Predicción de la estructura secundaria de proteínas globulares

Machine Learning

Introducción

El problema general de predecir la estructura terciaria de las proteínas plegadas no está todavía 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.

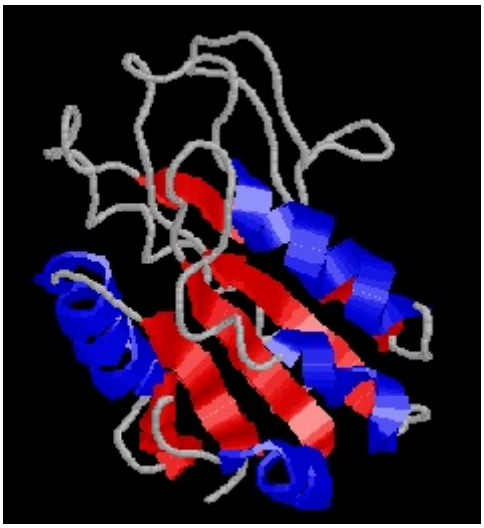

Figure 1: Proteína globular en la que se resalta la estructura secundaria, en azul los segmentos α-helix, en rojo los β-sheet y en gris los coil.

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 objetivo será predecir la clase del aminoácido 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 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 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 0 1 0 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 0 1 0 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 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 0 0
[223] 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 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 0 0 0 1 0 0 0 0 0 0 0 1 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 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[334] 0 0 0 0 0 1 0
```

usando el ordén alfabético del código de tres letras de los aminoácidos (ver wikipedia https://es.wikipedia.org/wiki/Nomenclatura_de_amino%C3%A1cidos)

En la PEC se implementará un algoritmo `knn` para predecir la clase de estructura secundaria correspondiente al aminoácido central (posición 9) de la secuencia de 17 aminoácidos que lee. Las clases de estructuras secundarias son α-helix, β-sheet y coil y se representan con los caracteres `h`, `e` y `_`, respectivamente.

Enunciado

1. Escribir en el informe una sección con el título "Algoritmo k-NN" en el que se haga una breve explicación de su funcionamiento y sus características. Además, se presente una tabla de sus fortalezas y debilidades.
2. Desarrollar una función en R (o Python) que implemente una codificación "one-hot" (*one-hot encoding*) de las secuencias.
3. Desarrollar un script en R (o Python) que implemente un clasificador `knn`. El script debe realizar los siguientes apartados:
   (a) Leer el fichero `data4.csv`. 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. Después de cargar los datos, crear una tabla donde se muestre el número de secuencias de cada clase.
   (b) Utilizar la función de codificación "one-hot" para representar las secuencias **NOTA:** En caso de no poder hacer la función, se puede descargar el fichero `oh_enc.csv` con las secuencias ya transformados.
   (c) Utilizando la semilla aleatoria `123`, separar los datos en dos partes, una parte para training (67%) y una parte para test (33%).
   (d) Utilizar un knn (k = 1, 3, 5, 7, 11) basado en el training para predecir la estructura secundaria de las secuencias del test.
   (e) Por otra parte, sabemos que las clases α-helix y β-sheet son del tipo non-coil. Realizar otro knn (k = 1, 3, 5, 7, 11) para esta nueva clasificación, coil y non-coil. Además, realizar una curva ROC para cada k y calcular su área bajo la curva (AUC).
   (f) Comentar los resultados de la clasificación para las tres clases de estructuras secundarias basado, como mínimo, en el error de clasificación y el valor de kappa. Además, comentar los resultados para las clases coil y non-coil en función del AUC, número de falsos positivos, falsos negativos y error de clasificación obtenidos para los diferentes valores de k. La clase que será asignada como positiva es la `non-coil`. Finalmente, comentar globalmente los resultados de ambas clasificaciones.

Informe de la PEC

El informe se presentará mediante un informe dinámico R markdown (o notebook Python) con la estructura habitual de los ejercicios no evaluables realizados hasta ahora. En primer lugar, el informe tendrá un título (igual que el de la PEC), el autor, la fecha de creación y el índice de apartados de la PEC. En segundo lugar,

se crea una sección con el título "Algoritmo k-NN" 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 realizan los diferentes apartados de la PEC.

Una característica que se valorará es hasta que punto el informe es "dinámico". En el sentido de adaptarse el informe a cambios en los datos. Además de la calidad del código, formato y estructura del documento, concisión y precisión en las respuestas.

Se entregarán dos ficheros:

1. Fichero ejecutable R (.Rmd) o notebook Python (.ipynb) que incluya un texto explicativo que detalle los pasos implementados en el script y el código de los análisis.
2. Informe (pdf) resultado de la ejecución del fichero Rmd o ipynb anterior.

En resumen, **se puede entregar la PEC programando en R o Python, según vuestra conveniencia**.

Puntuaciones de los apartados

Apartado 1 (5%), Apartado 2 (25%), Apartado 3 (60%), Calidad del informe dinámico (calidad del código, formato y estructura del documento, concisión y precisión en las respuestas) (10%).

Referencias

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

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