Estrategias de aprendizaje supervisado para un problema de renuncia en una agencia de marketing.

Ana Ospina, Alejandra Aguirre, Daniel Manco, Esteban Arcila

Departamento de Ingeniera Industrial
Universidad de Antioquia
Medellín, Colombia

INTRODUCCIÓN

La renuncia o abandono de puesto de un trabajador en una empresa puede estar ligado a diferentes motivos como mal entrono laboral, mala remuneración, poca posibilidad para crecer o una simple falta de reconocimiento por su esfuerzo.

Muchas empresas optan por diferentes estrategias para mitigar la renuncia de sus trabajadores y así tener una gestión en los recursos humanos ideal para cumplir con los objetivos de la organización. Uno de estas estrategias es hacer un estudio tanto cualitativo como cuantitativo acerca de todas las posibles causas que lleven al empelado a abandonar la organización

El objetivo de este estudio es analizar las posibles causas por las cuáles el 15% de los empleados de la agencia de marketing Sterling Cooper Advertising están renunciando cada año debido a que esto es perjudicial para la empresa por diferentes razones como:

-Los proyectos anteriores de los empleados antiguos se están retrasando, lo que afecta a los plazos y daña la reputación con clientes y socios.

-El departamento de recursos humanos enfrenta altos niveles de rotación, lo que requiere una inversión significativa en reclutamiento de nuevo talento, ralentizando otras áreas como formación y bienestar.

-La dirección quiere predecir y prevenir el abandono de empleados, por lo que ha contratado consultores para identificar factores clave y tomar acciones preventivas de retención. También buscan determinar cuál de estos factores es el más crítico y necesita atención inmediata.

MATERIALES Y MÉTODOS

Para el desarrollo de este trabajo, se aplicó estadística descriptiva y modelos de machine learning a través de herramientas de programación. Se aplicó un tratamiento sistemático a las bases de datos con el objetivo de limpiar, transformar y visualizar bajo el siguiente proceso.

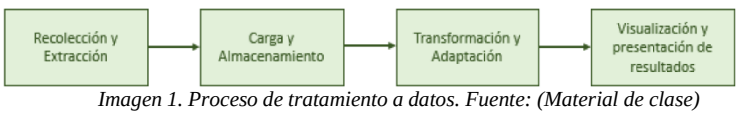

Imagen 1. Proceso de tratamiento a datos. Fuente: (Material de clase)

A. *Bases de datos*

La información necesaria para el proyecto se presenta en bases de datos. A continuación, se describirá cada una de las bases de datos trabajadas y su contenido:

- **general_data:** Información general del empleado.
- **employee_survey_data:** resultados de encuesta realizada a los empleados respecto a su nivel de satisfacción con su empleo actual.
- **manager_survey_data:** resultados obtenidos por los empleados en su última evaluación de desempeño.
- **time_work:** contiene el tiempo promedio de dedicación del empleado al día.

B. *Herramientas de programación*

Para aplicar cada una de las fases de tratamiento de datos y modelado, se dispuso de una de las herramientas de programación y consulta más conocida. A continuación, se describirán las herramientas de programación y que funciones de estas se utilizaron.

- Python: Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). [1]
- Una de las principales funciones o librerías utilizadas, además de las librerías para manipulación de datos numéricos y gráficos, de este lenguaje, fue *Pandas*. Es una librería de Python especializada en el manejo y análisis de estructuras de datos. [2]
- Otra de las librerías utilizadas fue *Sklearn* para todo el tema de machine learning, Esta librería proporciona una amplia variedad de algoritmos y herramientas para tareas como clasificación, regresión, agrupación, selección de características, reducción de dimensionalidad y más. Además, ofrece utilidades para la evaluación y validación de modelos. [3]

C. *Imputación*

Para realizar cualquier modelo lo primero que se hizo fue un análisis exploratorio de los datos donde lo que se pretende es hacer limpieza e imputación a su vez que se caracterizan las variables para que sean compatibles con todos los algoritmos que se van a utilizar posteriormente.

El primer paso fue identificar datos nulos en cada una de las bases de datos para posteriormente utilizar métodos de rellenado adecuados para cada tipo de variable, por ejemplo, rellenado con la mediana o la media de los datos.

Luego de realizar la limpieza a las 4 bases de datos se procede a juntarlas por medio de una función *merge* y así trabajar con un solo dataframe llamado *gdg*.

En el siguiente paso se procede a graficar las variables tanto categóricas como numéricas para observar su comportamiento de manera general.

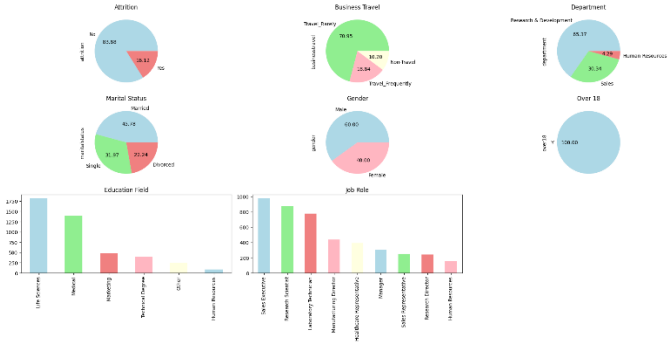

Imagen 1. Barras y tortas para variables categóricas – Elaboración propia

Se puede notar que, en uno de los gráficos de torta, la totalidad del mismo está dada por un solo valor lo que significa que la variable (Over18), toma el mismo valor en cada una de sus filas y se opta por eliminarla de la base de datos ya que se cree que no va a representar ninguna importancia dentro de los modelos posteriores.

Imagen 2. Histogramas para variables numéricas – Elaboración propia

Lo que se puede notar es que hay variables que presentan una distribución completamente uniforme dentro de un solo valor, lo que indica que toma el mismo valor en cada una de sus filas y se opta por eliminar las que se comporten de esta manera (employeecount y standarhours).

Finalmente se grafica la variable objetivo (atrittion) para saber cómo se distribuyen sus datos.

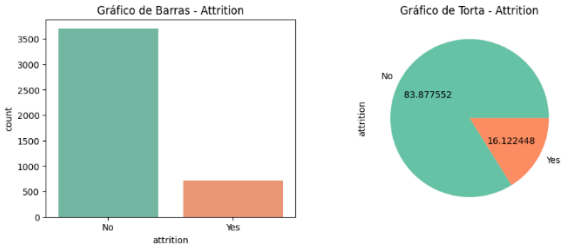

Imagen 3. Barras y torta para variable objetivo – Elaboración propia

Se puede notar que se presenta el No en una mayor proporción con aproximadamente un 84%, mientras que las personas que Sí renunciaron el último año tienen un porcentaje de 16%. Esto es muy importante al momento de crear el modelo.

D. Selección de variables

Lo primero que se hace es separar la variable objetivo (atrittion) del resto de variables que son las predictoras.

Después de asignar la variable objetivo, empezamos con la transformación, análisis y almacenamiento de las variables predictoras, las cuales estarán seleccionadas según el tipo de dato, ya sean object, numéricas o float.

- **Variables numéricas iniciales**: age, distancefromhome, education, joblevel, monthlyincome, percentsalaryhike, stockoptionlevel, traingintimelastyear, yearsatcompany, yearssincelastpromotion, yearswithcurrmanager, jobinvolvement, performancerating.

Luego de analizar estas variables por medio de histogramas, se llegó a la conclusión de que hay algunas que pueden ser de naturaleza categórica por lo que se procede a codificarlas y agregarlas en una base aparte de variables categóricas. Estas variables fueron: stockoptionlevel, performancerating, jobinvolvement, joblevel y education.

Ahora se procede a analizar las variables tipo *object*.

- **Variables object iniciales:** businesstravel, department, educationfield, gender, jobrole, maritalstatus.

Para este conjunto de variables se realizaron gráficos de torta donde se encontró que 4 variables se comportan como categóricas por lo que también se agregan a la base de datos aparte. Las variables fueron: maritalstatus, gender, department y businesstravel.

Finalmente se analizan las variables tipo *float.*

- **Variables float iniciales:** numcompaniesworked, totalworkingyears, enviromentsatisfaction, jobsatisfaction, worklifebalance, mean_time.

En este caso se encontraron 3 variables de naturaleza categórica. Las variables son: enviromentsatisfaction, jobsatisfaction y worklifebalance.

Finalmente se convierten en *dummies* todas aquellas variables que se identificaron como tipo categóricas ya que los modelos

sólo admiten variables numéricas.

El último paso antes de modelar es unir las variables *float*, *object* e *int* en una sola base de datos totalmente limpia que es con la que se va a trabajar de ahora en adelante. Esta base se llama *X_total*.

RESULTADOS Y ANÁLISIS DE MODELOS

A continuación, se mostrarán los resultados de los modelos realizados.

A. *Modelo de regresión logística (RL)*

El primer paso es separar el conjunto de datos de entrenamiento y testeo en donde se obtuvieron los siguientes resultados:

```
Tamaño del conjunto de entrenamiento. X:  (3528, 34)  Y:  (3528,)
Tamaño del conjunto de validación. X:  (882, 34)  Y:  (882,)
```

Imagen 3. Tamaño de conjuntos del modelo de RL – Elaboración propia

Luego se procede a realizar el modelo y su respectiva matriz de confusión.

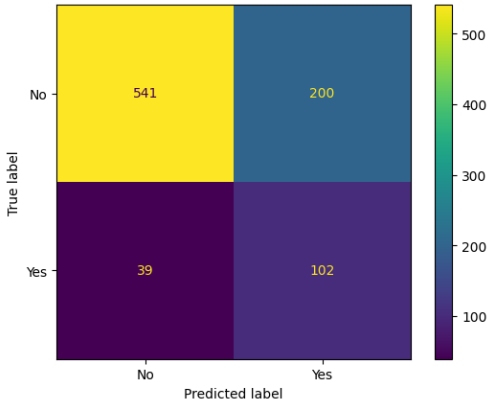

Imagen 4. Matriz de confusión modelo de RL – Elaboración propia

Cuando se analiza la matriz de confusión se puede notar que los falsos positivos son muy altos y son los que determinan la poca calidad del modelo. Estos falsos positivos lo que dicen es que el modelo no está logrando predecir correctamente las personas que de verdad NO van a renunciar a la empresa y lo que está haciendo es decirnos que van a renunciar cuando realmente NO lo van a hacer. Lo mismo pasa con los falsos negativos, pero en menor medida; lo que dice es que una persona no va a renunciar cuando realmente sí lo va a hacer.

```
Precision: 0.33774834437086093
Recuperacion: 0.723404255319149
F1-score: 0.4604966139954853
Especificidad: 0.7300944669365722
```

Imagen 5. Métricas del modelo de RL – Elaboración propia

Los resultados anteriores se ven reflejados en las métricas donde dicen que la precisión del modelo es de tan solo el 33.7% y el *f1-score* es de poco más del 46% lo que significa que uno de los problemas del modelo puede radicar en el clasificador de las variables. Además, cabe resaltar que, aunque todas las métricas son muy importantes, en este contexto es mucho más importante predecir quién se va a ir (recuperación) para tomar acciones a tiempo, que saber quién no se va a ir (precisión).

B. *Modelo de regresión logística con selector Lasso*

-Alpha de 0.01

Luego de aplicar un selector de variables al modelo se obtuvieron los siguientes resultados:

```
Precision: 0.3190789473684211
Recuperacion: 0.6879432624113475
F1-score:  {0.4359550561797753}
Especificidad: 0.7206477732793523
```

Imagen 6. Métricas del modelo de RL con selector Lasso con Alpha 0.01 – Elaboración propia

Como se dijo anteriormente, uno de los problemas del modelo puede ser el selector de variables, pero esta vez el selector Lasso desmejoró el modelo ya que bajó la precisión y el *f1-score* por lo que se decide cambiar el *Alpha.*

-Alpha de 0.05

```
Precision: 0.3190789473684211
Recuperacion: 0.6879432624113475
F1-score:  {0.4359550561797753}
Especificidad: 0.7206477732793523
```

Imagen 7. Métricas del modelo de RL con selector Lasso con Alpha 0.05 – Elaboración propia

Se puede notar que ninguna de las métricas cambió por lo que se decide hacer una última iteración con otro *Alpha.*

-Alpha de 0.01 con restricción a solo las 4 variables con mayor score.

```
Precision: 0.286144578313253
Recuperacion: 0.6737588652482269
F1-score:  {0.40169133192389006}
Especificidad: 0.680161943319838
```

Imagen 8. Métricas del modelo de RL con selector Lasso con Alpha 0.01 y sólo 4 variables – Elaboración propia

En este modelo se dejó el *Alpha* original pero ahora se restringió el selector a que sólo tomara las 4 variables con mayor *score* con el fin de saber qué tanto afecta el número de variables a las métricas del modelo. Los resultados arrojaron una precisión mas baja por lo que el problema se agravó y dice que el modelo no está siendo capaz de decir correctamente cuándo va a renunciar un trabajador por lo que la conclusión es explorar algún otro selector de variables.

C. *Modelo de regresión logística con selector prueba F y ANOVA.*

Se repite el paso separar el conjunto de datos de entrenamiento y testeo en donde se obtuvieron los siguientes resultados:

```
Tamaño del conjunto de entrenamiento. X:  (3528, 34)  Y:  (3528,)
Tamaño del conjunto de validación. X:  (882, 34)  Y:  (882,)
```

Imagen 9. Tamaño de conjuntos del modelo de RL con selector F y ANOVA – Elaboración propia

En este selector de variables no se realizó matriz de confusión, pero si se calcularon las siguientes métricas de desempeño: Accuracy, Precision, Recall, F1 score y ROC-AUC score.

A continuación, se presentan los resultados de estas métricas.

```
Accuracy:  0.8401360544217688
Precision:  0.0
Recall:  0.0
F1 Score:  0.0
ROC AUC Score:  0.5
```

Imagen 10. Métricas del modelo de RL con selector F y ANOVA – Elaboración propia

A pesar de que este selector de variables aplicado al modelo obtenga una exactitud del 84%, en cuanto a la precisión obtiene un desempeño nulo, indicando que identifica de manera incorrecta los casos positivos. Al no identificar correctamente los casos positivos, su Recall también tendrá un desempeño nulo, por tanto, este modelo no detectará casos positivos de manera efectiva. Este resultado también afectará

el resultado del F1-Score. En cuanto al ROC AUC su resultado es de 0.5, mostrando la incapacidad del modelo para reconocer las clases positivas y negativas de los datos.

D. Decision tree classifier.

-Hiperparámetros estándar.

El criterio que se eligió fue 'gini' que especifica como se toma la decisión para dividir los nodos en el árbol de decisión. Max_depth es equivalente a 6, lo que establece la profundidad máxima del árbol de decisión en 6. Para el caso de max_leaf_nodes=10, se limita el número máximo de nodos hoja (hojas) en el árbol a 10 y por último random_state=0: Esto fija la semilla del generador de números aleatorios para que los resultados sean reproducibles.

```
Train - Accuracy : 0.8645124716553289
Train - classification report:
              precision    recall  f1-score   support

           0       0.88      0.97      0.92      2958
           1       0.66      0.33      0.44       570

    accuracy                           0.86      3528
   macro avg       0.77      0.65      0.68      3528
weighted avg       0.85      0.86      0.85      3528

Test - Accuracy : 0.854875283446712
Test - classification report :               precision    recall  f1-score   support

           0       0.88      0.96      0.92       741
           1       0.59      0.30      0.40       141

    accuracy                           0.85       882
   macro avg       0.73      0.63      0.66       882
weighted avg       0.83      0.85      0.83       882
```

Imagen 11. Métricas DecisionTreeClasifier – Elaboración propia

A grandes rasgos se hace un promedio de las métricas (precisión, recall y F1-score) calculadas para cada clase por separado. En este caso, el promedio macro de precisión es 0.77, el de recall es 0.65, y el de F1-score es 0.68. En este caso, el promedio ponderado de precisión es 0.85, el de recall es 0.86, y el de F1-score es 0.85.

El modelo exhibe una precisión del 86.45% en el conjunto de entrenamiento, con un 88% de precisión para la clase "0" y un 66% para la clase "1". El recall es del 97% para la clase "0" y del 33% para la clase "1". El F1-score es del 92% para la clase "0" y del 44% para la clase "1". En el conjunto de prueba, la precisión es del 85.49%.

En resumen, el modelo muestra un buen rendimiento en términos de precisión, pero hay un desequilibrio en las clases, con una menor precisión, recall y F1-score para la clase "1" en comparación con la clase "0", lo que indica dificultades en la identificación de muestras de la clase "1" en particular.

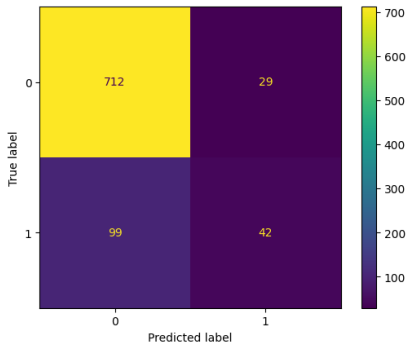

Imagen 16. Matriz de confusión del modelo Decision Tree Clasifier - Elaboración propia

Cuando se analiza la matriz de confusión se puede notar que los falsos negativos son representativos con 99 datos, por lo que reiteramos que el modelo no es confiable. Los falsos positivos con 29 nos muestran que el modelo no predice correctamente las personas que no van a renunciar a la empresa. Ahora en cuánto a este modelo surge un problema de *recall* dónde este se ve disminuido.

```
Precision: 0.5915492957746479
Recuperacion: 0.2978723404255319
F1-score:  {0.3962264150943396}
Especificidad: 0.9608636977058029
```

Imagen 13. Métricas del modelo de Decision Tree Clasifier – Elaboración propia

La conclusión anterior se corrobora con el análisis de las métricas donde se puede notar que la precisión mejoró, pero el *recall* disminuyó considerablemente dejando ver que el modelo no es capaz de predecir correctamente los casos de renuncia dentro de la empresa.

- *Tuneo de hiperparámetros.*

El primer paso es definir todos los hiperparámetros del *Decision Tree Classifier* de acuerdo a la documentación. Los hiperparámetros que mejor se ajustan al modelo son:

```
Mejores hiperparámetros: {'criterion': 'gini', 'max_depth': 10, 'max_leaf_nodes': 30}
```

Imagen 14. Mejores hiperparámetros del modelo de Decision Tree Clasifier – Elaboración propia

Luego de saber cuáles son los mejores hiperparámetros para el modelo, se procede a calcular la matriz de confusión y las métricas para saber cómo se está comportando.

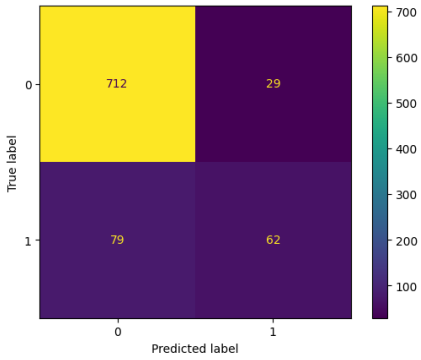

Imagen 15. Matriz de confusión del modelo Decision Tree Clasifier con tuneo de hiperparámetros- Elaboración propia

En la nueva matriz de confusión se puede notar que hubo una disminución en la cantidad de falsos negativos por lo que el modelo es un poco mejor en cuánto a la predicción de las personas que verdaderamente van a renunciar.

```
Precision: 0.6813186813186813
Recall: 0.4397163120567376
F1-score: 0.5344827586206897
Especificidad: 0.9608636977058029
```

Imagen 16. Métricas del modelo Decision Tree Clasifier con tuneo de hiperparámetros- Elaboración propia

La conclusión se corrobora al analizar las métricas después de haber tuneado los hiperparámetros donde se logra ver que tanto la precisión como el *recall* mejoraron y el modelo es capaz de predecir un poco mejor tanto las personas que van a renunciar como las que no lo van a hacer. Además, parece haber un equilibrio un poco más armónico entre las métricas

CONCLUSIONES

A continuación, se postularán las conclusiones más importantes de este estudio.

- Se evidencia que el modelo de regresión logística sin ningún selector es el de mayor *recall* lo que puede hacer pensar que es el mejor debido a que lo que se busca es que el modelo tenga la menor cantidad de falsos negativos ya que predecir que un empleado no va a renunciar cuando en realidad sí lo va a hacer, es mucho mas grave que el caso contrario.
- Los selectores como Lasso y Prueba F y ANOVA no significaron grandes mejores dentro del modelo por lo que su exclusión es inminente.
- Aunque el *Decision Tree Classifier* disminuyó el *recall* significativamente en relación a el modelo de RL, puede representar un buen modelo para el caso de estudio ya que mejora un poco las otras métricas.
- El *Decision Tree Classifier* con tuneo de hiperparámetros puede resultar como el mejor modelo si se toman las métricas en aspectos generales ya que aunque disminuyó la métrica mas importante para nuestro estudio como lo es el *recall*, aumentó todas las otras significativamente dejando como referencia que este es un modelo que se podría abarcar con mayor hincapié para conseguir unas métricas mucho mas altas en futuros estudios

REFERENCIAS

[1] Amazon, «AWS Amazon,» Amazon Web Services, 1 Enero 2022. [En línea]. Available: https://aws.amazon.com/es/what-is/python/. [Último acceso: 25 Septiembre 2022].

[2] A. S. Alberca, «Aprende con Alf,» Aprende con Alf, 14 Junio 2022. [En línea]. Available: https://aprendeconalf.es/docencia/python/manual/pandas/. [Último acceso: 25 Septiembre 2022]

[3] scikit-learn, « scikit-learn Machine Learning in Python» Diciembre 2022. [En línea]. Available: https://scikit-learn.org/stable/