Cuestionario 1. Julio A. Fresneda García - 49215154F

Ejercicio 1.

a) Los datos de aprendizaje que podríamos usar pueden ser el color de piel y la forma de los ojos, nariz y boca. El aprendizaje debe ser supervisado, ya que para los aprendizajes no supervisados debe haber un patrón claro (gráficamente deben haber clusters), y en este caso la línea entre una raza y otra es muy difusa. Por refuerzo tampoco serviría, por que hay varias razas, y a la hora de ajustar, en los casos de error el algortimo sabría que se ha equivocado, pero no sabría cual era la respuesta correcta para aprender.

b) Para clasificar las cartas por distrito postal necesitamos reconocer automáticamente los números del distrito postal de cada carta, por lo que el aprendizaje iría enfocado a ésto. Para eso, podemos usar datos como la simetría horizontal y vertical, la intensidad, si hay "agujeros" como en el número 6, 0 u 8, etc. En este aprendizaje, si los números fueran escritos a máquina/ordenador, podría ser no supervisado (se formarían clusters y bastaría con decirle al algoritmo a qué número corresponde cada clúster). Pero si los números son escritos a mano, la diferencia entre números es más difusa, y el algoritmo debería ser supervisado.

c) En este caso el aprendizaje debe ser no supervisado, ya que no hay respuesta correcta o incorrecta para aprender de ella: Sólo tenemos el historial de datos de los índices de mercado de valores, y quizás algunas reglas que la función de aprendizaje debe cumplir. Por lo tanto, el programa deberá estimar la función que sigue el índice de mercados, con el historial de datos.

d) Aquí podemos usar aprendizaje por refuerzo. Como datos, podemos usar la distancia que el robot ha recorrido, la localización del objeto a rodear. Con ésto, podemos aprender la función que nos dice la trayectoria que el robot debe seguir. Si el robot se choca, se aleja en otra dirección, etc. podemos medir el error numéricamente, por lo que el programa puede aprender de ese error numérico tomándolo como refuerzo, y corregir la trayectoria.

Ejercicio 2.

La diferencia entre aprendizaje y diseño es que las características de los problemas de aprendizaje no están definidas con precisión, por lo que el programa debe usar una base de datos para aprender. Los problemas de diseño sí que tienen características exactas de las cuales se puede obtener un algoritmo sin necesidad de aprender a partir de datos.

a) Aprendizaje, ya que las características típicas de un grupo de animal pueden darse en algún que otro animal de otro grupo. Por ejemplo, que tenga alas puede ser una característica determinante de las aves, pero un murciélago sería una excepción. No hay características exclusivas de un grupo de animales, por lo que no se puede diseñar un algoritmo que clasifique a partir de características, sin datos, por lo que necesitamos datos para aprender y clasificar.

b) Para asegurar el éxito de la campaña, lo mejor sería obtener datos de otras campañas pasadas y obtener un patrón, por lo que sería una tarea de aprendizaje.

c) Es un problema de aproximación por aprendizaje, ya que lo que una persona puede considerar spam, otra puede cosiderarla útil. Por ejemplo, un correo de Twitter informándote que tienes nuevos seguidores algunas personas pueden considerarlo spam, y otras puede considerarlo un correo útil. Por eso no se puede enfocar por diseño, no hay características concretas que indudablemente cataloguen de spam un correo. Se podría usar una aproximación por aprendizaje en la cual el programa aprenda teniendo en cuenta correos abiertos o correos no abiertos/borrados del usuario.

d) Aproximación por aprendizaje. No se puede obtener un patrón claro a partir del cual diseñar un algoritmo sin necesidad de datos de referencia.

e) Aproximación por diseño. Podemos saber perfectamente las características del problema (media de coches que pasan, tiempo de semáforos en rojo, etc) por lo que se podría diseñar un algoritmo sin necesidad de datos de referencia para aprender.

Ejercicio 3.

El primer paso sería obtener los datos de las frutas. Podemos obtener el color, la textura y el peso, por ejemplo.

X sería la matriz de características: $X_i = (x_{i1}, x_{i2}, x_{i3})$, X entre 1 y N, siendo N el número total de frutas de las cuales tenemos datos, cada columna corresponde al color, textura y peso, y cada fila contiene los datos de color, textura y peso de una fruta en concreto.

Y sería la etiqueta. Por ejemplo, si $y < -1$ la fruta se etiquetaría de papaya, si $y > 1$ la fruta se etiquetaría de guayaba y si y pertenece a $[-1, 1]$ la fruta se etiquetaría de mango.

$\mathcal{F}$ es la función ideal (óptima) del problema, $f : X \to Y$, es decir, la función que a partir de un trío x_{i1}, x_{i2} y x_{i3} obtiene la etiqueta de la fruta a la que corresponde.

D el conjunto de datos que tenemos para aprender: una matriz compuesta por filas (x_1, y_1),(x_2, y_2),(x_3, y_3)...

Considero que es un problema de etiquetas con ruido, ya que puede haber, por ejemplo, una cierta probabilidad de que un porcentaje de mangos posean un color similar al de una guayaba.

Ejercicio 4.

Sabemos que:

$X = UDV^T$
$D^T = D$
$X^T = (UDV^T)^T = VD^TU^T = VDU^T$
$U^TU = V^TV = I$

Operando:

$$X^TX = UDV^TVDU^T = UDIDU^T = UD^2U^T$$
$$XX^T = VDU^TUDV^T = VDIDV^T = VD^2V^T$$

Una propiedad de X^TX es que no tenemos información de V
Una propiedad de XX^T es que no tenemos información de U

SVD puede servir para ver si podemos agrupar los valores de las filas o columnas en sets o conceptos. U sirve para ver si podemos agrupar los valores de las filas en conceptos, V sirve para ver si podemos agrupar los valores de las columnas en conceptos, y D sirve para ver el peso o importancia de cada concepto, es decir, la cantidad de datos que se pueden agrupar en ese concepto. La suma de las diagonales de X^TX y XX^T sirve para ver el peso de la posible agrupación de filas-a-concepto y columnas-a-concepto. Si la suma de la diagonal de X^TX es mayor que la de XX^T, hay mas valores de filas que se pueden agrupar en conceptos que valores de columnas, y viceversa.

Ejercicio 5.

a)

$$11^T = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \times \begin{bmatrix} 1 & 1 & \dots & 1 \end{bmatrix}$$

$$11^TX = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & 1 \end{bmatrix} \times X = \begin{bmatrix} \sum_{i=1}^{N} cov(x_ix_1) & \sum_{i=1}^{N} cov(x_ix_2) & \dots & \sum_{i=1}^{N} cov(x_ix_N) \\ \sum_{i=1}^{N} cov(x_ix_1) & \sum_{i=1}^{N} cov(x_ix_2) & \dots & \sum_{i=1}^{N} cov(x_ix_N) \\ \sum_{i=1}^{N} cov(x_ix_1) & \sum_{i=1}^{N} cov(x_ix_2) & \dots & \sum_{i=1}^{N} cov(x_ix_N) \\ \sum_{i=1}^{N} cov(x_ix_1) & \sum_{i=1}^{N} cov(x_ix_2) & \dots & \sum_{i=1}^{N} cov(x_ix_N) \end{bmatrix}$$

En 11^TX cada elemento de cada fila contiene la suma de todos los valores de su columna de X, por lo que podemos ver si en general cada característica depende mucho o poco de las demas.

Ejercicio 6.

a) ¿Es H simétrica?

Sabemos que:
$H = X(X^TX)^{-1}X^T$
$(ABC)^T = C^TB^TA^T$

Entonces:

$H^T = (X(X^TX)^{-1}X^T)^T = X^{T^T}((X^TX)^{-1})^TX^T = X((X^TX)^T)^{-1}X^T = X(X^TX^{T^T})^{-1}X^T = X(X^TX)^{-1}X^T = H$

b) ¿Es $H^2 = H$?

$H^2 = (X(X^TX)^{-1}X^T)(X(X^TX)^{-1}X^T) = X(X^TX)^{-1}X^TX(X^TX)^{-1}X^T = X(X^TX)^{-1}(X^TX)(X^TX)^{-1}X^T = XI(X^TX)^{-1}X^T = X(X^TX)^{-1}X^T = H$

c) La matriz H se obtiene multiplicando la matriz de datos por la pseudoinversa de estos datos, y en modelos de regresión describe la influencia que tiene cada valor de los datos sobre cada valor ajustado

Ejercicio 7.

Ejercicio 8.

a)

$P(Y) = h(x)^Y(1 - h(x))^{1-Y}$

Ejercicio 9.

Tenemos la función:

$$\nabla E_{in}(w) = -\frac{1}{N}\sum_{n=1}^{N}\frac{y_n x_n}{1+e^{y_n w^T x_n}}$$

Esta función se verifica, ya si se comete un error, el denominador sería pequeño ($y_n w^T x_n \leq 0$, por lo que $e^{y_n w^T x_n}$ es pequeño), lo que hace que la fracción sea grande, e influya en el sumatorio más que si fuese pequeña, aumentando $\nabla E_{in}(w)$. Si no hubiese errores, $\nabla E_{in}(w)$ sería muy pequeño (y negativo), por lo que asumiríamos que el error es ≈ 0. En cambio, si hubiese errores, $\nabla E_{in}(w)$ se volvería positivo y más grande que si no hubiera errores.

Un ejemplo mal clasificado si que contribuye al gradiente más que un ejemplo más clasificado, por lo siguiente:

Si tenemos un ejemplo bien clasificado, el signo de y_n y de $w^T x_n$es el mismo, por lo que $y_n w^T x_n \geq 0$.
Si $y_n w^T x_n \geq 0$, $e^{y_n w^T x_n}$será muy grande, por lo que $\frac{y_n x_n}{1+e^{y_n w^T x_n}}$será muy pequeño, contribuyendo poco al gradiente.

En cambio, si tenemos un ejemplo mal clasificado, el signo de y_n y de $w^T x_n$es distinto, por lo que $y_n w^T x_n \leq 0$.
Si $y_n w^T x_n \leq 0$, $e^{y_n w^T x_n}$será muy pequeño, por lo que $\frac{y_n x_n}{1+e^{y_n w^T x_n}}$será muy grande, contribuyendo mucho al gradiente.

Ejercicio 10.

El SGD funciona así:

$w_j = w_j - \eta \frac{\partial E_{in}(w)}{\partial w_j}$

El PLA funciona así:

$w_{updated} = w_{current} + yx$ si $sign(w^T x_i) \neq y_i$
$w_{updated} = w_{current}$ si $sign(w^T x_i) = y_i$

Si tomamos la función $e_n(w) = max(0, -y_n w^T x_n)$ como función de error:
$e_n(w) = 0$ si $sign(w^T x) = y_n$, ya que $-y_n w^T x_n$ sería negativo, menor que 0.
$e_n(w) = -y_n w^T x_n$ si $sign(w^T x) \neq y_n$, ya que $-y_n w^T x_n$ sería positivo, mayor que 0.

Teniendo esto en cuenta, el PLA podría describirse así:
$w_{updated} = w_{current} - e_n(w)$

Y el SGD, con learning rate de 1 y tomando la función $e_n(w) = \frac{\partial E_{in}(w)}{\partial w_j}$, podría describirse igual:
$w_j = w_j - e_n(w)$

Por lo que sí se puede interpretar el algoritmo PLA como SGD en este caso.