Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

Tarea 2: Minería de Datos IIC2433

Fecha de entrega : 3 de noviembre de 2019, 23:59 hrs.

Introducción

Random Forest es uno de los algoritmos de aprendizaje supervisado más conocidos en Machine Learning [Breiman, 2001, Ho, 1995]. Lo que encontramos en este algoritmo, en palabras simple, es una colección de árboles no correlacionados, que luego son promediados. Con este algoritmo podemos realizar tareas de clasificación, cada árbol de clasificación retorna categorías/clases y luego por votación obtenemos la clasificación. En esta tarea serán guiados para que implementen su propio Random Forest y posteriormente la prueben sobre una base de datos real.

Random Forest

Como fue visto en clases, un árbol es una estructura que divide el espacio de variables clasificadoras en secciones y asigna un valor a la predicción a cada una, como se ilustra en la siguiente figura:

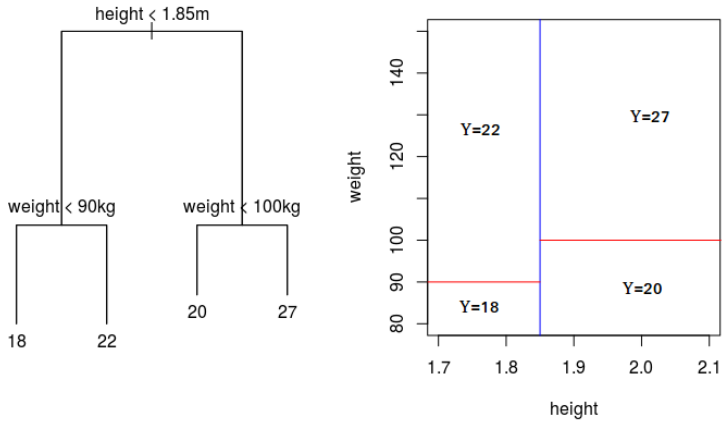

Fig 1. Puntaje basquetbolista según altura y peso

Para la construcción del árbol, por cada nodo se debe decidir la variable de división según una métrica que indique la "homogeneidad" de los hijos resultantes de ella. Una con la cual ustedes ya se encuentran familiarizados es según "Ganancia de Información" que emplea el concepto de Entropía [Shannon, 1948]. El árbol para clasificación retorna la clase más común una vez alcanzada una hoja.
En el caso de Random Forest tenemos n árboles de decisión, cada uno se entrena por separado, y al clasificar un nuevo dato, cada uno de estos árboles entrega una clasificación. Por mayoría de "votos" se decide cuál es la predicción.

Pasos a seguir

Para completar la tarea ustedes deben seguir los siguientes pasos:

1. **Preprocesar** la base de datos. Esta consiste en una base de datos astronómica *FATS_GAIA.dat*, que se encuentra adjunta al enunciado. Contiene *features* de series de tiempo astronómicas del *survey* GAIA [1]. El **target** (variable a predecir) corresponde a la columna *Class* y **data** (variables predictoras) a todas las otras columnas. Deben dividir esta base de datos en un set **train** (%80) y otro **test** (%20).

2. **Implementar** el algoritmo Random Forest desde cero. Para la implementación sólo podrá utilizar las librerías Numpy y Pandas. En este punto el código debe contar con funciones *fit* y *predict*:

   - *fit*: Debe aplicar el algoritmo a la base de datos. Debe recibir **data**, **target**, opcionalmente permitir definir el número de árboles *n_estimators* la profundidad máxima *max_depth* (para evitar overfitting), y el número mínimo de muestras requeridos para el split *min_samples_split*. Considere que las variables con que su algoritmo deberá trabajar son continuas, esto implica implementar una forma automática de encontrar el punto de split de cada nodo.
   - *predict*: Debe recibir **data** y retornar la predicción.

3. **Aplicar** el algoritmo a la base de datos de prueba. Entrene con **data_train** y haga predicciones sobre **data_test**. Debe entregar una medida de qué tan buena es su predicción sobre **data_test**.

4. **Visualizar** un árbol. En este paso puede usar *matplotlib*, *graphviz*, ó alguna librearía de visualización que le acomode.

5. **Explicar** y analizar los resultados obtenidos (puede apoyar su explicación en la visualización), comentar los experimentos realizados ¿Cómo espera que cambie la predicción al setear distintas cantidades de árboles y distintos niveles de profundidad del árbol?

1 Entrega

- La entrega debe ser realizada en un .zip con todos los archivos necesarios.

[1] http://sci.esa.int/gaia/

- El archivo de entrega debe ser subido al cuestionario abierto en el sistema SIDING específicamente para esta tarea hasta el día y hora señalada con el nombre $[numero\_alumno]\_T2$, tanto el .zip como el .ipynb.
- En caso de atraso, se aplicará un descuento lineal de nota 7 a 1 en 24 horas.
- La tarea es estrictamente individual y el algoritmo debe ser implementado 100% (no usar funciones previamente implementadas o re-utilizar código).
- El documento principal debe ser un jupyter notebook con el código.
- Cualquier instrucción adicional y necesaria para la revisión debe ser escrita en un archivo README.txt contenido en el .zip

References

L. Breiman. Random forests. *Machine Learning 45 (1)*, 1:5–32, 2001.

T. K. Ho. Random decision forest. *Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995*, pages 278–282, 1995.

C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 1948.