Ensamble Secuencial de Modelos de Aprendizaje Automático Supervisado

Introducción:

En este trabajo se da una posible solución al problema de crear un ensamble de modelos de aprendizaje automático supervisado. Para ello, algunos conceptos básicos que convendría repasar son:

* Aprendizaje supervisado: consiste en algoritmos que necesitan la variable objetivo para entrenar y predecir sus valores, con lo que podemos medirlos y ver cómo de buenos son. Por tanto, son contrarios al aprendizaje por refuerzo, en el que no se sabe la solución antes de empezar.
* Durante el curso se ha trabajado con Naive Bayes, KNN, árboles de decisión y redes neuronales. Todos ellos han sido probados en este trabajo, además de algunos otros, con intención de entender porqué o no son buenos para ser ensamblados.

Para ello, hemos utilizado una técnica similar a GradientBoosting (potenciación del gradiente) descrita en el pseudocódigo del documento introductorio, evaluando los modelos con evaluación cruzada, como se pedía en la propuesta.

Estructura y Decisiones de Diseño:

El notebook comienza con todos los imports, seguido de celdas de documentación. La celda 5 tiene la implementación del meta-modelo, todo lo que necesita para funcionar.

**Decisiones de diseño:**

* El formato de los datos es csv y se almacena (en memoria) en un objeto de tipo DataFrame de Pandas.
* El meta-modelo puede aceptar tanto tareas de regresión como de clasificación.
* El meta-modelo acepta o no normalización.
* Se inicializa la predicción inicial como la media de la variable objetivo, ya que eso garantiza un mejor resultado en menos iteraciones que inicializarla a 0 en el mayor de los casos.
* Se usa validación cruzada para la evaluación del ensamble.
* Cada modelo entrena con datos aleatorios.
* La métrica usada tanto para la parada temprana como para la validación cruzada es el coeficiente de determinación, *R*2.
* Para la parada temprana (early stopping) hemos definido un valor umbral para considerar si la mejora de la predicción del modelo actual respecto del anterior es útil, en vez de comprobar que no haya mejora, ya que, si no, con los pocos datos que tenemos y las iteraciones que este trabajo exige no se aplicaría nunca. De hecho, tenemos que trabajar con valores bastante elevados del umbral y con una paciencia baja para que pare. Esta evaluación se realiza con los datos de prueba no utilizados en el entrenamiento de cada modelo.
* El ensamble soporta regresores de redes neuronales.
* El ensamble soporta tareas de regresión y de clasificación?
* El ensamble hereda de BaseEstimator y RegressorMixin para poder utilizar diversos métodos de scikit learn que se salen del alcance de este trabajo.

Experimentación:

Para la experimentación se han usado los csv originales para las tareas de regresión y los mismos pero cambiando el valor de la variable objetivo a binaria. Se han utilizado los regresores DecisionTreeRegressor, el obligatorio, para el que se han obtenido los siguientes valores experimentando con su hipérparámetro max\_depth y los hiperparámetros n\_estimators, learning rate y sample\_size, tal como se exige en el trabajo:

|  |  |  |  |
| --- | --- | --- | --- |
| n\_estimators | Learning rate | Sample size | max\_depth |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Conclusiones:

Podemos observar que

Autodeclaración del uso de IA generativa:

En este proyecto la IA se ha usado exclusivamente para corrección de errores en tiempo de ejecución y para la ayuda puntual de alguna interpretación o la mejora del código existente (más legible y/o optimizado)

Bibliografía:

[ChatGPT](https://chatgpt.com/)

[RegressorMixin — scikit-learn 1.6.1 documentation](https://scikit-learn.org/stable/modules/generated/sklearn.base.RegressorMixin.html)

[cross\_val\_score — scikit-learn 1.6.1 documentation](https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html)