UNIVERSIDAD EAFIT

Vigilada Mineducación

Predicción de rotación de empleados usando modelos de aprendizaje automático

Employee turnover prediction using machine learning models

LUIS JAVIER PALACIO MESA

Tesis

Asesor

Biviana Marcela Suárez Sierra

Codirector

Juan Pablo Román Calderón

UNIVERSIDAD EAFIT

ESCUELA DE INGENIERÍAS

MAESTRÍA EN CIENCIAS DE LOS DATOS Y LA ANALÍTICA

MEDELLÍN

2023

Contenidos

Resumen **5**

Introduccción **6**

Planteamiento del problema **7**

Justificación **7**

Objetivos **8**

Objetivo General 8

Objetivos Específicos 8

Consideraciones específicas del reporte **9**

Producto esperado 9

Manejo de datos 9

Consideraciones éticas 9

Marco teórico **9**

Aprendizaje automático y predicción de rotación de empleados. 11

Regresión logística 11

Árboles de decisión 12

Random forest 14

Gradient boosting tree 15

Long short-term memory 18

Técnicas para manejar desbalance de clases 22

Metodología **23**

Breve descripción del conjunto de datos 23

Extracción de datos de empleados 24

Procesamiento de los datos 26

Estructuración de los datos 31

Exploración de los datos 35

Tratamiento de datos atípicos 39

Análisis de dependencias 41

Evaluación 47

Métricas de evaluación 47

Estrategia para la evaluación 48

Resultados **50**
Dummy classifier 52
Random Forest 53
Regresión logística 55
Gradient tree boosting 57
LSTM 58

Conclusiones **61**

Agradecimientos **64**

Índice de figuras

Árbol de decisión para clasificación 13
Ejemplo de *Random forest* en el caso binario 15
Bloque de memoria LSTM (Van Houdt et al. 2020). 19
Ejemplo de extracción de datos de un componente en una página web 25
Ejemplo de la estructura JSON de los datos recolectados usando web scrapping. 26
Organización de los datos para los modelos de regresión logística, *random forest y gradient boosting tree* 32
Distribución del número de roles excluyendo empleados con un único rol 32
Organización de los datos para *LSTM*, usando 3 roles como contexto 34
Distribución del número de roles excluyendo empleados con 3 roles o menos 35
Distribución de frecuencias del número de roles por año de inicio 35
Diagrama de cajas y bigotes de la variable *months in position* 36
Diagrama de cajas y bigotes de la variable *months in company acum* 37
Diagrama de cajas y bigotes de la variable *months experience* 38
Diagrama de frecuencias relativas de la cantidad de empleados por país 38
Gráfico de línea de tasa de desempleo 39
Gráfico de línea del crecimiento real del producto interno bruto 39
Diagrama de cajas y bigotes para las distancias de Mahalanobis 41
Gráfico de pares de las covariables 42
Diagramas de cajas y bigotes por clase 43
Matriz de correlación de Spearman 44
Months in position para predecir el 5 rol variando el número del rol 45
Months in company acum para predecir el 5 rol variando el número del rol 45
Unemployment rate para predecir el 5 rol variando el número del rol 45
Desbalance de clases primera estructura de datos usando 1 roles para predecir el siguiente 46

Desbalance de clases segunda estructura de datos usando cuatro roles para predecir la clase del quinto . 47
Diagrama de precision vs recuperación para *random forest*. 54
Matriz de confusión para *random forest* 54
Importancia de las características para *random forest* usando MDI 55
Diagrama de precision vs recuperación para la regresión logística. 56
Matriz de confusión para regresión logística 57
Diagrama de precision vs recuperación para *gradient tree boosting*. 58
Matriz de confusión para *gradient tree boosting* 58
Diagrama de precision vs recuperación para *LSTM* 60
Matriz de confusión para LSTM 61

Resumen

Las rotaciones de empleados son eventos de gran relevancia dentro de las compañías por múltiples razones. Una de estas razones es que el costo de buscar y contratar nuevos empleados que se ajusten a los puestos disponibles es elevado, por lo que los equipos de recursos humanos tratan de identificar a las personas con intenciones de irse y utilizar esta información para tomar mejores decisiones en beneficio tanto de los empleados como de la empresa. En los últimos años, la academia ha comenzado a aplicar muchos algoritmos de aprendizaje automático para predecir la rotación de empleados, y se ha sugerido a la comunidad científica estudiar nuevos modelos que tengan la capacidad de encontrar patrones con datos que posean una dimensión temporal. El propósito de esta investigación es estudiar modelos para la predicción de la rotación de empleados con los datos disponibles de una red social de empleados y realizar una comparativa del desempeño de técnicas que se han usado usualmente para resolver este problema como Regresión logística, *Random Forest* y *Extreme boosting* frente a técnicas que estén diseñadas para manejar secuencias de datos indexados con el tiempo como *LSTM (Long Short-Term Memory)*.

Introduccción

La rotación de empleados es un fenómeno de suma importancia para las empresas puesto que puede tener grandes efectos como la reducción en sus ganancias, la pérdida del conocimiento corporativo o incrementos en los costos de reclutamiento para contratar nuevo personal. Cada vez se ha venido buscando que recursos humanos y el equipo de gestión de la organización estén preparados para esta situación y puedan evitarla o evaluar si se requiere tener un plan de acción para cuando sus trabajadores tomen esa decisión.

Para tal propósito, nuevos enfoques desde el aprendizaje automático se han venido investigando para la creación de distintas soluciones, una de ellas es la predicción de la rotación de empleados de manera individual y colectiva. Sin embargo, la mayor parte de los estudios existentes no contemplan variables temporales, y, en consecuencia, los expertos del campo sugieren introducir e investigar más frecuentemente el uso de técnicas que exploren la información temporal de los datos, por lo tanto, en este estudio se procurará usar un tipo de red neuronal recurrente conocido por tener la capacidad de trabajar con secuencias de datos indexados con el tiempo pues permite memorizar y retener datos de elementos previos de la secuencia para utilizarlos en elementos posteriores. Finalmente se plantea comparar su desempeño con los demás modelos que se suelen usar para predecir la rotación de empleados.

Para este trabajo se realizará la recolección de datos de la red social de empleados conocida como LinkedIn a través del uso de la técnica de *web scrapping*. Estos datos consisten en las distintas experiencias y roles o posiciones que ha tenido cada empleado durante su carrera incorporando variables como la duración de cada rol. Se realizará una transformación de los datos de esta fuente a un formato inicial de tabla con el que se pueda empezar a hacer limpieza, ingeniería de características y exploración de datos. Posteriormente se organizarán los datos principalmente en dos estructuras, la primera de ellas en un formato usual de tabla para los modelos que no están especialmente diseñados para manejar secuencias de datos, y la segunda será un arreglo multidimensional para formar una secuencia de datos indexados con el tiempo para alimentar fácilmente modelos como las redes neuronales recurrentes. Se plantea el problema como uno de clasificación binaria para seguir la convención que han tenido muchas de las investigaciones de la rotación de empleados a nivel individual. Así pues, se analizará si existe desbalance de clases, y en ese escenario, procurar manejarlo con alguna técnica pertinente, luego se procederá a realizar el entrenamiento, buscar hiperparámetros y evaluar el desempeño de los modelos para, finalmente, efectuar una comparación entre estos y analizar los resultados obtenidos.

Palabras clave: Predicción de rotación de empleados, aprendizaje automático, LSTM, analítica de recursos humanos.

Keywords: Employee turnover prediction, machine learning, LSTM, HR analytics.

Planteamiento del problema

La rotación de empleados, entendida como la salida de trabajadores de una organización por cualquier motivo, es un problema realmente relevante y de gran interés para las compañías puesto que tiene muchas implicaciones notables: primero, se requiere capital humano, recursos económicos y tiempo para llevar a cabo el proceso de reclutamiento para seleccionar nuevos candidatos, por otro lado, la experticia y productividad de los empleados que dejaron la compañía es difícil de reemplazar y requiere tiempo de entrenamiento, todo esto afecta al cliente y la operación normal de la empresa (Cai et al. 2020; Jain et al. 2020; Saradhi & Palshikar, 2011).

Adicionalmente, la rotación de empleados tiene el potencial de afectar negativamente las finanzas de la compañía, y, por consiguiente, los departamentos de recursos humanos y de gestión organizacional pueden tener un firme fundamento para preocuparse por la deserción laboral. Para ilustrarlo mejor, en términos de ganancias promedio en dólares, la investigación que llevó a cabo Hancock et al. (2013) en una empresa financiera, estimó un incremento en las ganancias de más de $24.8 millones si se disminuyera en una desviación estándar la tasa de rotación de empleados.

Analizando otra peculiaridad de la problemática, cuando un empleado tiene intenciones de retirarse, o de hecho, se retira de la empresa, puede incitar a sus compañeros a pensar en retirarse también. Este fenómeno ocurre por el hecho de que los trabajadores tienen la tendencia de compararse con sus compañeros, de ahí que analicen la decisión de irse también, lo que a su vez puede influir en la tasa de rotación de empleados a través del denominado proceso de rotación de empleados contagioso, haciendo la problemática aún más sobresaliente (Bartunek et al. 2008; Felps et al. 2009). Sumado a esto, si su reemplazo no es encontrado prontamente, los empleados que decidan quedarse podrían verse en la obligación de tomar sus tareas, haciendo trabajo extra, lo que los afecta directamente como observó Nandialath et al. (2018).

Justificación

Como consecuencia de las razones expuestas previamente, la rotación de empleados es un tema verdaderamente notable para estudiar por parte de recursos humanos y del equipo de gestión organizacional, sin embargo, existen varios desafíos considerables para su análisis en virtud de la gran cantidad de variables internas y externas que pueden afectar la decisión de los trabajadores de irse. Por este motivo, de acuerdo a Garg et al. (2022) recientes investigaciones han venido explorando enfoques desde la inteligencia artificial y el aprendizaje automático para contribuir a la toma de decisiones basadas en datos, asimismo, desde la academia se hace un llamado a que los expertos del área de recursos humanos y los científicos de datos se unan. Con este trabajo conjunto se procura construir nuevas estrategias para tomar decisiones proactivas y así tener una mayor preparación para lograr mejores decisiones.

Específicamente, la predicción de la rotación de empleados, puede seguir tal propósito y facilitar a la

gestión de la organización y a recursos humanos la anticipación y elaboración de planes de acción para reducir posibles problemas que puedan emerger como resultado de esta situación (Cai et al. 2020). Con miras a llevar a cabo la predicción, diferentes técnicas de aprendizaje automático han sido usadas con variables del empleado (como satisfacción en el empleo, salario, edad, género, tiempo promedio de trabajo), de la organización (como pólizas de la compañía, percepción general del ambiente laboral, número de empleados) y muchas más (Garg et al. 2022).

A pesar de ello, muchas de estas variables no consideran datos históricos, dado que los investigadores usualmente toman en cuenta el último registro disponible en un tiempo específico o usan un estadístico para resumir o agregar la información disponible de ciertas variables (Rubenstein et al. 2017), por lo que es interesante iniciar investigaciones para comprobar si existe una pérdida de información importante que pueda afectar el desempeño de los modelos construidos al no usar secuencias de datos indexados con el tiempo.

Objetivos

Objetivo General

El propósito de esta investigación es estudiar modelos para la predicción de la rotación de empleados con los datos disponibles de una red social de empleados y realizar una comparativa del desempeño de técnicas que se han usado usualmente para resolver este problema como Regresión logística, *Random Forest* y *Extreme boosting* frente a técnicas que estén diseñadas para manejar secuencias de datos indexadas con el tiempo como *LSTM (Long Short-Term Memory)*.

Objetivos Específicos

- Extraer y realizar ingeniería de características de datos de la red social de los empleados para crear un conjunto de datos apropiado para el entrenamiento de modelos de aprendizaje automático.
- Estructurar los datos de la red social de los empleados para que sean adecuados para su uso en redes neuronales recurrentes como *LSTM (Long Short-Term Memory)*.
- Comparar el desempeño de las redes neuronales recurrentes LSTM para pronosticar rotación de empleados, frente a otros modelos propuestos en la literatura
- Realizar un informe escrito reportando las ventajas y desventajas de LSTM con algunas recomendaciones para futuras consideraciones.

Consideraciones específicas del reporte

Producto esperado

Informe de proyecto de graduación con la comparación de las técnicas para la rotación de empleados.

Manejo de datos

Parte de la información es recopilada de una red social de empleados públicos y será utilizada únicamente con fines académicos. Únicamente el alumno, el director y codirector tendrán acceso a los datos originales y transformados. Al final de la investigación, estos datos serán completamente destruidos y eliminados. Los resultados y conclusiones de este análisis no incluirán ningún dato personal por lo que es imposible identificar a un empleado en concreto.

Otras fuentes son de uso completamente gratuito, como el PIB o la tasa de desempleo de los países que está disponible en los datos del fondo monetario internacional (FMI)

Consideraciones éticas

Esta investigación podría ayudar a las organizaciones a construir mejores modelos para comprender a sus empleados y crear mejores estrategias para retener el talento mejorando sus condiciones laborales.

En cuanto al proceso de recopilación de estos datos, es legal de acuerdo con las últimas determinaciones de la corte de los EE. UU. que estableció que la adquisición y el procesamiento de datos de redes sociales de empleados públicos no viola el Código de Abuse Act (CFAA) (Bro, 2022).

Marco teórico

En los últimos 100 años se han realizado más de 1500 estudios relacionados con la rotación de empleados donde la teoría se ha desarrollado considerablemente, sin embargo, todavía hay muchas preguntas y grandes oportunidades para estudios futuros puesto que, aunque se han implementado una gran cantidad de estrategias y enfoques para obtener una varianza explicada alta de la tasa de rotación de empleados, aún se requiere más esfuerzos para mejorarla (Holtom et al. 2008). Por ejemplo, Holtom et al. (2008), Hom et al. (2017) y Rubenstein et al. (2017) recomiendan contemplar el tiempo en futuras investigaciones, es decir, considerar variables en diferentes momentos para que la secuencia brinde más información.

El consejo de incluir secuencias de datos indexadas con el tiempo en la rotación de empleados se debe principalmente al hecho de que la mayoría de los estudios han estado utilizando una práctica de investigación estándar, donde los académicos habitualmente, mediante el uso de una encuesta, miden las variables en un solo

período y luego en un tiempo posterior, meses después, si los empleados abandonaron la empresa, no obstante, este enfoque tiene una baja varianza explicada ya que lleva a ignorar los cambios y las dinámicas que ocurren durante el tiempo del fenómeno de rotación de empleados (Lee et al. 2017; Rubenstein et al. 2017). Adicionalmente, el peso de las variables puede variar con el tiempo, por ejemplo, Holtom et al. (2013) encontró que el efecto de la satisfacción del empleado es mayor durante los primeros tres años del empleado en la organización.

En cuanto a los atributos y según el meta-análisis realizado por Rubenstein et al. 2017 algunas de las variables más importantes para predecir la rotación en el aspecto individual son *tenure*, (es decir la antigüedad que el empleado tiene en una empresa), la edad, el número de hijos que informó el empleado, mientras que algunos de los menos relevantes son el sexo, el origen étnico o la raza, y las capacidades o habilidades, aunque algunos factores pueden actuar como moderadores de los efectos de otras variables con la rotación de empleados, por ejemplo, los impactos de la antigüedad en la rotación de empleados se vuelven más negativos cuando la organización tiene más empleados de alto nivel.

Igualmente, las variables que miden las alternativas de empleo son apropiadas en el análisis de la rotación de empleados debido a que permiten incorporar factores externos tales como la tasa de desempleo del país, en este caso, es una variable de contexto que representa las oportunidades de empleo disponibles en el mercado y también puede cambiar el efecto de otros atributos, para ilustrarlo, se ha evidenciado que el nivel máximo de educación en los empleados tiene efectos más fuertes en la tasa de rotación de empleados cuando la tasa de desempleo es alta (Rubenstein et al. 2017).

Otra cuestión bastante relevante entre algunos investigadores es la contemplación de la ubicación de los estudios de rotación de personal. La mayoría coincide en que una parte apreciable de la investigación sobre rotación de personal proviene de Estados Unidos con un bajo porcentaje de análisis en otros países (Hancock et al. 2013; Holtom et al. 2008; Rubenstein et al. 2017). Además, como aclara Holtom et al. 2008; Rubenstein et al. 2017 las tasas de rotación también varían dependiendo de la ocupación, más aún, en una misma industria pueden variar mucho. Sin embargo, la mayoría de los artículos norteamericanos sobre este tema son "de civiles en organizaciones privadas que realizan trabajos administrativos u hospitalarios"(Rubenstein et al. 2017). Por lo tanto, es necesario realizar más investigaciones en otras naciones, culturas e industrias, para descubrir qué factores pueden apoyar los conceptos desarrollados.

En particular, en esta investigación no consideraremos ninguna diferencia entre rotación voluntaria e involuntaria, ni evaluaremos el desempeño organizacional. Nos centraremos en predecir la rotación de empleados como una variable total que incluye tanto la rotación voluntaria como la involuntaria. Además, nos centraremos en el análisis de la rotación de empleados a nivel individual utilizando la información disponible en el sitio web de la red social donde extraeremos el conjunto de datos. Finalmente, dadas las recomendaciones para realizar estudios en otras culturas, extraeremos datos específicamente de América Latina para comparar nuestros resultados con la teoría desarrollada en América del Norte principalmente.

Aprendizaje automático y predicción de rotación de empleados.

De acuerdo con (Garg et al. 2022), el principal objetivo del aprendizaje automático en las últimas investigaciones relacionadas con la rotación de empleados es predecirla mediante el uso de variables personales y relacionadas con el trabajo de los empleados. En resumen, algunos académicos aplicaron algoritmos para predecir la rotación de empleados, como máquinas de soporte vectorial, árboles de decisión, *Naïve Bayes* y algoritmos de ensamble como el bosque aleatorio (Jain et al. 2020; Saradhi & Palshikar, 2011).

También se ha empleado la regresión logística en el escenario de la predicción de la rotación de empleados, por ejemplo, Rombaut y Guerry (2018) justificaron la elección de esta técnica en su investigación dado su amplio uso en problemas similares como la predicción de la fuga de clientes, además, resaltaron la simplicidad de este modelo y su gran interpretabilidad para explicar los factores influyentes para realizar la predicción. En su estudio utilizan un conjunto de datos con la información anual de cada empleado que el departamento de recursos humanos recolectó, y escogen como variable respuesta para cada individuo y para cada año si se retiró de la compañía o no.

Este enfoque les permite diseñar un problema de clasificación binaria, y además consiguen obtener las probabilidades que la regresión logística asigna a cada empleado sobre el evento de irse de la empresa, asimismo, sustentan este planteamiento indicando que las probabilidades pueden ser valiosas para el equipo de recursos humanos con la finalidad de planificar que acciones tomar en caso de querer retenerlo. Su investigación dio resultados muy positivos, es así que con el propósito de seguir una estrategia similar que se ha analizado desde la literatura de la rotación de empleados, en este proyecto se buscará realizar la predicción de si un empleado se irá de la empresa en menos de un año, por lo que se tratará como problema de clasificación binaria.

Sumado a los métodos anteriores, Cai et al. (2020) también usaron *Gradient Boost Trees* y *Extreme Gradient Boost*. En tal estudio trabajaron con características básicas tanto de empleados como de organizaciones, adicionalmente, lograron obtener nuevas características de la relación histórica entre empleados y empresas que está presente en las redes sociales con una nueva técnica que llamaron *Dynamic Bipartite Graph Embedding* basada en el modelado de las experiencias de los empleados en un gráfico bipartito y obteniendo una representación de baja dimensión de la red que se combinó con las características básicas para entrenar los algoritmos de aprendizaje automático y hacer la predicción.

Regresión logística

Siguiendo de ejemplo la formulación que presenta Friedman et al. (2010), sea $\boldsymbol{Y} \in \{\mathbf{0}, \mathbf{1}\}$ la variable aleatoria discreta respuesta en el caso binario y $\boldsymbol{X} = [\boldsymbol{X_1}\ \boldsymbol{X_2}\ \ldots\ \boldsymbol{X_d}]$ el vector de variables aleatorias que contiene las variables explicativas (covariables), entonces, el algoritmo de regresión logística usa una combinación lineal $\beta_0 + X^T\beta$ de las covariables que se introducen en una función conocida como sigmoide $\sigma(x) = (1/1 + e^{-x})$ (tiene la propiedad de acotar sus resultados en el intervalo (0, 1) para cualquier valor de entrada), para estimar la

probabilidad condicional de la clase deseada:

$$\Pr(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + X^T\beta)}} \tag{1}$$

Donde $\boldsymbol{\beta} = [\boldsymbol{\beta_1}\ \boldsymbol{\beta_2}\ \ldots\ \boldsymbol{\beta_d}]$ son los coeficientes asociadas con las covariables y β_0 es un parámetro que identifica la intersección, se puede notar que cuando no hay efectos de ninguna de las covariables (sus valores se hacen 0) este es el valor esperado que se introduce en la función sigmoide. Estos coeficientes se pueden entender como unos pesos que van a determinar la influencia de cada variable en el modelo final.

El modelo de regresión logística se optimiza encontrando los parámetros $\boldsymbol{\beta}$ y $\boldsymbol{\beta_0}$ que maximizan la log verosimilitud (condicional), asimismo, por lo general, se agrega un componente adicional que evita que en la optimización se prefieran coeficientes altos, por lo cual, se reduce el sobreajuste de los datos. Una forma de conseguir esto es con una suma de los valores absolutos de los parámetros o una suma de parámetros al cuadrado agregado a la función de pérdida de modo que la optimización busque reducir el tamaño de estos valores. Además, esa sumatoria se suele multiplicar por un hiperparámetro $\boldsymbol{\lambda}$ que permite ajustar su influencia en la función de pérdida. Si se tienen $\boldsymbol{N}$ observaciones y se usa el subíndice $\boldsymbol{i = 1, 2 \ldots N}$ en $\boldsymbol{X_i, y_i}$ para denotar la i-ésima observación de las covariables y de la variable respuesta, entonces la log verosimilitud se puede observar en ecuación 2 y la función a maximizar en el caso de usar la sumatoria de los valores absolutos de los parámetros β se puede apreciar en la ecuación 3:

$$\ell(\beta_0, \beta) = \sum_{i=1}^{N} y_i \log[\Pr(y = 1|X_i)] + (1 - y_i) \log[\Pr(y = 0|X_i)] \tag{2}$$

$$\max_{\beta_0, \beta} \left\{ \ell(\beta_0, \beta) + \lambda \sum_{i=1}^{p} |\beta_i| \right\} \tag{3}$$

Árboles de decisión

Los árboles de decisión son una familia de algoritmos que buscan dividir los datos en subconjuntos disjuntos a través de particiones hechas de manera recursiva con reglas creadas a partir de las distintas variables de entrada $\boldsymbol{X}$ (Loh, 2011). Por la naturaleza del algoritmo es fácil visualizar y comprender tales reglas que se van aprendiendo para realizar las partición. En la figura 1 se puede observar un árbol de decisión para clasificar dos clases, la clase representada por un triángulo y otra clase representada por un cuadrado. Es importante resaltar que aunque en este ejemplo se tienen dos clases el árbol de decisión puede usarse para clasificar múltiples clases. El árbol de decisión inicia con un nodo llamado raíz y está asociado con la regla $\boldsymbol{X_1 < 20}$ que usa como base para crear dos conjuntos disjuntos, la del nodo izquierdo y derecho. En este ejemplo las formas geométricas (triángulo y cuadrado) de los nodos terminales (es decir, nodos que ya no tiene más divisiones) representan las

clases mayoritarias de las observaciones asignadas a ese nodo. Por ejemplo, se puede notar que el primer nodo de la izquierda (de arriba hacia abajo) que tiene forma de triángulo es un nodo terminal y se le asigna la clase que sea mayoritaria en esa partición. Siguiendo con la ilustración, en contraste, el nodo a su derecha asociado con la regla de decisión $\boldsymbol{X_2 = 1}$ es un nodo intermedio y tiene dos nodos hijo terminales, en el nodo de la izquierda se tiene una mayoría observaciones de la clase cuadrada y en el nodo de la derecha hay una mayoría de observaciones con la clase triangular. A pesar de ser un algoritmo tan flexible e interpretable, tiene serios problemas con la generalización de las predicciones para nuevos datos ya que intenta adaptarse perfectamente a los datos de entrenamiento

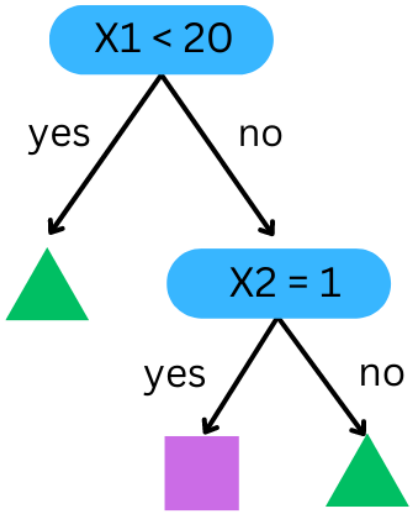

Figura 1: Árbol de decisión para clasificación

Para ilustrar el proceso de aprendizaje de este algoritmo, se puede analizar *Classification and regression tree* (CART) puesto que es uno de los algoritmos más conocidos para la construcción de los árboles de decisión. Concretamente para un problema de clasificación tiene como objetivo crear un árbol binario que minimice una función conocida como impureza, que puede interpretarse como una métrica que mide qué tan mezcladas son las clases de las observaciones en un nodo determinado (Rutkowski et al. 2014). Específicamente, se suele usar el índice Gini. Si su valor es igual a 0, significa que ese nodo solo tiene una clase asociada, y por tanto, hay máxima pureza, mientras que a mayor valor las clases están más mezcladas. En la ecuación 4 se observa la función Gini:

$$Gini(S_i) = 1 - \sum_{k=1}^{n} (p_{k,i})^2 \tag{4}$$

Donde $\boldsymbol{S_i}$ es cualquier subconjunto del conjunto de datos inicial asociado con el nodo $\boldsymbol{i}$, $\boldsymbol{k}$ representa la k-ésima clase si se tienen n de ellas, y $\boldsymbol{p_{k,i}}$ denota la proporción de observaciones con la clase k en el subconjunto $\boldsymbol{S_i}$. Se puede notar que si un nodo solo contiene instancias de una sola clase, entonces el indice Gini es 0, por lo que ese nodo es completamente puro. Adicionalmente se define como Gini ponderado una función qué mide la pureza del nodo izquierdo y derecho teniendo en cuenta la cantidad de observaciones en los dos subconjuntos. Denotando $\boldsymbol{S_{L,i}}$ y $\boldsymbol{S_{R,i}}$ como los subconjuntos del nodo izquierdo y derecho, y dado que $\boldsymbol{S_i = S_{L,i}\, U\, S_{R,i}}$

entonces el Gini ponderado se define como:

$$GiniWeighted(S_i, S_{L,i}) = \frac{|S_{L,i}|}{|S_i|} Gini(S_{L,i}) + \frac{|\overline{S_{L,i}}|}{|S_i|} Gini(\overline{S_{L,i}}) \tag{5}$$

El algoritmo CART busca de manera recursiva la mejor característica y regla de decisión que maximice la ganancia de Gini. Como se puede revisar en la ecuación 6, la ganancia de Gini se define como la diferencia entre el Gini del nodo i y el correspondiente Gini ponderado del nodo izquierdo y derecho.

$$g(S_i, S_{L,i}) = Gini(S_i) - GiniWeighted(S_i, S_{L,i}) \tag{6}$$

Un nodo dejará de dividirse cuando se llega a una condición de parada, por ejemplo, si hay un número máximo de nodos terminales establecido previamente, o si hay un valor mínimo requerido de ganancia Gini.

Random forest

Random forest es un técnica que generalmente se basa en una versión mejorada del método conocido como *bagging* (*bootstrap aggregating*), este es un método de ensamble (técnicas que permite combinar varios predictores) en el que se entrenan los modelos paralelamente mediante el uso de diferentes subconjuntos muestreados con reemplazo de los datos de entrenamiento originales. Específicamente, se define un *Random forest* como una colección de árboles de decisión entrenados al incluir un procedimiento aleatorio en cada uno de ellos. Una de las propiedades deseables en el ensamble, es que exista una baja correlación entre cada uno de los árboles que lo componen debido a que aumenta la generalización del modelo, (reduce la varianza). Una estrategia para lograr una baja correlación consiste en no considerar todas las características para ejecutar la división en cada nodo de los árboles, en lugar de ello se usa una selección aleatoria de todas las características o una combinación lineal de ellas (Breiman, 2001).

En la figura 2, se puede observar que los datos de entrenamiento se usan para crear ***n*** muestras con reemplazo, una por cada árbol que se va a construir. Con cada una de las muestras se entrena un árbol de decisión de manera independiente, así que la estructura de cada árbol puede ser distinta, por ejemplo, el primer árbol y el último tienen 3 nodos terminales, mientras que el árbol intermedio tiene 4 nodos terminales, de hecho, aunque el primer árbol y el último árbol tienen la misma cantidad de nodos terminales, las clases mayoritarias de cada uno son diferentes entre sí, se aprecia que el primer árbol tiene dos triángulos representando que en esos nodos la primera clase es la clase mayoritaria, en tanto que el último árbol tiene solo un triángulo y dos cuadrados representando que la segunda clase es mayoritaria en esos dos nodos, en definitiva, son modelos distintos.

Cuando se han entrenado todos los árboles de decisión con algún algoritmo como CART, ya es posible realizar predicciones para nuevas instancia de datos. Una nueva observación es clasificada por cada uno de los árboles al evaluar sus reglas de decisión, así pues, cada árbol predice la clase mayoritaria del nodo terminal correspondiente en el que la observación fue asignada luego de recorrer el árbol. Asumiendo que solo se entrenaron

3 árboles de decisión y siguiendo con la ilustración, el primer y último árbol predijeron la clase 1, mientras que el árbol intermedio predijo la clase 2, puesto que más modelos escogieron la clase 1, esa es la clase final para la instancia de datos que se está clasificando. En concreto, la clase predicha para una instancia corresponde a la clase mayoritaria seleccionada por la secuencia de modelos construidos previamente (Breiman, 1996).

Una de las bondades que posee este algoritmo y que favorece su uso para el problema de clasificación de rotación de empleados, es que no se limita a predecir la clase, añadido a eso se puede acceder a las probabilidades de que el empleado salga de esa empresa. Estas probabilidades se pueden conseguir al tomar la cantidad de árboles que clasificaron cierta clase, dividido, la cantidad de árboles en total. Una pequeña desventaja es que al aumentar la complejidad del modelo en comparación con un único árbol de decisión, ya no es tan sencillo interpretarlo, no obstante, sin duda alguna, tiene mecanismos sólidos y bien establecidos para responder al equipo de recursos humanos, cuáles son las variables más importantes e influyentes para realizar la clasificación, por ejemplo, midiendo la ganancia Gini para cada variable en cada árbol y promediando los valores, aquellas variables con mayor ganancia Gini promedio serán más relevantes puesto que son aquellas que permiten discriminar mejor las diferentes clases.

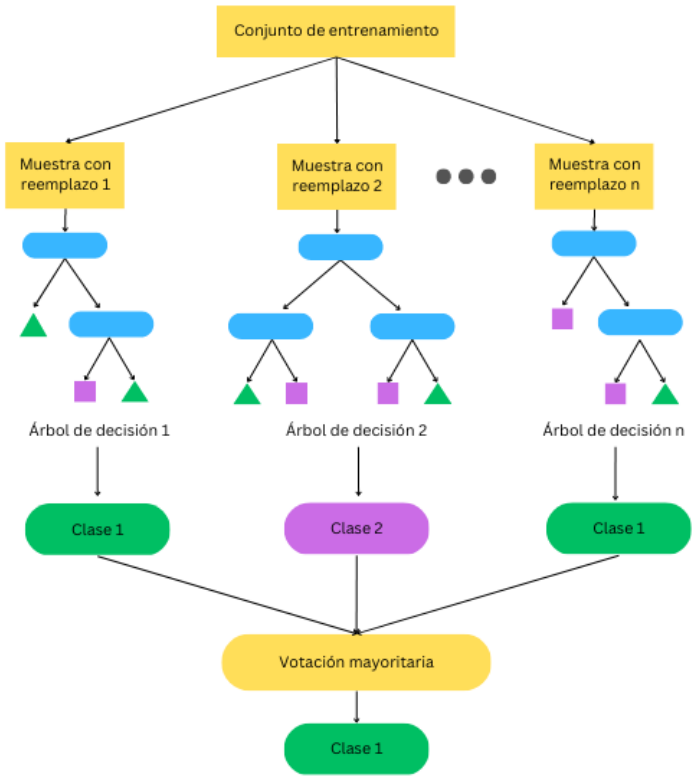

Figura 2: Ejemplo de *Random forest* en el caso binario

Gradient boosting tree

Gradient boosting tree también es un método de ensamble (técnicas que permite combinar varios predictores) muy popular con un proceso de optimización iterativo en el que cada predictor se construye a partir de los errores hechos por predictores anteriores. Esta técnica es un caso especial de *Gradient boosting machines* en el que

los predictores son árboles de decisión, aunque fue planteado para permitir usar otro tipo de modelos. Siguiendo la formulación que definió Friedman (2001) y describiendo el procedimiento para un problema de clasificación binaria, el algoritmo 1 requiere como entrada un conjunto de datos de entrenamiento $\boldsymbol{\{(x_i, y_i)_{i=1}^N\}}$ dónde i es el iterador de la i-ésima observación de $\boldsymbol{N}$, $\boldsymbol{x_i}$ representa un vector de variables aleatorias para la observación $\boldsymbol{i}$, mientras $\boldsymbol{y_i \in \{0, 1\}}$ representa la clase real asociada a esa observación. Asimismo, el algoritmo requiere definir una función de pérdida $\boldsymbol{L(y_i, F(x))}$ que permita medir qué tan buena es una predicción hecha con respecto al valor real.

Algoritmo 1 Gradient boosting tree

Require: Datos de entrada $(x_i, y_i)_{i=1}^N$, función de pérdida $L(y_i, F(x))$

1: $F_0(x) = arg\,min_\gamma \sum_{i=1}^N L(y_i, \gamma)$
2: **for** $m = 1$ to M **do**
3: $\widetilde{y}_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial (F(X_i))}\right]_{F(x)=F_{m-1}(x)},\ for\ i = 1 \dots N$
4: Entrenar un árbol de decisión para predecir los pseudo-residuales $\widetilde{y}_{im}$
5: $R_{jm} = \{x \mid x \in$ j-ésimo nodo terminal del árbol m$\},\ for\ j = 1 \dots J_m$
6: $\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L(y_i, \gamma + F_{m-1}(x_i)),\ for\ j = 1 \dots J_m$
7: $F_m(x) = F_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$
8: **end for**
9: **return** $F_M(x)$

Una función de pérdida muy común es el negativo de la log verosimilitud, inicialmente esta función está definida en términos de probabilidades como se puede observar en la ecuación 8, no obstante, se suele aplicar una transformación para que la función quede en términos de *log(odds)*. El *log(odds)* se puede definir como el logaritmo de la probabilidad de que ocurra el evento, dividido por la probabilidad de que no ocurra, es decir, $log(odds) = log[p/(1-p)]$. Esta transformación de la función de pérdida se utiliza porque cada función F(x) que se obtendrá con el algoritmo, al evaluarse dará como resultado un valor que estará en términos de *log(odds)*. Por lo tanto, usando adecuadamente la relación que existe entre log(odds) y la probabilidad y despejando para la probabilidad $\boldsymbol{p}$ como se muestra en la ecuación 7 se puede partir de la ecuación 8 para llegar a la ecuación 9 que será la que se utilizará en el algoritmo 1.

$$p = \frac{odds}{1 + odds} = \frac{e^{log(odds)}}{1 + e^{log(odds)}} \tag{7}$$

$$L(y, p) = -\left[y \log(p) + (1 - y) \log(1 - p)\right] \tag{8}$$

$$L(y, log(odds)) = -ylog(odds) + log(1 + e^{log(odds)}) \tag{9}$$

La primera instrucción del algoritmo pretende obtener una función inicial $\boldsymbol{F_0}$ que se usará como estima-

ción inicial para clasificar las observaciones. Esta función inicial se define como la constante γ que minimiza la sumatoria de la función de pérdida para cada observación $\boldsymbol{i}$. Se puede hallar a través de la derivada parcial de la función de pérdida con respecto a γ, igualando a 0 y resolver para γ. Sin embargo, en algunas implementaciones simplemente se pone un valor constante independientemente de cuál sea la función de pérdida, es el caso de XGboost del que se hablará posteriormente.

Una vez inicializado $\boldsymbol{F_0}$, en la instrucción número 2 del algoritmo se define el comienzo de una iteración o un ciclo ***for*** usando la letra $\boldsymbol{m}$ para denotar la m-ésima iteración de $\boldsymbol{M}$ que se van a realizar. Son $\boldsymbol{M}$ iteraciones, porque esta es la cantidad de árboles de decisión que se van a entrenar y que se van a ir combinando para obtener el predictor final. En la instrucción número 3 , ya dentro del ciclo, para el árbol $\boldsymbol{m}$, se procede a encontrar unos valores nombrados como pseudo-residuales y denotados con $\widetilde{\boldsymbol{y}}_{\boldsymbol{im}}$ que se hallan con el negativo del gradiente de la función de pérdida con respecto a $\boldsymbol{F_{m-1}}$ para cada observación $\boldsymbol{i}$. Con la función de pérdida que se escogió y que se observa en la ecuación 9, se resuelve la derivada parcial de la instrucción 3, y se deduce la ecuación 10, es decir, los pseudo-residuales resultan ser simplemente la diferencia entre $\boldsymbol{y_i}$ y $\boldsymbol{p_i}$ que denota la probabilidad asignada a la observación $\boldsymbol{i}$. Esta probabilidad se puede obtener a partir del predictor $\boldsymbol{F_{m-1}}$ cuya salida está en términos de *log odds*. Por tanto, usando de nuevo la relación entre probabilidad y *log odds* de la ecuación 7, se llega a la ecuación 11. En definitiva, los pseudo-residuales proporcionan un método para tener en cuenta los errores del modelo anterior y mejorarlos en la siguiente iteración.

$$\widetilde{y}_{im} = y_i - p_i \tag{10}$$

$$\widetilde{y}_{im} = y_i - \frac{e^{F_{m-1}(x_i)}}{1 + e^{F_{m-1}(x_i)}} \tag{11}$$

Seguidamente en la instrucción 4 se entrena un árbol de decisión usando los pseudo-residuales como variable respuesta, es decir, se entrena con el conjunto de datos $\boldsymbol{\{(x_i, \widetilde{y}_{im})_{i=1}^N\}}$. Este árbol de decisión define en la instrucción 5 un conjunto de regiones disjuntas $\boldsymbol{R_{jm}}$ con las observaciones que se asignaron a cada nodo terminal $\boldsymbol{j}$ de $\boldsymbol{J_m}$ nodos terminales que se construyeron en ese árbol $\boldsymbol{m}$. Posteriormente, por cada nodo terminal $\boldsymbol{j}$ del árbol actual $\boldsymbol{m}$, en la instrucción 6 se procura encontrar un valor de salida $\boldsymbol{\gamma_{jm}}$ que ayude a minimizar la función de pérdida de la agregación de $\boldsymbol{\gamma_{jm}}$ con el modelo global acumulado hasta el momento $\boldsymbol{F_{m-1}(x)}$. Precisamente, como se plantea en la ecuación 12 ese valor $\boldsymbol{\gamma_{jm}}$ es igual a la sumatoria de los pseudo-residuales del nodo terminal $\boldsymbol{j}$, dividido la sumatoria del producto de la probabilidad $\boldsymbol{p_i}$ frente a su complemento $\boldsymbol{1 - p_i}$. Las probabilidades $\boldsymbol{p_i}$ se obtienen al evaluar la función $\boldsymbol{F_{m-1}}$ con las observaciones $\boldsymbol{x_i}$ pertenecientes a la región de ese nodo terminal $\boldsymbol{j}$ (a través de la evaluación de las reglas del árbol de decisión) y, convirtiendo ese valor que está en términos de *log odds* a probabilidad a través de la ecuación 7.

$$\gamma_{jm} = \frac{\sum_{x_i \in R_{jm}} \widetilde{y}_{im}}{\sum_{x_i \in R_{jm}} p_i(1 - p_i)} \tag{12}$$

En la instrucción 7 se define el modelo $\boldsymbol{F_m(x)}$ como la suma del modelo anterior $\boldsymbol{F_{m-1}(x)}$ más el $\boldsymbol{\gamma_{jm}}$ que se consigue del árbol actual al evaluar las reglas de decisión para una observación $\boldsymbol{x}$, y, dependiendo de la región asignada, se llegará al valor $\boldsymbol{\gamma_{jm}}$ deseado. En esa instrucción, lo anterior se representa matemáticamente a través de una función indicador para sumar $\boldsymbol{\gamma_{jm}}$ si x pertenece a la región $\boldsymbol{R_{jm}}$, y puesto que son regiones disjuntas, tal condición solo será verdadera para uno de los $\boldsymbol{x}$. A continuación se repite el proceso desde la instrucción 3 hasta construir $\boldsymbol{M}$ árboles de decisión o hasta que se alcance algún criterio de parada. Finalmente la salida del algoritmo es $\boldsymbol{F_M(X)}$, un modelo global que será la combinación de todos los anteriores.

Una implementación notable de esta técnica, que usa árboles de decisión como aprendices débiles es XGBoost, como prueba Chen y Guestrin, 2016, es significativamente rápido, ya que está optimizado para ejecutarse en multiproceso o incluso computación distribuída, por lo tanto, escala excepcionalmente para conjuntos de datos grandes y diversos, logrando la mayoría de las veces excelentes métricas de rendimiento.

Long short-term memory

Antes de definir esta técnica se puede introducir dos conceptos relevantes que facilitarán su comprensión, la primera es el aprendizaje profundo y el segundo las redes neuronales recurrentes. El aprendizaje profundo es un conjunto de técnicas de aprendizaje automático que han sido capaces de resolver muchos problemas que son difíciles de resolver con otros métodos más tradicionales, ya que son más flexibles y requieren menos ingeniería de características para funcionar correctamente, además, pueden aprender funciones no lineales complejas combinando la salida de unidades simples llamadas neuronas (LeCun et al. 2015). Una de las arquitecturas de aprendizaje profundo más conocidas es el perceptrón multicapa, que incorpora el concepto de usar múltiples capas de neuronas (cuantas más capas, más profunda será la arquitectura), cada una de ellas con múltiples conexiones a otras neuronas (Lara-Benıtez et al. 2021). Estas conexiones están asociadas con algunos parámetros entrenables que, por lo general, se ajustan implementando un algoritmo de optimización de descenso del gradiente en el que se calcula un vector de gradiente opuesto para indicar la dirección que reduce la función de pérdida del modelo, como explicó LeCun et al. (2015).

Las arquitecturas de perceptrón multicapa no suelen ser adecuadas para problemas de secuencias de datos indexadas con el tiempo ya que no tienen un buen mecanismo para incluir la información temporal a lo largo de todas las capas, puesto que requieren juntar las entradas temporales en un vector espacial de longitud fija lo que hace que sea más complicado emplearlas con datos de secuencias con distintas longitudes, además, estas redes tienen problemas al diferenciar posiciones temporales absolutas con posiciones temporales relativas, por ejemplo, si los perceptrón multicapa tratan con secuencias de datos que se desplazan, no podrán encontrar una relación con las secuencias no desplazadas debido a que la representación que utilizan está estrechamente ligada al concepto de semejanza en un espacio vectorial (Elman, 1990).

En consecuencia, las redes neuronales recurrentes son técnicas del aprendizaje profundo que se diseñan especialmente para manejar datos secuenciales de manera adecuada, evitando los inconvenientes del perceptrón

multicapa al procesar con una neurona un solo elemento de entrada de la secuencia y alimentar esta información a la misma neurona para el procesamiento del siguiente elemento en la entrada (LeCun et al. 2015), esto permite usar efectivamente la información pasada en los siguientes elementos de la secuencia. Por esta razón se dice que LSTM posee una memoria a corto y largo plazo, puesto que puede recordar un dato de la secuencia y preservarlo en un largo o corto periodo de tiempo.

Long short-term memory (LSTM) es un modelo de aprendizaje profundo, específicamente, es un tipo red neuronal recurrente que es capaz de aprender dependencias a largo plazo y también usar información de eventos recientes en una secuencia (Van Houdt et al. 2020). Aunque existen muchas variaciones a los componentes y conexiones de estas redes, en la figura 3 se puede apreciar la estructura de un bloque conocido como *peephole LSTM*. Un bloque de LSTM procesa un elemento de una secuencia de datos, seguidamente decide que información enviarse a sí mismo como contexto al procesar el siguiente elemento de la secuencia hasta que toda haya sido analizada. Está compuesto por una celda de memoria denotada por la letra $\boldsymbol{c}$ que guarda la información histórica de eventos anteriores, y el flujo de información se da a través de tres componentes principales conocidos como compuertas: compuerta de entrada *(input gate)*, compuerta del olvido *(forget gate)* y compuerta de salida *(output gate)*. La compuerta de entrada controla cuándo agregar nueva información a la memoria, la compuerta de olvido es responsable de eliminar o restablecer la información en la memoria y la compuerta de salida regula cuándo leer de la memoria y luego enviar a los siguientes eventos en la secuencia. Adicionalmente, en la figura 3 se puede distinguir en linea punteada unas conexiones denominadas *peephole* que van desde la celda de memoria a cada una de las compuertas, con la finalidad de que puedan conocer el estado de la memoria a largo plazo incluso aunque la información de la salida anterior no haya sido enviada debido a que la compuerta de salida fue cerrada (Gers et al. 2002).

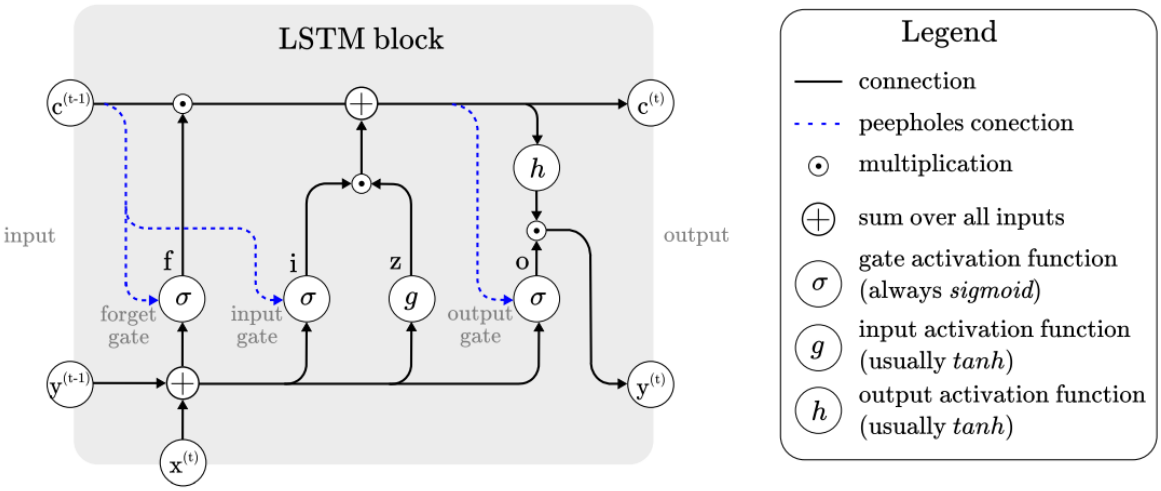

Figura 3: Bloque de memoria LSTM (Van Houdt et al. 2020).

Usando como base la notación presentada por Van Houdt et al. (2020) sea $\boldsymbol{t}$ un súper índice para indicar una posición en una secuencia con $\boldsymbol{t} \in [\boldsymbol{1}, \boldsymbol{T_x}]$ donde $\boldsymbol{T_x}$ es el tamaño de la secuencia de datos de entrada. Entonces, $\boldsymbol{x^{(t)}}$ es un vector columna de $\boldsymbol{d}$ variables aleatorias, $\boldsymbol{c^{(t)}}$ y $\boldsymbol{y^{(t)}}$ son la celda de memoria y la salida del bloque

LSTM para el elemento $\boldsymbol{t}$ de la secuencia, y ambos son vectores columna de dimensión $\boldsymbol{n_h \times 1}$, dónde $\boldsymbol{n_h}$ es la cantidad de unidades de procesamiento o neuronas que tendrá el bloque LSTM. Todas las compuertas tienen un conjunto de parámetros entrenables en las conexiones que hay entre los diferentes componentes, en consecuencia, las matrices denotadas por $\boldsymbol{W}$ se refieren a los parámetros asociados con el vector $\boldsymbol{x^{(t)}}$. Estas matrices tendrán una dimensión $\boldsymbol{n_h \times d}$, es decir, por cada neurona se realizará una combinación lineal de las variables aleatorias. Asimismo, las matrices $\boldsymbol{R}$ están asociadas con la salida del tiempo anterior $\boldsymbol{y^{(t-1)}}$, y tienen una dimensión $\boldsymbol{n_h \times n_h}$. Los vectores columna $\boldsymbol{p}$ de dimensión $\boldsymbol{n_h \times 1}$ contienen los parámetros de las conexiones *peepholes* y están asociadas con el estado de la memoria anterior $\boldsymbol{c^{(t-1)}}$ en todos los componente excepto en el de la compuerta de salida, dónde está asociada a la memoria actual $\boldsymbol{c^{(t)}}$. Finalmente, hay vectores de sesgo (la intersección de cada una de las combinaciones lineales) también con una dimensión $\boldsymbol{n_h \times 1}$ cada uno.

En las ecuaciones 13, 14, 15 se puede examinar las compuertas del olvido, entrada y salida que son funciones representadas con $\boldsymbol{f^{(t)}}$, $\boldsymbol{i^{(t)}}$, $\boldsymbol{o^{(t)}}$ y reciben como entrada los vectores columna $\boldsymbol{x^{(t)}}$, $\boldsymbol{y^{(t-1)}}$ dónde se utiliza producto matricial para combinarlos con sus respectivos matrices de parámetros $\boldsymbol{W}$ y $\boldsymbol{R}$, por otro lado, también reciben el vector columna $\boldsymbol{c^{(t-1)}}$ ($\boldsymbol{c^{(t)}}$ en el caso de la compuerta de salida) y se emplea la operación de producto punto $\odot$ entre los vectores $\boldsymbol{p}$ y $\boldsymbol{c}$, se agregan los resultados incluyendo el vector de sesgo $\boldsymbol{b}$ y finalmente se aplica la función sigmoide $\sigma(x) = (1/1 + e^{-x})$ denotada por $\boldsymbol{\sigma}$ que tiene la propiedad de que está acotada en el intervalo (0, 1), de aquí que estos componentes se pueden interpretar como compuertas, si su valor es cercano a 0, no dejarán fluir la información que llevan, otra forma de verlo es como un porcentaje de la información que van a dejar fluir a través de las conexiones. Así pues, los vectores columna de dimensión $\boldsymbol{n_h \times 1}$ de las compuertas del bloque de memoria LSTM de la figura 3 se calculan con las siguientes ecuaciones:

$$f^{(t)} = \sigma(W_f x^{(t)} + R_f y^{(t-1)} + p_f \odot c^{(t-1)} + b_f) \tag{13}$$

$$i^{(t)} = \sigma(W_i x^{(t)} + R_i y^{(t-1)} + p_i \odot c^{(t-1)} + b_i) \tag{14}$$

$$o^{(t)} = \sigma(W_o x^{(t)} + R_o y^{(t-1)} + p_o \odot c^{(t)} + b_o) \tag{15}$$

Para hallar la información que potencialmente puede ser agregada a la memoria de largo plazo, como se observa en la ecuación 16 se utiliza un denominado bloque de entrada denotado por $\boldsymbol{z^{(t)}}$ que combina la información actual $\boldsymbol{x^{(t)}}$, la salida anterior $\boldsymbol{y^{(t-1)}}$ ponderados por sus matrices de peso respectivas $\boldsymbol{W}$ y $\boldsymbol{R}$ y esto se agrega también a un vector de sesgo $\boldsymbol{b}$, por lo que se obtiene un vector de dimensión $\boldsymbol{n_h \times 1}$, posteriormente, cada elemento del vector se le aplica una función no lineal g para finalmente obtener el vector columna $\boldsymbol{z^{(t)}}$. Frecuentemente se utiliza la función tangente hiperbólica que está acotada en el intervalo (-1, 1), en consecuencia, permite que los valores dentro del bloque LSTM no crezcan o disminuyan indefinidamente, lo que, de no ser controlado, puede causar problemas con largas secuencias.

$$z^{(t)} = g(W_z x^{(t)} + R_z y^{(t-1)} + b_z) \tag{16}$$

Como se observa en la ecuación 17, la celda de memoria en el tiempo $\boldsymbol{t}$ se actualiza con base en el porcentaje de información que se mantiene de $\boldsymbol{z^{(t)}}$ al combinarla con la compuerta de entrada $\boldsymbol{i^{(t)}}$, e igualmente, se agrega el porcentaje de la información de la memoria anterior $\boldsymbol{c^{(t-1)}}$ que es retenida por la compuerta del olvido $\boldsymbol{f^{(t)}}$.

$$c^{(t)} = z^{(t)} \odot i^{(t)} + c^{(t-1)} \odot f^{(t)}. \tag{17}$$

A continuación, para hallar la salida $\boldsymbol{y^{(t)}}$ del bloque LSTM como se aprecia en la ecuación 18, se toma la memoria de largo plazo $c^{(t)}$, se transforma con una función $\boldsymbol{h}$ que habitualmente es la función de tangente hiperbólica y finalmente se combina con la compuerta de salida $\boldsymbol{o^{(t)}}$ para regular que porcentaje de esa memoria debe ser usado para el siguiente procesamiento de la red neuronal.

$$y^{(t)} = h(c^{(t)}) \odot o^{(t)} \tag{18}$$

Es importante resaltar, que muchas de estas formulaciones están basadas en el procesamiento de una única observación, es decir, una sola secuencia de datos $\boldsymbol{x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(T_\infty)}}$, por ejemplo, una secuencia de datos puede ser una frase del español, y cada elemento de la secuencia de datos se puede definir como las palabras que componen esa frase, en tal escenario, $\boldsymbol{x^{(t)}}$ es un vector columna que representa numéricamente una palabra dentro de una frase, y puesto que hay varias frases, se puede hablar de varias observaciones y por tanto, de varias secuencias de datos. Sin embargo, usualmente se suele procesar varias secuencias de datos al mismo tiempo, que en definitiva, mejora considerablemente el rendimiento del algoritmo, y además, cambia las dimensiones de varios componentes, por ejemplo, el vector $\boldsymbol{x^{(t)}}$ pasaría a convertirse en una matriz de dimensiones $d \times n$, dónde d es la cantidad de covariables y n es la cantidad de observaciones que se plantea procesar al mismo tiempo.

Específicamente para la rotación de empleados, es interesante usar este enfoque para mejorar la varianza explicada del modelo a través de las relaciones no lineales y temporales que puede aprender, por ejemplo, en un problema similar, prediciendo la rotación de clientes, Alboukaey et al. (2020) argumentan que LSTM puede aprender de los clientes las conductas muy antiguas y las recientes y usar esta información para predecir si el cliente abandonará los servicios de la empresa y descubrieron que es un buen modelo que podría superar, por ejemplo, un *Random forest* con características sin mucho procesamiento, por lo que es justificado una comparación en el escenario de rotación de empleados.

Técnicas para manejar desbalance de clases

Durante el desarrollo de este estudio se evidenció un claro desbalance de las clases de los datos lo que afecta en gran medida el desempeño de todos de los modelos que se iban a construir, incluyendo el LSTM, por lo tanto, a continuación se introducen algunas técnicas para ayudar a solventar esta situación. Primero, empezando por la definición, en un problema de clasificación, el desbalance de clases ocurre cuando se tiene una proporción bastante inferior de datos de una o varias clases en comparación con la cantidad de las demás, lo que precisa especial atención al abordar un problema de aprendizaje automático, puesto que algunos clasificadores suelen dar más importancia a la clase mayoritaria y no logran distinguir correctamente las minoritarias, como en el caso de la regresión logística y árboles de decisión, asimismo, esta situación requiere revisar cuidadosamente las métricas indicadas para evaluar el desempeño de los modelos, debido a que métricas como la exactitud, incluso siendo muy alta, no logran reflejar la capacidad que tienen para predecir la clase minoritaria (Haixiang et al. 2017).

Para ilustrar la inconveniencia de usar únicamente la exactitud como métrica de evaluación para modelos entrenados en conjuntos de datos desbalanceados, considérese un clasificador binario para predecir si un cliente declinará o no un servicio en un tiempo determinado, si el 90 % de los clientes no lo declinan, un modelo que siempre clasifique al usuario de que no va a declinar tendrá una exactitud aproximada del 90 %, sin embargo, este modelo no consigue clasificar correctamente a ningún usuario que efectivamente fuese a declinar, por lo que la exactitud no es una métrica apropiada en tal escenario si el negocio requiere identificar correctamente a esos clientes que se van a ir, por lo tanto, en la sección de evaluación de esta tesis se revisarán algunas métricas adecuadas para el problema que se desea resolver.

Continuando con el desbalance de clases, existen muchas estrategias para reducir sus efectos, algunas de ellas son las técnicas de remuestreo y el aprendizaje sensible al costo. Las técnicas de remuestreo buscan cambiar la distribución de las clases, principalmente se suele realizar en una etapa de preprocesamiento de los datos, y pueden ser divididos en tres categorías: Submuestreo *(undersampling)*, sobremuestreo *(oversampling)* e híbridos (Galar et al. 2012).

Las técnicas de submuestreo consiste en reducir la cantidad de datos de la clase mayoritaria, por ejemplo al eliminar aleatoriamente sus instancias, no obstante esto implica una posible pérdida de información valiosa para el entrenamiento del modelo aunque suele ser una técnica de desbalanceo muy veloz, en contraste, las técnicas de sobremuestreo se fundamentan en el incremento de instancias de las clases minoritarias, por ejemplo, al duplicar algunas de sus instancias de manera aleatoria, sin embargo, esto puede llevar al sobreajuste del modelo a los datos de entrenamiento, es decir podría afectar la generalización a nuevos datos, en consecuencia, otras técnicas buscan reducir esta contrariedad al agregar nuevos datos sintéticos creados a partir de los datos originales, como es el caso de la técnica SMOTE (*Synthetic minority oversampling technique*), que de manera aleatoria selecciona un dato de la clase minoritaria, halla los k vecinos más cercanos a este, selecciona uno de esos aleatoriamente, y a continuación, entre ese dato original y ese vecino más cercano escogido, se genera un nuevo dato sintético con

interpolación (Galar et al. 2012). En cuanto a las técnicas híbridas, se buscan una combinación entre las técnicas de sobremuestreo y submuestreo, para ilustrarlo, Haixiang et al. (2017) encontró que los académicos recomiendan combinar submuestreo con SMOTE.

Por otro lado, las técnicas de aprendizaje sensible al costo regularmente definen una matriz para asignar una penalidad por clasificar erróneamente cierta clase cuando en realidad se trataba de otra, por lo tanto, si bien estas técnicas pueden ser útiles en otras circunstancias, son una buena alternativa en el desbalance de clases debido que al definir un costo mayor por el error con la clase minoritaria, guiará a que el modelo le de mayor importancia en su proceso de aprendizaje, sin embargo, se requiere definir una matriz de costo apropiada y aunque es posible definir esta matriz a través de experiencias pasadas o con un experto de dominio, se puede tratar como un hiperparámetro para hallar una configuración adecuada al problema específico que se desea resolver (Johnson & Khoshgoftaar, 2019).

Además, a pesar de que las técnicas de aprendizaje sensible al costo son menos exigentes computacionalmente que técnicas como SMOTE, pueden requerir una modificación del algoritmo de aprendizaje automático para aceptar esta matriz, mientras que las técnicas de remuestreo no dependen del clasificador usado, lo que puede justificar que estas últimas sean más populares al ser elegidas por investigadores que no son expertos en aprendizaje automático como sugiere Haixiang et al. (2017), haciendo la salvedad de que muchos paquetes de software ya suelen incluir un flujo para incluir las matrices de costo en los algoritmos de aprendizaje automático.

Para el desarrollo de esta investigación se probaron las técnicas de costos por clasificación errónea, SMOTE, submustreo aleatorio y finalmente la combinación de estos dos últimos. En la sección de evaluación se darán más detalles sobre cómo se llevó a cabo la comparación de estas técnicas y el ajuste de sus hiperparámetros a través de validación cruzada anidada.

Metodología

Breve descripción del conjunto de datos

El conjunto de datos consta de 3369 empleados que residían en diferentes países de América Latina, a saber, Argentina, Chile, Colombia, México, Panamá y Perú. Los datos se adquirieron de una red social pública conocida como LinkedIn. De cada empleado se disponen datos históricos de sus diferentes roles y experiencias laborales. Las experiencias laborales corresponden con la información que se tiene de un empleado cuando estuvo trabajando en una única empresa durante un tiempo ininterrumpido.

Además, por cada experiencia, un empleado puede tener uno o más puestos o roles que ha desempeñado, y esta es la granularidad más baja. Se organizarán los datos por cada rol de los empleados y a partir de allí se construirán dos estructuras de datos principales, la primera de ellas será usada por las técnicas de regresión logística, *random forest* y *extreme boosting*. Estas técnicas utilizarán la información de un rol anterior para predecir

si se irá de la compañía actual en menos de 1 año. En tanto que la segunda estructura que será utilizada por *LSTM* usará 2 o más roles (este es un hiperparámetro) para determinar con el histórico laboral del empleado si se irá de la empresa en menos de 1 un año.

Como se mencionó en la revisión de literatura, se escoge el umbral de decisión de un año para seguir la metodología desarrollada por otros investigadores que usualmente tienen a disposición información anual. Adicionalmente, realizar la predicción con un año de anterioridad ofrece a el equipo de recursos humanos tiempo suficiente para analizar y tomar decisiones, además, no es un intervalo de tiempo muy amplio como para que el riesgo se minimice para recursos humanos por ser un evento más lejano.

Extracción de datos de empleados

Luego de realizar una intensa búsqueda se llegó a la conclusión de que no hay muchos conjuntos de datos de empleados reales disponibles al público, y hasta donde sabemos, no hay una fuente disponible que incluya alguna variable temporal. Por lo tanto, se inició una revisión sobre técnicas para la adquisición de datos, y una de esas posibilidades era el uso de datos generados sintéticamente, no obstante, con el objetivo de acercarse más al problema real, se descartó esta posibilidad.

En consecuencia, se optó por la técnica conocida como *web scraping* para extraer datos no estructurados (como el texto o imágenes) de sitios web, por ejemplo, Google usa esta técnica para indexar y clasificar las páginas web para su motor de búsqueda. Específicamente, para esta tesis se escogió *web scraping* para recolectar datos de la red social LinkedIn puesto que es una de las redes sociales de empleados más populares y además, porque ofrece a los empleados la posibilidad de agregar información histórica principalmente de sus experiencias laborales.

Por tal razón, se llevó a cabo la construcción de un software para realizar la extracción de estos datos, en concreto, se usó un conjunto de herramientas de software conocido como Selenium, que aunque originalmente fue creado para la automatización de pruebas en navegadores web, tiene una gran versatilidad que facilita que pueda ser usado para recolectar datos. Hay varias alternativas a Selenium, por nombrar una, la librería *beautiful soup*, no obstante, a pesar de que tienen una curva de aprendizaje menor, se ven limitadas en algunas situaciones, por ejemplo, Selenium es más adecuado especialmente cuando las páginas web son dinámicas, es decir, cuando se requiere interactuar con la página y hay tiempos de espera, puesto que se está consultando un servidor antes de retornar la información.

Selenium puede manejar una gran cantidad de escenarios debido a que logra emular correctamente la forma en la que un usuario real interactuaría con esa página, usando los componentes de una manera similar a través de acciones como los clicks, ofrece la posibilidad de hacer *scrolling* e incorpora esperas para que las pantallas de carga finalicen, o hasta que cierto elemento de la página se haga visible. Puesto que LinkedIn es una página web dinámica con varias pantallas de espera, Selenium fue una decisión acertada.

La extracción de los datos de los componentes de este sitio web se realizó usando selectores de Selenium,

que son estrategias para analizar la estructura de una página web y obtener el elemento o la lista de elementos deseados, por ejemplo, uno de los selectores más eficientes y flexibles es el CSS (*Cascading Style Sheets*) selector pues permite localizar elementos a través de bastantes atributos y jerarquías que definen cada elemento y su estilo en una página web. Para ilustrarlo, en la figura 4 se puede observar un componente de la página web que describe una experiencia como desarrollador, y en la parte inferior un recorte del html (un lenguaje que permite definir la estructura de una página web). Si se desea obtener el texto que define la fecha de esta experiencia laboral, un manera de localizar el elemento sería usar el CSS selector ".t-14.t-black–light >span", Selenium se encarga de analizar el contenido de la página web y buscar un elemento span inmediatamente adentro de otro que contiene un atributo de clase cuyo texto es t-14 y t-black–light. Sin embargo, en ocasiones, múltiples elementos pueden coincidir con los selectores más sencillos y se requiere usar patrones más complejos para localizarlos apropiadamente.

Figura 4: Ejemplo de extracción de datos de un componente en una página web

Luego de extraer todos los datos, estos se guardaron en archivos **JSON** que siguen la sintaxis de *(JavaScript Object Notation)*, esto se debe a que se empleó el patrón *Page-object model* que busca organizar los distintos componentes de las diferentes páginas, cada uno en una unidad distinta conocida como objeto, estos son componentes de software en los que se definen comportamientos a través de métodos y características a través de variables. Por lo tanto, el patrón *Page-object model* permite relacionar los objetos entre sí para llevar a cabo la construcción del algoritmo que permitirá la adquisición de los datos. Luego de tener la estructura de la página bien definida, obtener la representación en JSON desde estos objetos es simple. La razón de peso para incorporar este patrón fue que proporcionaba un método simple para organizar y definir todos los selectores, además, puesto que cada objeto correspondía a una sección en particular de las páginas, los selectores construidos a partir de este,

limitaban la búsqueda de elementos sólo dentro de esa sección de la página, lo que ofrecía una manera de reducir el espacio de búsqueda de los elementos deseados. Por consiguiente, se siguieron buenas prácticas de programación que mejoraron la eficiencia y eficacia del proceso de adquisición de datos.

En la figura 5 se puede apreciar la estructura de los datos definidos dónde se presentan 2 empleados ficticios (no se usa información real para evitar la identificación individual de los empleados), un analista de mercadeo y un gerente de ventas. Cada empleado posee una lista de varias experiencias y a su vez, cada experiencia está conformada por varios roles. Por ejemplo, la experiencia que tuvo el primer empleado en la compañía ficticia Nexus elevate marketing está conformada por dos roles del empleado, un rol como analista de mercado junior y otro como analista de mercado semisenior. Así que el siguiente paso fue convertir esto a una estructura de tabla para el análisis de datos.


```
[
    {
        "headline": "Analista de mercadeo en Nexus Elevate Marketing Colombia",
        "connections": "500+",
        "country": "Colombia",
        "experience": [
            [
                {
                    "_POSITION_TITLE": "Analista de mercadeo semisenior",
                    "_DATE": "Nov 2022 - Present \u00b7 3 mos",
                    "_POSITION_DESCRIPTION": "Como analista de marketing, mi papel es ayudar a las empresas a tomar decisiones basadas en datos sobre sus estrategia
                    "_COMPANY_NAME": "Nexus Elevate Marketing \u00b7 Full-time"
                },
                [
                    {
                        "_POSITION_TITLE": "Analista de mercadeo junior",
                        "_DATE": "Apr 2022 - Oct 2022 \u00b7 6 mos",
                        "_POSITION_DESCRIPTION": "Analizo las tendencias del mercado, el comportamiento del consumidor",
                        "_COMPANY_NAME": "Nexus Elevate Marketing"
                    }
                ]
            ],
            [
                {
                    "_POSITION_TITLE": "Analista de mercadeo practicante",
                    "_DATE": "Aug 2021 - Mar 2022 \u00b7 8 mos",
                    "_POSITION_DESCRIPTION": "Asistir en la recopilación y análisis de datos de mercado y de la competencia.",
                    "_COMPANY_NAME": "Horizon Marketing"
                }
            ]
        ]
    },
    {
        "headline": "Gerente de ventas en Impacto Marketing Panama",
        "connections": "500+",
        "country": "Panamá",
        "experience": [
            [
                {
                    "_POSITION_TITLE": "Gerente de ventas senior",
                    "_DATE": "Oct 2020 - Presente \u00b7 1 año y 6 meses",
                    "_POSITION_DESCRIPTION": "Mi función principal es liderar el equipo de ventas y desarrollar estrategias para alcanzar los objetivos de ventas",
```


Figura 5: Ejemplo de la estructura JSON de los datos recolectados usando web scrapping.

Procesamiento de los datos

Inicialmente se analizó la cantidad de valores nulos presentes en el conjunto de datos, algunos predictores tenían demasiados valores nulos por lo que se decidió omitirlos, ese es el caso de algunas variables iniciales que se habían recolectado como la descripción del empleado sobre sí mismo y la descripción de los roles. Otra variable que se había recolectado era conexiones del empleado que medía la cantidad de personas con las que estaba relacionado en la red social, no obstante, cuando el empleado llega a las 500 conexiones, el sitio web trunca el valor a +500, por lo que la mayoría de los datos correspondían a este valor y se decidió excluir esta variable también.

Además, se consideró agregar variables como la edad y el género, no obstante, tal información no estaba disponible directamente en el sitio web, por lo que en un inició se usó la foto del empleado para realizar la predicción de estas variables con un modelo preentrenado para tal fin. Sin embargo, no todos los empleados

poseían una foto en el perfil, por lo que en ese caso no era posible hacer la estimación y generaría un valor nulo, igualmente, se desconoce el momento exacto en el que se capturó la foto, por lo que no sería exacto asignar esa estimación de la edad que tenía en ningún rol del empleado. Por lo tanto, para evitar estas inexactitudes y descartar datos por los valores nulos que estarían asociados a estas variables, se prefirió descartarlas por completo.

Posteriormente se procedió a extraer y dar formato a las fechas de inicio y de fin de los distintos roles. Se requirió asegurarse de que la información fuera correcta, ya que la ingeniería de características comenzarían desde allí, es por eso que se limpiaron estas variables, se eliminaron los roles que no poseían correctamente las fechas, se ordenaron (para poder encontrar variables como el índice de experiencia) y se crearon casos de prueba para asegurarse de que los datos fueran válidos. A partir de ahí, se notó algunos roles y experiencias con rangos de fechas superpuestos, por lo que se optó por usar un algoritmo voraz (algoritmos que buscan solucionar un problema al escoger la opción óptima en el momento) para fusionar esta información, ya sea manteniendo solo uno de los registros superpuestos priorizando el que tiene un intervalo mayor o cambiando la fecha de inicio y la fecha de finalización de los roles de modo que los intervalos fueran disjuntos.

Fue de gran importancia arreglar las fechas superpuestas porque de lo contrario muchas de las covariables e incluso la variable de respuesta habrían tenido muchos errores, por ejemplo, no es un escenario real que exista un rol del histórico del empleado, es decir, un rol que inició previamente, que tenga una fecha final posterior al rol que el empleado está desempeñando en el presente, eso implicaría que un dato del histórico posea información del futuro. Por lo tanto, con este algoritmo voraz se buscó corregir los intervalos y el orden de los datos para que se pueda ver el histórico de cada rol y este solo contenga información del pasado.

Seguidamente se empezó a computar algunas variables para cada empleado. Primero se calcularon los meses que cada empleado lleva en cada rol (*months in position*), esto es simplemente la diferencia entre la fecha de fin del rol y su fecha inicial. Después se hallaron los meses en la compañía acumulados hasta el momento para cada rol (*months in company acum*). Teniendo en cuenta que cada experiencia laboral corresponde a una única compañía como se aclaró desde la breve descripción del conjunto de datos, para hallar *months in company acum*, los datos se agrupan por cada experiencia laboral y se obtiene un conjunto de roles asociados a esa experiencia. Los meses acumulados en la compañía hasta cierto rol es igual a la cantidad de meses transcurridos desde el momento en el que el empleado entró a la compañía hasta la finalización de ese rol.

Además, para cada empleado se calculó los meses acumulados en todas las compañías hasta cada rol (*experience months*) como la sumatoria de los meses que duraron los roles previos al actual en las diferentes experiencias que ha tenido el empleado. Puesto que en el procesamiento anterior se aseguró que los intervalos definidos por las fechas de inicio y de fin de los distintos roles del empleado en su carrera fueran disjuntos, no hay posibilidad de que en estas variables acumulativas se hayan sumado múltiples veces el mismo intervalo de tiempo. No obstante, estas variables pueden guardar cierto grado de correlación entre sí a pesar de ser variables que reflejan diferentes conceptos, este apartado se analizará posteriormente.

A continuación se buscó hallar la variable respuesta binaria, para ello se calculó una variable temporal

(una variable para hacer cálculos intermedios pero que no será utilizada ni como covariable ni como variable respuesta) que representaba la cantidad de meses que iba a tardar desde el inicio de cada rol para que la experiencia correspondiente llegara su fin, usando para ello la fecha final del último rol existente para esa experiencia. Debido a que algunas experiencias laborales todavía no habían finalizado al momento de la captura de los datos, se mantuvo una variable binaria temporal para indicar si esa experiencia todavía estaba en curso o por el contrario, ya había finalizado. Luego se calculó la variable respuesta binaria para cada rol con la siguiente estrategia: si la experiencia ya había finalizado, se agregaba la etiqueta 1 cuando desde ese rol faltaba menos de un año para que saliera de la compañía, y un 0 de lo contrario. Por otro lado, si la experiencia no había finalizado, se definieron dos caminos, si los meses que faltaban para llegar a la fecha de captura de los datos, eran más de 12, incluso aunque la experiencia no había finalizado, ya se podía concluir que el empleado no se iba a ir en menos de 12 meses de la compañía desde ese punto, por lo que se agregaba la etiqueta 0. Finalmente, si los meses que faltaban para llegar a la fecha de captura de los datos era menor que 12, desde ese rol no era posible saber si en realidad se iría de la empresa en menos de 12 meses o no, porque esa experiencia no había finalizado, por lo tanto, estos roles se descartaron puesto que no era posible definir la variable respuesta para estos.

En el cuadro 1 se puede observar un ejemplo del conjunto de datos construido hasta este punto dónde cada fila corresponde a un rol diferente. Las columnas de la 3 a la 5, (*employee*, *experience* y *position*) son índices para identificar al empleado, la experiencia y el rol de cada registro. Así pues, se tienen dos empleados, el primero con el índice 5 y el segundo con el índice 6. Se puede notar que cuando se cambia de empleado, el índice de experiencia se reinicia, en consecuencia, esta variable representa la n-ésima experiencia del empleado menos uno (esto es así porque en el lenguaje que se desarrolló la investigación se cuenta desde el 0). De igual manera el índice del rol representa el n-ésimo rol menos uno en cada experiencia, y por eso se reinicia cada vez que el índice de experiencia cambia. Las primeras dos columnas son la fecha de inicio y de fin de cada rol (*start date* y *end date*), y ambas están en formato yyyy-MM-dd, se aprecia que el día siempre es el primero del mes, esto se debe a que la granularidad que se tiene es mensual. Adicionalmente, se observa que para cada empleado, las fechas no se solapan y están ordenadas por la fecha de inicio correctamente.

Prosiguiendo con las siguientes columnas, la variable *mon in pos* es la cantidad de meses que el empleado estuvo en el rol, *mon in comp* es la cantidad de meses que lleva en esa compañía luego de finalizado ese rol, se destaca que la experiencia con índice 2 tiene 3 roles, así que la cantidad de meses en la compañía se va acumulando hasta que inicia una nueva experiencia. La variable que sigue es *mon experience* que es la cantidad de meses de experiencia que lleva el empleado acumulado en su carrera contando los meses de ese rol en particular en ese registro. *Mon to leave* es la cantidad de meses que faltaba desde el inicio de cada rol para terminar la experiencia, esta variable de ninguna manera se puede utilizar como covariable puesto que es información que en un escenario real no se tiene acceso ya que es una variable que se construye teniendo en cuenta los roles futuros de cada registro, sin embargo, sí permite construir la variable respuesta (*left*) evaluando si su valor es menor a 12. Se resalta que la variable *mon in comp* no genera la variable respuesta, por ejemplo, el penúltimo registro del cuadro 1 dice que el

empleado 6 al finalizar ese rol acumulará 45 meses en la compañía actual, sin embargo, desde el inicio de ese rol, es decir, en el 2013-03-01, al empleado le tomó 10 meses para irse de la compañía como se ve con la variable *mon to leave*, y en consecuencia, la variable *left* se deja un 1, incluso aunque *mon in comp* es mayor que 12.

Es de suma importancia notar que con la forma en la que se organizaron y calcularon las variables no se utilizará la información del rol actual para predecir la variable respuesta, puesto que en la realidad no se podría saber con certeza estos datos debido a que el rol no habría finalizado. En su lugar, se plantea usar la información de los roles del pasado para predecir la etiqueta presente, por ejemplo, siguiendo el cuadro 1, para la tercera fila, los datos que potencialmente se usarían para predecir la etiqueta que es 0 en este caso, corresponden a los de la primera fila y segunda fila.

start date	**end date**	**employee**	**experience**	**position**	**mon in pos**	**mon in comp**	**mon experience**	**mon to leave**	**left**
2014-10-01	2016-10-01	5	0	0	24	24	24	24	0
2017-11-01	2018-03-01	5	1	0	4	4	28	4	1
2018-03-01	2020-03-01	5	2	0	24	24	52	49	0
2020-03-01	2021-02-01	5	2	1	11	35	63	25	0
2021-02-01	2022-04-01	5	2	2	14	49	77	14	0
2007-09-01	2009-01-01	6	0	0	16	16	16	31	0
2009-01-01	2010-04-01	6	0	1	15	31	31	15	0
2010-04-01	2013-03-01	6	1	0	35	35	66	45	0
2013-03-01	2014-01-01	6	1	1	10	45	76	10	1
2014-01-01	2022-01-01	6	2	0	96	96	172	96	0

Cuadro 1: Ejemplo de variables obtenidas a partir de la fecha de inicio y fin de los roles

Otras variables que fueron agregadas fueron el crecimiento real del producto interno bruto (*gdp*) y la tasa de desempleo y (*unemployment_rate*) con la finalidad de agregar información que no fuera auto-reportada y que además suministrara datos del estado del mercado laboral a través del tiempo para cada país. Por lo tanto se adquirieron datos de estas variables del sitio web del fondo monetario internacional (FMI) y se fusionó con el conjunto de datos que se llevaba. Para ello se tuvo que considerar que estas variables se midieron con una frecuencia anual, en tanto que los roles estaban con frecuencia mensual, por tanto, se realizó una interpolación lineal para llenar los datos que faltaban, es decir, se construyeron funciones lineales que conectaban los datos anuales disponible.

Una variable que se incluyó también en el análisis fue el país (*country*) que tenía que transformarse usando variables indicadoras (*dummy*). Cada variable indicadora se pone como 1 si el empleado pertenecía al país y un 0 de lo contrario. Se agregaron entonces las variables country_Chile, country_Colombia, country_Mexico, y country_Panama, country_Peru. La variable indicadora de Argentina está implícita, puesto que si todas las demás variables que se agregan son igual a 0, entonces, se puede concluir que el empleado pertenece a Argentina. Se escogió este enfoque dado que los países no pueden tener una relación de orden que justifique el uso de una sola

variable numérica.

En el cuadro 2 se observan las variables que se obtuvieron para el entrenamiento de los modelos con su descripción y una columna que indica si es una variable que varía con el tiempo, esto debido a que los diferentes roles corresponden a distintos momentos en la carrera del empleado, formando una secuencia de datos ordenados cronológicamente. La única variable que no varía con el tiempo fue el país de residencia del empleado, ya que solo se pudo conseguir el valor que tenía en el momento de la captura de los datos y no para cada rol, en consecuencia, se crea una limitación con las variables que se agregaron de acuerdo con esta característica (tasa de desempleo y producto interno bruto) puesto que pueden poseer información incorrecta si el empleado ha cambiado de país de residencia.

Nombre	**Descripción**	**¿Varía con el tiempo?**
Left company	Verdadero si el empleado deja la empresa en menos de un año, falso en caso contrario	Sí
Months in company acum	Cantidad de meses que el empleado ha trabajado en la empresa actual	Sí
Months in position	Cantidad de meses que el empleado ha trabajado en el puesto o rol actual	Sí
Experience months	Cantidad de meses que el empleado ha trabajado en su carrera	Sí
Experience index	El índice de la experiencia actual en una lista ordenada por fecha de todas las experiencias del empleado	Varía con el cambio de empresa
GDP	Crecimiento real del producto interno bruto (variación percentual anual) del país de residencia del empleado	Sí
Unemployment rate	Proporción de la cantidad de personas sin trabajo en el país de residencia del empleado	Sí
Position index	El índice del rol en la experiencia actual, estando los roles de la experiencia ordenados por fecha	Sí
Country	País de residencia del empleado	No
Year	Año en el que finalizó el rol	Sí
Month	Mes en el que finalizó el rol	Sí

Cuadro 2: Variables para el entrenamiento de los modelos

Por otro lado, se evidencia que las variables externas como el crecimiento real del producto interno bruto y la tasa de desempleos son datos que medidos en distintos meses sufrirán cambios, esto porque a pesar de que inicialmente los datos eran anuales, se realizó una interpolación para tener una aproximación de los valores variando mes a mes. Asimismo, el índice del rol, los meses que dura cada rol, los meses que el empleado lleva en la actual compañía, así como la cantidad de meses que lleva de experiencia hasta ese punto de su carrera, dependen principalmente de un rol en particular, por esta razón se determina que también varían en el tiempo. Finalmente, el índice de la experiencia también varía en el tiempo, sin embargo, no depende del rol, en cambio, depende de la experiencia, así que se puede concluir que varía menos sin ser totalmente una variable estática.

Estructuración de los datos

Más adelante se procedió a organizar los datos en dos estructuras principales, la primera de ellas se enfoca fundamentalmente en las técnicas que no están diseñadas especialmente para trabajar con secuencias de datos, y que usualmente utilizan un formato de tabla, es el caso de regresión logística, *random forest y gradient boosting tree*. En este paso, el objetivo es hallar parejas de roles contiguos de los cuales la variable respuesta corresponde a si faltan menos de 12 meses para salir de la empresa en el rol más reciente, mientras, el vector de covariables se extrae del rol más antiguo.

Dicho de otra manera, si se tienen $\boldsymbol{N}$ empleados y se utiliza el subíndice $\boldsymbol{i = 1, 2 \ldots N}$ para identificar el i-ésimo empleado, y además el superíndice $\boldsymbol{t = 1, 2 \ldots T_i}$ para identificar el t-ésimo rol de $\boldsymbol{T_i}$ roles que tiene el empleado $\boldsymbol{i}$, entonces la predicción de la variable respuesta denotada como $\boldsymbol{\hat{y}_i^{(t)} \in \{0, 1\}}$ se buscará modelar como se muestra en la ecuación 19. En esta ecuación se observa que para la predicción del empleado $\boldsymbol{i}$ en el rol $\boldsymbol{t}$, primero se estima la probabilidad condicional de que $\boldsymbol{y_i^{(t)}}$ sea igual a 1 dado el vector de covariables del rol anterior $\boldsymbol{x_i^{(t-1)}}$. Finalmente, se usa una función indicadora para asignar la etiqueta positiva si la probabilidad superar cierto umbral $\boldsymbol{c}$, que usualmente puede tener un valor inicial de 0.5.

$$\hat{y}_i^{(t)} = I(\Pr(y_i^{(t)} = 1 |\, x_i^{(t-1)}) > c) \tag{19}$$

En la figura 6 se visualiza para un empleado la primera forma de organización de sus datos usando un único rol del pasado para predecir la etiqueta del más actual. Se presenta una tabla dónde cada fila representa la información de un rol, sus covariables y su variable respuesta. De esta tabla se pueden construir 3 ejemplos de entrenamiento para el modelo. Se utiliza el vector de covariables del primer rol como entrada para el modelo, este lo procesa y realiza la estimación de la clase para el segundo rol. En este caso, la predicción del modelo es la clase positiva y coincide con la verdadera etiqueta del segundo rol, por lo tanto se representa con una marca de verificación. De la misma manera, las covariables del segundo rol se usan para predecir la clase del tercer rol, y en este escenario, la etiqueta asignada por el modelo no coincide con la etiqueta real que se encuentra en la tabla superior para el tercer rol, en consecuencia esto se representa con el símbolo de que es incorrecto. Por último, usando las covariables del tercer rol la predicción del cuarto rol es correcta. Finalmente, las covariables que están resaltadas del cuarto rol, no serán introducidas en ningún momento en el modelo, puesto que para este empleado en particular no se conoce la etiqueta del quinto rol, por tanto, estos datos se ignoran. Asimismo, la primer etiqueta del primer rol es ignorada debido a que no hay covariables de un rol en el pasado para realizar la predicción.

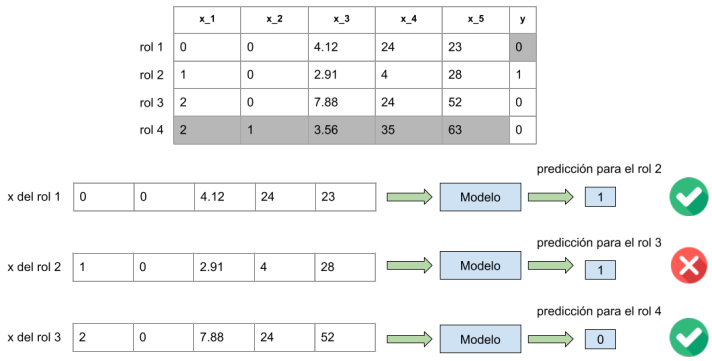

Figura 6: Organización de los datos para los modelos de regresión logística, *random forest y gradient boosting tree*

Puesto que para poder predecir la etiqueta en el rol actual, se requiere la información del rol pasado, se debieron excluir los empleados que tenían un único rol en su carrera. En la figura 7 se puede examinar la distribución del número de roles por cada empleado luego de la exclusión de los trabajadores con un único rol. En el eje **x** se mide el número de roles, acá se incluyen todos los roles de cada empleado incluso si fueron en diferentes compañías, y en el eje **y** la cantidad de empleados con esa cantidad de roles en particular, por ejemplo, 238 empleados tiene 2 roles en total. Finalmente, se nota que no hay ningún empleado que tenga un único rol. Se observa que hay un caso atípico de un empleado con 29 roles, analizando en detalle ese empleado, se descubre que lleva trabajando como consultor por más de 20 años, así que tiene sentido que haya tenido bastantes roles de duración corta en diversas empresas.

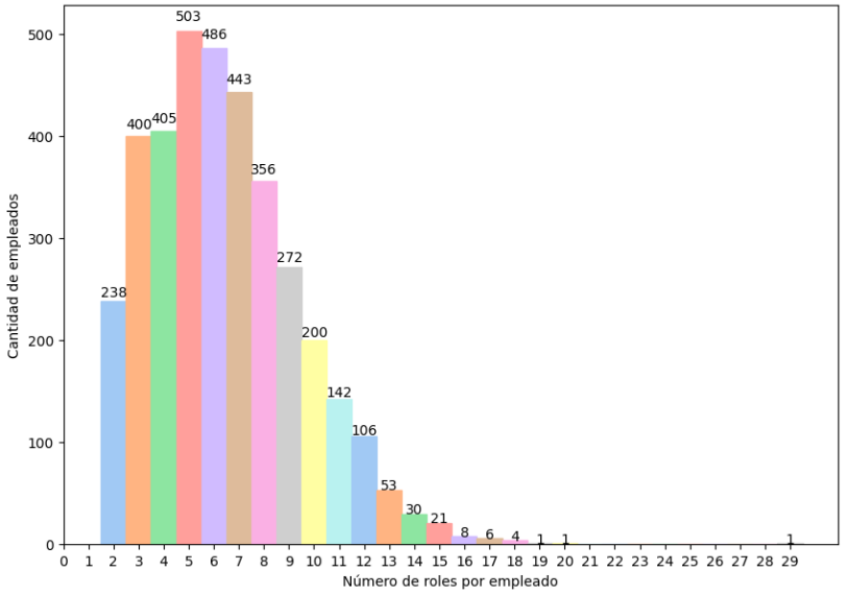

Figura 7: Distribución del número de roles excluyendo empleados con un único rol

En comparación, la segunda estructura para organizar los datos que será utilizada por *LSTM* busca aprovechar más datos previos del rol actual para la predicción de la etiqueta, es decir, una estructura de datos tipo panel.

En un problema similar en el que se buscaba crear modelos predictivos para identificar clientes que potencialmente dejarían cierta empresa de telecomunicaciones, Alboukaey et al. (2020) tomaron los datos de cada cliente y lo estructuraron como una secuencia de periodicidad diaria con un total de 90 días, es decir, para cada día se tenían variables como la cantidad de mensajes enviados, o la cantidad de minutos en llamada para ese día. Sin embargo, hay una diferencia bastante notable con los datos que se tienen para este estudio, y es que no es conveniente establecer la periodicidad de los datos, porque si bien el histórico de cierto empleado puede abarcar unos pocos años, el histórico de otro empleado puede abarcar décadas. Adicionalmente, puede ocurrir que el empleado con menor tiempo de experiencia tenga más registros que el más experimentado, puesto que puede cambiar de roles y de empresas mucho más seguido. Definir la periodicidad implicaría además descartar una gran cantidad de datos o crear información sintética para llenar los registros que no se poseen, lo que es un desafío enorme considerando que no se tienen muchos registros de cada empleado. Por lo tanto, en este trabajo se construirá la secuencia de registros multivariados de los roles de los empleados, ordenados por su fecha como en el formato presentado por Alboukaey et al. (2020), pero no se tendrá periodicidad, lo que implica que la distancia en meses entre dos roles contiguos no será fija.

Aclarado lo anterior, a continuación se establece la ecuación 20, que a diferencia de la ecuación 19, ahora halla la probabilidad usando dos o más vectores de covariables correspondientes a cada rol del pasado, hasta un total de $\boldsymbol{s}$ roles previos usados como contexto de tal modo que el último rol a incluir es $\boldsymbol{x_i^{(t-s)}}$. De esta manera se tendrá una secuencia de vectores de covariables que cambian entre roles, aclarando que de acuerdo con el cuadro 2 las variables que no varían entre los diferentes roles son variables estáticas que se mantienen igual a través de toda la secuencia de datos $\boldsymbol{x_i^{(t-1)}, x_i^{(t-2)}, \ldots, x_i^{(t-s)}}$, como es el caso del país del empleado. Sin embargo, la mayoría de variables tienen la indicación de que efectivamente varían entre cada elemento de la secuencia.

$$\hat{y}_i^{(t)} = I(\Pr(y_i^{(t)} = 1 |\, x_i^{(t-1)}, x_i^{(t-2)}, \ldots, x_i^{(t-s)}) > c) \tag{20}$$

En la figura 8 se representa la segunda forma de organización de los datos para construir los ejemplos para el modelo *LSTM* con un empleado que tiene un total de 5 roles. Si se escogen 3 roles para realizar la predicción de un cuarto rol, entonces de estos datos se pueden construir dos ejemplos para el entrenamiento del modelo, en concreto, las covariables del rol número 1, 2 y 3 para realizar la predicción del cuarto rol, y los roles 2, 3, 4 para predecir la etiqueta del quinto rol. Además, se observa que la predicción para el rol número 4 no coincide con el valor verdadero, mientras que la predicción del quinto rol si fue correcto.

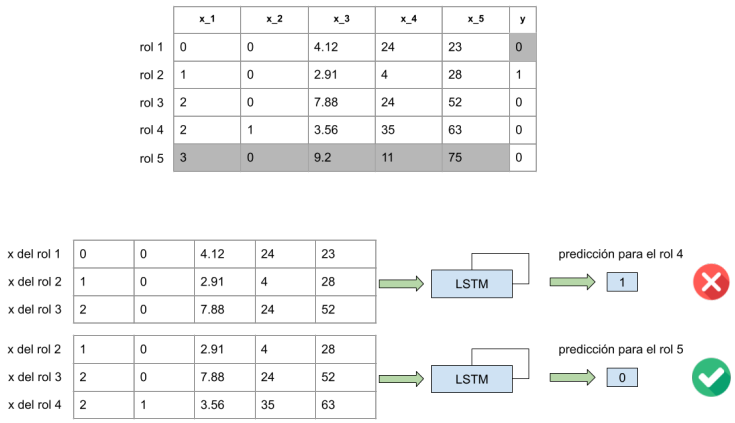

Figura 8: Organización de los datos para *LSTM*, usando 3 roles como contexto

A medida que se utilizan más roles en el pasado para construir los ejemplos y efectuar el entrenamiento del modelo, se excluyeron los empleados que no tienen la suficiente cantidad de roles en su histórico. Una alternativa para evitar esto era rellenar con 0s los datos que no se tenían, sin embargo, para no agregar más ruido o variaciones a los datos reales, se decidió excluirlos. Así pues, al cambiar la cantidad de roles que usaría el modelo, se realizaba un filtro para mantener solo los empleados que tenían la cantidad necesaria de roles y se procedía con el entrenamiento. En la figura 9 se observa cómo quedaría la distribución del número de roles por empleado usando 3 roles para predecir el cuarto, y por tanto se excluyen los empleados con 3 roles o menos. Esta gráfica es particularmente útil porque permite comprender el impacto en pérdida de datos al aumentar la cantidad de roles requeridos del histórico para el modelo de LSTM. Por ejemplo, si en vez de usar 3 roles para predecir el cuarto, se usan 4 roles para predecir el quinto, se estarían excluyendo adicionalmente 405 empleados, lo que empieza a perjudicar el entrenamiento por la reducción de los datos. Por esta razón se debe ser prudente con la elección del número de roles para realizar la clasificación, además, un número muy grande puede reducir la utilidad del modelo para recursos humanos pues no podrían usar este modelo si el empleado no tiene el mínimo requerido de roles.

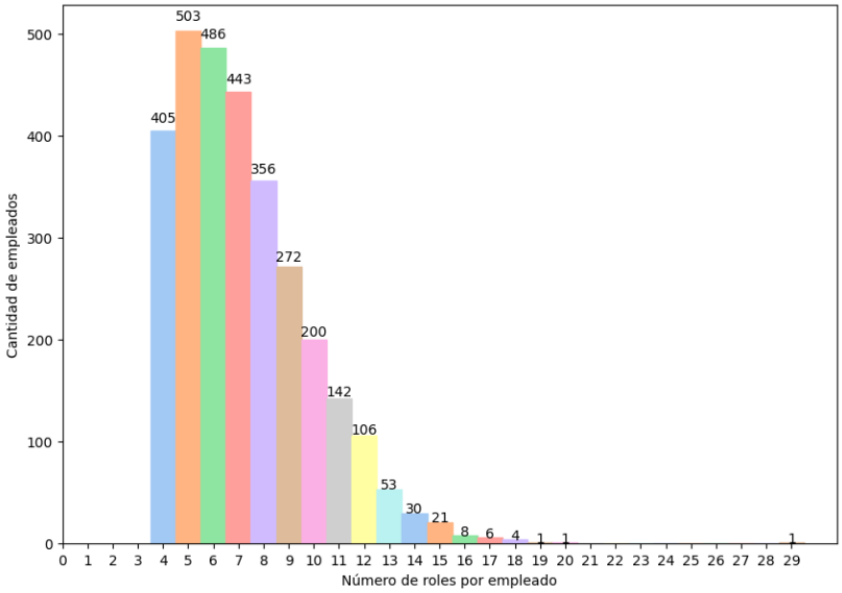

Figura 9: Distribución del número de roles excluyendo empleados con 3 roles o menos

Exploración de los datos

En la figura 10 se observa la distribución de la cantidad de roles que inician cada año. Se observa una cola muy grande hacia la izquierda hasta el año 1900. Se debe a un valor atípico de un empleado cuyas fechas no fueron capturadas correctamente, por tanto, se removieron los registros de este empleado. Luego de esta limpieza, el primer rol inicia en el año 1968. A pesar de que no se tienen muchos datos de estas fechas, tales registros sí parecen haber sido capturados de manera correcta pues se trata de empleados con una extensa experiencia laboral, así que no se eliminarán estos datos, en vez de ello, más adelante se mostrará cómo se eliminaron los valores atípicos de manera multivariante.

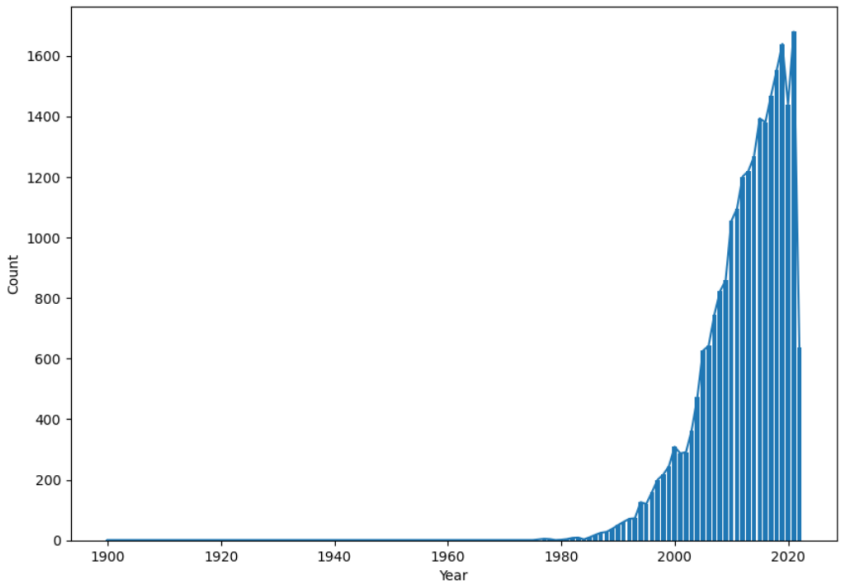

Figura 10: Distribución de frecuencias del número de roles por año de inicio

En la figura 11 se observa el diagrama de cajas y bigotes para la variable *months in position* y en el cuadro 3 un resumen de estadísticos básicos. La cantidad de registros de roles que hay es de 23919. Esto se debe a que si bien el conjunto de datos cuenta con 3369 empleados, cada uno tiene varios roles, por lo que la cantidad de registros aumentó a 23919. El número mínimo de meses que permanece un empleado en un rol es 1 mes, mientras el máximo son 432 meses. Se nota que hay una gran cantidad de puntos extremos, por lo que el promedio está un poco sesgado hacia la derecha con 27 meses cuando la mediana son 19 meses, es decir, se tiene una distribución asimétrica a la derecha. La desviación estándar es relativamente grande, lo que tiene sentido dada la gran diversidad de empleados que tiene la muestra de datos, puesto que pertenecen a distintos países, han pasado por distintas compañías, algunos empleados van iniciando su carrera mientras otros ya son muy experimentados como el empleado que empezó su primer rol en el 1968 como se discutió previamente. Además, el 75 % de los roles no duran más de 36 meses, lo que implica que son muy pocos los empleados del conjunto de datos que llegan a estar un tiempo prolongado en el mismo rol.

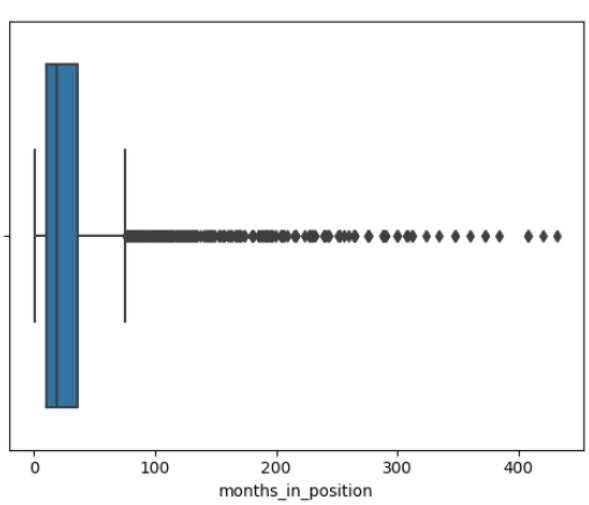

Figura 11: Diagrama de cajas y bigotes de la variable *months in position*

Estadística	**Valor**
Cantidad	23919
Promedio	27.24
Desviación estándar	31.72
Mínimo	1
Percentil 25	10
Percentil 50	19
Percentil 75	36
Máximo	432

Cuadro 3: Estadísticos *months in position*

Por otro lado, en la figura 12 se observa el diagrama de cajas y bigotes para la variable *months in company acum*, en comparación con *months in position* también es asimétrica hacia la derecha, sin embargo, como se aprecia en el cuadro 4 tiene una mayor variabilidad, así cómo un mayor promedio y una mediana superior, lo que evidencia que en general *months in company acum* es una variable que toma valores más grandes.

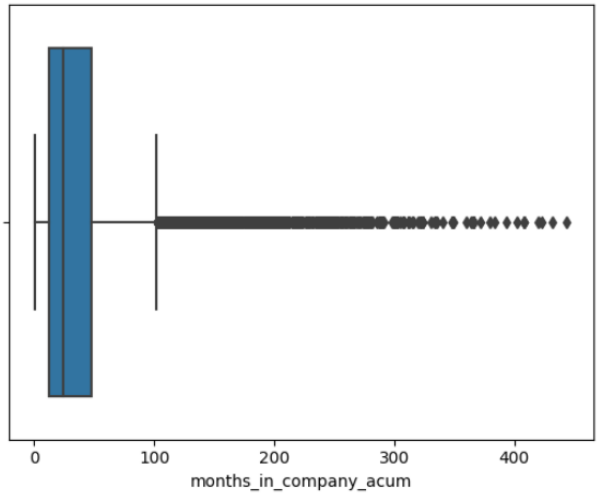

Figura 12: Diagrama de cajas y bigotes de la variable *months in company acum*

Estadística	**Valor**
Cantidad	23919
Promedio	39.07
Desviación estándar	45.27
Mínimo	1
Percentil 25	12
Percentil 50	24
Percentil 75	48
Máximo	444

Cuadro 4: Estadísticos *months in company acum*

La figura 13 es el diagrama de cajas y bigotes para la variable *months experience* y en el cuadro 5 el resumen de sus estadísticos correspondientes. También se constata que es asimétrica a la derecha, tiene un valor mínimo de 1 y máximo de 630. Este máximo valor corresponde al empleado que inició su experiencia en el 1968 como se analizó anteriormente. Esta variable tiene una mayor desviación estándar por lo que los datos se distribuyen más dispersos en comparación con las variables vistas hasta el momento. La mediana es de 88 meses o menos de experiencia, lo que serían 7.3 años, y el 75 % tiene 154 meses o menos, es decir, 12.8 años aproximadamente. Esto sumado a que solo el 25 % tiene menos de 42 meses de experiencia, es decir, 3.5 años, evidencia que hay bastantes empleados experimentados en el conjunto de datos.

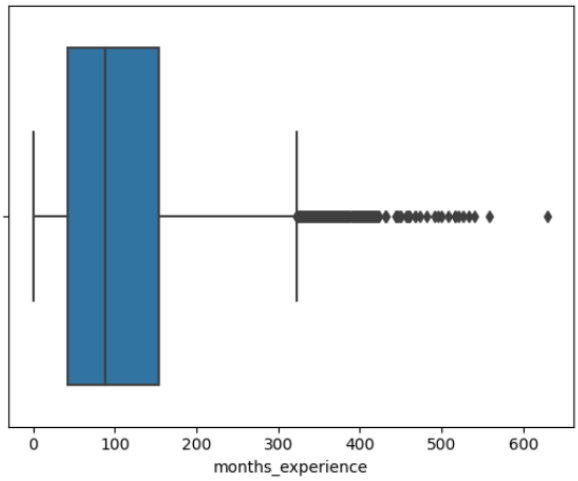

Figura 13: Diagrama de cajas y bigotes de la variable *months experience*

Estadística	**Valor**
Cantidad	23919
Promedio	107.1
Desviación estándar	82.03
Mínimo	1
Percentil 25	42
Percentil 50	88
Percentil 75	154
Máximo	630

Cuadro 5: Estadísticos *months experience*

La figura 14 corresponde al diagrama de frecuencias relativa de los países de los empleados. En la adquisición de los datos se buscó que ningún país tuviera una frecuencia muy grande en comparación con los demás y por eso se observa porcentajes razonablemente cercanos. Se evidencia una mayor cantidad de empleados de Colombia con el 18.6 % del total, seguido por Perú con 17 %, mientras que el país con menos empleados es Panamá con el 15.6 %, es decir que se aleja en un 3 % del mayor que es Colombia.

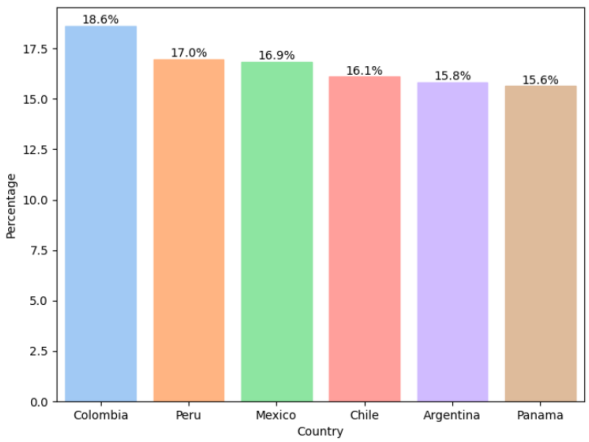

Figura 14: Diagrama de frecuencias relativas de la cantidad de empleados por país

Las figuras 15, 16 son los gráficos de líneas para las variables externas que se obtuvieron: la tasa de desempleo y el crecimiento real del producto interno bruto (variación porcentual). Lo primero que se debe notar

es que solo se pudo obtener datos desde 1980, no obstante, aun cuando se tienen roles que iniciaron antes de este año, todos finalizan en el año 1980 o después, por lo que no surgió ningún problema en este aspecto al fusionar los datos debido a que se usó el año de finalización de los roles de los empleados para asignar los valores correspondientes del crecimiento real del producto interno bruto y la tasa de desempleo. Otro aspecto relevante de estos gráficos es que el comportamiento de sus variables pueden dar una idea sobre el contexto macroeconómico del respectivo país del empleado. Por ejemplo, a inicios de los 2000 empezó un descenso del producto interno bruto de Argentina coincidiendo en este caso con el aumento de su tasa de desempleo, esto por una grave crisis social y económica en el que el país tuvo un alto nivel de endeudamiento que equivalía a un gran porcentaje del producto interno bruto argentino con lo que el gobierno tuvo que anunciar su imposibilidad para pagar las deudas contraídas (Gómez, 2002). Otro evento para ilustrar es la crisis económica que surgió debido a la pandemia de COVID, pues su impacto se evidencia en el producto interno bruto y en la tasa de desempleo de los distintos países que están presentes en los gráficos.

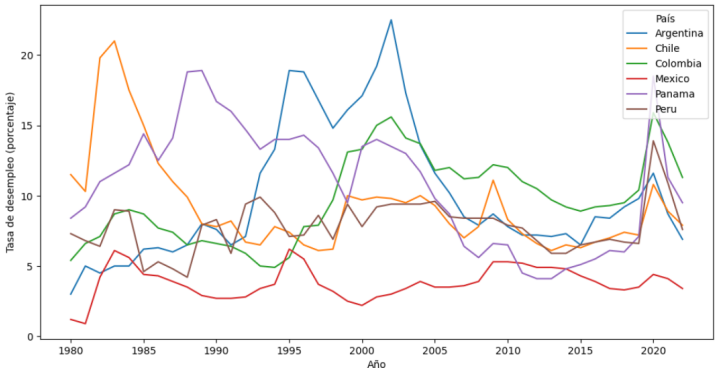

Figura 15: Gráfico de línea de tasa de desempleo

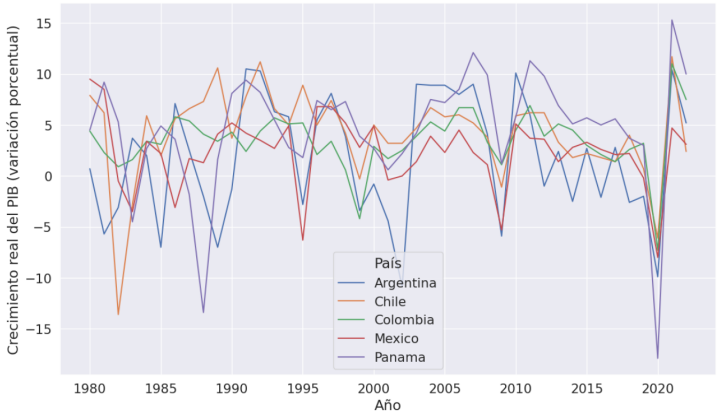

Figura 16: Gráfico de línea del crecimiento real del producto interno bruto

Tratamiento de datos atípicos

Para el tratamiento de datos atípicos se llevó a cabo un análisis multivariante a través del uso de las distancias entre las observaciones para determinar aquellos registros que se alejaban del centro de los datos. Para esto, dado que las variables tienen diferentes escalas, varianzas y pueden tener cierto grado de correlación se

buscó una distancia apropiada que fuera invariante a la escala y que tuviese en cuenta estas tendencias de los datos basándose para ello en la matriz de covarianzas que mide las dependencias entre los distintos atributos que se tienen (Ghorbani, 2019). La ecuación 21 corresponde a la fórmula para calcular la distancia entre un vector X y un vector de medias μ que representa el centro de la distribución de los datos. Se aprecia que usa la inversa de la matriz de covarianza denotada por la letra griega Σ, es por ello que si hay una colinealidad muy notable en los datos, esta matriz no tendría inversa, en consecuencia, se verificó el número condición y el determinante de esta matriz para evaluar qué tan invertible es.

$$D(X, \mu) = \sqrt{(X - \mu)^T \Sigma^{-1} (X - \mu)} \tag{21}$$

En un principio, el determinante de la matriz de covarianzas estaba alejado del 0 lo que indica que la matriz tiene inversa. No obstante, el número condición era muy alto lo que implicaba que pequeños cambios en los datos iban a cambiar notablemente la estimación de la matriz de covarianzas, por consiguiente, se optó por emplear el método de contracción de Ledoit y Wolf (2004) para estimar la matriz de covarianzas. Esta técnica se basa en el hecho de que la matriz más invertible es la matriz identidad, y que para lograr que una matriz tenga inversa, se puede usar la transformación de hacer un poco más grande los valores de su diagonal para que se asemeje a la matriz identidad.

Posteriormente se calculó las distancias de Mahalanobis usando la matriz estimada con el método de Ledoit y Wolf. En la figura 17 se observa el diagrama de cajas y bigotes para las distancias de Mahalanobis que se calcularon de cada observación de acuerdo a la ecuación 21. Se aprecia que hay una gran cantidad de datos que están alejados a más de 1.5 veces el rango intercuartílico del cuartil 3 y que están representados con los rombos de color negro a la derecha. A pesar de ser una gran cantidad de datos, se decidió excluir estos más de 1700 registros con la finalidad de evitar que los modelos se vieran afectados gravemente por estos datos atípicos.

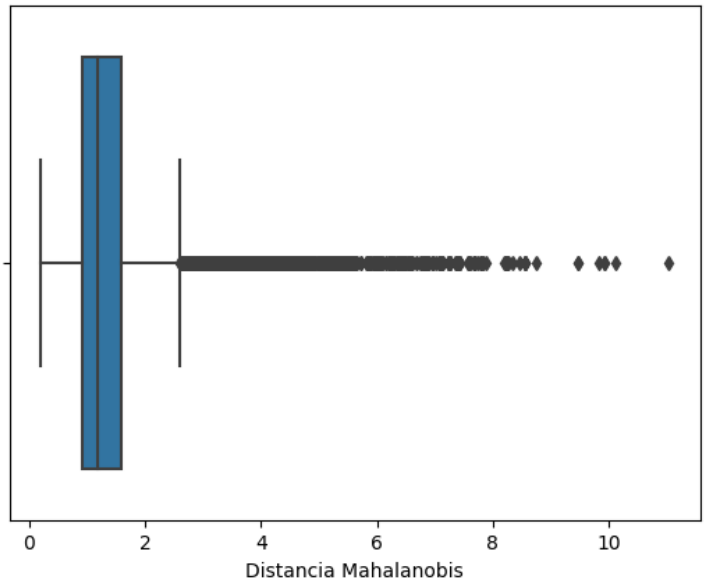

Figura 17: Diagrama de cajas y bigotes para las distancias de Mahalanobis

Análisis de dependencias

En la figura 18 se observan diagramas de dispersión para algunas combinaciones de las covariables. Adicionalmente cada observación se representa con una x o un círculo dependiendo del valor de la variable respuesta para el siguiente rol en la secuencia de datos indexada en el tiempo (en el diagrama se nombra como *left next position*). En otros términos, con una x se representan los datos de los registros de empleados que en su siguiente rol se irán de la compañía en menos de 12 meses, y con un círculo si en ese rol siguiente pasó lo opuesto. Es importante recordar que los datos de cada rol solo se pueden usar para predecir los roles del futuro, porque todos los registros reflejan el estado del rol una vez ha finalizado, en cuyo caso, ya se sabría la variable respuesta y no tendría sentido, es por ello que se creó esta variable *left next position* para visualizar la relación de todas las posibles covariables $x^{(t-1)}$ contra las variables respuesta $y^{(t)}$. Además, en la diagonal se observa la distribución de cada covariable de acuerdo al valor que toma la variable respuesta *left next position*.

Empezando con el análisis, en la diagonal se puede apreciar que no hay algún corte lineal efectivo que pueda separar las clases completamente usando una única variable pues las distribuciones se ven superpuestas. No obstante, se nota levemente que las observaciones de la clase positiva, $y^{(t)} = 1$, en variables como *months in company acum* tienen cierta tendencia a tomar valores más cercanos entre sí, mientras que la otra clase tienen mayor dispersión, esto se podrá visualizar mejor con diagramas de cajas y bigotes, así que más adelante se mostrará este gráfico. En los diagramas de dispersión se empiezan a observar varias correlaciones como *months in company acum* y *months in position*, sin embargo, en términos generales no se aprecia un par de variables que unidas sugieran una clara distinción de las observaciones. Por ejemplo, en la gráfica del producto interno bruto contra la tasa de desempleo, se observan los puntos bastante esparcidos en todo el área del gráfico lo que no permite deducir un patrón evidente.

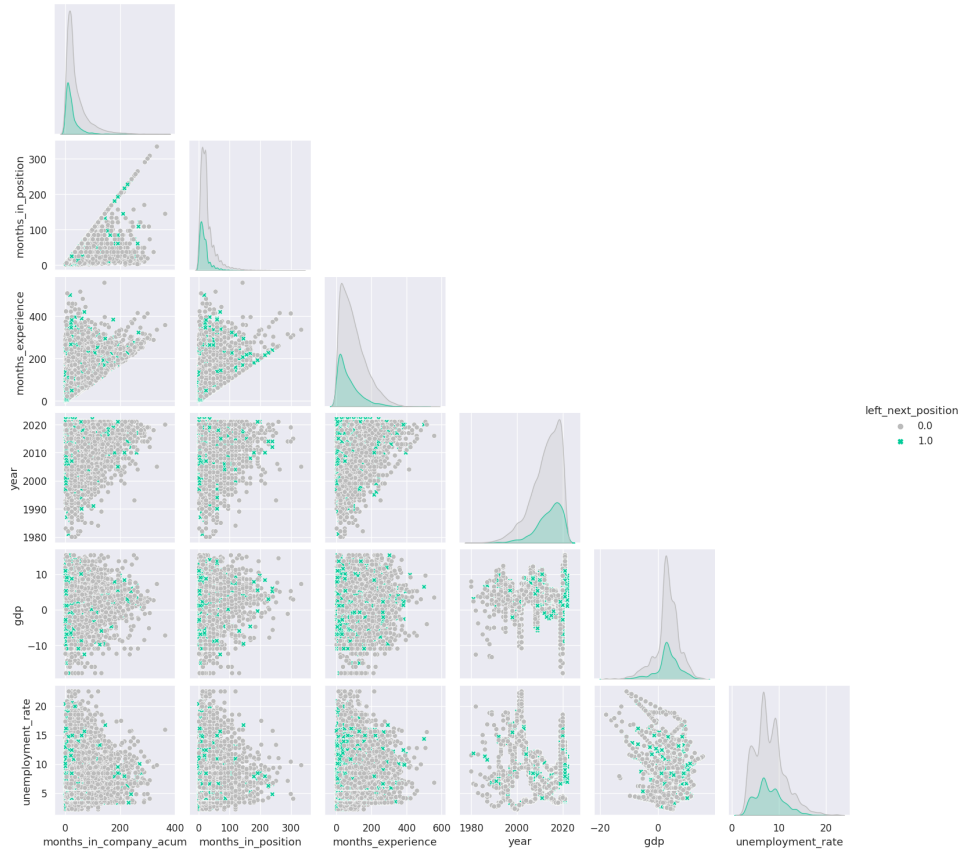

Figura 18: Gráfico de pares de las covariables

En la figura 19 se aprecian los diagramas de cajas y bigotes de cada covariable por clase. Acá se puede ver que la dispersión parece ser menor en las covariables *months in position, months in company acum, months experience* y *year* cuando la clase es positiva como se sospechaba desde el anterior gráfico, además de que sus medianas se observan ligeramente distintas. Este puede ser un aspecto que podrían aprovechar los modelos para lograr una mejora en la clasificación de las observaciones. En contraste, las variables *month, gdp* y *unemployment rate* parecen tener una distribución bastante similar entre clases, lo que podría conllevar a que estas variables no sean tan relevantes por sí solas, si bien esto se debe confirmar con los resultados del entrenamiento.

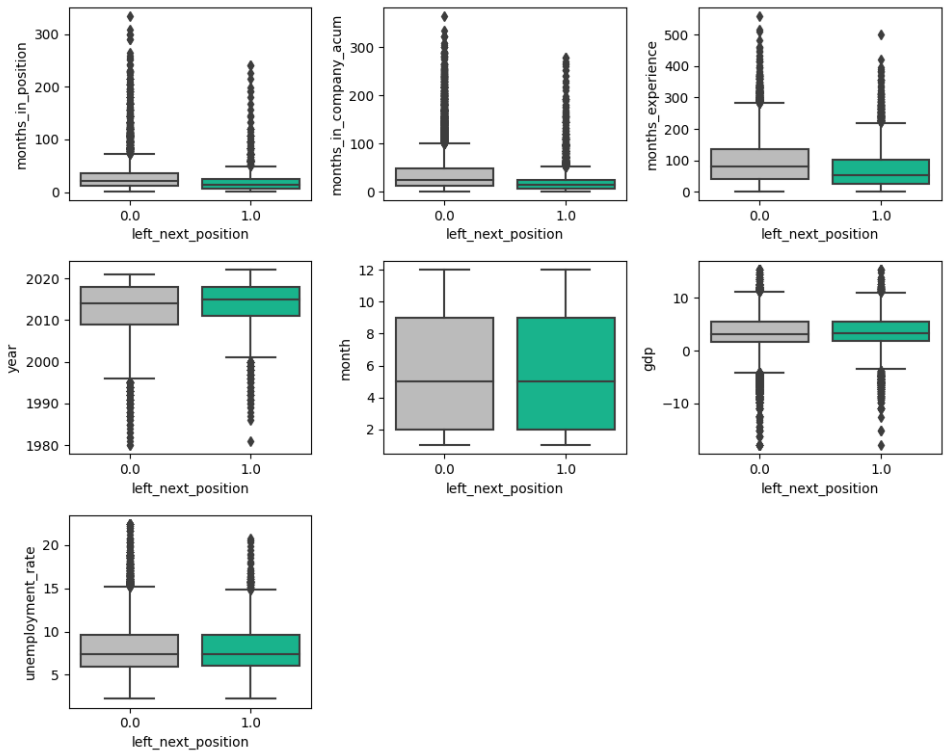

Figura 19: Diagramas de cajas y bigotes por clase

Para analizar más la sospecha de las diferencias entre las medianas de los atributos por clase, se realizó para cada covariable una prueba no paramétrica (no se asume una distribución en particular de las variables) conocida como Mann-Whitney U test, cuya hipótesis nula corresponde a que no hay diferencia entre dos medianas. Así pues, se usó la variable respuesta para partir cada covariable en dos grupos, el primero con los valores de las covariables cuya variable respuesta correspondía a la clase 1, es decir, que el empleado salía de la empresa en su siguiente rol en menos de 12 meses, y el segundo grupo los valores de las covariables restantes. Posteriormente, se asumió independencia y al realizar la prueba se obtuvo como resultado que las variables *months in position, months in company acum, months experience, year* y *gdp* si parecen tener medianas significativamente diferentes con un nivel de significancia del 5 %. Mientras que para *month* y *unemployment rate* no había evidencia suficiente para rechazar el test. Es interesante notar que la prueba además incluyó *gdp* dentro de las variables con diferencia, sin embargo, visualmente es difícil de percibir. No obstante, es posible que los grupos de hecho no sean independientes puesto que cada grupo puede tener datos del mismo empleado, por tanto, al final la forma definitiva de saber si la covariable puede ser un buen predictor de la etiqueta será con el entrenamiento de los modelos, aunque todo este análisis permite comprender un poco mejor el efecto de las variables.

En la figura 20 se observa la matriz de correlación de Spearman, se escoge por no asumir normalidad en los datos y permitir capturar algunas correlaciones no lineales. Se encuentra que hay algunas variables con una correlación considerable. Por ejemplo, la correlación entre *months in position* y *months in company acum* es de 0.81 y la correlación entre *months in company acum* con *months experience* es de 0.48. Adicionalmente, las variables de índice como *position index* y *experience index* también tienen correlaciones notables. Con la variable

respuesta la variable más correlacionada es *months in company acum* mientras que la menos correlacionada es *unemployment rate*. Si bien esta correlación con la variable respuesta no es exacta ya que no es ordinal y este es un requerimiento de Spearman, sí coincide en gran medida con los análisis realizados a partir de las gráficas presentadas previamente. Con base en esto es posible realizar una selección de variables, en las que se excluirán las variables de índice y *unemployment rate* (aun cuando teóricamente puede verse como una variable muy importante, las variables indicadoras del país y el producto interno bruto pueden representar el mercado laboral en ese instante de tiempo). Asimismo se quitarían las variables *months in position*, *months experience* y se dejaría *months in company acum*. No obstante, en modelos que son un poco más resistentes a la correlación de las variables como *random forest* se incluirán todas las variables y se llevará un análisis de importancia para corroborar estos resultados.

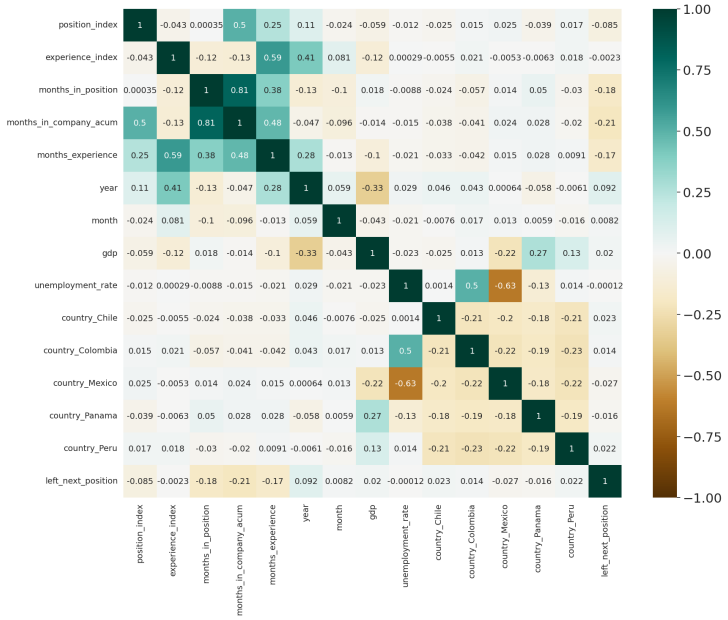

Figura 20: Matriz de correlación de Spearman

En la figura 21 se aprecia los diagramas de cajas y bigotes por clase para cada rol del pasado que se va a usar para predecir el quinto rol con la variable *months in position*. Es decir, de izquierda a derecha hay cuatro gráficos, porque son cuatro roles para predecir el quinto. La finalidad de este gráfico es observar si los datos en secuencia muestran algún patrón evidente y lo que se observa es que en todos los gráficos cuando la categoría del quinto rol corresponde a la clase positiva, entonces la mediana de la variable *months in position* es menor que si no perteneciera a la categoría positiva. Ya este patrón se había encontrado con los datos agregados (no divididos por el número del rol correspondiente) lo que puede sugerir que no hay grandes ganancias en usar los datos secuenciales con esta variable. Asimismo se puede observar el mismo comportamiento para la variable *months in company acum* en la figura 22.

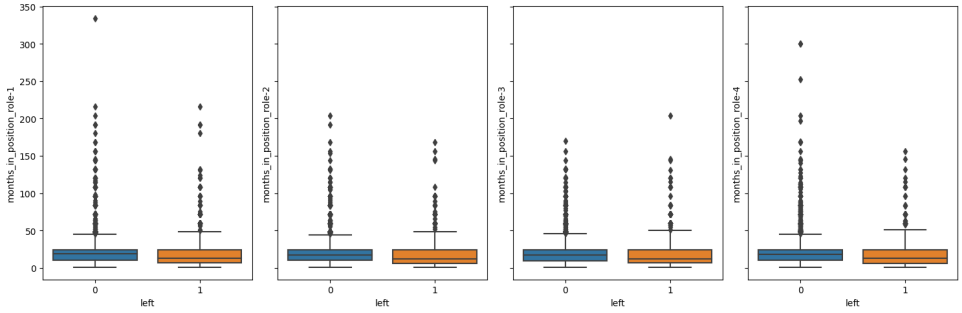

Figura 21: Months in position para predecir el 5 rol variando el número del rol

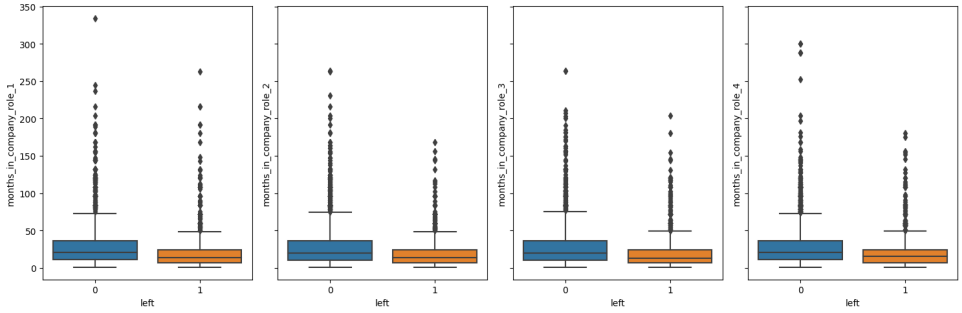

Figura 22: Months in company acum para predecir el 5 rol variando el número del rol

La misma gráfica realizada para la covariable de la tasa de desempleo (*unemployment rate*) se puede apreciar en la figura 23. Se nota una ligera diferencia de los dos grupos en el primer rol que corresponde a la primera subgráfica de la izquierda. Los registros del primer rol de los empleados que salieron en menos de 12 meses en el rol quinto, tienen una menor mediana en la tasa de desempleo, lo que es bastante interesante y podría ayudar al modelo LSTM a realizar la clasificación. Sin embargo, los demás roles no parecen aportar mucho a la clasificación del quinto rol.

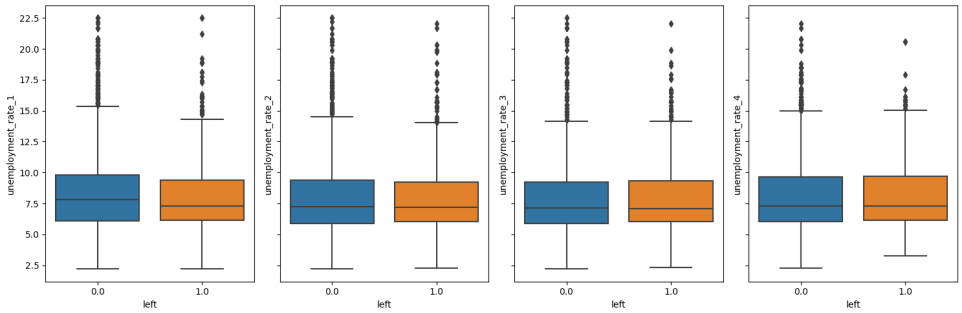

Figura 23: Unemployment rate para predecir el 5 rol variando el número del rol

En la figura 24 se puede apreciar el diagrama de frecuencias de las clases existentes para la primera estructura de datos, y en la figura 25 el mismo diagrama pero usando la segunda estructura de datos empleando cuatro roles para predecir el quinto. Lo primero a observar es que la cantidad de etiquetas puede ser menor a la cantidad de datos que se tenía inicialmente, esto se debe a dos razones fundamentales, la primera es que la etiqueta del rol más antiguo de cada empleado se descarta por no tener contexto del pasado para realizar su predicción. La segunda razón es que para la primera estructura se tuvo que excluir los empleados con un único rol, y para la

segunda estructura (para este gráfico en particular), se tuvo excluir los empleados que tenían cuatro roles o menos, pues se requería un rol adicional para construir un ejemplo de entrenamiento.

Prosiguiendo con las figuras 24 y 25, se evidencia que en ambas gráficas existe un desbalance bastante pronunciado en el que los ejemplos dónde el empleado se va en menos de 12 meses de la compañia (la clase positiva con etiqueta 1) es minoritaria. Para la segunda estructura de datos (la estructura para los datos secuenciales), este gráfico puede variar dependiendo de la cantidad de roles usados para llevar a cabo la clasificación, sin embargo, independientemente del número de roles elegidos, siempre se conserva un desbalance en el que la clase minoritaria es la clase positiva. Así pues, en la sección de modelado se describirá cómo tratar esta particularidad de estos datos.

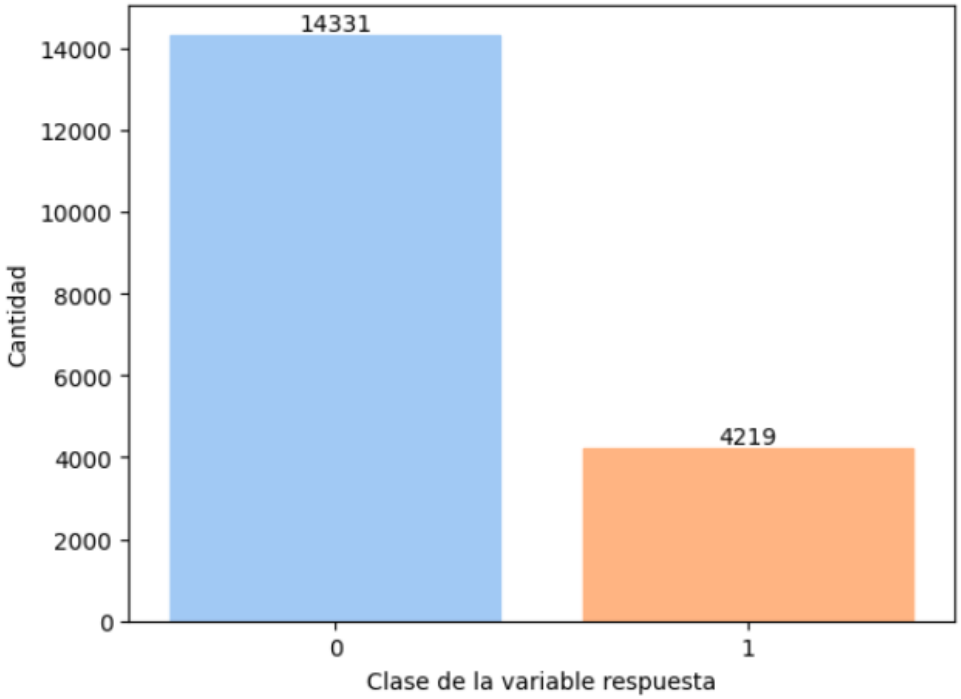

Figura 24: Desbalance de clases primera estructura de datos usando 1 roles para predecir el siguiente

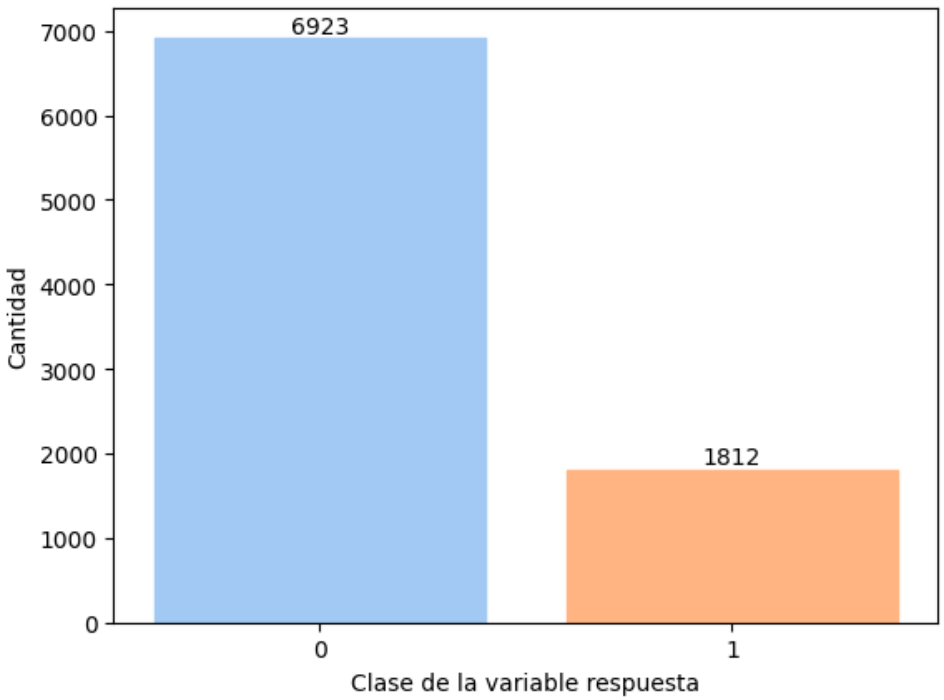

Figura 25: Desbalance de clases segunda estructura de datos usando cuatro roles para predecir la clase del quinto

Evaluación

Se van a entrenar los modelos de regresión logística, *random forest y gradient boosting tree* con la primera estructura de los datos mencionada en la sección del preprocesamiento de datos y que se puede visualizar en la figura 6. Asimismo se va a entrenar el modelo de LSTM con la segunda estructura de los datos y que se puede visualizar en la figura 8.

Métricas de evaluación

Dado que se va a predecir si un trabajador dejará una compañía en menos de 12 meses (se considera que es una respuesta positiva con etiqueta 1 si esta condición se cumple y una etiqueta 0 de lo contrario, no obstante, esta clase usualmente es nombrada por convención como la clase negativa), además como existe un desbalance de clases notorio, algunas métricas recomendadas en la literatura para medir el desempeño de los clasificadores especialmente en presencia de clases desbalanceadas son la precisión, recuperación, y el valor F_1, por lo tanto, siguiendo la formulación matemática presentada por Johnson y Khoshgoftaar (2019), a continuación se procede a definir cada métrica. Además se usará la exactitud para tener una noción adicional de qué tan bien predice el modelo en general para ambas clases.

En la ecuación 22 se observa que la precisión se expresa como el número de clasificaciones positivas correctas, o verdaderos positivos **VP** realizadas entre todos los valores predichos positivos por el modelo, es decir, sobre los verdaderos positivos **VP** agregado a los falsos positivos **FP**. Por otro lado, la recuperación o la sensibilidad se aprecia en la ecuación 23 y consiste en la cantidad de clasificaciones positivas correctas **VP** realizadas entre

todos los valores positivos verdaderos, que es la cantidad de los que el modelo acertó prediciendo como positivos **VP**, agregados a los que eran positivos pero que el modelo se equivocó, esto es, los falsos negativos **FN**. En otras palabras, la recuperación es una métrica que mide qué proporción recuperó el modelo de las observaciones que efectivamente eran de la clase positiva. En la ecuación 24 se muestra la métrica $\boldsymbol{F_1}$ que es la media harmónica entre la precisión y la recuperación. Su principal fortaleza consiste en buscar el equilibrio entre precisión y recuperación, puesto que generalmente, aumentar la precisión disminuirá la recuperación y viceversa. Por otro lado la exactitud es la cantidad de veces que el modelo acertó, sobre la cantidad total de observaciones a clasificar como se aprecia en la ecuación 25.

El motivo por el cual no se incluye la métrica conocida como el área bajo la curva ROC, se fundamenta, como explica Johnson y Khoshgoftaar (2019), en que esta métrica puede ofrecer resultados demasiado alentadores ya que si hay una cantidad considerable de elementos de la clase negativa clasificados correctamente, entonces la probabilidad de la falsa alarma puede verse reducida, lo que conlleva a una mejora en el valor de el área bajo la curva ROC. Sin embargo, para el contexto de este proyecto se requiere una métrica que de mayor importancia a la clasificación correcta de la clase positiva dado el desbalance que se tiene, por lo tanto, la métrica principal será $\boldsymbol{F_1}$,

$$Precision = \frac{VP}{VP + FP} \tag{22}$$

$$Recuperacion = \frac{VP}{VP + FN} \tag{23}$$

$$Valor\ F_1 = \frac{2 \times Recuperacion \times Precision}{Recuperacion + Precision} \tag{24}$$

$$Exactitud = \frac{VP + VN}{VP + FP + VN + FN} \tag{25}$$

$$\tag{26}$$

Estrategia para la evaluación

Para evitar que los modelos se ajusten para predecir únicamente la clase mayoritaria se van a probar varias técnicas que ayudarán a reducir o solucionar el desbalance de clases, por ejemplo, se asignarán distintos costos por la clasificación errónea de las clases en las funciones de pérdida de los distintos modelos, también se utilizará la técnica de sobremuestreo SMOTE, submustreo aleatorio y finalmente la combinación de estos dos últimos, es decir, primero se aplicará SMOTE hasta hasta lograr cierto porcentaje de la clase minoritaria y seguidamente se aplicará submuestreo aleatorio hasta lograr cierto porcentaje de la clase mayoritaria. Los porcentajes de la combinación serán tratados como hiperparámetros a optimizar.

Adicionalmente, para tener una mejor medición de la generalización de los modelos, se utiliza un enfoque de validación cruzada anidada con una validación cruzada interna para entrenar y encontrar buenos hiperparámetros de los modelos y de las técnicas que resuelven el desbalance (por ejemplo la matriz de costos para la función de pérdida del modelo, o la cantidad de vecinos más cercanos que usa SMOTE para crear una nueva observación), y una validación cruzada externa para evaluar la generalización de las diferentes las parejas de combinaciones creadas por los modelos y las técnicas que resuelven el desbalance para finalmente realizar la comparación del desempeño de los modelos. Esta metodología permitirá que el desbalance de clases solo se arregle en conjuntos de entrenamiento, sin embargo, el conjunto de prueba de la validación cruzada externa no se modifica con las técnicas para manejar el desbalance.

Este método de validación cruzada consiste en realizar una partición de los datos en k conjuntos de tamaño similar, escoger k-1 conjuntos para usar como un solo conjunto de entrenamiento y el conjunto restante como conjunto de prueba. A continuación, se procede a encontrar el mejor modelo para ese conjunto de entrenamiento y seguidamente se realiza la evaluación en los datos de prueba. Posteriormente, para finalizar una iteración del algoritmo se agregan las métricas obtenidas a una lista de resultados, luego se escogen otros k-1 conjuntos para realizar el entrenamiento y el restante como las pruebas hasta que cada uno de los k conjuntos haya actuado como conjunto de prueba al menos una vez. Al finalizar se tendrá una lista de k valores que permiten estimar qué tan bien generaliza el modelo.

Además, es llamado validación cruzada anidada porque para el proceso de encontrar el mejor modelo del conjunto de entrenamiento, se realiza nuevamente una validación cruzada interna, es decir, se crea una nueva división del conjunto de entrenamiento que particionó el ciclo de validación cruzada externo en k-1 conjuntos de entrenamiento y el restante se puede considerar como un conjunto de validación que va a permitir evaluar y escoger el mejor modelo. Esto se realiza con la finalidad de que al escoger el modelo con la mejor métrica (en este estudio en particular, con el mejor valor F_1), y que además ha escogido los mejores hiperparámetros no esté sobreajustado a los datos de prueba y por tanto, asegurar que la estimación de la generalización del modelo sea de muy buena calidad.

Un aspecto para resaltar es que la partición de los conjuntos se realiza usando una división estratificada, en otros términos, se asegura que la proporción de las clases se mantenga más o menos similar entre los k conjuntos que se obtienen, si bien cada conjunto de entrenamiento de la validación cruzada interna puede ser sometida a una técnica para arreglar el desbalance de clases. Adicionalmente, para optimizar los hiperparámetros se escoge una búsqueda aleatoria para explorar espacios de solución amplios dentro de una cantidad razonable de tiempo.

Específicamente para el ciclo de validación cruzada externa, se escogió usar 5 particiones (k=5), asimismo para el ciclo de validación cruzada interna que se usó para el ajuste de hiperparámetros, también se utilizó 5 particiones. Hay dos grandes razones detrás de la decisión de estos valores, la primera es asegurar que cada partición de prueba sea lo suficientemente grande para dar una buena evaluación de los modelos, puesto que con valores muy grandes de k se reduce el tamaño de cada partición, lo que en consecuencia puede llevar a que cada

partición no sea representativa.

La segunda razón, es una decisión por la complejidad computacional que implicaría el aumento de estos valores de k, esto es debido a que la validación cruzada anidada es muy exigente por la gran cantidad de modelos que se deben entrenar. Por ejemplo, suponga que el algoritmo de búsqueda aleatoria para los hiperparámetros, plantea 10 configuraciones candidatas para cierto modelo. Para saber cuál es la mejor configuración, el ciclo de validación cruzada interna empieza, por lo que con un k = 5, se deberá entrenar cada candidato 5 veces lo que conlleva un total de 50 entrenamientos del modelo. A continuación, se escoge el candidato con mejor desempeño promedio de la lista de resultados obtenida de la validación cruzada interna, al usar cada una de las k particiones como conjunto de validación una vez. Posteriormente, el mejor modelo se evalúa en el conjunto de prueba de esa iteración actual, y si se definieron 5 particiones para validación cruzada externa, este proceso deberá iterar cuatro veces más, es decir, se entrenarían un total de 250 modelos. Y para finalizar, si cada modelo en sí es costoso en tiempo computacional, como es el caso de LSTM, todo el proceso será bastante demorado. Por tal razón, si los valores que se escogen para realizar las particiones son muy altos, el costo computacional sería muy grande incluso usando programación concurrente. Adicionalmente, se decidió no llevar a cabo técnicas como selección de variables hacia adelante puesto que esto también incrementaría el coste computacional de gran manera. En su lugar se usarán las variables seleccionadas con el análisis hecho en la etapa exploratoria y también se comparará con las variables por importancia obtenidas con *random forest*.

Resultados

En los cuadros 6,7 se puede observar el promedio y la desviación estándar de las diferentes métricas obtenidas del proceso de validación cruzada anidada para cada una de las combinaciones entre los modelos con las diversas técnicas para resolver el desbalance de clases. En toda esta sección se llevará a cabo el análisis de estos resultados, la comparativa entre distintos modelos y también se darán a conocer algunas consideraciones de cada uno de los clasificadores que se entrenaron, a saber, *Dummy classifier*, regresión logística, *random forest*, *extreme boosting* y finalmente LSTM.

Modelo	Técnica de balanceo de datos	Exactitud	Precision	Recuperacion	Valor F_1
Dummy classifier	Sin balanceo	0.4973	0.2280	0.5072	0.3145
	SMOTE y Random Undersampling	0.5059	0.2333	0.5127	0.3206
	SMOTE	0.4978	0.2251	0.4947	0.3094
	Random Undersampling	0.4977	0.2274	0.5039	0.3134
Regresión logística	SMOTE y Random Undersampling	0.6178	0.3286	0.6464	0.4349
	SMOTE	0.6260	0.3322	0.6338	0.4354
	Random Undersampling	0.6201	0.3291	0.6409	0.4342
	Class-Weigths	0.6249	0.3318	0.6357	0.4353
Random forest	SMOTE y Random Undersampling	0.7073	0.3996	0.5632	0.4663
	SMOTE	0.7256	0.4129	0.4871	0.4468
	Random Undersampling	0.6750	0.3744	0.6397	0.4724
	Class-Weigths	0.6948	0.3942	0.6070	0.4755
Gradient tree boosting	SMOTE y Random Undersampling	0.6529	0.3550	0.6402	0.4563
	SMOTE	0.6373	0.3469	0.6729	0.4578
	Random Undersampling	0.6790	0.3763	0.6158	0.4657
	Class-Weigths	0.7144	0.4197	0.6122	0.4946
LSTM roles=4	SMOTE y Random Undersampling	0.6609	0.3294	0.6093	0.4271
	SMOTE	0.6943	0.3553	0.5778	0.4396
LSTM roles=3	Random Undersampling	0.6305	0.3223	0.6382	0.4240
LSTM roles=2	Class-Weigths	0.6260	0.3246	0.6437	0.4301

Cuadro 6: Media de las métricas de desempeño

Modelo	Técnica de balanceo de datos	Exactitud	Precision	Recuperacion	Valor F_1
Dummy classifier	Sin balanceo	0.0066	0.0065	0.0171	0.0094
	SMOTE y Random Undersampling	0.0100	0.0105	0.0253	0.0148
	SMOTE	0.0098	0.0092	0.0208	0.0127
	Random Undersampling	0.0125	0.0100	0.0211	0.0134
Regresión logística	SMOTE y Random Undersampling	0.0259	0.0136	0.0361	0.0044
	SMOTE	0.0222	0.0111	0.0334	0.0033
	Random Undersampling	0.0245	0.0107	0.0423	0.0023
	Class-Weigths	0.0244	0.0115	0.0391	0.0038
Random forest	SMOTE y Random Undersampling	0.0152	0.0148	0.0463	0.0105
	SMOTE	0.0090	0.0136	0.0138	0.0107
	Random Undersampling	0.0039	0.0059	0.0200	0.0097
	Class-Weigths	0.0189	0.0157	0.0361	0.0049
Gradient tree boosting	SMOTE y Random Undersampling	0.0185	0.0132	0.0314	0.0117
	SMOTE	0.0128	0.0122	0.0144	0.0137
	Random Undersampling	0.0234	0.0142	0.0578	0.0059
	Class-Weigths	0.0352	0.0319	0.0563	0.0113
LSTM roles=4	SMOTE y Random Undersampling	0.0196	0.0128	0.0331	0.0101
	SMOTE	0.0152	0.0137	0.0300	0.0114
LSTM roles=3	Random Undersampling	0.0561	0.0352	0.0800	0.0093
LSTM roles=2	Class-Weigths	0.0289	0.0108	0.0680	0.0103

Cuadro 7: Desviación estándar de las métricas de desempeño

Dummy classifier

No se tiene un precedente del valor de las métricas para un conjunto de datos suficientemente similar al construido en este estudio, por lo tanto, se usará un modelo conocido como *dummy classifier* para obtener métricas iniciales y así revisar que el desempeño de modelos más complejos si logren mejorar este valor. El modelo *dummy classifier* es conocido por realizar clasificaciones sin tener en cuenta las variables de entrada. Tiene dos estrategias de predicciones aleatorias, la primera es realizar predicciones aleatorias de manera estratificada, es decir, ignora las covariables y simplemente predice la clases de acuerdo a la distribución de las categorías en el conjunto de entrenamiento. En otras palabras, si en el conjunto de entrenamiento la clase 0 está asignada al 77 % de las observaciones, entonces este modelo va a predecir para cualquier entrada la clase 0 con una probabilidad del 77 %.

La segunda estrategia es realizar predicciones aleatorias usando la distribución uniforme. Otra posibilidad es predecir siempre la clase minoritaria, y aunque esto aumenta la métrica principal del valor $\boldsymbol{F_1}$, disminuye la exactitud a 0.22, por lo que se va a ignorar esta estrategia. Así pues, el único hiperparámetro para este modelo fue la estrategia de predicción (adicional a los hiperparámetros de las técnicas de balanceo), y la ganadora fue la

predicción uniforme, puesto que le da igual probabilidad de ser elegida a la clase minoritaria en contraste con la estrategia estratificada que le dará mayor probabilidad a la clase mayoritaria.

En el cuadro 6 el modelo de *dummy classifier* con las variantes de las distintas técnicas de balanceo de datos obtiene valores $\boldsymbol{F_1}$ cercanos a 0.32 y la métrica de exactitud apoya el hecho de que este es un modelo aleatorio al ser cercana a 0.5. Por consiguiente todos los modelos que se van a entrenar se pueden comparar con estas métricas para revisar si realmente tuvieron una mejora.

Random Forest

Para la técnica de *Random Forest* se usó la primera estructura de datos, sin embargo, se alimentó con todas las variables con la finalidad de revisar la importancia de estas y comparar estos resultados con la selección de variables realizadas a partir del análisis exploratorio. Se escogió realizar este análisis por importancia con *random forest* puesto que cada árbol de decisión que entrena se construye con una muestra con reemplazo que además solo incluye un subconjunto de las variables lo que puede lograr que no lo afecte tanto la multicolinealidad como sí afecta a otros modelos. Igualmente no fueron escaladas las variables puesto que las reglas de decisión se ajustan a los rangos de cada variable sin ofrecer mayor relevancia por tener distinta escala ya que la importancia se obtiene principalmente por la reducción de la impureza dada por el valor Gini.

En cuanto a los hiperparámetros, algunos de los que se agregaron para optimizar fueron el número de estimadores o árboles de decisión, la profundidad máxima del árbol (el número de nodos existentes en la ruta más larga desde el primer nodo hasta un nodo terminal), y el número máximo de características a tener en cuenta en cada muestra con reemplazo que se obtiene.

En relación con el cuadro 6 se evidencia que el *random forest* y todas las variantes de las distintas técnicas de balanceo tienen un mejor valor $\boldsymbol{F_1}$ y exactitud que *dummy classifier*, regresión logística y LSTM. Por otro lado, la variante de *random forest* con costos asociados a cada clase obtuvo un mejor valor $\boldsymbol{F_1}$ promedio y una menor desviación estándar de esta métrica que todas sus demás variantes con las distintas técnicas para solucionar el desbalance de clases, si bien los resultados son muy cercanos.

En la figura 26 se muestra el gráfico de precisión vs recuperación para el modelo de *random forest* con la técnica de balanceo de asignación de costos a cada clase, específicamente, en una de las iteraciones de la validación cruzada externa. Está gráfica permite observar la relación entre estas métricas si se cambia el umbral de decisión que con base en la probabilidad selecciona la clase. Si bien para todos los modelos se escogió el umbral de decisión que maximizaba el $\boldsymbol{F_1}$, se aprecia en esta figura que *random forest* tiene un poco de margen de maniobra para cambiar el umbral de decisión para ajustar un poco las métricas de recuperación y precisión si existiese alguna preferencia por alguna de las dos métricas, aunque tal determinación puede reducir un poco la métrica principal que es el valor $\boldsymbol{F_1}$.

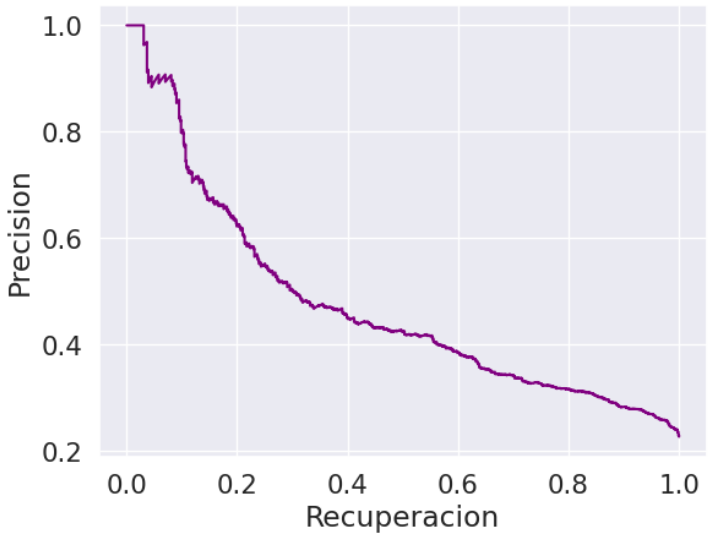

Figura 26: Diagrama de precision vs recuperación para *random forest*.

En la figura 27 se observa la matriz de confusión usando el mismo modelo que el de la gráfica previa. Se nota que se clasificó correctamente una gran cantidad de registros de empleados (2216) que se quedaron más de 12 meses en la empresa, lo que puede explica la alta exactitud que tiene el modelo a pesar de que las otras métricas son menores. Asimismo si bien se logra clasificar correctamente 466 empleados saliendo de su empresa, hay una mayor cantidad de falsas alarmas lo que lleva a que la precisión sea un poco menor a la recuperación.

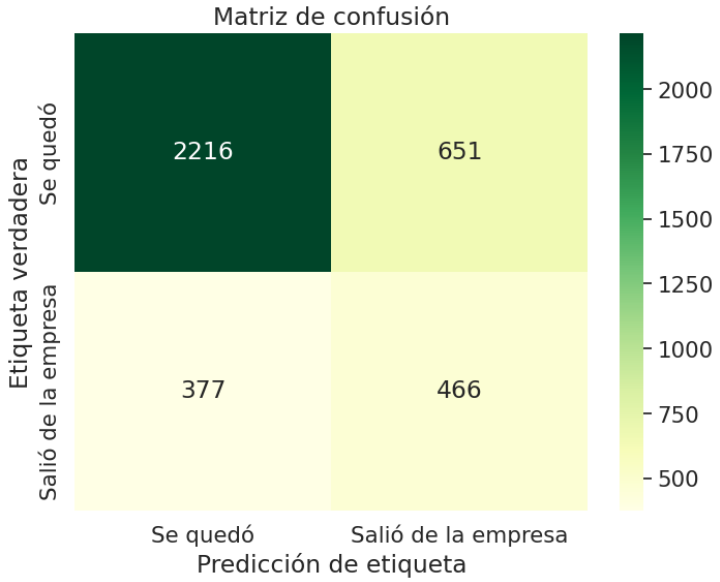

Figura 27: Matriz de confusión para *random forest*

La figura 28 muestra la disminución media en la impureza **(MDI)** que proporciona cada covariable en *random forest*, con la impureza siendo medida usando Gini. Entre mayor sea la disminución de la impureza se puede considerar que es una variable más relavante para lograr la separación de las clases en este modelo. Comparando con el análisis exploratorio realizado previamente se observa que ambos análisis coinciden en que la variable más importante es *months in company acum*. No obstante, hay una discrepancia y es que desde el análisis

exploratorio se marcaba la tasa de desempleo (*unemployment rate*) como la variable menos relevante, sin embargo, con *random forest* resultó ser el segundo atributo más influyente, y desde la teoría de la rotación de empleados tiene sentido que esta variable tenga un efecto por ser una medida que ayuda a comprender el contexto del mercado laboral en el que se encuentra el empleado, por tanto se toma la decisión de incluir la tasa de desempleo para el entrenamiento de los modelos restantes.

Por otro lado ambos análisis de selección de características concuerdan en que las variables de índice, el mes y el país son las que menos parecen tener buena capacidad predictiva para este estudio. Otra decisión que se tomó es que pese a que otras variables como *months experience* y *months in position* tenían un mayor disminución media en la impureza que otras, puesto que están correlacionadas con *months in company acum* se mantiene la elección de solo quedarse con esta última. En definitiva las covariables que se usarán para los demás modelos serán las siguientes: *months in company acum*, *unemployment rate*, *gdp* y *year*

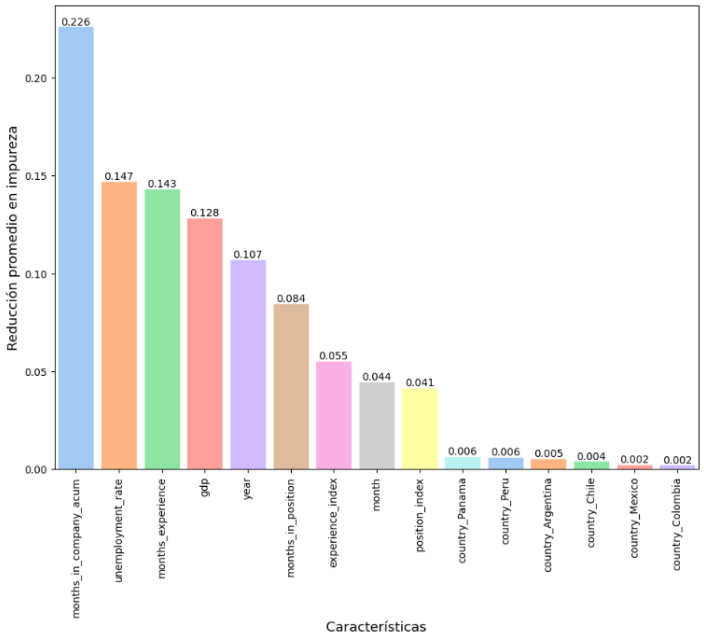

Figura 28: Importancia de las características para *random forest* usando MDI

Regresión logística

Para esta técnica se usó la primera estructura de datos y adicionalmente se escalaron las variables que no eran binarias para que tuvieran desviación estándar de 1 y media de 0 para que el modelo no considerasen más importante a las variables únicamente porque tienen una mayor escala. Por otro lado se optimizaron hiperparámetros como el método de optimización de la función de pérdida y el parámetro C que corresponde al inverso del factor de regularización, es decir, es un hiperparámetro que busca controlar los parámetros que se están aprendiendo y así evita que crezcan mucho y que el modelo se ajuste de manera perfecta a los datos de entrenamiento lo que causaría problemas al generalizar las predicciones a datos de prueba.

En el cuadro 6 se observa que para todos los modelos de regresión logística construidos, la media del

valor $\boldsymbol{F_1}$ es bastante mayor que la del modelo *dummy classifier* lo que demuestra que realmente se está logrando capturar algunos patrones en las covariables que permiten mejorar la predicción en comparación de realizar una predicción aleatoria. Para la regresión logística las distintas técnicas de balanceo de datos muestran un desempeño bastante similar con el valor medio de $\boldsymbol{F_1}$. En otro aspecto, como se aprecia en el cuadro 7 la regresión logística con SMOTE y *random under sampling* tuvo ligeramente mayor variabilidad en el valor $\boldsymbol{F_1}$.

Adicionalmente se observa que la métrica de recuperación fue favorecida, mientras que la precisión tiene un valor inferior, es decir, en general este modelo logra identificar una mayor cantidad de empleados que dejaran la empresa, sin embargo, a costo de tener una mayor cantidad de falsos positivos.

Tomando como base el modelo de regresión logística con la técnica de balanceo SMOTE para una de las iteraciones de la validación cruzada externa, en la figura 29 se observa su gráfico de precisión vs recuperación. en este gráfico se observa que la curva está muy cerca al eje x y es difícil escoger un umbral de decisión que permita lograr un valor más equilibrado de las métricas de precisión y recuperación, puesto que ello implicaría un menor bastante más reducido del valor $\boldsymbol{F_1}$ y este comportamiento se comparte entre todos los modelos de regresión logística con las distintas técnicas de balance de datos.

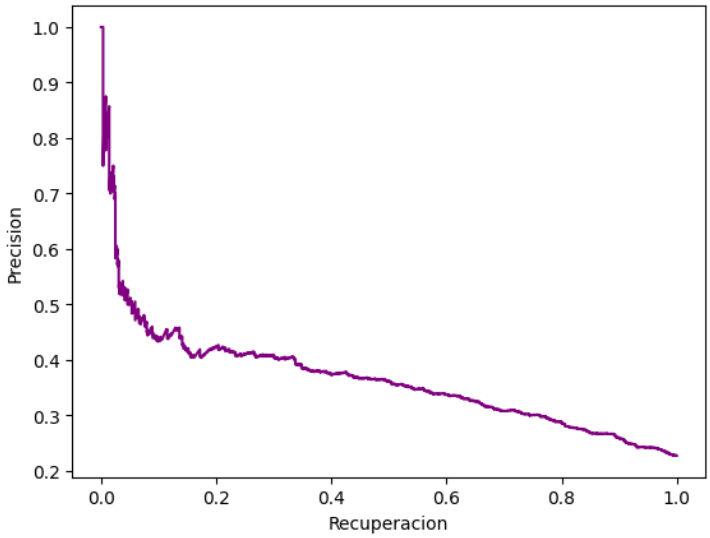

Figura 29: Diagrama de precision vs recuperación para la regresión logística.

En la figura 30 se observa la matriz de confusión usando el mismo modelo que en el gráfico anterior y se evidencia precisamente como el clasificador comete pocos errores cuando predice que el empleado se va a quedar en la empresa, pero bastantes errores cuando predice que saldrá de la misma. Además, el modelo de regresión logística comete aproximadamente el doble de falsos positivos que Random Forest. Si bien regresión logística tuvo mayor recuperación lo hizo sacrificando precisión, mientras *random forest* tiene un mejor balance y mayor fiabilidad.

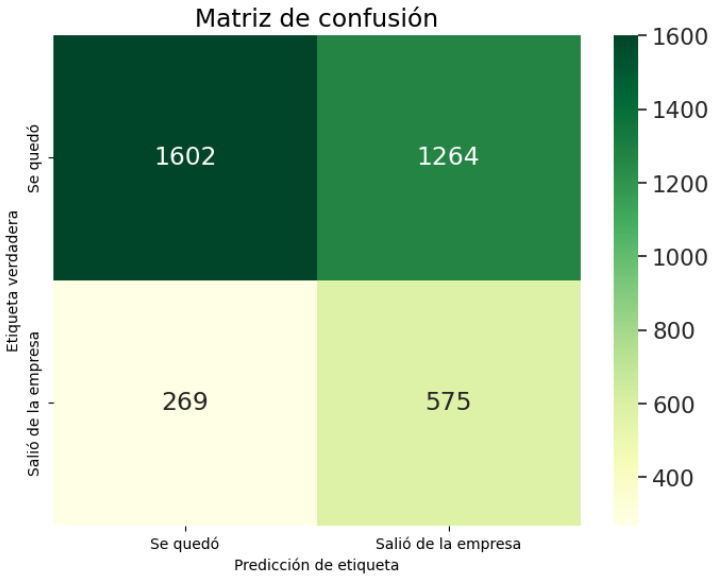

Figura 30: Matriz de confusión para regresión logística

Gradient tree boosting

Para esta técnica se usó la primera estructura de datos. Los hiperparámetros que se optimizaron fueron el número de árboles de decisión, la tasa de aprendizaje que regula la contribución de cada nuevo árbol a la predicción total lo cual ayuda a reducir el sobreajuste a los datos de entrenamiento. Adicionalmente se usó la profundidad máxima y el número máximo de nodos permitidos para cada árbol de decisión.

En el cuadro 6 se aprecia que el *gradient tree boosting* y todas sus variantes con las distintas técnicas de balanceo tienen un mejor valor $\boldsymbol{F_1}$ y exactitud que *dummy classifier*, regresión logística y LSTM. Por otro lado estos modelos obtienen un desempeño más cercano al de *random forest*. No obstante la variante de *extreme boosting* con *class weigths* obtuvo un mejor promedio de valor $\boldsymbol{F_1}$ que todos los demás modelos superando al *random forest*, aunque tuvo una de las variabilidades más altas como se puede comprobar en el cuadro 7. Además, *gradient tree boosting* logra un mejor equilibrio de las métricas de precisión y recuperación que con otros modelos.

A continuación en la figura 31 se muestra el gráfico de precisión vs recuperación para el modelo de *gradient tree boosting* con la técnica de balanceo de asignación de pesos a las clasificaciones, específicamente, en una de las iteraciones de la validación cruzada externa. En este gráfico se observa que la curva está más alejado del eje x en comparación con el gráfico para regresión logística, *dummy classifier* y LSTM. Se podría cambiar el umbral de decisión un poco dependiendo de los requerimientos que pueda preferir recursos humanos. Sin embargo, el valor $\boldsymbol{F_1}$ presentado en el cuadro 6 es el máximo por lo que el cambio en el umbral de decisión puede reducir los números en esta métrica.

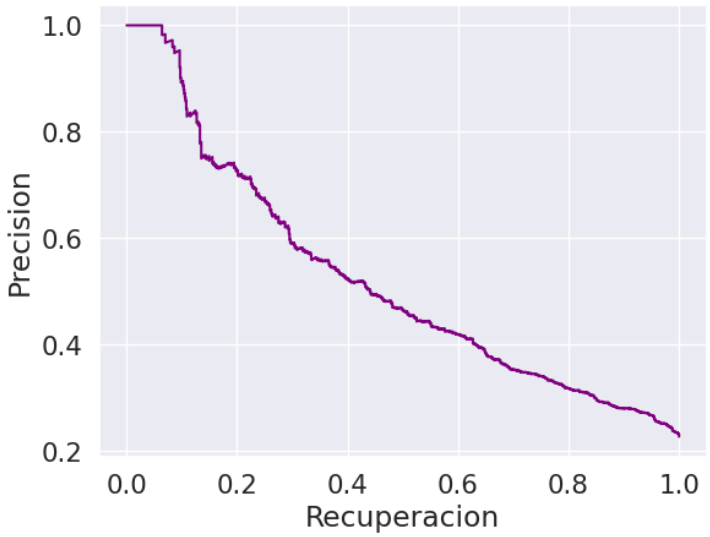

Figura 31: Diagrama de precision vs recuperación para *gradient tree boosting*.

En la figura 32 se observa la matriz de confusión usando el mismo modelo que el de la gráfica previa. Se evidencia que los falsos positivos son más frecuentes que los falsos negativos. Además se resalta que este modelo posee una menor cantidad de errores que otros modelos y logra clasificar correctamente que 529 empleados salieron de la empresa en menos de un año, mientras clasifica correctamente 2109 del total que se quedaron.

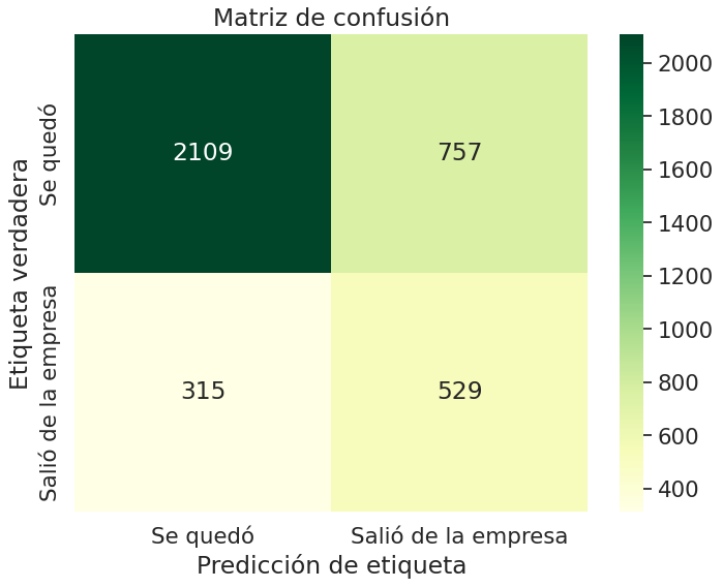

Figura 32: Matriz de confusión para *gradient tree boosting*

LSTM

Para esta técnica se usó la segunda estructura de datos y además se escalaron las variables que no eran binarias para que tuvieran desviación estándar de 1 y media de 0. Se decidió escalarlas debido a que el algoritmo de *LSTM* se optimiza usando gradientes que se pueden volver inestables, es decir, si la escala de las variables es distinta puede ocurrir que no se minimice el error de manera suave a una buena solución, si no que se puede

cambiar abruptamente entre soluciones en cada iteración del algoritmo lo que dificulta todo el proceso e incluso puede retardar la convergencia del mismo.

En cuanto a los hiperparámetros se contempló la cantidad de neuronas, el número de capas (la cantidad de componentes LSTM conectados entre sí), la tasa de aprendizaje para modificar la fuerza con la que se reducían los errores en cada iteración del algoritmo, el método de optimización de la función de pérdida, el número de neuronas de un perceptrón que podría ser o no agregado para procesar la salida del LSTM y obtener la función para la salida de la arquitectura neuronal. Además, se agregó un hiperparámetro conocido como la cantidad de épocas que es el número de veces que el modelo recorre todo el conjunto de entrenamiento mientras busca reducir la función de pérdida.

Adicionalmente se debió escoger el número de roles del pasado que se usarían para predecir la variable respuesta del presente. El valor de este hiperparámetro implica un cambio en la estructura del conjunto de datos para que tenga las dimensiones apropiadas. Usualmente el conjunto de datos se organiza en una matriz tridimensional con el formato $número\ de\ observaciones\ \times\ número\ de\ roles\ del\ pasado\ \times\ número\ de\ covariables$. Dependiendo de la cantidad de roles escogida, se construyen las secuencias que componen el conjunto de datos, por tal motivo, este hiperparámetro no se podía incluir dentro del proceso de validación cruzada

En consecuencia, para cada valor de este hiperparámetro, se llevó a cabo el proceso de validación cruzada anidada con las diferentes técnicas de balanceo de datos. Sin embargo, para facilitar la visualización de los resultados, en los cuadros 6 y 7 se incluyeron únicamente los mejores resultados de LSTM para cada una de las técnicas que resuelven el desbalance de clase. Adicionalmente se determinó variar la cantidad de roles entre dos y cinco para no excluir una gran cantidad de registros de empleados (para mayor información sobre por qué se excluyen empleados al construir las secuencias, revisar el apartado de la estructuración de los datos en la metodología).

En el cuadro 6 se observa los promedios de validación cruzada anidada con LSTM para cada una de las técnicas de solución al desbalance de clases. Debajo de LSTM aparece el número de roles que se usaron para organizar las secuencias de datos. Por ejemplo, el mejor modelo de LSTM con SMOTE y Random Undersampling fue aquel que se entrenó con cuatro roles en el pasado para predecir el quinto. Se observa que las dos primeras variantes usan cuatro roles, mientras que la tercera usa tres roles y por último, la cuarta usó dos roles. Se aprecia que las cuatro variantes presentadas tienen un mejor valor $\boldsymbol{F_1}$ y exactitud que *dummy classifier* lo cual evidencia que este modelo pudo superar las métricas base y pudo capturar algunos patrones a partir de las covariables para lograr la predicción. Por otro lado, al ser comparado con regresión logística, únicamente la variante de LSTM con SMOTE consiguió tener un mejor valor medio de $\boldsymbol{F_1}$ que las demás variantes de la regresión logística, no obstante, LSTM posee una mayor variabilidad como se puede examinar en el cuadro 7. Con relación a la exactitud, todas las variantes de LSTM tienen un mayor valor medio que en la regresión logística, por lo que el mejor modelo de LSTM se podría escoger razonablemente con base en el desempeño en los conjuntos de prueba sobre el mejor de regresión logística teniendo en cuenta las métricas de exactitud y valor $\boldsymbol{F_1}$.

Por el contrario, los mejores modelos de *random forest* y *gradient tree boosting* (los que tienen un mayor

valor $\boldsymbol{F_1}$), superan al mejor de LSTM en exactitud y valor $\boldsymbol{F_1}$. Asimismo, respecto al tiempo computacional, todos las demás técnicas fueron entrenadas y evaluadas en un menor tiempo, lo que puede suponer un menor gasto de recursos en un proyecto aplicado a empresa.

A continuación, en la figura 33 se muestra el gráfico de precisión vs recuperación para el modelo de *LSTM* con la técnica de balanceo de SMOTE, específicamente, en una de las iteraciones de la validación cruzada externa. El gráfico que se observa es similar al que corresponde a el de regresión logística, y de igual manera se podría cambiar el umbral de decisión, por ejemplo, se podría modificar de tal modo que la recuperación y la precisión tengan un valor cercano a 0.4. Sin embargo, esto reduciría también el valor $\boldsymbol{F_1}$ notablemente, por lo que teniendo la alternativa de *gradient tree boosting* cuyas métricas de precisión y recuperación están más equilibradas no sería muy ventajoso el cambio en el umbral de decisión para LSTM.

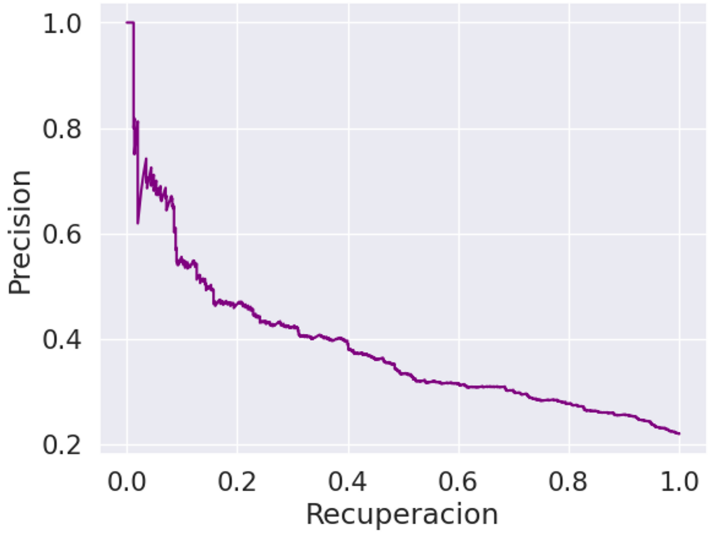

Figura 33: Diagrama de precision vs recuperación para *LSTM*

En la figura 34 se observa la matriz de confusión usando el mismo modelo que el de la gráfica previa. Se evidencia que los falsos positivos son más frecuentes que los falsos negativos y los verdaderos positivos. Se nota un modelo que de manera equilibrada trata de clasificar a ambas clases, esto se evidencia al sumar la cantidad de observaciones que el modelo asignó a cada clase, pero con dificultades para diferenciar entre ambas y más aún para identificar la clase positiva.

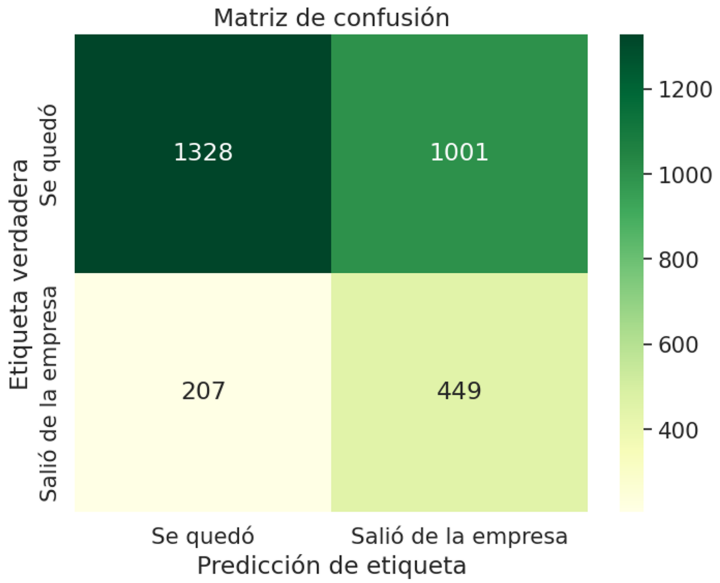

Figura 34: Matriz de confusión para LSTM

Conclusiones

Tras el análisis de los resultados de las métricas de evaluación se puede concluir que los modelos entrenados con regresión logística *random forest*, *gradient tree boosting* y *LSTM* sobrepasaron el modelo aleatorio que se tuvo como modelo base (*dummy classifier*) y que se propuso dado que el conjunto de datos fue construido como parte de este proyecto. En cuánto a las técnicas para solucionar el desbalance de las clases, no hubo una técnica que fuera superior para todos los modelos, pero en general, la técnica de asignación de costos por clasificación errónea dio buenos resultados siendo la primer o segunda técnica con mejor desempeño en cada tipo de modelo. Adicionalmente, el modelo con el mejor resultado de todos fue *gradient tree boosting* con asignación de costos por clasificación errónea cuyo valor $\boldsymbol{F_1}$ es aproximadamente 17 puntos porcentuales mayor que el de *dummy classifier*.

Se obtienen resultados bastante positivos y un aspecto adicional a considerar es que estos modelos obtenidos en esta investigación no solo se limitan a modelar la rotación de empleados de una única empresa, por el contrario, permiten realizar predicciones de la rotación de empleados de distintas empresas, debido al hecho de que los registros pertenecen a diversas compañías del mundo con empleados que residían en América Latina.

Pese a que para este estudio se tenían algunas limitantes en cuánto a la información que no se poseía de los empleados, como el salario, la satisfacción o la edad de los trabajadores, se consiguió extraer información de múltiples empleados en distintas compañías a través del uso de web scrapping para recolectar los datos disponibles en la red social de empleados. Seguidamente se logró construir dos estructuras de datos para el entrenamiento de diferentes modelos y se fusionó con variables exógenas como el producto interno bruto y la tasa de desempleo. Para este estudio se encontró además que dos de las variables más importantes fueron la cantidad de los meses en la compañía acumulados hasta el momento para cada rol (*months in company acum*) y la tasa de desempleo como

se pudo observar en el análisis por importancia de *random forest*. Mientras que las variables menos relevantes fueron el país de origen y el mes en el que finalizó cada rol.

Por otro lado, siguiendo la recomendación de los investigadores se evaluó el desempeño de un modelo que pudiera manejar algún tipo de información temporal del empleado. En este proyecto se contaba con múltiples roles del pasado que cada empleado había desempeñado en su vida laboral. Para ello se escogió la técnica de LSTM, que buscaba entrenar un modelo con la capacidad de procesar las secuencias creadas por la información de los roles de los empleados, sin embargo, sus resultados estuvieron alejados de los modelos de ensamble de árboles.

Hay varias causas que pueden explicar el motivo por el que LSTM tiene una menor calidad en comparación con otras técnicas. La primera es que, LSTM no pudo captar alguna relación temporal de las secuencias que fuera suficientemente relevante para permitirle superar a todos los demás modelos, esto va en concordancia con el análisis de dependencias realizado en la etapa exploratoria y que se puedo apreciar en las figuras 21, 22 dónde no se pudo hallar un patrón que pudiese marcar la diferencia entre usar solo un rol para predecir el pasado, o el uso de varios. Mientras la figura 23 mostraba que para la predicción de un rol en el futuro, el unemployment rate podría tener cierto efecto, esto no fue suficiente para que LSTM lograra tener un desempeño superior al desempeño de los modelos restantes.

Desde otro punto de vista, en este estudio no se pudo realizar encuestas a los empleados de las empresas para obtener más atributos relevantes y que han sido recomendados en la literatura. Por lo tanto, incluso aunque la mayoría de las variables adquirían diferentes valores en el tiempo, esto no fue suficiente. Adicionalmente, era posible observar en los datos que un empleado podía tener roles con períodos de tiempo extensos en el mismo rol, lo que implica que la regularidad con la que se actualizaba la información se reducía. Para ilustrarlo, considere un empleado cuya experiencia anterior a la actual tuvo una duración de 10 años. Bajo los datos disponibles para este estudio no se contaba con muchas variables de la misma experiencia que pudieran reflejar con mayor precisión el cambio en el tiempo de su situación en la empresa. Por ejemplo, a través del salario o la satisfacción anual del empleado en la compañía, lo que para futuras investigaciones sería interesante revisar con redes neuronales recurrentes para descubrir si estas variables que cambian con mayor regularidad pueden ofrecer mejores relaciones temporales que puedan mejorar el desempeño del modelo.

Comparando el mejor modelo que fue *gradient tree boosting* con LSTM se llega a varias apreciaciones. Lo primero a resaltar evidentemente es la capacidad que tuvo el modelo de ensamble para generalizar de mejor manera a datos no vistos y clasificar correctamente manteniendo un mejor equilibrio en métricas como la recuperación y precisión en comparación con el modelo de LSTM que tuvo una menor precisión. Otro aspecto fue el tiempo computacional en el que *gradient tree boosting*, a pesar de ser un modelo iterativo con los árboles de decisión combinados de manera secuencial, fue considerablemente más rápido que LSTM. Esto se debe a que LSTM en general tuvo una mayor cantidad de parámetros entrenables por todos los componentes que constituían la red neuronal recurrente, como las parámetros de las compuertas que regulaban el flujo de información, o las

conexiones *peephole* que permitían a otros componentes conocer el estado de la memoria y que incrementaba a medida que se agregaban más neuronas y se combinaban con otras capas como con el perceptrón para procesar la salida del LSTM.

No obstante, de haber existido mayor cantidad de registros y de variables más relevantes que cambiaran de valor a través del tiempo y que tuvieran un efecto más notorio en la clase final del futuro, probablemente LSTM podría haber tenido un mejor de desempeño y habría podido competir de mejor manera con el modelo de *gradient tree boosting* que no se diseñó exclusivamente para aprovechar las bondades y la riqueza de las dependencias temporales existentes en secuencias de datos como sí es el enfoque principal de LSTM.

Finalmente, se usó un conjunto de datos de tamaño mediano para realizar la predicción para cualquier empleado que se pudiera obtener de diferentes países y tipos de funciones. Debido a la gran variedad de empleados, las métricas de desempeño todavía permiten espacio a la mejora. Esto trae una nueva idea que se recomienda tomar en cuenta para futuras investigaciones: sería interesante analizar si tener empleados similares es mejor que tener empleados con grandes diferencias, para ilustrarlo, en nuestro caso los modelo pudieron haber usado información de un rol bastante distinto, como un empleado que trabaja para el ejército, para predecir si un ingeniero de sistemas se iría en menos de un año, lo cual puede generar dificultades para que los modelos generalicen apropiadamente. Por consiguiente, centrarse en trabajos estrechamente relacionados podría ser un buen enfoque para abordar el problema en el futuro.

Agradecimientos

Agradecimientos especiales para mi madre que siempre me apoyó y creyó en mí a pesar de todas las dificultades, a la universidad EAFIT por darme la oportunidad y apoyo económico para estudiar esta maestría, y además por las correcciones sugeridas para este trabajo que han permitido mejorar la calidad del mismo, a mi directora Biviana Marcerla Suarez que me enseñó muchos conceptos estadísticos necesarios para comprender las diferentes técnicas empleadas en este proyecto y por la confianza que depositó en mí para que no me rindiera nunca, a mi codirector Juan Pablo Román que me ayudó a entender y estructurar la teoría necesaria para la rotación de personal y a Nicolás Alberto Moreno por darme muchos consejos para resolver los diferentes problemas y aclarar muchos conceptos.

Referencias

Alboukaey, N., Joukhadar, A., & Ghneim, N. (2020). Dynamic behavior based churn prediction in mobile telecom. *Expert Systems with Applications*, *162*, 113779. https://doi.org/10.1016/j.eswa.2020.113779

Bartunek, J. M., Huang, Z., & Walsh, I. J. (2008). The development of a process model of collective turnover. *Human Relations*, *61*(1), 5-38. https://doi.org/10.1177/0018726707085944

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*(2), 123-140. https://doi.org/10.1007/BF00058655

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5-32. https://doi.org/10.1023/A:1010933404324

Bro, S. (2022). Ninth Circuit once again preserves competitor's data-scraping rights. *The National Law Review*. https://www.natlawreview.com/article/ninth-circuit-once-again-preserves-competitor-s-data-scraping-rights

Cai, X., Shang, J., Jin, Z., Liu, F., Qiang, B., Xie, W., & Zhao, L. (2020). DBGE: Employee Turnover Prediction Based on Dynamic Bipartite Graph Embedding. *IEEE Access*, *8*, 10390-10402. https://doi.org/10.1109/ACCESS.2020.2965544

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System, 785-794. https://doi.org/10.1145/2939672.2939785

Elman, J. L. (1990). Finding structure in time. *Cognitive Science*, *14*(2), 179-211. https://doi.org/10.1016/0364-0213(90)90002-E

Felps, W., Mitchell, T. R., Herman, D. R., Lee, T. W., Holtom, B. C., & Harman, W. S. (2009). Turnover Contagion: How Coworkers' Job Embeddedness and Job Search Behaviors Influence Quitting. *The Academy of Management Journal*, *52*(3), 545-561. http://www.jstor.org/stable/40390303

Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, *29*(5), 1189-1232. Recuperado de el 20 de marzo de 2023, de http://www.jstor.org/stable/2699986

Friedman, J. H., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, *33*(1), 1-22. https://doi.org/10.18637/jss.v033.i01

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(4), 463-484. https://doi.org/10.1109/TSMCC.2011.2161285

Garg, S., Sinha, S., Kar, A. K., & Mani, M. (2022). A review of machine learning applications in human resource management. *International Journal of Productivity and Performance Management*, *71*(5), 1590-1610. https://doi.org/10.1108/IJPPM-08-2020-0427

Gers, F., Schraudolph, N., & Schmidhuber, J. (2002). Learning Precise Timing with LSTM Recurrent Networks. *Journal of Machine Learning Research*, *3*, 115-143. https://www.jmlr.org/papers/volume3/gers02a/gers02a.pdf

Ghorbani, H. (2019). MAHALANOBIS DISTANCE AND ITS APPLICATION FOR DETECTING MULTIVARIATE OUTLIERS. *FACTA UNIVERSITATIS-SERIES MATHEMATICS AND INFORMATICS*, *34*(3), 583-595. https://doi.org/10.22190/FUMI1903583G

Gómez, L. V. (2002). *La Crisis Argentina: Lecciones para Colombia* (inf. téc.). Recuperado de el 15 de abril de 2023, de https://www.banrep.gov.co/sites/default/files/publicaciones/archivos/lca.pdf

Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, *73*, 220-239. https://doi.org/https://doi.org/10.1016/j.eswa.2016.12.035

Hancock, J. I., Allen, D. G., Bosco, F. A., McDaniel, K. R., & Pierce, C. A. (2013). Meta-Analytic Review of Employee Turnover as a Predictor of Firm Performance. *Journal of Management*, *39*(3), 573-603. https://doi.org/10.1177/0149206311424943

Holtom, B. C., Mitchell, T. R., Lee, T. W., & Eberly, M. B. (2008). Turnover and retention research: A glance at the past, a closer review of the present, and a venture into the future. *The Academy of Management Annals*, *2*(1), 231-274. https://doi.org/10.1080/19416520802211552

Holtom, B. C., Tidd, S. T., Mitchell, T. R., & Lee, T. W. (2013). A demonstration of the importance of temporal considerations in the prediction of newcomer turnover. *Human Relations*, *66*(10), 1337-1352. https://doi.org/10.1177/0018726713477459

Hom, P. W., Lee, T. W., Shaw, J. D., & Hausknecht, J. P. (2017). One hundred years of employee turnover theory and research. *Journal of Applied Psychology*, *102*(3), 530-545. https://doi.org/10.1037/apl0000103

Jain, P. K., Jain, M., & Pamula, R. (2020). Explaining and predicting employees' attrition: a machine learning approach. *SN Applied Sciences*, *2*(757). https://doi.org/10.1007/s42452-020-2519-4

Johnson, J. M., & Khoshgoftaar, T. M. (2019). Survey on deep learning with class imbalance. *Journal of Big Data*, *6*(1), 27. https://doi.org/10.1186/s40537-019-0192-5

Lara-Benıtez, P., Carranza-Garcıa, M., & Riquelme, J. C. (2021). An Experimental Review on Deep Learning Architectures for Time Series Forecasting. *International Journal of Neural Systems*, *31*(03), 2130001. https://doi.org/10.1142/s0129065721300011

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436-444. https://doi.org/10.1038/nature14539

Ledoit, O., & Wolf, M. (2004). Honey, I Shrunk the Sample Covariance Matrix. *The Journal of Portfolio Management*, *30*(4), 110-119. https://doi.org/10.3905/jpm.2004.110

Lee, T., Hom, P., Eberly, M., Li, J., & Mitchell, T. (2017). On The Next Decade of Research in Voluntary Employee Turnover. *The Academy of Management Perspectives*, *31*, amp.2016.0123. https://doi.org/10.5465/amp.2016.0123

Loh, W.-Y. (2011). Classification and regression trees. *WIREs Data Mining and Knowledge Discovery*, *1*(1), 14-23. https://doi.org/10.1002/widm.8

Nandialath, A. M., David, E., Das, D., & Mohan, R. (2018). Modeling the determinants of turnover intentions: a Bayesian approach. *Evidence-based HRM: a Global Forum for Empirical Scholarship*, *6*(1), 2-24. https://doi.org/10.1108/EBHRM-10-2016-0025

Rombaut, E., & Guerry, M.-A. (2018). Predicting voluntary turnover through human resources database analysis. *Management Research Review*, *41*(1), 96-112. https://doi.org/10.1108/MRR-04-2017-0098

Rubenstein, A., Eberly, M., Lee, T., & Mitchell, T. (2017). Surveying the forest: A meta-analysis, moderator investigation, and future-oriented discussion of the antecedents of voluntary employee turnover. *Personnel Psychology*, *71*. https://doi.org/10.1111/peps.12226

Rutkowski, L., Jaworski, M., Pietruczuk, L., & Duda, P. (2014). The CART decision tree for mining data streams. *INFORMATION SCIENCES*, *266*, 1-15. https://doi.org/10.1016/j.ins.2013.12.060

Saradhi, V. V., & Palshikar, G. K. (2011). Employee churn prediction. *Expert Systems with Applications*, *38*(3), 1999-2006. https://doi.org/10.1016/j.eswa.2010.07.134

Van Houdt, G., Mosquera, C., & Nápoles, G. (2020). A review on the long short-term memory model. *Artificial Intelligence Review*, *53*(8), 5929-5955. https://doi.org/10.1007/s10462-020-09838-1