HMM- State Decoding

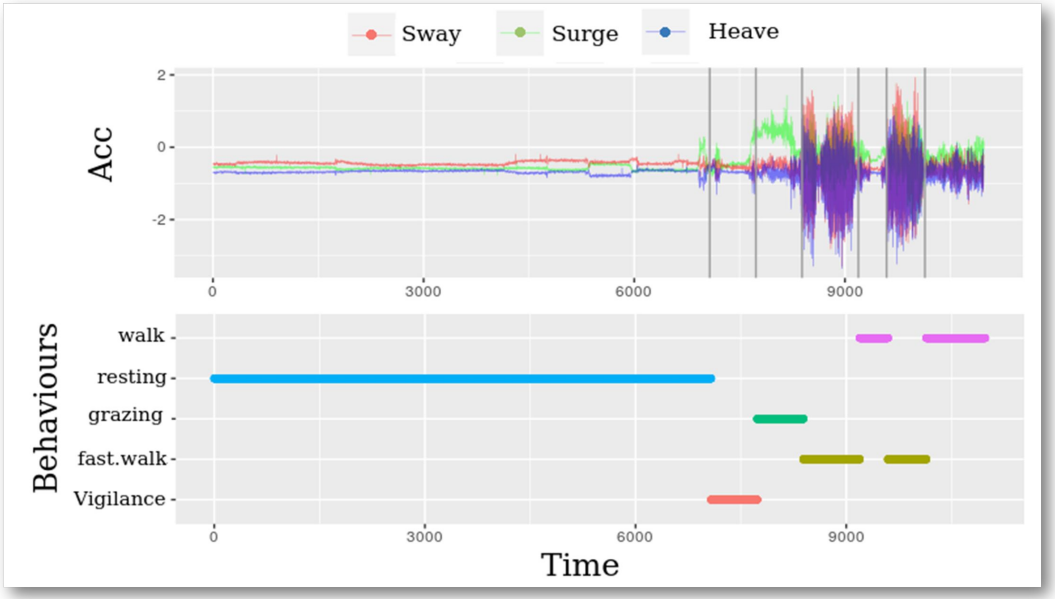
RIIAA

MEETING ON ARTIFICIAL INTELLIGENCE AND ITS APPLICATIONS

25-27 August 2021

Clasificación de series temporales

Objetivo: asignar a diferentes segmentos de los datos a un estado



- Reconocimiento de voz
- Identificación de comportamientos por datos de aceleración
- Análisis de electrocardiogramas

Clasificación

Enfoques

- Supervisado → Hay datos para los cuales conocemos sus estados reales → Asignar a una clase predefinida
- No Supervisado → No conocemos los estados reales de los datos → Identificar patrones en los datos

¿Por qué son útiles los HMM?

Permiten capturar la dependencia temporal de los datos dentro de la estructura del modelo

¿Qué es *State Decoding*?

Es el proceso de determinar los estados más probables que generaron las observaciones.

Dos formas → **Local Decoding.** Identificar el estado más probable para cada punto en el tiempo

→ **Global Decoding.** Identificar la secuencia de estados más probable

Local Decoding- *Forward Backward Algorithm*

$$\hat{C}_t = \underset{j=1,\ldots,J}{\operatorname{argmax}}\, P(C_t = j | \boldsymbol{X}_{1:T})$$

Se computa mediante el cálculo de las *probabilidades Forward y Backward*

Recursiones

$$\alpha_t(j) = P(\boldsymbol{X}_{1:t}, C_t = j) \qquad \alpha_t(j) = f_j(x_t) \sum_i \gamma_{i,j} \alpha_{t-1}(i)$$

$$\beta_t(j) = P(\boldsymbol{X}_{t+1:T} | C_t = j) \qquad \beta_t(j) = \sum_i \beta_{t+1}(i) \gamma_{i,j} f_i(x_{t+1})$$

$$P(C_t = j | \boldsymbol{X}_{1:T}) = \frac{\alpha_t(j)\beta_t(j)}{\Pr(\boldsymbol{X}_{1:T})}$$

Global Decoding- *Viterbi Algorithm*

$$\hat{C}_{1:T} = \max_{c_1, \ldots, c_T} P(\boldsymbol{C}_{1:T}, \boldsymbol{X}_{1:T})$$

No calculamos las j^T posibilidades!!

Definimos las probabilidades

$$\xi_{1i} = P(C_1 = i, X_1) = \delta_i f_i(x_1)$$

Para $t = 2, 3, \ldots, T$

$$\xi_{ti} = \max_{c_1, \ldots, c_{t-1}} P(C_{1:t-1}, C_t = i, X_{1:t})$$

Se puede probar que estas probabilidades siguen una recursión

Global Decoding- *Viterbi Algorithm*

Recursión

$$\xi_{tj} = (\max_i (\xi_{t-1,i}\gamma_{ij}))f_j(x_t)$$

Tenemos así una forma eficiente de calcular la matriz de TxJ elementos con orden lineal en T !

Luego **la secuencia de estados más probable** puede ser determinada de manera recursiva como

$$i_T = \underset{i=1,\ldots,J}{\operatorname{argmax}}\, \xi_{Ti}$$

$$i_t = \underset{i=1,\ldots,J}{\operatorname{argmax}}\, \xi_{ti}(\gamma_{i,i_{t+1}}) \quad t = T-1,\ldots,1$$

Cómo evaluamos el error de clasificación?

(caso supervisado)

Medidas

Para calcular la precisión de la clasificación

$$AC = \frac{1}{T} \sum_{t=1}^{T} I(\hat{c}_t = c_t)$$

Para calcular la incertidumbre asociada a la clasificación

$$CE = -\sum_{t=1}^{T} I(\hat{c}_t = c_t) \log\left(P(x_t|c_t)\right)$$

Estimación del error de clasificación

Técnicas de validación cruzada

CUIDADO!

Cuando se trata de analizar datos temporales **hay que considerar la correlación temporal cuando se dividen los conjuntos de entrenamiento y testeo**, sino las estimaciones estarán sesgadas y sobre estimará el error de clasificación

- Leave-one-time series out
- h-block-cross validation

Consideraciones a la hora de hacer clasificación

Si el objetivo final es hacer clasificación, no nos importa demasiado que el modelo utilizado esté correctamente especificado.

Lo que **nos importa es que las predicciones y sus incertidumbres asociadas sean lo más exactas posibles**

Si modelos más sencillos predicen igual que modelos más complejos pero que están mejor especificados preferimos el más sencillo

Ahora si lo que queremos poder simular datos hay que considerar la buena especificación de los modelos

Implementación

Vamos al código de R!