USO DE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO PARA IDENTIFICAR VARIABLES PREDICTORAS DE LA APROBACIÓN DEL CURSO INTRODUCTORIO A LA PROGRAMACIÓN

Jacqueline Köhler Casasempere, Universidad de Santiago de Chile, Jacqueline.Kohler@usach.cl
Luciano Hidalgo Sepúlveda, Universidad de Santiago de Chile, Luciano.Hidalgo@usach.cl
José Luis Jara Valencia, Universidad de Santiago de Chile, JoseLuis.Jara@usach.cl

RESUMEN

Dados los avances en automatización de procesos y servicios, la programación resulta un elemento clave en la formación de profesionales de carreras científicas. Sin embargo, programar no es una habilidad sencilla de adquirir. Con el objetivo de identificar qué variables son las que mayor correlación tienen con la aprobación a la hora de programar, se estudian los datos de estudiantes del curso de Fundamentos de Computación y Programación, común para estudiantes de la Facultad de Ingeniería de la Universidad de Santiago de Chile, con el objeto de predecir la aprobación o reprobación de la parte teórica del curso. Para esto se usan diversos clasificadores: Máquinas de Vectores Soporte (SVM), Regresión Logística, Árboles CART, *Extreme Learning Machines (ELM)*, *Random Forest* y *Extreme Gradient Boosting (XGB)*. Los algoritmos seleccionados se utilizan sobre un conjunto de datos de 3.130 estudiantes que cursaron la asignatura entre los años 2015 y 2019. El mejor clasificador se obtiene utilizando SVM con kernel radial y alcanza una exactitud de 68,6% para predecir si el estudiante aprueba o reprueba la parte teórica del curso.

PALABRAS CLAVES: Aprendizaje Automático, CS01, Programación en Ingeniería.

INTRODUCCIÓN

El mundo moderno necesita profesionales capaces de entender y escribir código (World Economic Forum, 2016). A raíz de esto, los cursos de programación, que antes eran parte de la formación específica de los profesionales de las ciencias de la computación, hoy en día han avanzado para integrarse en distintos contextos, desde la enseñanza de programación usando bloques en niños (Leidl et al. 2017) y adolescentes (De Kereki & Manataki, 2016) (Kalelioğlu, 2015), hasta su inclusión en la formación de profesionales de áreas tan distintas como la ingeniería (dos Santos et al. 2018), el diseño (Hansen, 2017), la medicina (Chen & Liu, 2022) y las ciencias sociales (Lee & Lien, 2019).

Pese a ello, programar presenta desafíos particulares (Piteira & Costa, 2012). Diversos estudios señalan que, para una persona sin preparación previa, el enfrentarse por primera vez a esta tarea puede representar un desafío de alta complejidad. Es por ello que entender cómo los novatos abordan la programación y cuáles son los factores que inciden en que algunos tengan éxito y otros no, es un campo de investigación activo (Emerson et al. 2019) (Sobral & Oliveira, 2021) y en evolución constante (Biamonte, 1964) (Leeper & Silver, 1982) (Bergin & Reilly, 2005).

Si bien existe consenso de que hay una correlación entre las habilidades en matemáticas y en ciencias básicas con facilidades a la hora de programar, en la actualidad no existe una respuesta definitiva respecto a cuáles son los factores que inciden en que un estudiante

obtenga un buen resultado en un curso introductorio de programación. Por un lado, autores como Loksa & Ko (2016) tratan de entender el fenómeno desde una perspectiva del proceso mismo de aprendizaje, planteando un método para resolución de problemas de esta naturaleza, mientras que Prather et al. (2018) levanta las dificultades desde una perspectiva meta-cognitiva. En una perspectiva distinta López et al. (2008) abordan el problema de las habilidades en programación separándolas según complejidad, desde la lectura de código hasta explicar en términos abstractos lo que un programa tiene como meta.

En un área más específica, existe un interés en determinar cuáles son las habilidades previas que inciden en una mayor facilidad a la hora de programar. En esta línea existen estudios que relacionan el éxito con habilidades de lenguaje y matemáticas (Qian & Lehman 2016), otros autores, como Hinckle et al. (2020) buscan relaciones con factores experienciales, psicológicos y de género. En el contexto chileno, Álvarez et al. (2019) estudia la relación entre actitudes, compromiso y capacidad de aprendizaje autónomo con la aprobación al final de un curso introductorio de programación en varios programas STEM, el mismo autor (Álvarez, 2019b) correlaciona la aprobación de un estudiante en un curso de programación con el valor percibido por ellos de sus habilidades en la disciplina. Por otro lado, Bellino et al. (2021) enfrentan la perspectiva motivacional, al considerarla uno de los factores determinantes del éxito del estudiante en un curso introductorio de programación.

El objetivo de este estudio es identificar si existen variables que, registradas previo al inicio de un curso de programación, permitan predecir si un estudiante aprobará o reprobará la parte teórica del curso. Esto con la intención de identificar cuáles son las características, en el contexto chileno, que presentan mayor correlación con que un estudiante pueda pasar el curso sin tener que repetirlo. Para esta tarea, siguiendo el mismo método que Bello et. Al (2020) y Köhler et al. (2020), a partir de variables socio-económicas, demográficas, de rendimiento académico pre-universitario y de rendimiento durante el primer semestre universitario, se busca establecer cuáles de ellas ayudan más a la predicción de un buen resultado, es decir la aprobación en un primer intento, en un curso introductorio de programación.

En este caso, se usan los datos de cuatro cohortes de estudiantes del curso de “Fundamentos de Computación y Programación” (FCYP) de la Universidad de Santiago de Chile, quienes rindieron el curso entre el segundo semestre de 2015 y el primer semestre de 2019. El curso está ubicado en el segundo semestre de los planes de estudio y es común para todos los estudiantes de régimen diurno de la Facultad de Ingeniería.

El resto del documento se estructura del siguiente modo: la sección 2 describe el conjunto de datos y los métodos utilizados. La sección 3 detalla los resultados obtenidos con los distintos métodos de aprendizaje automático utilizados. La sección 4 presenta la discusión de los resultados obtenidos. Finalmente, la sección 5 muestra las conclusiones obtenidas a partir de este trabajo y futuras direcciones en las cuales se orienta la investigación.

DATASET Y MÉTODOS UTILIZADOS

La asignatura de FCYP está ubicada en el segundo semestre lectivo y es común para todos los estudiantes de ingeniería de los planes de estudio diurno. Luego de un primer semestre de cinco cursos comunes: Álgebra I, Física I, Cálculo I, Introducción a la Ingeniería y Métodos de Estudio, los estudiantes pueden rendir FCYP si han aprobado el pre-requisito de Álgebra I. Al segundo semestre, junto con este curso, los estudiantes deberían rendir los cursos de Álgebra

II, Física II y Cálculo II, además de Química o Biología según el plan de estudios y, en algunos casos, el curso introductorio a la carrera de su especialidad.

Durante el período de observación (2015-2019), 6.516 estudiantes rindieron FCYP. Sin embargo, para esta experiencia se descartan los datos de estudiantes que: aprobaron el curso en instancias extraordinarias (cursos intensivos, pruebas especiales), que repiten el curso, estudiantes que venían de cohortes anteriores, que no pertenecían a carreras de ingeniería (principalmente estudiantes de los programas de Bachillerato) y aquellos con registros incompletos. Con esto, tras la limpieza y preprocesamiento de los datos, se trabajó con una muestra de 3.130 estudiantes. La variable a predecir es la situación final de teoría del curso, es decir, si el estudiante aprueba o reprueba. En este caso el conjunto de datos tiene 1.467 estudiantes de la clase “Aprueba” y 1.663 de la clase “Reprueba”.

Respecto a los datos de caracterización de los estudiantes, estos se agrupan en tres conjuntos de posibles variables predictoras: variables de ingreso (Tabla 1), que corresponden a datos declarados cuando el estudiante realiza su proceso de admisión en la Universidad; variables de primer semestre (Tabla 2), que son los resultados obtenidos previo a rendir el curso; y profesores de la asignatura, que considera el profesor con quien el estudiante rindió tanto teoría como laboratorio en la asignatura. Para cada modelo de aprendizaje automático, se realiza el proceso primero con el primer conjunto de datos, luego con el primero y el segundo y finalmente con todos los datos para identificar si existen variaciones.

Tabla 1 – Variables de ingreso consideradas.

VARIABLE	**TIPO**	**DESCRIPCIÓN**
CARRERA	Categórica	Código de cualquiera de las 20 carreras posibles consideradas
TIPO CARRERA	Binaria	Si la carrera es ingeniería de ejecución o no.
DEPARTAMENTO	Categórica	Departamento responsable por la carrera. Existen 9 departamentos posibles en el set de datos.
PREFERENCIA	Categórica	Orden de preferencia de la carrera al momento de postular.
PSU CIENCIAS	Entero	Puntaje Prueba de Selección Universitaria (PSU) Ciencias.
PSU LENGUAJE	Entero	Puntaje PSU Lenguaje.
PSU MATEMÁTICAS	Entero	Puntaje PSU Matemáticas.
PSU NEM	Entero	Puntaje PSU notas de enseñanza media.
PSU RANKING	Entero	Puntaje PSU ranking.
PUNTAJE TOTAL	Entero	Puntaje total de postulación.
GRATUIDAD	Binaria	Si el estudiante tiene o no gratuidad.
ESTABLECIMIENTO ENSEÑANZA MEDIA	Categórica	Diferencia si el estudiante proviene de un establecimiento de enseñanza media: Municipal, Particular o Particular subvencionado
QUINTIL	Categórica	Quintil de ingresos al que pertenece la familia al momento del ingreso.
IDH COMUNA	Flotante	Índice de desarrollo humano de la comuna de residencia del estudiante.

Para construir los predictores se utilizan las técnicas de aprendizaje automático de Máquinas de Vectores Soporte (SVM) (Noble, 2006) con *kernel* lineal y radial, Regresión Logística Multivariada (Menard, S., 2000) con eliminación recursiva de características, Árboles de clasificación CART (Steinberg, 2009), *Extreme Learning Machines* (ELM) (Ding et al. 2014), *Random Forest* (Cutler et. al. 2012) y *Extreme Gradient Boosting* (XGB) (Chen et al. 2015). En todos los casos se trabaja sin balanceo de datos, puesto que la cantidad de ejemplos para las clases de salida son cercanas al 50%.

Tabla 2 – Variables de primer semestre consideradas.

VARIABLE	**TIPO**	**DESCRIPCIÓN**
VECES QUE RINDIÓ ÁLGEBRA	Entero	Cantidad de veces que el estudiante rinde Álgebra I (Prerrequisito del curso observado).
CÁLCULO	Float	Nota final la primera vez que rinde Cálculo I.
FÍSICA	Float	Nota final la primera vez que rinde Física I.
ÁLGEBRA	Float	Nota final la primera vez que rinde Álgebra I.
MÉTODOS DE ESTUDIO	Float	Nota final la primera vez que rinde Métodos de Estudio.
INTRODUCCIÓN A LA INGENIERÍA	Float	Nota final la primera vez que rinde Introducción a la Ingeniería.
PPA	Float	Promedio ponderado acumulado (PPA), en este caso, considerando todas las asignaturas del primer semestre.

Para algunos modelos se requirió de la optimización de algunos parámetros con el objetivo de obtener mejores resultados y conseguir resultados en tiempos de cómputo razonables. En particular, se ajustaron:

- En *Random Forest* se ajusta el parámetro *mtry*, que determina la cantidad de variables seleccionadas aleatoriamente en cada split, en este caso se utiliza un número aleatorio entre 2 y la cantidad máxima de variables predictoras en cada paso. Por otro lado, *ntree*, que determina la cantidad árboles a construir, para este caso se considera siempre un valor fijo de 600.
- Para SVM se ajusta el costo, que regula la holgura del margen de separación dejando siempre un valor entre 2^{-10} y 2^{10}. En los casos en que se usa kernel radial también se ajusta la curvatura de la frontera de decisión (*sigma*) para los mismos valores.
- En Árbol CART se ajusta el parámetro de complejidad, manejando siempre valores entre 0 y 0,5.
- Con ELM se ajusta la cantidad de neuronas en la capa oculta para trabajar siempre con un máximo de 50 y la función de activación que se utiliza, en este caso se usó *purelin*, que corresponde a una función de transferencia lineal.
- En XGB se regulan los parámetros *nrounds*, que determina la cantidad de árboles en el modelo final, probando con valores entre 10 y 50. También se ajustan los valores de *lambda* con valores entre 0,8 y 1,2; y *alpha* con valor de 0,0 ambos parámetros tienen como objetivo regularizar los pesos L2 y L1 respectivamente. Para prevenir sobreajuste se prueban valores del ajuste de compresión de preso entre 0 y 0,5.

Todos los clasificadores han sido desarrollados utilizando R, versión 4.1.0 con RStudio 2022.02.3 Build 492.

RESULTADOS

Los resultados obtenidos tras el proceso de entrenamiento se presentan en la Tabla 3. Los mejores predictores, en este caso, se obtienen con SVM y Regresión Logística. Por otro lado, ELM obtiene resultados marginalmente inferiores, pero usando el conjunto de datos que no contempla la información de los profesores con los que el estudiante rindió la asignatura.

Considerando los mejores diez modelos, se observa que en algunos casos la inclusión de la información de los profesores que el estudiante tuvo en la asignatura no necesariamente mejora la precisión de los clasificadores. El mejor predictor obtiene un 68,6% de exactitud y utiliza SVM de *kernel* radial con un parámetro de costo de 2 y un sigma de 0,00390625. Cinco otros modelos obtienen precisiones iguales o superiores al 68% de exactitud. De ellos, dos utilizan ELM, uno regresión logística y los otros son variantes de SVM (radial y lineal).

Tabla 3 – Resultados obtenidos.

Modelo	**Dataset**	**Exactitud**	**Sensibilidad**	**Especificidad**
SVM radial	Ingreso + semestre 1 + profesores	0,686	0,615	0,748
Regresión logística	Ingreso + semestre 1 + profesores	0,683	0,634	0,726
ELM	Ingreso + semestre 1	0,681	0,639	0,717
SVM radial	Ingreso + semestre 1	0,681	0,596	0,756
SVM lineal	Ingreso + semestre 1 + profesores	0,681	0,629	0,727
ELM	Ingreso + semestre 1 + profesores	0,68	0,633	0,721
SVM lineal	Ingreso + semestre 1	0,679	0,625	0,727
Regresión logística	Ingreso + semestre 1	0,678	0,636	0,716
Random Forest	Ingreso + semestre 1 + profesores	0,678	0,593	0,753
Random Forest	Ingreso + semestre 1	0,671	0,581	0,75
XGB	Ingreso + semestre 1 + profesores	0,665	0,577	0,743
XGB	Ingreso + semestre 1	0,66	0,569	0,74
CART	Ingreso + semestre 1	0,657	0,601	0,706
CART	Ingreso + semestre 1 + profesores	0,654	0,59	0,71
Regresión logística	Ingreso	0,628	0,559	0,688
ELM	Ingreso	0,627	0,56	0,686
Random Forest	Ingreso	0,619	0,489	0,734
SVM radial	Ingreso	0,618	0,522	0,702
SVM lineal	Ingreso	0,617	0,523	0,7
CART	Ingreso	0,616	0,494	0,724
XGB	Ingreso	0,61	0,509	0,699

Desde otra vereda, es posible observar que los modelos generados con XGB y Árboles CART tienden a entregar, para los 3 conjuntos de datos utilizados, los peores resultados. Sin embargo, estos últimos no distan abismalmente de aquellos entregados por los mejores modelos y solo implican pérdidas de precisión de entre 2% y 4%, según sea el caso. Esto podría deberse a distintos factores: en XGB es posible que el ajuste de parámetros busque sobre un espacio de posibilidad demasiado limitado y que esto no permita encontrar mejores clasificadores. Por otro lado, al usar árboles CART con el parámetro de complejidad ajustado, se fuerza la generación de árboles pequeños que evitan el sobreajuste, pero que probablemente ignoren *valores atípicos,* lo que podría excluir algunos comportamientos que no necesariamente son tan poco frecuentes como el algoritmo estima.

Por otro lado, al observar aquellos clasificadores que realizan selección de características, es decir, CART y Regresión Logística, se observa que entre ellos no existe un acuerdo entre cuáles son las variables que se correlacionan más fuertemente con la variable a predecir. Al usar el conjunto de datos de ingreso, CART considera como características más importantes los puntajes de PSU de Matemática y de postulación, mientras que Regresión Logística considera como mejor predictor si el estudiante tiene gratuidad en su arancel, seguido por la carrera escogida y el departamento académico al que esta pertenece.

Al incorporar las variables del primer semestre, CART considera como mejores predictores el promedio ponderado acumulado del estudiante en su primer semestre, seguido por sus promedios individuales de Cálculo, Álgebra y Física. Mientras que el clasificador de Regresión Logística considera como predictores de mayor incidencia las veces que el estudiante ha rendido Álgebra, la carrera a la que pertenece, el promedio con el que aprobó Álgebra y si tiene o no gratuidad. Finalmente, al considerar el conjunto de datos completo, CART no modifica las variables escogidas, en cambio el modelo de Regresión incorpora la variable del profesor de teoría como segunda variable de relevancia, lo cual indicaría que el evaluador de cada estudiante es un factor de relevancia en la predicción.

DISCUSIÓN

Tras realizar los experimentos es posible señalar que, solo considerando datos previos a que el estudiante rinda la asignatura, es posible predecir en aproximadamente 2/3 de los casos la reprobación o aprobación de este. Esto puede analizarse desde varias ópticas, las cuáles se exponen a continuación.

En primer lugar, los resultados obtenidos implican que, debido a factores preexistentes, los estudiantes parten en una situación de desigualdad para rendir el curso, y que, en un número importante de casos, el trabajo realizado durante el semestre no alcanza para modificar el rumbo predicho. Si se combina este hallazgo con la distribución de aprobados y reprobados, es posible aventurar que se requiere fortalecer los mecanismos de diagnóstico y nivelación de los estudiantes en las primeras semanas del curso.

En segundo lugar, al considerar las variables predictoras, la selección realizada por el modelo de Regresión Logística representa un hallazgo importante, pues en vez de seleccionar características asociadas a un buen rendimiento universitario en general, como en el caso de la selección que hace CART, este modelo selecciona variables *ad-hoc* a la disciplina y concordantes con la literatura. Por un lado, la selección de las veces que el estudiante rinde Álgebra y la calificación obtenida en ese curso confirman la necesidad de este curso como

requisito para rendir FCYP, situación que se contradice con cursos de programación más recientes que ha comenzado a impartir la Facultad de Ingeniería.

Respecto a los otros predictores, tiene sentido que una de las variables relevantes sea la carrera, pues esta es un indicador de otra información que no necesariamente está visibilizada, por ejemplo: los intereses y el perfil de ingreso del estudiante, los requerimientos de ingreso de cada una (como el puntaje de corte), la infraestructura con la que cuentan los estudiantes para estudiar, entre otros. Pese a que la Facultad de Ingeniería tiene estudiantes de diversos orígenes y características, estos no necesariamente se distribuyen uniformemente en cada carrera, por lo que hay carreras con estudiantes mejor preparados para un curso de este estilo y otras que no, lo que significa un desafío a enfrentar por el cuerpo docente de FCYP.

En este punto vale la pena mencionar que la inclusión de los profesores de teoría y laboratorio al conjunto de variables fue una decisión del equipo investigador con el objetivo de comprobar si existían sesgos en el equipo docente que imparte el curso. En FCYP, por la forma en que se realiza el proceso de inscripción de asignaturas, los estudiantes normalmente tienen un único horario en el cuál inscribir el curso y solo pueden escoger a los profesores que dictan el curso en ese horario. Esto genera que exista una relación forzosa entre la carrera del estudiante y el profesor de teoría con el cual rinde el curso, pues estos tienden a mantener su horario en el tiempo. Por ello, la aparición de ambas variables como predictores es un tema que requiere mayor investigación, pues la aparición del profesor como predictor puede deberse simplemente al arrastre de que este normalmente trabaja con las mismas carreras, pero también podría indicar sesgos por parte de los docentes, ya sea en el proceso de enseñanza o en el de evaluación, lo que implicaría que por la restricción en la posibilidad de escoger a su profesor, algunos estudiantes estarían siendo beneficiados por sobre el resto.

Los resultados obtenidos permiten a los actores involucrados tanto en el diseño del curso como en su implementación orientar la discusión hacia acciones concretas para mejorar, por un lado, las tasas de aprobación de los estudiantes que rinden la asignatura sin comprometer los resultados de aprendizaje y, por otro lado, permitir una nivelación adecuada que elimine las diferencias de ingreso identificadas en este trabajo.

CONCLUSIONES

El presente trabajo consigue, mediante métodos de aprendizaje automático, predecir con un 68,6% de exactitud si un estudiante aprobará o reprobará la teoría del curso de Fundamentos de Computación y Programación. El modelo de mayor precisión en la predicción utiliza SVM con *kernel* radial y todos los modelos generados consideran únicamente variables registradas antes de que el estudiante comience a rendir el curso. El contar con un clasificador de este tipo permite orientar acciones concretas para identificar grupos en riesgo de reprobación y plantear estrategias que permitan superar estas dificultades. Esto abre una oportunidad para que la institución responda proactivamente a estas situaciones en vez de reaccionar una vez que el semestre ha terminado, como históricamente se ha hecho, considerando cursos intensivos para estudiantes reprobados o pruebas adicionales una vez terminado el semestre.

Dado que se utilizaron tanto métodos que realizan selección explícita de características como otros que no, la predicción de algunos funciona como caja negra, donde no es posible determinar cuánto peso tiene cada variable en la predicción. Sin embargo, los rendimientos de los mejores modelos en cada caso son comparables: 68,6% de SVM con *kernel* radial contra

68,3% con Regresión Logística. Considerando la posibilidad de mejorar la tasa de predicción, se plantean dos posibles vías a explorar a futuro: la primera es trabajar con técnicas de balanceo de datos, pues para el desarrollo de este trabajo se decidió no utilizarlas pues ambas clases de salida poseían un número comparable de ejemplos de entrenamiento. Otra opción sería generar ensambles de modelos, usando técnicas como votación (Raza, 2019), *bagging* (Breiman, 1996)*, stacking* (Divina et. al 2018) u otra, con el objeto de aprovechar las diferencias en las estrategias de predicción de los modelos usados para combinarlos y crear un mejor predictor.

Pese a que en este caso es posible predecir el resultado, esta situación no necesariamente es alentadora, pues implica que, en un número importante de casos, un semestre académico no alcanza para revertir la situación. Esto lleva a que existan estudiantes que deben rendir el curso en múltiples ocasiones para poder aprobarlo. En esta línea, se espera en el futuro identificar con datos recolectados durante el semestre cuáles son los puntos críticos en donde se puede apoyar al estudiante para escapar de una eventual situación de reprobación o abandono del curso.

Este trabajo ha demostrado que es posible predecir con cierta certeza el devenir de una o un estudiante en una asignatura específica. Sin embargo, se podría esperar que con conjuntos de datos similares, compuestos por variables que las instituciones de educación superior comúnmente manejan, fuera posible estimar el rendimiento que una cohorte de estudiantes podría tener en un curso determinado, lo que ayudaría en procesos de gestión docente y de infraestructura que normalmente son complejos e inciertos en los cursos de primer año y no necesariamente estables en el tiempo. Por otro lado, un entendimiento acabado de las variables que influyen en el rendimiento permite mejores diseños curriculares, pues con este tipo de modelos, es posible establecer si un prerrequisito para rendir una asignatura es realmente relevante para un buen desempeño en esta.

En esta línea, el método utilizado es replicable con distintos conjuntos de datos y variables de entrada, por lo que una perspectiva futura es estudiar las diferencias entre planes de estudio de ingeniería de ejecución y de ingeniería civil, pues pese a que el curso es común para todas y todos los estudiantes de ambos programas de estudio, los perfiles de ingreso difieren tanto desde la perspectiva académica como socio-económica, por lo que es necesario estudiarlos separadamente.

Finalmente, si bien las técnicas de aprendizaje automático representan herramientas potentes para establecer relaciones entre variables, el uso de estas debe ir aparejado con un tratamiento ético riguroso, tanto de los datos como de los hallazgos. Los buenos resultados en una predicción no implican que un o una estudiante en riesgo de reprobación esté destinado al fracaso, sino que revelan puntos ciegos sobre los cuales se debe actuar institucionalmente con el objetivo de revertir esta asociación. En este sentido, el desafío está en que las instituciones deben ser capaces de hacer intervenciones orientadas a apoyar a las y los estudiantes en riesgo sin individualizarlos, ni creando grupos diferenciados para ellos, pues idealmente una o un estudiante en riesgo debería poder acceder a estas ayudas sin saber que pertenece al grupo en peligro de deserción. Del mismo modo, lo ideal sería que ni los equipos docentes de las asignaturas ni los tomadores de decisiones conozcan el detalle de qué estudiantes pertenecen a los grupos de riesgo, a fin de que esta información no sea considerada en decisiones, tanto pedagógicas como administrativas, que se tomen sobre estudiantes en particular, como por

ejemplo, solicitudes de reincorporación o para rendir nuevamente una asignatura ya reprobada el máximo de veces.

REFERENCIAS

Álvarez, C., Fajardo, C., Meza, F., & Vásquez, A. (2019, November). An exploration of STEM freshmen's attitudes, engagement and autonomous learning in introductory computer programming. In *2019 38th International Conference of the Chilean Computer Science Society (SCCC)* (pp. 1-8). IEEE.

Alvarez, C., Wise, A., Altermatt, S., & Aranguiz, I. (2019b). Predicting academic results in a modular computer programming course. In *2nd Latin American Conference on Learning Analytics, LALA* (Vol. 2425, pp. 21-30).

Bellino, A., Herskovic, V., Hund, M., & Munoz-Gama, J. (2021). A real-world approach to motivate students on the first class of a computer science course. *ACM Transactions on Computing Education (TOCE)*, *21*(3), 1-23.

Bello, F. A., Köhler, J., Hinrechsen, K., Araya, V., Hidalgo, L., & Jara, J. L. (2020, November). Using machine learning methods to identify significant variables for the prediction of first-year Informatics Engineering students dropout. In *2020 39th International Conference of the Chilean Computer Science Society (SCCC)* (pp. 1-5). IEEE.

Bergin, S., & Reilly, R. (2005, February). Programming: factors that influence success. In *Proceedings of the 36th SIGCSE technical symposium on Computer science education* (pp. 411-415).

Biamonte, A. J. (1964, July). Predicting success in programmer training. In *Proceedings of the second SIGCPR conference on Computer personnel research* (pp. 9-12).

Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., & Chen, K. (2015). Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4), 1-4.

Chen, X., & Liu, W. (2022). The Value of Python Programming in General Education and Comprehensive Quality Improvement of Medical Students Based on a Retrospective Cohort Study. *Journal of Healthcare Engineering, 2022*.

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012). Random forests. In *Ensemble machine learning* (pp. 157-175). Springer, Boston, MA.

De Kereki, I. F., & Manataki, A. (2016, October). “Code Yourself” and “A Programar”: A bilingual MOOC for teaching computer science to teenagers. In *2016 IEEE Frontiers in education conference (FIE)* (pp. 1-9). IEEE.

Ding, S., Xu, X., & Nie, R. (2014). Extreme learning machine and its applications. *Neural Computing and Applications*, 25(3), 549-556.

Divina, F., Gilson, A., Goméz-Vela, F., García Torres, M., & Torres, J. F. (2018). Stacking ensemble learning for short-term electricity consumption forecasting. *Energies*, 11(4),

dos Santos, M. T., Vianna Jr, A. S., & Le Roux, G. A. (2018). Programming skills in the industry 4.0: are chemical engineering students able to face new problems?*. Education for Chemical Engineers*, 22, 69-76.

Emerson, A., Rodríguez, F. J., Mott, B., Smith, A., Min, W., Boyer, K. E., Smith, C., Wiebe, E. & Lester, J. (2019). Predicting Early and Often: Predictive Student Modeling for Block-Based Programming Environments. *International Educational Data Mining Society.*

Hansen, S. M. (2017). Deconstruction/Reconstruction: A pedagogic method for teaching programming to graphic designers. In *Generative Arts Conference 2017* (pp. 419-431). Generative Art Conference.

Hinckle, M., Rachmatullah, A., Mott, B., Boyer, K. E., Lester, J., & Wiebe, E. (2020, June). The relationship of gender, experiential, and psychological factors to achievement in computer science. In *Proceedings of the 2020 ACM conference on innovation and technology in computer science education* (pp. 225-231).

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code.org. *Computers in Human Behavior*, 52, 200-210.

Köhler, J., Bello-Robles, F. A., & Jara, J. L. (2020, November). Predictive model for estimating internal transfer of Informatics Engineering students. In *2020 39th International Conference of the Chilean Computer Science Society (SCCC)* (pp. 1-5). IEEE.

Lee, Y. J., & Lien, K. W. (2019, May). Reconstruct Programming 101 for Social Science Preference Students. In *2019 IEEE International Conference on Consumer Electronics-Taiwan* (ICCE-TW) (pp. 1-2). IEEE.

Leeper, R. R., & Silver, J. L. (1982). Predicting success in a first programming course. *ACM SIGCSE Bulletin,* 14(1), 147-150.

Leidl, K. D., Bers, M. U., & Mihm, C. (2017). Programming with ScratchJr: a review of the first year of user analytics. In *Conference Proceedings of International Conference on Computational Thinking Education* (pp. 116-121).

Loksa, D., & Ko, A. J. (2016, August). The role of self-regulation in programming problem solving process and success. In *Proceedings of the 2016 ACM conference on international computing education research* (pp. 83-91).

Lopez, M., Whalley, J., Robbins, P., & Lister, R. (2008, September). Relationships between reading, tracing and writing skills in introductory programming. In *Proceedings of the fourth international workshop on computing education research* (pp. 101-112).

Menard, S. (2000). Coefficients of determination for multiple logistic regression analysis. *The American Statistician*, 54(1), 17-24.

Noble, W. S. (2006). What is a support vector machine? *Nature biotechnology*, 24(12), 1565-1567.

Prather, J., Pettit, R., McMurry, K., Peters, A., Homer, J., & Cohen, M. (2018, August). Metacognitive difficulties faced by novice programmers in automated assessment tools. In *Proceedings of the 2018 ACM Conference on International Computing Education Research* (pp. 41-50).

Piteira, M., & Costa, C. (2012, June). Computer programming and novice programmers. In *Proceedings of the Workshop on Information Systems and Design of Communication* (pp. 51-53).

Qian, Y., & Lehman, J. D. (2016). Correlates of success in introductory programming: A study with middle school students. *Journal of Education and Learning*, *5*(2), 73-83.

Raza, K. (2019). Improving the prediction accuracy of heart disease with ensemble learning and majority voting rule. In *U-Healthcare Monitoring Systems* (pp. 179-196). Academic Press.

Sobral, S. R., & Oliveira, C. F. D. (2021). Predicting students performance in introductory programming courses: a literature review.

Steinberg, D. (2009). CART: classification and regression trees. In *The top ten algorithms in data mining* (pp. 193-216). Chapman and Hall/CRC.

World Economic Forum. (2016). The future of jobs: Employment, skills and workforce strategy for the fourth industrial revolution. *Global Challenge Insight Report*.