

# MODELOS Y OPTIMIZACIÓN PROCESOS GAUSSIANOS

## T. Gaussian Process Regresor GPR

Es un regresor basado en procesos gaussianos. Un proceso gaussiano se define completamente por su función de media  $m(x)$  y su kernel  $K(x, x')$

$$F(x) \sim \mathcal{G} \quad P(m(x), K(x, x'))$$

Donde:

$$m(x) = E[F(x)] \text{ es la media}$$

$$K(x, x') = E[(F(x) - m(x))(F(x') - m(x'))] \text{ es el kernel}$$

Para optimizar se debe maximizar el log marginal likelihood definido por:

$$\log P(y|G_n, \theta) = \mathcal{N}(y|0, K + \sigma_n^2 I) \quad (1)$$

Donde:

$y$  es el vector de observaciones

$m(x)$  es la función de media. Asumida como cero

$K$  es la matriz kernel

$\sigma_n^2$  es la varianza del ruido

$I$  es la matriz identidad

La maximización se logra minimizando el logaritmo negativo de (1)

$$\text{loss } \theta = -\log P(y|G_n, \theta)$$

El anterior logaritmo se puede calcular como:

Discrepancia entre datos observados y la medida

$$\log P(y|G_n, \theta) = -\frac{1}{2} y^T (K + G_n^{-2} I)^{-1} y$$

Penaliza la complejidad del modelo evitando

$$-\frac{1}{2} \log |K + G_n^{-2} I| \text{ sobreajuste}$$

$$-\frac{\alpha}{2} \log(2\pi) \text{ Regularización}$$

## 2.5 parse Gaussian Process Regression SGPR

Es una variación del GP diseñada para abordar problemas de escalabilidad. En lugar de utilizar todos los puntos de entrenamiento se utiliza un subconjunto + pequeño conocido como variables de inducción, ( $\tilde{z}$ )

Así entonces el GP se modela mediante

$$P(y|x, z, \theta) \sim \mathcal{N}(y|\tilde{m}, \tilde{K}) \quad (2)$$

Donde:

$\tilde{m}$  es la media aproximada

$\tilde{K}$  es el kernel aproximado calculado en función de las variables de inducción

$$\tilde{K}(x, x) = K(x, z) [K(z, z) + G_n^{-2} I]^{-1} K(z, x) + G_n^{-2} I$$

Donde:

$K(x, z)$  es la matriz de covarianza entre los puntos de entrenamiento y las variables de inducción

$K(z, z)$  es la matriz kernel entre los puntos de inducción

La optimización consistiría en maximizar la ecuación (2) pero en función de las variables de inducción

### 3. Variational Gaussian Process VGP

Los modelos GPR y SPGR necesitan que la probabilidad sea gaussiana pero los VGP permiten configurar la probabilidad.

Para ello se busca una aproximación variacional  $q(F)$  que minimice la divergencia KL entre la distribución posterior exacta y la aproximada

$$q(F) = \mathcal{N}(F|m, \Sigma)$$

Donde:

$m$  es la media

$\Sigma$  es la matriz de covarianza de los puntos inducidos

Para la optimización se debe maximizar el ELBO dado por:

$$\text{ELBO} = \underbrace{\mathbb{E}_{q(F)} [\log p(y|F)]}_{\text{log verosimilitud esperado}} - \underbrace{\text{KL}(q(F) || p(F))}_{\substack{\text{Divergencia KL} \\ \text{penaliza grandes diferencias}}}$$

El log verosimilitud esperado mide la capacidad del modelo para ajustar los datos

Con  $\log(p(y|F)) = \frac{1}{2\sigma^2} \sum_{i=1}^N (y_i - F(x_i))^2$

la evaluación se hace bajo la distribución aproximada

$$-\frac{N}{2} \log(2\pi\sigma^2)$$

## 4. Sparse Variational Gaussian Process (SVGP)

Este modelo utiliza una distribución variacional para aproximar la verdadera distribución posterior y emplear un conjunto de pseudopuntos para reducir la cantidad de datos necesarios para calcular el proceso.

$$F(x) = \int P(0, K(x, x')) \text{ (original)}$$

$Z = \{z_1, z_2, \dots, z_m\}$  donde  $M \ll N$  pseudopuntos con valores de la función correspondientes a estos pseudopuntos  $U = \{u_1, u_2, \dots, u_m\}$

$$q(u) = \mathcal{N}(u|m, \Sigma) \text{ Distribución variacional}$$

El proceso  $F(x)$  sigue una distribución condicional gaussiana

$$\begin{aligned} P(F(x) | u) &= \mathcal{N}(F(x) | K(x, z) K_{zz}^{-1} u, K(x, x)) \\ &\quad - K(x, z) K_{zz}^{-1} K(z, x) \end{aligned}$$

Donde

$K(x, z)$  es la covariancia entre los puntos  $x$  y  $z$ .

$K_{zz}$  es la covarianza entre los pseudopuntos

Para la optimización, el objetivo es ajustar la distribución variacional  $q(u)$  de los pseudopuntos a los datos observados  $y$ .

$$\text{SIEBO} = \sum_{i=1}^N \underbrace{E_{q(F(x_i))} [\log P(y_i | F(x_i))]}_{\text{Depende de los pseudopuntos inducidos}} - \text{KL}(q(u) || P(u))$$

Depende de los pseudopuntos inducidos

## 5. Gaussian Process Classification

Modela la probabilidad de pertenencia de un a clase como una función no lineal de los datos de entrada. Se utiliza una distribución gaussiana para representar la incertidumbre en las predicciones.

Sea  $X = \{x_1, x_2, \dots, x_N\}$  los datos de entrada y

$y = \{y_1, y_2, \dots, y_N\}$  las etiquetas

En lugar de modelar directamente  $y_i$ , se introduce una función latente  $F(x)$  que se transforma mediante una función de enlace para obtener probabilidades de clase. La función de enlace más utilizada es la sigmoidal

$$P(y_i=1 | x_i, F(x_i)) = \sigma(F(x_i)) = \frac{1}{1 + e^{-F(x_i)}}$$

Representa la probabilidad de que la clase sea 1 dado el valor de la función latente  $F(x_i)$

La predicción para una nueva entrada  $x$  no se realiza directamente sobre la función latente sino sobre la probabilidad posterior de pertenecer a la clase 1

$$P(y=1 | x_i, X, y) = \int \sigma(F(x_i)) P(F(x_i) | X, y, x_i) dF$$

La integral anterior no tiene solución analítica así que se puede aproximar mediante variational inference por ejemplo

Para la optimización se debe maximizar el log verosimilitud dado por

$$\log P(y|x) = \sum_{i=1}^N P(y_i|f(x_i))P(f(x_i)|x)$$

Dicha integral también debe aproximarse.

## MÉTODOS DE OPTIMIZACIÓN con SCIPY Y TENSOR FLOW

### 1. Broyden Fletcher Goldfarb Shanno BFGS

Utiliza una aproximación de la matriz Hessiana (segundas derivadas) para actualizar parámetros.

$$x_{k+1} = x_k - d_k H_k^{-1} \nabla f(x_k)$$

Donde

$x_k$  son los parámetros actuales

$d_k$  es la tasa de aprendizaje

$H_k^{-1}$  es la inversa de la aproximación de la matriz Hessiana

$\nabla f(x_k)$  es el gradiente de la función objetivo en  $x_k$

Ventaja Eficiente en problemas de optimización no lineal

Desventaja No es adecuado para grandes conjuntos de datos

### 2. L-BFGS-B

Es una variante de BFGS que impone límites superior e inferior.

$$x_{k+1} = \text{Proy}_{\beta} (x_k - d_k H_k^{-1} \nabla f(x_k))$$

Es la proyección del nuevo valor  $x_{k+1}$  sobre el conjunto de restricciones  $\beta$

Ventaja Es más eficiente que BFGS porque no almacena la matriz Hessiana completa

Desventaja No siempre converge de manera tan rápida

### 3. Conjugate Gradient CG

Se utiliza para minimizar funciones cuadráticas sin necesidad de calcular la matriz Hessiana

$$x_{n+1} = x_n + \alpha_n p_n$$

Donde:

$$p_n = -\nabla F(x_n) + \beta_n p_{n-1}$$

es la dirección conjugada  
coeficiente de  
conjugación

Ventaja es útil en problemas de gran escala

Desventaja la convergencia depende altamente de la elección de los coeficientes

Ahora con Tensor Flow tenemos:

### 7. Stochastic Gradient descent SGD

Actualiza los parámetros en función de los gradientes calculados para pequeños subconjuntos

$$x_{n+1} = x_n - \eta \nabla_x f(x_n)$$

Donde

$x_n$  son los parámetros actuales

η es la taza de aprendizaje

$\nabla f(x_n)$  es el gradiente de la función objetivo

Ventaja Maneja grandes conjuntos de datos

Desventaja Es propenso a quedar atrapado en mínimos locales.

## 2. Adaptive Moment Estimation Adam

Adam es un método que adapta la taza de aprendizaje en lugar de mantenerla fija como el gradiente

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla F(x_t)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \nabla^2 F(x_t)^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$x_{t+1} = x_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Donde

$m_t$  y  $v_t$  son estimaciones de los momentos de primer y segundo orden

$\beta_1$  y  $\beta_2$  son coeficientes de decaimiento

$\epsilon$  es un término de estabilidad numérica

$\eta$  es la taza de aprendizaje

Ventajas No requiere de mucha sintonización de parámetros.

Desventaja Podrá encontrar soluciones subóptimas en lugar de la global

### 3. RMS prop

Ajusta dinámicamente la taza de aprendizaje usando el promedio móvil del cuadrado de los gradientes

$$V_t = \gamma V_{t-1} + (1-\gamma) \nabla \times f(x_t)^2$$

$$x_{t+1} = x_t - \frac{\eta}{\sqrt{V_t + \epsilon}} \nabla \times f(x_t)$$

Donde

$V_t$  es el promedio móvil del cuadrado de los gradientes

$\gamma$  es el factor de decaimiento

$\eta$  es la taza de aprendizaje

Ventaja adecuado para problemas con gradientes ruidosos y no estacionarios.

Desventaja la taza de aprendizaje inicial puede ser difícil de ajustar.

Característica : Scipy . . . TensorFlow

Uso Optimización matemática general Aprendizaje automático y redes neuronales

Gradientes Explícitos o proximados Cálculo automático

Eficiencia Problemas pequeños Grandes modelos

Taza aprendizaje Fija o calculada Adaptativa