Predicción de la deserción estudiantil en la Tecnicatura Universitaria en Programación de la UTN - FRRe (Informe de Aplicación)

Doctorado en Informática (UNNE - UNAM - UTN)
Introducción al Aprendizaje de Máquina
Docente: Dr. Eduardo Zamudio
Autores: Branca, Fernando y Gaona, Germán

Motivación

La *deserción o abandono estudiantil* es un problema extendido en todos los niveles educativos, aunque generalmente se acentúa en el nivel medio y superior, por lo tanto, la Tecnicatura Universitaria en Programación (TUP) no es ajena a este fenómeno. La TUP es una carrera presencial de pregrado (carrera corta) arancelada que posee un período de cursado de 2 1⁄2 años, si bien es a ciclo cerrado, se dicta desde hace 25 años en la Facultad Regional Resistencia de la Universidad Tecnológica Nacional.
En particular, uno de los autores de este proyecto se desempeña como Coordinador de la TUP, por lo que es de especial interés *predecir aquellos alumnos que tienen altas probabilidades de abandonar la carrera*, con el objetivo de encarar acciones tendientes a prevenir dicho abandono. Se considera que un enfoque basado en aprendizaje de máquina podría servir para alcanzar el objetivo mencionado, a través de algoritmos de aprendizaje supervisado.

Metodología

Formulación del problema

Los alumnos de la TUP se pueden inscribir a cursar o a rendir varias materias por cuatrimestre, la carrera se compone de cuatro cuatrimestres repartidos en dos años más la Práctica Profesional que se realiza en el tercer año.
Los docentes registran la condición de regularidad (libre, aprobación de cursada o aprobación directa) del alumno en el sistema académico de la UTN (SysAcad), generalmente, al final de cada cuatrimestre, por lo que es difícil obtener un período de tiempo menor que este. Por otra parte, no se cuentan con registros históricos de la plataforma LMS utilizada (Moodle), ya que se suele reiniciar al inicio de cada ciclo lectivo. Por lo tanto, la fuente de datos elegida fue SysAcad, la que cuenta con registros de alumnos de la carrera desde el año 1996.

Tarea: Predecir si determinado alumno abandonará o no la carrera (Clasificación binaria)

¿Cómo se define deserción o abandono estudiantil para este trabajo?

Un alumno se considera desertor si no posee inscripciones a cursadas o finales, en los dos ciclos lectivos posteriores al de ingreso a la carrera.

Relevamiento de la literatura existente

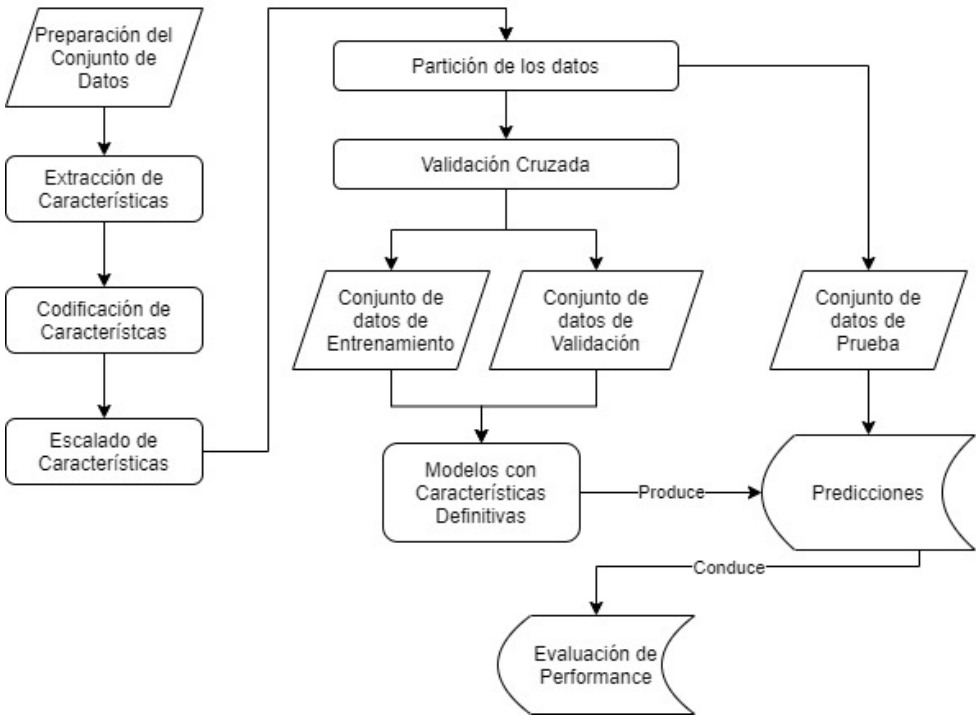
Con el objetivo de poder relevar los modelos que se usan en aplicaciones similares de aprendizaje automático, se efectuó una revisión (no sistemática) de la literatura. Para la búsqueda se usó el motor Google Scholar y se analizaron los 20 primeros resultados (ordenados por relevancia) de tres cadenas de búsqueda:

- Cadena 1: "time series dropout prediction", 15 de estos artículos, entre ellos (Haiyang et al., 2018; Tang et al., 2018; Wang et al., 2017), estudiaron la deserción estudiantil con registros provenientes de algún Learning Management System (LMS) que da soporte a Massive Open Online Courses (MOOC).
- Cadena 2: "university dropout prediction machine learning", se pudo acceder a 14 artículos, 6 de ellos fueron compatibles con escenarios de dictado de clases presenciales y además utilizaron un conjunto de datos similar al que se dispone para este trabajo, como ser: (Del Bonifro et al., 2020; Kotsiantis et al., 2003; Solis et al., 2018), los 8 restantes estaban vinculados a MOOC.
- Cadena 3: "university attrition prediction machine learning", una variante de la cadena anterior, 10 artículos fueron relevantes, entre ellos (Aulck et al., 2016; Berens et al., 2018; Chai & Gibson, 2015), el resto no se ajustaba al campo de aplicación.

Luego de analizar la muestra de artículos seleccionados relacionados con la deserción estudiantil, se detectó que todos aquellos que usan un enfoque de series temporales, son cursos (generalmente MOOC) que tienen como plataforma principal o de apoyo un LMS, y por consiguiente cuentan con gran cantidad de datos y de cortos períodos de tiempo.

Por otra parte, los artículos que intentan predecir la deserción estudiantil en entornos universitarios tradicionales (y sin datos provenientes de un LMS) se enfocan en características demográficas y académicas agrupadas por un período de tiempo, generalmente de uno o dos semestres, y el enfoque temporal no suele formar parte de su estrategia. Esto último es compatible con el entorno de aprendizaje y los datos históricos con los que se cuenta para este trabajo.

Figura 1 - Flujo de trabajo



Flujo de trabajo

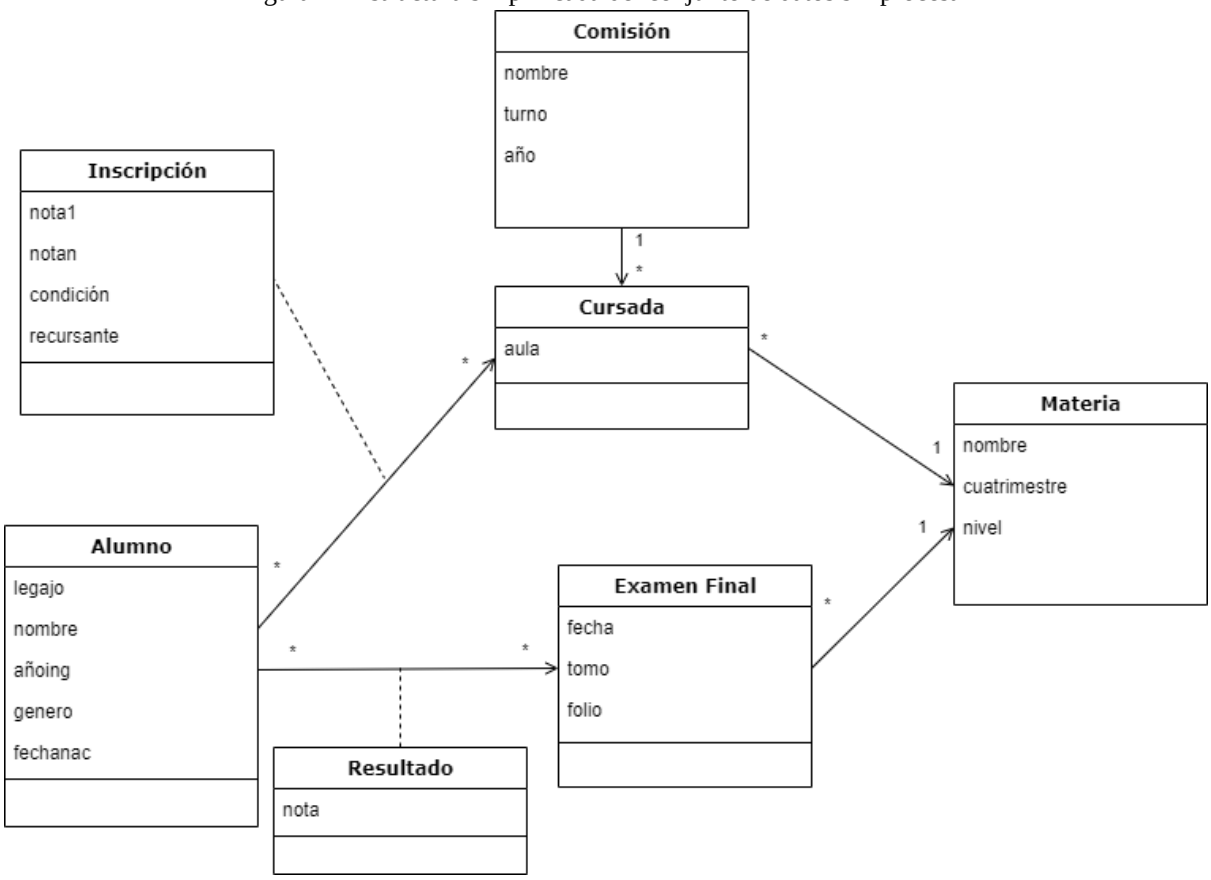
En la Figura 1 se presenta de manera resumida el flujo de trabajo seguido para desarrollar el modelo de aprendizaje de máquina con el objetivo de ejecutar la tarea ya especificada, este enfoque fue tomado de (Chai & Gibson, 2015).

Preparación del conjunto de datos

Análisis preliminar

Al igual que en muchos sistemas transaccionales, los datos provenientes de SysAcad se encuentran normalizados, por lo que aquellos vinculados a cada *alumno*, están distribuidos en múltiples registros. Por ej., un alumno podría tener varias notas por materia, varias cursadas y varios finales rendidos. En la Figura 2 se muestra un diagrama simplificado, este se enfoca en los datos de interés y su estructura.

Figura 2 - Estructura simplificada del conjunto de datos sin procesar



El conjunto de datos en crudo se extrajo de SysAcad, en formato MS Excel, ya que no se tuvo acceso a la base de datos subyacente. Para esto fue necesario acceder a tres vistas diferentes: alumnos, cursadas y exámenes. A continuación, se caracterizan cada una de ellas.

- Alumnos: permite obtener el total de alumnos que se inscribieron desde el año 1996 hasta 2021 a la carrera, con un total de *2506 entradas*. Los principales inconvenientes que surgieron se debieron a la registración errónea de la escuela secundaria, ya sea porque las escuelas tenían nombres inconsistentes, su tipo no fue correctamente identificado o directamente no estaba especificada la misma.

- Cursadas: permite obtener el listado de materias a las que se inscribieron los alumnos, indicando entre otros el año, la comisión, las notas y su condición final, desde el año 1996 hasta el 2021 totalizan *30453 entradas*. Los principales inconvenientes que surgieron son: la falta de las notas correspondientes a la cursada y el cambio de régimen de promoción.
- Exámenes: permite obtener el listado de inscripciones a exámenes finales por alumno y materia, indicando entre otros: el alumno, la materia, la fecha de examen y la nota final. El total de entradas asciende a *19053*. No se detectaron problemas en estos datos.

Aplicación de filtros

- Año de ingreso: considerando los inconvenientes mencionados, se optó por tomar los registros existentes desde 2010, ya que poseen los datos más completos y homogéneos. Por otra parte, para evitar el ruido que podría incluir el efecto de la pandemia en el abandono, se excluyeron los años 2020 y 2021. Asimismo, hay que considerar que según el planteo de la tarea y el criterio de abandono propuesto, se necesita observar un período de dos años consecutivos posteriores al ingreso. Como resultado, el rango va desde el 2010 al 2017 inclusive.
- Extensión áulica: se incluyeron solamente los registros vinculados a las cursadas en casa central ya que posee una planta de docentes más estables en el tiempo.

Con la aplicación de los filtros mencionados, la cantidad de alumnos ingresantes se redujo a 712, las cursadas a 12725 y los exámenes a 9144 (sin incluir el filtro de extensión)

Limpieza

De los alumnos restantes, se corrigieron los nombres de escuelas secundarias, que referían a la misma escuela, pero que estaban escritos de diferentes maneras. Así mismo, en base al nombre de las mismas, se imputaron los valores faltantes respecto de si se trataba de una escuela técnica o no técnica y pública o privada. Finalmente se eliminaron 37 registros que no indicaban la escuela secundaria, por considerarse una cantidad poco representativa, con lo que la *cantidad total de alumnos que forman el conjunto de datos alcanza 675*.

Transformación (cálculos previos)

Debido a la estructura en que los datos se encontraban almacenados, se consideró necesario realizar el procesamiento de estos en un motor de base de datos relacional y la aplicación de consultas SQL, en particular MS SQL Server y T-SQL. Las que requirieron mayor procesamiento fueron las características *x8* a *x13*, y la etiqueta de salida *y1* que indica si abandona o no la carrera.

Extracción de características

En base al relevamiento realizado de la literatura y el conjunto de datos obtenido, se elaboró el siguiente vector de características vinculadas al *primer ciclo lectivo del alumno* de la carrera. Cabe aclarar que difiere levemente del propuesto originalmente, ya que al analizar la fuente de datos algunas de ellas no eran posibles obtenerlas de manera confiable, como por ejemplo el turno de la cursada, sin embargo, se agregaron otras vinculadas al tipo de escuela secundaria (Tabla 1).

Codificación de datos categóricos

Las características *x4* y *x7*, contienen valores de tipo cadena representando categorías de 203 escuelas secundarias y 28 ciudades distintas respectivamente. Se experimentó con dos clases del módulo “preprocessing” de scikit-learn, primeramente, se usó “OrdinalEncoder”, que codifica los datos

categóricos como un arreglo de enteros y luego “OneHotEncoder”, que, a diferencia del anterior, crea una columna por cada categoría con valores 0 y 1 según corresponda. A pesar de que generalmente se recomienda el último codificador, el desempeño obtenido no fue satisfactorio, con la desventaja de que se requerían 231 columnas contra las 2 requeridas por la primera codificación, por lo que finalmente se eligió “OrdinalEncoder”.

Tabla 1 - Vector de características

Variable		Descripción	Estadísticas
y1	Abandona	Salida esperada, de clasificación binaria (0=No abandona, 1=Abandona)	Abandona: 43,11% No Abandona: 56,89%
x1	Sexo	0=Masculino, 1=Femenino	Masculino: 88,15% Femenino: 11,85%
x2	EdadIng	Edad del alumno al ingreso (entero)	Media: 23; DE: 6,63
x3	AñoIng	Año de ingreso a la carrera (entero)	2010 a 2012: 24,15% 2013 a 2014: 44,89% 2015 a 2017: 30,96%
x4	NomEscSec	Escuela secundaria a la que asistió (cadena)	203 escuelas distintas
x5	Pública	Si es una escuela secundaria de administración pública o privada (0=Privada, 1=Pública)	Pública: 77,48% Privada: 22,52%
x6	Técnica	Si es una escuela secundaria con orientación técnica o no (0=No técnica, 1=Técnica)	No técnica: 77,93% Técnica: 22,07%
x7	Ciudad	Ciudad de residencia del alumno (cadena)	Resistencia y alrededores: 84,15%
x8	C1Lib	Cantidad de materias libres en el 1er cuatrimestre en el 1er año (entero)	37,19% queda libre en todas las materias (5) y el 12% en ninguna.
x9	C1Reg	Cantidad de materias regularizadas en 1er cuatrimestre en el 1er año (entero)	50,07% no regulariza materias (74,26% queda libre en todas las materias)
x10	C1Prom	Cantidad de materias aprobadas directamente o promocionadas en 1er cuatrimestre en el 1er año (entero)	58,07% no promociona ninguna materia y el 1,33% todas (5)
x11	C2Lib	Cantidad de materias libres en el 2do cuatrimestre en el 1er año (entero)	44,15% queda libre en todas las materias (6) y el 18,67% en ninguna
x12	C2Reg	Cantidad de materias regularizadas en 2do cuatrimestre en el 1er año (entero)	70,96% no regulariza materias (62,21% queda libre en todas las materias)
x13	C2Prom	Cantidad de materias aprobadas directamente o promocionadas en 2do cuatrimestre en el 1er año (entero)	59,26% no promociona materias y el 29,33% 1 o 2

Escalado

Las características con valores numéricos poseen diferentes magnitudes en su representación original, por ejemplo, la edad de ingreso varía desde los 12 (valor anormal) hasta los 63, con una media de 23, en cambio el año de ingreso oscila entre 2010 y 2017, en tanto que las cantidades de materias lo hacen

entre 0 y 6. Dada esta situación, se evaluó conveniente aplicar escalado a estos valores utilizando la clase "MinMaxScaler" en el intervalo 0 y 1.

Particionamiento

Con el propósito de realizar el entrenamiento y las pruebas del modelo, se decidió particionar el conjunto de datos en dos, con una proporción de 80% y 20% respectivamente. Por otra parte, los datos se presentan en cierto orden, generalmente vinculado con el año de ingreso. Como esto no es deseable, se consideró conveniente mezclarlos aleatoriamente antes de usarlos. El método "train_test_split" de scikit-learn es útil para este fin, ya que permite realizar la partición de manera sencilla, con mezcla aleatoria y a la vez con la posibilidad de especificar una semilla para poder hacer repetible el experimento. En la sección siguiente se describe el uso de la técnica de validación cruzada sobre el conjunto de datos de entrenamiento.

Algoritmo de aprendizaje

Descripción del algoritmo

El tipo de aprendizaje considerado para el modelo es de tipo supervisado, ya que cada ejemplo en el conjunto de entrenamiento y prueba puede ser etiquetado como *abandona o no abandona*. Al tratarse de un problema de clasificación binaria el algoritmo de *regresión logística* se consideró apropiado para el modelo. La *regresión logística* se utiliza normalmente para estimar la probabilidad de que una instancia de un conjunto de datos pertenece o no a una clase particular, si la probabilidad es mayor al 50% entonces el modelo predice que dicha instancia corresponde a la clase, caso contrario no corresponde. De acuerdo con (Shalev-Shwartz & Ben-David, 2013) esta probabilidad viene dada por la función de hipótesis:

$$h_w(x) = \frac{1}{1 + \exp(-\langle w, x \rangle)}$$

en tanto que propone la función de pérdida que se debe minimizar como:

$$\underset{w \epsilon \Re^d}{\operatorname{argmin}} \frac{1}{m} \sum_{i=1}^{m} \log(1 + \exp(-y_i \langle w, x \rangle))$$

La aplicación de este algoritmo se efectuó mediante la clase "LogisticRegression" de scikit-learn, la que posee algunos parámetros importantes a considerar:

- penalty: establece la norma aplicada la función de costo ('l1', 'l2', 'elasticnet', 'none').
- C: parámetro de regularización que se comporta de manera inversa a $\boldsymbol{\lambda}$, es decir a menor valor de C mayor regularización y viceversa.
- solver: corresponde al algoritmo utilizado para la optimización, las posibilidades son ('newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga').

De acuerdo con la documentación de scikit-learn la función de costo completa con penalidad l2 es la siguiente:

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log\left(\exp(-y_i (X_i^T w + c)) + 1\right)$$

Como se puede apreciar el segundo término se asemeja a la función presentada anteriormente, aunque en este caso se agrega el factor C (que generalmente suele acompañar a la norma), mientras que el primer término es la norma del vector de pesos.

Métricas de desempeño

Para determinar si un modelo es confiable en cuanto a sus predicciones, es necesario recurrir a las métricas de desempeño. Una de las más importantes es la matriz de confusión (Tabla 2), ya que por un lado permite analizar con cierto detalle los resultados arrojados por el modelo, y al mismo tiempo sirve como base para el cálculo de otras métricas más sintéticas. En el caso de una clasificación binaria la matriz de confusión contiene: la cantidad de predicciones correctas positivas (VP), las correctas negativas (VN), las incorrectas positivas (FP) y las incorrectas negativas (FN).

Tabla 2 – Matriz de confusión

		Predicción	
		Positiva	Negativa
Real	Positiva	VP	FN
	Negativa	FP	VN

Para poder medir el desempeño se utilizaron las siguientes métricas

- Exactitud (Accuracy): es la razón entre la cantidad de predicciones correctas y el total de predicciones y es una medida general de la confiabilidad del modelo.

$$Acc = \frac{VP + VN}{VP + VN + FP + FN}$$

- Precisión: es la razón entre la cantidad de predicciones positivas correctas y el total de predicciones positivas (correctas e incorrectas). Es decir, refleja la tasa de positivos detectados correctamente.

$$Prec = \frac{VP}{VP + FP}$$

- Recall: es la razón entre la cantidad de predicciones positivas correctas y el total de predicciones positivas correctas y negativas incorrectas (las que deberían haber sido positivas). Es decir refleja la tasa de positivos detectados en relación a los positivos no detectados, por ello también se lo llama sensibilidad.

$$Recall = \frac{VP}{VP + FN}$$

Evaluación del modelo

Primeramente, se experimentó con la clase provista, ejecutando el método “fit” con los parámetros por defecto y posteriormente se procedió a probar su rendimiento, variando los parámetros “penalty” y “solver”, dejando constante C=1.

Tabla 3 - Variación de las iteraciones y la exactitud en función de penalty y solver

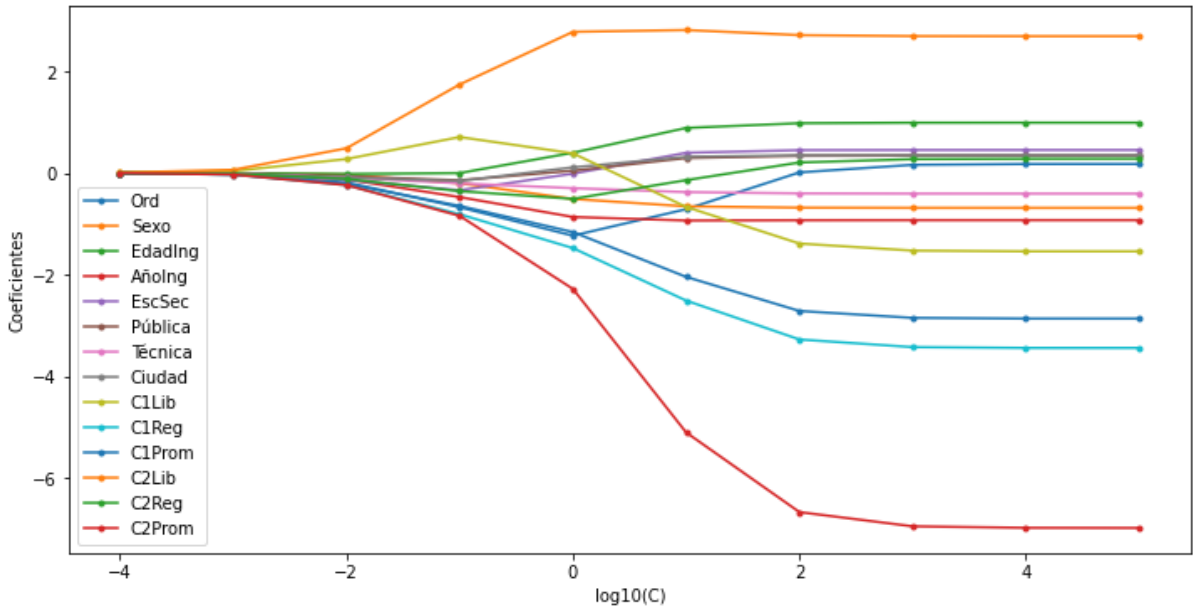
Penalty	Solver	Iteraciones	Exactitud
l1	liblinear	15-20	0.874
l2	liblinear	3-7	0.874
l2	lbfgs	27	0.874

Tabla 4 - Variación de los pesos en función de penalty y solver

w0	w1	w2	w3	w4	w5	w6	w7	w8	w9	w10	w11	w12	w13
-0.65	-0.42	0	-0.9	0	0	-0.25	0	0	-1.77	-0.96	2.78	0	-4.57
-1.24	-0.51	0.4	-0.87	0.02	0.04	-0.3	0.11	0.38	-1.48	-1.16	2.77	-0.51	-2.28
-2.11	-0.51	0.55	-0.74	0.24	0.22	-0.3	0.24	0.74	1.13	-0.93	2.86	-0.39	-2.13

El desempeño en cuanto a la exactitud fue idéntico en las tres ejecuciones, pero la combinación de “l2” y “liblinear” requirió menor cantidad de iteraciones, por ello en una segunda etapa se procedió a probar el impacto de la variación del parámetro C en el desempeño del algoritmo. Para esto se generaron 10 valores de C con un incremento exponencial (10x con x ∈ [-4,5]), con cada uno de estos se entrenó el modelo y se realizaron las predicciones usando el conjunto de prueba (Figura 3).

Figura 3 - Efecto de la variación de C en los pesos



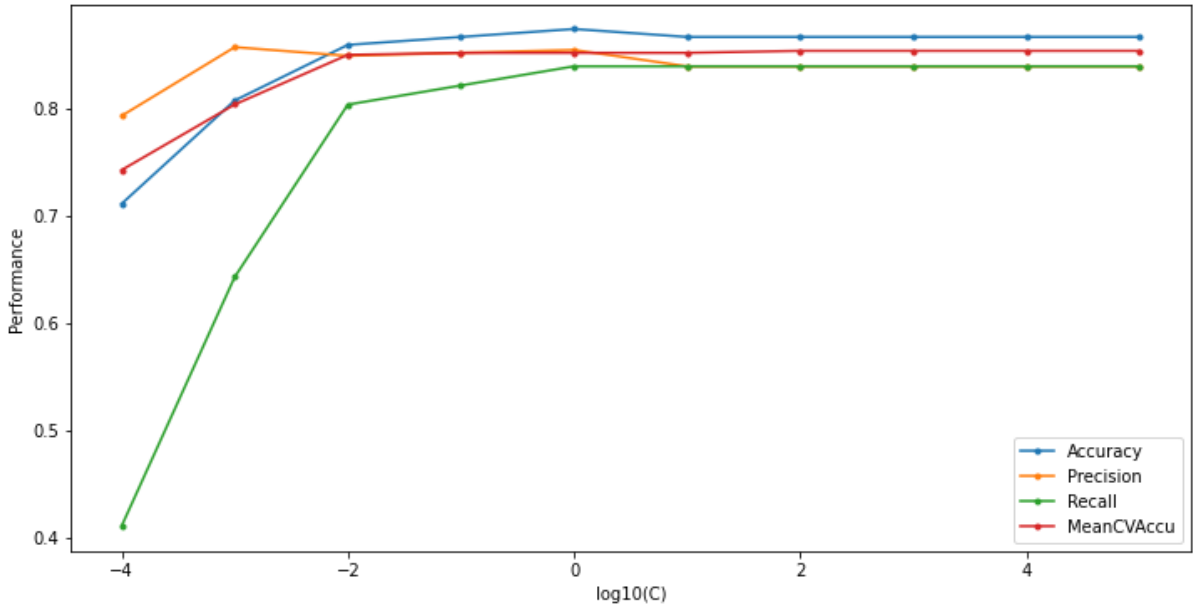
En primer lugar, se decidió observar cómo la variación del parámetro de regularización C afectaba los pesos vinculados a las características y a la ordenada al origen. En la Figura 3 y en la Tabla 5 se observa que a medida que los valores de C disminuyen, la fuerza de la regularización aumenta, con lo que se verifica que, en este caso C se comporta de manera inversa a λ y aproximadamente, con valores C superiores a 100, los pesos se estabilizan.

Tabla 5 - Efecto de la variación de C en los pesos

C	1E-4	1E-3	1E-2	1E-1	1E+0	1E+1	1E+2	1E+3	1E+4	1E+5
w0	-3,4E-3	-3,0E-2	-1,9E-1	-6,8E-1	-1,2E+0	-7,1E-1	9,3E-3	1,6E-1	1,7E-1	1,8E-1
w1	-4,9E-4	-4,5E-3	-3,5E-2	-2,0E-1	-5,1E-1	-6,6E-1	-6,8E-1	-6,9E-1	-6,9E-1	-6,9E-1
w2	-5,5E-4	-4,5E-3	-2,1E-2	-3,8E-3	4,0E-1	8,8E-1	9,8E-1	9,9E-1	9,9E-1	9,9E-1
w3	-2,2E-3	-2,0E-2	-1,3E-1	-4,8E-1	-8,7E-1	-9,4E-1	-9,3E-1	-9,3E-1	-9,3E-1	-9,3E-1
w4	-2,4E-3	-2,1E-2	-1,4E-1	-3,4E-1	-1,6E-2	4,0E-1	4,5E-1	4,5E-1	4,5E-1	4,5E-1
w5	-1,7E-3	-1,4E-2	-6,7E-2	-1,5E-1	4,3E-2	2,9E-1	3,3E-1	3,4E-1	3,4E-1	3,4E-1
w6	-1,7E-3	-1,5E-2	-9,2E-2	-2,1E-1	-3,0E-1	-3,8E-1	-4,0E-1	-4,1E-1	-4,1E-1	-4,1E-1
w7	-2,0E-3	-1,7E-2	-9,1E-2	-1,6E-1	1,1E-1	3,2E-1	3,5E-1	3,5E-1	3,5E-1	3,5E-1
w8	3,4E-3	3,5E-2	2,7E-1	7,0E-1	3,8E-1	-6,7E-1	-1,4E+0	-1,5E+0	-1,5E+0	-1,5E+0
w9	-3,4E-3	-3,2E-2	-2,4E-1	-8,1E-1	-1,5E+0	-2,5E+0	-3,3E+0	-3,4E+0	-3,4E+0	-3,4E+0
w10	-3,2E-3	-3,1E-2	-2,1E-1	-6,5E-1	-1,2E+0	-2,0E+0	-2,7E+0	-2,9E+0	-2,9E+0	-2,9E+0
w11	6,0E-3	6,0E-2	4,8E-1	1,7E+0	2,8E+0	2,8E+0	2,7E+0	2,7E+0	2,7E+0	2,7E+0
w12	-1,8E-3	-1,7E-2	-1,2E-1	-3,6E-1	-5,1E-1	-1,4E-1	2,0E-1	2,7E-1	2,8E-1	2,8E-1
w13	-3,5E-3	-3,3E-2	-2,4E-1	-8,5E-1	-2,3E+0	-5,1E+0	-6,7E+0	-7,0E+0	-7,0E+0	-7,0E+0

Para cada valor de C también se calcularon tres métricas de performance sobre los valores de prueba, estas pueden obtenerse a través de la matriz de confusión, por lo que se utilizó el método “confusion_matrix” de scikit-learn, el cual recibe el vector de salidas esperadas y las predichas.

Figura 4 - Efecto de la variación de C en las métricas de performance



Como el conjunto de datos es relativamente pequeño también se decidió probar la técnica de validación cruzada en cada iteración, utilizando “cross_val_score” y calcular la media de todas las exactitudes

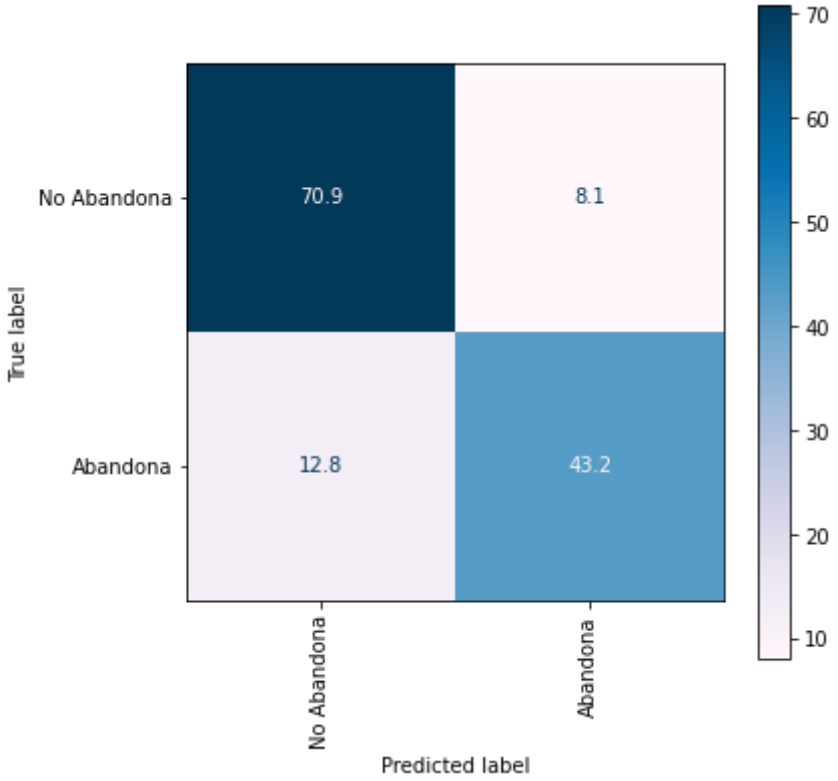
retornadas, el resultado se puede visualizar en la Figura 4 y la Tabla 6. En este caso el argumento de entrada es el conjunto de datos de entrenamiento que se particionó previamente.

Tabla 6 - Efecto de la variación de C en las métricas de performance

C	1E-4	1E-3	1E-2	1E-1	1E+0	1E+1	1E+2	1E+3	1E+4	1E+5
Exactitud	0,711	0,807	0,859	0,867	0,874	0,867	0,867	0,867	0,867	0,867
Precisión	0,793	0,857	0,849	0,852	0,855	0,839	0,839	0,839	0,839	0,839
Recall	0,411	0,643	0,804	0,821	0,839	0,839	0,839	0,839	0,839	0,839
Exactitud Media CV	0,743	0,804	0,850	0,852	0,852	0,852	0,854	0,854	0,854	0,854

Finalmente, en la Figura 5 se muestra la matriz de confusión que representa el resultado de obtener el promedio de todas las calculadas.

Figura 5 - Matriz de confusión promedio



Conclusión

En el trabajo presentado se describió el proceso de elaboración de un modelo para la predicción de la deserción estudiantil en la Tecnicatura Universitaria en Programación de la UTN - FRRe. Luego del planteo de la tarea a realizar (clasificación binaria), el relevamiento preliminar de la literatura y del conjunto de datos disponible, se optó por aplicar un algoritmo de regresión logística. El preprocesamiento de los datos crudos fue demandante y decisivo para definir el vector de características, debido a inconsistencias encontradas que requirieron modificar la propuesta inicial.

Con el conjunto de datos definitivo, se probaron varias configuraciones para el entrenamiento utilizando la clase “LogisticRegression” de scikit-learn, donde la penalización “l2” y el optimizador “liblinear” resultaron levemente más eficientes. Posteriormente, se comprobó el efecto del parámetro de regularización C sobre los pesos, con valores C>=100 estos se estabilizaban, entregando una confiabilidad razonable para la predicción de ejemplos nuevos (Exactitud=0.87; Precisión=0.84; Recall=0.84).
Como trabajo futuro, pretendemos modelar soluciones con redes neuronales y máquinas de vector soporte, por citar algunos, para evaluar el rendimiento de predicción contra el presentado en este informe con el fin de determinar el mejor modelo para implementar en la solución definitiva en el marco de este proyecto.

Referencias

Aulck, L., Velagapudi, N., Blumenstock, J., & West, J. (2016). *Predicting Student Dropout in Higher Education*. http://arxiv.org/abs/1606.06364

Berens, J., Schneider, K., Görtz, S., Oster, S., & Burghoff, J. (2018). Early Detection of Students at Risk – Predicting Student Dropouts Using Administrative Student Data and Machine Learning Methods. *CESifo Working Paper*, *7259*.

Chai, K. E. K., & Gibson, D. (2015). Predicting the risk of attrition for undergraduate students with time based modelling. *Proceedings of the 12th International Conference on Cognition and Exploratory Learning in the Digital Age, CELDA 2015*, *Celda*, 109–116.

Del Bonifro, F., Gabbrielli, M., Lisanti, G., & Zingaro, S. P. (2020). Student dropout prediction. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 12163 LNAI*. Springer International Publishing. https://doi.org/10.1007/978-3-030-52237-7_11

Haiyang, L., Wang, Z., Benachour, P., & Tubman, P. (2018). A time series classification method for behaviour-based dropout prediction. *Proceedings - IEEE 18th International Conference on Advanced Learning Technologies, ICALT 2018*, 191–195. https://doi.org/10.1109/ICALT.2018.00052

Kotsiantis, S. B., Pierrakeas, C. J., & Pintelas, P. E. (2003). Preventing student dropout in distance learning using machine learning techniques. *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, *2774 PART 2*, 267–274. https://doi.org/10.1007/978-3-540-45226-3_37

scikit-Learn (Ed.). (n.d.). Logistic regression. scikit. https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression. Último acceso 28/08/2021

Shalev-Shwartz, S., & Ben-David, S. (2013). Understanding machine learning: From theory to algorithms. (Vol. 9781107057135). https://doi.org/10.1017/CBO9781107298019

Solis, M., Moreira, T., Gonzalez, R., Fernandez, T., & Hernandez, M. (2018). Perspectives to Predict Dropout in University Students with Machine Learning. *2018 IEEE International Work Conference on Bioinspired Intelligence, IWOBI 2018 - Proceedings*, *August*. https://doi.org/10.1109/IWOBI.2018.8464191

Tang, C., Ouyang, Y., Rong, W., Zhang, J., & Xiong, Z. (2018). Time series model for predicting dropout in massive open online courses. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 10948 LNAI*. Springer International Publishing. https://doi.org/10.1007/978-3-319-93846-2_66

Wang, W., Yu, H., & Miao, C. (2017). Deep model for dropout prediction in MOOCs. *ACM International Conference Proceeding Series*, *Part F130655*, 26–32. https://doi.org/10.1145/3126973.3126990