Universidad Técnica Federico Santa María
Departamento de Informática

Pauta Certamen - Máquinas de Aprendizaje

Semestre II 2015 - Jueves 14.1.16

1 Preguntas Cortas

1. (a) ¿Es cierto que si resolvemos el problema de regresión lineal minimizando la suma de cuadrados con gradiente descendente podemos encontrar varios óptimos locales?

   (b) ¿El algortimo de backprogation genera una capacidad de generalización de la MLP independiente del número de capas que esta tenga?

   (c) ¿Un algoritmo que escoge una hipótesis en un espacio más abundante tendrá mejor capacidad de generalización?

   (d) Si tenemos un algoritmo de optimización que tarda el doble que el otro en realizar una iteración del algoritmo. ¿Ese algoritmo convergerá más lento que el otro?

   (e) ¿Si consideramos un gran número de iteraciones. Adaboost dará error de training cero independiente de los clasificadores que se estén usando?

   (f) **[postgrado]** Considere un punto bien clasificado, pero lejos de la frontera de decisión. ¿Por qué la frontera de decisión de la SVM no se ve afectada por este punto, en cambio, la frontera de decisión de la regresión logística si se ve afectada?

   (g) **[postgrado]** Considere el problema primal de la C-SVM vista en clases. ¿En qué valor fijaría la constante C si el problema es linealmente separable?

2 Regresión Lineal

2. Considere un modelo de regresión lineal con p parámetros, obtenido usando el método de mínimos cuadrados a partir de una muestra aleatoria

$$S_E = \{(x_1, y_1), \ldots, (x_N, y_N)\}.$$

   Sea $\hat{\beta}$ el estimador de mínimos cuadrados. Suponga que tenemos el conjunto de prueba $S_T = \{(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_M, \tilde{y}_M)\}$ obtenido aleatoriamente desde la misma población. Si $R_{tr}(\beta) = \frac{1}{N}\sum_{i=1}^{n}(y_i - \beta^T x_i)^2$ es el error cuadrático del conjunto de entrenamiento y $R_{ts}(\beta) = \frac{1}{M}\sum_{i=1}^{n}(\tilde{y}_i - \beta^T \tilde{x}_i)^2$ es el error cuadrático del conjunto de prueba.

   Demuestre que:

$$E[R_{tr}(\hat{\beta})] \leq E[R_{ts}(\hat{\beta})]$$

3 Teoría de Aprendizaje

3. Usando la técnica de los multiplicadores de Lagrange, muestre que la minimización de la función de error

$$\widehat{\beta}^{ridge} = \text{argmin}_\beta \left\{ \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\} \quad (1)$$

es equivalente a minimizar

$$\begin{aligned} \widehat{\beta}^{ridge} &= \text{argmin}_\beta \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \quad (2) \\ \text{s.a.} \quad & \sum_{j=1}^{p} \beta_j^2 \leq s \end{aligned}$$

Discuta la relación entre los parámetros λ y s

4 Kernels

4. Responda las siguientes preguntas (el puntaje para postgrado aparece en negrita) :

   (a) Si $K_1(x, z)$ y $K_2(x, z)$, demuestre que $K(x, z) = K_1(x, z) + K_2(x, z)$ es una función de Kernel. [40 %][**20 %**].

   (b) Demuestre que $K(x, z) = K_1(x, z)K_2(x, z)$ es una función de Kernel. [60 %] [**35 %**]

   (c) [**postgrado**] [**45 %**] Demuestre que el Kernel gaussiano

$$K(x, z) = e^{\frac{-||x-z||^2}{\sigma^2}},$$

   donde $\sigma^2 > 0$ es una constante positiva, es un kernel válido. [Ayuda: Considere que $||x - z||^2 = ||x||^2 - 2x^T z + ||z||^2$.]

5 Redes Neuronales

5. [**postgrado**] Considere una red neuronal convolucional, donde los pesos son restringidos para que tengan el mismo valor. Discuta como modificar el algoritmo back-propagation para estas restricciones puedan ser satisfechas al calcular las derivadas de la función de error respecto de los parámetros del modelo.

Soluciones

1. (a) No, ya que la función de costo es convexa, por lo tanto hay un sólo óptimo global.

   (b) No, el número de capas afecta a la magnitud del error que se traspasa en cada capa, generando posibles inestabilidades en el proceso de aprendizaje.

(c) No, mientras mayor sea el espacio de hipótesis, mayor es la tendencia a sobre ajustar, es decir, aprender demasiado bien el conjunto de entrenamiento, en desmedro de poder predecir bien datos no vistos en el entrenamiento.

(d) No necesariamente, una cosa es tiempo que demora cada iteración y otra es la velocidad de convergencia (que tan rápido me acerco al óptimo).

(e) No necesariamente, por ejemplo si combinamos clasificadores lineales la frontera de decisión será lineal, por lo que no podremos obtener error de entrenamiento cero si los datos no son linealmente separable.

(f) **[postgrado]** En la SVM la frontera de decisión está definida por los vectores de soporte, y éstos no cambia con la presencia de un punto lejos de frontera de decisión. En cambio, la frontera de decisión de la regresión logística está determinada por la función de verosimilitud que depende de todos los puntos de la muestra, por lo tanto, la frontera de decisión sufrirá una modificación.

(g) **[postgrado]** Infinito, ya que de esa manera forzamos a que las variables de holgura se aproximen a cero.

2. Aparecerá cuando terminen de entregar la Tarea 4.

3. El Lagrangiano del problema (2) está dado por

$$\mathcal{L}(\beta, \gamma) = ||Y - X\beta||^2 + \gamma(||\beta||^2 - s)$$

Y las condiciones KKT implican que

$$\begin{aligned} \nabla_\beta \mathcal{L}(\beta^*, \gamma^*) &= 0 && (3) \\ \gamma^*(||\beta^*||^2 - s) &= 0 && (4) \\ \gamma^* &\geq 0 && (5) \end{aligned}$$

Note que $\mathcal{L}(\beta, \gamma)$ es idéntico a a (2) expecto por el término $-\gamma s$ (que no depende de β). Si denotamos por $\beta(\lambda)^*$ a la solución del problema (1) para un λ fijo. Entonces $s = ||\beta(\lambda)^*||^2$ y $\gamma^* = \lambda$ satisfacerán las KKT.

4. (a) Sean G_1 y G_2 las matrices de kernel de los kernels K_1 y K_2 respectivamente. Tenemos que $G = G_1 + G_2$ la matriz de kernel de la función $K(x, z)$ Si $K(x, z)$ es una función de kernel válida, entonces G debe ser semi-definida positiva, esto es

$$\begin{aligned} \forall z, z^T G z &\geq 0 \\ z^T(G_1 + G_2)z &\geq 0 \\ z^T G_1 z + z^T G_2 z &\geq 0 \end{aligned}$$

Como G_1 y G_2 son matrices semi-definidas positivas, entonces ambos términos del lado izquierdo son positivos, por lo tanto, G también es semi-definida positiva.

(b) Sea $G_1 = \sum \mu_i m_i^T m_i$ y $G_2 = \sum \nu_i n_i^T n_i$. Entonces $G = G_1 \circ G_2$ es el producto de Hadamard (o elemento a elemento) entre G_1 y G_2:

$$G_1 \circ G_2 = \sum_{ij} \mu_i \nu_j (m_i^T m_i) \circ (n_j^T n_j) = \sum_{ij} \mu_i \nu_j (m_i \circ n_j)^T (m_i \circ n_j)$$

Cada matriz $(m_i \circ n_j)^T (m_i \circ n_j)$ es semi-definida positiva. Además, $\mu_i \nu_j > 0$ así la suma $G_1 \circ G_2$ también es semidefinida positiva.

(c) [**postgrado**] Primero que todo, debemos demostrar que si $K(x, z)$ es un Kernel válido, entonces $\exp(K(x, z))$ también lo es.
Usando la expansión de Taylor de la exponencial tenemos que:

$$\exp(K(x, z)) = \lim_{i \to \infty} \sum_{j=0}^{i} \frac{1}{j!} K(x, j)^j = \lim_{i \to \infty} K_i(x, z),$$

donde $K_i(x, z) = \sum_{j=0}^{i} \frac{1}{j!} K(x, z)^j$. K_i es un kernel válido ya que es la suma de productos de Kernels y escalares positivos. Si denotamos por K_i a la matriz del Kernel $K(x, z)$, tenemos que $z^T K_i z \geq 0$.
Analicemos ahora si $\exp(K) = \lim_{i \to \infty} K_i$ es semi-definida positiva, y por ende un Kernel válido:

$$\lim_{i \to \infty} z^T K_i z = \lim_{i \to \infty} z^T (K_i) z$$

Cada uno de los término es positivo ya que las K_is son matrices semi-definidas positivas. Además, sabemos que si una secuencia de números no negativos $a_i \geq 0$ tiene un límite $a = \lim_{i \to \infty} a_i \geq 0$. Por lo tanto,

$$\lim_{i \to \infty} z^T K_i z \geq 0.$$

Ahora tomamos en cuenta la ayuda para escribir

$$e^{\frac{-||x-z||^2}{\sigma^2}} = e^{\frac{-||x||^2}{\sigma^2}} e^{\frac{-||z||^2}{\sigma^2}} e^{\frac{2}{\sigma^2} x^T z}$$

Tenemos entonces que $e^{\frac{-||x-z||^2}{\sigma^2}} = f(x) f(z) K(x, z)$ donde los dos primeros términos son escalares positivos y el tercero es un Kernel válido, por lo tanto, el Kernel Gaussiano es un Kernel válido..

5. [**postgrado**] En la capa convolucional, los nodos se organizan en planos llamados *feature map*. Los nodos de cada feature map toman entradas de una pequeña región de la imágen y todas las unidades en el feature map deben tener los mismos pesos.

Por esa razón, el algoritmo back-propagation debe modificarse solamente en los pesos de los nodos de la capa convolucional.
Para modelar la restricción llamaremos $w^{(m)}$ al vector de pesos que comparten las unidades dentro del m-ésimo feature map. Así los errores $\delta^{(m)}$ de todos los nodos contribuirán a la derivada del vector de pesos. Por lo tanto,

$$\frac{\partial E_m}{\partial w_i^{(m)}} = \sum_j \frac{\partial E_m}{\partial a_j^{(m)}} \frac{\partial a_j^{(m)}}{\partial w_i^{(m)}} = \sum_j , \delta_j^{(m)} z_{ji}^{(m)}$$

donde, $a_j^{(m)}$ es la activación de la j-ésima neurona del m-ésimo feature map, $w_i^{(m)}$ es el i-ésimo elemento del vector $w^{(m)}$ y $z_{ji}^{(m)}$ es la entrada i-ésima del nodo j del m-ésimo feature map, ésta entrada puede ser ser una entrada o la salida de un nodo de una capa precedente.

Note que $\delta_j^{(m)} = \frac{\partial E_m}{\partial a_j^{(m)}}$ se calculará recursivamente desde los nodos de las capas siguientes. Si hay capas precediendo la capa convolucional, usaremos back-propagation estándar para ajustar sus pesos, considerando los pesos de la capa convolucional como parámetros independientes para computar los δs de las capas precedentes.