IMPORTAR LAS LIBRERÍAS Y LOS DATASETS

```
import numpy as np
import pandas as pd
import matplotlib.pyplot
import seaborn as sns
%matplotlib inline
!ls -lh /kaggle/input/*
```

```
-rw-r--r-- 1 root root 1.7K Jun 13 07:24 /kaggle/input/test_df.csv
-rw-r--r-- 1 root root 6.8K Jun 13 07:24 /kaggle/input/train_df.csv
```

```
train_df = pd.read_csv('/kaggle/input/train_df.csv')
train_df.head(3)
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Gender	HourlyRate
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	2	Female	94
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	3	Male	61
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	4	Male	92

3 rows × 35 columns

```
test_df = pd.read_csv('/kaggle/input/test_df.csv')
test_df.head(3)
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Gender	HourlyRate
0	32	No	Travel_Rarely	1059	Research & Development	3	4	Life Sciences	4	Female	71
1	26	No	Travel_Rarely	811	Research & Development	1	3	Medical	3	Male	67
2	32	No	Travel_Rarely	1329	Research & Development	10	3	Life Sciences	1	Male	86

3 rows × 35 columns

```
train_df.info()
```

```
<class'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       1470 non-null   int64
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64
 6   Education                 1470 non-null   int64
 7   EducationField            1470 non-null   object
 8   EnvironmentSatisfaction   1470 non-null   int64
 9   Gender                    1470 non-null   object
 10  HourlyRate                1470 non-null   int64
 11  JobInvolvement            1470 non-null   int64
 12  JobLevel                  1470 non-null   int64
 13  JobRole                   1470 non-null   object
 14  JobSatisfaction           1470 non-null   int64
 15  MaritalStatus             1470 non-null   object
 16  MonthlyIncome             1470 non-null   int64
 17  MonthlyRate               1470 non-null   int64
 18  NumCompaniesWorked        1470 non-null   int64
 19  OverTime                  1470 non-null   object
 20  PercentSalaryHike         1470 non-null   int64
 21  PerformanceRating         1470 non-null   int64
 22  RelationshipSatisfaction  1470 non-null   int64
 23  StandardHours             1470 non-null   int64
 24  StockOptionLevel          1470 non-null   int64
 25  TotalWorkingYears         1470 non-null   int64
 26  TrainingTimesLastYear     1470 non-null   int64
 27  WorkLifeBalance           1470 non-null   int64
 28  YearsAtCompany            1470 non-null   int64
 29  YearsInCurrentRole        1470 non-null   int64
 30  YearsSinceLastPromotion   1470 non-null   int64
 31  YearsWithCurrManager      1470 non-null   int64
 32  Unnamed: 0                1470 non-null   int64
 33  Unnamed: 0.1              1470 non-null   int64
 34  Unnamed: 0.1.1            1470 non-null   int64
dtype: object, int64, float64
memory usage: 403.9 KB
```

```
test_df.info()
```

```
<class'pandas.core.frame.DataFrame'>
RangeIndex: 490 entries, 0 to 489
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       490 non-null    int64
 1   Attrition                 490 non-null    object
 2   BusinessTravel            490 non-null    object
 3   DailyRate                 490 non-null    int64
 4   Department                490 non-null    object
 5   DistanceFromHome          490 non-null    int64
 6   Education                 490 non-null    int64
 7   EducationField            490 non-null    object
 8   EnvironmentSatisfaction   490 non-null    int64
 9   Gender                    490 non-null    object
 10  HourlyRate                490 non-null    int64
 11  JobInvolvement            490 non-null    int64
 12  JobLevel                  490 non-null    int64
 13  JobRole                   490 non-null    object
 14  JobSatisfaction           490 non-null    int64
 15  MaritalStatus             490 non-null    object
 16  MonthlyIncome             490 non-null    int64
 17  MonthlyRate               490 non-null    int64
 18  NumCompaniesWorked        490 non-null    int64
 19  OverTime                  490 non-null    object
 20  PercentSalaryHike         490 non-null    int64
 21  PerformanceRating         490 non-null    int64
 22  RelationshipSatisfaction  490 non-null    int64
 23  StandardHours             490 non-null    int64
 24  StockOptionLevel          490 non-null    int64
 25  TotalWorkingYears         490 non-null    int64
 26  TrainingTimesLastYear     490 non-null    int64
 27  WorkLifeBalance           490 non-null    int64
 28  YearsAtCompany            490 non-null    int64
 29  YearsInCurrentRole        490 non-null    int64
 30  YearsSinceLastPromotion   490 non-null    int64
 31  YearsWithCurrManager      490 non-null    int64
 32  Unnamed: 0                490 non-null    int64
 33  Unnamed: 0.1              490 non-null    int64
 34  Unnamed: 0.1.1            490 non-null    int64
dtype: object, int64, float64
memory usage: 134.4 KB
```

```
# Preprocesamiento de datos
train_df = train_df.drop(['Unnamed: 0', 'Unnamed: 0.1', 'Unnamed: 0.1.1'], axis=1)
test_df = test_df.drop(['Unnamed: 0', 'Unnamed: 0.1', 'Unnamed: 0.1.1'], axis=1)

# Mostrar las primeras filas del dataset de entrenamiento
print("Primeras filas del dataset de entrenamiento:")
train_df.head()
```

```
Primeras filas del dataset de entrenamiento:
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Gender	HourlyRate
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	2	Female	94
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	3	Male	61
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	4	Male	92
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	4	Female	56
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	Male	40

```
# Mostrar información del dataset
print("\nInformación del dataset:")
train_df.info()
```

```
Información del dataset:
```

```
# Mostrar estadísticas descriptivas
print("\nEstadísticas descriptivas:")
train_df.describe()
```

```
Estadísticas descriptivas:
```

	Age	DailyRate	DistanceFromHome	Education	EnvironmentSatisfaction	HourlyRate	JobInvolvement	JobLevel	JobSatisfaction	MonthlyIncome
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000	1470.000000
mean	36.923810	802.485714	9.192517	2.912925	2.721769	65.891156	2.729932	2.063946	2.728571	6502.931293
std	9.135373	403.509100	8.106864	1.024165	1.093082	20.329428	0.711561	1.106940	1.102846	4707.956783
min	18.000000	102.000000	1.000000	1.000000	1.000000	30.000000	1.000000	1.000000	1.000000	1009.000000
25%	30.000000	465.000000	2.000000	2.000000	2.000000	48.000000	2.000000	1.000000	2.000000	2911.000000
50%	36.000000	802.000000	7.000000	3.000000	3.000000	66.000000	3.000000	2.000000	3.000000	4919.000000
75%	43.000000	1157.000000	14.000000	4.000000	4.000000	83.750000	3.000000	3.000000	4.000000	8379.000000
max	60.000000	1499.000000	29.000000	5.000000	4.000000	100.000000	4.000000	5.000000	4.000000	19999.000000

```
# Información del dataset
print("\nInformación del dataset:")
train_df.info()
```

```
Información del dataset:
```

```
# Verificar valores nulos
print("\nValores nulos por columna:")
print(train_df.isnull.sum())
```

```
Valores nulos por columna:
```

```
# Crear una copia para el análisis
df = train_df.copy

# Convertir variables categóricas a numéricas para la matriz de correlación
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder

# Crear copia para variables numéricas
df_numeric = df.copy

# Convertir variables categóricas a numéricas
for column in df_numeric.select_dtypes(include=['object']):
    df_numeric[column] = le.fit_transform(df_numeric[column])

# Calcular la matriz de correlación
correlation_matrix = df_numeric.corr

# Mostrar las correlaciones con la variable objetivo (Attrition)
print("Correlaciones con la variable objetivo (Attrition):")
attrition_correlations = correlation_matrix['Attrition'].sort_values(ascending=False)
print(attrition_correlations)
```

```
Correlaciones con la variable objetivo (Attrition):
Attrition                 1.000000
OverTime                  0.246118
JobRole                   0.104779
MaritalStatus             0.103099
Department                0.063991
JobLevel                  0.057305
BusinessTravel            0.055061
TotalWorkingYears         0.054692
MonthlyIncome             0.050818
YearsInCurrentRole        0.050818
Age                       0.050818
YearsWithCurrManager      0.050818
YearsAtCompany            0.050818
JobInvolvement            0.050818
EnvironmentSatisfaction   0.050818
JobSatisfaction           0.050818
RelationshipSatisfaction  0.050818
WorkLifeBalance           0.050818
YearsSinceLastPromotion   0.050818
PerformanceRating         0.050818
PercentSalaryHike         0.050818
StockOptionLevel          0.050818
TrainingTimesLastYear     0.050818
DailyRate                 0.050818
HourlyRate                0.050818
MonthlyRate               0.050818
NumCompaniesWorked        0.050818
DistanceFromHome          0.050818
Education                 0.050818
EducationField            0.050818
Gender                    0.050818
StandardHours             0.050818
Name: Attrition, dtype: float64
```

VISUALIZACIÓN DEL DATASET

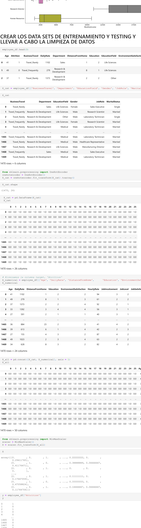
```
# Configurar el estilo para los gráficos
plt.style.use('seaborn')
sns.set_palette('Paletta')

# Crear figura para múltiples gráficos
fig, axes = plt.subplots(2, 2, figsize=(15, 10))

# Distribución de Attrition
sns.countplot(data=df, x='Attrition', ax=axes[0,0])
axes[0,0].set_title('Distribución de Attrition')

# Distribución de edad por Attrition
sns.boxplot(data=df, x='Attrition', y='Age', ax=axes[0,1])
axes[0,1].set_title('Distribución de edad por Attrition')

# Distribución de horas extras por Attrition
sns.countplot(data=df, x='OverTime', hue='Attrition', ax=axes[1,0])
axes[1,0].set_title('Horas extras vs Attrition')

# Distribución de salario por Attrition
sns.boxplot(data=df, x='Attrition', y='MonthlyIncome', ax=axes[1,1])
axes[1,1].set_title('Salario vs Attrition')

plt.tight_layout
plt.show
```

```
<ipykrnl.ipykernel.zmq.ipkernelkernel.IPyKernelKernel object at 0x7f9a4f0a0a00>
```

```
# Crear matriz de correlación para variables numéricas
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm', center=0)
plt.title('Matriz de correlación')
plt.show
```

```
# Crear gráfico de correlaciones con la variable objetivo
plt.figure(figsize=(10, 6))
attrition_correlations[1:].plot(kind='barh')
plt.title('Correlaciones con Attrition')
plt.xlabel('Correlación')
plt.tight_layout
plt.show
```

```
# Crear gráfico de distribución de edad por Attrition
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x='Age', hue='Attrition', kde=True, bins=30)
plt.title('Distribución de edad por Attrition')
plt.xlabel('Edad')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```




```
# Crear gráfico de distribución de salario por Attrition
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x='MonthlyIncome', hue='Attrition', kde=True, bins=30)
plt.title('Distribución de salario por Attrition')
plt.xlabel('Salario Mensual')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```



```
# Crear gráfico de distribución de horas extras por Attrition
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='OverTime', hue='Attrition')
plt.title('Distribución de horas extras por Attrition')
plt.xlabel('Horas Extras')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```

```
# Crear gráfico de boxplot para salario por Attrition
plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='Attrition', y='MonthlyIncome')
plt.title('Salario por Attrition')
plt.xlabel('Attrition')
plt.ylabel('Salario Mensual')
plt.tight_layout
plt.show
```


CREAR LOS DATA SETS DE ENTRENAMIENTO Y TESTING Y LLEVAR A CABO LA LIMPIEZA DE DATOS

```
train_df.head(3)
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Gender	HourlyRate
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	2	Female	94
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	3	Male	61
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	4	Male	92

```
# Crear copias para el entrenamiento
X = train_df.copy
y = train_df['Attrition']

# Eliminar la columna objetivo de las características
X = X.drop('Attrition', axis=1)

# Mostrar información sobre los datasets
print("Forma del dataset de entrenamiento:", X.shape)
print("Forma del dataset de testing:", test_df.shape)
```

```
Forma del dataset de entrenamiento: (1470, 34)
Forma del dataset de testing: (490, 35)
```

```
# Mostrar las primeras filas de cada dataset
print("\nPrimeras filas del dataset de entrenamiento:")
X.head(3)
```

```
Primeras filas del dataset de entrenamiento:
```

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Gender	HourlyRate	JobInvolvement
0	41	Travel_Rarely	1102	Sales	1	2	Life Sciences	2	Female	94	3
1	49	Travel_Frequently	279	Research & Development	8	1	Life Sciences	3	Male	61	2
2	37	Travel_Rarely	1373	Research & Development	2	2	Other	4	Male	92	2

```
# Mostrar información sobre las características
print("\nInformación sobre las características:")
X.info()
```

```
Información sobre las características:
```

```
# Mostrar las primeras filas del dataset de testing
print("\nPrimeras filas del dataset de testing:")
test_df.head(3)
```

```
Primeras filas del dataset de testing:
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EnvironmentSatisfaction	Gender	HourlyRate
0	32	No	Travel_Rarely	1059	Research & Development	3	4	Life Sciences	4	Female	71
1	26	No	Travel_Rarely	811	Research & Development	1	3	Medical	3	Male	67
2	32	No	Travel_Rarely	1329	Research & Development	10	3	Life Sciences	1	Male	86

```
# Mostrar información sobre el dataset de testing
print("\nInformación sobre el dataset de testing:")
test_df.info()
```

```
Información sobre el dataset de testing:
```

```
# Crear copia para el análisis
df = train_df.copy

# Convertir variables categóricas a numéricas para la matriz de correlación
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder

# Crear copia para variables numéricas
df_numeric = df.copy

# Convertir variables categóricas a numéricas
for column in df_numeric.select_dtypes(include=['object']):
    df_numeric[column] = le.fit_transform(df_numeric[column])

# Calcular la matriz de correlación
correlation_matrix = df_numeric.corr

# Mostrar las correlaciones con la variable objetivo (Attrition)
print("Correlaciones con la variable objetivo (Attrition):")
attrition_correlations = correlation_matrix['Attrition'].sort_values(ascending=False)
print(attrition_correlations)
```

```
Correlaciones con la variable objetivo (Attrition):
```

```
# Crear figura para múltiples gráficos
fig, axes = plt.subplots(2, 2, figsize=(15, 10))

# Distribución de Attrition
sns.countplot(data=df, x='Attrition', ax=axes[0,0])
axes[0,0].set_title('Distribución de Attrition')

# Distribución de edad por Attrition
sns.boxplot(data=df, x='Attrition', y='Age', ax=axes[0,1])
axes[0,1].set_title('Distribución de edad por Attrition')

# Distribución de horas extras por Attrition
sns.countplot(data=df, x='OverTime', hue='Attrition', ax=axes[1,0])
axes[1,0].set_title('Horas extras vs Attrition')

# Distribución de salario por Attrition
sns.boxplot(data=df, x='Attrition', y='MonthlyIncome', ax=axes[1,1])
axes[1,1].set_title('Salario vs Attrition')

plt.tight_layout
plt.show
```

```
<ipykrnl.ipykernel.zmq.ipkernelkernel.IPyKernelKernel object at 0x7f9a4f0a0a00>
```



```
# Crear matriz de correlación para variables numéricas
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm', center=0)
plt.title('Matriz de correlación')
plt.show
```



```
# Crear gráfico de correlaciones con la variable objetivo
plt.figure(figsize=(10, 6))
attrition_correlations[1:].plot(kind='barh')
plt.title('Correlaciones con Attrition')
plt.xlabel('Correlación')
plt.tight_layout
plt.show
```



```
# Crear gráfico de distribución de edad por Attrition
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x='Age', hue='Attrition', kde=True, bins=30)
plt.title('Distribución de edad por Attrition')
plt.xlabel('Edad')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```



```
# Crear gráfico de distribución de salario por Attrition
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x='MonthlyIncome', hue='Attrition', kde=True, bins=30)
plt.title('Distribución de salario por Attrition')
plt.xlabel('Salario Mensual')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```



```
# Crear gráfico de distribución de horas extras por Attrition
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='OverTime', hue='Attrition')
plt.title('Distribución de horas extras por Attrition')
plt.xlabel('Horas Extras')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```



```
# Crear gráfico de boxplot para salario por Attrition
plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='Attrition', y='MonthlyIncome')
plt.title('Salario por Attrition')
plt.xlabel('Attrition')
plt.ylabel('Salario Mensual')
plt.tight_layout
plt.show
```


ENTRENAR Y EVALUAR UN CLASIFICADOR UTILIZANDO LA REGRESIÓN LOGÍSTICA

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score, roc_curve
```

```
!pip install imblearn
```

```
Collecting imblearn
```

```
# Crear copia para el análisis
df = train_df.copy

# Convertir variables categóricas a numéricas
le = LabelEncoder

# Crear copia para variables numéricas
df_numeric = df.copy

# Convertir variables categóricas a numéricas
for column in df_numeric.select_dtypes(include=['object']):
    df_numeric[column] = le.fit_transform(df_numeric[column])

# Calcular la matriz de correlación
correlation_matrix = df_numeric.corr

# Mostrar las correlaciones con la variable objetivo (Attrition)
print("Correlaciones con la variable objetivo (Attrition):")
attrition_correlations = correlation_matrix['Attrition'].sort_values(ascending=False)
print(attrition_correlations)
```

```
Correlaciones con la variable objetivo (Attrition):
```

```
# Crear figura para múltiples gráficos
fig, axes = plt.subplots(2, 2, figsize=(15, 10))

# Distribución de Attrition
sns.countplot(data=df, x='Attrition', ax=axes[0,0])
axes[0,0].set_title('Distribución de Attrition')

# Distribución de edad por Attrition
sns.boxplot(data=df, x='Attrition', y='Age', ax=axes[0,1])
axes[0,1].set_title('Distribución de edad por Attrition')

# Distribución de horas extras por Attrition
sns.countplot(data=df, x='OverTime', hue='Attrition', ax=axes[1,0])
axes[1,0].set_title('Horas extras vs Attrition')

# Distribución de salario por Attrition
sns.boxplot(data=df, x='Attrition', y='MonthlyIncome', ax=axes[1,1])
axes[1,1].set_title('Salario vs Attrition')

plt.tight_layout
plt.show
```



```
# Crear matriz de correlación para variables numéricas
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm', center=0)
plt.title('Matriz de correlación')
plt.show
```



```
# Crear gráfico de correlaciones con la variable objetivo
plt.figure(figsize=(10, 6))
attrition_correlations[1:].plot(kind='barh')
plt.title('Correlaciones con Attrition')
plt.xlabel('Correlación')
plt.tight_layout
plt.show
```



```
# Crear gráfico de distribución de edad por Attrition
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x='Age', hue='Attrition', kde=True, bins=30)
plt.title('Distribución de edad por Attrition')
plt.xlabel('Edad')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```



```
# Crear gráfico de distribución de salario por Attrition
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x='MonthlyIncome', hue='Attrition', kde=True, bins=30)
plt.title('Distribución de salario por Attrition')
plt.xlabel('Salario Mensual')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```



```
# Crear gráfico de distribución de horas extras por Attrition
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='OverTime', hue='Attrition')
plt.title('Distribución de horas extras por Attrition')
plt.xlabel('Horas Extras')
plt.ylabel('Frecuencia')
plt.tight_layout
plt.show
```



```
# Crear gráfico de boxplot para salario por Attrition
plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='Attrition', y='MonthlyIncome')
plt.title('Salario por Attrition')
plt.xlabel('Attrition')
plt.ylabel('Salario Mensual')
plt.tight_layout
plt.show
```

```
[ 0 0 0
```

[1] : 0.0,0.0,0.0,