UNIVERSIDAD PERUANA UNIÓN

FACULTAD DE INGENIERÍA Y ARQUITECTURA

Escuela Profesional Ingeniería de Sistemas

Modelo de aprendizaje supervisado para pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión - Lima

Tesis para obtener el Título Profesional de Ingeniero de Sistemas

Por

Bach. Anthony Jose Baldoceda Ramírez

Bach. Hector Armando Mamani Ccallohuari

Asesor:

Mg. Nemias Saboya Ríos

Lima, 2020

DECLARACIÓN JURADA DE AUTORÍA DE TESIS

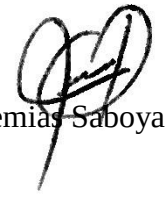
Nemias Saboya Rios, de la Facultad de Ingeniería y Arquitectura, Escuela Profesional de Ingeniería de Sistemas, de la Universidad Peruana Unión.

DECLARO:

Que la presente investigación titulada: **"Modelo de aprendizaje supervisado para pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión - Lima"** constituye la memoria que presenta el (la) Bachiller Anthony Jose Baldoceda Ramirez y Hector Armando Mamani Ccallohuari para aspirar al título de Profesional de Ingeniero de Sistemas, cuya tesis ha sido realizada en la Universidad Peruana Unión bajo mi dirección.

Las opiniones y declaraciones en este informe son de entera responsabilidad del autor, sin comprometer a la institución.

Y estando de acuerdo, firmo la presente declaración en la ciudad de Lima, a los 20 días del mes de enero del año 2021.



Mg. Nemias Saboya Rios

ACTA DE SUSTENTACIÓN DE TESIS

En Lima, Ñaña, Villa Unión, a los **09** días día(s) del mes de **octubre** del año 2020 siendo las **09:30 horas**, se reunieron en modalidad virtual u online sincrónica, bajo la dirección del Señor Presidente del jurado: **Dra. Erika Inés Acuña Salinas**, el secretario: **Mg. Geraldine Verónica Alvizuri Llerena**... y los demás miembros: **MSc. Fredy Abel Huanca Torres y el Mg. Herminio Paucar Curasma**..., y el asesor **Mg. Nemias Saboya Rios**, con el propósito de administrar el acto académico de sustentación de la tesis titulada: "Modelo de aprendizaje supervisados para pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima"

.....de el(los)/la(las) bachiller/es: a)..... **ANTHONY JOSE BALDOCEDA RAMIREZ**.....

.....b) .....**HECTOR ARMANDO MAMANI CCALLOHUARI** .....

.....conducente a la obtención del título profesional de

.....**INGENIERO DE SISTEMAS**.....
(Nombre del Título Profesional)

con mención en.....

El Presidente inició el acto académico de sustentación invitando ...a los ... candidato(a)/s hacer uso del tiempo determinado para su exposición. Concluida la exposición, el Presidente invitó a los demás miembros del jurado a efectuar las preguntas, y aclaraciones pertinentes, las cuales fueron absueltas por ... los ... candidato(a)/s. Luego, se produjo un receso para las deliberaciones y la emisión del dictamen del jurado.

Posteriormente, el jurado procedió a dejar constancia escrita sobre la evaluación en la presente acta, con el dictamen siguiente:

Candidato (a): ..... **ANTHONY JOSE BALDOCEDA RAMIREZ**
.....

CALIFICACIÓN	ESCALAS			Mérito
	Vigesimal	Literal	Cualitativa	
Aprobado	**17**	**B+**	**Con nominación muy bueno**	Sobresaliente

Candidato (b): ..... **HECTOR ARMANDO MAMANI CCALLOHUARI** .....

CALIFICACIÓN	ESCALAS			Mérito
	Vigesimal	Literal	Cualitativa	
Aprobado	**17**	**B+**	**Con nominación muy bueno**	Sobresaliente

(*) Ver parte posterior

Finalmente, el Presidente del jurado invitó ... a los ... candidato(a)/s a ponerse de pie, para recibir la evaluación final y concluir el acto académico de sustentación procediéndose a registrar las firmas respectivas.

Presidente Dra. Erika Inés Acuña Salinas		Secretario Mg. Geraldine Veronica Alvizuri Llerena
Asesor Mg. Nemias Saboya Rios	Miembro MSc. Fredy Abel Huanca Torres	Miembro Mg. Herminio Paucar Curasma
Candidato/a (a) Jose Baldoceda Ramirez		Candidato/a (b) Hector Armando Mamani Ccallohuari

DEDICATORIA

Dedico este trabajo de investigación a Dios y a mi familia. En especial a mis padres y mi hermana por su compañía y bendición divina en cada momento.

Anthony Baldoceda Ramirez

Este trabajo se la dedico principalmente a Dios por su compañía y bendición divina; y a mi familia por ser de gran ayuda en mi vida profesional.

Hector Mamani Ccallhuari

AGRADECIMIENTOS

Damos gracias a Dios por protegernos durante nuestro camino y darnos fuerzas para superar obstáculos y adversidades.

A nuestros amados padres: Jose Baldoceda, Blanca Ramirez, Hector Mamani, Elena Ccallohuari, por todo el ejemplo que nos ha inculcado y por su apoyo incondicional para cumplir nuestros objetivos con esfuerzo y dedicación.

A nuestro asesor de tesis**, Mg. Nemias Saboya Rios** por brindarnos su conocimiento y guiarnos en la investigación realizada con disposición, confianza y apoyo profesional.

A la Facultad de Ingeniería y Arquitectura, y a la Direccion General de Investigacion de la Universidad Peruana Unión – Lima por brindarnos la información solicitada para poder realizar este proyecto investigación.

A nuestros amigos, compañeros y otras personas que colaboraron con nosotros a lo largo de este proyecto y ampliaron el panorama del tema de investigación.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

SIMBOLOS USADOS

- IA: Inteligencia Artificial
- SVM: Abreviatura en ingles de Support Vector Machines
- CV: Abreviatura en ingles de Cross Validation
- CRISP-DM: Cross Industry Standard Process for Data Mining
- SINEACE: Sistema Nacional de Evaluación, Acreditación y Certificación de la Calidad Educativa
- SUNEDU: Superintendencia Nacional de Educacion Superior Universitaria
- UPEU: Universidad Peruana Union
- UNTELS: Universidad Nacional Tecnológica de Lima Sur
- OCR: Reconocimiento óptico de caracteres
- KNN O K-NN: K-Nearest Neighbors
- CWI : Centrun Wiskunde & Informatica
- XGBOOST: Extra Gradient boosting
- AUC: Area Under Curve
- ROC: Receiver Operating Characteristic
- MSE: Mean Squared Error

RESUMEN

La presente investigación tiene como principal objetivo determinar el nivel de eficacia del modelo de aprendizaje supervisado para el pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima. El estudio fue de tipo aplicada y se utilizó el método de CRISP-DM para el desarrollo del modelo.

Los datos fueron extraídos del sistema académico de la Universidad Peruana Unión considerando el período 2009-2019. Estos fueron clasificados en factores personales, financieros y académicos, siendo un total de 3161 registros. La metodología del estudio contó con 6 etapas: Comprensión del negocio, comprensión de los datos, preparación de los datos, diseño del modelo, evaluación del modelo y la implementación del modelo. A partir del análisis de las 16 variables iniciales y la transformación realizada, se obtuvo un diccionario de datos con 26 variables. Con este diccionario de datos se procedió a la identificación de las técnicas de modelos de aprendizaje para cada carrera, del cual se obtuvo que Decision Tree, Naive Bayes, KNN y Random Forest, fueron los que se adaptaron mejor a la realidad de cada carrera. Al evaluar estos modelos con las métricas “ratio de verdaderos positivos” (TPR) y “balanced accuracy”, se obtuvo que el modelo eficaz para cada carrera fue: Ing. Sistemas (Random Forest), Ing. Civil (Decision Tree), Ing. Alimentos (KNN), Ing. Ambiental (KNN) y Arquitectura (KNN). Finalmente, estos modelos fueron implementados en un API REST, en el cual se demostró su funcionamiento para realizar futuras predicciones de deserción estudiantes, sin necesidad de volver a entrenar el modelo.

Palabras clave:

Machine Learning, Modelo Predictivo, Desercion estudiantil, XGBOOST, árbol de clasificación.

ABSTRACT

The main objective of this research is to determine the level of effectiveness of the supervised learning model for the prediction of students dropping out of the Faculty of Engineering and Architecture of the Universidad Peruana Unión - Lima. The study was applied and the CRISP-DM method was used to develop the model.

The data were extracted from the academic system of the Universidad Peruana Unión considering the period 2009-2019. These were classified into personal, financial and academic factors, for a total of 3161 records. The study methodology had 6 stages: understanding the business, understanding the data, preparing the data, designing the model, evaluating the model and deploying the model. From the analysis of the 16 initial variables and the transformation carried out, obtained a data dictionary with 26 variables. With this data dictionary, we proceeded to identify the learning model techniques for each race, from which it was obtained that Decision Tree, Naive Bayes, KNN and Random Forest, were the ones that best adapted to the reality of each race. When evaluating these models with the metrics "ratio of true positives" (TPR) and "balanced accuracy", it was obtained that the effective model for each race was: Ing. Sistemas (random forest), Ing. Civil (decision tree), Ing. Food (KNN), Environmental Engineer (KNN) and Architecture (KNN). Finally, these models were implemented in an API REST, in which their operation was demonstrated to make future predictions of student dropout, without the need to retrain the model.

Keyword: Machine Learning, Predictive Model, Student Dropout, XGBOOST, Data Analysis

CAPÍTULO I. El Problema

1.1. Identificación del problema

El análisis de datos ha ido ocupado un lugar muy importante en la toma de decisiones en los últimos tiempos. Hoy en día existen muchas herramientas que facilitan la explotación de los datos con el objetivo de buscar patrones que ayuden o faciliten a la solución de problemas a diferentes campos de la sociedad usando inteligencia artificial, minería de datos y análisis predictivo. La oportunidad que brinda la tecnología e innovación en el análisis de datos, ofrece al mundo nuevos productos, soluciones e innovaciones para cambiar la vida de las instituciones [1].

El modelo predictivo como el proceso mediante el cual un modelo se crea o se elige para tratar de predecir mejor la probabilidad de un resultado [2]. Las técnicas empleadas en estos modelos, hacen uso de procesos desde diferentes perspectivas, apoyados en herramientas y estrategias con el único fin de realizar una precisa predicción.

Los modelos predictivos son usados en diferentes rubros empresariales: pesqueros, financiero, salud, agrónomo, minero, y también en el sector educativo; todos con el propósito de minimizar errores de decisión y anticipar soluciones eficientes a los problemas que se pueden presentar en cada área. La deserción universitaria a nivel mundial es un problema latente, por ejemplo, en Estados Unidos, la tasa de deserción de estudiantes universitarios es más del 50%, en Hungría y Nueva Zelanda más del 40% y en Reino Unido más del 30%. Así también en Latinoamérica se evidencia que, en Bolivia, Nicaragua y Colombia, se presentan las tasas de deserción más altas, ya que alrededor del 36% de todas las personas que dejan de estudiar en Colombia, lo hacen al final de su primer año [3].

En una universidad de China, se probó la eficacia de 3 modelos para reconocer a futuro la deserción de sus estudiantes en sus cursos online: árboles de decisión, redes neuronales y redes bayesianas, de los cuales se demostró que todos fueron relativamente efectivos, pero que el mejor fue árboles de decisión [4].

En República Dominicana también se realizó un modelo a través de técnicas de minerías de datos para predecir el riesgo de deserción de los estudiantes del nivel básico y medio en su sistema educativo a 72 centros, recolectando el historial académico de los alumnos junto a sus factores

socioeconómicos y ambientales comprendidos entre el 2009 y 2014. Las variables que más se relacionaron fueron sus edades con el nivel de grado alcanzado. Por consiguiente, el autor realizó el modelo con algoritmos de árboles de decisión el cual obtuvo una precisión del 82.69% y una eficacia del 82.62% en la predicción de los estudiantes que desertaron [5].

Los principales factores en la deserción universitaria son personales, académicos, socioeconómicos e institucionales, siendo el caso de una universidad colombiana en la que el mayor factor por la cual sus alumnos desertan es el factor socioeconómico (42.5%) seguido por el factor personal (39.2%) [7]. Además, ello, existen autores que consideran que los factores psicológico y sociológico también ayudan a predecir la decersión de los estudiantes [6].

En el ámbito nacional, la deserción universitaria alcanza una tasa del 30% del problema en educación [8]. El Ministerio de Educación del Perú realiza programas como "Beca Doble Oportunidad" que ofrece financiar los dos últimos años escolares y una carrera técnica, así también el "Programa Beca 18" que ofrece financiamiento en carreras universitarias, ayuda a disminuir la deserción en las instituciones educativas. En el Perú, la preocupación por asegurar la calidad de la educación universitaria, se creó el Sistema Nacional de Evaluación, Acreditación y Certificación de la Calidad Educativa (SINEACE) con el objetivo de que las instituciones educativas cumplan los requisitos de calidad. Se inició el proceso de licenciamiento de universidades por parte de la SUNEDU (Superintendencia Nacional de Educación Superior Universitaria) con el fin de garantizar la calidad de la educación universitaria.

Los problemas mencionados anteriormente también tienen lugar en la Universidad Peruana Unión (UPeU), universidad reconocida licenciada por la SUNEDU con 36 años de vida institucional que cuenta con 5 facultades académicas: Facultad de Ciencia Empresariales (FCE), Facultad de Teología (FACTEO), Facultad de Ciencias Humana y Educación (FACIHED), Facultad de Ciencias de la Salud (FCS) y Facultad de Ingeniería y Arquitectura (FIA) siendo esta última la que cuenta con la mayor cantidad de estudiantes. Cabe resaltar que esta universidad cuenta con un sistema de información académico en el cual se almacena los datos de todos los estudiantes desde el 2000 hasta la actualidad.

En la escuela de la Facultad de Ingeniería y Arquitectura, todos los años se trabajan proyectos a favor de la comunidad universitaria con la finalidad de formar al estudiante en diferentes aspectos

como: académico, social, espiritual. Dadas las necesidades de los estudiantes, la escuela pone sus mayores esfuerzos en la atención de las necesidades de los posibles desertores, sin embargo, aún se necesita mayor precisión en los factores que influyen en la identificación de los mismos.

Es complejo reconocer al futuro desertor debido a varios factores que no se pueden identificar a simple vista, de modo que, con la ayuda de la data histórica, la tecnología y el avance de la ciencia de los datos, es posible solucionar este problema a través de modelos que ayudan a identificar patrones de comportamiento que permiten cremodelos para predecir con mayor exactitud, los futuros estudiantes desertores.

1.2. Objetivos.

1.2.1. Objetivo general

Determinar el nivel de eficacia del modelo de aprendizaje supervisado para el pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima

1.2.2. Objetivos específicos

- Analizar los datos almacenados de los estudiantes del 2009-2019 de la Facultad de Ingeniería de la Universidad Peruana Unión en el sistema académico para la comprensión del entorno.

- Identificar las técnicas de los modelos de aprendizaje supervisados de Machine Learning que mas se adecuan para la solución del pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima.

- Validar el modelo de aprendizaje supervisado en función a los resultados obtenidos de las técnicas de Machine Learning para el pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima.

- Implementar el modelo predictivo validado para evidenciar los resultados del pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima.

1.3. Justificación de la investigación.

1.3.1. Justificación Teórica

La importancia de esta investigación reside en la identificación de patrones de deserción estudiantil y en la elaboración de su modelo predictivo a través de Machine Learning, en los estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión. De esta manera, los directores de escuela tendrán mayor facilidad para identificar a los potenciales desertores y de acuerdo a ello, tomar medidas en su gestión, para evitar este problema.

1.3.2. Justificación Metodológica

Este estudio presenta un aporte metodológico porque el modelo facilitará el trabajo de los administrativos de la facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión al contar con la utilización de las buenas prácticas de herramienta a través del método CRISP-DM. Esta es una metodología más detallada y actualizada, adaptada a enfoques más ágiles especialidad para Data Science con plantillas predeterminadas, siendo el resultado un modelo predictivo más factible en su implementación. Además de ello, esta metodología se desarrolla con el lenguaje de programación Python con el que no será necesario requerir servidores externos. Asimismo, la relevancia metodológica de la investigación radica en los modelos predictivos a través de la metodología Machine Learning, y la obtención de una data histórica limpia de estudiantes (2000-2017). Los modelos serán puestos a disposición de la comunidad científica y las instituciones adventistas comprometidas en el estudio para ser utilizadas libremente.

1.3.3. Justificación Práctica

El presente trabajo de investigación será de utilidad para la obtención de resultados oportunos en tiempo real, para la planificación y desarrollo de programas focalizados en acompañamiento y retención en la población universitaria acordes a las necesidades de la organización, de manera que se aumente del grado de fidelización de los estudiantes y estos puedan concluir sus estudios de pregrado con éxito.

CAPÍTULO II. Marco teórico

2.1. Revisión de la Literatura.

2.1.1. Ambito Internacional

En el 2015, Tan Mingjie dio a conocer en su trabajo de investigación "Prediction of Student Dropout in E-Learning Program through the use of Machine Learning Method", la alta tasa de deserción como problema grave en los programas de E-Learning, siendo este, una preocupación de los administradores e investigadores de educación. Este estudio seleccionó las características personales y el rendimiento académico de los estudiantes como atribuciones de entrada, después desarrolló los modelos de predicción usando Redes Neuronales Artificiales (ANN), árbol de decisión (DT) y Redes Bayesianas (BNS). Los resultados de los modelos de predicción usando DT fue un 71,91% , con BN fue de un 69,19% y con ANN fue de un 65,65% de precisión, siendo el Modelo DT el más eficaz y preciso en la predicción [1].

Por otro lado, M. Rodríguez, J. González y J. Patricio, investigaron acerca del "Modelo Predictivo para la permanencia en la educación superior" realizado el 2017 en Valparaíso de Chile, el cual tuvo como objetivo reportar el modelo multivariado predictivo para la permanencia universitaria a partir de atributos previos al ingreso a la universidad y de variables medidas durante el primer semestre de 2017 de los estudiantes de la Universidad de Playa Ancha. El estudio utilizó el diseño no experimental, transaccional y correlacional además trabajó con una muestra de 58 estudiantes. Para la recolección de información se utilizó el compromiso del estudiante, sistema académico e integración universitaria obteniendo un nuevo modelo con un coeficiente de determinación de 0.89, donde la variabilidad del Rendimiento Medio es explicada por los predictores NEM (0.08777), ASISTENCIA (0.42711) y APROB (-2.72077), todos significativos al 5%, sin sobreposición. Además se definieron nuevas regresiones, una con un nivel de explicación de 36% siendo significativo (p-valor =0.0004) y una tercera cuyo poder predictivo es del 75%, siendo altamente significativo (p-valor=0.000) [2].

El trabajo de investigación "Un Modelo Predictivo de Fracaso/Éxito Académico a partir de indicadores de ingreso, en estudiantes de una universidad estatal del norte de Chile" realizado por R. Ferrer, V. Karmelic, H. Beck y R. en 2019 , tuvo como objetivo desarrollar una propuesta empírica para predecir el fracaso/éxito académico en estudiantes universitarios a partir de indicadores de ingreso temprano, académicos y sociodemográficos, considerando el efecto

mediador de la tasa de aprobación de primer año. El estudio fue de diseño longitudinal, de cohortes y de alcance correlacional. Para la recolección de información, la universidad les proporciónó los datos de los matriculados en primer año del período 2001-2016, obteniendo como resultado que los predictores de entrada fueron mejores predictores proximales (i.e. tasas de aprobación en primer año) que distales (fracaso/éxito). Por lo tanto el mejor predictor de fracaso/éxito corresponde a la tasa de aprobación de primer año siendo las notas de enseñanza media el mejor predictor transversal de desempeño al momento del ingreso, excepto en ingeniería [3].

En el 2016, S. Merchán y J. Duarte en su trabajo de investigación denominado "Analysis of Data Mining Techniques for Constructing a Predictive Model for Academic Performance" realizado en Colombia, planteó una propuesta para construir un modelo predictivo para el rendimiento académico de los estudiantes. El desarrollo del modelo se realizó a través de sentencias SQL de la base de datos local con la metodología CRISP-DM, asimismo trabajó con una herramienta WEKA, esta herramienta le permitió construir el modelo con una muestra de 932 estudiantes. Los resultados fueron significativos ya que el algoritmo escogido para realizar esta tarea fue el árbol de decisión J48, por su buen desempeño promedio en las anteriores iteraciones y otras pruebas realizadas en los datos [4].

En su artículo de investigación de Lorenz Kemper llamado "Predicting Student Dropout: a Machine Learning Approach" realizado en Alemania en el 2018, tuvo como objetivo realizar dos modelos de aprendizaje automático: regresión logística y árboles de decisión para predecir el abandono escolar en el Instituto de Tecnología de Karlsruhe (KIT); el estudio utilizó un enfoque metódico y además trabajo con el rendimiento académico (notas de los exámenes) y las características personales de sus alumnos, obteniendo como resultado que arboles de decisión produjeron altas precisiones de predicción con un 95% de precisión [5]

2.1.2. Ambito Nacional

En el 2015, Myrna Caycho investigó acerca del "Modelo predictivo para la identificación de patrones de la deserción estudiantil en la Universidad Nacional Tecnológica de Lima Sur (Untels)", el cual tuvo como objetivo determinar los patrones del entorno que impactan en la deserción de los estudiantes. El investigador elaboró una base de datos socioeconómica y académica de los estudiantes de la cohorte 2007-I a 2011-I, que incluye los casos de deserción como variable dependiente. Por otro lado, elaboró seis veces el mismo modelo utilizando el

operador Decisión Tree de RapidMiner, con y sin validación cruzada, y con parámetros modificados, además de la implementación de la herramienta Weka, W-J48. La implementación de W-J48 con parámetros modificados y análisis de sensibilidad mediante proporción de ganancia de información y sistema de validación cruzada de 10 particiones, ofreció la precisión más alta, así como un árbol simple de uso y de interpretación. El modelo final detectó las siguientes características o patrones del entorno que impactan en la deserción de los estudiantes de la Untels: Número de matrículas en los cuatro semestres consecutivos a su ingreso, promedio de su segunda matrícula, edad de ingreso, promedio en su cuarta matrícula, año de ingreso, semestre de ingreso y número de cursos aprobados en su primera matrícula. Logró un 90.10% de clasificación correcta, con una desviación estándar de 2.08%. El principal patrón detectado para los desertores fue el número de matrículas que realiza el alumno en los cuatro semestres consecutivos a su ingreso sea menor o igual que 3 matrículas, con una precisión de 88% de deserción [6].

En su informe final de tesis denominada "Predicción de rendimiento académico mediante regresión y redes neuronales en los estudiantes de la escuela profesional de ingeniería estadística e informática de la Universidad Nacional del Altiplano – PUNO", realizado el 2017 por H. Paja , tuvo como objetivo determinar la mejor técnica de predicción en el rendimiento académico utilizando regresión y redes neuronales en estudiantes de la Facultad mencionada; el proyecto utilizó el diseño cuasi experimental con estrategia transversal y además trabajo con 696 estudiantes; por otro lado para la recolección de información se apoyó de registros, actas y material informático obteniendo como resultado que las RNA's es la mejor técnica en predicción; puesto que tienen una diferencia de 0.534 error en modelo y 0.1307 error en predicción respecto a la regresión [7].

La tesis de A. Jiménez llamada "Análisis Predictivo para los procesos de Admisión de la Universidad Nacional del Altiplano – Puno" sustentada en el año 2017, tuvo como objetivo analizar la información de postulantes e ingresantes para predecir la tendencia del futuro de las diferentes escuelas profesionales y la formación brindada en las escuelas de educación secundaria, públicas y privadas de la región de Puno. El proyecto utilizó la metodología CRISP-DM y además trabajó con 135836 estudiantes; por otro lado la información fue dada por la misma universidad obteniendo como resultado los modelos lineales y polinomios que permitieron predecir y

confirmar el nivel de crecimiento de las Escuelas Profesionales como Ing. Civil, Ciencias Contables y otros [8].

El investigador O. Sifuentes en su investigación denominada “Modelos predictivos de la deserción estudiantil en una universidad privada peruana” realizado en el 2018, tuvo como objetivo determinar la contribución del uso de modelos predictivos en asignaturas críticas para identificar a los estudiantes en riesgo de deserción. El estudio fue de diseño cuasi experimental de corte transversal y de tipo descriptivo probabilístico. Trabajó con la información histórica de 4478 estudiantes y obtuvo como resultado que los modelos predictivos contribuyeron a reducir en un 40% y 50% los niveles de desaprobación y las variables que mejor la predijeron fueron la carrera que estudian (vocación), el número de veces que se matriculan en la asignatura y la nota que tuvieron en matemática o comunicación cuando cursaron el quinto año del nivel secundaria [9].

2.2. Base Teorico

2.2.1. Machine Learning

El Machine Learning o aprendizaje automático tiene como propósito que las máquinas o computadoras aprendan automáticamente, es decir que la máquina pueda identificar patrones complejos dentro de una gran cantidad de datos obtenidos mediante ejemplos, experiencia o instrucciones predefinidas [18, p.10]. La idea de aprendizaje se basa en un algoritmo que revisa los datos y es capaz de predecir comportamientos futuros adaptándose a la incorporación de información adicional y recalibrando los resultados [10].

2.2.2. Beneficios que aporta el Machine Learning

Dentro de la Inteligencia Artificial, el Machine Learning se ha convertido en el método de elección para desarrollar software práctico con visión artificial, procesamiento de lenguaje natural, reconocimiento de voz y otras aplicaciones. Un ejemplo de ellos se encuentra en la Figura 1[11].

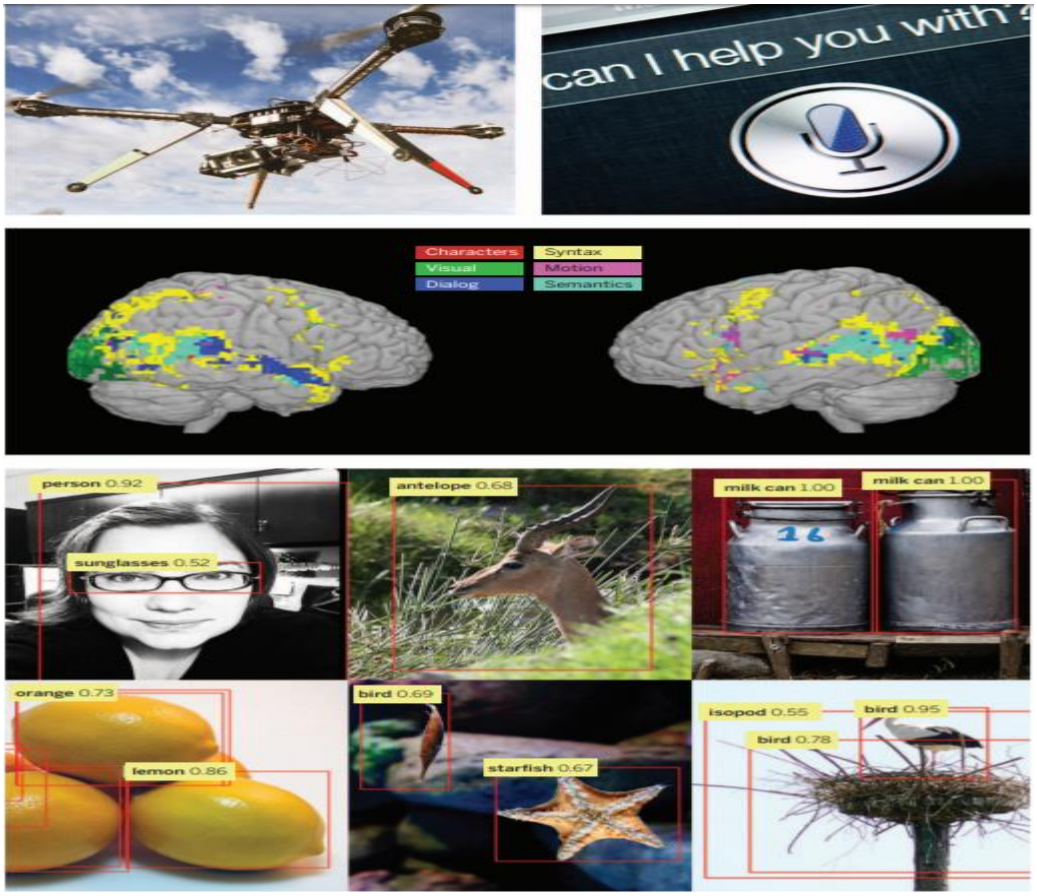


Figura 1. Aplicaciones del Aprendizaje Automático
Fuente: M. I. Jordan and T. M. Mitchell (2015). Machine learning: Trends, perspectives, and prospects [11]

Es posible no identificar al 100% el proceso, pero se puede construir una aproximación buena. Aún cuando se identifica el proceso completo, se puede detectar patrones o regularidades, siendo los patrones el nido del Machine Learning que ayuda a comprender el proceso, o para realizar predicciones suponiendo que el futuro sea cercano y no haya mucha la diferencia entre el pasado y las predicciones futuras [12].

Pero cuando hay un tiempo considerado desde que se construyó el modelo predictivo, es necesario considerar un nuevo análisis para evitar una mala información para la toma de decisiones. También para que un sistema sea inteligente debe tener la capacidad de aprender a tales cambios que el diseñador del sistema se beneficie al no necesitar prever soluciones a todos los

tipos de situaciones [12]. Machine learning ha sido implementado a lo largo de varios campos del comercio, medicina, bancos y otros.

El Machine Learning también ayuda en la Reducción de Dimensión Transformada, para la representación de los ítems inicial en una representación de baja dimensión, preservando las propiedades de la inicial representación. Un ejemplo de esto se encuentra en la reprocesamiento de imágenes digitales [13].

Otras tareas donde encontramos la aplicación de Machine Learning son el procesamiento de lenguaje natural, reconocimiento de sonido, reconocimiento de caracteres ópticos (OCR) y reconocimiento de rostro [14].

2.2.3. Clasificación del Machine Learning

Machine Learning puede ser ampliamente definida como métodos computacionales que usan la experiencia para mejorar el desempeño de las predicciones y hacerlas más precisas [15]. Cuando se refiere a experiencias, está relacionado directamente con la información histórica recolectada que se utiliza para los procesos de entrenamiento

El aprendizaje automático se ha ramificado en varios campos que se ocupan diferentes tipos de tareas incluyendo motores de búsquedas, diagnósticos médicos, análisis de mercado de valores y otros [16]. Machine Learning se clasifica en dos tipos aprendizaje: supervisado y el no supervisado.

2.2.3.1. *Aprendizaje Supervisado*

El aprendizaje automático supervisado es la búsqueda de algoritmos que razonan a partir de instancias proporcionadas externamente para producir hipótesis generales, que luego hacen predicciones sobre instancias futuras [17].

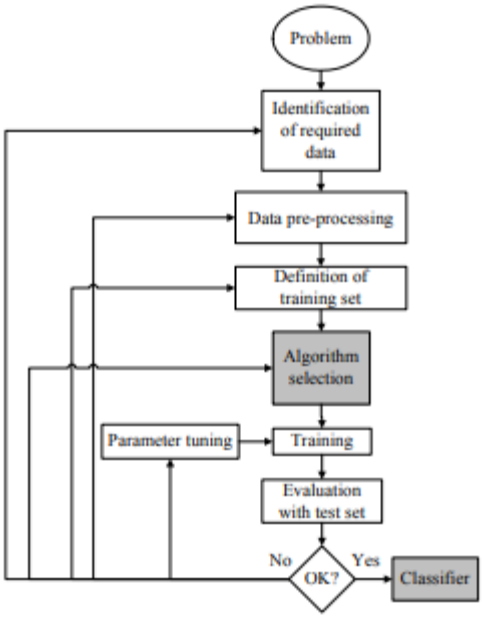


Figura 2. El proceso de Aprendizaje Supervisado en Machine Learning
Fuente: S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas (2006), Supervised Machine Learning: A Review of Classification Techniques [17].

En la Figura 2 se muestra el proceso de manera general. El aprendizaje supervisado tiene 2 clasificaciones, las cuales son:

Clasificación: En este caso la idea central de la aplicación de la técnica es identificar a qué clase pertenece una nueva entrada; tales ejemplos se aplican en la clasificación de documentos, imágenes, diagnóstico médico [18].

Regresión: Predice un valor real para cada ítem, por ejemplo la predicción de la demanda, stocks de inventarios, variables económicas, tasas y otros [19] .

2.2.3.1.1. *K-Nearest Neighbors*

La intuición subyacente en la clasificación del vecino más cercano es bastante sencilla, los ejemplos se clasifican según la clase de sus vecinos más cercanos. A menudo es útil tener en cuenta a más de un vecino, por lo que la técnica se denomina más comúnmente k-Vecino más cercano (k-NN) [20].

KNN usa el conjunto de datos de entrenamiento para realizar predicciones con el objetivo de obtener una nueva instancia(x), buscando las K instancias más similares. En regresión, es la

variable de salida promedio, pero en caso de clasificación es el valor más común para poder determinar el número de instancias K más similares a una nueva entrada.

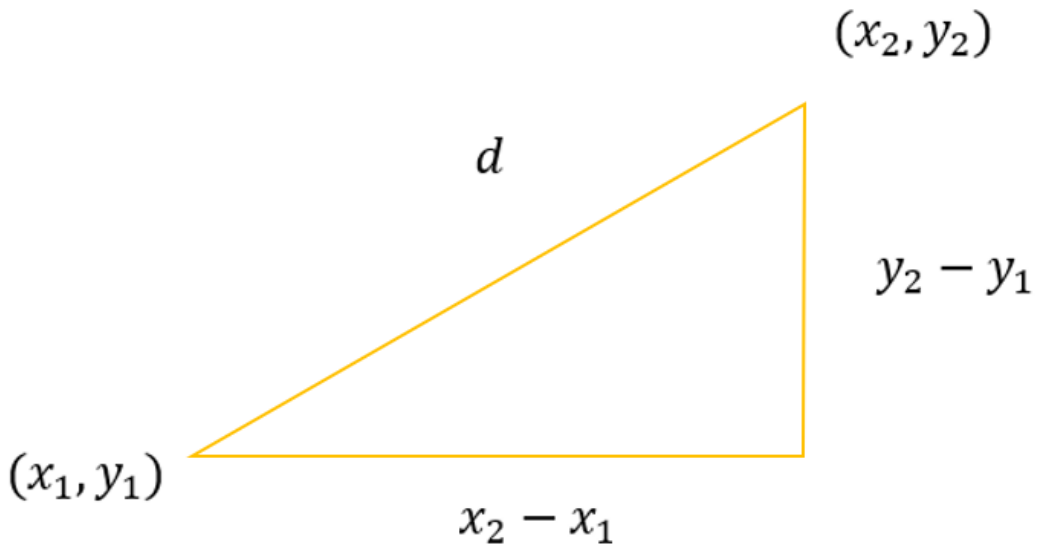


Figura 3. Distancia Euclidania
Fuente: Propia.

Distancia Euclidiana, Hamming distancia, Manhattan Distancia, y Minkowski Distancia, pero la que utilizaremos es la Distancia Euclidiana, siendo su formula:

$$DE = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Se recomienda que el número de instancias K no sea muy grande ni tampoco muy pequeño. Si es muy grande es posible que se tienda a ignorar los patrones más pequeños, y si es muy pequeño se genería mucha variabilidad en la elección. También se recomienda utilizar la técnica de normalización de los datos de (0 a 1) u otorgar una mayor ponderación a los datos para cambiar la estructura de los datos de forma adversa.

En la elección de este modelo hay que tomar en cuenta que no se hace ninguna suposición previa a los datos subyacentes. De todas maneras esto genera muchos errores por ser sensible a valores atípicos. Por ello, se tiene que poner mayor énfasis en la etapa de preparación con el objetivo de obtener un modelo sólido.

2.2.3.1.2. *Support Vector Machine*

Este nivel del procesamiento del lenguaje natural tiene el objetivo de buscar interpretar el significado de las palabras individuales, y para ello los sistemas de PLN se soportan de varios tipos de procesamiento, una de ellas es la llamada comúnmente part-of-speech tag o la asignación de etiquetas individuales a cada palabra del texto, "estas etiquetas determinan la categoría léxica de la palabra y en el caso de que una palabra pueda tener diferentes funciones dentro de la sentencia, será etiquetado con la categoría léxica más probable en función del contexto" [21, p. 23].

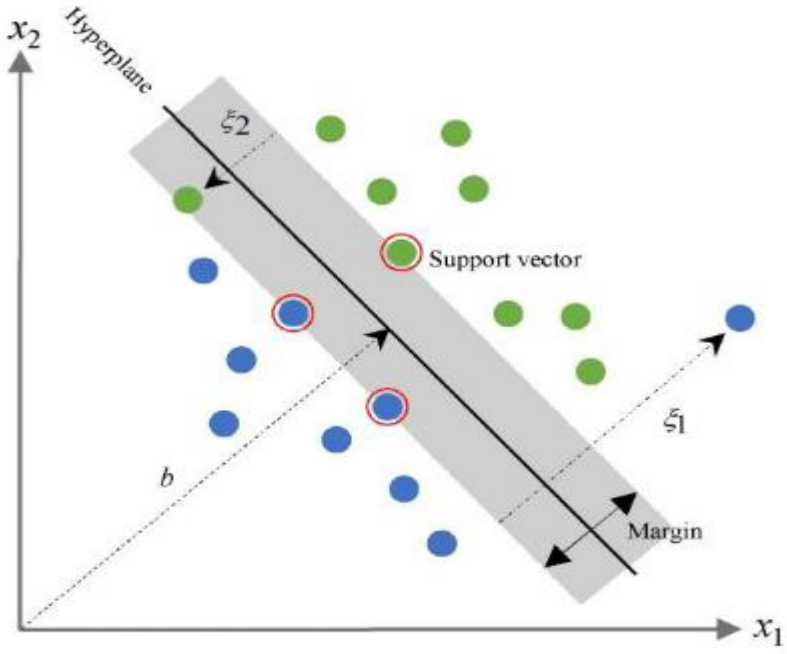


Figura 4. Hiperplano de Separacion Bidimensional
Fuente: M. A. Nanda, K. B. Seminar, D. Nandika, and A. Maddu (2018). A comparison study of kernel functions in the support vector machine and its application for termite detection. [22].

Los cuatro conceptos básicos son:

a) Hiperplano Separador: La definición matemática de un hiperplano es bastante simple. En el caso de dos dimensiones, el hiperplano se describe acorde a la ecuación de una recta:

$$\beta_0 + \beta_1(x^1) + \beta_2(x^2) = 0$$

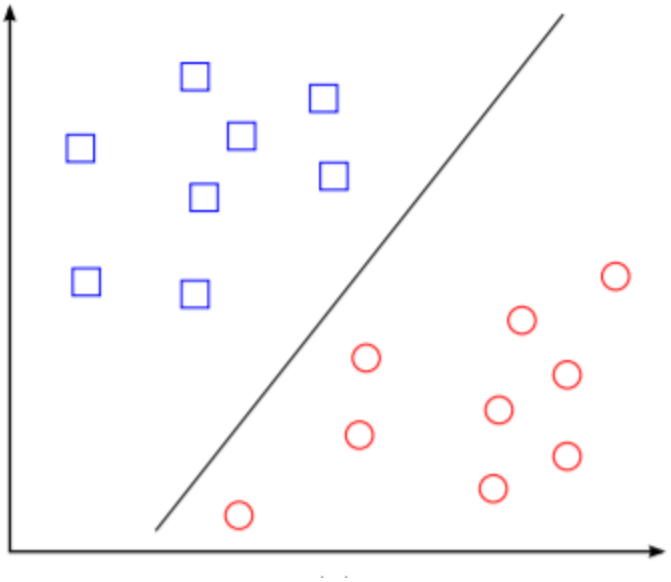


Figura 5. Hiperplano de Separador
Fuente: Abstract Support Vector Machine [23]

Según la figura 5 el hiperplano separador es el que divide los casos según su clase, por un lado están los casos de color azul y por otro lado los casos de color.

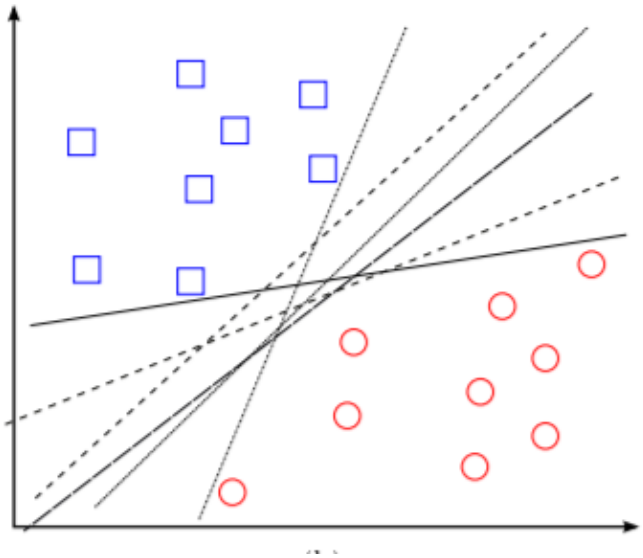


Figura 6. Hiperplano de Separacion Bidimensional entre los infinitos posibles
Fuente: Abstract Support Vector Machine [23]

En término general en un espacio de alta dimensión una línea recta es un hiperplano, siendo el hiperplano separador, pero el buscar el hiperplano separador resulta en más de un posible hiperplano como se puede observar la Figura 6, bajo esa necesidad existe el método hiperplano de margen máximo.

La definición de hiperplano para casos perfectamente separables linealmente resulta en un número infinito de posibles hiperplanos, lo que hace necesario un método que permita seleccionar uno de ellos como clasificador óptimo.

b) Hiperplano de Margen Máximo

El margen "m" maximiza la distancia mínima entre las observaciones y el hiperplano separador lo máximo posible. Como se observa en la Figura 4

c) Margen Suave

Conocido los vectores de soporte, se busca el margen que pueda separar las observaciones, algunas de estas, al ser tan similares es recomendable que esté dentro del margen.

d) Función del Núcleo (Kernel)

Una función del núcleo permite proyectar datos de un espacio de baja dimensión a un espacio de dimensión superior para poder hallar el hiperplano separador, sin perder sus características.

2.2.3.1.3. ***Naive Bayes***

Los clasificadores bayesianos asignan la clase más probable a un ejemplo dado descrito por su vector de características. El aprendizaje de estos clasificadores se puede simplificar en gran medida suponiendo que las características son independientes según la clase [24].

En la Fórmula del Teorema de Bayes $P(B|A) = \frac{P(A|B)*P(B)}{P(A)}$, A es lo que ha ocurrido y B la probabilidad de que ocurra. Siendo A y B la evidencia e hipótesis respectivamente. Este algoritmo se divide en los siguientes tipos:

a) Bayes Ingenuos Multinomiales

Utiliza tanto la probabilidad de bayes y la ley multinomial, siendo este último el que asume que las palabras en el documento tienen una distribución multinomial subyacente, en el cual, las palabras son independientes una de otras.

b) Bernoulli Naive Bayes

Siendo similar a Bayes ingenuos multinomiales, los predictores son booleanas. Refiriéndose a que solo existe dos opciones si la palabra aparece o no. En comparación a su simular que usa la frecuencia de términos. Usualmente este tipo se utiliza en la clasificación de documentos igual que su simular.

c) Gaussian Naive Bayes

Usualmente es utilizada cuando las variables independientes son valores continuos y no discretos y tienen una distribución normal. Realiza sus predicciones en base a que solo necesita estimar la media y la desviación estándar de los datos de entrenamiento.

2.2.3.1.4. *Decision Tree*

Los árboles de decisión, tradicionalmente conocido como análisis de decisión se usaron para ayudar a alcanzar un objetivo con mayor probabilidad.

En 1984 por primera vez, Breiman usó el término CART que significa “Classification And Regression Trees”

Las ventajas de utilizar Arboles de Decisión es su fácil interpretación y útil representación gráfica.

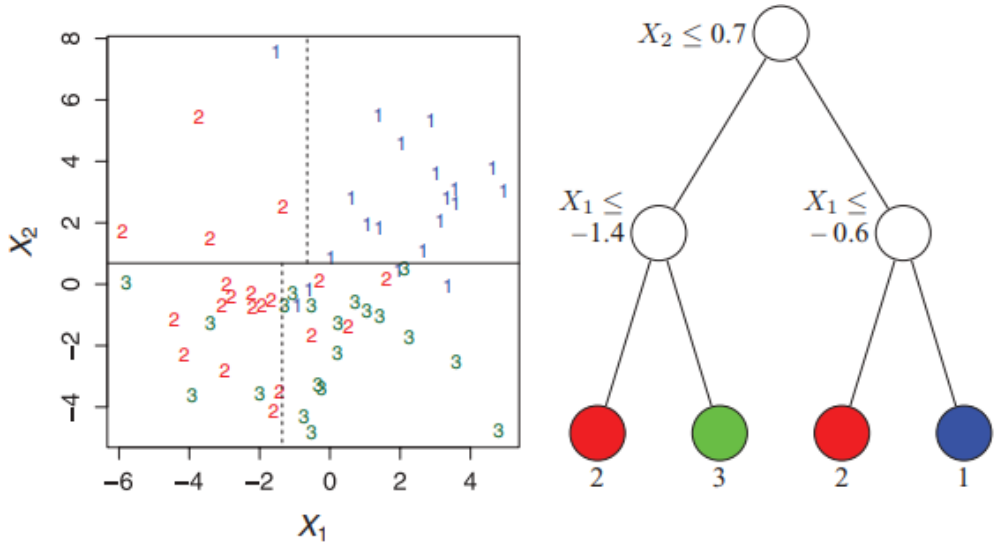


Figura 7. Arboles de Decision
Fuente: Classification and regression trees [25].

Según la Figura 7 en el lado izquierdo es encuentra las particiones junto con los puntos de datos y en la derecha la estructura de árbol de decisión.

La diferencia en la demostración de las condiciones es que en lado izquierdo esta limitado a 2 variables como máximo, en cambio a la derecha nos muestra la lógica de como esta distribuido las diferentes clases, pero estos algoritmos pueden sufrir el problema de equilibrio bias que se refiere a cuánto se alejan en promedio las predicciones respecto a los valores reales y de varianza se refiere que tanto puede afectar la muestra utilizada en el entrenamiento. Y lo recomendable es que se pueda conseguir un equilibrio entre la bias y varianza: Es por ello que se utilizan los métodos de ensemble como:

A) Boosting

Ajusta modelos secuenciales utilizando múltiples weak learners, en la que aprende los errores de modelos anteriores y mejora en cada iteración sin recurrir a muestreos repetitivos. Cabe resaltar que esta también contiene una cantidad considerable de hiperparámetros.

B) Bagging

Con el fin de reducir la varianza, se obtiene múltiples muestras de la población para ajustar un modelo distinto para cada uno, y realizar la media a variables cuantitativas o la moda en caso de las variables cualitativas.

Pero en vez de obtener múltiples muestras, se aplica bootstrapping para generar pseudo-muestras en los diferentes modelos y después agregarlos. El árbol de decisión contiene diferentes tipos de algoritmos y son los siguientes:

- Random Forest

Un bosque aleatorio es un clasificador que consiste en una colección de clasificadores estructurados en árbol {h (x, k), k = 1, ...} donde {k} son vectores aleatorios distribuidos de forma idéntica, independientes y cada árbol emite un voto unitario para la clase más popular en la entrada x [26]

Se entiende que el bosque aleatorio crea múltiples árboles de decisión con el fin de obtener una mejor precisión.

Si bien tiene similitudes al método de ensamblaje Bagging, en esta existe una alta correlación entre los árboles que permite realizar una selección aleatoria de "m" predictores, siendo comúnmente "m" el único hiperparámetro.

-XGBOOST

La potenciación de árboles es un método de aprendizaje automático muy eficaz y ampliamente utilizado [27].

XGBOOST utiliza el método de "ensemble methods" el cual usa múltiples algoritmos de aprendizaje para mejorar el rendimiento predictivo, diferente a usar uno solo. Tiene tanto algoritmo de resolución de modelos lineales como algoritmos de aprendizaje de árboles; también utiliza un meta-algoritmo que ayuda a mejorar la estabilidad y precisión entrenando a cada modelo, utilizando un subconjunto aleatoriamente del conjunto de entrenamiento.

El objetivo mas resaltante del algoritmo es el de optimizar la función objetivo que consiste en dos partes, por una función de perdida del entrenamiento ($\mathcal{d}$) y un término de regularización(β).

$$\Omega(\theta) = \sum_{i=1}^{n} \mathrm{d}(y\imath, \hat{y}\imath) + \sum_{k=1}^{K} \beta(f_k)$$

Donde $\hat{y}\imath$ es el valor predictivo, n es el número de instancias de la data de entrenamiento, k es el número de árboles generados y f_k es un árbol del conjunto de árboles [28].

$$\beta(f_t) = \gamma \mathrm{T} + \frac{1}{2}\left[\alpha \sum_{j=1}^{T} |c_j| + \lambda \sum_{j=1}^{T} c_j^2\right]$$

Donde γ es la reducción mínima de pérdida dividida, es la representación de la regularización en el peso y c es el peso asociado a cada hoja, donde T es el número de hojas [28].

La utilización de computación paralela y distribuida hace que el aprendizaje sea más rápido, lo que permite una exploración más rápida del modelo.

2.2.3.1.5. *Redes Neuronales*

Las redes neuronales son el desarrollo más reciente en la identificación asistida por computadora y representan una técnica muy diferente de las analizadas anteriormente. Una red neuronal ofrece un enfoque de caja negra [29].

Es una técnica que simula vagamente el comportamiento del homólogo biológico y que anhela resolver problemas de la misma forma que un ser humano inspirado que observa su comportamiento en su homólogo biológico.

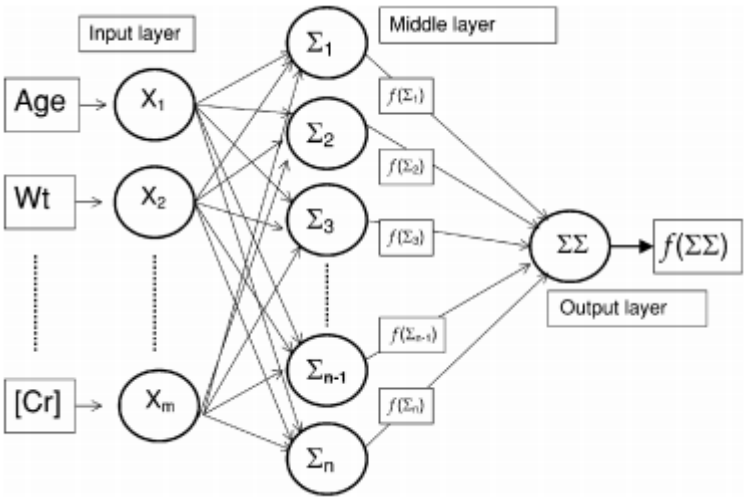


Figura 8. Arquitectura de una ANN
Fuente: N. Tangri, D. Ansell, and D. Naimark (2008). Predicting technique survival in peritoneal dialysis patients: Comparing artificial neural networks and logistic regression. [30].

En la Figura 8 se muestra que los nodos individuales están interconectados a los nodos de entrada, media y salida; haciendo que están relacionadas con los nodos que están unidos entre sí formando una capa de nodos interconectados. Primero la capa de entrada obtiene los valores de los datos, luego la capa del medio recibe la suma de sus entradas ponderadas junto con el peso de la conexión y se aplica una función de activación. Finalmente, la neurona de salida obtiene la suma ponderada y aplica una función.

A lo largo de la historia de los Neuronal Networks se han realizado investigaciones que fomentan la creación de nuevos patrones, comenzando con el Perceptrón que usaba dos capas [31]. Esta utilizaba adiciones y sustracciones simples hasta que la investigación de Rosenblat encontró dos limitaciones; la primera que no podían procesar el circuito de o-exclusivo y la segunda que los ordenadores en ese entonces estaban limitados a un procesamiento bajo, lo que hacía necesitar un tiempo muy largo para realizar su ejecución.

- **BackPropagation**

En su esencia, la propagación hacia atrás es simplemente un método exacto para calcular todos los derivados de una sola cantidad objetivo (como error de clasificación de patrón) con respecto a un gran conjunto de cantidades de entrada (como los parámetros o ponderaciones en una regla de clasificación) [32].

BackPropagation es un algoritmo de entrenamiento que utiliza el perceptrón multicapa. El perceptrón multicapa se define por su arquitectura que conforma el número de neuronas, capas y otros, siendo los métodos de activación, la especificación del descenso del gradiente y las especificaciones de la presentación de eventos [33]. Adicional a esto, se usa el descenso de gradiente para encontrar el conjunto de pesos que minimizan el error.

2.2.3.2. *Aprendizaje No Supervisado.*

Este aprendizaje consigue producir conocimiento de las variables de entrada que se proporciona sin etiquetas de salida, ya que divide los datos en datasets que no están etiquetados en función de algunas características ocultas en los datos, debido a que no se puede evaluar el resultado puede ser útil para tareas como la detección de anomalías; también tiene la ventaja de que los datos que sirven para entrenar son menos costosos de conseguir. [34]

El aprendizaje no supervisado tiene solo una clasificación: el clustering. El clustering es una aplicación fuertemente utilizada en procesos comerciales para segmentar clientes y productos para facilitar los procesos de decisiones referentes a qué vender y a quiénes [35].

2.2.4. CRISP-DM

La metodología CRISP-DM (Cross Industry Standard Process for Data Mining) es una estructura de procesos intersectoriales para proyectos de minería de datos siendo ampliamente utilizada debido a su flexibilidad, utilidad práctica para la utilización de análisis para resolver problemas empresariales difíciles.

Describe los enfoques comunes que suelen venir de las experiencias de expertos de minería de datos y de los procedimientos estándares conocidos, las cuales son estructuradas en 6 fases bidireccionales, permite en el desarrollo de una fase es posible revisar los anteriores procesos o de manera general.

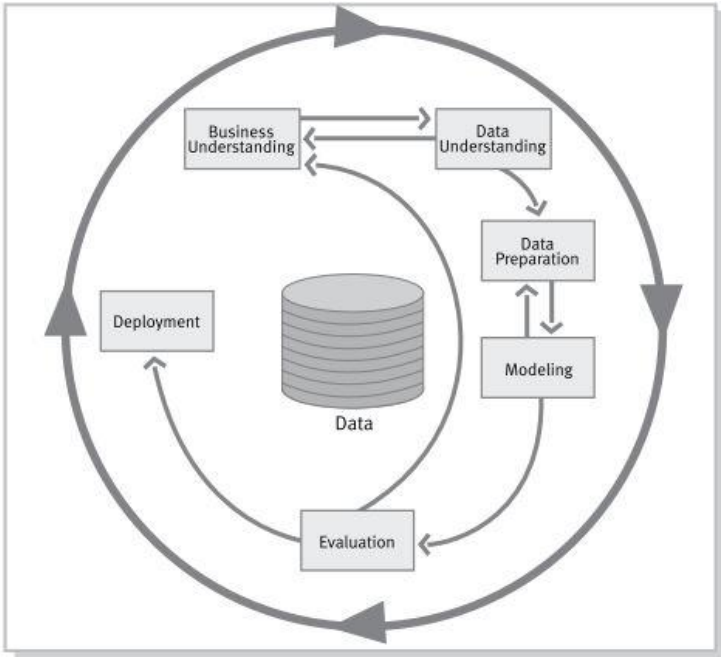


Figura 9. Metodología Crisp-DM.
Fuente: P. Chapman *et* (2000). CRISP-DM 1.0 Step-by-step

En la figura 9 se aprecia como la metodología se estructura en seis fases el desarrollo de esta serie de fases o etapas funcionan de manera cíclica e iterativa [36], cada una cuenta con tareas generales y específicas que permiten cumplir con los objetivos del proyecto.

A continuación, se describen las fases de la metodología:

2.2.4.1. *Business Understanding (Comprensión del negocio).*

Esta fase inicial se enfoca en la comprensión de los requisitos y objetivos del cliente [37]. Después se convierte el conocimiento en objetivos técnicos y en un plan preliminar diseñado para alcanzar los objetivos.

2.2.4.2. *Data Understanding (Comprensión de los datos).*

La fase de entendimiento de datos comienza con establecer el primer contacto con el problema a través de tareas como la recolección de datos [37], identificar los problemas de calidad, descubrir

conocimiento preliminar sobre los datos y establecer las relaciones más resaltantes para formar hipótesis en cuanto a la información oculta.

2.2.4.3. *Data Preparation (Preparación de los Datos).*

La fase de preparación de datos cubre las actividades como la limpieza de los datos [36], preparación para la fase de modelación, técnicas de normalización, discretización, tratamiento de valores perdidos, generación de variables adicionales, integración de diferentes orígenes de datos, cambios de formato, entre otros para construir el conjunto final de datos a partir de los datos en bruto recolectado para la utilización en la herramientas de modelado [38].

2.2.4.4. *Modeling (Modelamiento).*

En esta fase se selecciona y aplica las técnicas de modelado apropiados para el proyecto, y calibrar sus parámetros a valores óptimos [37]. Existen varias técnicas para el mismo tipo de problema por ello se debe realizar bajo ciertos criterios: adecuado para el problema, disponer de la calidad de los datos y conocimiento de la técnica porque algunas tienen requerimientos específicos sobre la forma de los datos.

2.2.4.5. *Evaluation (Evaluación).*

En esta etapa el proyecto se ha construido uno o varios modelos que alcanzan la calidad suficiente considerando el cumplimiento de los criterios de éxito del proyecto. Es preciso en determinar si hay alguna cuestión de negocio que no haya sido considerado lo suficiente y revisar el proceso, considerando los datos obtenidos, para poder repetir alguna actividad o fase que se haya cometido un error y validar si los datos obtenido cumplen con el problema planteado [36]. Al final de esta fase, se debería obtener una decisión sobre si el modelo generado cumple los criterios de éxito establecidos, se procede a la explotación del modelo.

2.2.4.6. *Implementación del Modelo*

Usualmente, el modelo final no es el final del proyecto, a pesar de que el objetivo del modelo sea aumentar el conocimiento de los datos, se procede a organizar y presentar para el uso del cliente. Dependiendo del conjunto de requisitos la fase de implementación puede incluir desde el caso mas simple hasta un informe [37].

2.2.5. Analisis de Preprocesamiento de Datos

2.2.5.1. *Analisis Exploratorio de los Datos*

Es un paso esencial en cualquier análisis de investigación ya que su objetivo principal es examinar los datos de distribución, valores atípicos y anomalías para dirigir pruebas específicas de la hipótesis. También proporciona herramientas para la generación de hipótesis mediante la visualización y comprensión de los datos, generalmente a través de la representación gráfica. [39]

La razón de la gran dependencia de los gráficos es que, por su propia naturaleza, el papel principal de del análisis exploratorio es explorar, y los gráficos le dan a los analistas un poder incomparable para hacerlo, mientras están listos para obtener información sobre los datos.[40]

El análisis exploratorio de los datos se puede resumir en los siguientes objetivos:

- Maximizar el conocimiento y comprender la estructura de la base de datos
- Detectar valores atípicos y anomalías (valores que son significativamente diferente en las otras observaciones)
- Desarrollar modelos parsimoniosos (un modelo predictivo o explicativo que funcione con la menor cantidad de variables de exposición posible) o una selección preliminar de modelos apropiados
- Extraer y crear variables académicamente relevantes.

2.2.5.2. *Transformacion de los Datos*

La informacion original no es adecuado tanto para adquirir conocimiento a travez de ella. En estas situaciones es necesario aplicar diferentes técnicas de transformación entre estas se encuentan la normalización de variables cuantitativas transformacion exponencial y logaritmica, recodificacion de variables cuantitativas a cualitativas.

En la Figura 10 se muestra los diferentes tipos de variables que se puede encontrar en un dataset son numéricos, intervalos, ordianles, categóricos, binario y textual.

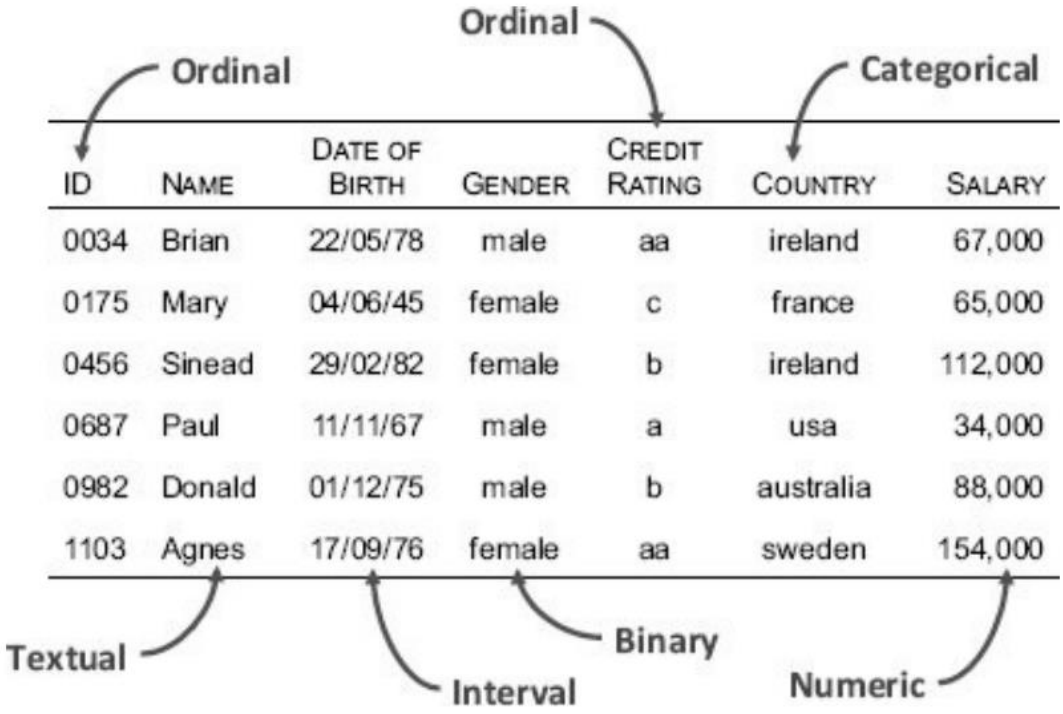


Figura 10. Data set con diferentes tipos de datos
Fuente: He, Haibo and Ma, Yunqian (2013). Imbalanced Learning: Foundations, Algorithms, and Applications [41]

Frecuentemente se divide en dos tipos generales variables cuantitativas (numericos e intervalos), mientras las variables cualitativas (categóricos, ordinales, binarios y texto). Existen otras características derivadas comunes como los agregados que se formulan a travez del recuento, suma, promedio, mínimo o maximo dentro de un grupo con una característica en común, mientras que las ratios utilizan dos o mas variables continuas sin procesar para formar una relación representada en una variable y por ultimo las asignaciones es utilizado cuando se encuentra con mucha frecuencia los mismos valores en las variables continuas en diferentes niveles para crear una función categorica.

2.2.5.3. *Estandarizacion y Normalizacion de Datos*

Se refiere a comprimir o extender los valores de la variable, la normalización intenta dar a todos los atributos un peso igual. La normalizacion es particularmente util para algoritmos de redes neuronales y agrupacion de vecino mas cercanos.

Esto es enormemente favorable para atributos con rangos inicialmente grandes como el tema de salarios. Existen muchos metodos entre las cuales esta:

-Escalado de variables

Siendo que realiza transformacion linean en los datos originales tomando el minimo y maximo valor de un atributo. Lo malo de este metodo es que si existe un excesivo ruido esto va a a ser ampliado[42].

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

-Puntuacion estándar

Este método se utiliza cuando es desconocido el minimo y máximo real del atributo. Robusta en casos donde existen valores atípicos, gracias a que las desviaciones medias no esta al cuadrado[42].

$$X_{normalized} = \frac{X - X_{mean}}{X_{stddev}}$$

2.2.5.4. *Tecnicas de Discretizacion*

Siendo el proceso de discretización de transformar variables cuantitativas en variables cualitativas entre los diferentes tipos. Segun considera cinco familias principales de medidas de evaluación[43] :

-Estadistica: Siendo que implica la correlación entre los atributos tales como ChiMerge el cual consta en inicializar y un proceso de fusión ascendente, donde combina intervalos hasta que se cumple una condición de terminación.

ChiMerge: Discretization of numeric attributes

-Binning: esta categoría se refiere a la ausencia de una medida de evaluación. Es el mas simple método para discretizar un atributo creando un número especificado de contenedores. Cada

contenedor se define un priori y asigna un número especificado de valores por atributo siendo que se divide en dos igual amplitud e igual frecuencia.

-Informacion: Esta familia incluye métodos de entropía es una medida de pureza que mide la cantidad de información necesaria para especificar a que clase pertenece y otras derivadas como el índice de Gini [44].

-Conjuntos en bruto: Son métodos que evaluan de acuerdo a las medidas y propiedades del conjunto aproximado[43].

-Envoltorio: Son métodos que se basa en el error proporcionado por un clasificador entre ello podemos encontrar de clasificación de votación o clasificadores como Naive Bayes[43].

2.2.5.5. *Balanceo de Datos*

Según el trabajo de Fernandez, Galar, del Jesus y Herrera se demuestra que los datos desequilibrados de varias clases fácilmente se pierde el rendimiento de la clase minorista en comparación con las demás variables [45].

Puede ser tomado bajo 3 enfoques, el primero es mediante métodos a nivel de datos que consiste en modificar la colección de datos para equilibrar las distribuciones, el segundo es a nivel de algoritmo que modifican los existentes algoritmos con el objetivo de aliviar el sesgo de los objetos mayoritarios y el último es la la combinación de los dos anteriores [46].

A nivel de datos los métodos más utilizados para el tratamiento de datos desequilibrados son:

Sub Muestreo

El submuestreo elimina los casos de clase mayoritaria, de los datos de entrenamiento. Esto con el fin de equilibrarlo con la clase desbalanceada, teniendo como efecto secundario la reducción de la información obtenida [41].

Sobre Muestreo

Este método utiliza la clase minorista de la data de entrenamiento, replicando las observaciones hasta equilibrar la distribución de las clases. Si bien este método no realiza la pérdida de

información, el inconveniente de realizar casos replicados de la data original puede ocasionar un posible sobreajuste.

2.2.6. Evaluación del Modelo

Según Amazon [47] un modelo siempre debe ser evaluado para determinar si se realizará un buen trabajo de predicción para nuevos y futuros datos de destino. Dado que las futuras instancias tienen valores de destino desconocidos, se debe comprobar la métrica de precisión del modelo de ML en relación con los datos de los que ya sabe la respuesta de destino y utilizar esta comprobación como proxy de precisión predictiva para futuros datos.

2.2.7. Métricas de Evaluación

Precision Equilibrada (Balanced Accuaracy) en problemas de clasificación binaria y multiclase para hacer frente a conjuntos de datos desequilibrados. Se define como el promedio de recuerdo obtenido en cada clase. El mejor valor es 1 y el peor valor es 0 cuando adjusted=False.

Tasa de verdaderos positivos, esta métrica nos mostrará que, de todos los desertores reales, cuántos se predice correctamente. La formula es $TPR = \frac{TP}{(TP+FN)}$

Precisión o tasa de error, es una de las métricas más comunes en la práctica utilizada por muchos investigadores para evaluar la capacidad de generalización de los clasificadores [48].Se calcula el número de clasificadores correctos sobre el número total de predicciones.

Kappa es una alternativa de precisión de clasificación para casos donde exista un problema de desequilibrio entre las clases. La fórmula es $Kappa = \frac{P_o - P_e}{1 - P_e}$

Donde el P(o) es el acuerdo observado y el P (e) es la probabilidad hipotética de acuerdo por azar.

Sensitivity (Recall) son las observaciones de clase positiva que predicen correctamente. Según la formula $Sensitivity(Recall) = \frac{TP}{TP+FN}$, la sensibilidad se calcula con el cociente entre el número de verdaderos positivos y la suma de este mismo término con el número de falsos negativos.

Specificity también llamado tasa negativa verdadera. Es la proporción de casos negativos reales que están predichas correctamente.

Según la fórmula $Specificity = \frac{TN}{TN+FP}$ la especificidad se calcula con el cociente entre el número de verdaderos negativos y la suma de este mismo térmico con el número de falsos positivos.

2.2.8. Métodos de Evaluación

A) Matriz de Confusión

También conocida como matriz de error, es una matriz que valora la capacidad predictiva de un modelo de clasificación.

Hair et al. en el año 1999 describe que se construye tabulando de forma cruzada el miembro del grupo correcto con el miembro del grupo predicho, donde los números de la diagonal de la matriz representan clasificaciones correctas y los números fuera de la diagonal son clasificaciones incorrectas [49].

Esta matriz es una tabla de contingencia de dos variables en la cual muestra los verdaderos positivos, los falsos positivos, verdaderos negativos y falsos negativos (FN) que se han obtenido en el proceso de clasificación para cada clase.

B) Curva ROC

Es posible comparar el desempeño de un clasificador con otro mediante el área bajo la curva ROC, la métrica de evaluación más utilizada. La curva ROC se forma trazando la tasa de verdaderos positivos (sensibilidad) y la tasa de falsos positivos (especificidad). El área bajo la curva (AUC) será la medida de mayor desempeño para evaluar los clasificadores. Esta medida es la más recomendada para datos desbalanceados, y será la principal medida para la toma de decisiones al momento de elegir los modelos. Sin embargo, se recalca que a pesar que las curvas ROC han sido (y siguen siendo) ampliamente utilizados en la literatura científica, estas deberían complementarse con otras curvas tales como la curva PRC (curva de precisión y sensibilidad).

C) Curva de Aprendizaje

Una curva de aprendizaje muestra una medida del rendimiento predictivo en un dominio dado en función de alguna medida de cantidades variables de esfuerzo de aprendizaje

Es una herramienta que muestra la puntuación de entrenamiento y validación de un estimador, en función de alguna medida de cantidades variables de esfuerzo de aprendizaje (Learning Curves in Machine Learning) y determinar si el estimador sufre más de un error de varianza o un error de sesgo.

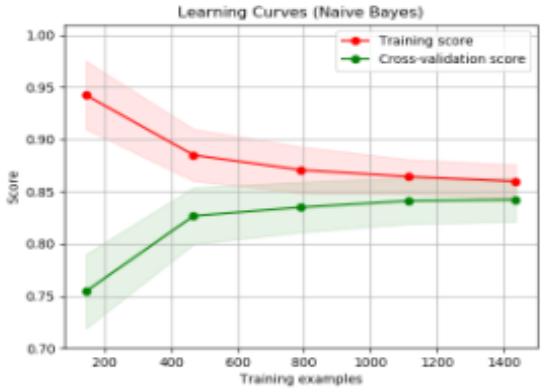


Figura 11. Curva de Aprendizaje de Naive Bayes
Fuente: Plotting Learning Curves, scikit-learn (2007-2020) [50]

En la figura 11 se puede visualizar la curva de aprendizaje de un clasificador de Bayes, en el que se tiene en cuenta que el puntaje de entrenamiento y validación cruzada no son muy buenos al final porque el puntaje de entrenamiento tiene una curva de aprendizaje decreciente (es muy alto al principio y va disminuyendo). En cuanto al puntaje de validación cruzada, se tiene una curva de aprendizaje creciente (muy bajo al principio y va aumentando según a la cantidad de datos entrenamiento), por lo tanto en estos casos se tendrá que usar un estimador o una parametrización del estimador actual para que tenga un sesgo menor.

2.2.9. Validación del Modelo

Se refiere a que el modelo debe mostrar una calidad similar a los datos de prueba con el que se realizó la evaluación del modelo.

2.2.10. Métricas de Validación

Bias: Siendo la diferencia entre la predicción esperada del modelo y los valores correctos que se están tratando de predecir. Cuando el modelo se vuelve más complejo, disminuye el bias.

Variance: El modelo con alta variación tiende a prestar más atención a los datos de entrenamiento, por consecuencia no generaliza y se obtiene un modelo que funciona muy bien con esos mismos datos. Con los datos de prueba se tiene un elevado índice de error.

2.2.11. Métodos de Validación

A) Hold- out validation

En este método, los datos se dividen en dos conjuntos separados, es decir, datos de entrenamiento y datos de prueba. La proporción entre los datos de entrenamiento y los datos de prueba no es vinculante, pero para garantizar que la variante en el modelo no sea demasiado amplia, generalmente se usan 2/3 de los datos como entrenamiento y los otros 1/3 se usan como datos de prueba [51]**.**

Basados en esta definición se deduce que es muy simple, siendo rápida en su ejecución pero difícil de calcular cualquier variación o intervalos de confianza por solo tener un conjunto de datos [52].

B) K-Fold CROSS-VALIDATION

Este método asigna aleatoriamente las n observaciones a una de las particiones de tal manera que las particiones tengan un tamaño casi igual [53].

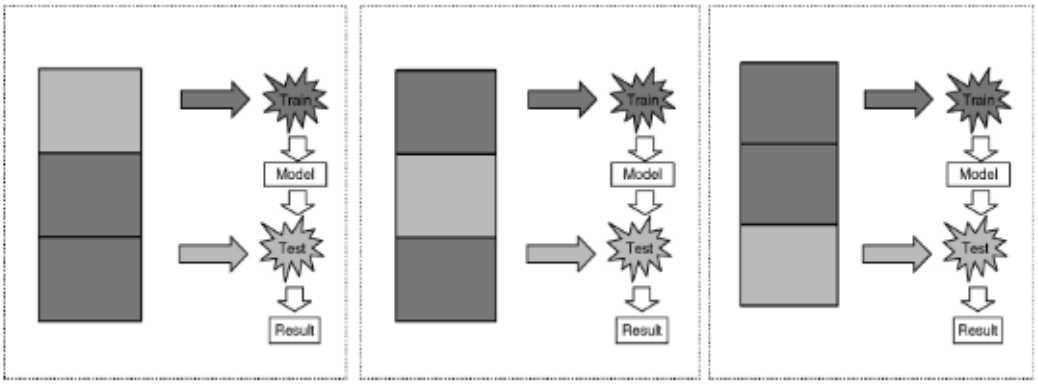


Figura 11. CROSS - VALIDATION
Fuente: C. K. Park and D. G. Kim (2012). Cross-Validation [54].

En la Figura 12 se observa que está dividida en 3 segmentos de igual tamaño. La sección oscura se usa para el entrenamiento y la más clara para la validación.

Según la fórmula del MSE $CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i$ $.MSE_i$, el error cuadrático medio se calcula sobre el pliegue retenido anteriormente, repitiendo el procedimiento k veces, y finalmente para calcular la estimación se promedian los valores.Se utiliza el MSE para cuantificar el error de prueba pero esto solo ocurre cuando son problemas de regresión, en cambio en problemas de clasificación se usa el número de observaciones mal clasificados. Por lo tanto en este método es común dividir el conjunto de datos en 10 segmentos.

C) Hiperparámetros

En el aprendizaje automático, la optimización o ajuste del hiperparámetro es el problema de elegir un conjunto de hiperparámetros óptimos para un algoritmo de aprendizaje [55].

Son medidas que no se aprende durante el entrenamiento del modelo ya que su valor tiene que ser especificado por el usuario en base a la problemática y mediante el uso de la validación cruzada.

Por ejemplo:

- Número mínimo de observaciones que debe tener un nodo para ser divido.
- Número mínimo de observaciones que deben tener los nodos terminales.
- Número de divisiones de la rama más larga (profundidad máxima).
- Número máximo de nodos terminales.

Se usa para distinguirlo de los parámetros estándar del modelo y sirve de apoyo para que el modelo arroje mejores resultados.

C) Curva de Validación

En la validación de un modelo se necesita la función de puntuación, por ejemplo: precisión para los clasificadores. La forma correcta de elegir múltiples hiperparámetros de un estimador es la búsqueda de grillas o métodos similares que seleccionan el hiperparámetro con la puntuación máxima de un conjunto de validación o múltiples conjuntos de validación, teniendo en cuenta que, si se optimiza los hiperparámetros en una función de una puntuación de validación, esta estaría sesgada y ya sería una buena estimación de generalización. Por lo tanto, para obtener una estimación conveniente de generalización se debe calcular el puntaje en otro conjunto de prueba.

```
>>> import numpy as np
>>> from sklearn.model_selection import validation_curve
>>> from sklearn.datasets import load_iris
>>> from sklearn.linear_model import Ridge

>>> np.random.seed(0)
>>> iris = load_iris()
>>> X, y = iris.data, iris.target
>>> indices = np.arange(y.shape[0])
>>> np.random.shuffle(indices)
>>> X, y = X[indices], y[indices]

>>> train_scores, valid_scores = validation_curve(Ridge(), X, y, "alpha",
...                                               np.logspace(-7, 3, 3))
>>> train_scores
array([[ 0.94...,  0.92...,  0.92...],
       [ 0.94...,  0.92...,  0.92...],
       [ 0.47...,  0.45...,  0.42...]])
>>> valid_scores
array([[ 0.90...,  0.92...,  0.94...],
       [ 0.90...,  0.92...,  0.94...],
       [ 0.44...,  0.39...,  0.45...]])
```

Figura 12. Algoritmo de la Curva de Validación
Fuente: Propia.

En la figura 13 se puede visualizar el uso de la función validation_curve que permite determinar los puntajes de entrenamiento y hacer las pruebas variando los valores del hiperparámetro.

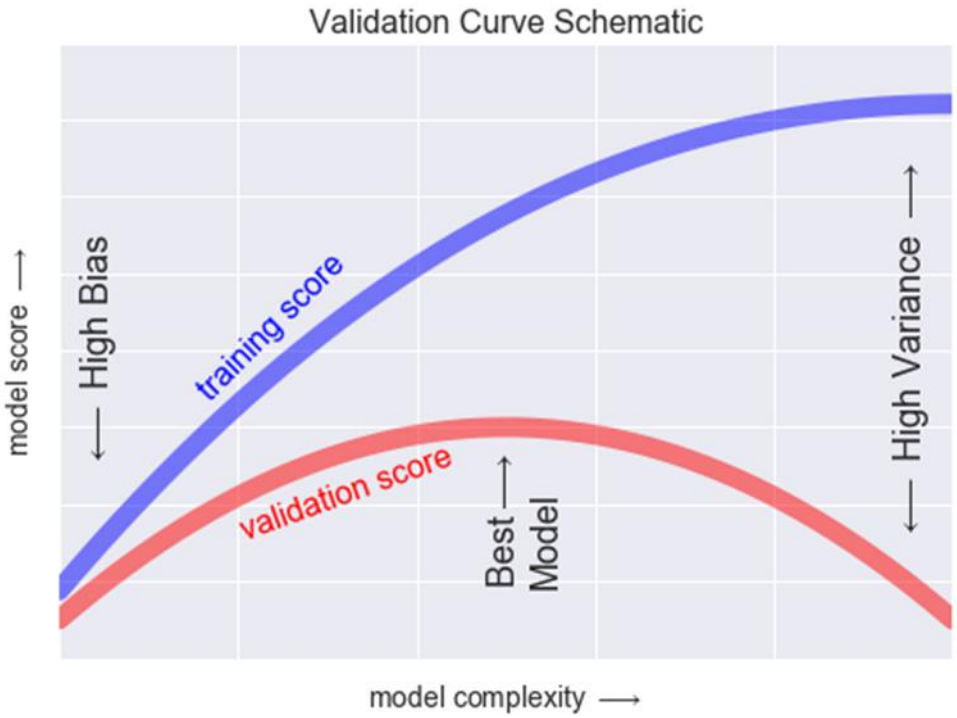


Figura 13. Curva de Validación
Fuente: Python Data Sciencie [56]

Por consiguiente, en la Figura 14 se visualiza en un gráfico de líneas los puntajes de validación y entrenamiento, esto quiere decir que, si el puntaje de entrenamiento y validación son bajos, el estimador será insuficiente; y si el puntaje de entrenamiento es alto y el puntaje de validación es bajo, el estimador esta sobreajustado, y si fuera lo contrario, el modelo está funcionando muy bien. Se aclara que por lo general no es posible obtener un bajo puntaje de capacitación y un alto puntaje de validación.

2.2.12. Deserción Estudiantil.

Es una interrupción en la trayectoria académica individual, la cual genera un distanciamiento entre las expectativas del estudiante frente a su proyecto educativo y las posibilidades objetivas de llevarlo a cabo.

Según Parámo y Correa [57] se deberá entender por deserción estudiantil como el abandono definitivo de las aulas de clase por diferentes razones y la no continuidad en la formación académica.

Se puede interpretar a la situación que enfrenta el estudiante como el término de su carrera universitaria en el tiempo planeado según el Programa Estudiantil.

Respecto a los factores de descerción, los estudios señalan que son variados y van deesde los económicos hasta los académicos [58], lo que hace difícil la implementación de políticas contundentes que permitan disminuir los indicadores de deserción. Este problema puede estar ocasionado por falta de recursos económicos, falta de conocimientos y habilidades del estudiante o malos servicios académicos por parte de la institución.

La deserción ocasión pérdidas, primero, por el costo de oportunidad en el que incurre el estudiante que no lo logra culminar sus estudios universitarios y segundo porque las universidades se afectan financieramente al reducir sus ingresos (universidades privadas), y seguir teniendo altos costos en recursos de docentes, infraestructura y personal administrativos

2.2.12.1. *Teorías de la Deserción Estudiantil.*

Existen muchas investigaciones relevantes referidas a la situación estructural en las que se destacan particularidades como la de Vicent Tinto quien hace una revisión exhaustiva sobre la deserción estudiantil y los diferentes factores que implican. Para ubicar estos factores es posible categorizar en las siguientes teorías:

a) Psicológicas

b) Societal

c) Económico

d) Organizacional

e) Interaccional

El primer tipo, incluye las teorías que resaltan los atributos psicológicos individuales. El segundo, tercero y cuarto comprenden teorías que destacan distintas influencias del ambiente sobre la conducta estudiantil. La ultima categoría considera que la conducta estudiantil es influenciada por atributos individuales, como por ejemplo el ambiente, sobre todo aquel que se localiza en el entorno inmediato de la institución [59].

2.2.12.2. *Clasificación de la deserción estudiantil*

Se pueden diferenciar dos tipos de abandonos en estudiantes universitarios: con respecto al tiempo y al espacio

2.2.12.2.1. Deserción respecto al tiempo

En la figura 15 se puede visualizar la deserción con respecto al tiempo, la cual se clasifica a su vez en:

- Deserción precoz: Persona que ingresa a la universidad habiendo dado su examen de admisión, pero no se llega a matricular.
- Deserción temprana: Es la persona que se retira de la universidad en sus primeros cuatros semestres
- Deserción tardía: Es la persona que abandona sus estudios después del quinto semestre en adelante.

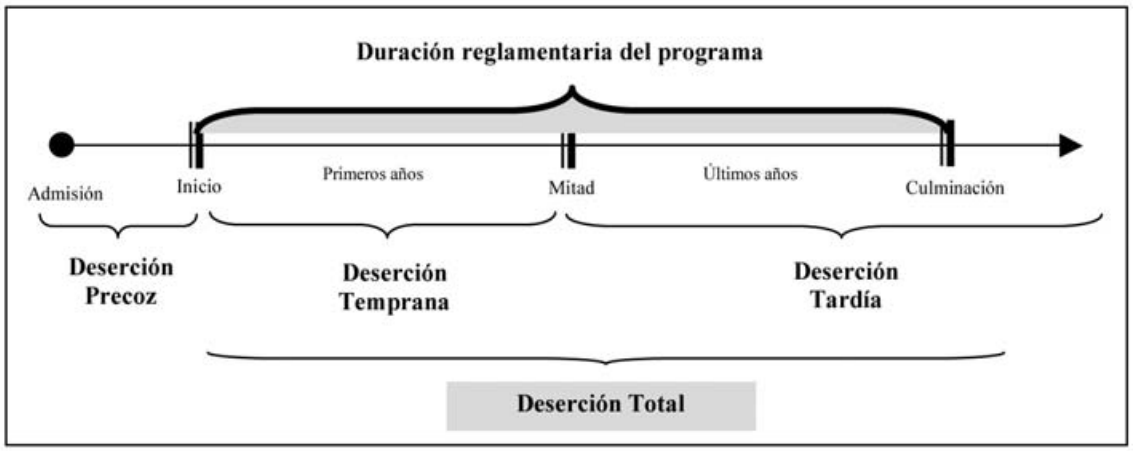


Figura 14. Deserción respecto al Tiempo
Fuente: E. Castaño, S. Gallón, K. Gómez, Johanna Vásquez, and Vásquez, (2008), Análisis de los factores asociados a la deserción estudiantil en la Educación Superior [60].

2.2.12.2.2. Deserción de acuerdo al espacio

En la figura 16 se observa la deserción con respecto al espacio, la cual se divide en: (i)deserción institucional: caso en el cual el estudiante abandona la universidad, (ii)deserción interna o del programa académico: se refiere al alumno que decide cambiar su programa académico por otro que ofrece la misma institución universitaria y (iii) la deserción del sistema educativo [61].

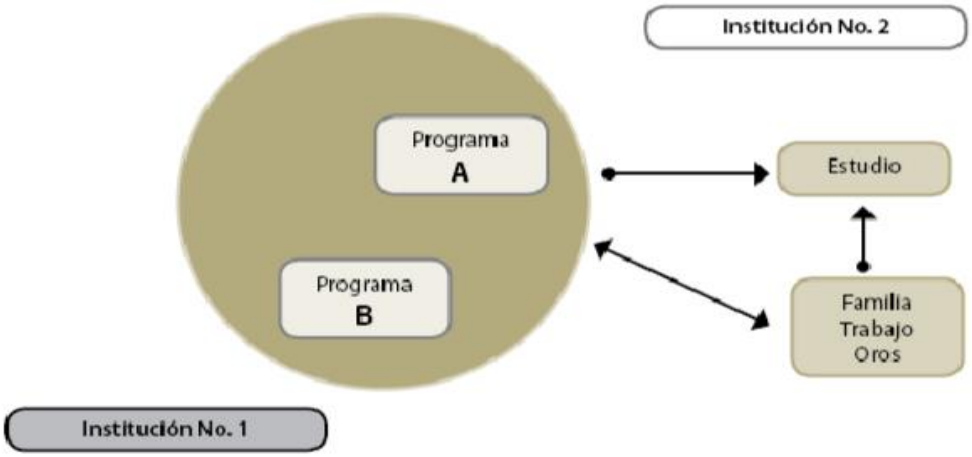


Figura 15. Deserción de acuerdo al espacio
Fuente: E. Castaño, S. Gallón, K. Gómez, Johanna Vásquez, and Vásquez, (2008), Análisis de los factores asociados a la deserción estudiantil en la Educación Superior [60].

2.2.12.3. *Determinantes de la deserción estudiantil.*

Se han involucrado gran cantidad de variables explicativas relacionadas con las condiciones socioeconómicas y el desempeño académico de los estudiantes, encontrando, por ejemplo, que estudiantes con menores ingresos al momento de iniciar sus estudios tienen mayores probabilidades de desertar al igual que los alumnos con padres de menor nivel de educación. Sin embargo, en términos generales y de acuerdo con la revisión de la literatura, existen más trabajos en el que destacan la perspectiva institucional en donde los diferentes conjuntos de variables (institucionales, socioeconómicas, académicas y personales) son analizadas de manera independiente y no como un conjunto de factores que determinan la decisión de desertar [61] como se muestra en la figura 17.

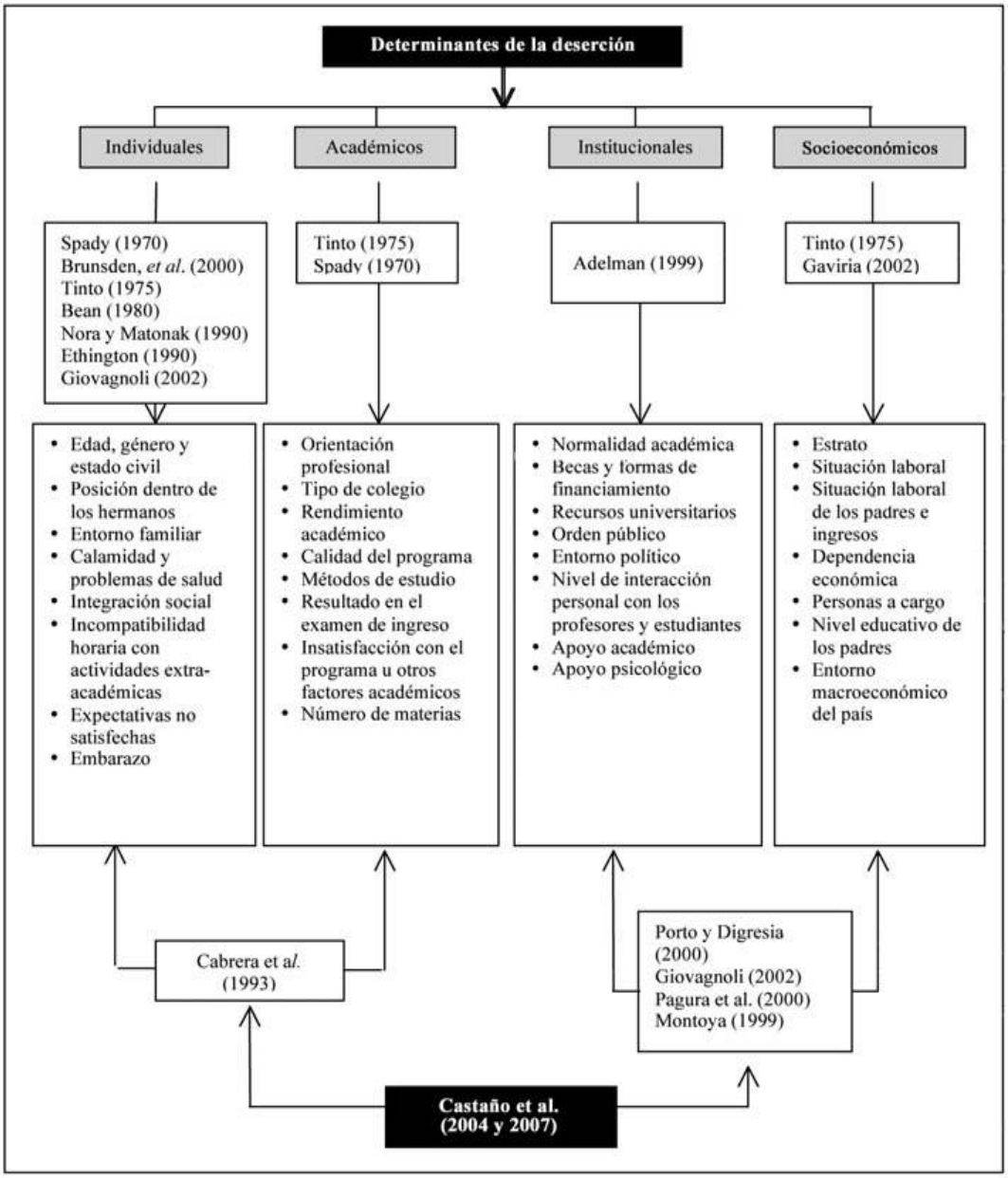


*Figura 16.*Mapa de los Determinantes de la Deserción
Fuente: E. Castaño, S. Gallón, K. Gómez, Johanna Vásquez, and Vásquez, (2008), Análisis de los factores asociados a la deserción estudiantil en la Educación Superior [60].

2.3. Marco Conceptual

2.3.1. Deserción

La deserción es una interrupción en la trayectoria académica individual, la cual genera un distanciamiento entre las expectativas del estudiante frente a su proyecto educativo y las posibilidades objetivas de llevarlo a cabo [57].

Se puede interpretar a la situación que enfrenta el estudiante el término de su carrera universitaria en el Tiempo Planeado según el Programa Estudiantil. Qué factores ocasionan la deserción:

Los estudios señalan que los factores que motivan la no permanencia de los estudiantes en las universidades son variados, desde los económicos hasta los académicos [62].

Siendo que la deserción estudiantil puede ser ocasionada por falta de recursos económicos, falta de conocimientos y habilidades para la formación del Estudiante o por los servicios académicos por parte de la Institución.

Es uno de los principales problemas que enfrentan las instituciones de educación superior, por la complejidad de los factores involucrados en su determinación, lo que hace difícil la implementación de políticas contundentes que permitan disminuir los indicadores de deserción.

La deserción ocasión pérdidas, primero, por el costo de oportunidad en el que incurre el estudiante que no lo logra culminar sus estudios universitarios, segundo las universidades se afectan financieramente porque se reducen sus ingresos (universidades privadas), y tercero por los altos costos en recursos de docentes, infraestructura y personal administrativos.

2.3.2. Modelo Predictivo

El término modelado se refiere al estadístico de datos como al modelo matemático. Siendo su objetivo desarrollar un mapeo parametrizado entre el dominio de datos y el conjunto de respuestas [63] .

Otra definición del modelado predictivo es el proceso de desarrollar un modelo matemático que genere una predicción precisa [64].

2.3.3. Machine Learning

Es un conjunto de método que detecta patrones de los datos automáticamente, que pueda ser usado para predecir datos futuros o para la toma de decisiones bajo incertidumbre [65] .

Es una disciplina que abarca los estudio de métodos para detectar patrones a un conjunto de datos bajo el análisis de datos [66].

2.3.4. Aprendizaje Supervisado

Es la búsqueda de algoritmos que a partir de unas instancias obtenidas del conjunto de datos pueda producir hipotesis generales para realizar predicciones sobre instancias futuras [17]

También se refiere al proceso de aprendizaje automático que tenga la capacidad de aprender una función que realice una predicción desde los de tipo entrada a los de tipo de salida a partir de datos proporcionados [67].

2.3.4.1. *K-Nearest Neighbors*

La idea básica es cuando se necesite clasificar unas nuevas observaciones algoritmo cuando encuentra los k vecinos mas cercanos entre los datos pondera a sus vecinos para asignarle la clasificación. La complejidad de KNN radica en encontrar una buena medida de similitud [62].

Es una recopilación de datos M, siendo que el objeto de datos M(i) es el vecino mas cercano a un objeto de datos, donde el dist(q,M(i)) minimiza ,siendo una medida de distancia definida para los objetos en cuestión [53].

2.3.4.2. *Support Vector Machine*

Son una clase de algoritmos lineales que puede ser usado para la regresión, clasificación, detección de novedades y otras aplicaciones [53].

El termino se generalizo a maquina del núcleo. En cual la máquina de núcleo son métodos de margen máximo que la suma de influencias de un subconjunto de las instancias de entrenamiento, siendo las influencias dadas por los núcleos de similitud de la aplicación para fines de clasificación, regresión, detección de valores atípico, reducción de dimensiones [68].

2.3.4.3. *Naive Bayes*

Es un algoritmo de aprendizaje que usa la regla de Bayes junto con una suposición de que los atributos sean condicionalmente independientes [53].

Otra definición se basa en que dado el valor objetivo los valores de los atributos sean condicionalmente independientes [63].

2.3.4.4. *Decisión Tree*

Es un modelo de clasificación siendo su estructura como un árbol, entendible desde usuarios no expertos y que pueda inducir de manera eficiente a partir de datos [53].

Es una estructura de datos jerárquica que usa la estrategia de dividir y conquistar, que se puede utilizar tanto en regresión como en la clasificación. Siendo que su estructura fácilmente se convierte en un conjunto de reglas simples [45].

2.3.4.5. *Redes Neuronales*

Son algoritmos de aprendizaje basados en la analogía general del funcionamiento sobre el cerebro humano específicamente en sus redes neuronales, su aprendizaje se logra al ajustar los pesos entre las conexiones entre los diferentes nodos [46].

Siendo una de desarrollo mas reciente en la identificación asistida por computadora y en comparación a las demás es una técnica muy diferente, nos ofrece un enfoque de caja negra [27].

2.3.5. Aprendizaje No Supervisado

Llamado como auto supervisado porque desarrolla cualidades para detectar características que representan información espacial o temporal localizada en los datos.

En el aprendizaje no supervisado es superfluo ubicar la respuesta correcta en los datos de entrenamiento. Ya que no se rastrea la propagación de un resultado acreditado, sino el hallazgo de nuevos patrones o resultado [59].

2.3.5.1. *Clusteres K-Means*

Es uno de los métodos de agrupación en clúster más populares, siendo la idea básica de este algoritmo es una agrupación inicial pero no óptima, que reubica cada punto a su nuevo centro más cercano, actualiza los centros de agrupación en clústeres calculando la media de los puntos

miembros, repite la reubicación y el proceso de actualización hasta que se cumplan los criterios de convergencia (como el número predefinido de iteraciones, la diferencia en el valor de la función de distorsión) [46].

Es un método comúnmente utilizado para particionar automáticamente un conjunto de datos en k grupos que procede seleccionando k centros de agrupamiento iniciales y luego refinándolos iterativamente [60].

2.3.5.2. *Agrupación Jerárquico*

Es una técnica conocida por una diversidad de términos, esto es, además de análisis de conglomerados se puede encontrar como análisis de clúster y análisis de grupos que pertenece al igual que el escalamiento multidimensional, a los métodos de interdependencia dentro del conjunto de técnicas multivariantes [69].

Es una técnica que configuran grupos con estructura arborescente, de forma que clústeres de niveles más bajos van siendo englobados en otros clústeres de niveles superiores [70].

2.3.5.3. *Mezclas de Modelos Gaussianas*

Es un modelo probabilístico cuando el caso común en que las n-distribución sean gaussianas, se ajusta un modelo de mezcla, si el número n- de distribuciones es distinguido [71] .

También según [72] Es un modelo de distribución de probabilidad de mezclas finitas y ha sido aplicado de forma exitosa a sistemas de reconocimiento de patrones.

2.3.6. Algoritmo

Según [63] en el científico informático, el termino algoritmo son instrucciones sistemáticas para resolver un problema usando la computadora. Siendo que en el ámbito de aprendizaje automático se usa para entrenar, validar y probar el modelo para encontrar el mejor valor para los parámetros, validar y evaluar su rendimiento.

También según [73] son instrucciones secuencialmente para proporcionar una solución a un problema. Siendo que la predicción del algoritmo depende de los datos de entrada y sus parámetros.

2.3.7. Metodología

Es una estrategia general que guía a los procesos y actividades en un dominio determinado, siendo que no depende de herramientas o tecnologías. Proporciona al científico de datos un marco sobre el cual utilizar métodos, procesos y heurísticas con el objetivo para obtener respuestas o resultados [74].

Otra definición según [75] es un enfoque general que guía a las técnicas y actividades dentro de un dominio especifico. Nos ofrece un marco para adquirir respuestas utilizando una variedad de métodos, procesos y heurísticas.

2.3.7.1. *CRISP-DM*

Según [76] es una metodología de minería de datos detallada y utilizada frecuentemente por que proporciona una guía muy explícita sobre las diferentes fases de un proyecto de minería de datos

También Según (CRISP-DM SPSS) CRISP-DM se describe en un modelo de proceso jerárquico, que consiste en el conjunto de tareas en cuatro niveles de abstracción que son fase, tarea genérica, tarea especializada e instancia de proceso siendo desde lo más general a especifico.

CAPÍTULO III. Materiales y métodos

3.1. Lugar de ejecución

El lugar de ejecución del proyecto se realizó en la Facultad de Ingeniería y Arquitectura con los datos de los estudiantes desde el 2009 hasta la actualidad, extraídos de la Dirección General de Tecnología de Información bajo la autorización de secretaria general.

Secretaria general es un órgano de apoyo académico y administrativo de toda la estructura orgánica de la Universidad Peruana Unión (UPeU), el cual está a cargo del Secretario General, fedatario de la universidad, nombrado por el Consejo Universitario a propuesta del rector.

3.2. Población y Muestra

3.2.1. Población

La población estuvo compuesta por los datos personales, académicos y financieros de los estudiantes de la FIA del periodo de Enero del 2009 hasta Julio de 2019, siendo un total 3161 registros.

3.2.2. Muestra

La muestra para el estudio se considero el total que representa la población debido a las características que exige el método para el desarrollar un modelo predictivo a través de Machine Learning.

3.3. Tipo de investigación

La Investigación Aplicada también es una investigación original para adquirir nuevos conocimientos. Sin embargo, se dirige principalmente hacia un objetivo u objetivo práctico específico [77].

Debido a que el objetivo de esta tesis es llevar a cabo realizar prototipos experimentales de diferentes modelos sobre la deserción estudiantil para evaluar el modelo más eficaz para su puesta en práctica en la facultad de FIA de la Universidad Peruana Unión.

3.4. Enfoque de Investigación

Cuantitativo: Los resultados del modelo de pronóstico se muestran los valores de cada ítem que ayuden en la mejora de Toma de Decisiones.

La investigación cuantitativa se ocupa de datos cuantificables, medidas objetivas que se pueden repetir siempre y en cualquier lugar, dado que todos los parámetros que podrían influir en el proceso de medición se han analizado y especificado en el diseño de la investigación. Esta es la metodología de investigación predominante en todas las ciencias naturales [78].

3.5. Diseño de la Investigación

Dado al tipo de datos que han sido recolectados, se considero el diseño pre-experimental condiserando uno de los tipos mencionados por Campbell y Stanlley: Diseño de Estudio con un grupoe estático, ya que este diseño trabaja con dos grupos, uno de ellos es denominado grupo experimiental y es el que recibe nuestra variable independiente (Modelo de Aprendizaje Supervisado de Machine Learning) y otro grupo llamado grupo de control el cual no recibe ninguna variable, teniendo como característica principal que ambos grupos serán evaluados después de que un grupo experimental sea intervenido.

X1 (Variable Independiente) ⟶ O1 (Grupo Experimental)

O2 (Grupo de Control) ⟶ O2

3.6. Formulación de Hipótesis

3.6.1. Hipótesis General

El mejor modelo predictivo a través de los algoritmos de Machine Learning es eficaz en el pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima.

3.7. Definición y Medición de Variables

3.7.1. Identificación de las Variables

VD=Pronóstico de Deserción de estudiants de la FIA.

VI= Modelo de Aprendizaje Supervisado de Machine Learning.

3.7.2. Operacionalización de las Variables

Tabla 1. Operacionalizacion de Variables

Variables	Dimensión	Indicador
Pronóstico de la Deserción	Personal	-Genero -Religión -Nacionalidad -Edad en el último Ciclo
	Financiero	-Debe -Credito
	Academico	-Escuela -Número de Cursos Desaprobados -Cursos repetitivos 1 vez -Cursos repetitivos 2 veces -Cursos repetitivos 3 veces -Cursos repetitivos 4 veces -Curso repetitivo -Número de Curso Generales reprobados -Número de Cursos Especiales Reprobados -Número de Cursos Específicos Reprobados -Retirado -Número de Traslados Internos -Número de Contratos del Estudiante de la Carrera Actual -Número de Contratos del Estudiante de la Anterior Carrera -Número de Contratos del Estudiante dentro de la Universidad -Incremento de Nota -Desincremento de Nota -Nota final del Ultimo Ciclo -Vacacionales
Algoritmos de Machine Learning	Efectividad del Modelo	Balanced Accuracy Bias Varianza TPR
	Fases de Machine Learning	
	Patrones Predecibles	

3.8. Técnica de Recolección de Información

La técnica de recolección fue a través de la observación y recolección de información de la data que esta almacenada en el Sistema Académico, administrada por la Dirección General de Tecnología de Información – DIGETI, asi mismo esta data fue otorgada a los investigadores previo cumplimiento de su procedimiento respectivo y autorización por la Direccion General de Investigacion – DGI.

3.9. Tratamiento de la Información

La información fue adquirida mediante formatos csv y luego se tuvo que realizar una limpieza y generar la información del alumno por contrato mediante una aplicación en Laravel, para luego ser analizada mediante el IDE Spyder.

3.10. Evaluación de los Datos

La evaluación de los datos fue a travez de técnicas estadísticas de comparación de medias para una muestra, considerando los datos de los estudiantes ingresantes del 2018-1 para ser comparadas con los datos de estos mismos en el ciclo 2019-1. De manera que se compare a travez de contraste de hipótesis el resultado de deserción de los estudiantes con el modelo predictivo y según la realidad de deserción que se encuentra los estudiantes ahora.

3.11. Materiales de la Investigación.

3.11.1. Python.

Lenguaje de programación de alto nivel que en los últimos años ha sido constantemente usado por los desarrolladores de software. Se creó a principios de los noventa por un investigador holandés llamado Guido Van Rossum que trabaja en el centro de investigación CWI (Centrun Wiskunde & Informática) sacando su primera versión en 1991 en una plataforma bajo una licencia open source propia. En septiembre para su versión 1.6, decide cambiar la licencia por una que sea compatible con la licencia GLP a la cual la denominó Python Software Foundation License, este hecho implicó que fuese posible modificar el código de fuente.[79]

Se caracteriza por ser un lenguaje tipado dinámico, es decir que no es necesario declarar el tipo de dato que va a encerrar una determinada variable, ya que su tipo de dato se determinará en el tiempo de la compilación según el tipo de valor que se le indique. Resulta que también es fuertemente tipado ya que no permite tratar a una variable como si fuera de un tipo distinto al que

tiene. Además de ello, es conocido por ser multiplataforma porque se puede instalar en diferentes sistemas operativos y porque por ser un lenguaje orientado a objetos, es decir, los conceptos del mundo real relevantes para nuestro problema llegan a ser llamados en el programa, objetos y clases. [80]

Por otro lado, cuenta con la facilidad de interactuar con otros lenguajes de programación gracias a sus módulos y extensiones, es decir su código se puede ejecutar desde diferentes sistemas. [79]

Python cuenta con librerías para el desarrollo de ciencia de los datos, algunas de ellas son:

a) NumPy: Según [81] son representaciones estándar numéricos que permite la implementación de cálculos numéricos en un lenguaje de alto nivel.

b) Pandas: Proporciona funciones y estructuras de datos tabulares orientados a columnas y series para facilitar el trabajo. [82].

c) Imbalanced-learn: Según [83] proporciona un número de métodos para enfrentar el problema de datos desequilibrados.

CAPÍTULO IV. Propuesta de la Ingenieria

4.1. Desarrollo Metodológico

En la figura 19 se visualiza la metodología que se aplicó en la investigación bajo la estructura de CRISP-DM, esta metodología cuenta con 6 fases las cuales se explican a continuación:

La primera fase consta de la comprensión del negocio en el que se determinó los objetivos por el cual se requiere desarrollar esta investigación. El objetivo fue desarrollar un modelo que permita facilitar la toma de decisiones de la administración de la Facultad de Ingeniería y Arquitectura (FIA) frente al incremento de la deserción de los estudiantes durante el ciclo académico, bajo un modelo de machine learning.

En la segunda fase, se trata la comprensión de los datos, lo cual implicó reconocer las variables disponibles de la institución, hacer una selección de las variables, validarlas y categorizarlas en 3 dimensiones: personal, académico, financiero. Finalmente se obtuvo un diccionario preliminar de datos que pasaron por un proceso de transformación.

En la tercera fase se realizó la preparación de los datos, el cual implicó la limpieza de datos y transformaciones. En algunos casos, las transformaciones generaron variables adicionales y en otros, se eliminaron. A algunas variables se les aplicó el método de estandarización con el objetivo de realizar una normalización de los datos, y obtener un diccionario de datos actualizados que finalmente quedó con 15 variables.

La cuarta fase se trató del diseño de los modelos predictivos, específicamente los modelos supervisados. Se realizó la división de los datos validados en 2 partes: entrenamiento y testeo, después, se seleccionaron las técnicas del modelo más apropiadas para la clasificación binaria de la variable predictiva, esto se realizó según los datos que se registró en el diccionario de datos.

Seguidamente se extrajeron los hiperparametros para saber cuáles serían los mejores algoritmos del modelo y finalmente se verifica estos mediante una curva de validación.

La quinta fase de la validación del modelo se inició después de haber obtenido las mejores técnicas del modelo con los parámetros definidos. Teniendo en cuenta esto, se utiliza los datos de testeo para realizar la matriz de confusión, en tanto que la data de entrenamiento se usó para realizar la curva de aprendizaje, los cuales tienen como fin, determinar el modelo predictivo eficaz.

Finalmente, en la fase de Implementación del Modelo, se empaquetó el modelo determinado para que se implementa en un REST API bajo el lenguaje de programación Python

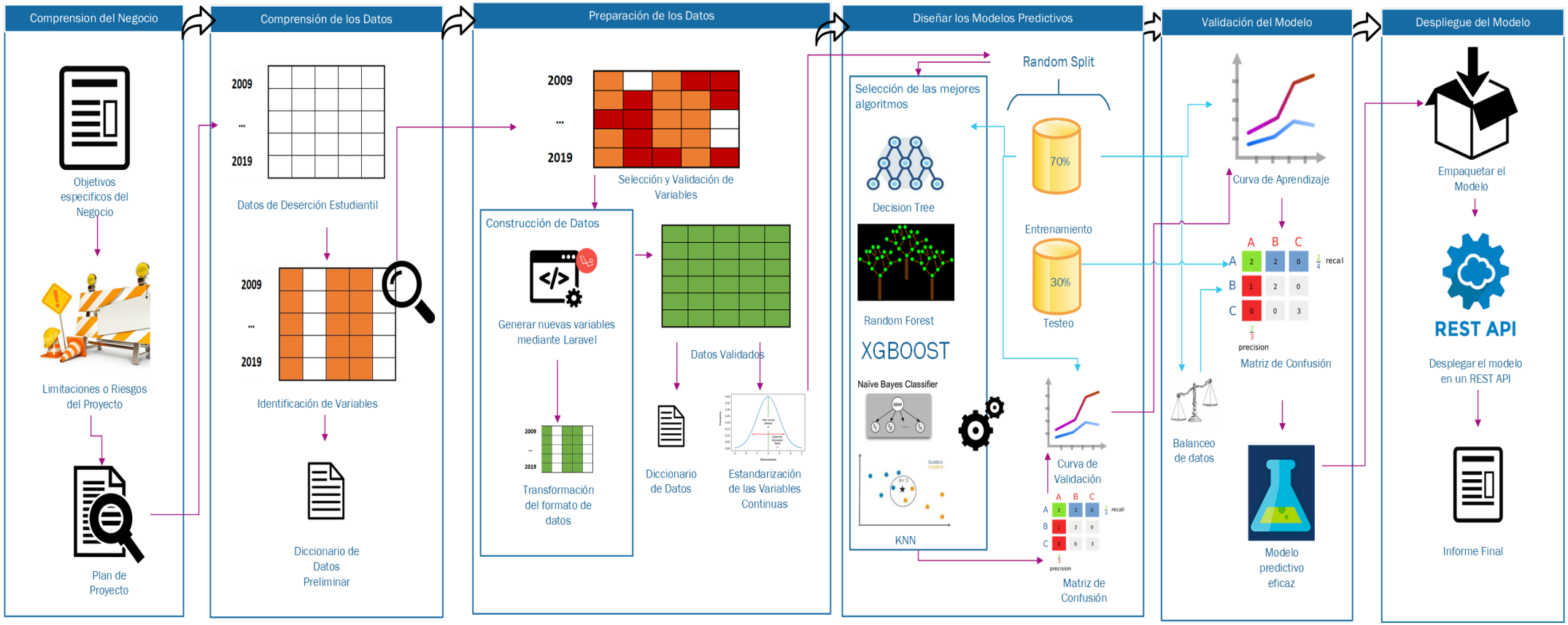


Figura 17. Arquitectura de Solucion.
Fuente: Propia.

4.1.1. Comprensión del negocio

Actualmente en las universidades en el Perú, se están implementando mecanismos y herramientas tecnológicas de inteligencia artificial para identificar los patrones de las acciones y/o comportamientos de los estudiantes desertores, con el propósito de reducir la tasa de deserción de los programas académicos. Estas herramientas cumplen un rol importante, porque permite crear sistemas innovadores a diversas instituciones educativas de nivel universitario.

La Universidad Peruana Unión no es ajena a estas necesidades, ya que es una institución educativa particular diferenciada que cada año genera diversos tipos de información que pueden servir para realizar distintos análisis utilizando el machine learning. En dicha universidad, cada programa académico cuenta con un director de escuela, el cual es el responsable de gestionar diferentes mecanismos para apoyar el avance académico de los estudiantes brindando becas, realizando visitaciones y tutorías académicas, entre otros, para que el estudiante se mantenga en la carrera. En la figura 20 se muestran dos escenarios que presentan un programa de estudio de la universidad respecto a la deserción estudiantil, la primera se refiere al estudiante que se retira durante ciclo académico ya sea formalmente o no, y la segunda se refiere al estudiante que termina su ciclo académico, pero no regresa el siguiente ciclo; de estos dos escenarios, solo se analizará el segundo.

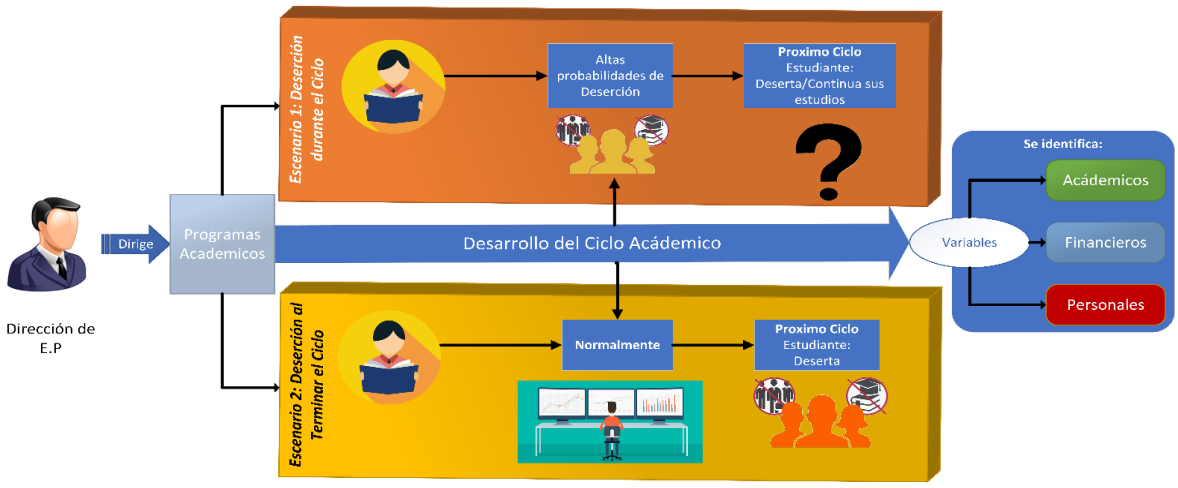


Figura 18. Escenarios de la Deserción Estudiantil
Fuente: Propia.

4.1.2. Comprensión de los datos

Consiste en la obtención de los datos proporcionados por la misma universidad bajo una previa selección de los factores relacionados a la deserción de estudiantes: personal, académico y financiero, consultados a la base de datos del sistema académico mediante SQL, lo cual se muestra en la Figura 21 el diagrama de obtención de variables.

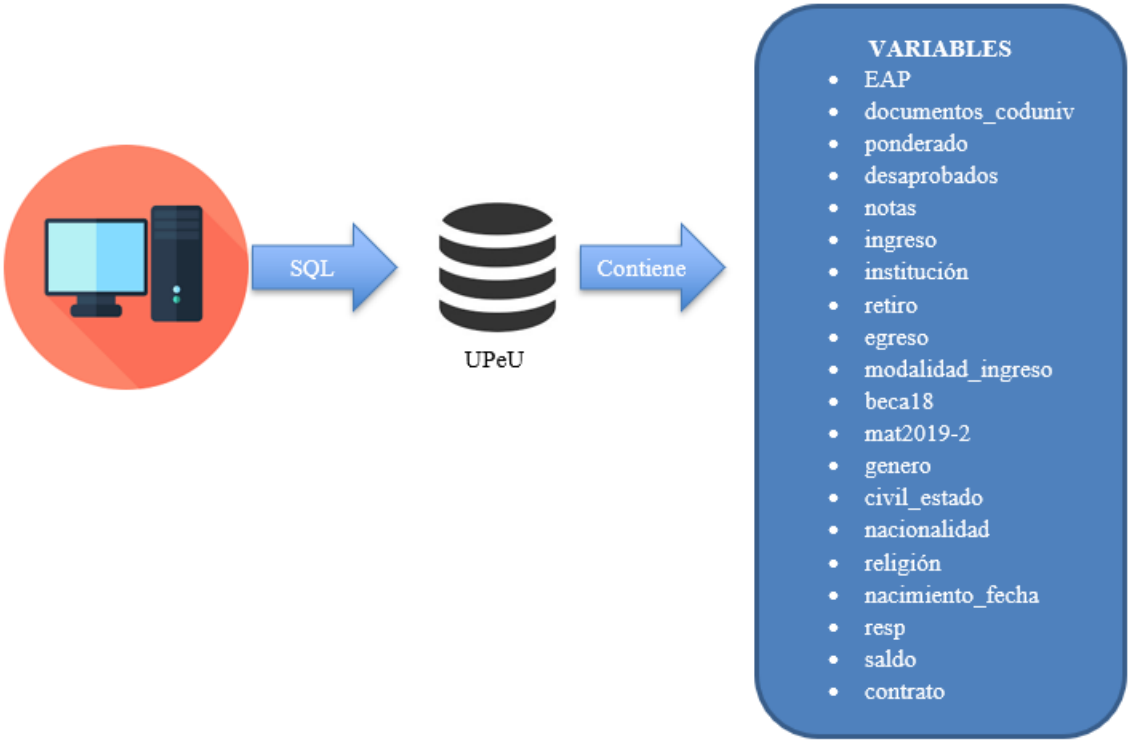


Figura 19. Diagrama de Obtención de Variables de la UPeU
Fuente: Propia

En la figura 22, se muestran las variables de los factores: personal, académico y financiero; algunas de estas fueron eliminadas por tener diferente formato de datos e inconsistencias que puedan afectar al análisis, estos son: civil_estado, institución, ponderado, ingreso, modalidad_ingreso y resp.

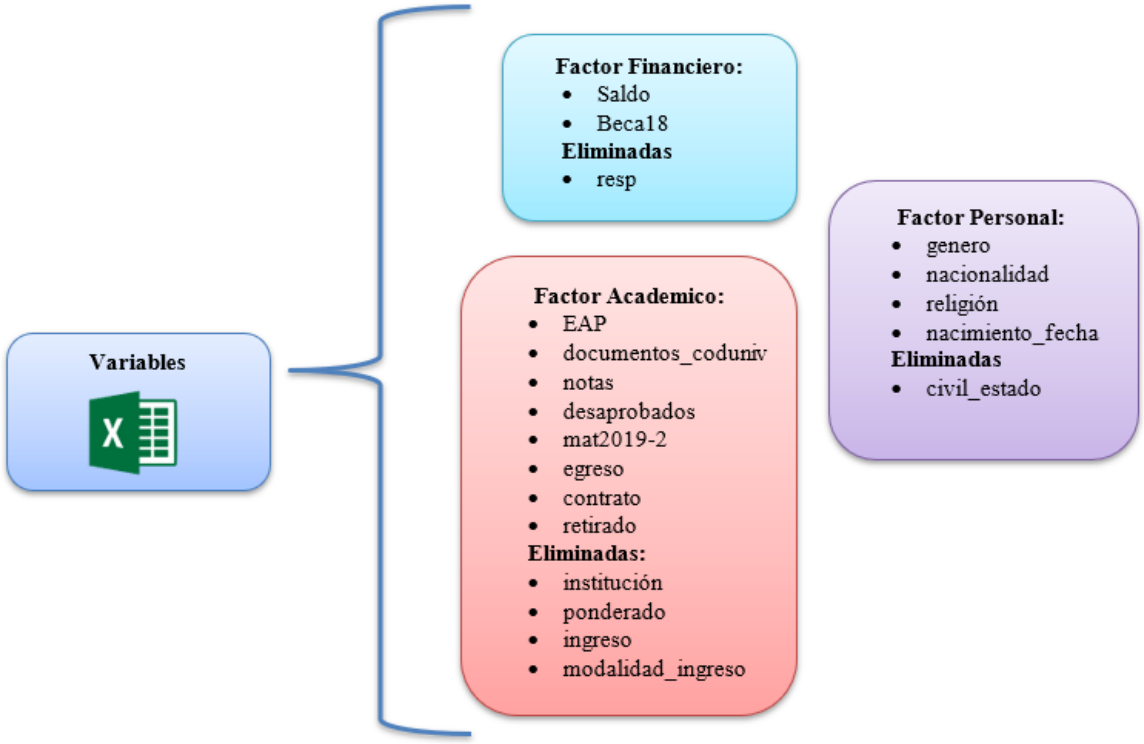


Figura 20. Factores de la Deserción Estudiantil con sus variables
Fuente: Propia

En la tabla 2 se muestra el diccionario preliminar de datos, con el objetivo de describir el formato actual de la variable y su respectiva descripción para un mejor entendimiento. Este diccionario fue formado a base de las variables seleccionadas en la Figura 20.

Tabla 2. Diccionario de Datos Preliminar

CLASIFICACIÓN	VARIABLE	TIPO DE DATO	DESCRIPCION
ACADÉMICAS	**EAP**	Texto	Escuela Académico Profesional
	documentos_coduniv	Numérico	Documento de identidad
	desaprobados	Texto	Cursos desaprobados
	notas	Texto	Notas por ciclo
	retiro	Numérico	Año y ciclo de retiro
	egreso	Numérico	Año de egreso
	contrato	Texto	Contratos del estudiante por ciclo
	retirado	Texto	Semestres Retirados
	repetición	Numérico	Número de registros repetidos por código
	mat2019-2	Texto	Matriculado en el ciclo 2019-2
PERSONALES	**genero**	Texto	Tipo de genero
	nacionalidad	Texto	Nacionalidad
	religión	Texto	Tipo de religión
	nacimiento_fecha	Texto	Fecha de nacimiento
FINANCIERAS	**beca18**	Texto	Ayuda PRONABEC
	saldo	Texto	Saldo financiero

4.1.3. Preparación de los datos

En la figura 23 se visualiza la preparación de datos de las variables seleccionadas, el cual contempla la limpieza, transformación y estandarización de los datos. La limpieza de datos implicó homogenizar y eliminar registros nulos e irrelevantes. Asimismo, en la transformación se generó nuevas variables con el objetivo de mejorar el contenido de los registros para la solución de la problemática. En la estandarización, se realizó la normalización de los datos para que el modelo tenga un mejor aprendizaje.

Figura 21. Etapas de la Preparación de Datos
Fuente: Propia

En la primera etapa se realizó la limpieza de datos mediante el software de Microsoft Excel 2016, de modo que en la tabla 3 se muestran las variables con sus respectivas tareas.

Tabla 3. Actividades de la Pre-Limpieza de Datos - Microsoft Excel

VARIABLES	TAREAS
EAP	● Se eliminaron las carreras que ya no existen.
documentos_coduniv	● Se eliminaron los códigos universitarios menores al año 2019. ● Se eliminaron los códigos universitarios que no tenían el mismo formato.
desaprobados	● No se realizó limpieza.
notas	● Se presenció datos faltantes, por lo tanto se eliminó el registro (fila).
retiro	● No se realizó limpieza.
beca18	● No se realizó limpieza.
mat2019-2	● No se realizó limpieza.
genero	● No se realizó limpieza.
nacionalidad	● No se realizó limpieza.
religión	● Se presenció datos faltantes, por esa razón, se eliminaron los datos del alumno en las demás variables.
nacimiento_fecha	● No se realizó limpieza.
saldo	● No se realizó limpieza.
contrato	● No se realizó limpieza.
repetición	● No se realizó limpieza

En la figura 24 se presenta la esquematización de la etapa “transformación de los datos”, en el cual se usó como input los 22822 registros de los estudiantes de todas las carreras de la UPeU. Estas pasaron por tres fases:

- Fase 1: Eliminación de Registros Erróneos
- Fase 2: Obtención de los ciclos desertores y retirados
- Fase 3: Obtención de casos de los diferentes contratos por registro.

Obteniendo como resultado final un total de 36402 registros (ahora exclusivos del FIA). Esto se realizó con el lenguaje de programación PHP.

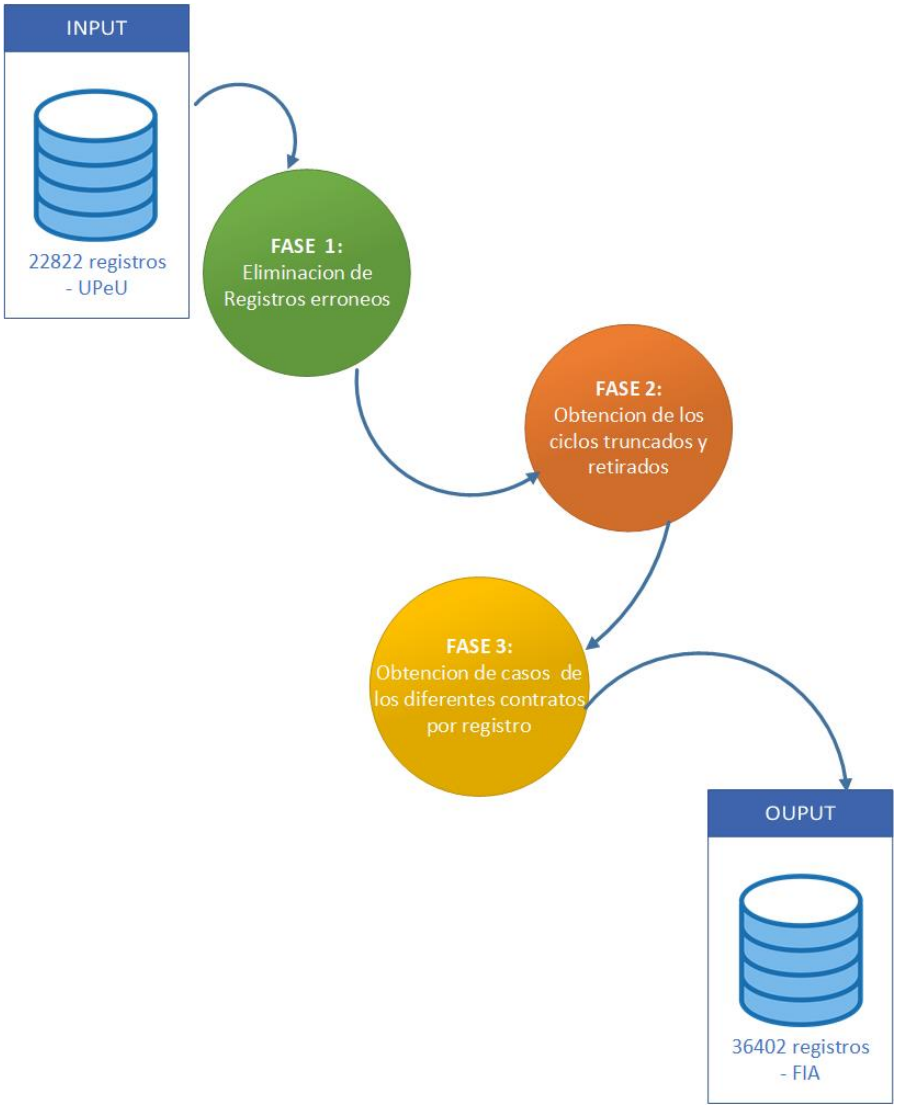


Figura 22. Fases de la Limpieza de Datos
Fuente: Propia

Cabe resaltar que cada fase pasó por un proceso de transformación detallado en la Figura 25, del cual sus resultados se muestran en el Anexo 1.

Figura 23. Estructura de Transformación
Fuente: Propia

Fase 1: Eliminación de registros erróneos

Tal como se observó en la figura 25, se realiza en primer lugar, la eliminación de los registros erróneos por presentar incoherencias e inconsistencias. Un caso de ejemplo son los registros en donde no coincidía la cantidad de notas con la cantidad de contratos, como en el caso de estudiantes de traslado interno para quienes se necesitó acumular los registros de notas de su carrera anterior y actual.

En la Tabla 4 se muestra todas las acciones realizadas a las variables, comenzando por ingresar en la sección de entrada a las variables: contratos, notas, código, carrera y repetición. Seguidamente en la sección transformación se elaboró un código en PHP que permitió comparar los registros de cada variable e identificar inconsistencias entre ellos. En la sección salida se obtuvo las variables: los cursos desaprobados, crédito, débito, responsable financiero, carrera académica y repetidos; y finalmente, se obtiene un nuevo registro con las inconsistencias eliminadas. Cabe resaltar que el caso 1 se da cuando los datos del estudiante no muestran inconsistencia entre el número de contrato y notas, en tanto que el caso 2, lo opuesto.

Tabla 4. Filtro de registros erróneos y historial de programas académicos anteriores

<table>
<tr><th colspan="8">CASO 1</th></tr>
<tr><td>Codigo</td><td>201420198</td><td>201420198</td><td rowspan="3">Contratos_acum=""
Sem_acum=""
nro_esc_tras=1
For i in range (1, repeticion):
#Iteracion 2 i=2
#Ultimo Contrato del Registro 2
contrato_actual=2018-2
#Ultimo Contrato del Registro 1
contrato_anterior=2015-2
#Solo continua las siguientes transformaciones si la variable "contrato_actual" es mayor al "contrato_anterior",
en caso contrario termina</td><td rowspan="3"></td><td>Codigo</td><td>201420298</td><td>201420298</td></tr>
<tr><td>Carrera</td><td>Contabilidad</td><td>Ing.Sistemas</td><td>Carrera</td><td>Contabilidad</td><td>Ing.Sistemas</td></tr>
<tr><td>Repeticion</td><td>2</td><td>2</td><td>Repeticion</td><td>2</td><td>2</td></tr>
<tr><td rowspan="2">Notas</td><td rowspan="2">Semestre 2015-1 (15,78), Semestre 2015-2</td><td rowspan="2">Semestre 2018-1 (16,82), Semestre 2018-2 (13,47)</td><td rowspan="2">#Las notas anteriores acumuladas
Notas_anteriores=Semestre 2015-1 (15,78) , Semestre 2015-2 (18,40)
#Las notas del registro actual
Notas_actuales=Semestre 2018-1 (16,82),Semestre 2018-2 (13,47)
#Se genera una nueva variable Notas con las Notas_anteriores y las Notas_actuales
Notas = Semestre 2015-1 (15,78),Semestre 2015-2 (18,40),Semestre 2018-1 (16,82), Semestre 2018-2 (13,47)</td><td rowspan="2">#El numero de Contratos_actuales
n_contra_actual=2
#El numero de Notas_actuales
n_notas_actual=2
#Si existe diferencia entre la variable n_contra_actual y n_notas_actual, entonces la variable "Eliminar" pasa a Si</td><td>Eliminar</td><td>No</td><td>No</td></tr>
<tr><td>Notas</td><td>Semestre 2015-1 (15,78), Semestre 2015-2 (18,40)</td><td>Semestre 2015-1 (15,78), Semestre 2015-2 (18,40) Semestre 2018-1(16,82), Semestre 2018-2 (13,47)</td></tr>
</table>

Contrato	2015-1,2015-2	2018-1,2018-2	#Los contratos anteriores acumulados
			Contratos_acumuladas=2015-1,2015-2
			#Los contratos del registro actual
			Contratos_actuales=2018-1,2018-2
			#Se genera una nueva variable Contrato
			Contrato = 2015-1,2015-2,2018-1,2018-2
					Contrato	2015-1,2015-2	2015-1,2015-2, 2018-1,2018-2

CASO 2							
Codigo	201420198	201420198			**Codigo**	201420298	201420298

Carrera	Contabilidad	Ing.Sistemas	Contratos_acum=""
			Sem_acum=""
			nro_esc_tras=1
			For i in range (1,repeticion):
			#Iteracion 2 i=2
			#Ultimo Contrato del Registro 2
			contrato_actual=2018-2
			#Ultimo Contrato del Registro 1
			contrato_anterior=2015-2
			#Solo continua las siguientes transformaciones si la variable "contrato_actual" es mayor al "contrato_anterior",
			en caso contrario termina
					Carrera	Contabilidad	Ing.Sistemas

Repeticion	2	2			Repeticion	2	2
Notas					Eliminar	Si	Si

	Semestre 2015-1 (15,78) , Semestre 2015-2 (18,40)	Semestre 2018-1 (16,82), Semestre 2018-2 (13,47)	**#Las notas anteriores acumuladas** **Notas_acumulada=Semestre 2015-1 (15,78) , Semestre 2015-2 (18,40)** **#Las notas del registro actual** **Notas_actuales=Semestre 2018-1 (16,82),Semestre 2018-2 (13,47)** **#Se genera una nueva variable Notas con las Notas_acumuladas y las Notas_actuales** **Notas = Semestre 2015-1 (15,78),Semestre 2015-2 (18,40),Semestre 2018-1 (16,82), Semestre 2018-2 (13,47)**	**#El numero de Contratos_actuales** **n_contra_actual=1** **#El numero de Notas_actuales** **n_notas_actual=2** **#Si existe diferencia entre la variable n_contra_actual y n_notas_actual, entonces la variable "Eliminar" pasa a Si**	Notas	Semestre 2015-1 (15,78) , Semestre 2015-2 (18,40)	Semestre 2015-1 (15,78), Semestre 2015-2 (18,40), Semestre 2018-1 (16,82), Semestre 2018-2 (13,47)
Contrato	2015-1,2015-2	2018-2	**#Los contratos anteriores acumulados** **Contratos_acumuladas=2015-1,2015-2** **#Los contratos del registro actual** **Contratos_actuales=2018-2** **#Se genera una nueva variable Contrato** **Contrato = 2015-1,2015-2,2018-1,2018-2**		Contrato	2015-1,2015-2	2015-1,2015-2, ,2018-2

Leyenda: Bucle Variable Variables Inicial Condició Comentar

FASE 2: Obtención de ciclos truncados y retirados

El desarrollo de la fase se observa en la Tabla 5 y consta de las siguientes acciones:

- Obtención de ciclos truncados

  Esto se obtiene de la variable “contratos”, al observar registros que no muestren continuidad en los ciclos de estudio. Por ejemplo, si se registraron matrículas sucesivas de un estudiante, y de pronto se observa que al siguiente ciclo no se matriculó, este se considera como un ciclo truncado (a excepción de los alumnos graduados).

- Obtención de ciclos retirados

  Esto se obtiene de la variable “notas”, considerando la regla de que, si un estudiante tiene 05 de nota en promedio final, se cuenta como ciclo retirado informalmente.

  Esto se sumó a los ciclos retirados formalmente (variable “retirado”).

Finalmente se modificó la variable “contrato” en base a la identificación de ciclos retirados.

Tabla 5. Obtención de los ciclos desertores y retirados

<table>
<tr><th colspan="2">ENTRADA</th><th rowspan="2">TRANSFORMACION</th><th colspan="2">SALIDA</th></tr>
<tr><th>Variable</th><th>Registro 1</th><th>Variable</th><th>Registro 1</th></tr>
<tr><td colspan="5">CASO 1</td></tr>
<tr><td rowspan="2">Contratos</td><td rowspan="2">2017-2,2018-1,2018-2</td><td rowspan="3">#Los contratos de la variable Retirado se añaden a la variable Contratos

Contratos = 2017-2,2018-1,2018-2 + 2019-1</td><td>Contratos</td><td>2017-2, 2018-1, 2018-2, 2019-1</td></tr>
<tr><td>Contratos = 2017-2,2018-1,2018-2,2019-1
Detectar el contrato faltante de la variable Contratos desde el Ciclo Inicial hasta el ciclo final.
Desertor = ""</td><td>Desertor</td><td>-</td></tr>
<tr><td>Retirado</td><td>2019-1</td><td rowspan="2">#Unimos la variable "Retirado" con la variable "retirado_informal"

Retirado = 2019-1 + ""</td><td rowspan="2">Retirado</td><td rowspan="2">2019-1</td></tr>
<tr><td>Notas</td><td>Semestre 2017-2 (15,78) , Semestre 2018-1(15,40) , Semestre 2018-2(13,44)</td><td>#Las notas promedio menor a 5, es considerado un ciclo retirado informal

retiro_informal = ""</td></tr>
<tr><td colspan="5">CASO 2</td></tr>
</table>

		#Los contratos de la variable Retirado se añaden a la variable Contratos Contratos = 2017-2,2018-1 + 2019-1		Contratos	2017-2, 2018-1, 2019-1
Contratos	2017-2,2018-1		Contratos:2017-2,2018-1,2019-1 # Detectar el contrato faltante de la variable Contrato desde el Ciclo Inicial hasta el ciclo final. Desertor = "2018-1"	Desertor	2018-1
Retirado	2019-1		#Unimos la variable "Retirado" con la variable "retirado_informal" Retirado = 2019-1 + 2018-1	Retirado	2018-1,2019-1
Notas	Semestre 2017-2 (15,78), Semestre 2018-1(4,03)	#Las notas promedio menor a 5, es considerado un ciclo retirado informal retiro_informal = 2018-1			

Leyenda: Bucle Variable Variables Inicial Condició Comentar

FASE 3: Obtención de casos de los diferentes contratos por registro

Esta fase detallada en la Tabla 6, tiene por objetivo obtener el historial del estudiante por cada contrato (ciclo académico). De todas las variables contempladas en el registro, solo se usaron las variables de entrada contrato y notas, el cual después de haberlas transformado, resultaron en las variables denominadas “contratos” e “incremento de notas”.

La transformación implicó destinar un registro por cada contrato de modo que, si un estudiante tuvo 10 contratos, también tendrían 10 registros.

Tabla 6. Obtención de nuevos registros por ciclo

ENTRADA		TRANSFORMACION		SALIDA		
Variable	Registro 1			Variable	Registro 1	Registro 2

Contratos	2017-2,2018-1	Contratos_acum = " "
		Incremento_nota=0
		
		#Se recorre la variable Contratos por cada contrato
		
		For contra in Contratos:
		#Iteracion 1 2017-2
		#Iteracion 2 2018-1
			# Cada contrato que muestra la variable "contra" es agregado a la varaible "Contratos_acum"
			Contatos_acum = 2017-2 + 2018-1
			
			#La nueva variable Contratos es obtenida de
				Contratos	2017-2	2017-2,2018-1

			Contratos_acum Contratos = Contratos_acum			
Notas	Semestre 2017-2 (14,30) , Semestre 2018-1(16,77)		#La diferencia entre la nota del ciclo actual y la nota del Ciclo Anterior Diferencia_nota= 16.77 - 14.30 Incremento_nota= 0 + 2.47	Incremento_nota	0	2.47

Leyenda: Bucle Variable Variables Inicial Condició Comentar

Teniendo los datos limpios y organizados, se realizó una selección de variables para el análisis exploratorio de datos con el objetivo de definir las variables continuas y categóricas, de las cuales solo se estandariza las variables continuas para que los parámetros tengan una distribución normal (media cero y varianza, uno), con el fin de que el algoritmo realice una mejor predicción en base a los pesos aprendidos.

Finalmente se filtraron estos datos específicamente a la población de los estudiantes de la FIA, el cual se particiona en sus diferentes programas académicos: alimentos, sistemas, civil, ambiental, arquitectura; cada uno con sus respectivas variables (Figura 26), con el objetivo de generar un modelo predictivo eficaz de cada uno de los programas académicos.

Figura 24. Base de Datos por Programa Académico
Fuente: Propia

4.1.4. Diseñar los modelos predictivos

Con la data final de las variables definidas en el anexo 1 (diccionario de datos), se usó los algoritmos: K de vecinos próximos (KNN), Naive Bayes, Extra Gradient Boosting (XGBoost), Arboles de Decision (Decision Tree) y Bosques Aleatorios (Random Forest), para predecir la deserción de estudiantes durante el periodo 2009-2019. Cabe resaltar que para la normalización de los datos (sección 1.1.3) no se aplicaron métodos estadísticos sino solo la estandarización de las variables continuas.

Los pasos para predecir la deserción de los estudiantes son los siguientes:

1. Dividir los datos validados utilizando la librería Ramdon Split en 2 partes: Entrenamiento y Testeo.

2. De los algoritmos mencionados, se seleccionó los mejores que se adecúan al modelo, mediante la librería GridSearch CV que determina los hiperparámetros bajo la métrica "balanced_accuracy". A continuación, se muestra en la Tabla 7 con la cuadricula de parámetros para realizar diferentes combinaciones y encontrar los hiperparametros.

Tabla 7. Cuadricula de Parametros por Algoritmo Definido

Algoritmo	Algoritmo Definido	Cuadricula de Parámetros
K-NN	`KNeighborsClassifier()`	`[{'n_neighbors': range(1,10), 'weights': ['uniform'], 'metric': ['euclidean']}, {'n_neighbors': range(1,10), 'weights': ['uniform'], 'metric': ['euclidean']}, {'n_neighbors': range(1,10), 'weights': ['distance'], 'metric': ['manhattan']}, {'n_neighbors': range(1,10), 'weights': ['distance'], 'metric': ['manhattan']}]`
Naive Bayes	`GaussianNB()`	`[{"var_smoothing":[1e-09,1e-8, 1e-7, 1e-6, 1e-5, 1e-4] }]`
Decision Tree	`DecisionTreeClassifier()`	`[{"max_depth": range(1,4), "random_state":[7],'min_samples_split' : range(10,500,20)}]`
Random Forest	`RandomForestClassifier()`	`[{'max_depth': [1,2,3], 'n_estimators': [100, 200,300, 500], 'max_features': ['auto', None,'log2']}]`
XGBOOST	`XGBClassifier( learning_rate=0.02, n_estimators=600, objective='binary:logistic',nthread=1)`	`[{'min_child_weight': [1, 5, 10], 'gamma': [0.5, 1, 1.5, 2, 5], 'subsample': [0.6, 0.8, 1.0], 'colsample_bytree': [0.6, 0.8, 1.0], 'max_depth': [1,2,3] }]`

Con los hiperparametros de los algoritmos realizamos predicciones para obtener los resultados de la metrica “balanced_accurary” y “tpr” con la data de Testeo.

3. Validamos si existe un sobreajuste de los algoritmos mediante el parámetro n_neighbors para KNN, mientras que “var_smoothing” para naive bayes y finalmente el “max_depth” para Decision Tree, Random Forest y XGBOOST, si en caso existe un sobreajuste se paso a cambiar el numero del parámetro o consideramos otro algoritmo

4.1.5. Validación del modelo

Se evaluaron 4 modelos predictivos seleccionados: Naive de Bayes, XGBOOST, Árboles de Decisión (Decision Tree) y Bosques Aleatorios (Random Forest), los cuales se utilizaron para construir el modelo de predicción de la deserción de alumnos. A cada uno de estos modelos se le aplicó una técnica de Balanceo de Datos para equilibrar la cantidad de los estudiantes que continuaron el siguiente ciclo con la cantidad de los que no regresaron.

Para la obtención de un mejor modelo predictivo, se prefirió usar la métrica “balanced accuracy” para encontrar el equilibrio entre las predicciones correctas e incorrectas entre la categoría de la variable desertor. La precisión del equilibrio deriva toda su información de los elementos diagonales y las sumas de filas (tabla 7)

Matriz de Confusión de la Tabla 8 se compone de dos filas y dos columnas dicotómicas (binaria): tipo de clase positiva y negativa. Las observaciones que efectivamente eran positivas y que se predijo como positivo, es llamado Verdadero Positivo (TP). En caso de una predicción incorrecta de la clase positiva, es llamada Falso Positivo (FP). Mientras que las observaciones que efectivamente eran negativas y que se predijo como negativo, es llamado Verdadero Negativo (TN) en caso contrario es llamado Falso Negativo (FN).

Tabla 8. Modelo de Matriz de Confusión

Clase Verdadera / Predicción	+	-
+	Verdadero Positivo	Falso Positivo
-	Falso Negativo	Verdadero Negativo

A partir de las variables obtenidas de la Tabla 8 se pueden adquirir indicadores de desempeño de los modelos. Para este caso se generaron los indicadores más importantes de la matriz de confusión, estos son la tasa positiva verdadera (TPR) de cada clase y Balanced Accuracy,

$$Tasa\ positiva\ verdadera(TPR) = \frac{TP}{TP + FN}$$

$$Tasa\ negativa\ verdadera(TNR) = \frac{TN}{TN + FP}$$

$$Balanced\ Accuracy\ = \frac{TPR + \mathrm{TNR}}{2}$$

4.1.6. Implementacion del modelo

Durante esta etapa se realizaron las siguientes actividades:

- **Almacenamiento del modelo predictivo eficaz:** se preservó la información del modelo predictivo eficaz que permitió realizar predicciones sin la necesidad de volver a entrenar el modelo. Los pasos fueron los siguientes:

    - Generación de la persistencia del modelo creando un objeto arbitrario en una cadena de bits mediante la función ".dump()" especificando el nombre del archivo.
    - Verificación de la creación del archivo, cargando el archivo utilizando la funciona ".load()" con el nombre del archivo.

- Realización de una predicción utilizando la función ".predict()" con las variables independientes.

En la Figura 27 se muestra un ejemplo de la función JobLib, la cual proporciona un pipeline ligero para grandes cargas de datos en Python.

```
>>> import numpy as np
>>> import joblib
>>> obj = [('a', [1, 2, 3]), ('b', np.arange(10))]
>>> joblib.dump(obj, '/tmp/test.pkl')
['/tmp/test.pkl', '/tmp/test.pkl_01.npy']
>>> joblib.load('/tmp/test.pkl')
[('a', [1, 2, 3]), ('b', array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9]))]
```

Figura 25. Ejemplo de la función JobLib presentado en PyDataParis2016
Fuente: Propia

- **Elaboración del servicio Api Rest:** Una vez obtenido la persistencia del modelo, se requiere proveer un servicio de Api Rest a la universidad para que pueda ser utilizado en sus aplicaciones como su página web. Esto se simuló temporalmente en un servidor local para poder tener una visión más clara acerca del producto final. Los pasos fueron los siguientes:
  - Creación del entorno del Api Rest, definiendo las variables que se requieren dentro del Json.
  - Levantamiento del modelo mediante la función ".load()" con la ruta del archivo de la persistencia del modelo.
  - En la figura 28 se muestra que establecemos la media y varianza de cada variable continua de las diferentes carreras académicas que obtuvimos de la función de StandarScaler().

```
def obtener_standar(escuela):
    if escuela=="civil":
        array_means=[14.372510674637061, 1.4344833475661825,2.466865926558497,15.016920095756078,1.99
        array_vars=[6.96745996465975,3.991403127604849,5.986890775363339,2.605197671135698,7.81809762
        clf_from_joblib=load('model_civil.pkl')
    elif escuela=="sistemas":
        array_means=[14.098004147226543,1.5808657335406946,2.491575946086055,14.667581063865859,1.947
        array_vars=[8.459548478987466,6.416839323081901,9.106855888607516,2.739618832903685,7.0874607
        clf_from_joblib=load('model_sistemas.pkl')
    elif escuela=="arquitectura":
        array_means=[13.569535367545074,1.9527773925104024,3.0031865464632452,14.302243616667255,2.98
        array_vars=[8.32201656636356,8.0120827507233,12.504564013744107,2.6798187715025468,11.1529087
        clf_from_joblib=load('model_arquitectura.pkl')
    elif escuela=="alimentos":
        array_means=[15.090997008973082,1.7089631106679959,1.6542572283150552,15.209717087457252,1.22
        array_vars=[4.321966304078791,3.610442992657123,3.565111586874471,2.16123419803184,3.82524013
        clf_from_joblib=load('model_alimentos.pkl')
    elif escuela=="ambiental":
        array_means=[14.724427983539094,1.6893909465020576,2.0802139917695475,15.094775785208002,1.69
        array_vars=[6.902077237954919,4.525061741536689,6.399846373960608,2.4405332139868703,7.666247
        clf_from_joblib=load('model_ambiental.pkl')
    means=np.array(array_means)
    vars=np.array(array_vars)
    return means,vars,clf_from_joblib
```

Figura 26. Función que almacena la persistencia, mediana y varianza
Fuente: Propia

Con el propósito de estandarizar las variables continuas de un nuevo registro mediante la siguiente formula.

$$x = \frac{(value - means)}{vars^{0.5}}$$

Uso de las variables obtenidas del Json para realizar la predicción mediante la función ".predict()".

Devolución de la respuesta mediante un json.

En la figura 29 se muestra el esquema de un funcioamiento de un Servicio Api Rest, donde se visualiza los puntos explicados anteriormente.

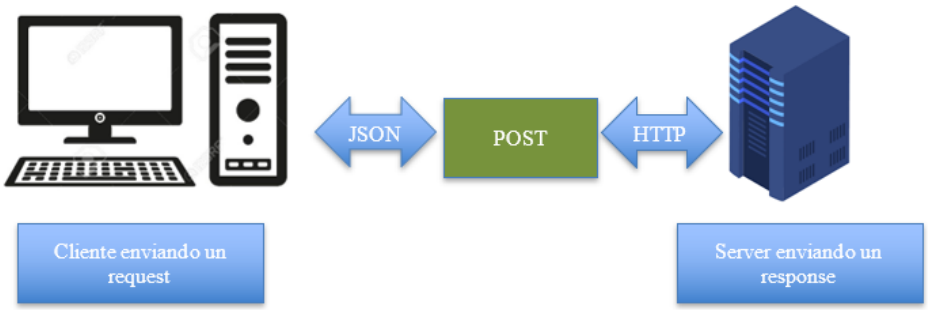


Figura 27. Funcionamiento de un Api Rest
Fuente: Propia

El código de elaboración del Api REST se encuentra en el Anexo 4

- **Elaboración de Recursos:** Finalmente se generó los siguientes entregables que especifican las actividades que el cliente debe realizar para instalar su servicio Api Rest:
    - Documento de instalación: Este documento incluye los códigos de las diferentes fases de la metodología CRISP-DM, teniendo paquetes necesarios para los procesos de instalación
    - Manual de usuario: Documento en el cual se describe el uso del API REST por parte del usuario de cómo debe utilizar el API REST

CAPÍTULO V. Resultados y Discusión

5.1. Resultados del Modelo Predictivo

5.1.1. Resultados de Análisis de los Datos Almacenados

Con respecto al primer objetivo planteado, se tienen los siguientes resultados de la comprensión de los datos iniciales y de su respectiva preparación para su posterior evaluación de la calidad de datos.

En la tabla 9 se muestra el diccionario de datos que consiste en los resultados de la transformación de las variables originales a las variables finales. En un principio se tuvieron 13 variables originales, y a partir de ella, se constituyó el diccionario de datos comprendido por las 26 variables finales con su respectivo tipo de variable y descripción. Estas nuevas variables aportaron al desarrollo del modelo predictivo para robustecer los resultados. (Anexo 1)

Tabla 9. Diccionario de Datos

Factores	Variables Originales	Variable Finales	Tipo de Variable	Descripción
Personal	Contratos, EGRESO	EdadUltimoCiclo	Continua	Edad con la que el estudiante finalizo el ciclo.
	Genero	Genero	Categórica	El género del estudiante del alumno
	Religion	Religión	Categórica	Si es adventista del séptimo dia o no
	Nacionalidad	Nacionalidad	Categórica	Si es un estudiante nacional o un extranjero.
Académica	Escuela	Escuela	Categórica	La escuela al cual pertenece el Estudiante.
	Cursos Desaprobados	curso_repetitivo_1	Continua	El numero de cursos desaprobados 1 vez
		curso_repetitivo_2	Continua	El numero de cursos desaprobados 2 veces.
		curso_repetitivo_3	Continua	El numero de cursos desaprobados 3 veces.
		curso_repetitivo_4	Continua	El numero de cursos desaprobados 4 veces.
		curso_repetitivo	Continua	El numero de cursos desaprobados de 2 a mas veces.
		numero_generales	Continua	El numero de curso tipo general desaprobados

		numero_especificos	Continua	El numero de cursos tipo especificos desaprobados.
		numero_especialidad	Continua	El numero de cursos tipo especialidad desaprobados.
		nro_desaprobados_total	Continua	El numero de cursos desaprobados
	Contratos y Dni	cont_junt	Continua	El numero de contratos del estudiante.
		nro_esc_traslado_ant	Continua	El numero de veces que el estudiante se traslado interno.
		nro_contra_ant	Continua	El numero de contratos del estudiante anteriores a la Carrera actual.
		cont_contratos_total	Continua	El numero de Contratos del estudiante dentro de la Universidad.
	Retirados, Notas y Contratos	Retirado	Categórica	Si se retiro en el ciclo academico.
	Contratos y Cursos Desaprobados	nro_desaprobados_ant	Continua	El numero de cursos desaprobados. anteriormente a la Carrera Actual
	Notas	incremento_nota	Continua	Si la diferencia entre la nota anterior con la actual es positiva se acumula.
		desincremento_nota	Continua	Si la diferencia entre la nota anterior con la actual es negativa se acumula.
		nota_final	Continua	La ultima nota del ciclo academico.
	Vacacionales	Vacacionales	Continua	Número de veces que ha estudiado en las vacaciones
Financiera	Debe	Debe	Continua	Acumulación de dinero que tiene el saldo disponible del estudiante.
	Credito	Credito	Continua	Acumulación de dinero que tiene deuda el estudiante.

5.1.2. Resultados de la identificación de las técnicas de aprendizaje

Con respecto al segundo objetivo planteado, se tienen los resultados de la evaluación de cada técnica de aprendizaje supervisado y sus respectivos parámetros, con los que finalmente se pudo identificar a los mejores, para cada carrera de la facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima.

Se consideraron utilizar las técnicas de clasificación k-nearest neighbours (KNN), naive bayes, decisión tree, random forest y Gradient Boosting (XGBOOST) por escuela profesional, asimismo se buscó los mejores hiperparametros mediante la búsqueda de cuadrícula validada cruzada sobre una cuadrícula de parámetros bajo la métrica F1-Macro, también se obtuvo los parámetros para la validación cruzada y los resultados de los hiperparametros de cada técnica, a continuación se detalla los resultados de la matriz de confusión con los hiperparametros:

A) Ingeniería de Sistemas

En la Tabla 10 se muestra los resultados de la matriz de confusión de Ingenieria de Sistemas con las métricas balanced_accuracy y tpr de cada modelo desarrollado.

Tabla 10. Matriz de Confusion por Algoritmo - Ingenieria de Sistemas

Algoritmos	Variable Predictiva	Desertor	No Desertor	Support
KNN	Desertor (1)	92	94	186
	No Desertor (0)	178	872	1050
	Accuracy	0.66	TPR	0.49
Naïve de Bayes	Desertor (1)	109	77	186
	No Desertor (0)	225	825	1050
	Accuracy	0.69	TPR	0.59
Decision Tree	Desertor (1)	52	134	186
	No Desertor (0)	39	1011	1050
	Accuracy	0.62	TPR	0.28
Random Forest	Desertor (1)	55	131	186
	No Desertor (0)	23	1027	1050
	Accuracy	0.64	TPR	0.3
XGBOOST	Desertor (1)	57	129	186
	No Desertor (0)	28	1022	1050
	Accuracy	0.64	TPR	0.31

Con respecto al modelo KNN, los resultados muestran que de los 186 estudiantes clasificados como “Desertor” predijo correctamente en un 0.49, mientras que el “balanced_accuray” muestra la tasa de verdaderos promedios entre las dos clases siendo un 0.66; por otro lado el modelo Naive Bayes predijo correctamente el 0.59, mientras que el

"balanced_accuray" del modelo es de 0.69; de la misma manera el modelo de Decision Tree predijo correctamente el 0.28 como desertor, mientras que la tasa de verdaderos promedios es de 0.62; asimismo el modelo Random Forest predijo correctamente el 0.30 como "Desertor" y tuvo un "balanced_accuray" de 0.64; finalmente el modelo XGBOOST predijo correctamente el 0.31 como "Desertor", mientras que el "balanced_accuray" es de un 0.64. Analizando las variables se consideró que los modelos analizados se ordenaron de mayor a menor de la siguiente manera: Naive Bayes, KNN, XGBOOST, Random Forest y Decision Tree

B) Ingeniería Civil

En la Tabla 11 se muestra los resultados de la matriz de confusión de Ingenieria Civil con las métricas balanced_accuracy y tpr de cada modelo desarrollado.

Tabla 11. Matriz de Confusion por Algoritmo - Ingenieria Civil

Algoritmos	Variable Predictiva	Desertor	No Desertor	Support
KNN	Desertor (1)	30	48	78
	No Desertor (0)	87	840	927
	Accuracy	0.65	TPR	0.38
Naïve de Bayes	Desertor (1)	34	44	78
	No Desertor (0)	90	837	927
	Accuracy	0.67	TPR	0.44
Decision Tree	Desertor (1)	25	53	78
	No Desertor (0)	16	911	927
	Accuracy	0.65	TPR	0.32
Random Forest	Desertor (1)	15	63	78
	No Desertor (0)	7	920	927
	Accuracy	0.59	TPR	0.19
XGBOOST	Desertor (1)	17	61	78
	No Desertor (0)	10	917	927
	Accuracy	0.60	TPR	0.21

Con respecto al modelo KNN, los resultados muestra que de los 78 estudiantes clasificados como "Desertor" predijo correctamente el 0.38, mientras que el "balanced_accuray" muestra la tasa de verdaderos promedios entre las dos clases siendo un

0.65; por otro lado el modelo Naive Bayes predijo correctamente el 0.44, mientras que el "balanced_accuray" del modelo es de 0.67; de la misma manera el modelo Decision Tree predijo correctamente el 0.32 como Desertor, mientras que la tasa de verdaderos promedios es de 0.65; asimismo el modelo Random Forest predijo correctamente el 0.19 como Desertor y tuvo un "balanced_accuray" es de un 0.59; finalmente el modelo XGBOOST predijo correctamente el 0.21, mientras que el "balanced_accuray" es de un 0.60. Analizando las variables se consideró que los modelos analizados se ordenaron de mayor a menor de la siguiente manera: **Naive Bayes, KNN, Decision Tree**, XGBOOST y Random Forest

C) Ingeniería Alimentos

En la tabla 12 se muestra los resultados de la matriz de confusión de Ingenieria de Alimentos con las métricas balanced_accuracy y tpr de cada modelo desarrollado.

Tabla 12. Matriz de Confusión por Algoritmo – Ingeniera de Alimentos

Algoritmos	Variable Predictiva	Desertor	No Desertor	Support
KNN	Desertor (1)	17	20	37
	No Desertor (0)	33	361	394
	Accuracy	0.69	TPR	0.46
Naïve de Bayes	Desertor (1)	16	21	37
	No Desertor (0)	26	368	394
	Accuracy	0.68	TPR	0.43
Decision Tree	Desertor (1)	11	26	37
	No Desertor (0)	7	387	394
	Accuracy	0.64	TPR	0.29
Random Forest	Desertor (1)	11	26	37
	No Desertor (0)	7	387	394
	Accuracy	0.64	TPR	0.29
XGBOOST	Desertor (1)	13	24	37
	No Desertor (0)	7	387	394
	Accuracy	0.67	TPR	0.35

Con respecto al modelo KNN, los resultados muestra que de los 37 estudiantes clasificados como "Desertor" predijo correctamente el 0.46, mientras que el "balanced_accuray" muestra la tasa de verdaderos promedios entre las dos clases siendo un 0.69; por otro lado el modelo Naive Bayes predijo correctamente el 0.43, mientras que el

"balanced_accuray" del modelo es de 0.68; de la misma manera el modelo Decision Tree predijo correctamente el 0.29 como Desertor, mientras que la tasa de verdaderos promedios es de 0.64; asimismo el modelo Random Forest predijo lo mismo que el modelo mencionado anteriormente; finalmente el modelo XGBOOST predijo correctamente el 0.35, mientras que el "balanced_accuray" es de un 0.67. Analizando las variables se consideró que los modelos analizados se ordenaron de mayor a menor de la siguiente manera: **KNN, Naive Bayes, XGBOOST,** Decision Tree y Random Forest**.**

D) Ingeniería Ambiental

En la tabla 13 se muestra los resultados de la matriz de confusión de Ingenieria Ambiental con las métricas balanced_accuracy y tpr de cada modelo desarrollado

Tabla 13. Matriz de Confusión por Algoritmo - Ingenieria Ambiental

Algoritmos	Variable Predictiva	Desertor	No Desertor	Support
KNN	Desertor (1)	25	59	84
	No Desertor (0)	117	1362	1479
	Accuracy	0.61	TPR	0.3
Naïve de Bayes	Desertor (1)	31	53	84
	No Desertor (0)	100	1379	1479
	Accuracy	0.65	TPR	0.37
Decision Tree	Desertor (1)	16	68	84
	No Desertor (0)	11	1468	1479
	Accuracy	0.59	TPR	0.19
Random Forest	Desertor (1)	14	70	84
	No Desertor (0)	10	1469	1479
	Accuracy	0.58	TPR	0.17
XGBOOST	Desertor (1)	17	67	84
	No Desertor (0)	15	1464	1479
	Accuracy	0.60	TPR	0.20

Con respecto al modelo KNN, los resultados muestra que de los 84 estudiantes clasificados como "Desertor" predijo correctamente el 0.30, mientras que el "balanced_accuray" muestra la tasa de verdaderos promedios entre las dos clases siendo un 0.61; por otro lado el modelo Naive Bayes predijo correctamente el 0.37, mientras que el "balanced_accuray" del modelo es de 0.65; de la misma manera el modelo Decision Tree

predijo correctamente el 0.19 como Desertor, mientras que la tasa de verdaderos promedios es de 0.59; asimismo el modelo Random Forest predijo correctamente el 0.17 como Desertor y tuvo un "balanced_accuray" es de un 0.58; finalmente el modelo XGBOOST predijo correctamente el 0.20, mientras que el "balanced_accuray" es de un 0.60. Analizando las variables se consideró que los modelos analizados se ordenaron de mayor a menor de la siguiente manera: **Naive Bayes, KNN XGBOOST,** Decision Tree y Random Forest**.**

E) Arquitectura

En la tabla 14 se muestra los resultados de la matriz de confusión de Arqiotectira con las métricas balanced_accuracy y tpr de cada modelo desarrollado

Tabla 14. Matriz de Confusión por Algoritmo – Arquitectura

Algoritmos	Variable Predictiva	Desertor	No Desertor	Support
KNN	Desertor (1)	92	94	186
	No Desertor (0)	178	872	1050
	Accuracy	0.66	TPR	0.49
Naïve de Bayes	Desertor (1)	109	77	186
	No Desertor (0)	225	825	1050
	Accuracy	0.69	TPR	0.59
Decision Tree	Desertor (1)	52	134	186
	No Desertor (0)	39	1011	1050
	Accuracy	0.62	TPR	0.28
Random Forest	Desertor (1)	55	131	186
	No Desertor (0)	23	1027	1050
	Accuracy	0.64	TPR	0.30
XGBOOST	Desertor (1)	57	129	186
	No Desertor (0)	28	1022	1050
	Accuracy	0.64	TPR	0.31

Con respecto al modelo KNN, los resultados muestra que de los 186 estudiantes clasificados como "Desertor" predijo correctamente el 0.49, mientras que el "balanced_accuray" muestra la tasa de verdaderos promedios entre las dos clases siendo un 0.66; por otro lado el modelo Naive Bayes predijo correctamente el 0.59, mientras que el "balanced_accuray" del modelo es de 0.69; de la misma manera el modelo Decision Tree

predijo correctamente el 0.28 como Desertor, mientras que la tasa de verdaderos promedios es de 0.62; asimismo el modelo Random Forest predijo correctamente el 0.30 como Desertor y tuvo un "balanced_accuray" es de un 0.64; finalmente el modelo XGBOOST predijo correctamente el 0.31, mientras que el "balanced_accuray" es de un 0.64. Analizando las variables se consideró que los modelos analizados se ordenaron de mayor a menor de la siguiente manera: **Naive Bayes, KNN XGBOOST,** Decision Tree y Random Forest**.**

5.1.3. Resultados respecto a la Validación del Modelo

Con respecto al tercer objetivo planteado, fue necesario utilizar la data de entrenamiento para aplicar la curva de validación, con el objetivo de identificar la existencia de sobreajustes en los diferentes modelos aplicados. Las figuras que se muestran a continuación, evidencia el comportamiento del sobreajuste de cada uno de los modelos con el paramentro más influyente por carrera. A continuación, se detalla los resultados de la curva de validación (Anexo 2).

Curva de Validación

A) Ingenieria de Sistemas

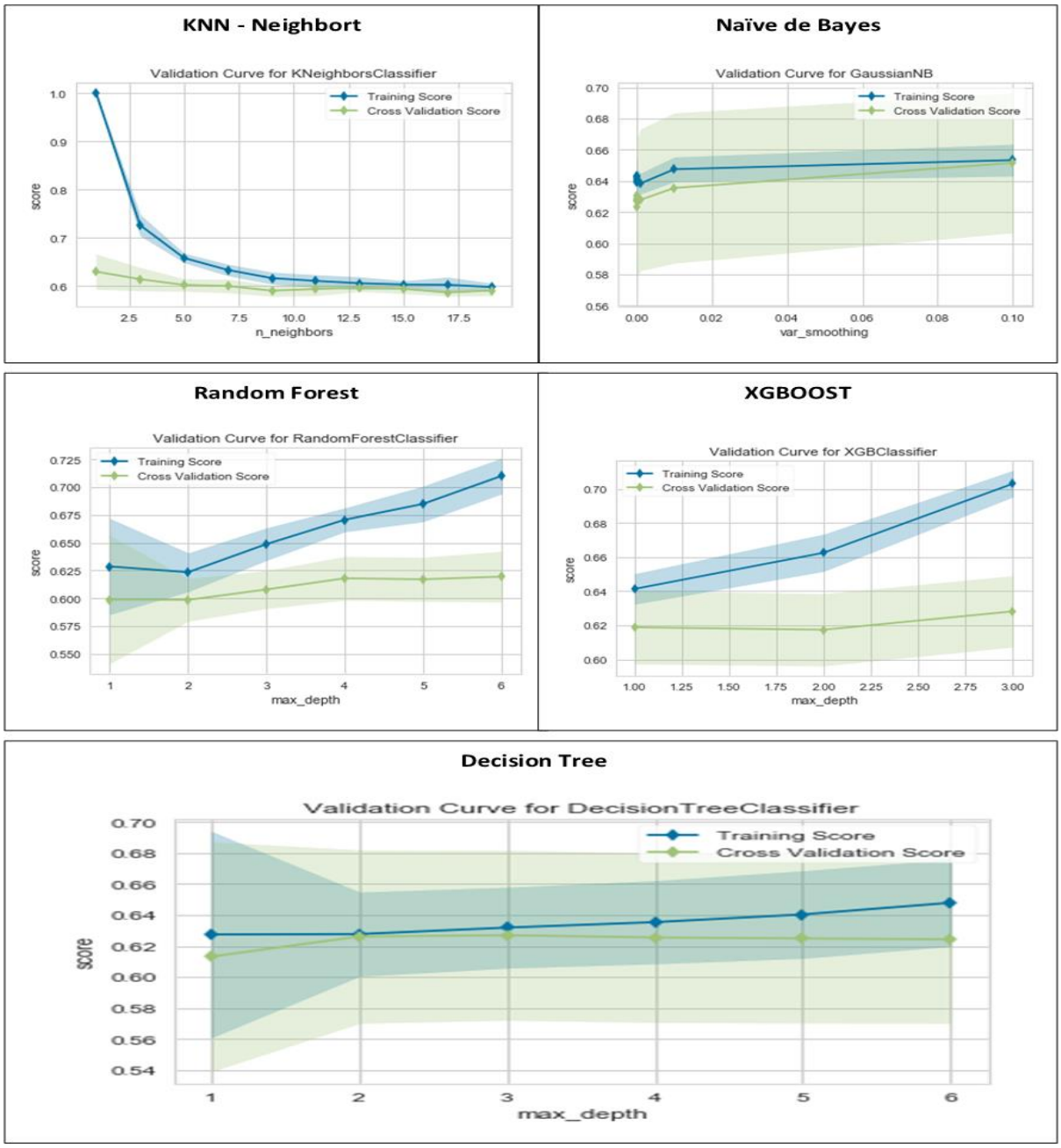


Figura 28. Curva de Validación aplicados a los algoritmos del modelo - Ing. de Sistemas
Fuente: Propia

En la Figura 30 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento. En el algoritmo KNN, se visualiza que a través parámetro n_neighbour, a menor cantidad de grupos, los valores de score son más altos, pero

se muestra un alto sobreajuste entre las dos curvas. En cambio, a partir de 9 grupos de adelante, los valores de score son más bajos, pero se muestra un subajuste entre las dos curvas. En el algoritmo Naive Bayes, se visualiza que a través parámetro var_smoothing, no existe sobreajuste ni subajuste, los valores de score son aceptables, y existe una mínima variabilidad entre las dos curvas. Los siguientes tres algoritmos hacen uso del parámetro max_depth, de modo que en el algoritmo Random Forest, se visualiza un score aceptable con baja variabilidad entre las curvas y que partir del valor 2 en adelante, aumenta su variabilidad mostrando un sobreajuste del modelo. Asimismo, el modelo XGBOOST muestra un score aceptable, pero a la vez un sobreajuste en todos sus valores; finalmente el modelo Decision Tree muestra un score aceptable en toda la gráfica, pero solo una variabilidad aceptable al inicio, porque a partir del valor 1 en adelante, se presenta una alta variabilidad.

B) Ingenieria Civil

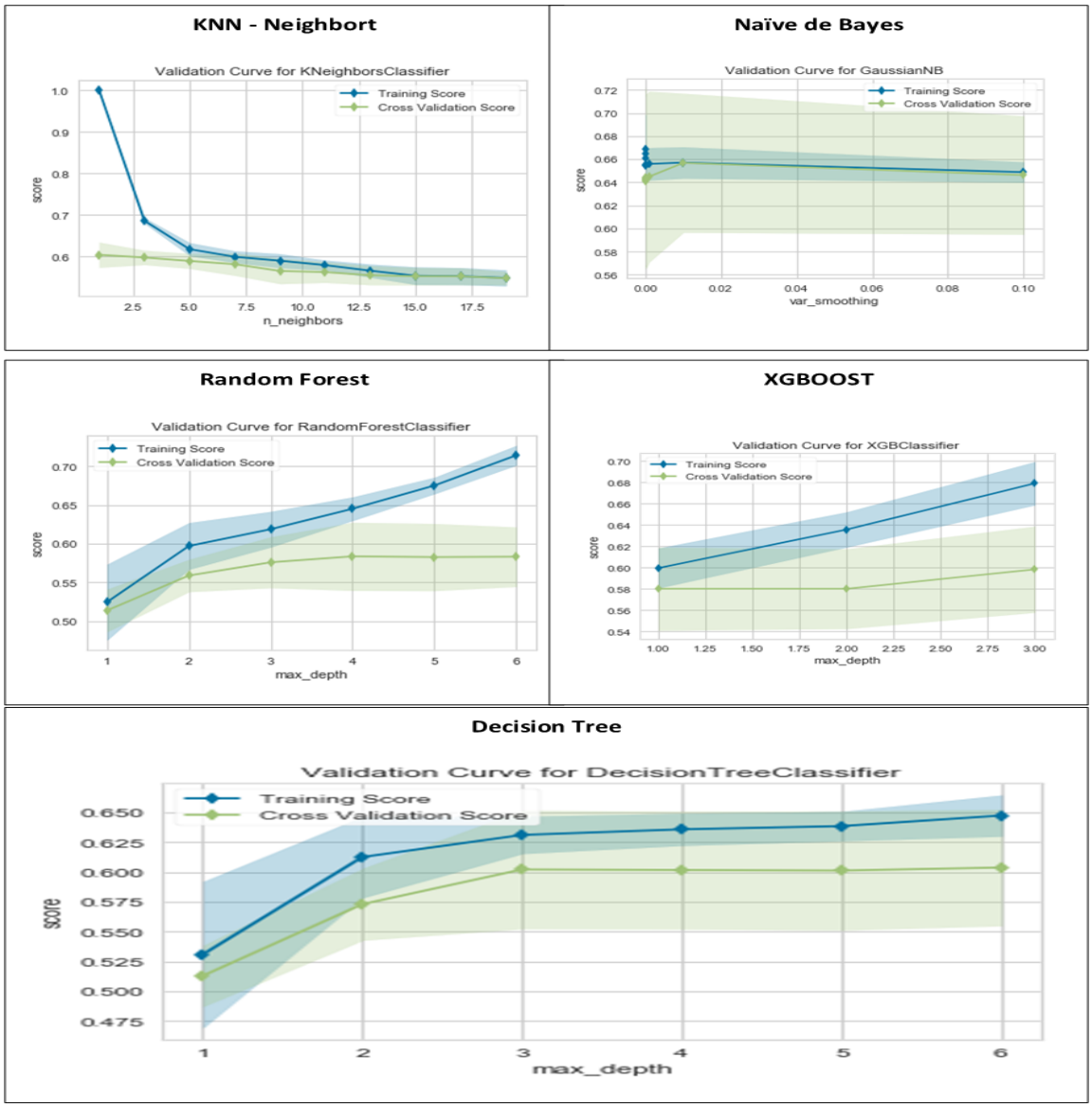


Figura 29. Curva de Validación aplicados a los algoritmos del modelo - Ing. Civil
Fuente: Propia

En la Figura 31 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento. En el algoritmo KNN, se visualiza que a través parámetro n_neighbour, a menor cantidad de grupos, los valores de score son más altos, pero se muestra un alto sobreajuste entre las dos curvas. En cambio, a partir de 9 grupos de adelante, los valores de score son más bajos, pero se muestra un subajuste entre las dos curvas. En el algoritmo Naive Bayes, se visualiza que a través parámetro var_smoothing, no existe sobreajuste ni subajuste, los valores de score son aceptables, y existe una mínima

variabilidad entre las dos curvas. Los siguientes tres algoritmos hacen uso del parámetro max_depth, de modo que en el algoritmo Random Forest, se visualiza un score aceptable con baja variabilidad entre las curvas y que partir del valor 5 en adelante, aumenta su variabilidad mostrando un sobreajuste del modelo. Asimismo, el modelo XGBOOST muestra un score aceptable, pero a la vez un sobreajuste en todos sus valores; finalmente el modelo Decision Tree muestra un score aceptable en toda la gráfica, pero solo una variabilidad aceptable al inicio, porque a partir del valor 5 en adelante, se presenta una alta variabilidad.

C) Ingenieria de Alimentos

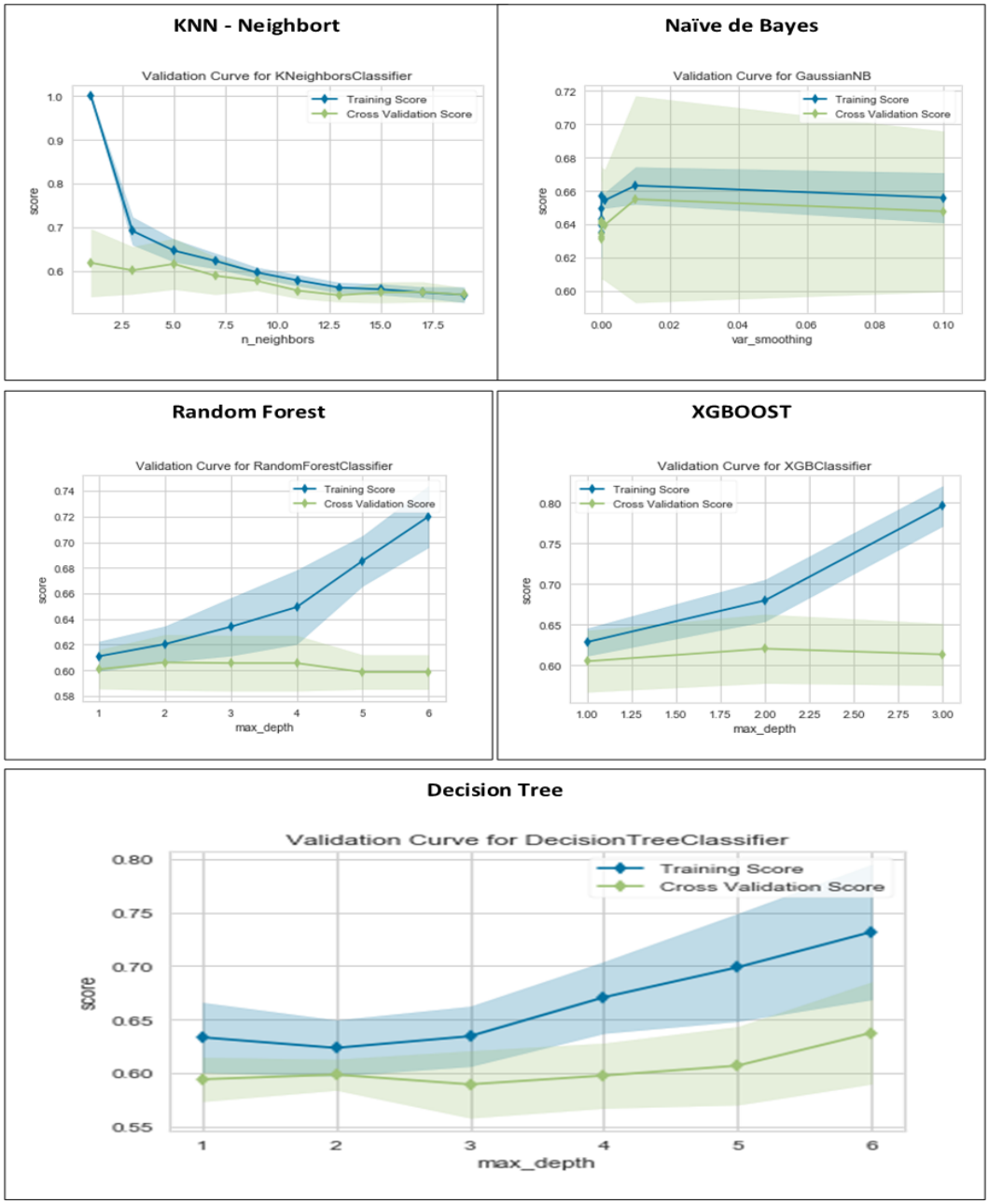


Figura 30. Curva de Validación aplicados a los algoritmos del modelo - Ing. Alimentos
Fuente: Propia

En la Figura 32 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento. En el algoritmo KNN, se visualiza que a través parámetro n_neighbour, a menor cantidad de grupos, los valores de score son más altos, pero se muestra un alto sobreajuste entre las dos curvas. En cambio, a partir de 9 grupos de adelante, los valores de score son más bajos, pero se muestra un subajuste entre las dos

curvas. En el algoritmo Naive Bayes, se visualiza que a través parámetro var_smoothing, no existe sobreajuste ni subajuste, los valores de score son aceptables, y existe una mínima variabilidad entre las dos curvas. Los siguientes tres algoritmos hacen uso del parámetro max_depth, de modo que en el algoritmo Random Forest, se visualiza un score aceptable con baja variabilidad entre las curvas y que partir del valor 2 en adelante, aumenta su variabilidad mostrando un sobreajuste del modelo. Asimismo, el modelo XGBOOST muestra un score aceptable, pero a la vez un sobreajuste en todos sus valores; finalmente el modelo Decision Tree muestra un score aceptable en toda la gráfica, pero solo una variabilidad aceptable al inicio, porque a partir del valor 2 en adelante, se presenta una alta variabilidad.

D) Ingenieria Ambiental

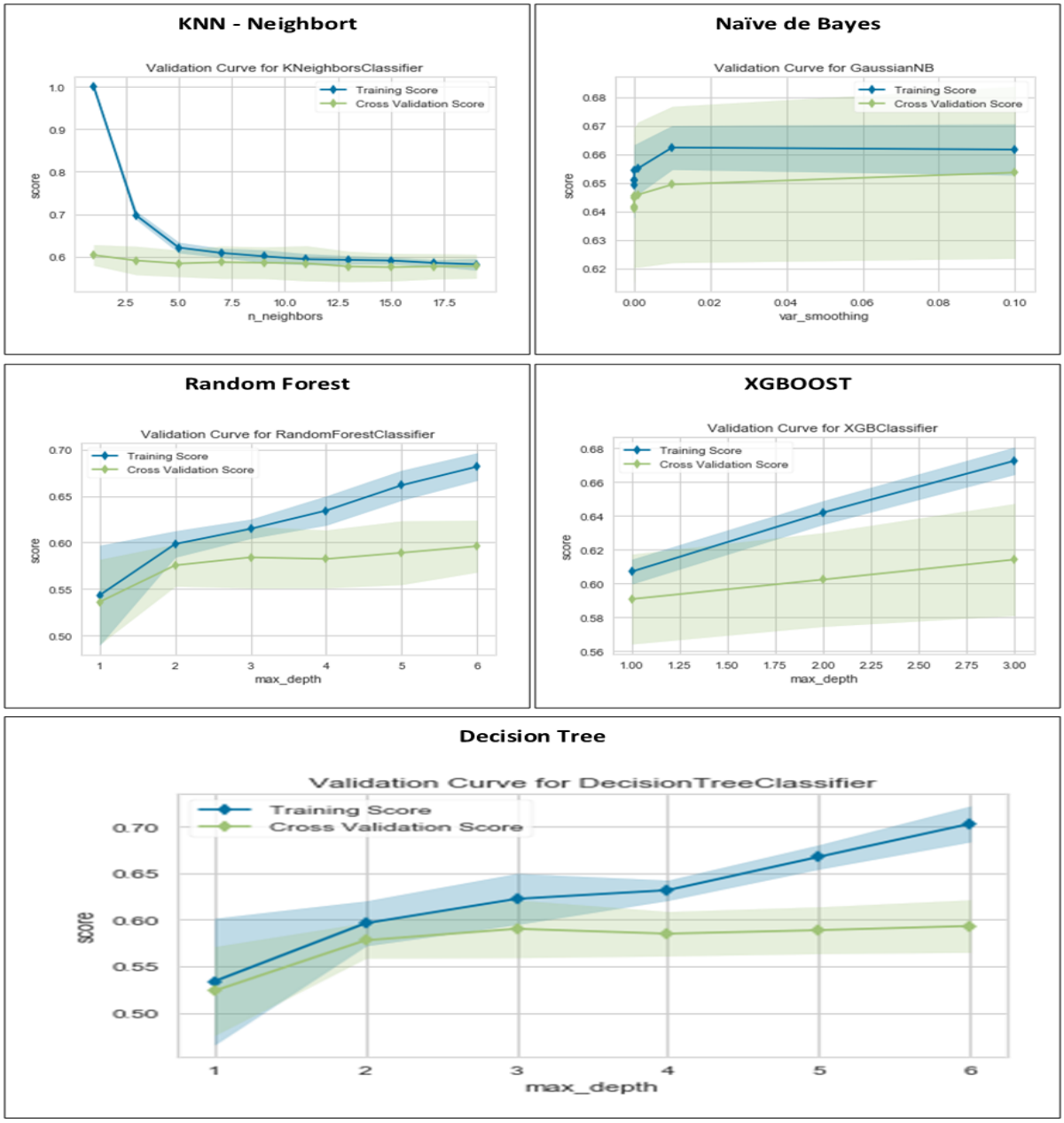


Figura 31. Curva de Validación aplicados a los algoritmos del modelo - Ing. Ambiental
Fuente: Propia

En la Figura 33 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento. En el algoritmo KNN, se visualiza que a través parámetro n_neighbour, a menor cantidad de grupos, los valores de score son más altos, pero se muestra un alto sobreajuste entre las dos curvas. En cambio, a partir de 7 grupos de adelante, los valores de score son más bajos, pero se muestra un subajuste entre las dos

curvas. En el algoritmo Naive Bayes, se visualiza que a través parámetro var_smoothing, no existe sobreajuste ni subajuste, los valores de score son aceptables, y existe una mínima variabilidad entre las dos curvas. Los siguientes tres algoritmos hacen uso del parámetro max_depth, de modo que en el algoritmo Random Forest, se visualiza un score aceptable con baja variabilidad entre las curvas y que partir del valor 2 en adelante, aumenta su variabilidad mostrando un sobreajuste del modelo. Asimismo, el modelo XGBOOST muestra un score aceptable, pero a la vez un sobreajuste en todos sus valores; finalmente el modelo Decision Tree muestra un score aceptable en toda la gráfica, pero solo una variabilidad aceptable al inicio, porque a partir del valor 2 en adelante, se presenta una alta variabilidad.

E) Arquitectura

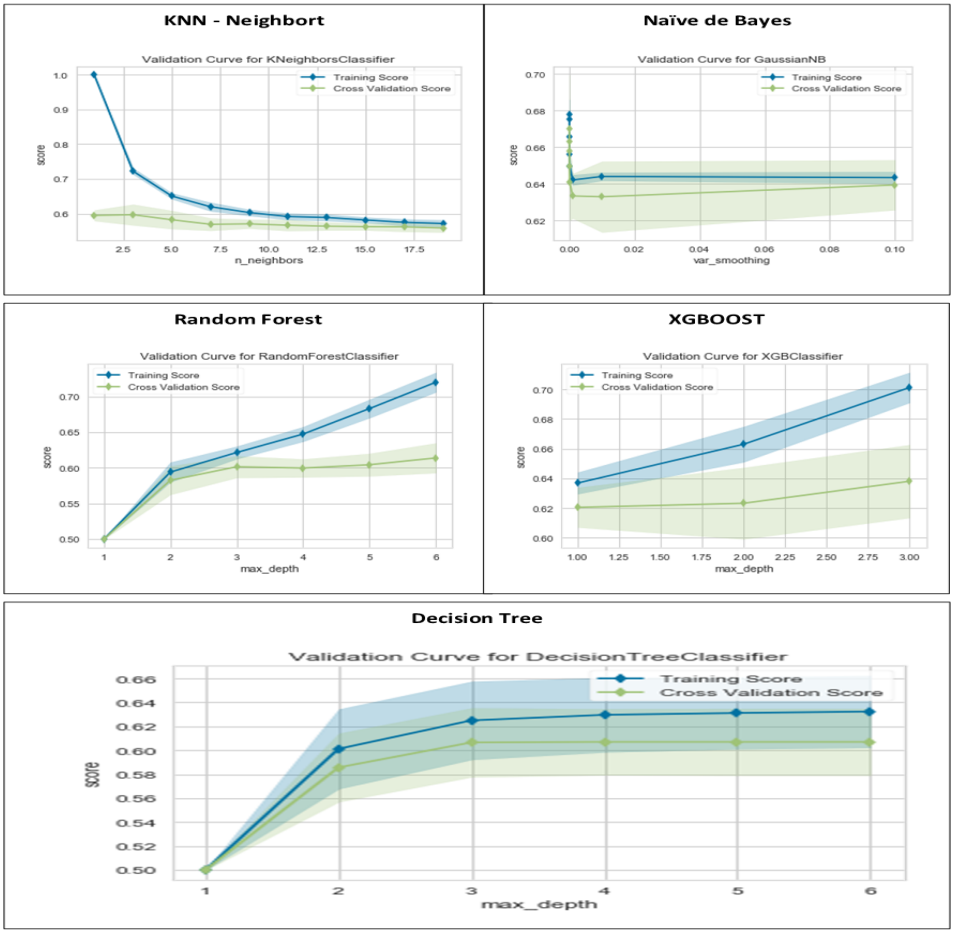


Figura 32. Curva de Validación aplicados a los algoritmos del modelo – Arquitectura
Fuente: Propia

En la Figura 34 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento. En el algoritmo KNN, se visualiza que a través parámetro n_neighbour, a menor cantidad de grupos, los valores de score son más altos, pero se muestra un alto sobreajuste entre las dos curvas. En cambio, a partir de 9 grupos de adelante, los valores de score son más bajos, pero se muestra un subajuste entre las dos curvas. En el algoritmo Naive Bayes, se visualiza que a través parámetro var_smoothing, no existe sobreajuste ni subajuste, los valores de score son aceptables, y existe una mínima variabilidad entre las dos curvas. Los siguientes tres algoritmos hacen uso del parámetro max_depth, de modo que en el algoritmo Random Forest, se visualiza un score aceptable con baja variabilidad entre las curvas y que partir del valor 2 en adelante, aumenta su variabilidad mostrando un sobreajuste del modelo. Asimismo, el modelo XGBOOST muestra un score aceptable apartir del varlo 1 en adelante; finalmente el modelo Decision Tree muestra un score aceptable en toda la gráfica, pero solo una variabilidad aceptable al inicio, porque a partir del valor 5 en adelante, se presenta una alta variabilidad.

Curva de Aprendizaje

Esta actividad se realiza con el objetivo de evaluar si existe un sobreajuste de los datos a través de una gráfica que muestra los puntajes en función del tamaño de los datos de entrenamiento.

A) Ingeniería de Sistemas

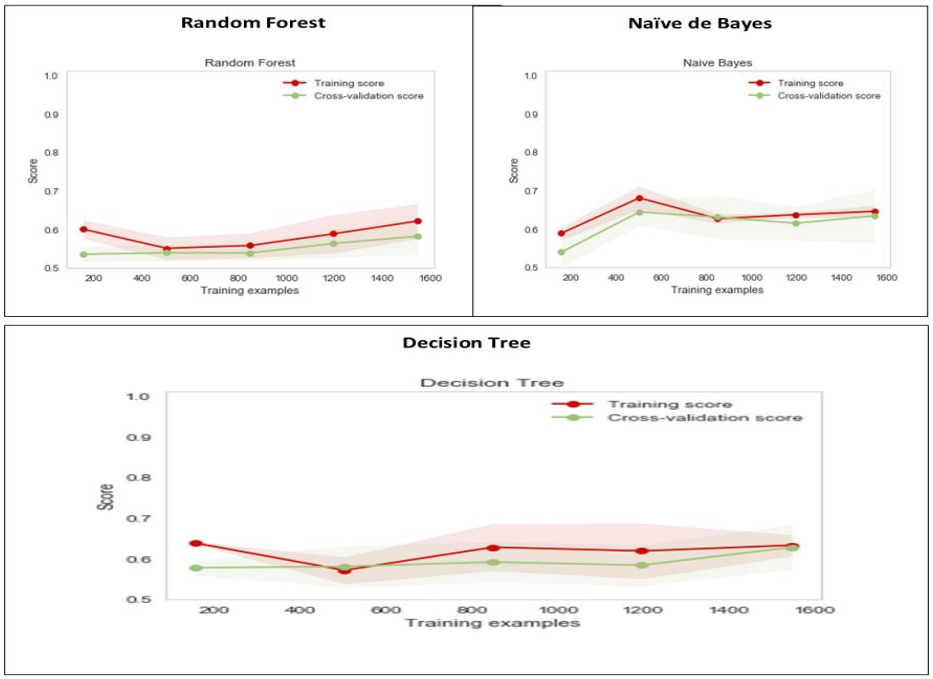


Figura 33. Curva de Aprendizaje aplicados a los algoritmos mas influyentes del modelo - Ing. Sistemas

Fuente: Propia

En la Figura 35 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento a partir de los datos de entrenamiento En el algoritmo Random Forest, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio existe una alta variabilidad entre las curvas, y a partir de los 500 datos de entrenamiento en adelante, el score aumenta y a su vez, disminuye la variabilidad. En el algoritmo Naive de Bayes, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio hay una alta variabilidad entre las curvas, y a partir de los 500 datos de entrenamiento en adelante, el score disminuye ligeramente, pero hay menos variabilidad. Finalmente, en el algoritmo Decision Tree, se visualiza en general, un score aceptable para toda la gráfica y una alta variabilidad al inicio de las curvas; a partir de los 500 datos de entrenamiento en adelante, el score se mantiene estable y existe una poca variabilidad.

B) Ingenieria Civil

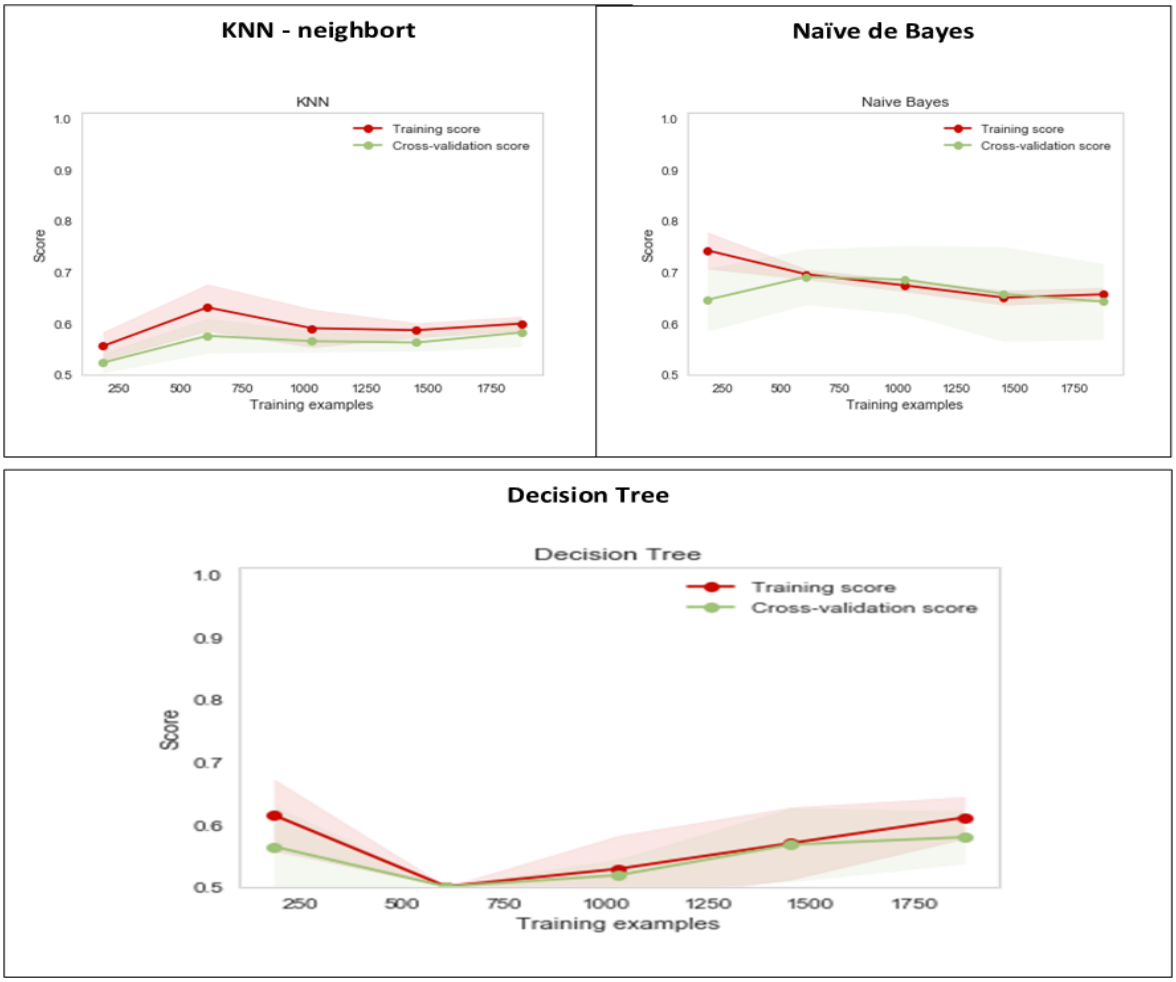


Figura 34. Curva de Aprendizaje aplicados a los algoritmos mas influyentes del modelo - Ing. Civil

Fuente: Propia

En la Figura 36 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento a partir de los datos de entrenamiento En el algoritmo Random Forest, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio existe una alta variabilidad entre las curvas, y a partir de los 1000 datos de entrenamiento en adelante, el score aumenta y a su vez, disminuye la variabilidad. En el algoritmo Naive de Bayes, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio hay una alta variabilidad entre las curvas, y a partir de los 600 datos de entrenamiento en adelante, el score disminuye ligeramente, pero hay menos variabilidad. Finalmente, en el algoritmo Decision Tree, se visualiza en general, un score aceptable para

toda la gráfica y una alta variabilidad al inicio de las curvas; a partir de los 600 datos de entrenamiento en adelante, el score se mantiene estable y existe una poca variabilidad.

C) Ingenieria de Alimentos

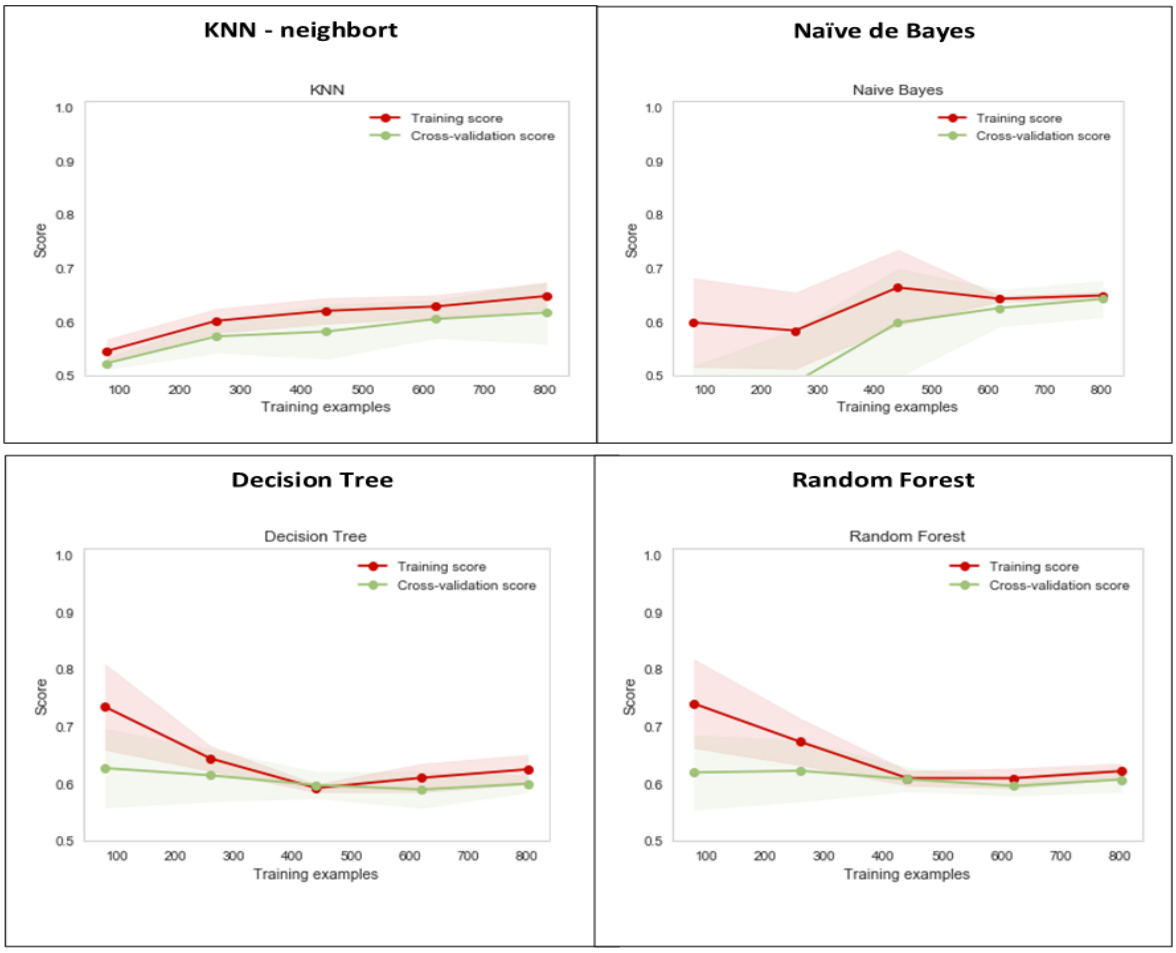


Figura 35. Curva de Aprendizaje aplicados a los algoritmos mas influyentes del modelo - Ing. Alimentos

Fuente: Propia

En la Figura 37 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento a partir de los datos de entrenamiento En el algoritmo Random Forest, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio existe una alta variabilidad entre las curvas, y a partir de los 100 datos de entrenamiento en adelante, el score aumenta y a su vez, disminuye la variabilidad. En el algoritmo Naive de Bayes, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio hay una alta variabilidad entre las curvas, y a partir de los 600 datos de entrenamiento en adelante, el score disminuye ligeramente, pero hay menos variabilidad.

Finalmente, en el algoritmo Decision Tree, se visualiza en general, un score aceptable para toda la gráfica y una alta variabilidad al inicio de las curvas; a partir de los 300 datos de entrenamiento en adelante, el score se mantiene estable y existe una poca variabilidad. Finalmente, el modelo Random Forest se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio hay una alta variabilidad entre las curvas; y a partir de los 400 datos de entrenamiento en adelante, el score disminuye ligeramente, pero hay menos variabilidad.

D) Ingenieria Ambiental

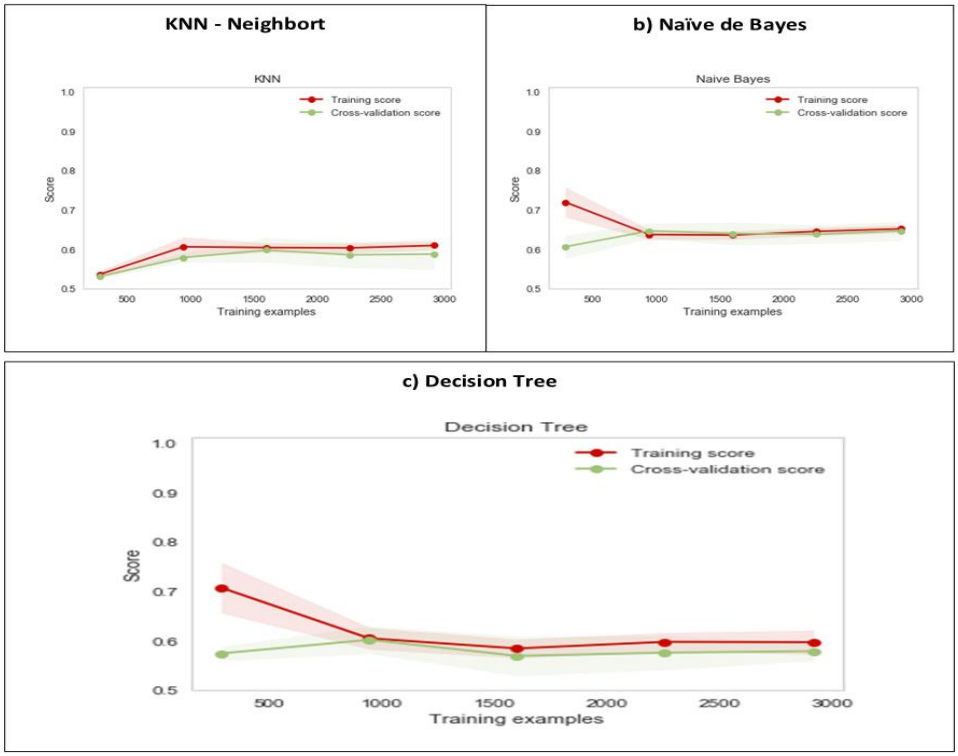


Figura 36. Curva de Aprendizaje aplicados a los algoritmos mas influyentes del modelo - Ing. Ambiental

Fuente: Propia

En la Figura 38 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento a partir de los datos de entrenamiento En el algoritmo Random Forest, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio existe una alta variabilidad entre las curvas, y a partir de los 1000 datos de entrenamiento en adelante, el score aumenta y a su vez, disminuye la variabilidad. En el algoritmo Naive de Bayes, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio hay una alta variabilidad entre las curvas, y a partir de los 900 datos

de entrenamiento en adelante, el score disminuye ligeramente, pero hay menos variabilidad. Finalmente, en el algoritmo Decision Tree, se visualiza en general, un score aceptable para toda la gráfica y una alta variabilidad al inicio de las curvas; a partir de los 900 datos de entrenamiento en adelante, el score se mantiene estable y existe una poca variabilidad.

E) Arquitectura

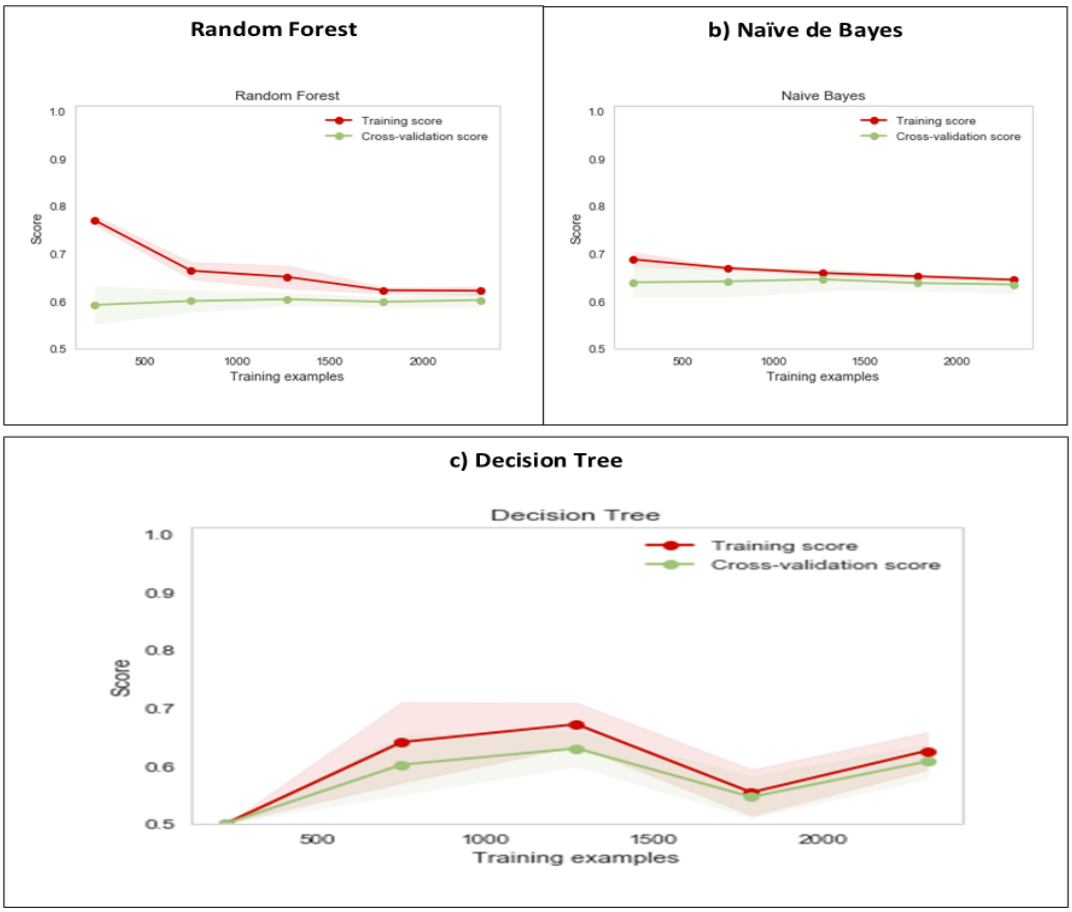


Figura 37. Curva de Aprendizaje aplicados a los algoritmos mas influyentes del modelo – Arquitectura

Fuente: Propia

En la Figura 39 se observa, para todos los gráficos, el comportamiento de la curva de validación y la curva de entrenamiento a partir de los datos de entrenamiento En el algoritmo Random Forest, se visualiza en general, un score aceptable para toda la gráfica, sin embargo, al inicio existe una alta variabilidad entre las curvas, y a partir de los 1700 datos de entrenamiento en adelante, el score aumenta y a su vez, disminuye la variabilidad. En el algoritmo Naive de Bayes, se visualiza en general, un score aceptable para toda la gráfica,

sin embargo, al inicio hay una alta variabilidad entre las curvas, y a partir de los 750 datos de entrenamiento en adelante, el score disminuye ligeramente, pero hay menos variabilidad. Finalmente, en el algoritmo Decision Tree, se visualiza en general, un score aceptable para toda la gráfica y una alta variabilidad al inicio de las curvas; a partir de los 750 datos de entrenamiento en adelante, el score se mantiene estable y existe una poca variabilidad.

A partir de los algoritmos definidos con sus respectivos parámetros se realizó diferentes predicciones con o sin técnicas de balanceo de datos. Para visualizar los resultados mediante una matriz de confusión y obtener las métricas balanced_accuracy (Accuracy Balanceado) y el TPR (Ratio de Verdadero Positivos).

A) Ingenieria de Sistemas

Tabla 15. Matriz de Confusion de los algoritmos mas influyente al modelo - Ing. Sistemas

Algoritmos	Técnica	Variable Predictiva	Desertor	No Desertor	Support
Naive Bayes	SmoteTomek	**Desertor (1)**	43	46	89
		No Desertor (0)	109	630	739
		Balanced Accuracy	0.67	**TPR**	0.48
Decision Tree	OverSampling	**Desertor (1)**	70	19	89
		No Desertor (0)	255	484	739
		Balanced Accuracy	0.72	**TPR**	0.79
Random Forest	SmoteTomek	**Desertor (1)**	67	22	89
		No Desertor (0)	153	586	739
		Balanced Accuracy	0.77	**TPR**	0.75

La Tabla 15 muestran resultados del mejor resultado de cada algoritmo junto con la técnica de balanceo de datos. El resultado del algoritmo Naive Bayes con la técnica SmoteTomek predijo correctamente el 0.48 como Desertor y tuvo un “balanced_accuracy” es de un 0.67; por otro lado el algoritmo Decision Tree con la técnica OverSampling predijo correctamente el 0.79 como Desertor y tuvo un “balanced_accuracy” es de un 0.72; finalmente el algoritmo Random Forest con la técnica SmoteTomek predijo correctamente el 0.75 como Desertor y tuvo un “balanced_accuracy” es de un 0.77. Analizando las métricas se ordenaron los algoritmos de mayor a menor de la siguiente manera: Random Forest, Decision Tree y Naive Bayes.

B) Ingenieria Civil

Tabla 16. Matriz de Confusion de los algoritmos mas influyente al modelo - Ing. Civil

Algoritmos	Tecnica	Variable Predictiva	Desertor	No Desertor	Support
KNN	UnderSampling	Desertor (1)	52	26	78
		No Desertor (0)	220	707	927
		Balanced Accuracy	0.71	TPR	0.67
Naive Bayes	SmoteTomek	Desertor (1)	42	36	78
		No Desertor (0)	114	813	927
		Balanced Accuracy	0.71	TPR	0.54
Decision Tree	UnderSampling	Desertor (1)	55	23	78
		No Desertor (0)	258	669	927
		Balanced Accuracy	0.71	TPR	0.71

La Tabla 16 muestran resultados del mejor resultado de cada algoritmo junto con la técnica de balanceo de datos. El resultado del algoritmo Naive Bayes con la técnica SmoteTomek predijo correctamente el 0.67 como Desertor y tuvo un “balanced_accuracy” es de un 0.71; por otro lado el algoritmo Decision Tree con la técnica OverSampling predijo correctamente el 0.54 como Desertor y tuvo un “balanced_accuracy” es de un 0.71; finalmente el algoritmo Random Forest con la técnica SmoteTomek predijo correctamente el 0.71 como Desertor y tuvo un “balanced_accuracy” es de un 0.71. Analizando las métricas se ordenaron los algoritmos de mayor a menor de la siguiente manera: Decision Tree, KNN y Naive Bayes.

C) Ingenieria de Alimentos

Tabla 17. Matriz de Confusion de los algoritmos mas influyente al modelo - Ing. Alimentos

Algoritmos	Tecnica	Variable Predictiva	Desertor	No Desertor	Support
KNN	UnderSampling	Desertor (1)	28	9	37
		No Desertor (0)	114	280	394
		Balanced Accuracy	0.73	TPR	0.76
Naive Bayes	SmoteTomek	Desertor (1)	19	18	37
		No Desertor (0)	25	369	394
		Balanced Accuracy	0.51	TPR	0.73
Decision Tree	OverSampling	Desertor (1)	26	11	37
		No Desertor (0)	139	255	394
		Balanced Accuracy	0.68	TPR	0.7
Random Forest	OverSampling	Desertor (1)	26	11	37
		No Desertor (0)	113	281	394
		Balanced Accuracy	0.71	TPR	0.7

La Tabla 17 muestran resultados del mejor resultado de cada algoritmo junto con la técnica de balanceo de datos. El resultado del algoritmo KNN con la técnica UnderSampling predijo correctamente el 0.76 como Desertor y tuvo un “balanced_accuracy” es de un 0.73; por otro lado el algoritmo Naive Bayes con la técnica SmoteTomek predijo correctamente el 0.73 como Desertor y tuvo un “balanced_accuracy” es de un 0.51; de la misma manera el algoritmo Decision Tree con la técnica OverSampling predijo correctamente el 0.70 como Desertor y tuvo un “balanced_accuracy” es de un 0.68; finalmente el algoritmo Random Forest con la técnica OverSampling predijo correctamente el 0.70 como Desertor y tuvo un “balanced_accuracy” es de un 0.71. Analizando las métricas se ordenaron los algoritmos de mayor a menor de la siguiente manera: KNN, Random Forest, Decision Tree y Naive Bayes.

D) Ingenieria Ambiental

Tabla 18. Matriz de Confusion de los algoritmos mas influyente al modelo - Ing. Ambiental

Algoritmos	Tecnica	Variable Predictiva	Desertor	No Desertor	Support
KNN	UnderSampling	**Desertor (1)**	58	26	84
		No Desertor (0)	339	1140	1479
		Balanced Accuracy	0.73	**TPR**	0.69
Naive Bayes	SmoteTomek	**Desertor (1)**	40	44	84
		No Desertor (0)	122	1357	1479
		Balanced Accuracy	0.70	**TPR**	0.48
Decision Tree	OverSampling	**Desertor (1)**	47	37	84
		No Desertor (0)	218	1261	1479
		Balanced Accuracy	0.71	**TPR**	0.56

La Tabla 18 muestran resultados del mejor resultado de cada algoritmo junto con la técnica de balanceo de datos. El resultado del algoritmo KNN con la técnica UnderSampling predijo correctamente el 0.69 como Desertor y tuvo un "balanced_accuracy" es de un 0.73; por otro lado el algoritmo Naive Bayes con la técnica SmoteTomek predijo correctamente el 0.48 como Desertor y tuvo un "balanced_accuracy" es de un 0.70; finalmente el algoritmo Decision Tree con la técnica OverSampling predijo correctamente el 0.56 como Desertor y tuvo un "balanced_accuracy" es de un 0.71. Analizando las métricas se ordenaron los algoritmos de mayor a menor de la siguiente manera: KNN, Decision Tree y Naive Bayes.

E) Arquitectura

Tabla 19. Matriz de Confusion de los algoritmos mas influyente al modelo – Arquitectura

Algoritmos	Tecnica	Variable Predictiva	Desertor	No Desertor	Support
KNN	UnderSampling	**Desertor (1)**	58	26	84
		No Desertor (0)	339	1140	1479
		Balanced Accuracy	0.73	**TPR**	0.69
Naive Bayes	SmoteTomek	**Desertor (1)**	40	44	84
		No Desertor (0)	122	1357	1479
		Balanced Accuracy	0.70	**TPR**	0.48
Decision Tree	OverSampling	**Desertor (1)**	47	37	84
		No Desertor (0)	218	1261	1479
		Balanced Accuracy	0.71	**TPR**	0.56

La Tabla 19 muestran resultados del mejor resultado de cada algoritmo junto con la técnica de balanceo de datos.El resultado del algoritmo KNN con la técnica SmoteTomek predijo correctamente el 0.48 como Desertor y tuvo un “balanced_accuracy” es de un 0.67; por otro lado el algoritmo Decision Tree con la técnica OverSampling predijo correctamente el 0.79 como Desertor y tuvo un “balanced_accuracy” es de un 0.72; finalmente el algoritmo Random Forest con la técnica SmoteTomek predijo correctamente el 0.75 como Desertor y tuvo un “balanced_accuracy” es de un 0.77. Analizando las métricas se ordenaron los algoritmos de mayor a menor de la siguiente manera: Random Forest, Decision Tree y Naive Bayes.

Siendo que el mayor puntaje obtenido en la evaluación de cada algoritmo junto con una técnica de balanceo de datos fue considerado el Modelo Predictivo Eficaz.

Finalmente, se obtuvó las variables finales dependiendo si el modelo predictivo eficaz de las carreras haya sido un Decision, Random Fores y XGBOOST (Ver Anexo 3).

5.1.4. Resultados respecto a la Implementación del Modelo

Con respecto al cuarto objetivo planteado, en la Figura 40 se tienen los resultados de la implementación del modelo en una API REST, en donde se encuentran todas las persistencias del modelo predictivo eficaz para cada programa académico.

```
    df = pd.DataFrame({"RETIRADO": [retirado],
                       "curso_repetitivo_3": [curs_rep3],
                       "count_array_retiro_desof": [retirado_count],
                       "count_array_trunc": [desertor_count],
                       "CONT_JUNT": [change_sede],
                       "curso_repetitivo_4": [curs_rep4],
                       "NACIONALIDAD": [nacionalidad],
                       "GENERO": [genero],
                       "nro_esc_traslado_ant": [esc_traslado],
                       "nro_contra_ant": [contra_ant],
                       "nro_desaprobados_ant": [desaprobado_ant],
                       "RELIGION": [religion],
                       "NOTA_FINAL": [nota_final],
                       "incremento_nota": [incremento_nota],
                       "desincremento_nota": [des_incremento_nota],
                       "PROMEDIO": [promedio],
                       "curso_repetitivo_1": [curs_rep1],
                       "curso_repetitivo_2": [curs_rep2],
                       "curso_repetitivo": [curs_rep],
                       "numero_generales": [n_generales],
                       "numero_especificos": [n_especificos],
                       "numero_especialidad": [n_especialidad],
                       "NRO_DESAPROBADOS_TOTAL": [desaprobado_total],
                       "CONT_CONTRATOS_TOTAL": [contra_total],
                       "debe_total": [debe],
                       "credito_total": [credito],
                       "Edad": [edad_ultimo]
                       })

    df[numeric_feature_cols] = scale_data(df[numeric_feature_cols],means_ , vars_)

    pred2=clf_from_joblib_.predict(df)

    data={'Tipo de Desertor':str(pred2[0])}

    return jsonify(data)
```

Figura 38. Código sobre la predicción del estudiante en el Api Rest
Fuente: Propia

Finalmente, en la Figura 41 se muestran los resultados del API REST, para el cual es necesario abrir el terminal de Windows e insertar un código para enviar un Json con las variables para realizar predicciones.

```
C:\Users\HectorHS>curl -i -X POST -H "Content-Type: application/json" -d "{\"carrera\": \"civil\",\"genero\": \"0\",\"re
tirado_count\": \"0\",\"desertor_count\": \"1\",\"change_sede\": \"1\",\"esc_traslado\": \"0\",\"contra_ant\": \"0\",\"d
esaprobado_ant\": \"0\",\"contra_total\":\"5\",\"desaprobado_total\":\"14\",\"adventista\": \"1\",\"retirado\": \"1\",\"
nacionalidad\": \"1\",\"promedio\": \"12.0225\",\"debe\": \"5832.8900\",\"credito\": \"0\",\"inc_nota\": \"4.9\",\"des_i
ncnota\": \"12.61\",\"nota_final\": \"3.82\",\"edad_ultimo_ciclo\": \"19\",\"curs_rep1\": \"9\",\"curs_rep2\": \"1\",\"c
urs_rep3\": \"1\",\"curs_rep4\": \"0\",\"curs_rep\": \"2\",\"curso_generales\": \"5\",\"curso_especificos\": \"2\",\"cur
so_especialidad\": \"7\" }" http://127.0.0.1:5000/student
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 37
Server: Werkzeug/1.0.0 Python/3.7.7
Date: Fri, 11 Sep 2020 02:33:34 GMT

{
  "Tipo de Desertor": "DESERTOR"
}
```

Figura 39. Data enviada en formato JSON al sistema
Fuente: Propia

5.2. Resultado contraste a la Hipotesis

Los resultados de los diferentes modelos eficaces de cada carrera fueron desplegados en un Api Rest y a partir de ello se realizó predicciones con el objetivo de realizar un contraste de hipotesis con los datos desde el año 2018-1 hasta el 2019-1 de los estudiantes de la Facultad de Ingeniería y Arquitectura, siendo una data de 381.

Tabla 20. Resultados de las predicciones con la data de contraste de hipotesis

N° de Contratos	Carrera	Verdaderos Desertores	Falsos Desertores	Verdadero No Desertores	Falsos No Desertores
1	Ing.Alimentos	5	0	8	2
	Ing.Ambiental	9	1	6	4
	Arquitectura	8	2	8	2
	Ing.Civil	7	3	9	1
	Ing.Sistemas	6	4	8	2
2	Ing.Alimentos	7	3	6	4
	Ing.Ambiental	8	2	6	4
	Arquitectura	8	2	9	1
	Ing.Civil	5	5	9	1
	Ing.Sistemas	8	2	8	2
3	Ing.Alimentos	7	2	9	1
	Ing.Ambiental	6	1	8	2
	Arquitectura	6	4	8	2
	Ing.Civil	2	2	6	4
	Ing.Sistemas	6	4	7	3
4	Ing.Alimentos	7	2	9	1
	Ing.Ambiental	7	3	7	3
	Arquitectura	7	3	7	3
	Ing.Civil	7	3	9	1
	Ing.Sistemas	5	2	7	3
TOTAL		131	50	154	46

A partir del total de los *Veraderos Desertores*, *Falsos Desertores*, *Veraderos No Desertores y Falsos No Desertores* que se muestran en la Tabla 20, obtenemos los valores para completar una matriz de confusión y obtener las métricas de “TPR” y “Balanced Accuracy”

Tabla 21. Matriz de Confusion del contraste de hipotesis

Variable Predictiva	Desertor	No Desertor	Support
Desertor (1)	131	50	186
No Desertor (0)	46	154	1050
Balanced Accuracy	0.75	**TPR**	0.72

Los resultados mostrados en la Tabla 2, indica que de los 168 estudiantes clasificados como “Desertor” predijo correctamente el 0.72, mientras que el “Balanced Accuray” muestra la tasa de verdaderos promedios entre las dos clases siendo un 0.75.

5.3. Discusión

De acuerdo a los resultados de la presente investigación, solo se pudo identificar las variables más influyentes en el modelo final, para las escuelas de ingeniería de sistemas, ingeniería de alimentos e ingeniería civil. De las otras dos escuelas (arquitectura e ingeniería ambiental), no se pudo identificar ello dado que los algoritmos aplicados no tienen la característica de obtener como resultado las variables más importantes.

Los resultados para las tres primeras escuelas ya mencionadas indican que la nota final, el número de contratos y la religión fueron las variables con más influencia en el modelo. Por su parte Manco [6] obtuvo en su investigación que las variables más influyentes fueron número de contratos en los cuatro semestres consecutivos a su ingreso, promedio de su segunda matrícula, edad de ingreso, promedio en su cuarta matrícula, año de ingreso, semestre de ingreso y número de cursos aprobados en su primera matrícula. Manco usó 10 variables iniciales de las cuales 7 fueron las más representativas, en tanto que en esta investigación se usaron 20 variables iniciales, pero solo 3 fueron las más representativas, de modo que se puede deducir que Manco obtuvo mejores patrones que muestran la realidad de la deserción de estudiantes.

De las 3 variables con mayor influencia en la deserción de estudiantes, la más predominante fue la nota final. Tal como lo indica García [43] en su investigación, el rendimiento académico en los cursos es un predictor muy importante en la trayectoria universitaria del estudiante, incluso menciona que las notas de los primeros ciclos son aún más influyentes en la deserción estudiantil. Asimismo, Pariona [84] destacan en su

publicación, que el bajo rendimiento académico está asociado a la deserción estudiantil. Tang y Shao [1] también usaron el desempeño académico de estudiantes como atribuciones de entrada para el modelo productivo.

La siguiente variable con mayor influencia es el número de contratos, es decir, la cantidad de ciclos en el que los alumnos se matriculan pero que no necesariamente lo terminan. Usualmente se ha observado que varios estudiantes que se matriculan a inicios de ciclo, no permanecen hasta concluirlo por diversos factores ya sean problemas económicos, de salud, distancia, entre otros. Esta situación que causa la pérdida del ciclo y por tanto una siguiente matrícula, hace que el número de contratos se acumulen más y más, lo cual puede ser agotador para el estudiante que finalmente opta por desertar de la universidad al no contar con los recursos que deben ser afrontados por ellos mismos o sus apoderados. Finalmente, la tercera variable con mayor influencia hace referencia a la religión, el cual tiene que ver más con un tema social. Muchos estudiantes desarrollan su identidad institucional cuando participan de las actividades extracurriculares que contribuye a su permanencia en el centro de estudios. [85]

En cuanto al uso de técnicas de aprendizaje supervisado de clasificación, Manco [6] solo usó la técnica Decisión Tree de RapidMiner. En cambio, en este estudio, se usó la misma técnica Decisión Tree de RapidMiner y 4 técnicas más: KNN, Naive Bayes, Random Forest y XGBOOST. Cabe resaltar que, a pesar de que Manco obtuvo buenos resultados aplicando una sola técnica, existe la posibilidad de que aplicando las otras, Manco haya podido encontrar mejores resultados. Las mejoras en las técnicas de los algoritmos de aprendizaje automático también pueden ayudar a aumentar la precisión de la predicción. Tang y Shao [1] hicieron uso de un modelo único para la predicción en este estudio, sin embargo, también afirman que mientras que el uso del algoritmo integrado de múltiples modelos ayudará a mejorar la precisión hasta cierto punto.

Para el caso de la presente investigación, el haber usado más de un modelo predictivo se debe a que cada escuela académica tiene una realidad distinta en el que el comportamiento de los estudiantes varía según su contexto. Además de ello, se debe a que los algoritmos tienen sus respectivos grados de complejidad que permiten seleccionar el que mejor se ajusta

a los datos analizados. Por esta razón, es recomendable usar un algoritmo menos complejo cuando existe riesgo de sobreajuste.

Un claro ejemplo de ello es la diferencia que hay en los resultados de los algoritmos más relevantes para cada escuela académica. Por ejemplo, para ingeniería de sistemas, ingeniería de alimentos e ingeniería civil, se usaron los algoritmos más complejos (decision tree, random forest y xgboost); en tanto que, para ingeniería ambiental y arquitectura, se usó el menos complejo (KNN) para poder obtener mejor precisión en la predicción. Tal como concluyeron Borges, Marques y Bernardino [86] en su investigación, no existe una sola herramienta o técnica que sea la mejor en cualquier situación, ya que, el valor de precisión debe considerar lo que se espera de un clasificador en un escenario de aplicación de caso real; mientras que un escenario puede requerir una alta precisión, otro puede aceptar una baja precisión.

En cuanto a la precisión, al comparar los resultados del modelo predictivo más eficaz de esta investigación con la de Mingjie [1] , se observa que el modelo DT para Mingjie tiene una precisión del 71.91% en tanto que el de la presente tesis, tiene una precisión del 71,27%, una diferencia mínima entre ambas. Esto se debe a que ambas técnicas usaron métricas similares en cuestión de casos donde existe un desbalance de datos entre dos clases. La métrica que usó Mingjie fue F-score, y la que se usó en la tesis fue Balanced Accuracy. Según Alaa Tharwat [87], el F-Score es la media armónica de precisión y recall de la clase positiva el cual es muy similar a Balanced Accuracy porque son métricas que se pueden usar frente a conjuntos de datos desequilibrados dado que es la media aritmética entre el recall de la clase positiva y negativa. Además de ello, el balanced accuracy es una métrica que permite obtener un modelo que no beneficie a la clase mayorista (no desertores), en tanto que, el TPR permite dar mayor prioridad a la clase minorista (desertores).

Al desglosar la precisión para cada escuela académica, se tiene que las tres mejores fueron las carreras de ingeniería de sistemas con 79%, ingeniería de alimentos con 76% e ingeniería civil con 71%. Tal como se vino explicando en párrafos anteriores, era de esperar que estas tres obtengan las mejores precisiones porque se usaron los modelos decision tree. Por el contrario, las carreras de ingeniería ambiental y arquitectura obtuvieron una precisión similar de 69%, evidentemente por haber usado el algoritmo KNN. Tal como se observa, la precisión

no ha superado el 80% lo cual puede deberse en gran manera a que los datos no estaban los suficientemente completos en gran parte de las variables, incluso a causa de aquel problema, se tuvieron que descartar variables. Este es un problema común en temas de machine learning. Por ejemplo, Tang y Shao [1] tuvieron los mismos problemas de precisión en la predicción porque las atribuciones de los datos no estaban lo suficientemente completas.

En efecto, dado que los datos de este estudio se obtuvieron de los sistemas de gestión académica de la misma universidad, el estudio está limitado por la amplitud de los datos adquiridos; por lo tanto, solo hemos utilizado las características personales y el rendimiento académico de los estudiantes como variables de entrada en el modelo predictivo, lo que tiene un impacto en la precisión de la predicción. Investigaciones relacionadas sugieren que hay muchos factores que afectan la deserción de los estudiantes y las diferencias pueden ser grandes debido a diferencias individuales. A pesar de que en Perú se vienen realizando cambios sociopolíticos, la deserción estudiantil es una problemática poco estudiada. La tendencia es que ello se de en los primeros ciclos académicos, aunque también darse en los últimos tal como lo demuestra la presente investigación. Este abandono que puede darse de manera voluntaria o por el bajo rendimiento académico, reprobación, que el estudiante presenta durante el transcurso de sus estudios, debe seguir siendo estudiado por las instancias correspondientes para las mejoras en la gestión universitaria. [8]

CAPÍTULO VI. Conclusiones y recomendaciones

6.1. Conclusiones

Conforme a los procedimientos, los datos y los resultados del trabajo de investigación se llego a las siguientes conclusiones:

Con respecto al objetivo general, se obtuvo que el nivel de eficacia del modelo de aprendizaje supervisado para el pronóstico de la deserción de estudiantes de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima, es de 0.77 para la carrera de Ingeniería de Sistemas con el algoritmo Decision Tree, 0.71 para Ingeniería Civil con el algoritmo Random Forest y 0.73 para las carreras de Ingeniería de Alimentos, Ingeniería Ambiental y Arquitectura con el algoritmo KNN. Por otro lado, se obtuvo un 0.75 de nivel de eficacia como resultado de desempeño del modelo predictivo implementado con una data de 381 registros que abarca 3 ciclos académicos.

Con respecto al primer objetivo, de las 13 variables y 3 factores consideradas en la base de datos inicial de los estudiantes del 2009-2019 de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión, se obtuvo un diccionario de datos de 26 variables finales manteniéndose los factores.

Con respecto al segundo objetivo, los algoritmos de aprendizaje Naive Bayes y Decision Tree fueron las más viables para todas las carreras de la Facultad de Ingeniería y Arquitectura de la Universidad Peruana Unión – Lima. El algoritmo KNN fue más viable para las carreras de Ingeniería Civil, Ingeniería de Alimentos, Ingeniería Ambiental, y el algoritmo Random Forest para las carreras de Ingeniería de Sistemas, Ingeniería de Alimentos y Arquitectura.

Con respecto al tercer objetivo, el algoritmo de aprendizaje supervisado KNN es el que se adecúa mejor a las carreras de Ingeniería de Alimentos, Ingeniería Ambiental y Arquitectura. El algoritmo Random Forest se adecúa mejor a la carrera de Ingeniería de Sistemas, y el algoritmo Decision Tree, se adecúa mejor a la carrera de Ingeniería Civil. La técnica de balanceo de datos que usaron los algoritmos KNN y Decision Tree fue el UnderSampling, y el algoritmo Random Forest, SmoteTomek.

Con respecto al cuarto objetivo se obtuvo el despliegue del modelo en el cual fue posible realizar la predicción sin necesidad de volver a entrenar el modelo y demostrando su funcionamiento mediante un ApiRest.

Finalmente se realizó una implementación del modelo predictivo para poner aprueba el modelo predictivo en una aplicación para la contrastación de la hipótesis.

6.2. Recomendaciones

Se recomienda lo siguiente:

- Ampliar la investigación utilizando variables psicológicas ya que en este estudio se adoleció de estos tipos de variables.
- Crear un área especializada que pueda tener acceso a realizar específicamente de Inteligencia de Negocios (BI) que nos permita tener acceso a la información, para realizar investigaciones futuras
- La Universidad debe aplicar una ingeniería inversa a su estructura de su base de datos para que la información que se requiera en un momento determinado sea mas preciso.
- Elaborar modelos predictivos para las otras facultades de la universidad ya que manejan realidades distintas.
- Implementar un modulo en un sistema que soporte el modelo y sea utilizable para que los directores puedan ver la data en tiempo real, asimismo le sirva como una herramienta de apoyo

REFERENCIAS

[1] M. Tan and P. Shao, “Prediction of student dropout in E-learning program through the use of machine learning method,” *Int. J. Emerg. Technol. Learn.*, vol. 10, no. 1, pp. 11–17, 2015, doi: 10.3991/ijet.v10i1.4189.

[2] V. M. Rodríguez, J. G. Campos, and J. P. Aguilera, “Modelo Predictivo para la Permanencia en la Educación Superior,” *Congr. CLABES*, vol. 0, no. 0, 2017, [Online]. Available: http://www.revistas.utp.ac.pa/index.php/clabes/article/view/1588/2326.

[3] R. Ferrer-Urbina, V. Karmelic-Pavlov, H. Beck-Fernández, and R. Valdivia-Pinto, “Un Modelo Predictivo de Fracaso/Éxito Academico a partir de indicadores de ingreso, en estudiantes de una universidad estatal del norte de Chile,” vol. 44, no. 1, pp. 23–29, 2019, [Online]. Available: https://www.interciencia.net/wp-content/uploads/2019/01/23-FERRER-44-01.pdf.

[4] S. M. Merchán and J. A. Duarte, “Analysis of Data Mining Techniques for Constructing a Predictive Model for Academic Performance,” *IEEE Lat. Am. Trans.*, vol. 14, no. 6, pp. 2783–2788, 2016, doi: 10.1109/TLA.2016.7555255.

[5] L. Kemper, “Predicting Student Dropout: a Machine Learning Approach,” *ResearchGate*, no. February, pp. 0–33, 2018, doi: 10.13140/RG.2.2.24863.87206.

[6] M. M. Caycho, “Modelo predictivo para la identificación de patrones de la deserción estudiantil en la Universidad Nacional Tecnológica de Lima Sur (UNTELS),” *Untelsciencia-Perú*, vol. 1, no. 4, pp. 4–17, 2015.

[7] H. E. PAJA, “Predicción de rendimiento académico mediante regresión y redes neuronales en los estudiantes de la escuela profesional de ingeniería estadística e informática de la universidad nacional del altiplano Puno. 2015,” 2017.

[8] A. C. Jiménez Chura, “ANÁLISIS PREDICTIVO PARA LOS PROCESOS DE ADMISIÓN DE LA UNIVERSIDAD NACIONAL DEL ALTIPLANO – PUNO,” *Factores Socioeconómicos Que Determ. El Comer. Informal En La Ciudad Juliaca, Caso “Mercado Mi Peru” 2018*, p. 113, 2016, [Online]. Available: http://repositorio.unap.edu.pe/bitstream/handle/UNAP/9408/Rosa_Enriquez_Yuca.pdf?sequence=1&isAllowed=y.

[9] O. Sifuentes Bitocchi, “Modelos predictivos de la deserción estudiantil en una

universidad privada peruana," *Ind. Data*, vol. 21, no. 2, p. 47, 2018, doi: 10.15381/idata.v21i2.15602.

[10] D. Trujillo Fernandez, "Aplicación de Metodologías Machine Learning en la Gestión de Riesgo de Crédito," *Esc. Tec. Super. Ing. Informaticos*, vol. 6, pp. 5–9, 2017.

[11] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science (80-.).*, vol. 349, no. 6245, pp. 255–260, 2015, doi: 10.1126/science.aaa8415.

[12] E. Alpaydin, *Introduction to Machine Learning Second Edition*. 2010.

[13] J. Galarza Hernández, "Reducción de dimensionalidad en Machine Learning.," *Univ. Politec. Val.*, 2017, [Online]. Available: https://riunet.upv.es/bitstream/handle/10251/92565/GALARZA - Reducción de dimensionalidad en Machine Learning. Diagnóstico de cáncer de mama bsado e....pdf?sequence=1.

[14] K. J. M. A. ARENAS, "Diseño de un sistema de clasificación de señales de tránsito vehicular utilizando redes neuronales convolucionales," pp. 1–102, 2016.

[15] T. Mehrya, Mohri; Afshin, Rostamizadeh; Ameet, *Foundation of Machine Learning*, vol. 20, no. 4. 2012.

[16] S. Shalev-Shwartz and S. Ben-David, *Understanding machine learning: From theory to algorithms*, vol. 9781107057. 2013.

[17] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, "Supervised Machine Learning: A Review of Classification Techniques," *Artif. Intell. Rev.*, vol. 26, pp. 159–190, 2006, doi: 10.1007/s10462-007-9052-3.

[18] J. Cárdenas, G. Olivares, and R. Alfaro, "Clasificación automática de textos usando redes de palabras," *Rev. signos*, vol. 47, no. 86, pp. 346–364, 2014, doi: 10.4067/S0718-09342014000300001.

[19] A. Pastor, M. Tutor, : Jesús, and J. Ruiz, "Modelo de predicción de la demanda eléctrica mediante regresión dinámica," 2015, [Online]. Available: http://oa.upm.es/43848/1/TFG_ALFREDO_PASTOR_MARTINEZ.pdf.

[20] P. Cunningham and S. J. Delany, "k-Nearest Neighbour Classifiers," *Mult. Classif. Syst.*, vol. 34, no. APRIL 2007, pp. 1–17, 2007, doi: 10.1016/S0031-3203(00)00099-6.

[21] M. del P. S. Zárate, “Detección de Patrones Psicolingüísticos para el Análisis de Lenguaje Subjetivo en Español,” 2017.

[22] M. A. Nanda, K. B. Seminar, D. Nandika, and A. Maddu, “A comparison study of kernel functions in the support vector machine and its application for termite detection,” *Inf.*, vol. 9, no. 1, 2018, doi: 10.3390/info9010005.

[23] D. De Inteligencia, E. T. S. D. I. Informática, U. Nacional, and D. Educación, “Abstract Support Vector Machine 1 Introducción,” no. November, pp. 1–27, 2016.

[24] I. Rish, “An empirical study of the naive Bayes classifier,” *Int. Jt. Conf. Artif. Intell. 2001 Work. Empir. Methods Artif. Intell.*, no. January, pp. 41–46, 2001.

[25] W.-Y. Loh, “Classification and regression trees,” *Classif. Regres. Trees*, vol. 1, no. January, pp. 14–23, 2011, doi: 10.1201/9781315139470.

[26] A. Cutler, D. R. Cutler, and J. R. Stevens, “Random forests,” *Ensemble Mach. Learn. Methods Appl.*, pp. 157–175, 2012, doi: 10.1007/9781441993267_5.

[27] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” 2016, doi: 10.1145/2939672.2939785.

[28] I. L. Cherif and A. Kortebi, “On using eXtreme Gradient Boosting (XGBoost) Machine Learning algorithm for Home Network Traffic Classification,” *IFIP Wirel. Days*, vol. 2019-April, pp. 1–6, 2019, doi: 10.1109/WD.2019.8734193.

[29] M. Edwards and D. R. Morse, “The potential for computer-aided identification in biodiversity research,” *Trends Ecol. Evol.*, vol. 10, no. 4, pp. 153–158, 1995, doi: 10.1016/S0169-5347(00)89026-6.

[30] N. Tangri, D. Ansell, and D. Naimark, “Predicting technique survival in peritoneal dialysis patients: Comparing artificial neural networks and logistic regression,” *Nephrol. Dial. Transplant.*, vol. 23, no. 9, pp. 2972–2981, 2008, doi: 10.1093/ndt/gfn187.

[31] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychol. Rev.*, vol. 65, no. 6, pp. 1–23, 1958, doi: 10.1098/rspb.1976.0087.

[32] P. J. Werbos, “Backpropagation Through Time: What It Does and How to Do It,” *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990, doi: 10.1109/5.58337.

[33] F. Murtagh, “Multilayer perceptrons for classification and regression,” *Neurocomputing*, vol. 2, no. 5–6, pp. 183–197, 1991, doi: 10.1016/0925-

2312(91)90023-5.

[34] M. Luckert and M. Schaefer-Kehnert, “Using machine learning methods for evaluating the quality of technical documents,” 2016, [Online]. Available: http://www.diva-portal.org/smash/record.jsf?pid=diva2:920202.

[35] Tan, Steinbach, and Kumar, “Data Mining Cluster Analysis : Basic Concepts and Algorithms,” pp. 1–82, 2010.

[36] A. Jaramillo and H. Paz, “Aplicación de Técnicas de Minería de Datos para Determinar las Interacciones de los Estudiantes en un Entorno Virtual de Aprendizaje,” *Rev. Tecnológica ESPOL – RTE*, vol. 28, no. 1, pp. 64–90, 2015, [Online]. Available: http://rte.espol.edu.ec/index.php/tecnologica/article/viewFile/351/229.

[37] O. Niakšu, “CRISP Data Mining Methodology Extension for Medical Domain,” *Balt. J. Mod. Comput.*, vol. 3, no. 2, pp. 92–109, 2015.

[38] M. A. Rendon Herrera and V. J. David Acosta, “Estudio sobre el estado de las soluciones ICT y de los casos practicos de aplicación de la mineria de datos a nivel mundial en al menos 5 casos representativos,” pp. 1–228, 2006.

[39] M. Natrella, *e-Handbook of Statistical Methods*. 2010.

[40] M. Komorowski, D. C. Marshall, J. D. Salciccioli, and Y. Crutain, “Secondary Analysis of Electronic Health Records,” *Second. Anal. Electron. Heal. Rec.*, no. September, pp. 1–427, 2016, doi: 10.1007/978-3-319-43742-2.

[41] H. He and Y. Ma, *Imbalanced Learning: Foundations, Algorithms, and Applications*, 1st ed. Wiley-IEEE Press, 2013.

[42] S. Agarwal, *Data mining: Data mining concepts and techniques*. 2014.

[43] S. García, J. Luengo, J. A. Sáez, V. López, and F. Herrera, “A survey of discretization techniques: Taxonomy and empirical analysis in supervised learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 4, pp. 734–750, 2013, doi: 10.1109/TKDE.2012.35.

[44] L. A. Muhammed, “Entropy-based Algorithm for Discretization,” 2011.

[45] A. Fernández, V. López, M. Galar, M. J. Del Jesus, and F. Herrera, “Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches,” *Knowledge-Based Syst.*, vol. 42, pp. 97–110, 2013, doi: 10.1016/j.knosys.2013.01.018.

[46] P. Mirski, R. Bernsteiner, and D. Radi, “Analytics in Human Resource Management the OpenSKIMR Approach,” *Procedia Comput. Sci.*, vol. 122, pp. 727–734, 2017, doi: 10.1016/j.procs.2017.11.430.

[47] AWS, “Amazon Machine Learning Guía del desarrollador,” p. 156, 2019, [Online]. Available: https://docs.aws.amazon.com/es_es/machine-learning/latest/dg/machinelearning-dg.pdf#binary-classification.

[48] H. M and S. M.N, “A Review on Evaluation Metrics for Data Classification Evaluations,” *Int. J. Data Min. Knowl. Manag. Process*, vol. 5, no. 2, pp. 01–11, 2015, doi: 10.5121/ijdkp.2015.5201.

[49] M. M. Álvarez Suárez, A. Caballero, P. E. Assistant, and G. P. Lechuga, “Análisis Multivariante: Clasificación, Organización Y Validación De Resultados,” *Fourth LACCEI Int. Lat. Am. Caribb. Conf. Eng. Technol.*, no. June, pp. 21–23, 2006, [Online]. Available: http://www.laccei.org/LACCEI2006-PuertoRico/Papers - pdf/EDU072_Alvarez.pdf.

[50] “Plotting Learning Curves — scikit-learn 0.24.0 documentation.” https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html (accessed Jan. 02, 2021).

[51] I. K. Hadihardaja, M. Cahyono, and I. Soekarno, “A Study of Hold-Out and K-Fold Cross Validation for Accuracy of Groundwater Modeling in Tidal Lowland Reclamation Using Extreme Learning Machine,” *2nd Int. Conf. Technol. Informatics, Manag. Eng. Environ.*, pp. 228–233, 2014.

[52] S. Yadav and S. Shukla, “Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification,” *Proc. - 6th Int. Adv. Comput. Conf. IACC 2016*, no. Cv, pp. 78–83, 2016, doi: 10.1109/IACC.2016.25.

[53] A. M. Molinaro, R. Simon, and R. M. Pfeiffer, “Prediction error estimation: A comparison of resampling methods,” *Bioinformatics*, vol. 21, no. 15, pp. 3301–3307, 2005, doi: 10.1093/bioinformatics/bti499.

[54] C. K. Park and D. G. Kim, “Cross-Validation,” *Curr. Futur. Manag. Brain Metastasis*, vol. 25, pp. 1–12, 2012, doi: 10.1159/000331070.

[55] G. Barrero, “Optimización de hiperparámetros de algoritmos machine learning usado para analisis de la calidad del software,” no. January, 2019, doi: 10.13140/RG.2.2.15055.74405.

[56] J. Vanderplas, *Python Data Science*. O'Reilly Media, Inc., 2006.

[57] G. J. Paramo and C. A. Correa, "Desercion Estudiantil Universitaria-Conceptualizacion." 1999.

[58] L. Patiño Garzón and A. Cardona Perez, María, "Revisión De Algunos Estudios Sobre La Deserción Estudiantil Universitaria En Colombia Y Latinoamérica," *Theoria*, vol. 21, no. 1, pp. 9–20, 2012, doi: 10.1017/CBO9781107415324.004.

[59] M. G. Galaz Sánchez, "Factores relacionados con la deserción escolar de los alumnos del Departamento de Enfermería de la Universidad de Sonora," Universidad de Sonora, 1999.

[60] E. Castaño, S. Gallón, K. Gómez, Johanna Vásquez, and Vásquez, "Análisis de los factores asociados a la deserción estudiantil en la Educación Superior: un estudio de caso Analysis of the Factors Associated with the Drop-out Rate of Students in Higher Education: a Case Study," *Rev. Educ.*, vol. 345, no. 345, pp. 255–280, 2008, doi: 10.19052/issn.0120-1700.

[61] INEI, "Perú : Características Socio económicas de los Hogares," 2007.

[62] ODES, "Deserción en la Educación Superior," *ODES-boletín 5, julio 2017*, p. 18, 2017, [Online]. Available: file:///C:/Users/SARA/Documents/ADMINISTRACIÒN/INVESTIGACION/odes.pdf.

[63] S. Suthaharan, *Machine Learning Models and Algorithms for Big Data Classification Thinking with Examples for Eff ective Learning*. Springer US, 2016.

[64] M. Kuhn and K. Johnson, *Applied Predictive Modeling*. 2013.

[65] M. Borovcnik, H.-J. Bentz, and R. Kapadia, *A Probabilistic Perspective*. 2011.

[66] F. Nelli, "Machine Learning with scikit-learn," *Python Data Anal.*, pp. 237–264, 2015, doi: 10.1007/978-1-4842-0958-5_8.

[67] G. I. W. (Eds. . Claude Sammut, "Encyclopedia of Machine Learning," p. 1061, 2011, doi: 10.1007/978-0-387-30164-8.

[68] C. J. G. Bellosta, "R para profesionales de los datos: una introducción," p. 139, 2018.

[69] M. Sosa Rojano, "Estudio experimental para la clasificación del grado de una quemadura Capítulo 9: Análisis Jerárquico de Agrupaciones," p. 16.

[70] S. Wagstaf, Kiri.Cardie, Claire.Rogers, Seth.Schroedl, "Constrained K-means with

background knowledge," pp. 577–584, 2001, doi: 10.1109/TPAMI.2002.1017616.

[71] E. Ramírez Aldama, "Mezcla de Gaussianas para clasificación 3.1," *Reconoc. del Habl. Median. extracción Caracter. empleando la Transform. Wavelet*, p. 6, 2015.

[72] J. P. Carvajal González, "Metodología de entrenamiento de modelos de mezclas Gaussianas empleando criterios de gran margen para la detección de patologías en bioseñales.," p. 58, 2019.

[73] P. M. Pardalos, P. Conca, G. Giuffrida, and G. Nicosia, Eds., *Machine Learning, Optimization, and Big Data*. Springer, Cham, 2015.

[74] J. B. Rollins, "Foundational Methodology for Data Science," p. 6, 2015, [Online]. Available: https://www.slideshare.net/JohnBRollinsPhD/foundational-methodology-for-data-science.

[75] F. Foroughi and P. Luksch, "DATA SCIENCE METHODOLOGY FOR CYBERSECURITY PROJECTS," pp. 205–218, 2018.

[76] S. Sharma and K. M. Osei-Bryson, "Organization-ontology based framework for implementing the business understanding phase of data mining projects," *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, pp. 1–10, 2008, doi: 10.1109/HICSS.2008.339.

[77] L. Adler Lomnitz and C. Laura, "Basic, Applied and Technological Research: Computer Science and Applied Mathematics at the National Autonomous University of Mexico," vol. 29, no. 1, pp. 113–134, 2014.

[78] E. Hunt and A.-M. Lavoie, "Qualitative and quantitative research methods.," *Recherche En Soins Infirmiers*, vol. 88. pp. 25–30, 2011, doi: 10.4135/9781446279137.n790.

[79] fernandez montero aturo, "python 3 AL DESCUBIERTO," p. 264, 2013.

[80] R. González Duque, "Python para todos," *Web B.*, p. 160, 2017, [Online]. Available: https://launchpadlibrarian.net/18980633/Python para todos.pdf.

[81] S. Van Der Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: A structure for efficient numerical computation," *Comput. Sci. Eng.*, vol. 13, no. 2, pp. 22–30, 2011, doi: 10.1109/MCSE.2011.37.

[82] W. P.D. and L. C.Y., *Python for Data Analysis*, vol. 344, no. 4. 2012.

[83] G. Lemaître, F. Nogueira, and C. K. Aridas, "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 559–563, 2016, doi:

http://www.jmlr.org/papers/volume18/16-365/16-365.pdf.

[84] A. Cortez, H. Vega, and J. Pariona, “Procesamiento de lenguaje natural,” *Rev. Investig. Sist. e Informática*, vol. 6, no. 2, pp. 45–54, 2009.

[85] C. Piñeda-Baez, A. Pedraza-Ortiz, and I. Moreno, “Efectividad de las estrategias de retencion universitaria,” *Educ. y Educ.*, vol. 14, no. 1, pp. 119–135, 2011.

[86] L. C. Borges, V. M. Marques, and J. Bernardino, “Comparison of data mining techniques and tools for data classification,” *ACM Int. Conf. Proceeding Ser.*, no. July, pp. 113–116, 2013, doi: 10.1145/2494444.2494451.

[87] A. Tharwat, “Classification assessment methods,” *Appl. Comput. Informatics*, 2018, doi: 10.1016/j.aci.2018.08.003.

ANEXOS

Anexos 1. Codigo de Transformacion de los Datos

A continuación, se demuestra las diferentes transformaciones o generación de variables a partir de las variables a partir de herramientas de excel y php, se visualizará la principal parte del código de cada uno con el objetivo de tener una data final.

Tabla 1. Transformación o Generación a partir de la variable “EAP”

Variable	Antes	Transformación/Generación	Resultado
EAP	Psicología, Psicologia-Taropoto	Transformación: Estandarizar el nombre de las carreras	EAP : Psicología
	Arquitectura, Psicologia-Taropoto	Generación: Si el alumno cambio de carrera o no	nro_esc_traslado_ant :1

Luego de identificar la transformación o generación a partir de la variable “EAP” en la Tabla 1, la parte principal de cada una de ellas.

```
$escuela = trim($sheet["eap"]);
//FIA
$array_SISTEMAS = array('EP de Ingenieria de Sistemas', 'EP de Ingenieria de
Sistemas - Filial Juliaca', 'EP de Ingenieria de Sistemas - Filial Tarapoto');
$array_AMBIENTAL = array('EP de Ingenieria Ambiental', 'EP de Ingenieria
Ambiental - Filial Juliaca', 'EP de Ingenieria Ambiental - Filial Tarapoto');

$array_ALIMENTOS = array('EP de Ingenieria de Alimentos', 'EP de Ingenieria de
Industrias Alimentarias', 'EP de Ingenieria de Alimentos - Juliaca', 'EP de
Ingenieria de Industrias Alimentarias - Filial Juliaca');
$array_CIVIL = array('EP de Ingenieria Civil', 'EP de Ingenieria Civil -
Juliaca');
$array_ARQIOTECTURA = array('EP de Arquitectura', 'EP de Arquitectura - Filial
Tarapoto');
$nro_escuela = 0;
if (in_array($escuela, $array_SISTEMAS)) {
    $nro_escuela = 1;
} elseif (in_array($escuela, $array_AMBIENTAL)) {
    $nro_escuela = 2;
} elseif (in_array($escuela, $array_ALIMENTOS)) {
   $nro_escuela = 3;
} elseif (in_array($escuela, $array_CIVIL)) {
   $nro_escuela = 4;
} elseif (in_array($escuela, $array_ARQIOTECTURA)) {
   $nro_escuela = 5;
} else {
   dd("No encontro la carrera");
}
```

Figura 1. Código Central de la Variable Transformada “EAP”
Fuente: Propia

Según la Figura 1 se visualiza una parte del código sobre la transformación de la variable EAP demuestra que se registró las diferentes variaciones de los nombres de una escuela, para ser reconocido como una sola.

```
$nro_esc_traslado_ant=0;
foreach ($datos as $key => $row) {//Los datos del estudiante
                        $nro_esc_traslado_ant=0;
        foreach ($array_nroesc as &$valor_nroesc) {//Todos las carreras unicas
que el alumno estudio
              foreach ($datos as $key2 => $row2) {
   if ($nroescuela == $nroescuela_each3 ) {
                                  }else{
                                      if($nroescuela_each3==$valor_nroesc){
                                          if( $aniofinal_each1 > $aniofinal_each3){
                                              $nro_esc_traslado_ant++;
                                          }
                                          if( $aniofinal_each1 >=
$aniofinal_each3){


$nro_contra_ant=$nro_contra_ant+$row2['nro_contratos'];

$nro_desaprobados_ant=$nro_desaprobados_ant+$row2['nro_desaprobados'];
                                                  $debe_ant=$debe_ant+$row2['debe'];

$credito_ant=$credito_ant+$row2['credito'];

                                          }else{

                                          }

                                      }else{

                                      }
                                  }
              }
       }
}
```

Figura 2. Código central de la Variable Generada "nro_esc_traslado_ant"
Fuente: Propia

Según la Figura 2 se visualiza una parte del código sobre la generación de la variable "nro_esc_traslado_ant" demuestra que se contabilizo el número de escuelas anteriores.

Tabla 2. Transformación o Generación a partir de la variable "SALDO"

Variable	Antes	Transformación/ Generación	Resultado
SALDO	2018(300,00),2009(253,57)	Transformación: Si la deuda es positiva se acumula en "crédito"	Crédito: 553,57
	2018(-21,23),2019(-,4)	Transformación: Si la deuda es positiva se acumula en "debito"	Debito: 21,27

Luego de identificar la transformación o generación a partir de la variable “SALDO” en Tabla 2, la parte principal de cada una de ellas.

```
foreach ($datos as $key => $row) {//Los datos del estudiante
$nro_esc_traslado_ant=0;
$nro_desaprobados_ant=0;
$debe_ant=0;
$credito_ant=0;
$nro_contra_ant=0;
$aniofinal_each1 = $row['anio_final'];
        foreach ($array_nroesc as &$valor_nroesc) {//Todos las carreras unicas
que el alumno estudio
         $nroescuela = $row['nroescuela'];
              foreach ($datos as $key2 => $row2) {
         $nroescuela_each3 = $row2['nroescuela'];
          $aniofinal_each3 = $row2['anio_final'];
   if ($nroescuela == $nroescuela_each3 ) {
                                  }else{
                                      if($nroescuela_each3==$valor_nroesc){
                                          if( $aniofinal_each1 > $aniofinal_each3){
                                              $nro_esc_traslado_ant++;
                                          }
                                          if( $aniofinal_each1 >=
$aniofinal_each3){
$nro_contra_ant=$nro_contra_ant+$row2['nro_contratos'];
$nro_desaprobados_ant=$nro_desaprobados_ant+$row2['nro_desaprobados'];
$debe_ant=$debe_ant+$row2['debe'];
$credito_ant=$credito_ant+$row2['credito'];
                                          }else{
                                          }
                                      }else{
                                      }
                                  }
             }
       }
}
```

Figura 3. Código central de la variable transformada “Credito_ant” y “Debito_ant”
Fuente: Propia

Según la Figura 3 se visualiza una parte del código sobre la transformación del “saldo” para obtener el “debe” y “credito” antes de su carrera actual.

```
$michi="#";
        $saldo1=explode("),", $objectx1->saldo);
        $debe=0;
        $credito=0;
        foreach ($saldo1 as $valorsaldo1){
            $valorsaldo2 = explode("(", $valorsaldo1);
            if (strpos($valorsaldo2[1], ')') !== false) {
                $valorsaldo3 = explode(")", $valorsaldo2[1]);
                $resultado = str_replace(",", ".", $valorsaldo3[0]);
                $saldo =(double)$resultado;
                if($saldo>=0){
                    $debe=$debe+$saldo;
```

```
                }else{
                    $credito=$credito+$saldo;
                }
            }else{
                $resultado_else = str_replace(",", ".", $valorsaldo2[1]);
                $saldo =(double)$resultado_else;
                if($saldo>=0){
                    $debe=$debe+$saldo;
                }else{
                    $credito=$credito+$saldo;
                }
            }
            $financiero=$financiero.$michi.$saldo;
        }
        $credito = str_replace("-", "", $credito);
        $credito=(double)$credito;
```

Figura 4. Código central de la Variable transformada "Credito" y "Debito"
Fuente: Propia

Según la Figura 4 se visualiza una parte del código sobre la transformación del "saldo" para obtener el "debe" y "credito" en su carrera actual.

Por último, se suma para totalizar el "credito_total" y "debe_total"

```
df['credito_total'] = df['credito'] + df['credito_ant']
df['debe_total'] = df['debe'] + df['debe_ant']
```

Figura 5, Código central de la Variable transformada "Credito_total" y "Debito_total"
Fuente: Propia

Según la Figura 5 se visualiza una parte del código sobre la transformación final para sumar él debe antes de su carrera con la actual así también como el crédito.

Tabla 3. Transformación o Generación a partir de la variable "DESA"

Variable	Antes	Transformación/ Generar	Resultado
DESA	DESA: Contabilidad II (2010-1)	Generar: -Se contabiliza de acuerdo al tipo de curso sea general, específico o especialidad	Generales: 0 Específicos: 1 Especialidad: 0
		Generar: -Se contabiliza el número de veces que repite el mismo curso.	Curso_repetitivo_1: 1 Curso_repetitivo_2: 0 Curso_repetitivo_3: 0 Curso_repetitivo_4: 0

Luego de identificar la transformación o generación a partir de la variable “DESA” en la Tabla 3, la parte principal de cada una de ellas.

```
$array_GENERALES_AMBIENTAL = array('Antropologia Biblica','Informatica');
$array_ESPECIFICOS_AMBIENTAL = array('Investigacion en Ingenieria Ambiental
I','Quimica Organica y Bioquimica');
$array_ESPECIALIDAD_AMBIENTAL = array('Electivo II Instrumentos de Gestion
Ambiental','Tratamiento de Aguas');
                            if (in_array($name2, $array_GENERALES_AMBIENTAL)) {
                                $numero_generales++;
                            }elseif (in_array($name2,
$array_ESPECIFICOS_AMBIENTAL)) {
$numero_especificos++;
                            }elseif (in_array($name2,
$array_ESPECIALIDAD_AMBIENTAL)) {
                                $numero_especialidad++;
                            }elseif ($name2==null) {
                                dd("null me aparece1");
                            }else{
                                dd($objectx1->eap.$slash.$name2);
                                $numero_outliers++;
                            }
```

Figura 6. Código Central de la Variable generada “numero_general”, “numero_especialiad” y “numero. especifico”
Fuente: Propia

Según la Figura 6 se visualiza una parte del código sobre la generación de todos los tipos de cursos desaprobados y se contabiliza para la variable numero_generales, números_especificos y números_especialidad.

```
foreach ($array_cursodesap as $val_curso_desap_2nivel){
                    $numero_curso_repetitivo=0;
                    foreach ($array_cursodesap_total as
$val_curso_desap_2nivel_total){
if($val_curso_desap_2nivel==$val_curso_desap_2nivel_total){
                            $numero_curso_repetitivo++;
                        }
                    }
                    if($numero_curso_repetitivo==1){
                        $numero_curso_repetitivo_1++;
                    } elseif($numero_curso_repetitivo==2){
                        $numero_curso_repetitivo_2++;
                    }elseif($numero_curso_repetitivo==3){
                        $numero_curso_repetitivo_3++;
                    }elseif($numero_curso_repetitivo>=4){
                        $numero_curso_repetitivo_4++;
                    }
                    else{
                         dd("aca hay un error".$numero_curso_repetitivo);
                    }
                }
```

Figura 7. Código central de la variable generada curso_repetitivo_1, curso_repetitivo_2, curso_repetitivo_3 y curso_repetititivo_4
Fuente: Propia

Según la Figura 7 se visualiza una parte del código sobre la generación de numero de cursos repetitivos.

Tabla 4. Transformación o Generación a partir de la variable predictora “CONTRATOS”

Variable	Antes	Transformación/ Generación	Resultado
CONTRATOS	Contratos: 2009-1,2010-0,2010-1,2010-2	Transformación: Se contabiliza el número de contratos que tiene el alumno	Nro_contratos: 3

Luego de identificar la transformación o generación a partir de la variable “CONTRATOS” en la Tabla 4, la parte principal de cada una de ellas.

```
foreach ($datos as $key => $row) {//Los datos del estudiante
$nro_contratos_ant=0;

$aniofinal_each1 = $row['anio_final'];
        foreach ($array_nroesc as &$valor_nroesc) {//Todos las carreras unicas
que el alumno estudio
         $nroescuela = $row['nroescuela'];
              foreach ($datos as $key2 => $row2) {
         $nroescuela_each3 = $row2['nroescuela'];
          $aniofinal_each3 = $row2['anio_final'];
   if ($nroescuela == $nroescuela_each3 ) {
                                  }else{
                                      if($nroescuela_each3==$valor_nroesc){
                                          if( $aniofinal_each1 > $aniofinal_each3){
                                              $nro_contratos_ant++;
                                          }
                                          if( $aniofinal_each1 >=
$aniofinal_each3){

$nro_desaprobados_ant=$nro_desaprobados_ant+$row2['nro_desaprobados'];

                                          }else{

                                          }

                                      }else{

                                      }
                                  }
              }
       }
}
```

Figura 8. Código Central de la Variable generada “nro_contratos_ant”
Fuente: Propia

Según la Figura 8 se visualiza una parte del código sobre la generación para obtener el número de contratos en total antes de cambiar a otra carrera dentro de la Institución UpeU.

```
public function f_countcontratos($contratos){
      //  dd($contratos);
        $count=0;
        $descomprimir = explode(",", $contratos);
        foreach ($descomprimir as &$valor3){
            $porcion_ciclonota = explode("-", $valor3);
            $anio2 = $porcion_ciclonota[0];
            $ciclo2 = $porcion_ciclonota[1];
            if(trim($ciclo2)=="M" ||trim($ciclo2)=="0"){
            }else{
                $count++;
            }
        }
        return $count;

    }
```

Figura 9. Código central de la variable generada "CONT_CONTRATOS".
Fuente: Propia

Según la Figura 9 se visualiza una parte del código sobre la generación para obtener número de contratos que ha tenido en su actual carrera.

Por último, se suma para totalizar el "nro_contra_ant" y "CONT_CONTRATOS"

```
df['CONT_CONTRATOS_TOTAL'] = df['nro_contra_ant'] + df['CONT_CONTRATOS']
```

Figura 10. Código central de la Variable transformada "nro_contra_ant" y "cont_contratos"
Fuente: Propia

Según la Figura 10 se visualiza una parte del código sobre la transformación final para sumar él debe antes de su carrera con la actual así también como el crédito.

Tabla 5. Transformación o Generación a partir de la variable predictora "NOTAS"

Variable	Antes	Transformación/ Generación	Resultado
NOTAS	Semestre 2009-0 (14,20) , Semestre 2009-1 (14,91), Semestre 2009-2 (11,48)	Generación: Si en el semestre anterior obtuvo una menor nota entonces se acumula la diferencia	Incremento_nota:5.42
		Generación: Si en el semestre anterior obtuvo una mayor nota entonces se acumula la diferencia	Disminuye_nota:4.42
		Generación: El promedio de las notas de cada semestre	Promedio: 13.20
		Generación: La nota final del último semestre	Nota_Final: 11.48

Luego de identificar la transformación o generación a partir de la variable “NOTAS” en la Tabla 5, la parte principal de cada una de ellas.

```
array_multisort(array_column($datos, "codigo"), SORT_ASC, $datos);
                $CONT_DATOS = 0;
                $guion = "/";
                $incremento_nota=0;
                $desincremento_nota=0;
                $cont_array=0;
                $notav1=0;
                $notav2=0;
                foreach ($datos as $key => $row) {//Se calcula si hubo un
incremento o desincremento a comparación del ciclo anterior
                    $cont_array++;
                    $CONT_DATOS++;
                    $codigo[$key] = $row['codigo'];
                    $nota[$key] = $row['nota'];
                    $resultado_nota = str_replace(",", ".", $nota[$key]);
                    $resultado_nota2 =(double)$resultado_nota;
                    if ($CONT_DATOS == 1) {
                        $notas_juntadas = $resultado_nota2;
                    } else {
                        $notas_juntadas = $notas_juntadas . $guion
.$resultado_nota2;
                    }
                    if($cont_array==1){
                        $notav1=$resultado_nota2;
                    }else{
                        $notav2=$resultado_nota2;
                        $diferenciador=$notav2-$notav1;
```

Figura 11. Código Central de la Variable generada “diferenciador”
Fuente: Propia

Según la Figura 11 se visualiza una parte del código sobre la generación del diferenciador entre la nota actual del ciclo y la nota anterior del ciclo.

```
if($diferenciador>=0){
                            $incremento_nota=$incremento_nota+$diferenciador;
                        }else{

$desincremento_nota=$desincremento_nota+$diferenciador;
                        }
                    }
                }
```

Figura 12. Código Central de la Variable generada “incrementada_nota” y “desincremento_nota”
Fuente: Propia

Según la Figura 12 se visualiza una parte del código sobre la generación de “incremento_nota” y “desincremento_nota”.

```
$porcion_juntada = explode("/", $notas_juntadas);
        $count_porcion_juntada=count($porcion_juntada);
        $promedio=0.00;
        if($count_porcion_juntada==1){
            foreach($porcion_juntada as $valor_nota){
                if($valor_nota=="-"){
                }else{
                    $promedio=$valor_nota;
                }
            }
        }else{
            foreach($porcion_juntada as $valor_nota){
                if($valor_nota=="-"){
                }else{
                    $promedio=$promedio+$valor_nota;
                }
            }
            $promedio=(double)$promedio/$count_porcion_juntada;
        }
```

Figura 13. Código central de la variable generada "promedio".
Fuente: Propia

Según la Figura 13 se visualiza una parte del código sobre la generación de la variable "promedio" del estudiante hasta un ciclo.

```
public function f_notafinal($notas,$contrato_final){
 $contratos = explode(",", $contrato);
 $descomprimir = explode("),", $notas);
   foreach ($descomprimir as &$valor3){
               $porciones3 = explode(" ", $valor3);
               $porciones4 = explode("(", $porciones3[2]);
               if (strpos($porciones4[1], ')') !== false) {
                   $porciones5 = explode(")", $porciones4[1]);
                   $nota4 = $porciones5[0];
               } else {
                   $nota4 = $porciones4[1];
               }
               $nota4 = str_replace("(,", "(0.", $nota4);
               $nota4 = str_replace(",", ".", $nota4);
               $porcion_ciclonota = explode("-", $porciones3[1]);
               $anio2 = (int)trim($porcion_ciclonota[0]);
               $ciclo2 = trim($porcion_ciclonota[1]);
               $ciclo3 = $ciclo2;
               if(trim($ciclo2)=="M"){
                   $ciclo2=2;
               }else{
$ciclo2=(int)$ciclo2;
                   $anio_and_ciclo=$anio2.$guion.$ciclo2;
                   if (in_array($anio_and_ciclo, $contratos)) {
                       $cont++;
                       if($cont==1){
                           $nota=$nota4;
                       }else{
                           dd("error no puede aver mas de un contrato inicial");
                   }
                   }
               }
           }
 }
```

Figura 14. Código central de la variable generada "nota_final".
Fuente: Propia

Según la Figura 14 se visualiza una parte del código sobre la generación de la variable "nota_final" del estudiante hasta un ciclo.

Tabla 6. Transformación o Generación de la variable "RETIRADO"

Variable	Antes	Transformación/Generación	Resultado
RETIRADO NOTAS	Retirado: 2013-1 Contratos: 2010-1,2010-2 Notas: Semestre 2010-1 (10,17), Semestre 2010-2 (4,75)	Generar: -Si la nota del semestre es menor a 5 es considerado un ciclo retirado. -Se compara los ciclos retirados con el ciclo final en la Fase 3	RETIRADO: 1

Luego de identificar la transformación o generación a partir de la variable "RETIRADO" y "NOTAS" en la Tabla 6, la parte principal de cada una de ellas.

```
$retirado=0;
 $array_retirado = explode(",", $valor->retirado_general);
   if (in_array($ciclofinal, $array_retirado)) {
                              $retirado=1;
                          }
```

Figura 15. Código central de la variable generada "retirado"
Fuente: Propia

Según la Figura 15 se visualiza una parte del código sobre la generación de la variable "retirado" donde se compara si el ciclo final se encuentra en los ciclos retirados.

Tabla 7. Transformación o Generación a partir de la variable "DESERTOR"

Variable	Antes	Transformación/Generación	Resultado
CONTRATOS EGRESADO	Contratos: 2010-1,2010-2,2012-1 Egreso: -	Generación: -Si no existe el ciclo después se considera un ciclo Desertor. -No se considera el último ciclo de un egresado -Se compara los ciclos desertores con el ciclo final en la Fase 3	DESERTOR: 1

Luego de identificar la transformación o generación a partir de la variable "CONTRATOS" y "EGRESADO" en la Tabla 7, la parte principal de cada una de ellas.

```
$desertor=0;
 $array_trunca = explode(",", $valor->trunca);
   if (in_array($ciclofinal, $array_trunca)) {
                              $desertor=1;
                         }
```

Figura 16. Código central de la variable generada "desertor"
Fuente: Propia

Según la Figura 16 se visualiza una parte del código sobre la generación de la variable "desertor" donde se compara si el ciclo final se encuentra en los ciclos retirados.

Tabla 11. Transformación o Generación a partir de la variable "Edad"

Variable	Antes	Transformación/Generación	Resultado
CONTRATOS FECHA_NAC	Contratos: 2010-1,2010-2,2012-1 Fecha_nac: 24/08/1991 00:00:00	Generación: -De los contratos se encuentra el ciclo final -Se utiliza una fórmula de Excel para obtener	Edad: 20

Luego de identificar la transformación o generación a partir de la variable "CONTRATOS" y "FECHA_NAC" en la Tabla 11, la parte principal de cada una de ellas.

```
public function ciclo_final($contratos){
    $guion="-";
    $porciones = explode(",",  $contratos);
    $datos=[];
    foreach ($porciones as &$valor){
        $porciones2 = explode("-", $valor);
        if(count($porciones2)==1){
            dd("porcion ciclo 1 ciclo inicial");

        }
        $anio=intval($porciones2[0]);

        $ciclo=intval($porciones2[1]);

        $datos[] = array('ciclo_final' => $anio.$ciclo,);
    }
    array_multisort(array_column($datos, "ciclo_final"), SORT_ASC, $datos);
    $variable = (int)$datos[0]["ciclo_final"];
    return $variable;
}
```

Figura 17. Código Central de la Variable generada "ciclo_final"
Fuente: Propia

Según la Figura 17 se visualiza una parte del código sobre la generación de la variable "ciclo_final"

```
=SIFECHA(FECHA(IZQUIERDA(F2,4), EXTRAE(F2,6,2), EXTRAE(F2,9,2)),
CONCATENAR(SI(DERECHA(Q2,1) ="2","05/12/","05/07/"), IZQUIERDA(Q2,4)),"y")
```

Figura 18. Código Central de la Variable generada "Edad"
Fuente: Propia

Posteriormente de haber obtenido el ciclo final se realiza una fórmula del ciclo.final y fecha de nacimiento. (Figura 18).

Para más explicaciones ver los pasos en la formula a continuación:

PASO 1

Se obtiene la fecha de nacimiento del alumno

=FECHA(IZQUIERDA(F2,4), EXTRAE(F2,6,2), EXTRAE(F2,9,2))

PASO 2

Se obtiene la fecha del año final del caso

=CONCATENAR(SI(DERECHA(Q2,1) ="2","05/12/","05/07/"), IZQUIERDA(Q2,4))

PASO 3

Se resta la fecha del año final con la fecha de nacimiento en años

=SIFECHA (G2, R2,"y")

Anexos 2. Codigo utilizado para encontrar el Modelo Predictivo Eficaz

Importar dependencias necesarias

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import model_evaluation_utils as meu
from sklearn.model_selection import train_test_split
from collections import Counter
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv('data/pregrado_final.csv', encoding = "ISO-8859-1")

df=df[[
  'Arquitectura','Alimentos','Ambiental','Civil','Sistemas',
   'NOTA_FINAL',
    'RETIRADO',
  'count_array_retiro_desof','count_array_trunc',
    'nro_contra_ant','nro_esc_traslado_ant','debe_ant','credito_ant','nro_desaprobados_ant',
  'debe','credito',
  'incremento_nota','desincremento_nota'
   ,'curso_repetitivo_1','curso_repetitivo_2','curso_repetitivo_3','curso_repetitivo_4'
   ,'CONT_JUNT'
    ,'numero_generales','numero_especificos','numero_especialidad'
   ,'Edad','NACIONALIDAD','GENERO','CONT_CONTRATOS'
   ,'RELIGION','PROMEDIO','DESERTOR','SEDE_FINAL'
   ]]
```

CONVERSION

```
df = df.dropna()
#Cambiar el modelo de object
df = df.astype({"DESERTOR":'object'})
#Unir los diferentes variables
df['CONT_CONTRATOS_TOTAL'] = df['nro_contra_ant'] + df['CONT_CONTRATOS']
df['debe_total'] = df['debe'] + df['debe_ant']
df['credito_total'] = df['credito'] + df['credito_ant']
df['NRO_DESAPROBADOS'] = df['numero_generales'] + df['numero_especificos'] + df['numero_especial
idad']
df['NRO_DESAPROBADOS_TOTAL'] = df['NRO_DESAPROBADOS'] + df['nro_desaprobados_ant']
df['curso_repetitivo'] =  df['curso_repetitivo_2'] + df['curso_repetitivo_3']  + df['curso_repetitivo_4']

#Transformar nacionalidad
```

```
df['NACIONALIDAD'] = df['NACIONALIDAD'].astype('str')
df['NACIONALIDAD'] = df['NACIONALIDAD'].replace(['Ecuatoriano(a)', 'Venezolano(a)','Colombiano (a
)','Boliviano (a)','Paname?o(a)'
                                ,'Argentino (a)','Belice?o(a)','Brasile?o (a)','Guatemalteco(a)','Espa?ol(a)','Pap&
uacute'
                                ,'Hondure?o(a)'
                                ], '0')
df['NACIONALIDAD'] = df['NACIONALIDAD'].replace(['Peruano(a)'], '1')
df['NACIONALIDAD'] = df['NACIONALIDAD'].astype('int64')

#Transformar genero

df['GENERO'] = df['GENERO'].astype('str')
df['GENERO'] = df['GENERO'].replace(['Masculino'], '1')
df['GENERO'] = df['GENERO'].replace(['Femenino'], '0')
df['GENERO'] = df['GENERO'].astype('int64')

#Transformar religion

df['RELIGION'] = df['RELIGION'].astype('str')
df['RELIGION'] = df['RELIGION'].replace(['Otro', 'catolico','EVANGELICO','Testigo de Jehova','Israelita d
el Nuevo Pacto'
                                ],'0')
df['RELIGION'] = df['RELIGION'].replace(['adventista'],'1')
df['RELIGION'] = df['RELIGION'].astype('int')


df['numero_generales'] = df['numero_generales'].astype('int')
df['numero_especificos'] = df['numero_especificos'].astype('int')
df['numero_especialidad'] = df['numero_especialidad'].astype('int')

df['nro_esc_traslado_ant'] = df['nro_esc_traslado_ant'].astype('int')
df['nro_desaprobados_ant'] = df['nro_desaprobados_ant'].astype('int')
df['count_array_retiro_desof'] = df['count_array_retiro_desof'].astype('int')
df['count_array_trunc'] = df['count_array_trunc'].astype('int')

#Cambiar el modelo de object
df = df.astype({"DESERTOR":'object'})
#Unir los diferentes variables
df['CONT_CONTRATOS_TOTAL'] = df['nro_contra_ant'] + df['CONT_CONTRATOS']
df['debe_total'] = df['debe'] + df['debe_ant']
df['credito_total'] = df['credito'] + df['credito_ant']
df['NRO_DESAPROBADOS'] = df['numero_generales'] + df['numero_especificos'] + df['numero_especial
idad']
df['NRO_DESAPROBADOS_TOTAL'] = df['NRO_DESAPROBADOS'] + df['nro_desaprobados_ant']
```

```
df['curso_repetitivo'] =  df['curso_repetitivo_2'] + df['curso_repetitivo_3']  + df['curso_repetitivo_4']

#Transformar nacionalidad

df['NACIONALIDAD'] = df['NACIONALIDAD'].astype('str')
df['NACIONALIDAD'] = df['NACIONALIDAD'].replace(['Ecuatoriano(a)', 'Venezolano(a)','Colombiano (a
)','Boliviano (a)','Paname?o(a)'
                                ,'Argentino (a)','Belice?o(a)','Brasile?o (a)','Guatemalteco(a)','Espa?ol(a)','Pap&
uacute'
                                ,'Hondure?o(a)'
                                ], '0')
df['NACIONALIDAD'] = df['NACIONALIDAD'].replace(['Peruano(a)'], '1')
df['NACIONALIDAD'] = df['NACIONALIDAD'].astype('int64')

#Transformar genero

df['GENERO'] = df['GENERO'].astype('str')
df['GENERO'] = df['GENERO'].replace(['Masculino'], '1')
df['GENERO'] = df['GENERO'].replace(['Femenino'], '0')
df['GENERO'] = df['GENERO'].astype('int64')

#Transformar religion

df['RELIGION'] = df['RELIGION'].astype('str')
df['RELIGION'] = df['RELIGION'].replace(['Otro', 'catolico','EVANGELICO','Testigo de Jehova','Israelita d
el Nuevo Pacto'
                                ],'0')
df['RELIGION'] = df['RELIGION'].replace(['adventista'],'1')
df['RELIGION'] = df['RELIGION'].astype('int')


df['numero_generales'] = df['numero_generales'].astype('int')
df['numero_especificos'] = df['numero_especificos'].astype('int')
df['numero_especialidad'] = df['numero_especialidad'].astype('int')

df['nro_esc_traslado_ant'] = df['nro_esc_traslado_ant'].astype('int')
df['nro_desaprobados_ant'] = df['nro_desaprobados_ant'].astype('int')
df['count_array_retiro_desof'] = df['count_array_retiro_desof'].astype('int')
df['count_array_trunc'] = df['count_array_trunc'].astype('int')
```

```
#Definir los registros que su ultimo contrato sea en Lima y sea de la
carrera de Alimentos.
df=df[df.SEDE_FINAL=='LIMA']
df=df[df.Alimentos==1]
#   'Arquitectura','Alimentos','Ambiental','Civil','Sistemas',
df=df[[
   'NOTA_FINAL',
    'RETIRADO',
     'count_array_retiro_desof',
  'count_array_trunc',
  'debe','credito',
  'incremento_nota','desincremento_nota'
    ,'curso_repetitivo_1'
  ,'curso_repetitivo_2'
  ,'curso_repetitivo_3'
  ,'curso_repetitivo_4'

    ,'debe_ant','credito_ant'
    ,'CONT_JUNT'
   ,'numero_generales','numero_especificos'
    ,'numero_especialidad'
   ,'Edad','NACIONALIDAD','GENERO'
  ,'CONT_CONTRATOS'
  ,'nro_contra_ant'

  ,'nro_esc_traslado_ant'
    ,'nro_desaprobados_ant'
  ,'RELIGION','PROMEDIO'
 #  ,'CONT_CONTRATOS'
  ,'CONT_CONTRATOS_TOTAL'
  ,'debe_total'
  ,'credito_total'
   ,'NRO_DESAPROBADOS'
  ,'NRO_DESAPROBADOS_TOTAL'
   ,'curso_repetitivo'
  ,'DESERTOR'
   ]]

df.shape#Numero de Filas y de columnas
Out[27]:(1434, 34)
df['count_array_retiro_desof'] = np.where(df['count_array_retiro_desof'] > 0, 1, 0)
df['count_array_trunc'] = np.where(df['count_array_trunc'] > 0, 1, 0)
df['nro_esc_traslado_ant'] = np.where(df['nro_esc_traslado_ant'] > 0, 1, 0)
df['curso_repetitivo_4'] = np.where(df['curso_repetitivo_4'] > 0, 1, 0)
```

```
#Obtener las desviaciones estándar de cada variable
df.std()
```

Out[29]:

```
NOTA_FINAL                     2.318123
RETIRADO                       0.108271
count_array_retiro_desof       0.000000
count_array_trunc              0.234914
debe                        3913.792340
credito                      819.238605
incremento_nota                1.874753
desincremento_nota             1.897533
curso_repetitivo_1             1.997218
curso_repetitivo_2             0.631759
curso_repetitivo_3             0.083350
curso_repetitivo_4             0.000000
debe_ant                     201.274958
credito_ant                    0.000000
CONT_JUNT                      0.164727
numero_generales               1.523330
numero_especificos             1.228797
numero_especialidad            0.791147
Edad                           2.833977
NACIONALIDAD                   0.164727
GENERO                         0.460459
CONT_CONTRATOS                 2.806359
nro_contra_ant                 0.351413
nro_esc_traslado_ant           0.140812
nro_desaprobados_ant           1.007082
RELIGION                       0.464201
PROMEDIO                       1.512939
CONT_CONTRATOS_TOTAL           2.832165
debe_total                  3927.890249
credito_total                819.238605
NRO_DESAPROBADOS               2.670559
NRO_DESAPROBADOS_TOTAL         2.966168
curso_repetitivo               0.645079
DESERTOR                       0.281157
dtype: float64
```

```
#Definir las variables categoricas
cat_attr_list=[
  'RETIRADO'
  ,'curso_repetitivo_3'
  ,'count_array_retiro_desof'#Las veces que se retiro
```

```
  ,'count_array_trunc'#Las veces que se retiro
  ,'CONT_JUNT'#Las veces que se cambio de sede
  ,'curso_repetitivo_4'
  ,'NACIONALIDAD','GENERO'
  ,'nro_esc_traslado_ant'#Las veces que se retiro
  ,'nro_contra_ant'#
   ,'nro_desaprobados_ant'#
  ,'RELIGION'
  #,'DESERTOR'
    ]
#Definir las variables numericas
numeric_feature_cols=[
   'NOTA_FINAL','incremento_nota','desincremento_nota','PROMEDIO'
  ,'curso_repetitivo_1','curso_repetitivo_2' ,'curso_repetitivo'
  ,'numero_generales','numero_especificos','numero_especialidad','NRO_DESAPROBADOS_TOTAL'
  ,'CONT_CONTRATOS_TOTAL'
  ,'debe_total','credito_total'
  ,'Edad'
    ]
#Definir las variables finales
df = df[[
  'RETIRADO'
  ,'curso_repetitivo_3'
  ,'count_array_retiro_desof'#Las veces que se retiro
  ,'count_array_trunc'#Las veces que se retiro
  ,'CONT_JUNT'#Las veces que se cambio de sede
  ,'curso_repetitivo_4'
  ,'NACIONALIDAD','GENERO'
  ,'nro_esc_traslado_ant'#Las veces que se retiro
  ,'nro_contra_ant'#
   ,'nro_desaprobados_ant'#
  ,'RELIGION'
  ,'NOTA_FINAL','incremento_nota','desincremento_nota','PROMEDIO'
  ,'curso_repetitivo_1','curso_repetitivo_2' ,'curso_repetitivo'
  ,'numero_generales','numero_especificos','numero_especialidad','NRO_DESAPROBADOS_TOTAL'
  ,'CONT_CONTRATOS_TOTAL'
  ,'debe_total','credito_total'
  ,'Edad'
  ,'DESERTOR'
  ]]
```

Cargar y combinar conjuntos de datos

```
df['DESERTOR'] = df['DESERTOR'].apply(lambda value: 'NO DESERTOR' if value == 0 else 'DESERTOR
')

df = df.sample(frac=1, random_state=42).reset_index(drop=True)
```

Preparar conjuntos de datos de entrenamiento y testeo

```
df_features = df.iloc[:,:-1]
df_class_labels = np.array(df['DESERTOR'])
wqp_label_names = ['DESERTOR', 'NO DESERTOR']
X_train, X_test, y_train, y_test = train_test_split(df_features, df_class_labels,
                                                    test_size=0.3, random_state=42)
print(Counter(y_train), Counter(y_test))
```

```
Counter({'NO DESERTOR': 916, 'DESERTOR': 87}) Counter({'NO DESERTOR': 394
, 'DESERTOR': 37})
```

Escalado de Caracteristicas

```
# Definir la escala
wqp_ss = StandardScaler().fit(X_train[numeric_feature_cols])

# Escalar la data de entrenamiento
X_train[numeric_feature_cols] = wqp_ss.transform(X_train[numeric_feature_cols])

# Escalar la data de testeo
X_test[numeric_feature_cols] = wqp_ss.transform(X_test[numeric_feature_cols])

train_SX = X_train
test_SX = X_test
```

Obtener los array de estandarizacion

```
means = wqp_ss.mean_
vars = wqp_ss.var_
means
```

Out[20]:

```
array([1.50909970e+01, 1.70896311e+00, 1.65425723e+00, 1.52097171e+01,
       1.22831505e+00, 1.01694915e-01, 1.07676969e-01, 4.45663011e-01,
       7.41774676e-01, 2.62213360e-01, 1.57328016e+00, 4.54536391e+00,
       3.03268359e+03, 2.61627368e+02, 2.00837488e+01])
```

```
vars
```

Out[21]:

```
array([4.32196630e+00, 3.61044299e+00, 3.56511159e+00, 2.16123420e+00,
       3.82524013e+00, 4.34324146e-01, 4.53011852e-01, 2.34276035e+00,
       1.54348917e+00, 6.50087623e-01, 8.82688326e+00, 7.99071380e+00,
```

```
       1.76120082e+07, 5.96927770e+05, 8.14054347e+00])
```

```
from past.builtins import xrange
def obtener_array(array_x):
  arrayfinal=[]
  for i in xrange(len(array_x)):
    #print(array_x[i])
    arrayfinal.append(array_x[i])
  return (arrayfinal)
array_means=obtener_array(means)
array_means
#np.array(array_means) #Convertir
```

Out[23]:

```
[15.090997008973082,
 1.7089631106679959,
 1.6542572283150552,
 15.209717087457252,
 1.2283150548354935,
 0.1016949152542373,
 0.10767696909272183,
 0.4456630109670987,
 0.7417746759720838,
 0.2622133599202393,
 1.5732801595214356,
 4.545363908275174,
 3032.683589232303,
 261.62736789631106,
 20.083748753738785]
```

```
array_vars=obtener_array(vars)
array_vars
```

Out[24]:

```
[4.321966304078791,
 3.610442992657123,
 3.565111586874471,
 2.16123419803184,
 3.825240132046533,
 0.4343241462054515,
 0.4530118517826381,
 2.342760353038591,
 1.5434891735561014,
 0.6500876234705653,
 8.826883258499677,
 7.990713800771166,
 17612008.223453917,
 596927.7703298617,
```

```
 8.140543474263152]
```

Generar Modelo Predictivo

Obtener los mejores modelos con sus mejores parametros

Definir el array con los algoritmos

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from xgboost import XGBClassifier
list_grid =[

["KNN",KNeighborsClassifier(),

[{'n_neighbors': range(1,10), 'weights': ['uniform'],
                'metric': ['euclidean']},
               {'n_neighbors': range(1,10), 'weights': ['uniform'],
                'metric': ['euclidean']},
               {'n_neighbors': range(1,10), 'weights': ['distance'],
                'metric': ['manhattan']},
               {'n_neighbors': range(1,10), 'weights': ['distance'],
                'metric': ['manhattan']}
               ]
    ]
  ,

["Naive Bayes",GaussianNB(),

[{"var_smoothing":[1e-09,1e-8, 1e-7, 1e-6, 1e-5, 1e-4] }]
    ]
,

["Decision Tree",DecisionTreeClassifier(),
[{"max_depth": range(1,4), "random_state":[7],'min_samples_split' : range(10,500,20)}]
    ]
,
["Random Forest",RandomForestClassifier(random_state=42),
[{ 'max_depth': [1,2,3],
          'n_estimators': [100, 200, 300, 500],
          'max_features': ['auto', None, 'log2']
         }]
    ]
,
["XGBOOST",XGBClassifier(learning_rate=0.02, n_estimators=600, objective='binary:logistic',
              nthread=1),
```

```
[{ 'min_child_weight': [1, 5, 10],
      'gamma': [0.5, 1, 1.5, 2, 5],
      'subsample': [0.6, 0.8, 1.0],
      'colsample_bytree': [0.6, 0.8, 1.0],
      'max_depth': [1,2,3]
         }]
      ]
]

#0 nombre
#1 algoritmo
#2 gridsearchcv

from sklearn.model_selection import GridSearchCV
for x in range(0, len(list_grid)):
      print(x)

      param_grid = list_grid[x][2]
      wqp_clf = GridSearchCV(list_grid[x][1], param_grid, cv=5,
                        scoring='balanced_accuracy')
      wqp_clf.fit(train_SX, y_train)
      print(wqp_clf.best_params_)
      list_grid[x].append(wqp_clf.best_estimator_)

      #list_grid[x][2]=wqp_clf.best_estimator_
0
{'metric': 'euclidean', 'n_neighbors': 2, 'weights': 'uniform'}
1
{'var_smoothing': 1e-05}
2
{'max_depth': 2, 'min_samples_split': 10, 'random_state': 7}
3
{'max_depth': 2, 'max_features': None, 'n_estimators': 500}
4
{'colsample_bytree': 1.0, 'gamma': 1, 'max_depth': 2, 'min_child_weight':
1, 'subsample': 0.8}

for x in range(0, len(list_grid)):
   print("")
   print(list_grid[x][0])
   wqp_rf_predictions = list_grid[x][3].predict(test_SX)
   balanced_score,tpr=meu.display_model_performance_metrics(true_labels=y_test, predicted_labels=
wqp_rf_predictions,
                           classes=wqp_label_names)
   list_grid[x].append(balanced_score)
```

```
  list_grid[x].append(tpr)
  print("---------------------------------")
  print("---------------------------------")

KNN

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           17           20
NO DESERTOR        33          361
Model Performance metrics:
------------------------------
TPR: 0.4595
Balanced Accuracy: 0.6879
Precision: 0.8954
Recall: 0.877
F1 Score: 0.8852

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.34      0.46      0.39        37
 NO DESERTOR       0.95      0.92      0.93       394

    accuracy                           0.88       431
   macro avg       0.64      0.69      0.66       431
weighted avg       0.90      0.88      0.89       431

---------------------------------
---------------------------------
Naive Bayes
Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           16           21
NO DESERTOR        26          368
Model Performance metrics:
------------------------------
TPR: 0.4324
Balanced Accuracy: 0.6832
Precision: 0.8975
Recall: 0.891
F1 Score: 0.8941
```

```
Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.38      0.43      0.41        37
 NO DESERTOR       0.95      0.93      0.94       394

    accuracy                           0.89       431
   macro avg       0.66      0.68      0.67       431
weighted avg       0.90      0.89      0.89       431

----------------------------------
----------------------------------

Decision Tree

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           11           26
NO DESERTOR         7          387
Model Performance metrics:
------------------------------
TPR: 0.2973
Balanced Accuracy: 0.6398
Precision: 0.9091
Recall: 0.9234
F1 Score: 0.9111

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.61      0.30      0.40        37
 NO DESERTOR       0.94      0.98      0.96       394

    accuracy                           0.92       431
   macro avg       0.77      0.64      0.68       431
weighted avg       0.91      0.92      0.91       431

----------------------------------
----------------------------------
```

```
Random Forest
Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           11           26
NO DESERTOR         7          387
Model Performance metrics:
------------------------------
TPR: 0.2973
Balanced Accuracy: 0.6398
Precision: 0.9091
Recall: 0.9234
F1 Score: 0.9111

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.61      0.30      0.40        37
 NO DESERTOR       0.94      0.98      0.96       394

    accuracy                           0.92       431
   macro avg       0.77      0.64      0.68       431
weighted avg       0.91      0.92      0.91       431

-----------------------------------
-----------------------------------

XGBOOST

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           13           24
NO DESERTOR         7          387
Model Performance metrics:
------------------------------
TPR: 0.3514
Balanced Accuracy: 0.6668
Precision: 0.9166
Recall: 0.9281
F1 Score: 0.9181

Model Classification report:
------------------------------
```

```
              precision    recall  f1-score   support

    DESERTOR       0.65      0.35      0.46        37
 NO DESERTOR       0.94      0.98      0.96       394

    accuracy                           0.93       431
   macro avg       0.80      0.67      0.71       431
weighted avg       0.92      0.93      0.92       431

-----------------------------------
-----------------------------------
```

df_hiperparametros = pd.DataFrame(list_grid, columns =['name','algoritmo','parametros','modelo_hiperparametros','Balanced_Accuray','TPR'])
df_hiperparametros

Out[23]:

	name	algoritmo	parametros	modelo_hiperparametros	Balanced_Accuray	TPR
0	KNN	KNeighborsClassifier(algorithm='auto', leaf_si...	[{'n_neighbors': (1, 2, 3, 4, 5, 6, 7, 8, 9), ...	KNeighborsClassifier(algorithm='auto', leaf_si...	0.687852	0.459459
1	Naive Bayes	GaussianNB(priors=None, var_smoothing=1e-09)	[{'var_smoothing': [1e-09, 1e-08, 1e-07, 1e-06...	GaussianNB(priors=None, var_smoothing=1e-05)	0.683221	0.432432
2	Decision Tree	DecisionTreeClassifier(ccp_alpha=0.0, class_we...	[{'max_depth': (1, 2, 3), 'random_state': [7],...	DecisionTreeClassifier(ccp_alpha=0.0, class_we...	0.639765	0.297297
3	Random Forest	RandomForestClassifier(bootstrap=True, ccp_alp...	[{'max_depth': [1, 2, 3], 'n_estimators': [100...	(DecisionTreeClassifier(ccp_alpha=0.0, class_w...	0.639765	0.297297
4	XGBOOST	XGBClassifier(base_score=None, booster=None, c...	[{'min_child_weight': [1, 5, 10], 'gamma': [0....	XGBClassifier(base_score=0.5, booster='gbtree'...	0.666792	0.351351

```
df_hiperparametros[["modelo_hiperparametros"]].values
```

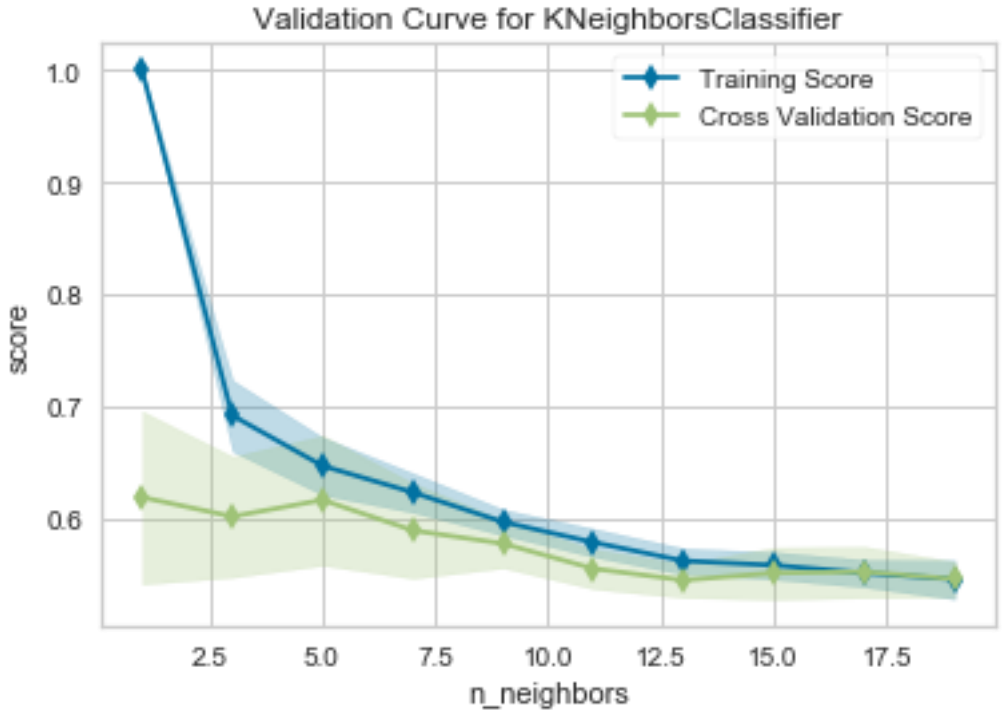
Out[24]:

```
array([[KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='eucl
idean',
                     metric_params=None, n_jobs=None, n_neighbors=2, p=2,
                     weights='uniform')],
       [GaussianNB(priors=None, var_smoothing=1e-05)],
       [DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterio
n='gini',
                         max_depth=2, max_features=None, max_leaf_nodes=Non
e,
                         min_impurity_decrease=0.0, min_impurity_split=None
,
                         min_samples_leaf=1, min_samples_split=10,
                         min_weight_fraction_leaf=0.0, presort='deprecated'
,
                         random_state=7, splitter='best')],
       [RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weigh
t=None,
                         criterion='gini', max_depth=2, max_features=None,
                         max_leaf_nodes=None, max_samples=None,
                         min_impurity_decrease=0.0, min_impurity_split=None
,
                         min_samples_leaf=1, min_samples_split=2,
                         min_weight_fraction_leaf=0.0, n_estimators=500,
                         n_jobs=None, oob_score=False, random_state=42, ver
bose=0,
                         warm_start=False)],
       [XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel
=1,
              colsample_bynode=1, colsample_bytree=1.0, gamma=1, gpu_id=-
1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.02, max_delta_step=0, max_depth=2,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=600, n_jobs=1, nthread=1, num_parallel_tree=1,
              objective='binary:logistic', random_state=0, reg_alpha=0,
              reg_lambda=1, scale_pos_weight=1, subsample=0.8,
              tree_method='exact', validate_parameters=1, verbosity=None)
]],
      dtype=object)
```

Generar las curvas de validación

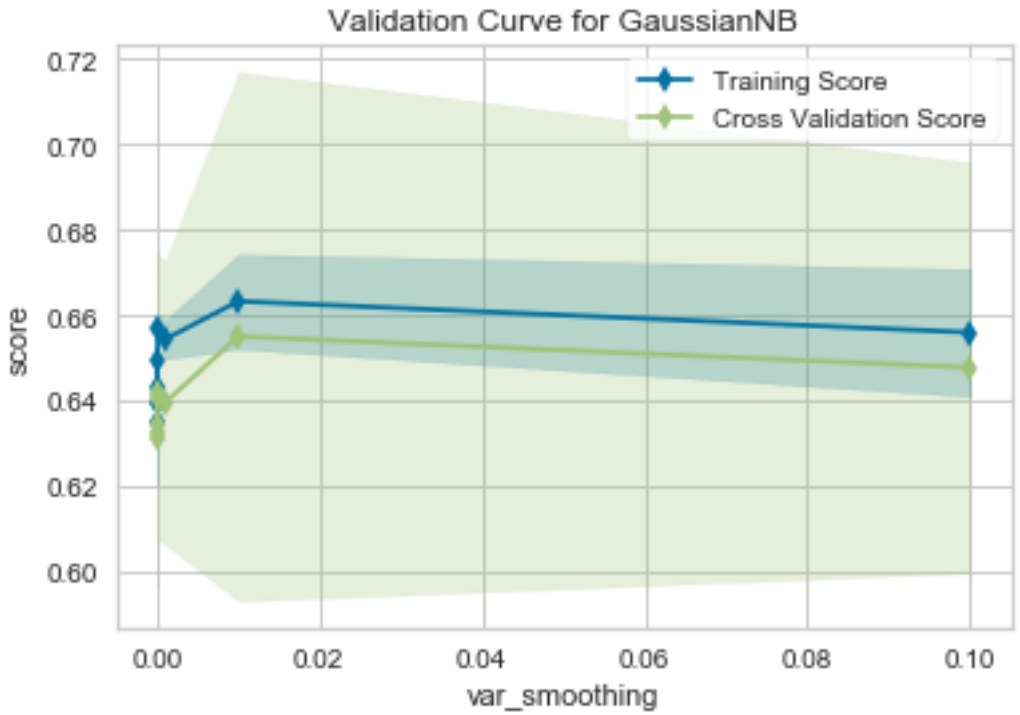
KNN

```
meu.validation_curve(train_SX, y_train,
                     KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
                      metric_params=None, n_jobs=None, p=2,
                      weights='uniform')

                     ,"n_neighbors",np.arange(1,20,2))
```



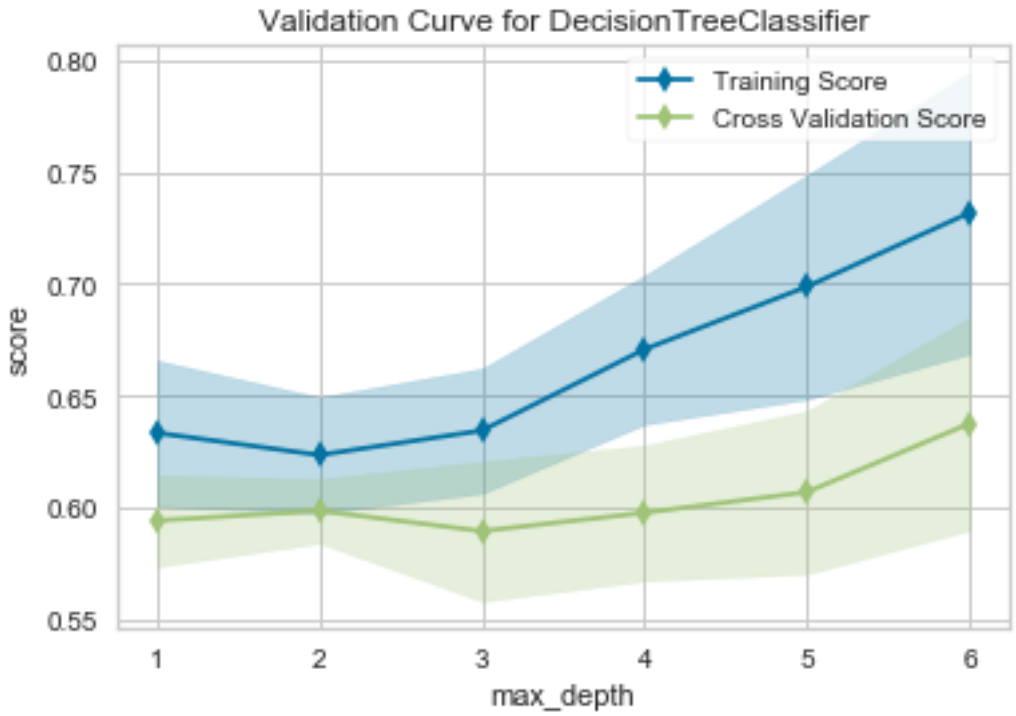
Naive Bayes

```
arrayx=[0.1,0.01,0.001,0.0001,0.00001,0.000001,0.0000001,0.00000001]
np_x=np.asarray(arrayx, dtype=np.float32)
meu.validation_curve(train_SX, y_train,GaussianNB(priors=None),"var_smoothing",np_x)
```



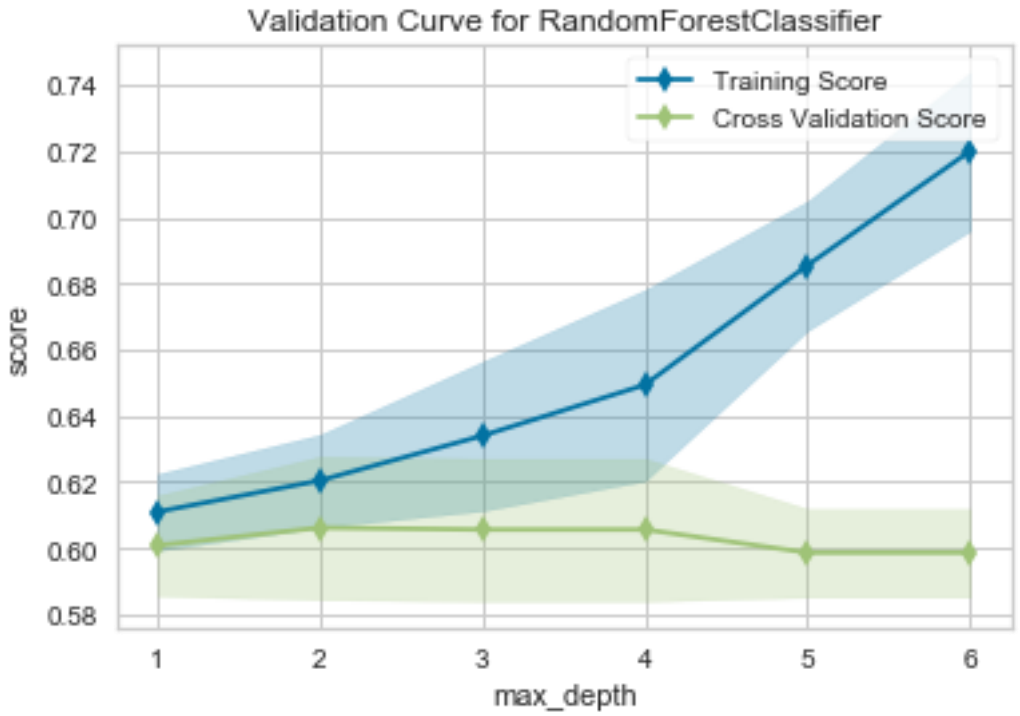
Decision Tree

```
meu.validation_curve(train_SX, y_train,
            DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                max_features=None, max_leaf_nodes=None,
               min_impurity_decrease=0.0, min_impurity_split=None,
               min_samples_leaf=1, min_samples_split=10,
               min_weight_fraction_leaf=0.0, presort='deprecated',
               random_state=7, splitter='best')
             ,"max_depth",np.arange(1,7))
```



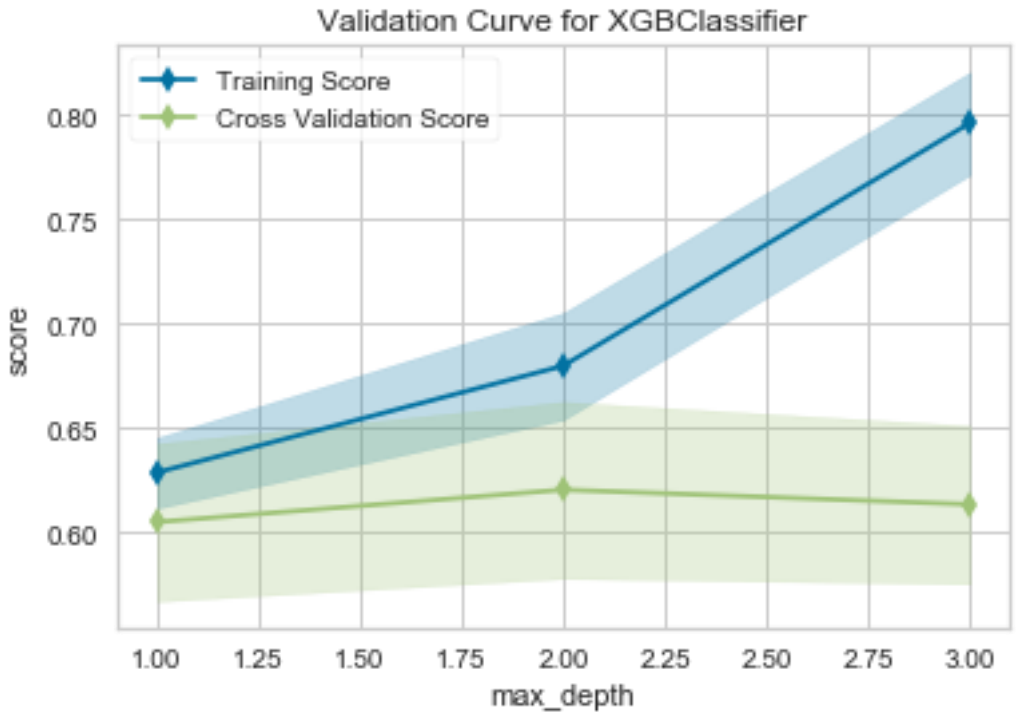
Random Forest

```
meu.validation_curve(train_SX, y_train,
            RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
              criterion='gini', max_features=None,
              max_leaf_nodes=None, max_samples=None,
              min_impurity_decrease=0.0, min_impurity_split=None,
              min_samples_leaf=1, min_samples_split=2,
              min_weight_fraction_leaf=0.0, n_estimators=500,
              n_jobs=None, oob_score=False, random_state=42, verbose=0,
              warm_start=False)
            ,"max_depth",np.arange(1,7))
```



XGBOOST

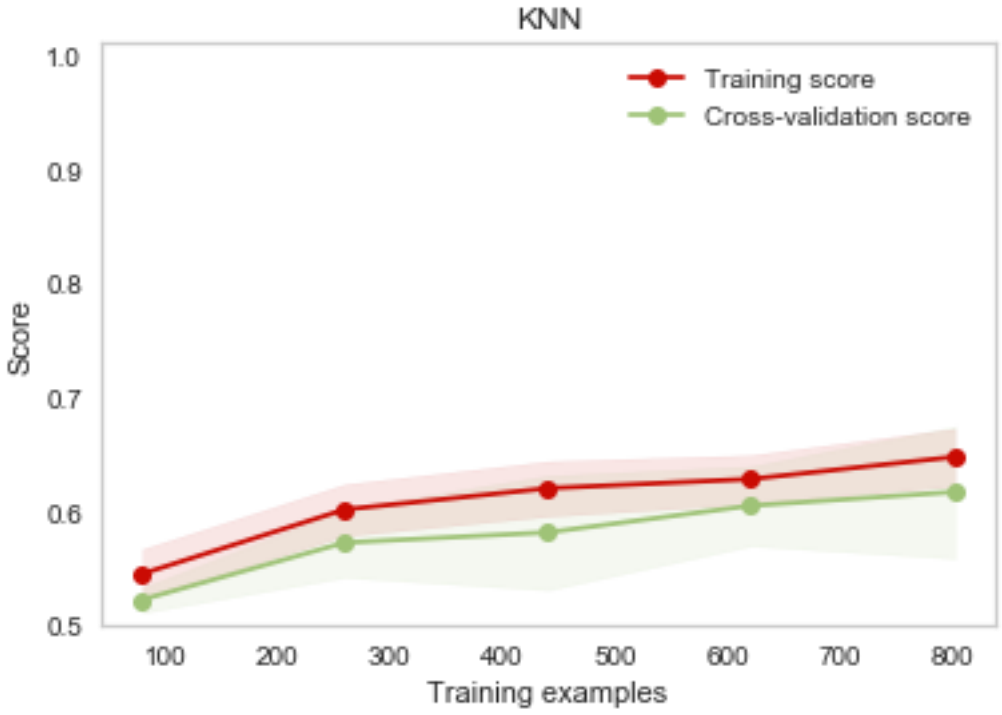
```
meu.validation_curve(train_SX, y_train,
        XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
         colsample_bynode=1, colsample_bytree=1.0, gamma=1, gpu_id=-1,
         importance_type='gain', interaction_constraints='',
         learning_rate=0.02, max_delta_step=0,
         min_child_weight=1,  monotone_constraints='()',
         n_estimators=600, n_jobs=1, nthread=1, num_parallel_tree=1,
         objective='binary:logistic', random_state=0, reg_alpha=0,
         reg_lambda=1, scale_pos_weight=1, subsample=0.8,
         tree_method='exact', validate_parameters=1, verbosity=None)
        ,"max_depth",np.arange(1,4))
```

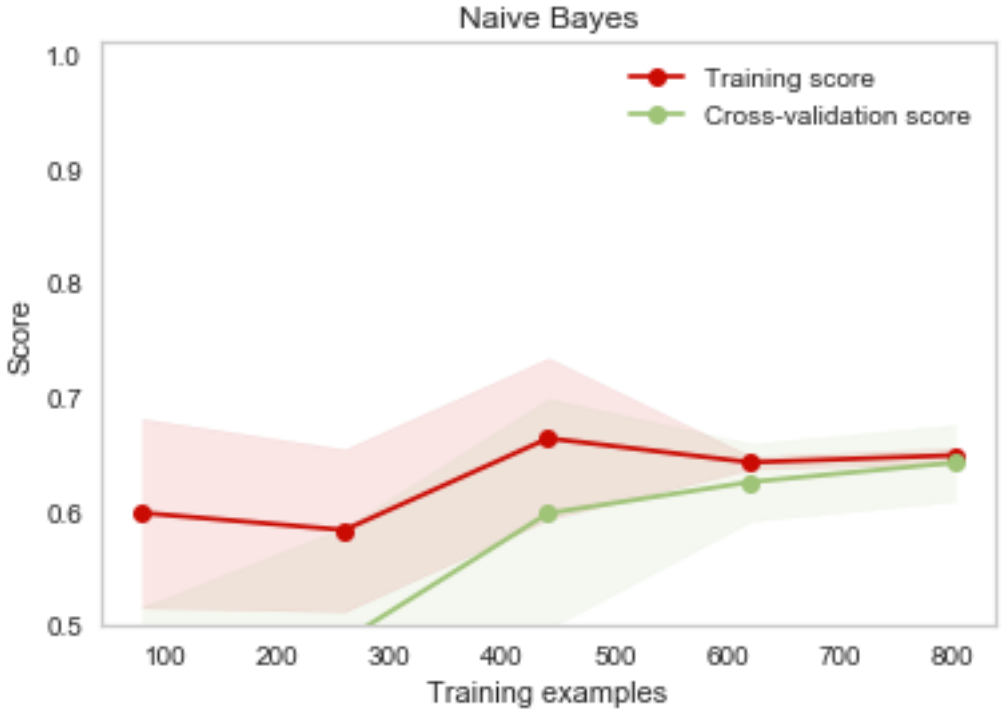


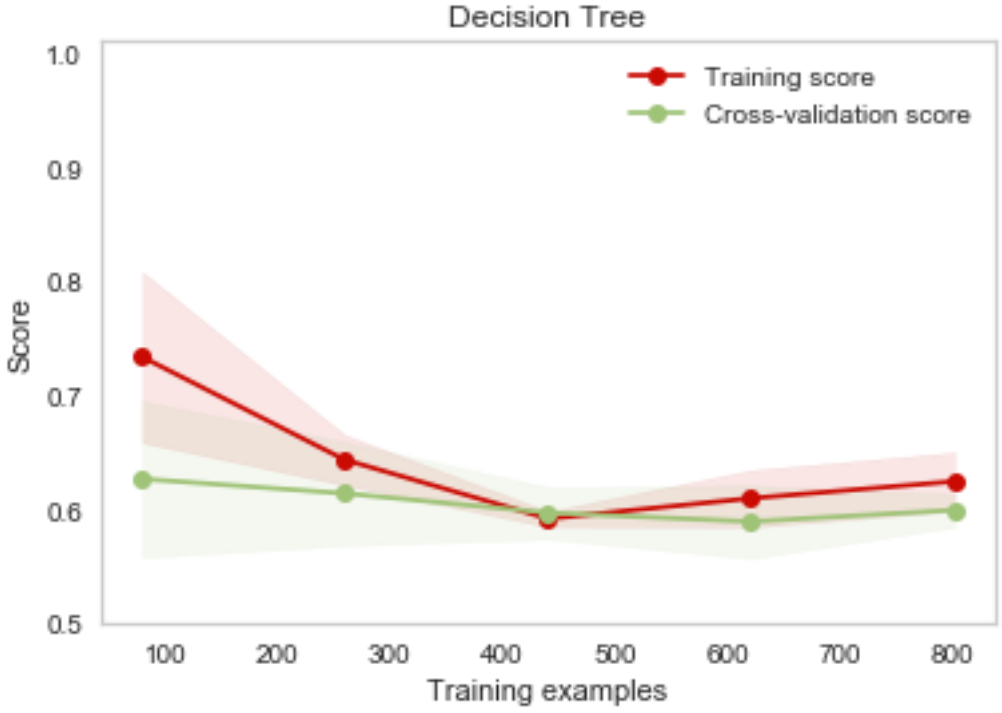
Curvas de Aprendizaje

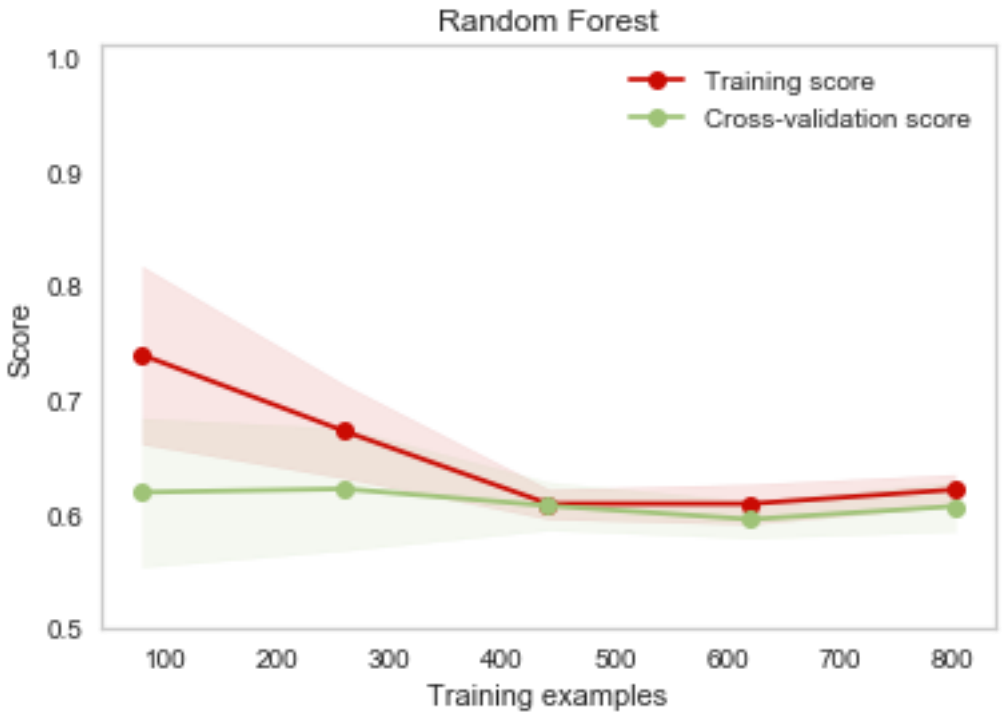
```
cv=5

array_curva = [
   ["KNN",
  KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
                metric_params=None, n_jobs=None, p=2,n_neighbors=5,
                weights='uniform')
   ]
   ,
   ["Naive Bayes",
   GaussianNB(priors=None, var_smoothing=1e-05)
   ]
   ,
   ["Decision Tree",
    DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                max_depth=2, max_features=None, max_leaf_nodes=None,
                min_impurity_decrease=0.0, min_impurity_split=None,
                min_samples_leaf=1, min_samples_split=10,
                min_weight_fraction_leaf=0.0, presort='deprecated',
                random_state=7, splitter='best')
   ]
   ,

    ["Random Forest",
    RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
```

```
            criterion='gini', max_features=None,max_depth=2,
            max_leaf_nodes=None, max_samples=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=500,
            n_jobs=None, oob_score=False, random_state=42, verbose=0,
            warm_start=False)
  ]
]
array_alg = array_curva.copy()
for w in range(0, len(array_curva)):
  meu.plot_learning_curve(array_curva[w][1], array_curva[w][0], train_SX, y_train, ylim=(0.5, 1.01), cv
=5, n_jobs=4).show()
```




Naive Bayes
Training score
Cross-validation score
Score
1.0
0.9
0.8
0.7
0.6
0.5
100
200
300
400
500
600
700
800
Training examples


Decision Tree
Training score
Cross-validation score
Score
1.0
0.9
0.8
0.7
0.6
0.5
100
200
300
400
500
600
700
800
Training examples



Matriz de Confusion

```
#tecnicas de balanceo de clases
from imblearn.combine import SMOTEENN, SMOTETomek
from imblearn.over_sampling import RandomOverSampler
from imblearn.under_sampling import TomekLinks,RandomUnderSampler


list_balanced =[
  ["None",
  None
  ]
  ,
  ["RandomUnderSampler",
  RandomUnderSampler()
  ]
  ,
  ["RandomOverSampler",
  RandomOverSampler()
  ]
  ,
  ["SMOTETomek",
  SMOTETomek()
  ]

]
```

```
array_final=[]
for w in range(0, len(array_alg)):
  for q in range(0, len(list_balanced)):
    array_file=[array_alg[w][0]+"_"+list_balanced[q][0],
          array_alg[w][1],
          list_balanced[q][0],
          list_balanced[q][1]
         ]
    array_final.append(array_file)


array_final[2]
```

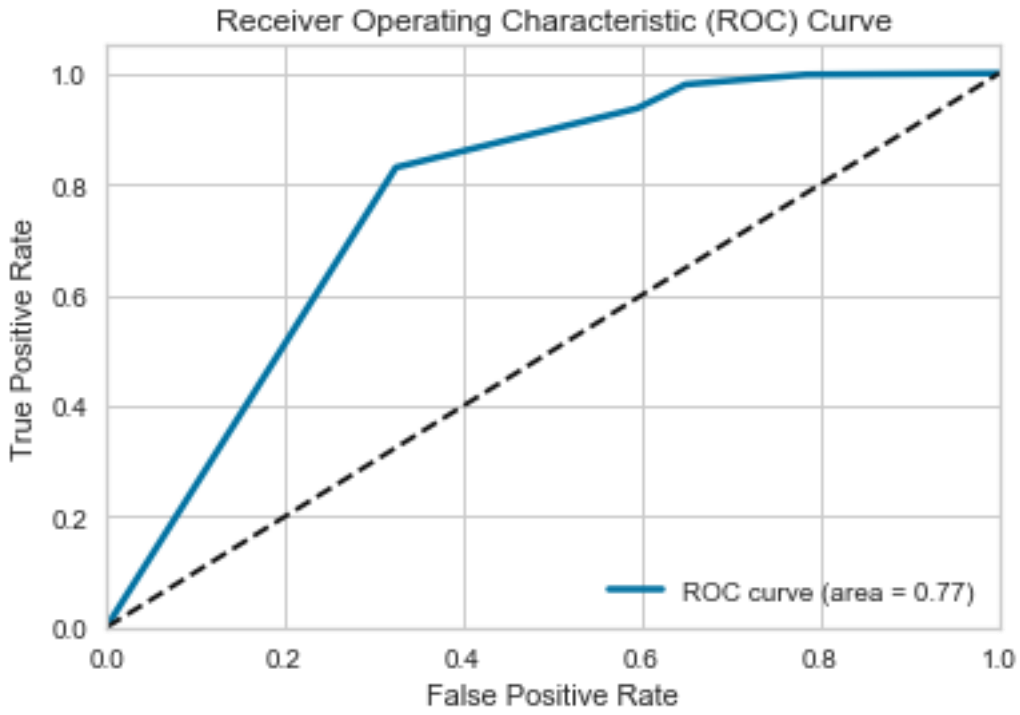
Out[38]:

```
['KNN_RandomOverSampler',
 KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
                      metric_params=None, n_jobs=None, n_neighbors=5, p=2
,
                      weights='uniform'),
 'RandomOverSampler',
 RandomOverSampler(random_state=None, sampling_strategy='auto')]
```

```
rows_list_score = []
for x in range(0, len(array_final)):
    print(array_final[x][0])
    print("----------------------------------------")
    print("----------------------------------------")
    if(array_final[x][2]=="None"):
      X_train_tecnica = train_SX
      y_train_tecnica = y_train
    else:
      X_train_tecnica, y_train_tecnica = array_final[x][3].fit_sample(train_SX, y_train)

    model0=array_final[x][1]

    model0.fit(X_train_tecnica, y_train_tecnica)

    wqp_rf_predictions = model0.predict(X_test)

    balanced_score,tpr = meu.display_model_performance_metrics(true_labels=y_test, predicted_label
s=wqp_rf_predictions,
                          classes=wqp_label_names)
    meu.plot_model_roc_curve(model0, X_test, y_test)
    rows_list_score.append([array_final[x][0],balanced_score,tpr])
    print("----------------------------------")
    print("----------------------------------")
```

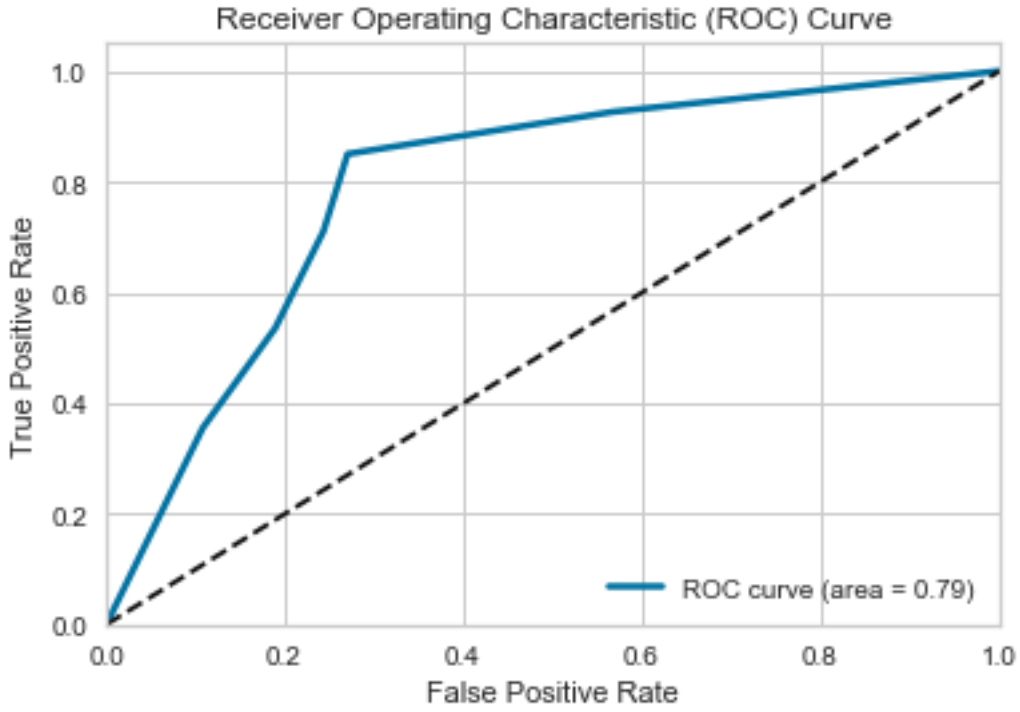
```
KNN_None
```

```
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           13           24
NO DESERTOR         8          386
Model Performance metrics:
------------------------------
TPR: 0.3514
Balanced Accuracy: 0.6655
Precision: 0.9138
Recall: 0.9258
F1 Score: 0.9163

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.62      0.35      0.45        37
 NO DESERTOR       0.94      0.98      0.96       394

    accuracy                           0.93       431
   macro avg       0.78      0.67      0.70       431
weighted avg       0.91      0.93      0.92       431
```




```
-----------------------------------
-----------------------------------
KNN_RandomUnderSampler
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           28            9
NO DESERTOR       114          280
Model Performance metrics:
------------------------------
TPR: 0.7568
Balanced Accuracy: 0.7337
Precision: 0.9026
Recall: 0.7146
F1 Score: 0.7764

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.20      0.76      0.31        37
 NO DESERTOR       0.97      0.71      0.82       394
```

accuracy			0.71	431
macro avg	0.58	0.73	0.57	431
weighted avg	0.90	0.71	0.78	431



KNN_RandomOverSampler

Prediction Confusion Matrix:

	DESERTOR	NO DESERTOR
DESERTOR	19	18
NO DESERTOR	42	352

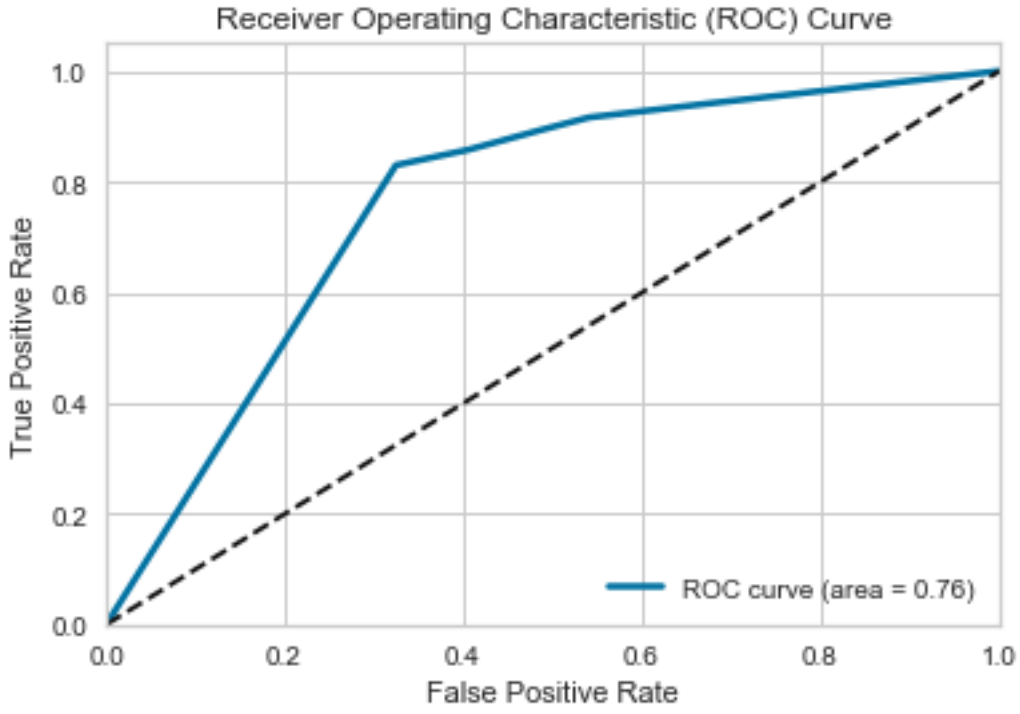
Model Performance metrics:

TPR: 0.5135
Balanced Accuracy: 0.7035
Precision: 0.8964
Recall: 0.8608
F1 Score: 0.8756

Model Classification report:
---

	precision	recall	f1-score	support

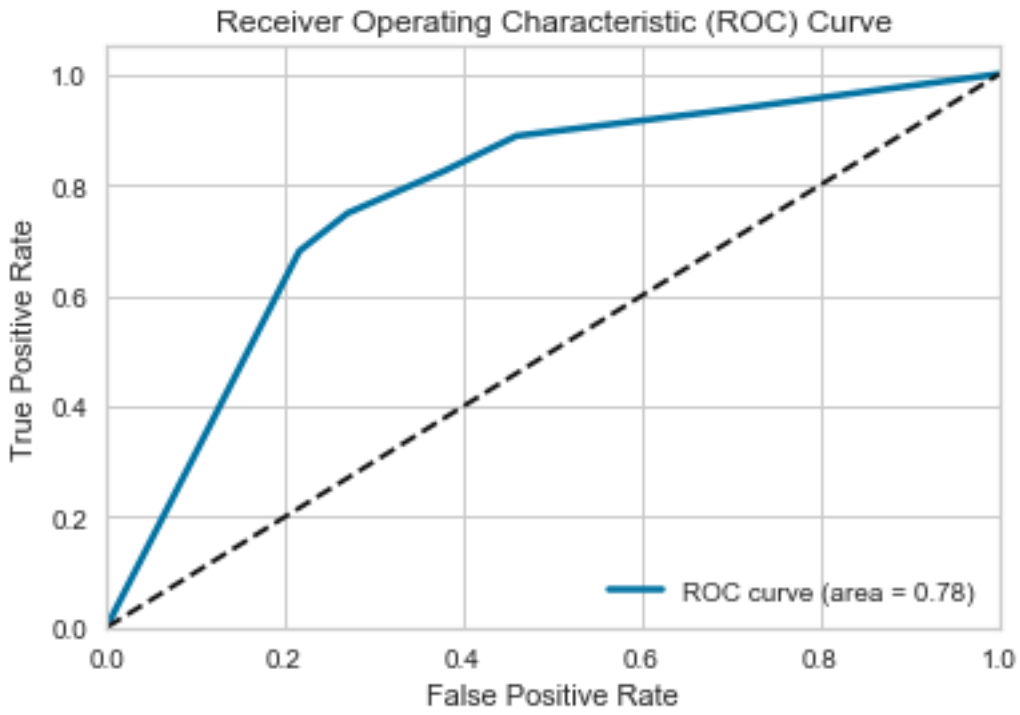
```
    DESERTOR       0.31      0.51      0.39        37
 NO DESERTOR       0.95      0.89      0.92       394

    accuracy                           0.86       431
   macro avg       0.63      0.70      0.65       431
weighted avg       0.90      0.86      0.88       431
```




```
-----------------------------------
-----------------------------------
KNN_SMOTETomek
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           23           14
NO DESERTOR        69          325
Model Performance metrics:
------------------------------
TPR: 0.6216
Balanced Accuracy: 0.7232
Precision: 0.8979
Recall: 0.8074
F1 Score: 0.8413
```

Model Classification report:
----

	precision	recall	f1-score	support
DESERTOR	0.25	0.62	0.36	37
NO DESERTOR	0.96	0.82	0.89	394
accuracy			0.81	431
macro avg	0.60	0.72	0.62	431
weighted avg	0.90	0.81	0.84	431



Naive Bayes_None

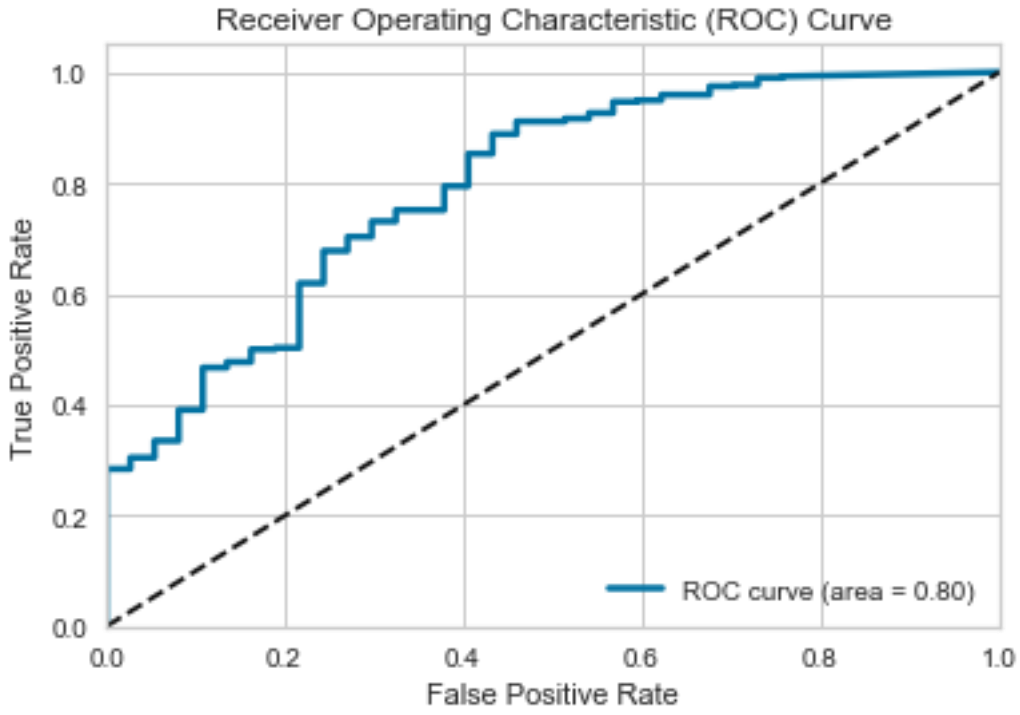
Prediction Confusion Matrix:

	DESERTOR	NO DESERTOR
DESERTOR	16	21
NO DESERTOR	26	368

Model Performance metrics:

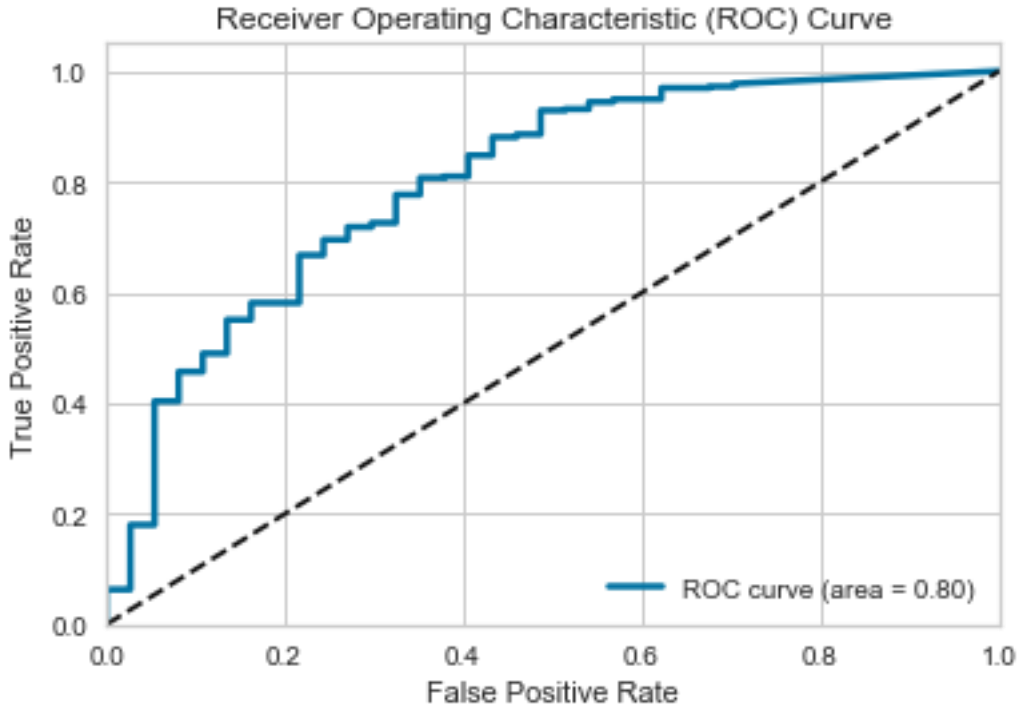
TPR: 0.4324

```
Balanced Accuracy: 0.6832
Precision: 0.8975
Recall: 0.891
F1 Score: 0.8941

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.38      0.43      0.41        37
 NO DESERTOR       0.95      0.93      0.94       394

    accuracy                           0.89       431
   macro avg       0.66      0.68      0.67       431
weighted avg       0.90      0.89      0.89       431
```



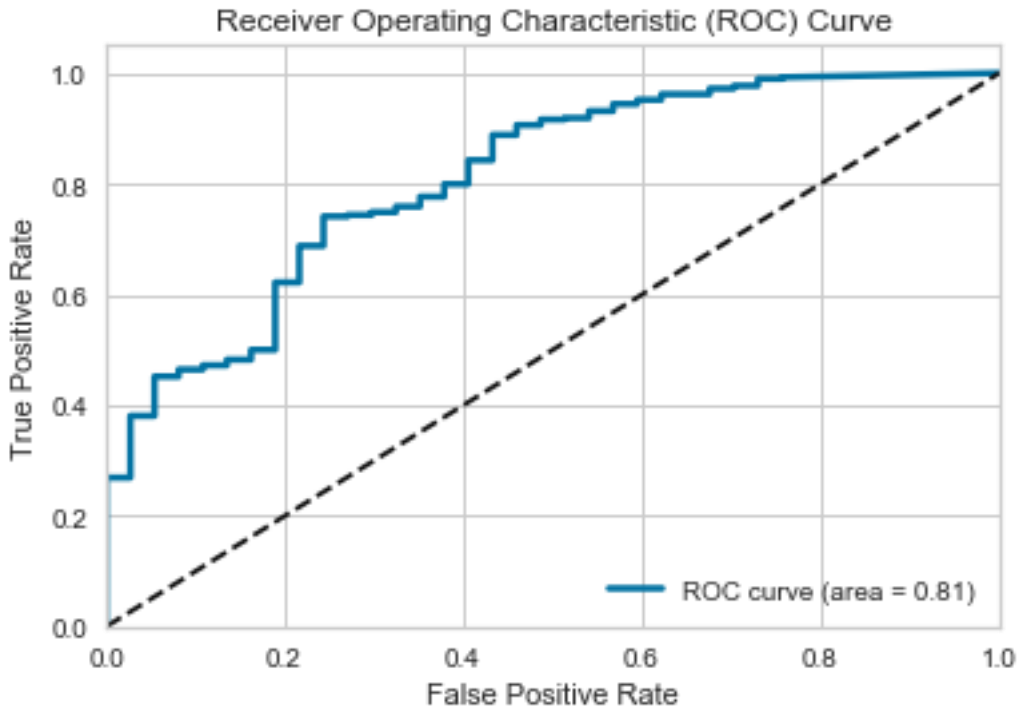

```
-----------------------------------
-----------------------------------
Naive Bayes_RandomUnderSampler
-----------------------------------------
-----------------------------------------


Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
```

```
DESERTOR           14           23
NO DESERTOR        15          379
Model Performance metrics:
------------------------------
TPR: 0.3784
Balanced Accuracy: 0.6702
Precision: 0.9033
Recall: 0.9118
F1 Score: 0.9069

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.48      0.38      0.42        37
 NO DESERTOR       0.94      0.96      0.95       394

    accuracy                           0.91       431
   macro avg       0.71      0.67      0.69       431
weighted avg       0.90      0.91      0.91       431
```

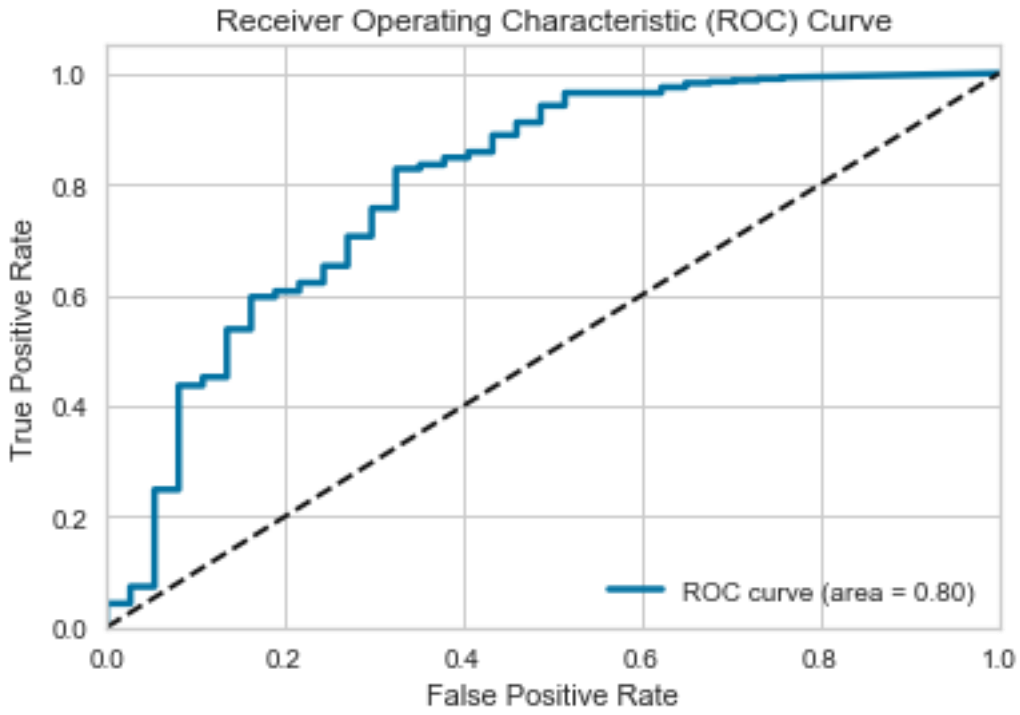



```
-----------------------------------
-----------------------------------
Naive Bayes_RandomOverSampler
-----------------------------------------
-----------------------------------------
```

```
Prediction Confusion Matrix:
------------------------------
              DESERTOR  NO DESERTOR
DESERTOR            17           20
NO DESERTOR         31          363
Model Performance metrics:
------------------------------
TPR: 0.4595
Balanced Accuracy: 0.6904
Precision: 0.8968
Recall: 0.8817
F1 Score: 0.8885

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.35      0.46      0.40        37
 NO DESERTOR       0.95      0.92      0.93       394

    accuracy                           0.88       431
   macro avg       0.65      0.69      0.67       431
weighted avg       0.90      0.88      0.89       431
```

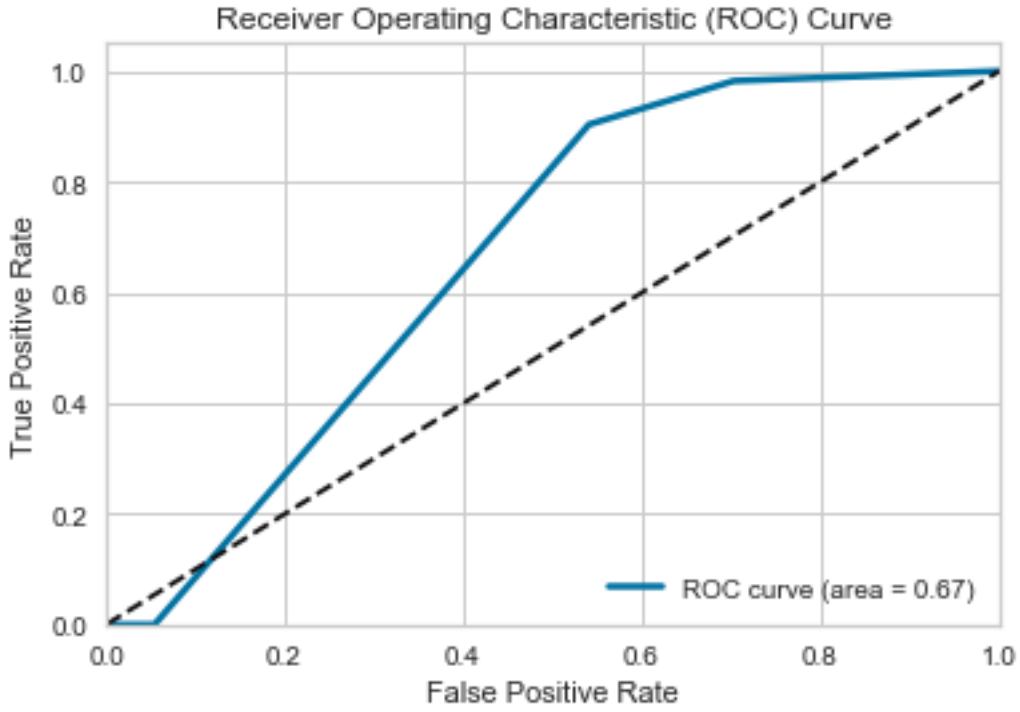


```
-----------------------------------
Naive Bayes_SMOTETomek
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           19           18
NO DESERTOR        25          369
Model Performance metrics:
------------------------------
TPR: 0.5135
Balanced Accuracy: 0.725
Precision: 0.9087
Recall: 0.9002
F1 Score: 0.9041

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.43      0.51      0.47        37
 NO DESERTOR       0.95      0.94      0.94       394

    accuracy                           0.90       431
   macro avg       0.69      0.73      0.71       431
weighted avg       0.91      0.90      0.90       431
```



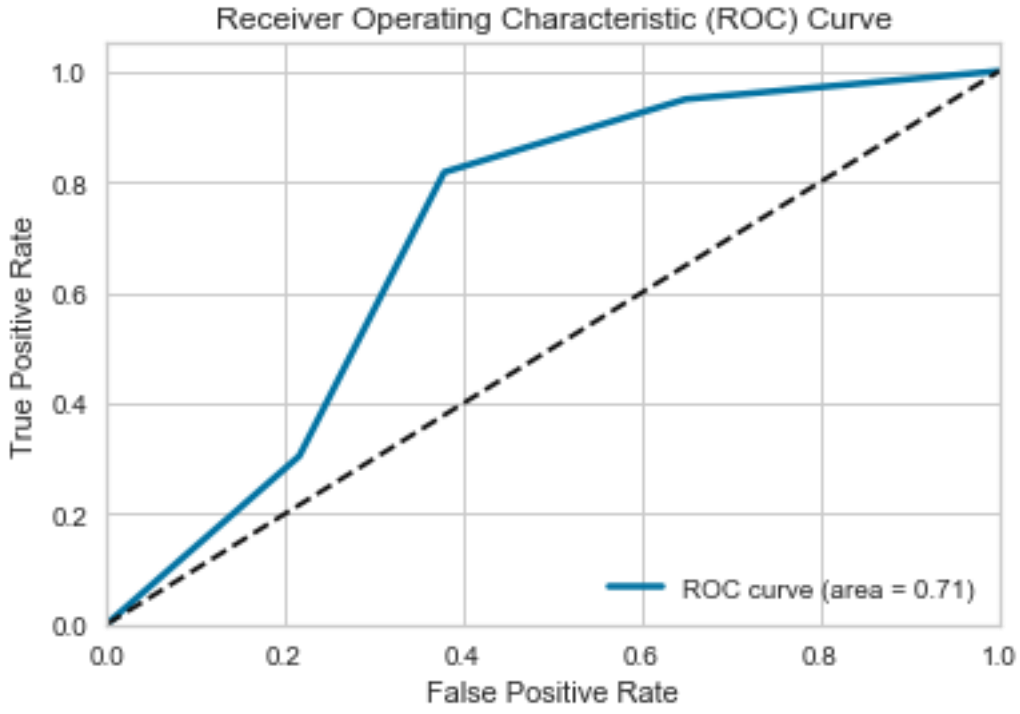

```
-----------------------------------
-----------------------------------
Decision Tree_None
-------------------------------------------
-------------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           11           26
NO DESERTOR         7          387
Model Performance metrics:
------------------------------
TPR: 0.2973
Balanced Accuracy: 0.6398
Precision: 0.9091
Recall: 0.9234
F1 Score: 0.9111

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.61      0.30      0.40        37
 NO DESERTOR       0.94      0.98      0.96       394
```

accuracy			0.92	431
macro avg	0.77	0.64	0.68	431
weighted avg	0.91	0.92	0.91	431




```
-----------------------------------
-----------------------------------
Decision Tree_RandomUnderSampler
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
-----------------------------
              DESERTOR  NO DESERTOR
DESERTOR            23           14
NO DESERTOR         72          322
Model Performance metrics:
-----------------------------
TPR: 0.6216
Balanced Accuracy: 0.7194
Precision: 0.8968
Recall: 0.8005
F1 Score: 0.8364

Model Classification report:
-----------------------------
              precision    recall  f1-score   support
```

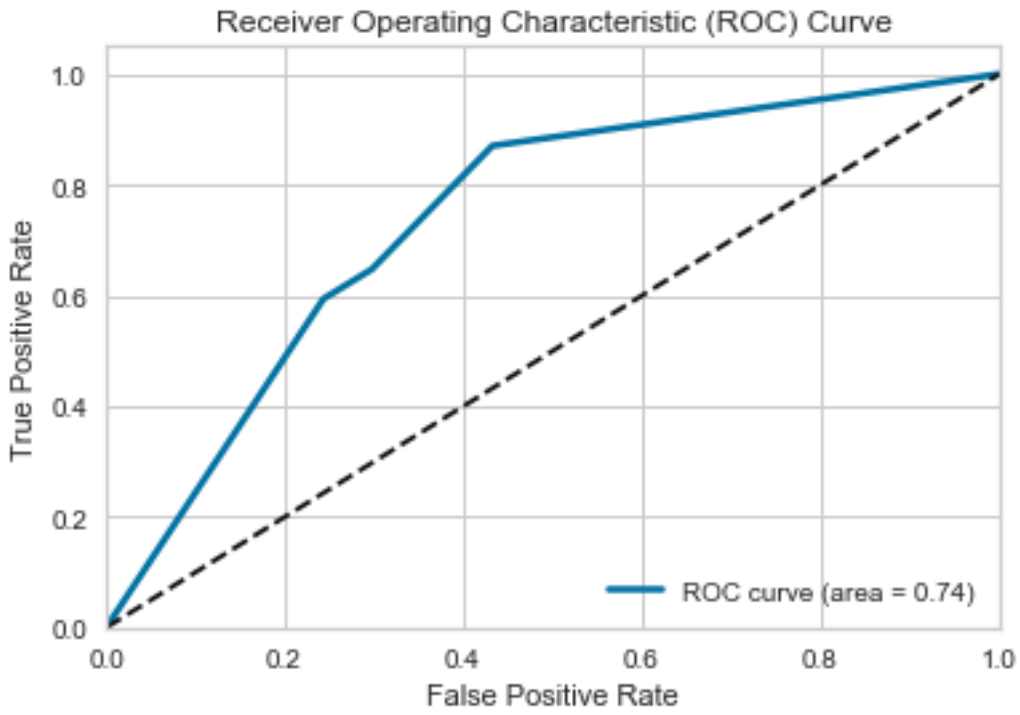
```
    DESERTOR       0.24      0.62      0.35        37
 NO DESERTOR       0.96      0.82      0.88       394

    accuracy                           0.80       431
   macro avg       0.60      0.72      0.62       431
weighted avg       0.90      0.80      0.84       431
```




```
-----------------------------------
-----------------------------------
Decision Tree_RandomOverSampler
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           26           11
NO DESERTOR       139          255
Model Performance metrics:
------------------------------
TPR: 0.7027
Balanced Accuracy: 0.675
Precision: 0.8899
Recall: 0.652
F1 Score: 0.7285
```

Model Classification report:
----

	precision	recall	f1-score	support
DESERTOR	0.16	0.70	0.26	37
NO DESERTOR	0.96	0.65	0.77	394
accuracy			0.65	431
macro avg	0.56	0.67	0.52	431
weighted avg	0.89	0.65	0.73	431



Decision Tree_SMOTETomek

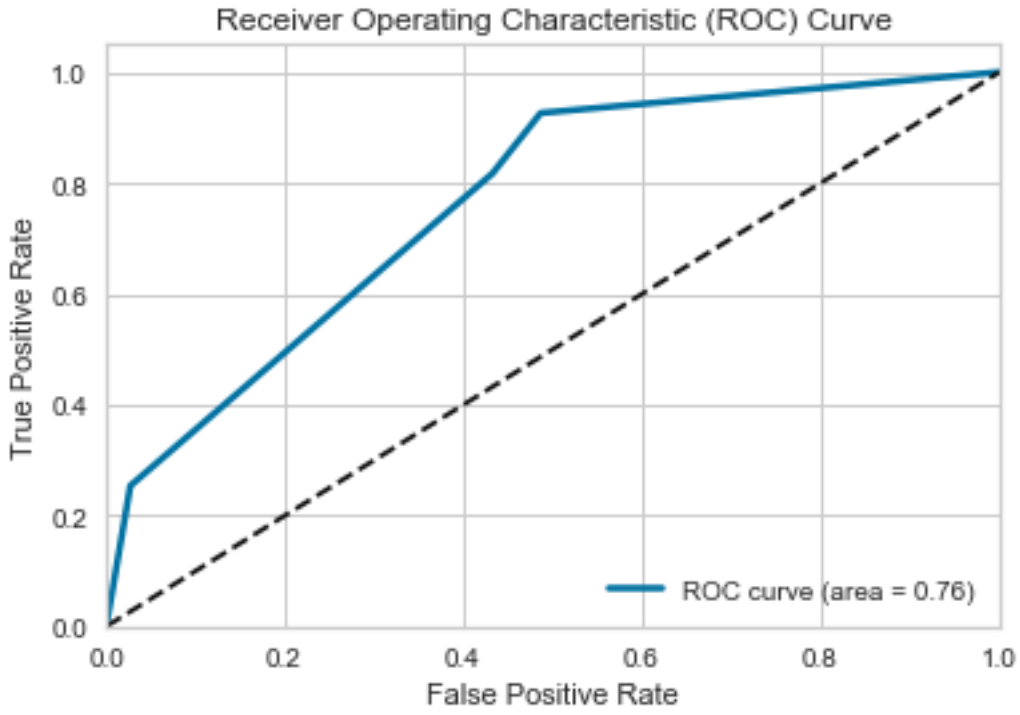
Prediction Confusion Matrix:

	DESERTOR	NO DESERTOR
DESERTOR	21	16
NO DESERTOR	72	322

Model Performance metrics:

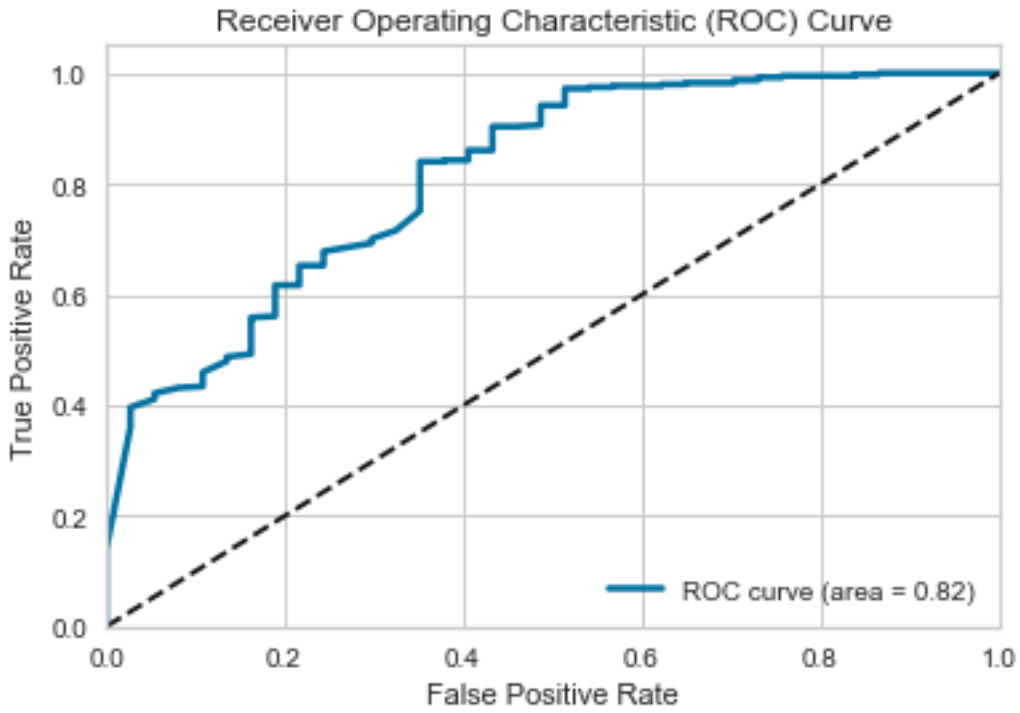
TPR: 0.5676

```
Balanced Accuracy: 0.6924
Precision: 0.8903
Recall: 0.7958
F1 Score: 0.832

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.23      0.57      0.32        37
 NO DESERTOR       0.95      0.82      0.88       394

    accuracy                           0.80       431
   macro avg       0.59      0.69      0.60       431
weighted avg       0.89      0.80      0.83       431
```



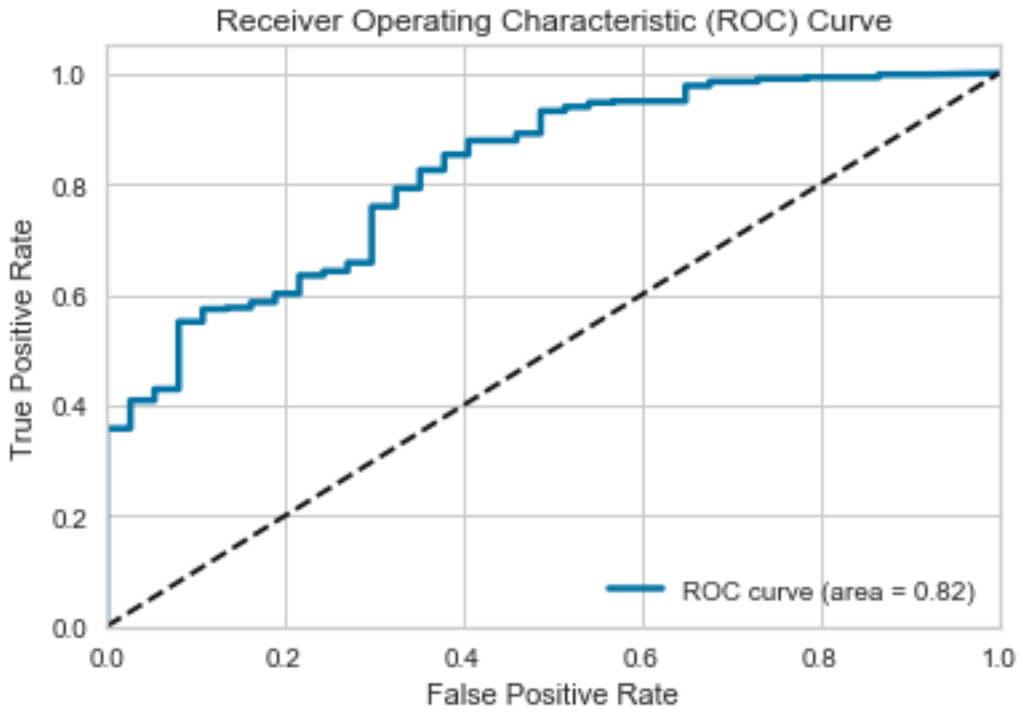

```
-----------------------------------
-----------------------------------
Random Forest_None
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           11           26
```

```
NO DESERTOR         7          387
Model Performance metrics:
------------------------------
TPR: 0.2973
Balanced Accuracy: 0.6398
Precision: 0.9091
Recall: 0.9234
F1 Score: 0.9111

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.61      0.30      0.40        37
 NO DESERTOR       0.94      0.98      0.96       394

    accuracy                           0.92       431
   macro avg       0.77      0.64      0.68       431
weighted avg       0.91      0.92      0.91       431
```

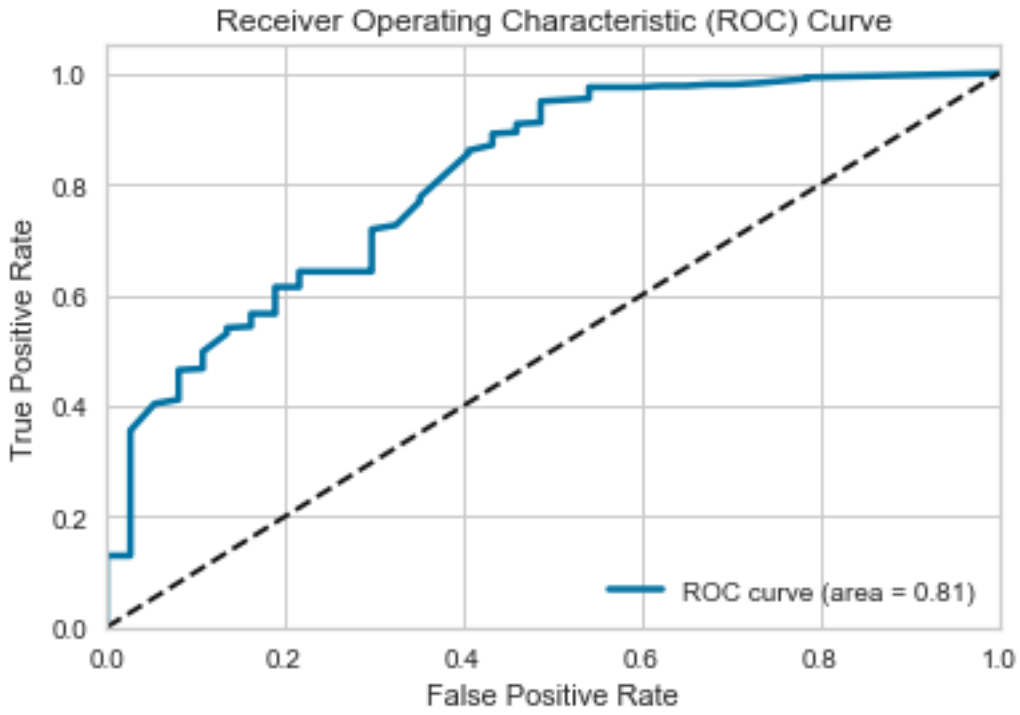



```
-----------------------------------
-----------------------------------
Random Forest_RandomUnderSampler
-----------------------------------------
-----------------------------------------
```

```
Prediction Confusion Matrix:
------------------------------
              DESERTOR  NO DESERTOR
DESERTOR            25           12
NO DESERTOR         92          302
Model Performance metrics:
------------------------------
TPR: 0.6757
Balanced Accuracy: 0.7211
Precision: 0.8976
Recall: 0.7587
F1 Score: 0.8077

Model Classification report:
------------------------------
               precision    recall  f1-score   support

    DESERTOR       0.21      0.68      0.32        37
 NO DESERTOR       0.96      0.77      0.85       394

    accuracy                           0.76       431
   macro avg       0.59      0.72      0.59       431
weighted avg       0.90      0.76      0.81       431
```

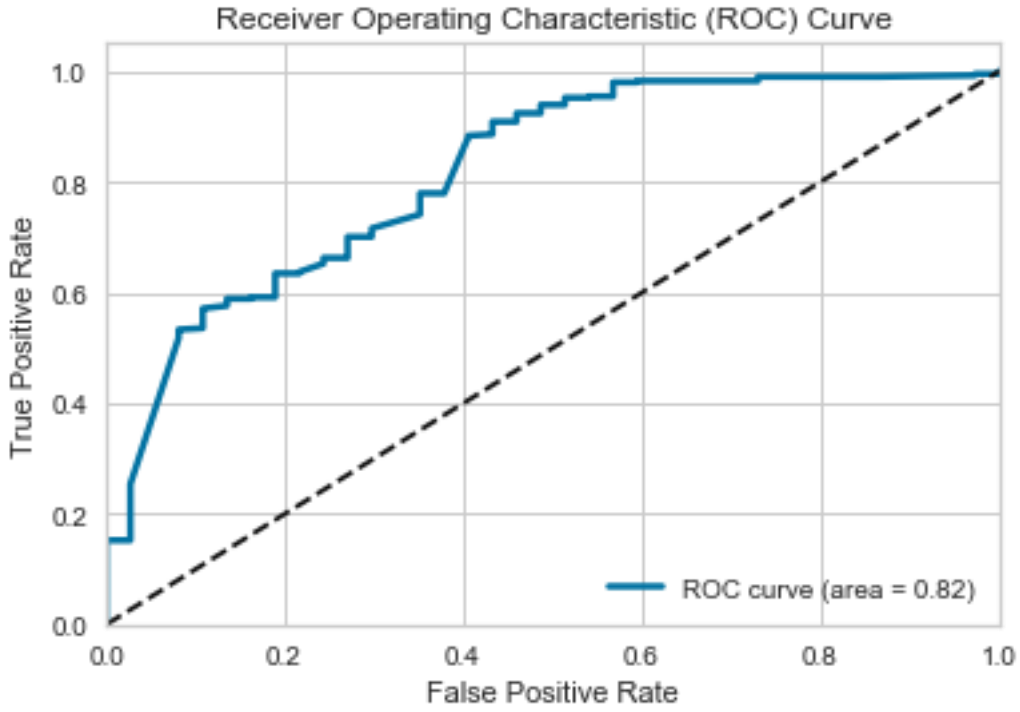



```
Random Forest_RandomOverSampler
----------------------------------------
----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           26           11
NO DESERTOR       113          281
Model Performance metrics:
------------------------------
TPR: 0.7027
Balanced Accuracy: 0.708
Precision: 0.8958
Recall: 0.7123
F1 Score: 0.7743

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.19      0.70      0.30        37
 NO DESERTOR       0.96      0.71      0.82       394

    accuracy                           0.71       431
   macro avg       0.57      0.71      0.56       431
weighted avg       0.90      0.71      0.77       431
```




```
-----------------------------------
-----------------------------------
Random Forest_SMOTETomek
-----------------------------------------
-----------------------------------------

Prediction Confusion Matrix:
------------------------------
             DESERTOR  NO DESERTOR
DESERTOR           21           16
NO DESERTOR        43          351
Model Performance metrics:
------------------------------
TPR: 0.5676
Balanced Accuracy: 0.7292
Precision: 0.9025
Recall: 0.8631
F1 Score: 0.879

Model Classification report:
------------------------------
              precision    recall  f1-score   support

    DESERTOR       0.33      0.57      0.42        37
 NO DESERTOR       0.96      0.89      0.92       394
```

```
    accuracy                           0.86       431
   macro avg       0.64      0.73      0.67       431
weighted avg       0.90      0.86      0.88       431
```




```
-----------------------------------
-----------------------------------
```

rows_list_score

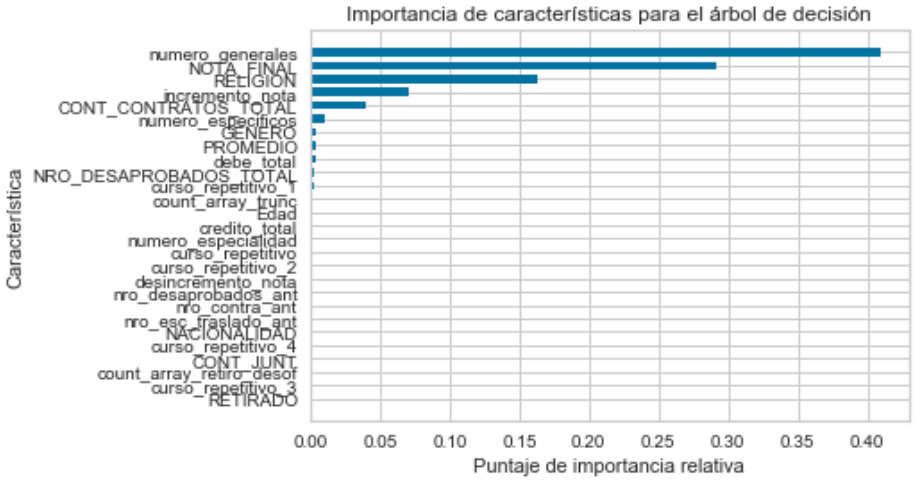
Out[40]:

```
[['KNN_None', 0.6655233914117163, 0.35135135135135137],
 ['KNN_RandomUnderSampler', 0.7337083276169571, 0.7567567567567568],
 ['KNN_RandomOverSampler', 0.70345726437097, 0.5135135135135135],
 ['KNN_SMOTETomek', 0.723247359034161, 0.6216216216216216],
 ['Naive Bayes_None', 0.6832212923583483, 0.43243243243243246],
 ['Naive Bayes_RandomUnderSampler', 0.6701536561942654, 0.3783783783783784],
 ['Naive Bayes_RandomOverSampler', 0.6903896282068871, 0.4594594594594595],
 ['Naive Bayes_SMOTETomek', 0.7250308684318836, 0.5135135135135135],
 ['Decision Tree_None', 0.6397653999176842, 0.2972972972972973],
 ['Decision Tree_RandomUnderSampler', 0.7194402524351763, 0.6216216216216216],
 ['Decision Tree_RandomOverSampler', 0.674955412265057, 0.7027027027027027],
 ['Decision Tree_SMOTETomek', 0.6924132254081492, 0.5675675675675675],
 ['Random Forest_None', 0.6397653999176842, 0.2972972972972973],
```

```
 ['Random Forest_RandomUnderSampler', 0.7210865688023048, 0.6756756756756
757],
 ['Random Forest_RandomOverSampler', 0.7079503361229249, 0.70270270270270
27],
 ['Random Forest_SMOTETomek', 0.729215255865002, 0.5675675675675675]]
```

Ver las características importantes del modelo de árbol de decisión

```
wqp_dt_feature_importances = array_final[14][1].feature_importances_
wqp_dt_feature_names, wqp_dt_feature_scores = zip(*sorted(zip(df.columns, wqp_dt_feature_importan
ces),
                                                  key=lambda x: x[1]))
y_position = list(range(len(wqp_dt_feature_names)))
plt.barh(y_position, wqp_dt_feature_scores, height=0.6, align='center')
plt.yticks(y_position , wqp_dt_feature_names)
plt.xlabel('Puntaje de importancia relativa')
plt.ylabel('Característica')
t = plt.title('Importancia de características para el árbol de decisión')
```



 # ['KNN_RandomUnderSampler', 0.7337083276169571, 0.7567567567567568],
 # ['Random Forest_RandomOverSampler', 0.7079503361229249, 0.7027027027027027],

Exportar el Modelo Predictivo Eficaz

```
from imblearn.combine import SMOTEENN, SMOTETomek
from imblearn.over_sampling import RandomOverSampler
from imblearn.under_sampling import TomekLinks,RandomUnderSampler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from xgboost import XGBClassifier
```

```
X_train_tecnica, y_train_tecnica = RandomUnderSampler().fit_sample(train_SX, y_train)
model = KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
            metric_params=None, n_jobs=None, p=2,n_neighbors=5,
            weights='uniform')

model.fit(X_train_tecnica,y_train_tecnica)
```

Out[30]:

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='euclidean',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
```

```
from sklearn.externals import joblib
joblib.dump(model, 'model_joblib/model_alimentos.pkl')
```

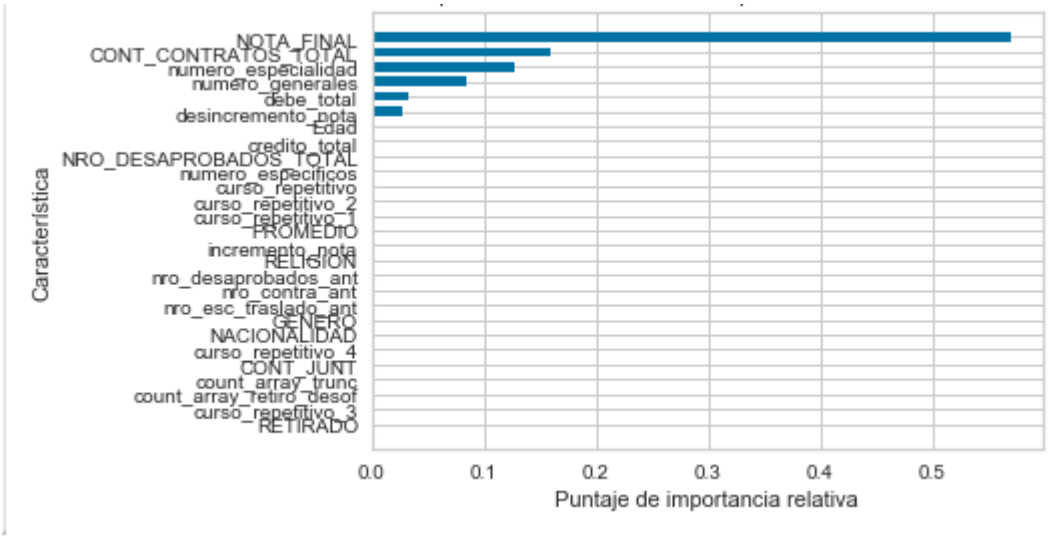
Out[31]:

```
['model_joblib/model_alimentos.pkl']
```

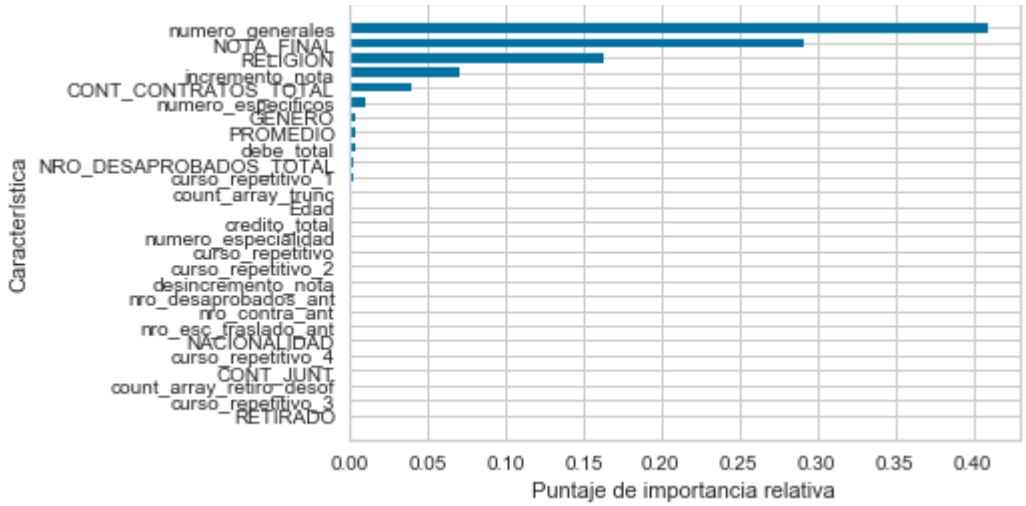
Anexos 3. Codigo utilizado para encontrar el Modelo Predictivo Eficaz

Las variables mas importantes de cada carrera si ha sido un modelo Decision Tree, Random Forest o XGBOOST:

- Ingeniería de Sistemas



- Ingeniería de Alimentos



- Ingeniería de Civil

Anexos 4. Codigo del API Rest

```
# -*- coding: utf-8 -*-
"""
Spyder Editor
This is a temporary script file.
"""
from flask import Flask,jsonify,request
#from sklearn.externals import joblib
from joblib import dump, load
import pandas as pd
import numpy as np
import csv, operator
numeric_feature_cols=[
'NOTA_FINAL','incremento_nota','desincremento_nota','PROMEDIO'
,'curso_repetitivo_1','curso_repetitivo_2' ,'curso_repetitivo'
,'numero_generales','numero_especificos','numero_especialidad','NRO_DESAPROBADO
S_TOTAL','CONT_CONTRATOS_TOTAL','debe_total','credito_total','Edad']
#Funcion para escalar
def scale_data(array,means,vars):
    stds=vars **0.5
    return (array-means)/stds
def obtener_standar(escuela):
     print(escuela)
     if escuela=="civil":
         array_means=[14.372510674637061,
1.4344833475661825,2.466865926558497,15.016920095756078,1.9974380871050383,0.
```

26515798462852264,0.30187873612297184,0.4077711357813834,0.7280102476515798,
1.5055508112724167,2.8582408198121265,4.571306575576431,4067.214863364646,445
.0158198121264,20.107600341588387]

array_vars=[6.96745996465975,3.991403127604849,5.986890775363339,2.60519767113
5698,7.818097621060046,0.6115870587300116,0.8102355822207766,1.51049469057590
91,1.5771744354201775,7.6752467290578394,21.431654975310686,7.823907686540877
,17827075.979851283,1811549.7327211683,7.49653489065744]
        clf_from_joblib=load('model_civil.pkl')
    elif escuela=="sistemas":

array_means=[14.098004147226543,1.5808657335406946,2.491575946086055,14.66758
1063865859,1.9476412649040953,0.22965266977708657,0.26283048211508553,0.72939
34681181959,0.7480559875583204,1.0326594090202177,2.589942975635044,4.4116122
343183,2753.7747796785898,243.8942716433385,19.840850181441162]

array_vars=[8.459548478987466,6.416839323081901,9.106855888607516,2.7396188329
03685,7.087460740153092,0.6144447212487043,0.8396811537416548,2.2844703941719
535,1.7623406998202922,3.8646669036971106,16.875399115462585,8.58640740588453
6,16641128.941720339,377397.676322033,8.667776571126199]
        clf_from_joblib=load('model_sistemas.pkl')
    elif escuela=="arquitectura":

array_means=[13.569535367545074,1.9527773925104024,3.0031865464632452,14.3022
43616667255,2.98127600554785,0.4237170596393897,0.526005547850208,1.133148404
9930651,0.7503467406380028,2.2680305131761442,4.2617891816920945,4.8002773925
10402,3403.858796809986,805.5268932038836,19.850554785020805]

array_vars=[8.32201656636356,8.0120827507233,12.504564013744107,2.679818771502
5468,11.152908774028985,0.9889798034398978,1.471237719802786,3.77075971306611
,1.7545941932244664,9.08800707812581,29.94498930923109,10.410180228185155,291
83568.1563343,19566384.630152203,9.186057251159488]
        clf_from_joblib=load('model_arquitectura.pkl')
    elif escuela=="alimentos":

array_means=[15.090997008973082,1.7089631106679959,1.6542572283150552,15.2097
17087457252,1.2283150548354935,0.1016949152542373,0.10767696909272183,0.44566
30109670987,0.7417746759720838,0.2622133599202393,1.5732801595214356,4.545363
908275174,3032.683589232303,261.62736789631106,20.083748753738785]

array_vars=[4.321966304078791,3.610442992657123,3.565111586874471,2.1612341980
3184,3.825240132046533,0.4343241462054515,0.4530118517826381,2.34276035303859
1,1.5434891735561014,0.6500876234705653,8.826883258499677,7.990713800771166,1
7612008.223453917,596927.7703298617,8.140543474263152]
        clf_from_joblib=load('model_alimentos.pkl')
    elif escuela=="ambiental":

```
array_means=[14.724427983539094,1.6893909465020576,2.0802139917695475,15.0947
75785208002,1.6930041152263375,0.24691358024691357,0.2927297668038409,0.65486
96844993142,1.1887517146776405,0.4839506172839506,2.437037037037037,4.8614540
46639232,3498.517138545953,340.7324773662552,20.174485596707818]

array_vars=[6.902077237954919,4.525061741536689,6.399846373960608,2.4405332139
868703,7.6662473538925315,0.7648224356043285,1.1952420682634572,2.43122800085
0518,5.388789799808444,1.902417314433775,21.850425240054868,8.331696651180469
,21154956.07520435,813210.5699836843,8.334438178461955]
        clf_from_joblib=load('model_ambiental.pkl')
     means2=np.array(array_means)
     vars2=np.array(array_vars)
     return means2,vars2,clf_from_joblib
app = Flask(__name__)
@app.route("/student",methods=["POST"])
def student():
    escuela=str(request.json['carrera'])
    means_ , vars_ ,clf_from_joblib_= obtener_standar(escuela)
    genero=int(request.json['genero']) ##GENERO'
    retirado_count=int(request.json['retirado_count'])#count_array_retiro_desof
    desertor_count=int(request.json['desertor_count'])#count_array_trunc
    change_sede=int(request.json['change_sede'])#CONT_JUNT
    esc_traslado=int(request.json['esc_traslado'])#nro_esc_traslado_ant
    contra_ant=int(request.json['contra_ant'])#nro_contra_ant
    desaprobado_ant=int(request.json['desaprobado_ant'])#nro_desaprobados_ant
    contra_total=int(request.json['contra_total'])#CONT_CONTRATOS_TOTAL

desaprobado_total=int(request.json['desaprobado_total'])#NRO_DESAPROBADOS_TOT
AL
    religion=int(request.json['adventista']) #'RELIGION'
    retirado=int(request.json['retirado']) #'RETIRADO
    nacionalidad=int(request.json['nacionalidad']) #NACIONALIDAD
    promedio=float(request.json['promedio']) #'PROMEDIO'
    debe=float(request.json['debe']) #debe_total
    credito=float(request.json['credito']) #credito_total
    incremento_nota=float(request.json['inc_nota']) #'incremento_nota',
    des_incremento_nota=float(request.json['des_incnota'])#'desincremento_nota'
    nota_final=float(request.json['nota_final'])#'NOTA_FINAL'
    edad_ultimo=int(request.json['edad_ultimo_ciclo'])#'Edad'
    curs_rep1=int(request.json['curs_rep1'])# curso_repetitivo_1
    curs_rep2=int(request.json['curs_rep2'])# curso_repetitivo_2
    curs_rep3=int(request.json['curs_rep3'])#'curso_repetitivo_3'
    curs_rep4=int(request.json['curs_rep4'])# curso_repetitivo_4
    curs_rep=int(request.json['curs_rep']) #curso_repetitivo
    n_generales =int(request.json['curso_generales'])#numero_generales
    n_especificos  =int(request.json['curso_especificos'])#numero_especificos
    n_especialidad =int(request.json['curso_especialidad'])#numero_especialidad
```

```
    df = pd.DataFrame({"RETIRADO": [retirado],
                       "curso_repetitivo_3": [curs_rep3],
                       "count_array_retiro_desof": [retirado_count],
                       "count_array_trunc": [desertor_count],
                       "CONT_JUNT": [change_sede],
                       "curso_repetitivo_4": [curs_rep4],
                       "NACIONALIDAD": [nacionalidad],
                       "GENERO": [genero],
                       "nro_esc_traslado_ant": [esc_traslado],
                       "nro_contra_ant": [contra_ant],
                       "nro_desaprobados_ant": [desaprobado_ant],
                       "RELIGION": [religion],
                       "NOTA_FINAL": [nota_final],
                       "incremento_nota": [incremento_nota],
                       "desincremento_nota": [des_incremento_nota],
                       "PROMEDIO": [promedio],
                       "curso_repetitivo_1": [curs_rep1],
                       "curso_repetitivo_2": [curs_rep2],
                       "curso_repetitivo": [curs_rep],
                       "numero_generales": [n_generales],
                       "numero_especificos": [n_especificos],
                       "numero_especialidad": [n_especialidad],
                       "NRO_DESAPROBADOS_TOTAL": [desaprobado_total],
                       "CONT_CONTRATOS_TOTAL": [contra_total],
                       "debe_total": [debe],
                       "credito_total": [credito],
                       "Edad": [edad_ultimo]
                       })
    df[numeric_feature_cols] = scale_data(df[numeric_feature_cols],means_ , vars_)
    pred2=clf_from_joblib_.predict(df)
    data={'Tipo de Desertor':str(pred2[0])}
    return jsonify(data)
def my_function(fname):
    array=[]
    codigo=[]
    with open('export_dataframejunto_v11.csv') as csvarchivo:
        entrada = csv.reader(csvarchivo)
        for reg in entrada:
            escuela=str(reg[0])
            means_ , vars_ ,clf_from_joblib_= obtener_standar(escuela)
            genero=int(reg[8]) ##GENERO'
            retirado_count=int(reg[3])#count_array_retiro_desof
            desertor_count=int(reg[4])#count_array_trunc
            change_sede=int(reg[5])#CONT_JUNT
            esc_traslado=int(reg[9])#nro_esc_traslado_ant
            contra_ant=int(reg[10])#nro_contra_ant
            desaprobado_ant=int(reg[11])#nro_desaprobados_ant
```

```
contra_total=int(reg[24])#CONT_CONTRATOS_TOTAL
desaprobado_total=int(reg[23])#NRO_DESAPROBADOS_TOTAL
religion=int(reg[12]) #'RELIGION'
retirado=int(reg[1]) #'RETIRADO'
nacionalidad=int(reg[7]) #NACIONALIDAD
promedio=float(reg[16]) #'PROMEDIO'
debe=float(reg[25]) #debe_total
credito=float(reg[26]) #credito_total
incremento_nota=float(reg[14]) #'incremento_nota',
des_incremento_nota=float(reg[15])#'desincremento_nota',
nota_final=float(reg[13])#'NOTA_FINAL'
edad_ultimo=int(reg[27])#'Edad
curs_rep1=int(reg[17])# curso_repetitivo_1
curs_rep2=int(reg[18])# curso_repetitivo_2
curs_rep3=int(reg[2])#'curso_repetitivo_3'
curs_rep4=int(reg[6])# curso_repetitivo_4
curs_rep=int(reg[19]) #curso_repetitivo
n_generales =int(reg[20])#numero_generales
n_especificos  =int(reg[21])#numero_especificos
n_especialidad =int(reg[22])#numero_especialidad
df = pd.DataFrame({"RETIRADO": [retirado],
        "curso_repetitivo_3": [curs_rep3],
        "count_array_retiro_desof": [retirado_count],
        "count_array_trunc": [desertor_count],
        "CONT_JUNT": [change_sede],
        "curso_repetitivo_4": [curs_rep4],
        "NACIONALIDAD": [nacionalidad],
        "GENERO": [genero],
        "nro_esc_traslado_ant": [esc_traslado],
        "nro_contra_ant": [contra_ant],
        "nro_desaprobados_ant": [desaprobado_ant],
        "RELIGION": [religion],
        "NOTA_FINAL": [nota_final],
        "incremento_nota": [incremento_nota],
        "desincremento_nota": [des_incremento_nota],
        "PROMEDIO": [promedio],
        "curso_repetitivo_1": [curs_rep1],
        "curso_repetitivo_2": [curs_rep2],
        "curso_repetitivo": [curs_rep],
        "numero_generales": [n_generales],
        "numero_especificos": [n_especificos],
        "numero_especialidad": [n_especialidad],
        "NRO_DESAPROBADOS_TOTAL": [desaprobado_total],
        "CONT_CONTRATOS_TOTAL": [contra_total],
        "debe_total": [debe],
        "credito_total": [credito],
        "Edad": [edad_ultimo]
```

```
                })
        df[numeric_feature_cols] = scale_data(df[numeric_feature_cols],means_ , vars_)
        pred2=clf_from_joblib_.predict(df)
        array.append(str(pred2[0]))
        codigo.append(edad_ultimo)
    return array
@app.route("/list_student",methods=["POST"])
def list_student():
    data=[]
    data=my_function(data)
    return jsonify(data)
if __name__ == '__main__':
    app.run(debug=True)
```