Predicción de la estructura secundaria de proteínas globulares con Artificial Neural Networks (ANN) y Support Vector Machines (SVM)

Introducción

El problema general de predecir la estructura terciaria de las proteínas plegadas no está todavia resuelto. No obstante, la información sobre la estructura secundaria de una proteína puede ser útil para determinar sus propiedades estructurales. La mejor manera de predecir la estructura secundaria de una nueva proteína es encontrar una proteína homóloga cuya estructura ha sido determinada previamente. Si no se conocen proteínas homólogas con estructura conocida, existen métodos de `machine learning` que se pueden usar para predecir estructuras secundarias.

El objetivo de esta actividad es utilizar la información disponible en una base de datos de estructura secundaria de proteínas para evaluar la capacidad de predecir la estructura secundaria de proteínas para las que no hay homólogos conocidos.

Estos métodos explotan principalmente, de diferentes maneras, las correlaciones entre aminoácidos y la estructura secundaria local. Por local, nos referimos a una influencia en la estructura secundaria de un aminoácido por otros que no están más que a unos diez residuos de distancia.

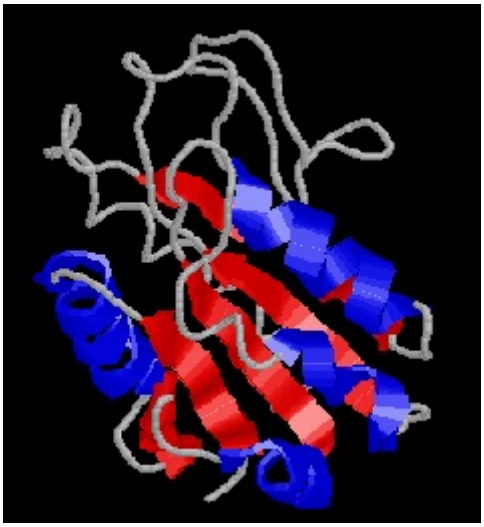

Figure 1: Proteína globular en la que se resalta la estructura secundaria, en azul los segmentos α-helix, en rojo los β-sheet y en gris los coil.

La base de datos de proteínas con estructura secundaria conocida se obtuvo del Laboratorio Nacional Brookhaven (USA), de la cual se han seleccionado una muestra representativa de 101 proteínas (ver Referencias).

Formato original de la base de datos

En el formato de la base de datos las proteínas se codifican mediante una única secuencia (primera columna del dataset) indicando el inicio de cada proteína con el carácter `<>` y el final mediante el carácter `end`. En la segunda columna del dataset se indica para cada aminoácido su participación en segmentos correspondientes a estructuras secundarias del tipo α-helix, β-sheet o coil, indicándolo con los caracteres `h`, `e` o `-`, respectivamente.

	aa	class
1	<>	
2	G	_
3	V	_
4	G	_
5	T	_
6	V	_
7	P	_
8	M	_
9	T	_
10	D	_
11	Y	_
12	G	_
13	N	_
14	D	_
15	V	_
16	E	_
17	Y	_
18	Y	_
19	G	_
20	Q	_
21	V	e
22	T	e
23	I	_
24	G	_
25	T	_

Formato adaptado a una ventana

Para tener en cuenta la estructura local, la información de entrada de los algoritmos que se van a aplicar se definirá por medio de una "ventana". El objectivo será predecir la clase del aminoacido en la posición central de la ventana. Se ha escogido una ventana con 17 posiciones. La nueva base de datos se forma con secuencias que resultan de desplazar la ventana un aminoácido a la vez a través de las proteínas en la base de datos original.

Para facilitar la realización de la actividad se proporciona un fichero con secuencias de 17 aminoácidos obtenidas a partir de las proteínas descargadas de la base de datos y la clase (`h`,`e`,`-`) del aminoácido en la posición central (posición 9) de cada secuencia.

No obstante, hace falta remarcar que se deberá emplear una codificación `one-hot` de los aminoácidos de las secuencias. Por tanto, el vector de entrada (input) de los métodos tendrá $17 \cdot 20 = 340$ componentes.

Ejemplo de los 6 primeros registros

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18
1	G	V	G	T	V	P	M	T	D	Y	G	N	D	V	E	Y	Y	_
2	V	G	T	V	P	M	T	D	Y	G	N	D	V	E	Y	Y	G	_
3	G	T	V	P	M	T	D	Y	G	N	D	V	E	Y	Y	G	Q	_
4	T	V	P	M	T	D	Y	G	N	D	V	E	Y	Y	G	Q	V	_

```
5 V P M T D Y G N D V E Y Y G Q V T _
6 P M T D Y G N D V E Y Y G Q V T I _
```

y ejemplo de la codificación one-hot del primer registro en la lista anterior:

```
  [1] 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [38] 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [75] 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
[112] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[149] 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[186] 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[223] 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[260] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0
[297] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[334] 0 0 0 0 0 1 0
```

Enunciado

1. Escribir en el informe dos secciones con los títulos: "Algoritmo Red Neuronal Artificial" y "Algoritmo Support Vector Machine" en el que se haga una breve explicación de su funcionamiento y sus características. Además, se presente una tabla de sus fortaleza y debilidades para cada algoritmo.
2. Desarrollar una función en R que implemente la codificación "one-hot" (*one-hot encoding*) de las secuencias.
3. Desarrollar un código en R que implemente un clasificador de red neuronal artificial. El código en R debe:
   (a) Leer el fichero `data4.csv` donde cada registro contiene una secuencia de 17 aminoácidos y la clase de estructura secundaria correspondiente al aminoácido central (posición 9), donde los caracteres ‘h‘, ‘e‘ y ‘-‘ representan α-helix, β-sheet y coil, respectivamente.
   (b) Utilizando la semilla aleatoria `12345`, separar los datos en dos partes, una parte para training (67%) y una parte para test (33%).
   (c) Antes de ejecutar cada uno de los modelos de clasificacion que se piden a continuación, poner como semilla generadora el valor `1234567`.
   (d) Crear dos modelos de red neuronal artificial de una sola capa oculta con 10 nodos y 40 nodos, respectivamente. Aplicar los datos de training para ajustar los modelos y posteriormente, predecir el tipo de estructura secundaria en los datos del test.
   (e) Comentar los resultados de la clasificación en función de los valores generales de la clasificación como "accuracy" y otros. Comparar los resultados de clasificación obtenidos para los diferentes valores de nodos usados en la capa oculta.
   (f) Usar el paquete caret modelo ‘mlp‘ para implementar la arquitectura de 40 nodos en la capa oculta, usando 5-fold crossvalidation. Repetir el análisis balanceando las tres clases, ‘h‘, ‘e‘ y ‘-‘, a 1000 observaciones por clase y usar la semilla `12345` para el muestreo. Comentar los resultados.
4. Desarrollar un código en R que implemente un clasificador de SVM. El código en R debe:
   (a) Leer el fichero `data4.csv` donde cada registro contiene una secuencia de 17 aminoácidos y su clase de estructura secundaria del aminoácido central (posición 9), donde los caracteres ‘h‘, ‘e‘ y ‘-‘ representan α-helix, β-sheet y coil, respectivamente.
   (b) Utilizando la semilla aleatoria `12345`, separar los datos en dos partes, una parte para training (67%) y una parte para test (33%).
   (c) Antes de ejecutar cada uno de los modelos de clasificación que se piden a continuación, poner como semilla generadora el valor `1234567`.

(d) Utilizar la función lineal y la RBF para ajustar un modelo de SVM basado en el training para predecir el tipo de estructura secundaria en los datos del test.

(e) Comentar los resultados de la clasificación en función de los valores generales de la clasificación como "accuracy" y otros. Comparar los resultados de clasificación obtenidos para los diferentes funciones kernel usadas.

(f) Usar el paquete caret modelo svmRBF para aplicar el algoritmo de SVM con 5-fold crossvalidation. Repetir el análisis balanceando las tres clases, 'h', 'e' y '-', a 1000 observaciones por clase y usar la semilla `12345` para el muestreo. Comentar los resultados.

5. Comentar todos los resultados obtenidos y escoger que modelo puede ser el mejor.

NOTA

Tener en cuenta que el **coste computacional** de los algoritmos para el análisis de estos datos puede requerir **varios minutos** para su finalización dependiendo de las prestaciones de hardware de tu ordenador.

Informe de la PEC

Las soluciones se presentarán mediante un informe dinámico R markdown con la estructura habitual de los ejercicios no evaluables realizados hasta ahora. En primer lugar, el informe tendra un titulo (igual que el de la PEC), el autor, la fecha de creación y el indice de apartados de la PEC. En segundo lugar, se crea una sección con el título "Algoritmo Red Neuronal Artificial" donde se haga una breve explicación de su funcionamiento y sus características. Además, se presenta la tabla de sus fortaleza y debilidades. En tercer lugar, se crea la sección "Algoritmo Support Vector Machine" similar a la anterior sección. En cuarto lugar se realizan los diferentes apartados de la PEC pero con la estructura de Step1 hasta Step5 para cada tipo de algoritmo. Al final se crea una sección "Discusión final" para comentar todos los resultados obtenidos y escoger el mejor modelo.

Un característica que se valorará es hasta que punto es el informe "dinámico". En el sentido de adaptarse el informe a cambios en los datos.

Se subiran al registro de entregas un **zip** con los siguientes ficheros:

1. Fichero ejecutable (.Rmd) que incluya un texto explicativo que detalle los pasos implementados en el script y el código de los análisis. No olvidar de incluir todos los ficheros complementarios que hagan falta para la correcta ejecución: *ficheros de datos, fichero de bibliografia, imagenes, ...* NOTA: Para facilitar la ejecución, no usar un ruta fija para la lectura del fichero, asociarlo al area de trabajo donde este el fichero .Rmd.
2. Informe (pdf) resultado de la ejecución del fichero Rmd anterior.

Antes de enviar el zip, se recomienda **verificar la reproducibilidad del fichero .Rmd** para obtener el informe en formato pdf sin ninguna dificultad.

Puntuacions de los apartados

Apartado 1 y Apartado 2 (5%), Apartado 3 (40%), Apartado 4 (40%), Apartado 4 (5%), Calidad del informe dinámico (10%).

Referencias

Molecular Biology (Protein Secondary Structure) Data Set del repositorio:

https://archive.ics.uci.edu/ml/datasets/Molecular+Biology+(Protein+Secondary+Structure)

Ning Qian and Terrnece J. Sejnowski (1988), "Predicting the Secondary Structure of Globular Proteins Using Neural Network Models" in Journal of Molecular Biology 202, 865-884. Academic Press.