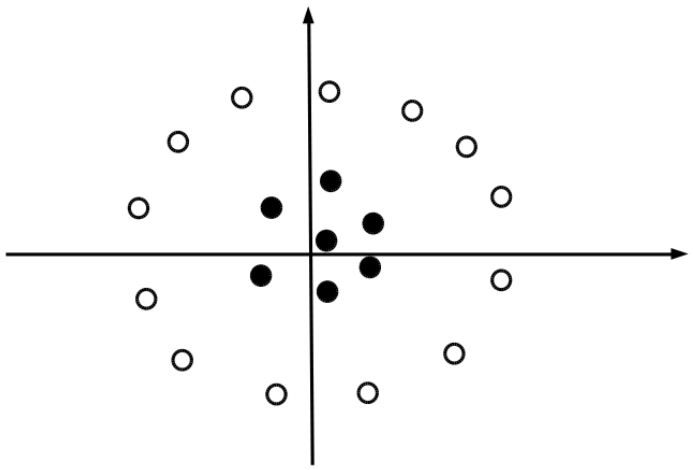
Es decir, para predecir la clase de una instancia, basta con evaluar esta función discriminante para cada clase y conservar la que retorne el mayor valor.

(a) Demostrar esta igualdad para el caso $p = 1$.

(b) Demostrar que la frontera de decisión para el caso con $k = 2$, $\pi_1 = \pi_2$ es $x = (\hat{\mu}_1 + \hat{\mu}_2)/2$.

(c) La palabra *linear* en LDA se debe a que $\delta_k(x)$ es una función lineal en x. Mostrar que si se elimina la suposición de que todas las clases tienen la misma varianza, entonces la función discriminante pasa a ser cuadrática en x. (A esa técnica se la conoce como *quadradic discriminant analysis*, o QDA).

Ejercicio 6.6. En la Figura 6 se muestran 20 puntos bidimensionales en el espacio de atributos. Explicar qué puede hacer SVM con algún kernel para discriminarlos correctamente, y por qué SVM con un kernel lineal fallaría inexorablemente.



Ejercicio 6.7. Describir el sesgo inductivo de Gaussian Naive Bayes (recordar la siguiente fórmula). Pista 1: hay dos símbolos de approx ($\approx$) en la fórmula. Pista 2: ver los subíndices en la última línea de la fórmula.

$$
\begin{aligned}
Pred(x^{(i)}) &= \arg\max_{c \in Clases} P(Y{=}c | X{=}x^{(i)}) && (1)\\
&= \arg\max_{c \in Clases} \frac{P(Y{=}c) P(X{=}x^{(i)} | Y{=}c)}{P(X{=}x^{(i)})} && (2)\\
&= \arg\max_{c \in Clases} P(Y{=}c) P(X{=}x^{(i)} | Y{=}c) && (3)\\
&= \arg\max_{c \in Clases} P(Y{=}c) P(X_1 = x_1^{(i)} \wedge \cdots \wedge X_p = x_p^{(i)} | Y{=}c) && (4)\\
&\approx \arg\max_{c \in Clases} P(Y{=}c) \prod_{j=1}^{p} P(X_i = x_j^{(i)} | Y{=}c) && (5)\\
&= \arg\max_{c \in Clases} P(Y{=}c) \prod_{j=1}^{p} pdf_c(x_j^{(i)}) && (6)\\
&\approx \arg\max_{c \in Clases} \hat{P}(Y{=}c) \prod_{j=1}^{p} f_{norm}(x_j^{(i)}; \hat{\mu_{c,j}}, \hat{\sigma_j}) && (7)
\end{aligned}
$$

Ejercicio 6.8. *La maldición de la dimensión* - Cuando el número de atributos p es grande, suele haber un deterioro en el rendimiento de KNN y otros algoritmos que realizan predicciones usando solo observaciones cercanas a la que queremos predecir. Este fenómeno es conocido como la maldición de la dimension, y se refiere a que los métodos no paramétricos suelen rendir mal cuando p es grande. Vamos a investigar esta maldición.

1. Supongamos que tenemos un conjunto de instancias, cada una con mediciones de un atributo $p = 1$, X. Asumimos que X está uniformemente distribuida en $[0, 1]$. Nuestro objetivo es predecir el valor de respuesta para una nueva instancia. Para realizar esta predicción, utilizaremos un subconjunto específico de instancias existentes. Seleccionaremos únicamente aquellas instancias cuyo valor de X se encuentre dentro del rango de los valores 10 % más cercanos al valor de X de la nueva instancia. Por ejemplo, para predecir la respuesta de una instancia cualquiera que tenga como atributo $X = 0.6$, usaremos instancias que se encuentren en el rango $[0.55, 0.65]$. En promedio, ¿qué proporción de las instancias disponibles usaremos para hacer la predicción?

2. Ahora supongamos que tenemos un conjunto de observaciones, cada una con mediciones de $p = 2$ atributos, X_1 y X_2. Asumimos que (X_1, X_2) están uniformemente distribuidas en $[0, 1] \times [0, 1]$. Nuestro objetivo es predecir el valor de respuesta para una nueva instancia. Para hacer esta predicción, utilizaremos un subconjunto específico de instancias existentes. Seleccionaremos solo aquellas instancias cuyos valores de X_1 y X_2 se encuentren simultáneamente dentro del 10 % más cercano a los valores correspondientes de X_1 y X_2 de la nueva instancia. Por ejemplo, para predecir la respuesta de una instancia con $X_1 = 0.6$ y $X_2 = 0.35$, usaremos las instancias que cumplan que se encuentren en el rango $[0.55, 0.65]$ para X_1 y en el rango $[0.3, 0.4]$ para X_2. En promedio, ¿qué proporción de las instancias disponibles usaremos para hacer la predicción?

3. Ahora supongamos que tenemos un conjunto de instancias con $p = 100$ atributos. Nuevamente, las instancias están uniformemente distribuidas en cada atributo, y cada atributo varía de 0 a 1. Nuestro objetivo es predecir el valor de respuesta para una nueva instancia. Para hacerlo, utilizaremos las instancias más similares a ella. Específicamente, consideraremos aquellas instancias cuyos atributos se encuentren dentro del 10 % más cercano en cada dimensión con respecto a la nueva instancia. ¿Qué proporción de las instancias disponibles usaremos para hacer la predicción?

4. Teniendo en cuenta los items anteriores, ¿qué sucede a medida que p se hace más grande? ¿cómo afecta esto a KNN?

5. Ahora supongamos que queremos hacer una predicción para una nueva instancia creando un hipercubo en p dimensiones centrado en esta instancia que contenga, en promedio, el 10 % de las observaciones de entrenamiento. Para $p = 1$, 2 y 100, ¿cuál es la longitud de cada lado del hipercubo?