UNIVERSIDAD DE ANTIOQUIA

INTRODUCCIÓN A LA INTELIGENCIA ARTIFICIAL PARA LAS CIENCIAS E INGENIERÍAS

PROYECTO DE CURSO - ENTREGA 2-

LEONARDO MARÍN RESTREPO, 98633850
ANDERSON VALENCIA BERMÚDEZ, 1000869230
WILLIAM ALEXANDER TORRES ZAMBRANO, 71784722

DOCENTE

RAÚL RAMOS POLLÁN

FACULTAD DE INGENIERÍA
INGENIERÍA DE SISTEMAS
ABRIL 23 DE 2023

Predict_students_dropout_and_academic_success

ENTREGA 2

1) Descripción del progreso alcanzado

1.1 Preparación del dataset a trabajar: el dataset seleccionado ha de cumplir con los siguientes requisitos:

1. al menos 5000 instancias (filas, imágenes, etc.)
2. al menos 30 columnas
3. al menos el 10% de las columnas han de ser categóricas
4. al menos ha de tener un 5% de datos faltantes en al menos el 3 columnas.

Como el dataset seleccionado https://www.kaggle.com/datasets/thedevastator/higher-education-predictors-of-student-retention no cumple en su versión original de *kaggle* con alguna de las dos últimas condiciones, LO HEMOS SIMULADO. Es decir, hemos eliminado de manera aleatoria un 5% por ciento de datos en 3 distintas columnas (*Marital status, Previous qualification, Gender*). Esto lo logramos con un notebook creado en google colab, de la siguiente manera:

```
[7]  import pandas as pd
     import numpy as np
```

```
[8]  # cargar el archivo CSV
     df = pd.read_csv('/dataset.csv')
```

```
[13] import pandas as pd

     # cargar el archivo modificado
     df_new = pd.read_csv('/content/newdataset.csv')

     # revisar cantidad de valores faltantes en las columnas de interés
     columnas = ['Marital status', 'Previous qualification', 'Gender']
     for col in columnas:
         n_missing = df_new[col].isna().sum()
         print(f"La columna {col} tiene {n_missing} valores faltantes.")
```

```
La columna Marital status tiene 221 valores faltantes.
La columna Previous qualification tiene 221 valores faltantes.
La columna Gender tiene 221 valores faltantes.
```

```
# eliminar datos al azar de las columnas indicadas
columnas = ['Marital status', 'Previous qualification', 'Gender']
porcentaje_faltante = 0.05

for columna in columnas:
    n_filas = len(df)
    n_eliminar = int(n_filas * porcentaje_faltante)
    indices_eliminar = np.random.choice(n_filas, n_eliminar, replace=False)
    df.loc[indices_eliminar, columna] = np.nan

# guardar el archivo modificado
df.to_csv('newdataset.csv', index=False)
```

1.2 Exploración inicial de los datos: Para empezar a explorar y visualizar los datos en Python, seguimos los siguientes pasos utilizando las bibliotecas Pandas, Matplotlib y Seaborn:

```
[14] # Importar las bibliotecas necesarias:
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
```

```
[15] # Cargar los datos desde el archivo CSV:
     df = pd.read_csv('/content/newdataset.csv')
```

```
[16] # Verificar la estructura de los datos:
     df.head() # muestra las primeras filas del dataframe
     df.info() # muestra información sobre las columnas y tipos de datos
     df.describe() # muestra estadísticas descriptivas de las variables numéricas
```

```
# Visualizar la distribución de los datos:
sns.histplot(df['Age at enrollment'], kde=False) # histograma de la variable Age at enrollment
sns.boxplot(x='Gender', y='Previous qualification', data=df) # diagrama de caja de la variable Calificación
```

```
#Explorar relaciones entre variables:
sns.pairplot(df[['Gender', 'Previous qualification', 'Debtor']]) # gráfico de dispersión de las variables
sns.heatmap(df.corr(), annot=True) # mapa de calor de la matriz de correlación entre variables numéricas
```

Luego escribimos un código en python que nos permita explorar y visualizar los datos utilizando las bibliotecas Pandas, Matplotlib y Seaborn, de la siguiente manera:

```
[2] import pandas as pd
    import matplotlib.pyplot as plt
    import seaborn as sns
```

```
# cargar el archivo CSV
df = pd.read_csv('/content/newdataset.csv')

# mostrar las primeras filas del dataset
print(df.head())

# obtener información general del dataset
print(df.info())

# obtener estadísticas descriptivas del dataset
print(df.describe())

# obtener la distribución de cada variable
for col in df.columns:
    sns.displot(df[col])
    plt.title(f"Distribución de {col}")
    plt.show()
```

```
[ ] # obtener la relación entre dos variables
    sns.scatterplot(x="Gender", y="Age at enrollment", data=df)
    plt.title("Relación entre Gender y Age at enrollment")
    plt.show()

    # obtener la relación entre varias variables
    sns.pairplot(df, hue="Target")
    plt.show()

    # obtener la correlación entre las variables
    sns.heatmap(df.corr(), annot=True)
    plt.title("Correlación entre las variables")
    plt.show()
```

```
[ ]  # obtener un boxplot para cada variable numérica
     for col in df.select_dtypes(include='number'):
         sns.boxplot(x=df[col])
         plt.title(f"Boxplot de {col}")
         plt.show()
```

```
[ ]  # obtener un boxplot para cada variable categórica
     for col in df.select_dtypes(include='object'):
         sns.boxplot(x=df[col], y=df['Target'])
         plt.title(f"Boxplot de {col} por Target")
         plt.show()
```

1.3. Empezar a trabajar en la limpieza de los datos, identificando y completando los valores faltantes

Hemos escrito notebook con los siguientes procesos:

```
#empezar a trabajar en la limpieza de los datos, identificando y completando los valores faltantes
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[1]  import pandas as pd
     import numpy as np

     # cargar el archivo CSV
     df = pd.read_csv('/content/newdataset.csv')
```

```
[ ]  # mostrar las primeras filas del dataset
     print(df.head())
```

```
[ ]  # obtener información general del dataset
     print(df.info())
```

```
[ ]  # obtener estadísticas descriptivas del dataset
     print(df.describe())
```

```
[ ]  # identificar valores faltantes
     print(df.isnull().sum())
```

```
[ ]  # obtener porcentaje de valores faltantes por columna
     print(df.isnull().mean() * 100)
```

```
[ ]  # eliminar columnas con más del 50% de valores faltantes
     umbral = 0.5
     df.dropna(thresh=umbral*len(df), axis=1, inplace=True)
```

```
[ ]  # completar los valores faltantes en las columnas restantes con la media
     df.fillna(df.mean(), inplace=True)
```

Descripción del proceso: usamos la función **isnull()** para identificar los valores faltantes y la función **sum()** para contarlos. Luego, usamos la función **mean()** para obtener el porcentaje de valores faltantes por columna y eliminamos aquellas que superen el umbral del 50% con la función **dropna()**. Finalmente, usamos la función **fillna()** para completar los valores faltantes en las columnas restantes con la media. Es importante tener en cuenta que hay muchas técnicas diferentes para la imputación

de datos y la eliminación de filas y columnas con valores faltantes, y que la elección de la técnica adecuada depende del tipo de datos y del análisis que se desee realizar.

Muy importante: cuando se ejecuta **df.fillna(df.mean(), inplace=True)**, los valores faltantes en las columnas del dataframe se completan con la media de cada columna respectiva. Los valores completados se quedan en su posición original en el **dataframe**, reemplazando los valores faltantes. La opción **inplace=True** indica que el dataframe original se modificará directamente, en lugar de crear una copia con los valores completados.

1.4 Construcción inicial del modelo: para construir un modelo de predicción en Python, lo primero que se debe hacer es importar las bibliotecas necesarias para cargar y manipular los datos, y para construir el modelo.

```
[ ]  #Construcción incial del modelo
     import pandas as pd
     import numpy as np
     from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LogisticRegression
     from sklearn.metrics import accuracy_score, confusion_matrix
```

```
[ ]  #se cargan los datos en un DataFrame de pandas
     df = pd.read_csv('/content/newdataset.csv')
```

```
[ ]  #Se dividen los datos en conjuntos de entrenamiento y prueba.
```

```
     X = df.drop('Target', axis=1)
     y = df['Target']
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

estamos dividiendo los datos en un 80% para entrenamiento y un 20% para prueba.

Luego, se debe construir el modelo de predicción adecuado para los datos y problema específico "***Predict_students_dropout_and_academic_success***" . En función de la naturaleza de los datos y del problema que se está tratando de resolver, es posible que consideremos inicialmente diferentes tipos de modelos, como modelos de regresión lineal, modelos de árboles de decisión, modelos de redes neuronales, entre otros.

Intentamos construimos inicialmente un modelo de red neuronal de la siguiente manera:

```
[65] #Construcción incial del modelo
     import numpy as np
     import pandas as pd
     import tensorflow as tf
     from tensorflow.keras.models import Sequential
     from tensorflow.keras.layers import Dense, Dropout
     from tensorflow.keras.callbacks import EarlyStopping
     from sklearn.model_selection import train_test_split
```

```
[66] # Separar las variables dependientes e independientes
     data = pd.read_csv("/content/newdataset.csv")
     X = data.drop('Application order', axis=1)
     y = data['Inflation rate']
```

```
[67] from sklearn.model_selection import train_test_split

     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```python
from sklearn.preprocessing import StandardScaler

# Cargar datos con pandas
data = pd.read_csv('/content/newdataset.csv')

# Eliminar columnas no numéricas si es necesario
data = data.drop('Gender', axis=1)

# Cambiar el nombre de la columna
data = data.rename(columns={'Previous qualification': 'Age at enrollment'})

# Separar variables dependientes e independientes
X = data.drop('Age at enrollment', axis=1)
y = data['Age at enrollment']

# Escalar los datos
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Hay una columna con valores no numéricos en tus datos. Se debe verificar si hay alguna columna con valores no numéricos en el conjunto. Acá vamos en esta parte profe.

Recordemos que el propósito del conjunto de datos analizados proporciona una visión integral de los estudiantes matriculados en varios títulos universitarios que se ofrecen en una institución de educación superior. Incluye datos demográficos, factores socioeconómicos e información sobre el rendimiento académico que se pueden utilizar para analizar los posibles predictores de la deserción y el éxito académico de los estudiantes.

La métrica con la cual será analizado el rendimiento del modelo es **Área bajo la curva ROC (AUC):** la medida de la capacidad del modelo para distinguir entre los estudiantes que abandonan o tienen éxito académico y los que no lo hacen.

Referencias

Curso de Introducción a la Inteligencia Artificial para las Ciencias e Ingeniería
https://rramosp.github.io/ai4eng.v1/content/M00_intro_udea.html

Predict students' dropout and academic success

https://www.kaggle.com/datasets/thedevastator/higher-education-predictors-of-student-retention