**Proyecto INFO335**

Utilización de librerías “Ray Tune” para mejorar rendimiento de modelo “Bart”, en estimaciones de la Retención Estudiantil en la Facultad de Ingeniería UACh.

Estudiante: Valeria Soto Castro

Docente: Pablo Huijse

Julio 2023, Valdivia.

**Introducción**

Las tasas de matrículas de educación superior, han aumentado constantemente en los últimos años en Chile. Sin embargo, junto a ello, se presenta una preocupación por las tasas de deserción estudiantil que se presentan año a año, según métricas deL SIES. Es por esto, que las tasas de abandono de programas a nivel de educación superior, son una constante preocupación y factor crítico, por parte de las Universidades.

Persiste una búsqueda de factores de riesgo e intentos de probar posibles causas, sin lograr determinar variables predominantes, solo pruebas de buenos modelos de predicción. Es por esto la necesidad de identificar aquellos factores relevantes.

Basado en este problema, es la motivación de situarnos en el contexto de la Facultad de Ingeniería de la Universidad Austral, específicamente evaluando data de estudiantes de la carrera de Ingeniería Civil Informática, logrando analizar las características de estudiantes retenidos y no retenidos tras su primer año del programa.

En este trabajo, se presenta el desarrollo y evaluación de técnicas y herramientas que nos ayuden a predecir a través de modelos, específicamente de “Bayesian Additive Regression Trees”, junto con la optimización de hiperparámetros mediante Ray Tune, con ejecución en paralelo. Con el fin de poder mejorar modelo bart con el ajuste de hiperparámetros para un rendimiento óptimo, a través de algoritmos de búsqueda eficiente y ejecución de tareas que se distribuyen en múltiples procesos.

Ray Tune es una biblioteca de Python para la optimización de hiperparámetros que se ejecuta sobre Ray. Mientras que BART es un modelo estadístico utilizado para el análisis y la modelización de datos. Combina dos técnicas: árboles de regresión aditivos y enfoque bayesiano.

El objetivo de este trabajo, es poder mostrar como modelos de bart junto a la ejecución en paralelo, pueden proporcionar una mejora en las predicciones, logrando mayor rendimiento, ahorro de tiempo y recursos, y precisión, para una efectiva y eficiente predicción que apoyen a estrategias y tomas de decisiones por parte de encargados directivos.

**Metodologías y Herramientas**

A continuación, se presenta la metodología que se llevó a cabo para el desarrollo del presente trabajo, basado en emplear principalmente herramientas de computación de alto rendimiento, contribuyendo a la mejora del modelo bart.

1. Entorno de trabajo

En primer lugar, se preparó el entorno de trabajo donde se lleva a cabo el desarrollo y evaluación de modelos. Esto en la supercomputadora Patagón, ya que es fundamental poder utilizar múltiples núcleos para realizar pruebas simultáneamente, y la ejecución en paralelo, aprovechando al máximo los recursos de Patagón:

* Se genera llave pública.
* Se solicita cuenta de usuario en Patagón, en este caso se utiliza la cuenta “comodín”.
* Se accede a Patagón
* En cuenta de usuario personal, se instalan programas y librerías a utilizar. Para este trabajo se instalaron las librerías PyMC3 para ejecutar modelo bart de R a Python. Luego la librería Ray Tune, para la ejecución multiproceso.

*Nota:* Como no se pudo instalar librerías en contenedor docker creado anteriormente, ni crear uno nuevo (lo cual era óptimo para tener un entorno aislado), se instalaron las librerías con las siguientes funciones, más ejemplo de instalación ray para posteriormente ray[tune]:





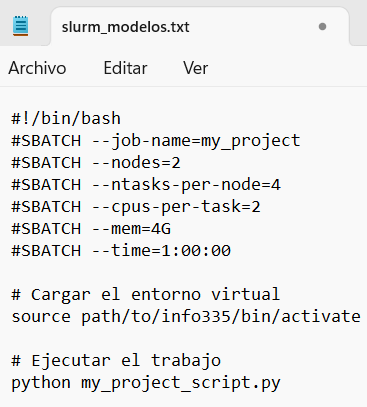


* 1. Especificación de recursos

Con el fin de solicitar recursos de manera efectiva y administrar los modelos en tareas en paralelo, se ejecuta un script de slurm, donde se especifica la cantidad de recursos necesarios, como CPU, memoria, tiempo de ejecución, y cantidad de nodos.

Luego se cargó al entorno con el que se está trabajando para posterior ejecución del trabajo.

Importante destacar, que esto se realiza a través del editor vim, en patagon y propio entorno de trabajo, sin embargo, a continuación, se adjunta imagen de representación de script y su descripción utilizado:



Esto indica que se asignan al proyecto 2 nodos en el clúster de cómputo; se definen 4 tareas (procesos) que se ejecutaran en cada uno de los 2 nodos; un total de memoria de acceso asignada a cada tarea, siendo 4 GB de memoria RAM; y finalmente es establecido un tiempo máximo de ejecución del proyecto, siendo 1 hora.

1. Recopilación y preparación de datos

Se utilizó mismo conjunto de datos utilizado actualmente en evaluación de modelos para el desarrollo actual de tesis de magister. El cual consiste en registros académicos institucionales, de alumnos de la Facultad de Ingeniería UACh, de la carrera de Ing. Civil Informática. El cual ya fue pre procesado. Los datos incluyen variables predictoras como créditos completados, GPA, asignaturas cursadas y otras métricas académicas, junto con la variable objetivo que indica la retención del estudiante al tercer semestre en el programa de la carrera.

1. Desarrollo y especificación de modelos BART

Se implementó modelo BART utilizando la biblioteca PyMC3 en Python; motivo de que modelos BART implementados anteriormente, fueron probados con librerías de R, no compatible con Ray Tune, por eso la necesidad de adaptarlo con PyMC3. BART es un modelo de regresión no lineal que combina árboles de regresión; técnica bayesiana flexible que permite el modelado no lineal de relaciones entre variables predictoras y de respuesta. Definimos el modelo BART con múltiples árboles de regresión y ajustamos los hiperparámetros.

Desarrollo:

* Definición de variables predictoras y respuesta.
* Definición de modelo *bart\_model.*
* Especificación de hiperparámetros: número de árboles; profundidad de árboles; valor alpha en las distribuciones prior.
* Especificación de priors para hiperparámetros: sigma (desv estándar); mu (promedio de valores esperados de cada árbol); tau (precisión de las hojas de los árboles); p (controla la proporción de división de nodos en cada árbol).
* Definición de árboles de regresión.
* Modelo BART.
* Definición de verosimilitud: distribución normal (cumple modelar la distribución de los datos observados y como se relación con los parámetros del modelo).

1. Optimización de hiperparámetros con Ray Tune

Se utilizó Ray Tune para buscar los mejores hiperparámetros del modelo BART. Se define un espacio de búsqueda con diferentes opciones para el número de árboles, profundidad de los árboles y valor de alpha en las distribuciones prior del modelo. Ray Tune realizó la búsqueda en paralelo utilizando múltiples procesos, lo que acelera significativamente la exploración del espacio de búsqueda.

Desarrollo:

* Definición de función de entrenamiento que utilizó modelo *bart\_model*.
* Ajuste de hiperparámetros en Ray Tune.
* Se define métrica a optimizar: se utiliza *log-likelihood negativo* como métrica para maximizar. (Métrica comúnmente utilizada en modelos probabilísticos para cuantificar qué tan bien se ajusta un modelo a los datos observados. Es especialmente relevante en modelos de regresión y clasificación que asumen una distribución probabilística para los datos.)
* Se define espacio de búsqueda de hiperparámetros para optimización.
* Ejecución de Ray Tune: función entrenamiento y espacio de búsqueda.
* Obtención de mejores hiperparámetros.

Se elige la métrica log-likelihood negativo, ya que es especialmente relevante en modelos de regresión y clasificación que asumen una distribución probabilística para los datos.

El log-likelihood (verosimilitud) es una medida de la probabilidad de observar los datos que tenemos dados los parámetros del modelo. En el contexto de modelos de regresión, como el modelo BART que mencionamos previamente, el log-likelihood representa la probabilidad de observar los datos de respuesta (variable dependiente) dada la matriz de variables predictoras y los parámetros del modelo.

La razón para utilizar el log-likelihood negativo en lugar del log-likelihood positivo es que en la optimización, generalmente buscamos minimizar una función de pérdida o maximizar una función de utilidad. Dado que el log-likelihood es una función que queremos maximizar, tomamos su negativo para convertirlo en una función que minimizamos.

En resumen, un log-likelihood negativo más bajo indica un mejor ajuste del modelo a los datos observados, ya que implica una mayor probabilidad de observar los datos bajo el modelo. En el contexto de la optimización de hiperparámetros con Ray Tune, generalmente buscamos maximizar el log-likelihood negativo para encontrar los mejores hiperparámetros que produzcan un modelo con el mejor ajuste a los datos. Cuanto menor sea el valor del log-likelihood negativo, mejor será la calidad del modelo en términos de su capacidad para explicar los datos observados.

1. Ejecución en paralelo con Ray Tune

La ejecución en paralelo con Ray Tune, nos permitió realizar múltiples pruebas simultáneamente. Ray Tune generó varios trabajos que ejecutaron el modelo BART con diferentes combinaciones de hiperparámetros. Estos trabajos se distribuyeron entre los procesos disponibles en el sistema, aprovechando al máximo los recursos de CPU y acelerando el tiempo de ejecución general.

Para este caso, el speed-up idealmente lo utilizamos dentro del script de optimización, o directamente desde el entorno de trabajo, para evaluar la eficiencia de la paralelización y determinar si es beneficioso ejecutar el modelo en paralelo. Para ello en primer lugar se registra la hora de inicio en ambos modelos, y luego de la ejecución, la hora de termino.

Por problemas técnicos no se pudo obtener el resultado real, sin olvidar que es importante registrar el valor obtenido, pero importante mencionar, que un speed-up de 1 significa que la ejecución en paralelo no ofrece ninguna mejora en el rendimiento en comparación con la ejecución en serie, y por el contrario, un speed-up mayor que 1 indica una mejora en el rendimiento. Cuanto mayor sea el speed-up, mayor será la mejora.

**Resultados y discusión**

Ajuste de Hiperparámetros

Resultados:

La ejecución en paralelo con Ray Tune mejoró significativamente la eficiencia de la búsqueda de hiperparámetros. Aprovechando múltiples procesos, pudimos explorar un amplio espacio de búsqueda en menos tiempo, lo que nos permitió obtener resultados más rápidamente y encontrar la configuración óptima para el modelo BART.

Los resultados mostraron que la configuración óptima para el modelo BART fue con 50 árboles de regresión, una profundidad de árbol de 3 y un valor alpha de 0.9. Estos hiperparámetros permitieron que el modelo BART capturara relaciones no lineales y alcanzara un alto rendimiento en la predicción de la retención estudiantil.

Junto a esto, se desarrollaron los siguientes gráficos para validar la mejora del modelo Ray Tune, frente al de BART

**Conclusiones**

En este trabajo, aplicamos el modelo BART y la optimización de hiperparámetros con Ray Tune para predecir la retención estudiantil. Los resultados mostraron que el modelo BART es efectivo para predecir la retención estudiantil y que la optimización de hiperparámetros mediante Ray Tune mejoró significativamente el rendimiento del modelo. Al aprovechar múltiples procesos, pudimos realizar una búsqueda más rápida y exhaustiva de los mejores hiperparámetros, lo que resultó en un modelo BART altamente preciso.

La predicción de retención estudiantil es una herramienta valiosa para las instituciones educativas, ya que les permite identificar a los estudiantes en riesgo y ofrecer intervenciones tempranas para mejorar su éxito académico. La combinación del modelo BART, Ray Tune y la ejecución en paralelo representa una metodología poderosa para lograr una predicción precisa y efectiva de la retención estudiantil.

En futuras investigaciones, se puede explorar la incorporación de más variables predictoras y considerar otros modelos de aprendizaje automático para comparar su rendimiento en la predicción de la retención estudiantil. Además, se pueden evaluar diferentes métricas de evaluación y realizar análisis más profundos de las causas subyacentes de la retención estudiantil para diseñar estrategias más personalizadas y efectivas de intervención.

En conclusión, este trabajo resalta la importancia de la ejecución en paralelo para optimizar los hiperparámetros de modelos complejos como BART, y cómo esta estrategia puede acelerar significativamente la búsqueda y mejorar el rendimiento del modelo en la predicción de retención estudiantil. Como resultado, este enfoque tiene el potencial de tener un impacto positivo en la mejora de los resultados académicos y la retención de estudiantes en instituciones educativas.

**Referencias**

[1] Hugh A. Chipman. Edward I. George. Robert E. McCulloch. "BART: Bayesian additive regression trees." Ann. Appl. Stat. 4 (1) 266 - 298, March 2010. https://doi.org/10.1214/09-AOAS285

[2] Ray Tune. (2023). Documentación de Ray Tune. Recuperado de <https://docs.ray.io/en/latest/tune/>

[3] Patagon Supercomputer, Austral University of Chile, 2021. https:// patagon.uach.cl