

Proyecto Final

# “¿Qué necesitas para un alto rendimiento laboral”

Romero Tototzintle Marcos Ernesto

27 de Noviembre de 2024

Introducción a Ciencia de Datos

M.C. Jaime A. Romero Sierra



## 1. Introducción:

Como objetivo principal del proyecto queremos buscar como se relacionan los contextos individuales de cada empleado y como se refleja directamente en su rendimiento laboral.

¿Cuál es la importancia de estudiar estas relaciones?

La importancia radica en lo importante que es para la empresa que todos sus empleados se encuentren en la mejor forma y se refleje directamente en su forma de trabajo, esto se logra de la siguiente manera:

Reconocer todas las características que hacen a los empleados rendir al máximo nivel y como sus individualidades afectan directamente, ser capaz de reconocer esto y analizarlo mediante un análisis exploratorio nos da todas las herramientas necesarias para, una vez encontradas las características ser capaces de predecir y saber al 100% que hacer para tener un mayor rendimiento individual en los empleados.

Como nuestra fuente de datos principal tenemos la base de datos de la empresa en la que se encuentran todos los empleados con sus respectivas características, aquí se encuentra toda la información que necesitamos, en primera instancia la tenemos sucia, pero procederemos a limpiarla para poder trabajar de manera correcta y los datos que tenemos son  $126157 \times 20$  columnas como datos “crudos”.

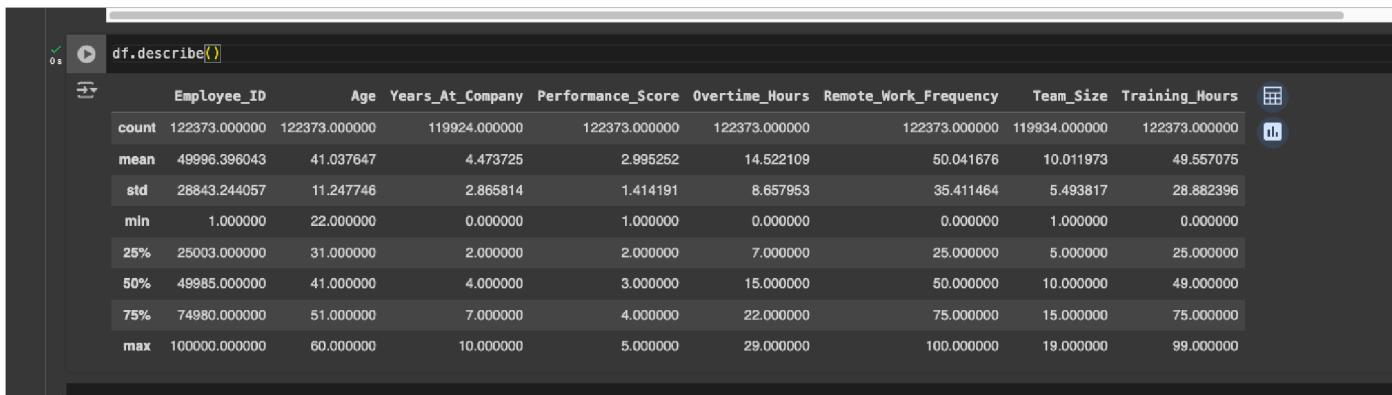
Notamos que como características principales tenemos una gran cantidad de datos que nos permite tener mejor información y para modelos posteriores nos será de gran utilidad es cantidad, además al contar con 20 columnas tenemos muchas características correspondientes a cada empleado, con todo eso tenemos muchas opciones distintas para poder atacar el problema y de la misma manera eso nos abre las posibilidades para poder descubrir la relación de todas las columnas con nuestra variable principal que es el rendimiento laboral.

# Proyecto Final

## 2. Metodología

### Proceso de limpieza de datos:

Al momento de cargar nuestra base de datos lo primero que salta a la vista es la cantidad de datos, la base contaba con 126, 157 filas y 20 columnas, dando un vistazo rápido podemos notar columnas de suma importancia como lo son “Employee\_ID” y “Performance\_Score”, a continuación se presenta un pequeño resumen estadístico de la base en formato de DataFrame.

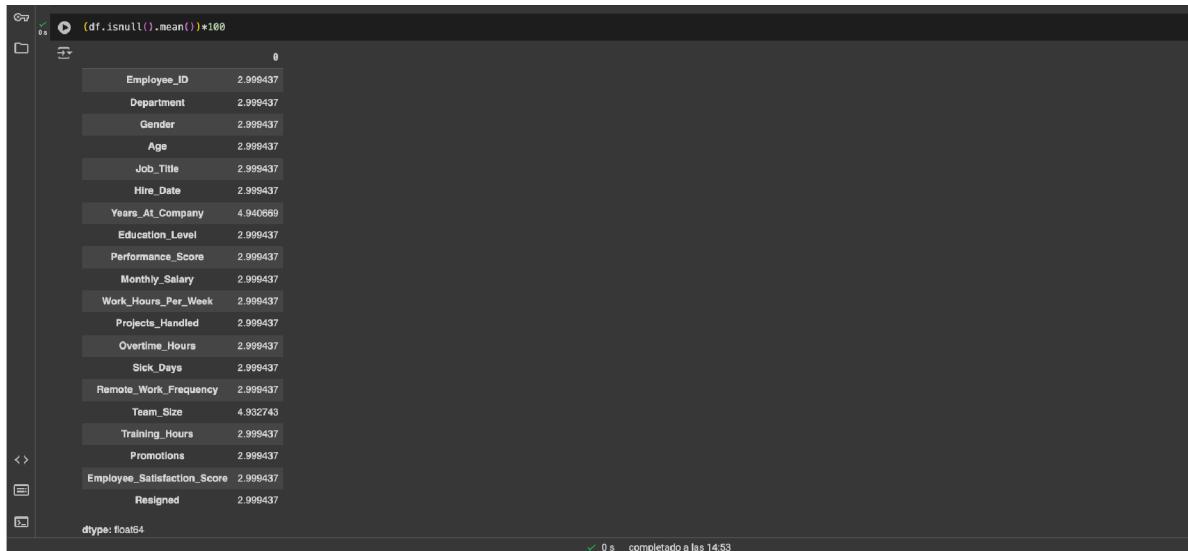


A screenshot of a Jupyter Notebook cell showing the output of `df.describe()`. The table provides statistical summary data for the columns: Employee\_ID, Age, Years\_At\_Company, Performance\_Score, Overtime\_Hours, Remote\_Work\_Frequency, Team\_Size, and Training\_Hours. The output includes counts, mean, standard deviation, minimum, 25th percentile, 50th percentile (median), 75th percentile, and maximum values.

	Employee_ID	Age	Years_At_Company	Performance_Score	Overtime_Hours	Remote_Work_Frequency	Team_Size	Training_Hours
count	122373.000000	122373.000000	119924.000000	122373.000000	122373.000000	122373.000000	119934.000000	122373.000000
mean	49996.396043	41.037647	4.473725	2.995252	14.522109	50.041676	10.011973	49.557075
std	28843.244057	11.247748	2.865814	1.414191	8.657953	35.411484	5.493817	28.882396
min	1.000000	22.000000	0.000000	1.000000	0.000000	0.000000	1.000000	0.000000
25%	25003.000000	31.000000	2.000000	2.000000	7.000000	25.000000	5.000000	25.000000
50%	49985.000000	41.000000	4.000000	3.000000	15.000000	50.000000	10.000000	49.000000
75%	74980.000000	51.000000	7.000000	4.000000	22.000000	75.000000	15.000000	75.000000
max	100000.000000	60.000000	10.000000	5.000000	29.000000	100.000000	19.000000	99.000000

Aquí podemos notar la falta de columnas, lo que nos indica que están mal etiquetadas de manera incorrecta entonces notamos un paso clave que tenemos que revisar si queremos hacer imputaciones con los datos.

A continuación se muestra una tabla que incluye los porcentajes de NAN por columna.



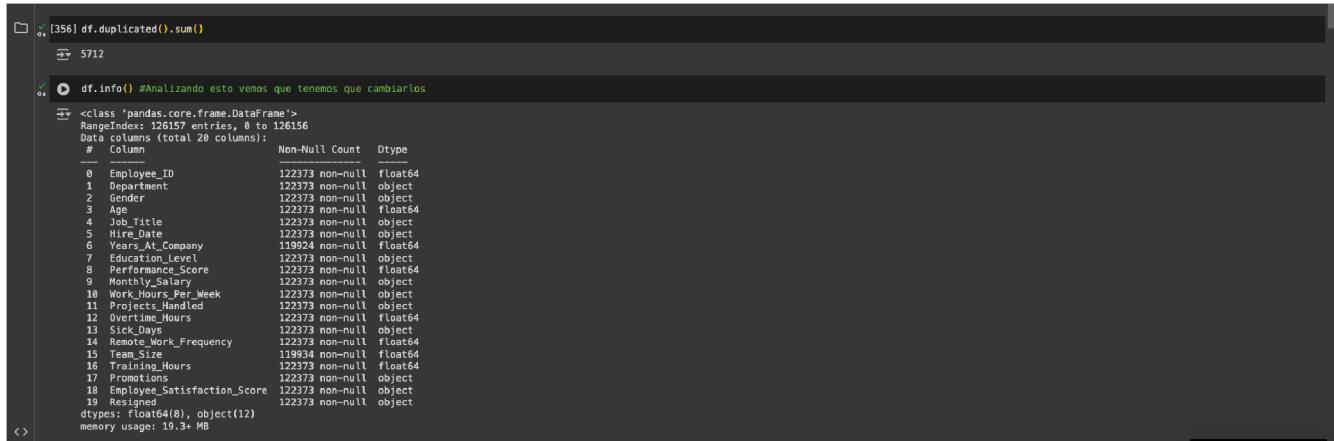
A screenshot of a Jupyter Notebook cell showing the output of `(df.isnull().mean())*100`. The table displays the percentage of missing values for each column: Employee\_ID, Department, Gender, Age, Job\_Title, Hire\_Date, Years\_At\_Company, Education\_Level, Performance\_Score, Monthly\_Salary, Work\_Hours\_Per\_Week, Projects\_Handled, Overtime\_Hours, Sick\_Days, Remote\_Work\_Frequency, Team\_Size, Training\_Hours, Promotions, Employee\_Satisfaction\_Score, and Resigned. All columns show a 2.999437% missing value rate.

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Education_Level	Performance_Score	Monthly_Salary	Work_Hours_Per_Week	Projects_Handled	Overtime_Hours	Sick_Days	Remote_Work_Frequency	Team_Size	Training_Hours	Promotions	Employee_Satisfaction_Score	Resigned
%	2.999437	2.999437	2.999437	2.999437	2.999437	2.999437	4.940668	2.999437	2.999437	2.999437	2.999437	2.999437	2.999437	2.999437	4.932743	2.999437	2.999437	2.999437	2.999437	

## Proyecto Final

Con esto notamos que no tenemos porcentajes tan altos individuales pero tomando en cuenta la magnitud de los datos representa contidades considerablese de datos.

En la ultima parte de nuestro analisis inicial mostraremos la cantidad de filas duplicadas que encontramos en conjunto con la descripción del formato en que se encuentran nuestros datos y un pequeño parrafo de los problemas que pudimos encontrar



```
[356] df.duplicated().sum()
5712

[357] df.info() #Analizando esto vemos que tenemos que cambiarlos
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 126157 entries, 0 to 126156
Data columns (total 20 columns):
 #   Column      Non-Null Count  Dtype  
0   Employee_ID    122373 non-null   float64
1   Department     122373 non-null   object  
2   Gender         122373 non-null   object  
3   Age            122373 non-null   float64
4   Job_Title      122373 non-null   object  
5   Hire_Date      122373 non-null   object  
6   Year_In_Compny 119834 non-null   float64
7   Education_Level 122373 non-null   object  
8   Performance_Score 122373 non-null   float64
9   Monthly_Salary 122373 non-null   object  
10  Work_Hours_Per_Week 122373 non-null   object  
11  Projects_Handled 122373 non-null   object  
12  Overtime_Hours 122373 non-null   float64
13  SICK_LEAVE_HRS 122373 non-null   object  
14  Remote_Work_Frequency 122373 non-null   float64
15  Team_Size       119834 non-null   float64
16  Training_Hours 122373 non-null   float64
17  Promotions      122373 non-null   object  
18  Employee_Satisfaction_Score 122373 non-null   object  
19  Resigned        122373 non-null   object  
dtypes: float64(8), object(12)
memory usage: 19.3+ MB
```

Notamos que aproximadamente un 5% de nuestras filas están repetidas y nos confirma lo que ya habiamos visto, valores etiquetados de manera incorrecta que representa un problema para nuestra limpieza, por lo que lo que sigue ahora es ver que errores traen nuestros datos para empezar a resolverlos uno por uno.

### Proceso de limpieza

Ocupamos todos los comandos visto en clase, ya sea un su forma literal o modificados ligeramente para que se adaptaran a nuestras necesidades especificas, por nombrara algunos serian:

#### 1. Eliminación de duplicados

Aquí lo utilicé en dos formas distintas, una lo podríamos llamar la forma “general” para todo el “df” y otra especializada para una columna, y de esa manera tener un mejor control sobre que y como queria eliminar los duplicados.

#### 2. Imputación de valores nan

## Proyecto Final

En mi caso solo hice dos imputaciones de valores, cambiar los nan por el promedio y la mediana, ya que la moda no era algo que me sirviera en mi análisis, pero para hacer esto primero tenía que reemplazar los valores invalidos y tener mis datos en formato “float”.

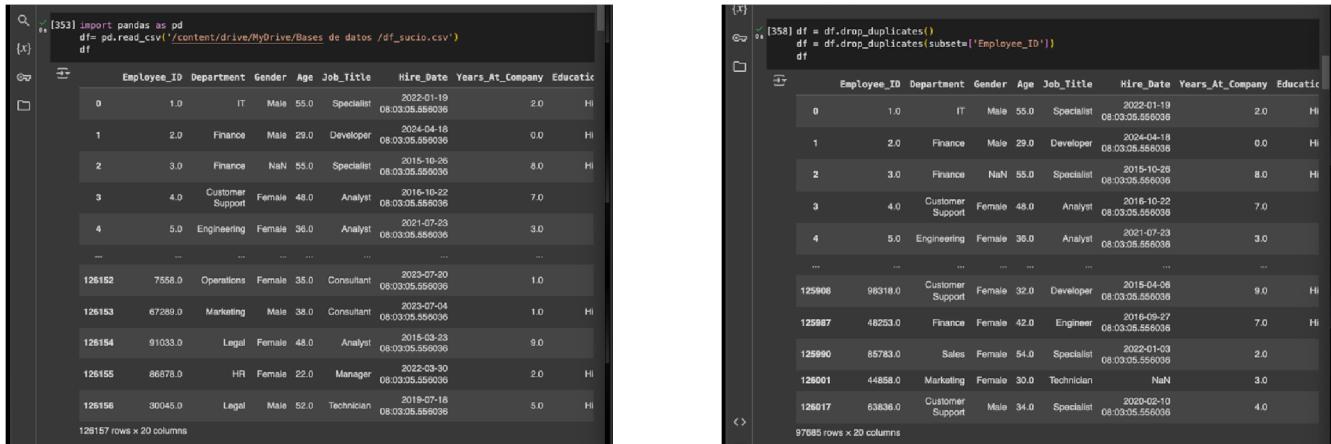
### 3. Cambio de formato de valores

Como se menciona en el párrafo anterior esto era un paso crucial para poder hacer la imputación de valores de manera efectiva, al tenerlos en el formato ideal podemos trabajar de mejor manera, sin embargo, esto tampoco sería posible sin hacer el reemplazo de valores.

### 4. Reemplazo de valores

Esto es un paso crucial al quitar los valores invalidos que en mi caso eran “bbb” y reemplazarlos con valores nan y de esa manera poder realizar todas las modificaciones que fueron necesarias.

Por nombrar algunos, en seguida se mostraran imágenes del antes y el después de aplicar estos comandos a nuestro “df”, y veremos como modificó nuestra base de datos.



```
[353]: import pandas as pd
df = pd.read_csv('/content/drive/MyDrive/Bases de datos /df_socio.csv')
df
```

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Educational
0	1.0	IT	Male	55.0	Specialist	2022-01-19	2.0	Hi
1	2.0	Finance	Male	29.0	Developer	2024-04-18	0.0	Hi
2	3.0	Finance	NaN	55.0	Specialist	2015-10-26	8.0	Hi
3	4.0	Customer Support	Female	48.0	Analyst	2016-10-22	7.0	
4	5.0	Engineering	Female	36.0	Analyst	2021-07-23	3.0	
...	...	...	...	...	...	...	...	...
125152	7558.0	Operations	Female	38.0	Consultant	2023-07-20	1.0	
125153	67289.0	Marketing	Male	38.0	Consultant	2023-07-04	1.0	Hi
125154	91033.0	Legal	Female	48.0	Analyst	2015-03-23	9.0	
125155	66878.0	HR	Female	22.0	Manager	2022-03-30	2.0	Hi
125156	30045.0	Legal	Male	52.0	Technician	2019-07-18	5.0	Hi

126157 rows x 20 columns

```
[358]: df = df.drop_duplicates()
df = df.drop_duplicates(subset=['Employee_ID'])
df
```

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Educational
0	1.0	IT	Male	55.0	Specialist	2022-01-19	2.0	Hi
1	2.0	Finance	Male	29.0	Developer	2024-04-18	0.0	Hi
2	3.0	Finance	NaN	55.0	Specialist	2015-10-26	8.0	Hi
3	4.0	Customer Support	Female	48.0	Analyst	2016-10-22	7.0	
4	5.0	Engineering	Female	36.0	Analyst	2021-07-23	3.0	
...	...	...	...	...	...	...	...	...
125968	98518.0	Customer Support	Female	32.0	Developer	2015-04-08	9.0	Hi
125967	48253.0	Finance	Female	42.0	Engineer	2016-09-27	7.0	Hi
125990	65783.0	Sales	Female	54.0	Specialist	2022-01-03	2.0	
128001	44858.0	Marketing	Female	30.0	Technician	NaN	3.0	
1286017	63836.0	Customer Support	Male	34.0	Specialist	2020-02-10	4.0	

97585 rows x 20 columns

Aquí se muestra el antes y el después de aplicar la línea de código que elimina los duplicados, como se nota hay dos comandos ejecutándose al mismo tiempo y, esto se debe a que fue uno de los grandes problemas que tuve cuando me encontraba haciendo la limpieza de los datos, si solo dejara la primera línea de código todo marcharía normal y los duplicados se eliminarían pero al final aparecerían duplicados, fue un paso donde me tardé mucho pero poniendo códigos para buscar duplicados por columnas encontré que en “Employee\_ID” tenía

## Proyecto Final

repetidos entonces fue hacer una pequeña modificación y se solucionó, con esto nos redujo en un aproximado de 30, 000 datos.

Una forma que pensé era con una función y hacer que el numero de empleado no se repitiera para poder tomar esos datos en el análisis pero leyendo las instrucciones mencionaba la eliminación de duplicados que fue lo que hice.

The screenshot shows a Jupyter Notebook cell with the following Python code:

```
[359] df['Employee_ID']= df['Employee_ID'].replace('bbb', None)
df['Department']= df['Department'].replace('bbb', None)
df['Gender']= df['Gender'].replace('bbb', None)
df['Job_Title']= df['Job_Title'].replace('bbb', None)
df['Hire_Date']= df['Hire_Date'].replace('bbb', None)
df['Education_Level']= df['Education_Level'].replace('bbb', None)
df['Performance_Score']= df['Performance_Score'].replace('bbb', None)
df['Resigned']= df['Resigned'].replace('bbb', None)

df['Age']= df['Age'].replace('bbb', None)
df['Years_At_Company']= df['Years_At_Company'].replace('bbb', None)
df['Monthly_Salary']= df['Monthly_Salary'].replace('bbb', None)
df['Work_Hours']= df['Work_Hours'].replace('bbb', None)
df['OverTime_Handled']= df['OverTime_Handled'].replace('bbb', None)
df['Overtime_Hours']= df['Overtime_Hours'].replace('bbb', None)
df['Sick_Days']= df['Sick_Days'].replace('bbb', None)
df['Remote_Work_Frequency']= df['Remote_Work_Frequency'].replace('bbb', None)
df['Team_Size']= df['Team_Size'].replace('bbb', None)
df['Training_Hours']= df['Training_Hours'].replace('bbb', None)
df['Promotions']= df['Promotions'].replace('bbb', None)
df['Employee_Satisfaction_Score']= df['Employee_Satisfaction_Score'].replace('bbb', None)
df
```

Below the code is a table preview of the DataFrame:

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Educational
0	1.0	IT	Male	55.0	Specialist	2022-01-19	2.0	Hi
1	2.0	Finance	Male	29.0	Developer	2024-04-18	0.0	Hi
2	3.0	Finance	Nan	55.0	Specialist	2015-10-26	8.0	Hi
3	4.0	Customer Support	Female	48.0	Analyst	2016-10-22	7.0	Hi
4	5.0	Engineering	Female	36.0	Analyst	2021-07-23	3.0	Hi
...	...	...	...	...	...	...	...	...
125908	98318.0	Customer Support	Female	32.0	Developer	2015-04-06	9.0	Hi

Este paso fue de suma importancia al reemplazar los valores invalidos ("bbb") por nan para poder realizar todas las conversiones necesarias, este no quita valores solo reemplaza.

The screenshot shows a Jupyter Notebook cell with the following Python code:

```
[362] df = df.dropna(subset=['Employee_ID'])
df = df.dropna(subset=['Department'])
df = df.dropna(subset=['Gender'])
df = df.dropna(subset=['Job_Title'])
df = df.dropna(subset=['Hire_Date'])
df = df.dropna(subset=['Education_Level'])
df = df.dropna(subset=['Resigned'])
df = df.dropna(subset=['Performance_Score'])
```

Below the code is a table preview of the DataFrame:

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Educational
0	1.0	IT	Male	55.0	Specialist	2022-01-19	2.0	Hi
1	2.0	Finance	Male	29.0	Developer	2024-04-18	0.0	Hi
3	4.0	Customer Support	Female	48.0	Analyst	2016-10-22	7.0	Hi
4	5.0	Engineering	Female	36.0	Analyst	2021-07-23	3.0	Hi
6	7.0	IT	Male	37.0	Technician	2023-08-28	1.0	Hi
...	...	...	...	...	...	...	...	...
125862	86205.0	Operations	Male	57.0	Analyst	2016-07-10	8.0	Hi
125867	91011.0	Legal	Female	38.0	Specialist	2016-06-16	6.0	Hi
125908	98318.0	Customer Support	Female	32.0	Developer	2015-04-06	9.0	Hi
125987	48253.0	Finance	Female	42.0	Engineer	2016-09-27	7.0	Hi
125990	85783.0	Sales	Female	54.0	Specialist	2022-01-03	2.0	Hi

## Proyecto Final

Aquí eliminamos los valores nan de las columnas que contenian texto ya que estas no puede ser imputadas al contener formato str, por lo que decidimos prescindir de esas filas.

```
df['Hire_Date'] = pd.to_datetime(df['Hire_Date'])
df['Monthly_Salary'] = df['Monthly_Salary'].astype(float)
df['Work_Hours_Per_Week'] = df['Work_Hours_Per_Week'].astype(float)
df['Projects_Handled'] = df['Projects_Handled'].astype(float)
df['Sick_Days'] = df['Sick_Days'].astype(float)
df['Promotions'] = df['Promotions'].astype(float)
df['Employee_Satisfaction_Score'] = df['Employee_Satisfaction_Score'].astype(float)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 75832 entries, 0 to 125998
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Employee_ID      75832 non-null   float64
 1   Department        75832 non-null   object 
 2   Gender            75832 non-null   object 
 3   Age               73589 non-null   float64
 4   Job_Title         75832 non-null   object 
 5   Hire_Date         75832 non-null   datetime64[ns]
 6   Years_At_Company 72097 non-null   float64
 7   Education_Level  75832 non-null   object 
 8   Performance_Score 75832 non-null   float64
 9   Monthly_Salary    72154 non-null   float64
 10  Work_Hours_Per_Week 72129 non-null   float64
 11  Projects_Handled  72122 non-null   float64
 12  Overtime_Hours    73538 non-null   float64
 13  Sick_Days          72178 non-null   float64
 14  Remote_Work_Frequency 73587 non-null   float64
 15  Team_Size          72125 non-null   float64
 16  Training_Hours    73557 non-null   float64
 17  Promotions          72098 non-null   float64
 18  Employee_Satisfaction_Score 72066 non-null   float64
 19  Resigned            75832 non-null   object 
dtypes: datetime64[ns](1), float64(14), object(5)
memory usage: 14.2+ MB
```

En los pasos anteriores identificamos las columnas mal etiquetadas, con esa información y el reemplazo de los valores invalidos procedemos a cambiar el formato de las columnas erroneas al adecuados y lo comprobamos de manera inmediata.

```
#Lo siguiente que haré será hacer imputación específica para cada columna, tanto de media, media y moda seg
#Primero empezaremos con la mediana ya son datos muy importantes para el análisis y no podemos simplemente
df['Age']= df['Age'].fillna(df['Age'].mode()[0])
df['Years_At_Company']= df['Years_At_Company'].fillna(df['Years_At_Company'].mode()[0])
df['Work_Hours_Per_Week']= df['Work_Hours_Per_Week'].fillna(df['Work_Hours_Per_Week'].mode()[0])
df['Projects_Handled']= df['Projects_Handled'].fillna(df['Projects_Handled'].mode()[0])
df['Remote_Work_Frequency']= df['Remote_Work_Frequency'].fillna(df['Remote_Work_Frequency'].mode()[0])
df['Team_Size']= df['Team_Size'].fillna(df['Team_Size'].mode()[0])
df['Training_Hours']= df['Training_Hours'].fillna(df['Training_Hours'].mode()[0])

#Ahora iremos con las columnas que cambiaremos por el promedio
df['Monthly_Salary']= df['Monthly_Salary'].fillna(df['Monthly_Salary'].mean())
df['Overtime_Hours']= df['Overtime_Hours'].fillna(df['Overtime_Hours'].mean())
df['Sick_Days']= df['Sick_Days'].fillna(df['Sick_Days'].mean())
df['Promotions']= df['Promotions'].fillna(df['Promotions'].mean())
df['Employee_Satisfaction_Score']= df['Employee_Satisfaction_Score'].fillna(df['Employee_Satisfaction_Score'].mean())

df
```

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Education_Level	Per
0	1.0	IT	Male	55.0	Specialist	2022-01-19	2.0	High School	
1	2.0	Finance	Male	29.0	Developer	2024-04-18	0.0	High School	
3	4.0	Customer Support	Female	48.0	Analyst	2016-10-22	7.0	Bachelor	
4	5.0	Engineering	Female	36.0	Analyst	2021-07-23	3.0	Bachelor	
5	7.0	IT	Male	37.0	Technician	2023-08-28	1.0	Postgraduate	

# Proyecto Final

Gracias a todos los pasos importantes anteriores y los intermedios es que podemos hacer la imputación de valores de manera correcta, aquí podemos notar que algunas columnas son mediana y otras promedio, esto se decidió despues de analizar los datos y darnos cuenta cual era el más idoneo para la base de datos.

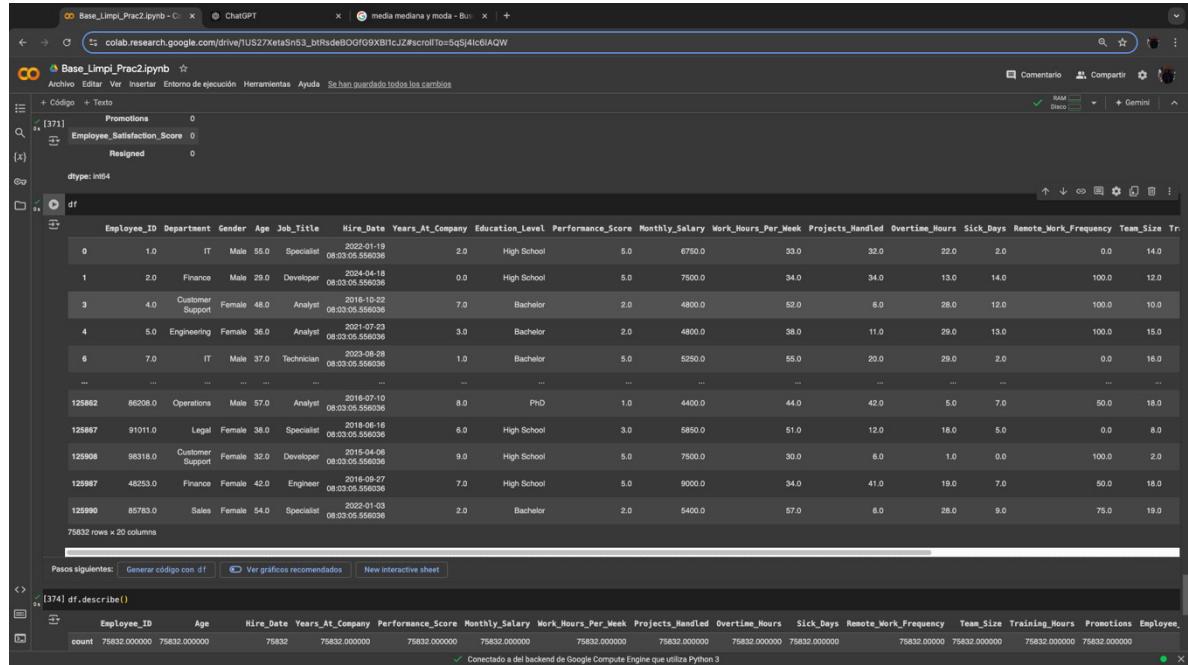
## Conclusión

Después de este proceso de limpieza pudimos identificar de manera clara y concisa los problemas lo que nos permitió manejarlos de una manera sencilla, en consecuencia tenemos una base de datos que mejoró en la calidad de los datos que contiene y nos permitirá hacer un mejor análisis en el futuro.

Sin embargo, es importante mencionar que debido a la eliminación de filas por falta de información puede llegar a existir sesgos en el análisis.

Procederemos a presentar los resultados obtenidos.

## Resultados



The screenshot shows a Google Colab notebook titled "Base\_Limpia\_Prac2.ipynb". The code cell displays the cleaned dataset "df" with 75832 rows and 20 columns. The columns include Employee\_ID, Department, Gender, Age, Job\_Title, Hire\_Date, Years\_At\_Company, Education\_Level, Performance\_Score, Monthly\_Salary, Work\_Hours\_Per\_Week, Projects\_Handled, Overtime\_Hours, Sick\_Days, Remote\_Work\_Frequency, Team\_Size, Training\_Hours, Promotions, and Employee\_Satisfaction\_Score. The data shows various employee details like hire dates, job titles, and performance scores. The interface includes a sidebar with file operations and a bottom toolbar with execution controls.

Aquí se muestra el resultado de nuestra base de datos despues de la limpieza y lo primero que salta al vista es reducción en las filas de casi 60,000 datos, datos que contenian errores y trabajamos en ellos.

# Proyecto Final

```
0 s df.info()

<class 'pandas.core.frame.DataFrame'>
Index: 75832 entries, 0 to 125990
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Employee_ID      75832 non-null   float64
 1   Department        75832 non-null   object 
 2   Gender            75832 non-null   object 
 3   Age               75832 non-null   float64
 4   Job_Title         75832 non-null   object 
 5   Hire_Date         75832 non-null   datetime64[ns]
 6   Years_At_Company 75832 non-null   float64
 7   Education_Level   75832 non-null   object 
 8   Performance_Score 75832 non-null   float64
 9   Monthly_Salary    75832 non-null   float64
 10  Work_Hours_Per_Week 75832 non-null   float64
 11  Projects_Handled 75832 non-null   float64
 12  Overtime_Hours   75832 non-null   float64
 13  Sick_Days         75832 non-null   float64
 14  Remote_Work_Frequency 75832 non-null   float64
 15  Team_Size          75832 non-null   float64
 16  Training_Hours    75832 non-null   float64
 17  Promotions         75832 non-null   float64
 18  Employee_Satisfaction_Score 75832 non-null   float64
 19  Resigned           75832 non-null   object 

dtypes: datetime64[ns](1), float64(14), object(5)
memory usage: 14.2+ MB
```

```
Q 0 s df.duplicated().sum()
{x} [369] df.duplicated().sum()
0
df == 'bbb').sum()
0
Employee_ID 0
Department 0
Gender 0
Age 0
Job_Title 0
Hire_Date 0
Years_At_Company 0
Education_Level 0
Performance_Score 0
Monthly_Salary 0
Work_Hours_Per_Week 0
Projects_Handled 0
Overtime_Hours 0
Sick_Days 0
Remote_Work_Frequency 0
Team_Size 0
Training_Hours 0
Promotions 0
Employee_Satisfaction_Score 0
Resigned 0
dtype: int64
df
Conectado a el backend de Google Compute Engine que uti
```

```
0 s (df.isnull().mean())*100
0
Employee_ID 0.0
Department 0.0
Gender 0.0
Age 0.0
Job_Title 0.0
Hire_Date 0.0
Years_At_Company 0.0
Education_Level 0.0
Performance_Score 0.0
Monthly_Salary 0.0
Work_Hours_Per_Week 0.0
Projects_Handled 0.0
Overtime_Hours 0.0
Sick_Days 0.0
Remote_Work_Frequency 0.0
Team_Size 0.0
Training_Hours 0.0
Promotions 0.0
Employee_Satisfaction_Score 0.0
Resigned 0.0
dtype: float64
```

# Proyecto Final

## Análisis Exploratorio de datos (EDA):

### - Visión general.

Notamos que la base de datos ya está limpia y eliminamos todos los datos que nos perjudicaban y ahora tenemos una nueva distribución que queda de la siguiente manera 75832 filas × 20 columnas, mostraremos el resumen y el df.

```
df=pd.read_csv('/content/drive/MyDrive/Bases de datos /Base_limpia_Prac2.csv')
df
```

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Education_Level	Performance_Score	Monthly_Salary	Work_Hours_Per_Week	Projects_Handled	Overtime_Hours	Sick_Days	Remote_Work_Frequency	Team_Size	Training_Hours
0	1.0	IT	Male	55.0	Specialist	2022-01-19 08:03:05.556036	2.0	High School	5.0	6750.0	33.0	32.0	22.0	2.0	0.0	14.0	66.0
1	2.0	Finance	Male	29.0	Developer	2024-04-18 08:03:05.556036	0.0	High School	5.0	7500.0	34.0	34.0	13.0	14.0	100.0	12.0	61.0
2	4.0	Customer_Support	Female	48.0	Analyst	2016-10-22 08:03:05.556036	7.0	Bachelor	2.0	4800.0	52.0	6.0	28.0	12.0	100.0	10.0	0.0
3	5.0	Engineering	Female	36.0	Analyst	2021-07-23 08:03:05.556036	3.0	Bachelor	2.0	4800.0	38.0	11.0	29.0	13.0	100.0	15.0	9.0
4	7.0	IT	Male	37.0	Technician	2023-09-28 08:03:05.556036	1.0	Bachelor	5.0	5250.0	55.0	20.0	29.0	2.0	0.0	16.0	27.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
75827	86208.0	Operations	Male	57.0	Analyst	2016-07-10 08:03:05.556036	8.0	PhD	1.0	4400.0	44.0	42.0	5.0	7.0	50.0	18.0	88.0
75828	91011.0	Legal	Female	38.0	Specialist	2018-06-16 08:03:05.556036	6.0	High School	3.0	5850.0	51.0	12.0	18.0	5.0	0.0	8.0	13.0
75829	98318.0	Customer_Support	Female	32.0	Developer	2015-04-06 08:03:05.556036	9.0	High School	5.0	7500.0	30.0	6.0	1.0	0.0	100.0	2.0	86.0
75830	48253.0	Finance	Female	42.0	Engineer	2016-09-27 08:03:05.556036	7.0	High School	5.0	9000.0	34.0	41.0	19.0	7.0	50.0	18.0	18.0
75831	85783.0	Sales	Female	54.0	Specialist	2022-01-03 08:03:05.556036	2.0	Bachelor	2.0	5400.0	57.0	6.0	28.0	9.0	75.0	19.0	84.0

75832 rows × 20 columns

Pasos siguientes: Generar código con df | Ver gráficos recomendados | New interactive sheet

```
[43]: df.info()
```

```
0 s
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 75832 entries, 0 to 75831
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Employee_ID      75832 non-null   float64
 1   Department       75832 non-null   object 
 2   Gender            75832 non-null   object 
 3   Age               75832 non-null   float64
 4   Job_Title         75832 non-null   object 
 5   Hire_Date         75832 non-null   object 
 6   Years_At_Company 75832 non-null   float64
 7   Education_Level  75832 non-null   object 
 8   Performance_Score 75832 non-null   float64
 9   Monthly_Salary   75832 non-null   float64
 10  Work_Hours_Per_Week 75832 non-null   float64
 11  Projects_Handled 75832 non-null   float64
 12  Overtime_Hours   75832 non-null   float64
 13  Sick_Days         75832 non-null   float64
 14  Remote_Work_Frequency 75832 non-null   float64
 15  Team_Size          75832 non-null   float64
 16  Training_Hours    75832 non-null   float64
 17  Promotions          75832 non-null   float64
 18  Employee_Satisfaction_Score 75832 non-null   float64
 19  Resigned            75832 non-null   bool    
dtypes: bool(1), float64(14), object(5)
memory usage: 11.1+ MB
```

## Proyecto Final

### - Tipos de Variables.

Es facil notar que las variables de nuestro df se dividen en numericas y categoricas, mostraremos todas y haremos una breve descripción de cada una de ellas.

```
[ ] df.columns
→ Index(['Employee_ID', 'Department', 'Gender', 'Age', 'Job_Title', 'Hire_Date',
       'Years_At_Company', 'Education_Level', 'Performance_Score',
       'Monthly_Salary', 'Work_Hours_Per_Week', 'Projects_Handled',
       'Overtime_Hours', 'Sick_Days', 'Remote_Work_Frequency', 'Team_Size',
       'Training_Hours', 'Promotions', 'Employee_Satisfaction_Score',
       'Resigned'],
      dtype='object')
```

### Variables numerica

Son variables que toman valores numéricos y pueden ser medidas, estas variables representan cantidades o magnitudes y se pueden operar matemáticamente, ya sea en las operaciones basicas o en modelos de aprendizaje. En nuestro data frame tenemos las siguientes:

Department, Gender, Job\_Title, Education\_Level, Resigned.

### Variable Categorica

Son variables que representan categorías o grupos. No se pueden operar matemáticamente como las variables numéricas, pero pueden ser representadas por etiquetas o clases, en nuestro caso son las siguientes:

Age, Years\_At\_Company, Performance\_Score, Monthly\_Salary,  
Work\_Hours\_Per\_Week, Projects\_Handled, Overtime\_Hours, Sick\_Days,  
Remote\_Work\_Frequency, Team\_Size, Training\_Hours, Promotions,  
Employee\_Satisfaction\_Score.

# Proyecto Final

## - Resumen estadístico

Luego de presentar y explicar todos los valores tanto numéricos como categóricos procedemos a un breve resumen estadístico en que presentamos valores como la media, mediana, desviación estándar, valores mínimos y máximos para las variables numéricas y la frecuencia de categoría en las variables categóricas.

[21] df.describe()																		
	Employee_ID	Age	Hire_Date	Years_At_Company	Performance_Score	Monthly_Salary	Work_Hours_Per_Week	Projects_Handled	Overtime_Hours	Sick_Days	Remote_Work_Frequency	Team_Size	Training_Hours	Promotions	Employee_Satisfaction_Score			
count	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000	75832.000000		
mean	49995.774866	41.362314	2019-09-14 11:06:16.966832832	4.351013	2.993815	6402.276381	45.063680	23.551693	14.514285	7.015062	49.28724	9.712140	48.877506	0.999953	3.002110			
min	1.000000	22.000000	2014-09-07 08:03:05.556036	0.000000	1.000000	3850.000000	30.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.000000	1.000000			
25%	25068.750000	32.000000	2017-03-20 02:03:05.55603096	2.000000	2.000000	5400.000000	38.000000	10.000000	7.000000	3.000000	25.000000	4.000000	26.000000	0.000000	2.060000			
50%	49963.000000	42.000000	2019-09-19 08:03:05.556038096	4.000000	3.000000	6402.276381	46.000000	23.000000	14.514285	7.000000	50.000000	10.000000	48.000000	1.000000	3.002110			
75%	74931.250000	52.000000	2022-03-11 08:03:05.55603096	7.000000	4.000000	7200.000000	52.000000	36.000000	22.000000	11.000000	75.000000	15.000000	74.000000	2.000000	3.940000			
max	99999.000000	60.000000	2024-09-03 08:03:05.556036	10.000000	5.000000	9000.000000	60.000000	49.000000	29.000000	14.000000	100.000000	19.000000	99.000000	2.000000	5.000000			
std	28862.950603	11.251720	NaN	2.845085	1.413459	1338.667945	8.733967	14.667502	8.538492	4.228329	35.08721	5.506283	28.727585	0.795314	1.122015			

### Age (Edad):

La edad promedio de los empleados es 41.36 años, con un rango de 22 a 60 años. La mediana es de 42 años, lo que indica que la distribución de la edad es relativamente equilibrada. La desviación estándar es de 11.25 años, lo que sugiere una gran variabilidad en la edad de los empleados.

### Years\_At\_Company (Años en la Empresa):

Los empleados tienen un promedio de 4.35 años de antigüedad, con un rango de 0 a 10 años. Esto indica una mezcla de empleados nuevos y empleados con mucha antigüedad. La mediana es de 4 años, lo que sugiere que la mayoría de los empleados tienen una permanencia moderada en la empresa.

### Performance\_Score (Puntuación de Desempeño):

La puntuación de desempeño tiene un promedio de 3.00, lo que indica que los empleados tienen un rendimiento promedio en la escala de 1 a 5. Con una desviación estándar de 1.41, los datos muestran cierta dispersión, lo que sugiere que existen empleados tanto con alto rendimiento como con bajo rendimiento.

## Proyecto Final

### **Monthly\_Salary (Salario Mensual):**

El salario promedio es de 6402.28, con un rango entre 3850 y 9000, lo que refleja una gran variabilidad. La desviación estándar de 1338.67 muestra que los salarios varían considerablemente entre los empleados. La mayoría de los salarios están entre los 5400 y 7200, lo que podría ser el rango típico para la mayoría de los empleados.

### **Work\_Hours\_Per\_Week (Horas de Trabajo por Semana):**

El promedio de horas de trabajo es de 45.06 horas/semana, lo que indica una jornada laboral a tiempo completo. La desviación estándar es de 8.73, lo que sugiere que algunos empleados podrían estar trabajando horas extras o tienen horarios flexibles.

### **Projects\_Handled (Proyectos Manejados):**

El promedio de proyectos manejados por empleado es de 23.55, con un mínimo de 0 (lo que puede indicar empleados que no gestionan proyectos o falta de registros). Este valor refleja que, en general, los empleados están involucrados en una cantidad razonable de proyectos a lo largo del tiempo.

### **Overtime\_Hours (Horas Extras):**

Los empleados trabajan en promedio 14.51 horas extras, con un máximo de 29 horas, lo que sugiere que algunos empleados están sobrecargados de trabajo o hacen muchas horas extras, mientras que otros tienen menos.

### **Sick\_Days (Días de Enfermedad):**

El promedio de días de enfermedad es de 7.02 días. Esto indica que, en promedio, los empleados faltan alrededor de una semana al año por razones de salud. El valor máximo de 14 días podría representar ausencias excepcionales.

## Proyecto Final

### **Remote\_Work\_Frequency (Frecuencia de Trabajo Remoto):**

El promedio de frecuencia de trabajo remoto es de 49.29, lo que parece indicar que los empleados tienen una frecuencia moderada de trabajo remoto. Sin embargo, la interpretación exacta depende de la escala específica utilizada para medir esta variable.

### **Team\_Size (Tamaño del Equipo):**

El tamaño promedio del equipo es de 9.71 empleados, con un máximo de 19 empleados, lo que refleja que los equipos son generalmente de tamaño medio. Este dato sugiere que los empleados tienden a trabajar en equipos grandes.

### **Training\_Hours (Horas de Entrenamiento):**

El promedio de horas de entrenamiento por empleado es de 48.88 horas. Algunos empleados tienen un máximo de 99 horas, lo que sugiere que algunos reciben una formación intensiva. Este valor podría reflejar una política de capacitación bien establecida en la empresa.

### **Promotions (Ascenso):**

El promedio de promociones es de 2.99, lo que indica que, en promedio, los empleados reciben varias promociones a lo largo de su tiempo en la empresa. El máximo de 5 promociones podría indicar que algunos empleados ascienden rápidamente.

### **Employee\_Satisfaction\_Score (Puntuación de Satisfacción del Empleado):**

La puntuación promedio de satisfacción es 3.00, lo que sugiere que los empleados tienen una satisfacción promedio con su trabajo. Esto podría reflejar que la empresa está haciendo un esfuerzo por mantener a los empleados satisfechos, pero no necesariamente es sobresaliente en términos de bienestar laboral.

Ahora explicaremos brevemente la desviación estandar, recordemos que esta mide la dispersión o variabilidad de los datos con respecto a su promedio. Un valor bajo

## Proyecto Final

Indica que los datos están más concentrados alrededor del promedio y un valor más alto indica que los datos se encuentran más dispersos.

### **Variables Numéricas con Alta Desviación Estándar:**

#### **Monthly\_Salary (Salario Mensual):**

La desviación estándar de 1,338.67 sugiere una gran variabilidad en los salarios, lo que indica que los salarios de los empleados pueden variar significativamente.

#### **Age (Edad):**

La desviación estándar de 11.25 indica que hay una amplia diversidad de edades entre los empleados, lo que refleja una fuerza laboral de diferentes rangos de edad.

#### **Work\_Hours\_Per\_Week (Horas de trabajo por semana):**

Con una desviación estándar de 8.73, la variabilidad en las horas trabajadas por los empleados es considerable, lo que sugiere que algunos empleados podrían trabajar muchas más horas que otros.

### **Variables con Desviación Estándar Baja:**

#### **Years\_At\_Company (Años en la empresa):**

La desviación estándar de 2.85 es relativamente baja, lo que indica que la mayoría de los empleados han estado en la empresa entre 2 y 7 años, con una variabilidad más controlada.

#### **Performance\_Score (Puntaje de desempeño):**

La desviación estándar de 1.41 también es moderada, lo que sugiere que la mayoría de los empleados tienen un puntaje de desempeño cercano al promedio (3).

## Proyecto Final

### Variables con Desviación Estándar Muy Baja:

#### Remote\_Work\_Frequency (Frecuencia de trabajo remoto):

La desviación estándar de 35.09 es significativamente alta, lo que refleja una gran diversidad en los patrones de trabajo remoto, aunque este dato parece ser un error o estar mal escalado.

#### Team\_Size (Tamaño del equipo):

Muestra una desviación estándar de 5.50, lo que indica que el tamaño de los equipos varía, pero no en exceso.

```
# Obtener la frecuencia de las categorías para todas las columnas
for column in df.select_dtypes(include='object').columns: # 'object' es el tipo de las variables categóricas
    print(f"Frecuencia en la columna {column}:\n")
    print(df[column].value_counts())
    print("\n")
```

Frecuencia en la columna Department:

Department	Count
Marketing	8528
Operations	8516
Finance	8495
Customer Support	8447
Legal	8427
IT	8392
Sales	8369
HR	8355
Engineering	8303

Name: count, dtype: int64

Frecuencia en la columna Gender:

Gender	Count
Male	36419
Female	3601
Other	3012

Name: count, dtype: int64

Frecuencia en la columna Job\_Title:

Job_Title	Count
Specialist	10952
Manager	10886
Technician	10867
Engineer	10832
Analyst	10806
Consultant	10766
Developer	10723

Name: count, dtype: int64

Frecuencia en la columna Education\_Level:

Education_Level	Count
Bachelor	38049
High School	22667
Master	11281
PhD	3835

Name: count, dtype: int64

Frecuencia en la columna Resigned:

Resigned	Count
False	68254
True	7578

Name: count, dtype: int64

De la variables categoricas tenemos lo siguiente en terminos de su frecuencia.

# Proyecto Final

## **Department (Departamento)**

La distribución de empleados por departamento es bastante equilibrada, con la mayoría de los departamentos (como Marketing, Operations, Finance y Customer Support) con alrededor de 8,000 a 8,500 empleados. No hay un departamento dominante, lo que indica que los recursos están bien distribuidos a través de las diferentes áreas de la empresa.

## **Gender (Género)**

La distribución de género está muy equilibrada entre Masculino (36,419 empleados) y Femenino (36,401 empleados). El grupo de "Otro" (3,012 empleados) es pequeño, lo que sugiere que hay una representación limitada de géneros no especificados.

## **Job\_Title (Título del Trabajo)**

Los empleados están distribuidos equitativamente entre los diferentes títulos de trabajo, con posiciones como Specialist, Manager, Technician y Engineer liderando la cantidad de empleados. No hay una sobrerrepresentación de un título en particular, lo que sugiere que la empresa tiene una estructura organizacional diversa.

## **Education\_Level (Nivel Educativo)**

La mayoría de los empleados tienen un título universitario de licenciatura (38,049 empleados), mientras que 22,667 tienen educación secundaria.

Los empleados con Maestría (11,281) y Doctorado (3,835) son menos comunes, lo que refleja una fuerza laboral con una sólida base educativa, pero con una menor representación de altos estudios.

## **Resigned (Renunció)**

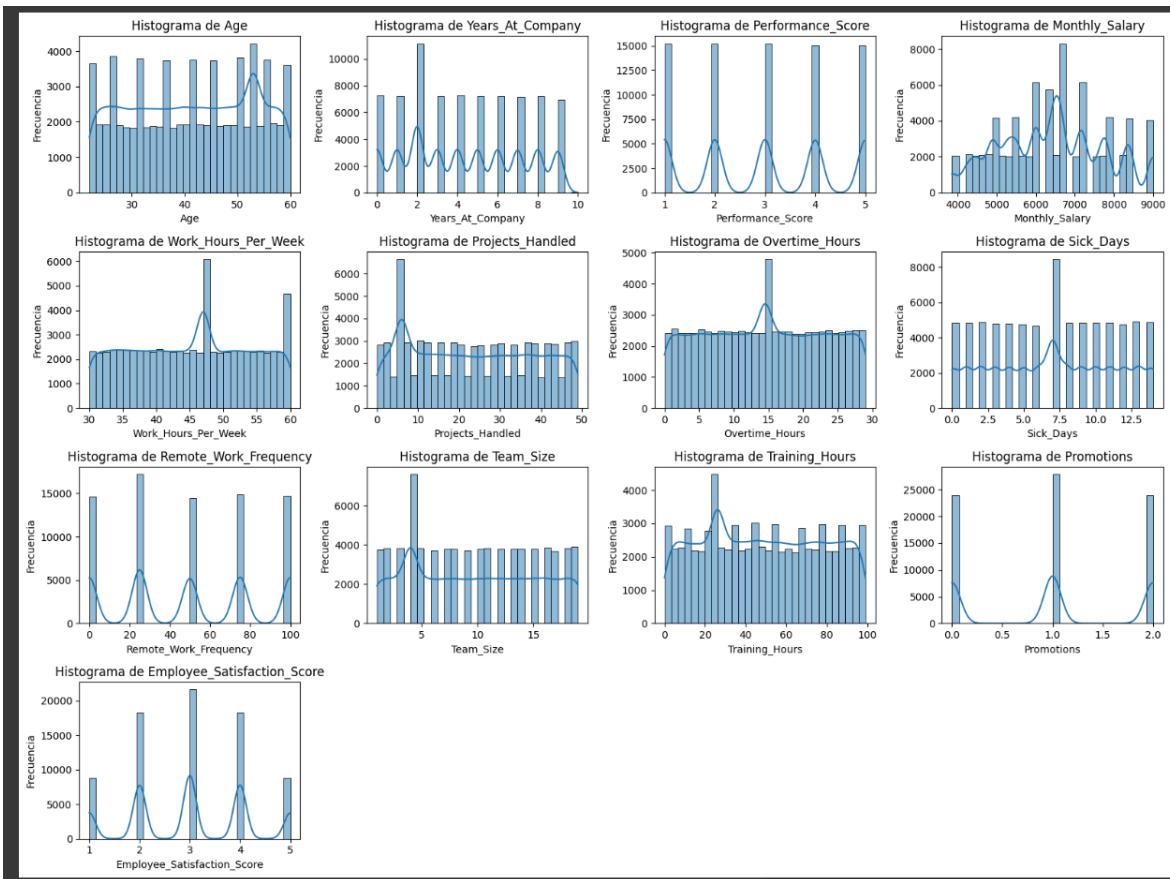
La mayoría de los empleados no han renunciado (68,254), mientras que solo 7,578 empleados han dejado la empresa. Esto sugiere una alta tasa de retención, con menos del 10% de los empleados que renuncian, lo que podría indicar un ambiente de trabajo favorable o políticas efectivas de recursos humanos.

# Proyecto Final

## Visualización y Distribución de Variables Individuales

### - Variables numéricas

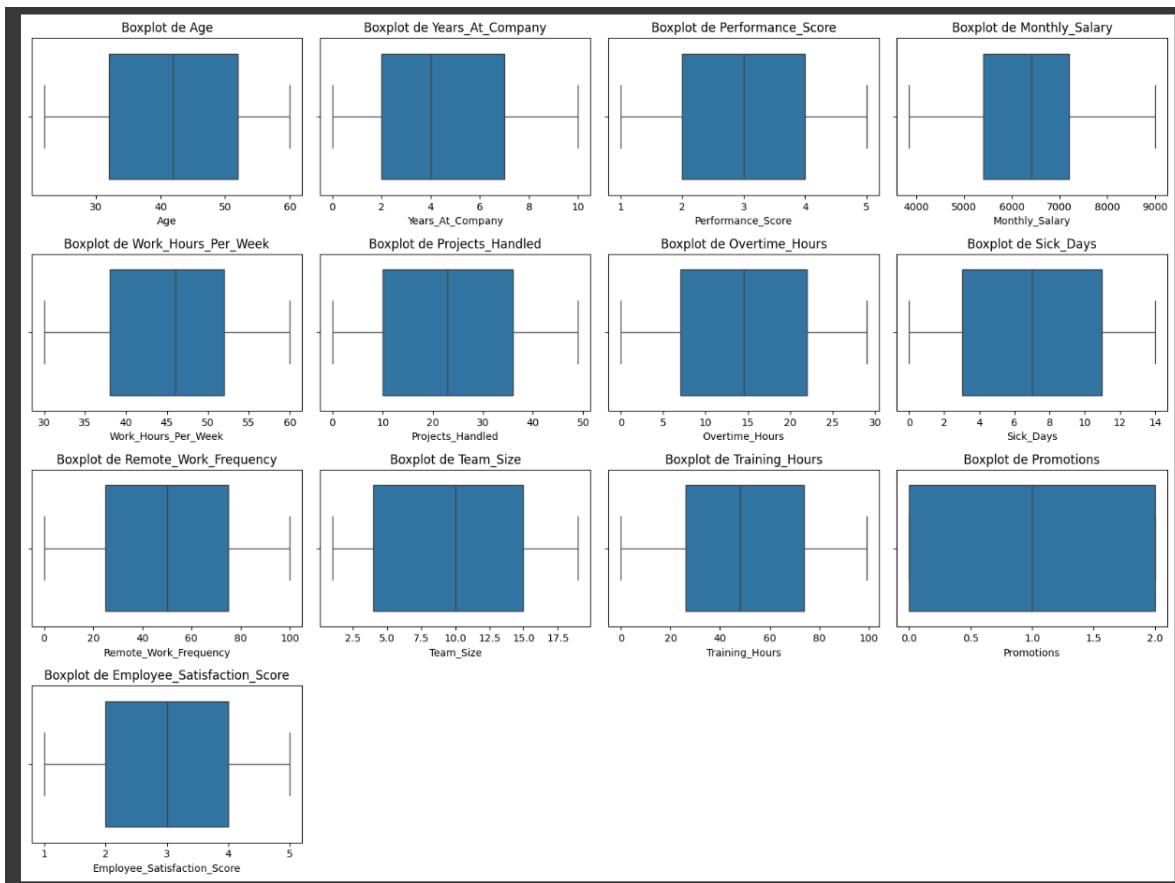
Procederemos a mostar graficas de las variables y a comentar lo que observamos en ellas, primero con los histogramas.



Vemos desde aquí de manera rápida que no parece existir sesgos o valores atípicos, lo primero que salta a la vista de todas los histogramas de las variabels numericas es que hay una columna que destaca más que todas, esto no es necesariamente malo ya que nos está diciendo el valor más común dentro de esa variable.

Como la máxima observación que tenemos que en todas las variables numéricas destaca por sobre todo una columnas que represeta la moda de nuestra respectiva variable de análisis y de la misma manera notamos que no presentamos datos atípicos o sesgos de información dado que todos están dentro del rango esperado. Lo comprobaremos en las gráficas posteriores.

# Proyecto Final



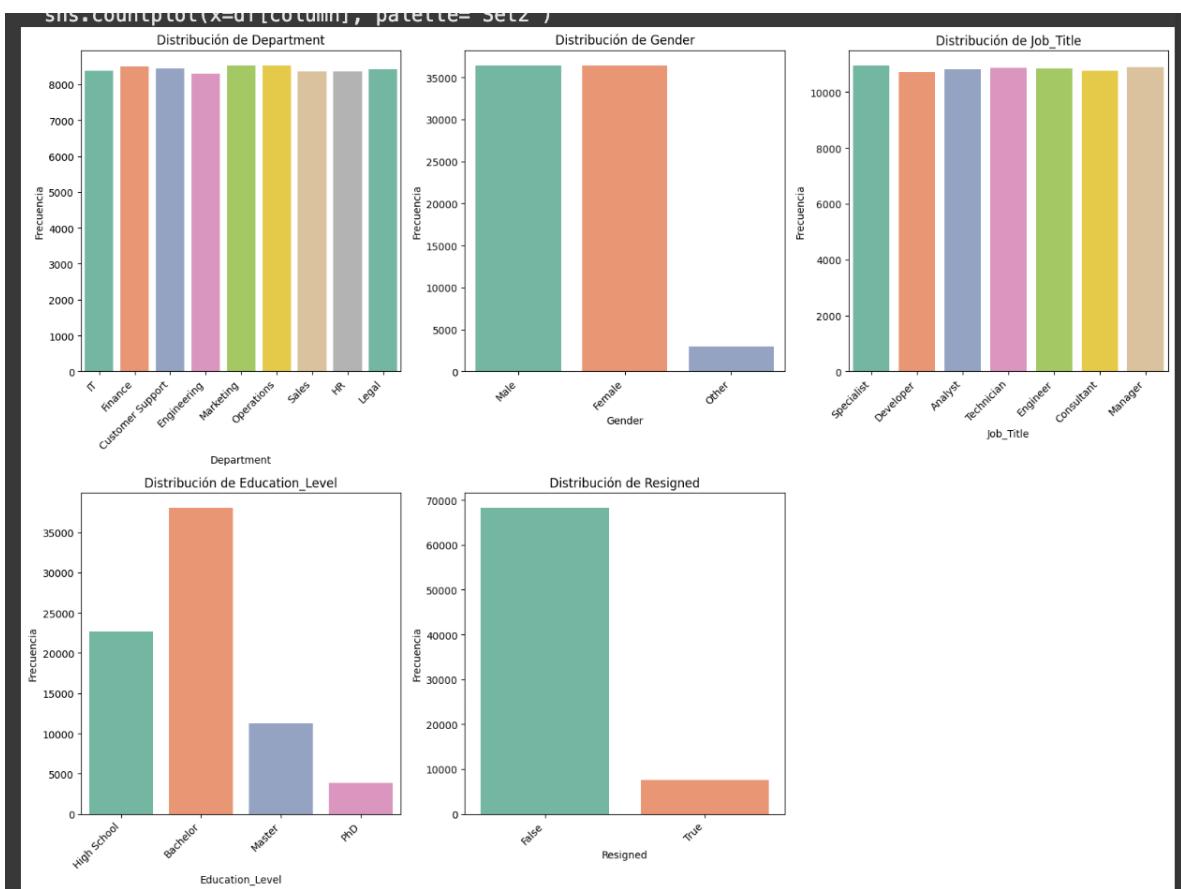
Con esto quedas mas que confirmado lo que habíamos planteado con las gráficas anteriores y es que categorizamos de valores atípicos y faltantes, todas nuestras columnas se encuentran dentro de un rango esperado y con el que podemos trabajar de manera eficiente.

## - Variables categóricas

Procederemos a mostrar graficas de las variables y a comentar lo que observamos en las categóricas y de esa manera lograr visualizar si es que existen categorias dominantes o desequilibradas y al mismo tiempo ver si hay posibles formas de manejar las disparidades o únicamente son nuestros datos y no debemos preocuparnos por eso.

Todas son graficas de barra para mantener el orden visual pero de igual manera es factible hacer elementos graficos en forma de pastel.

## Proyecto Final



Vemos en estas gráficas que efectivamente hay categorías dominantes en 3 de las 5 graficas pero analizando más a fondo no representa un problema ya que es lógico, las dominantes en las gráficas de género son Female y Male, en las gráficas de resigned la dominante es False y por último en la gráfica de nivel educativo tenemos como dominante a Bachelor.

Esto nos da pie a poder decir con seguridad que no existen categorías desequilibradas pero si existen categorías dominantes, no en todas las gráficas ya que la restantes tienen un equilibrio considerable y no hay ninguna que destaque de sobre manera.

# Proyecto Final

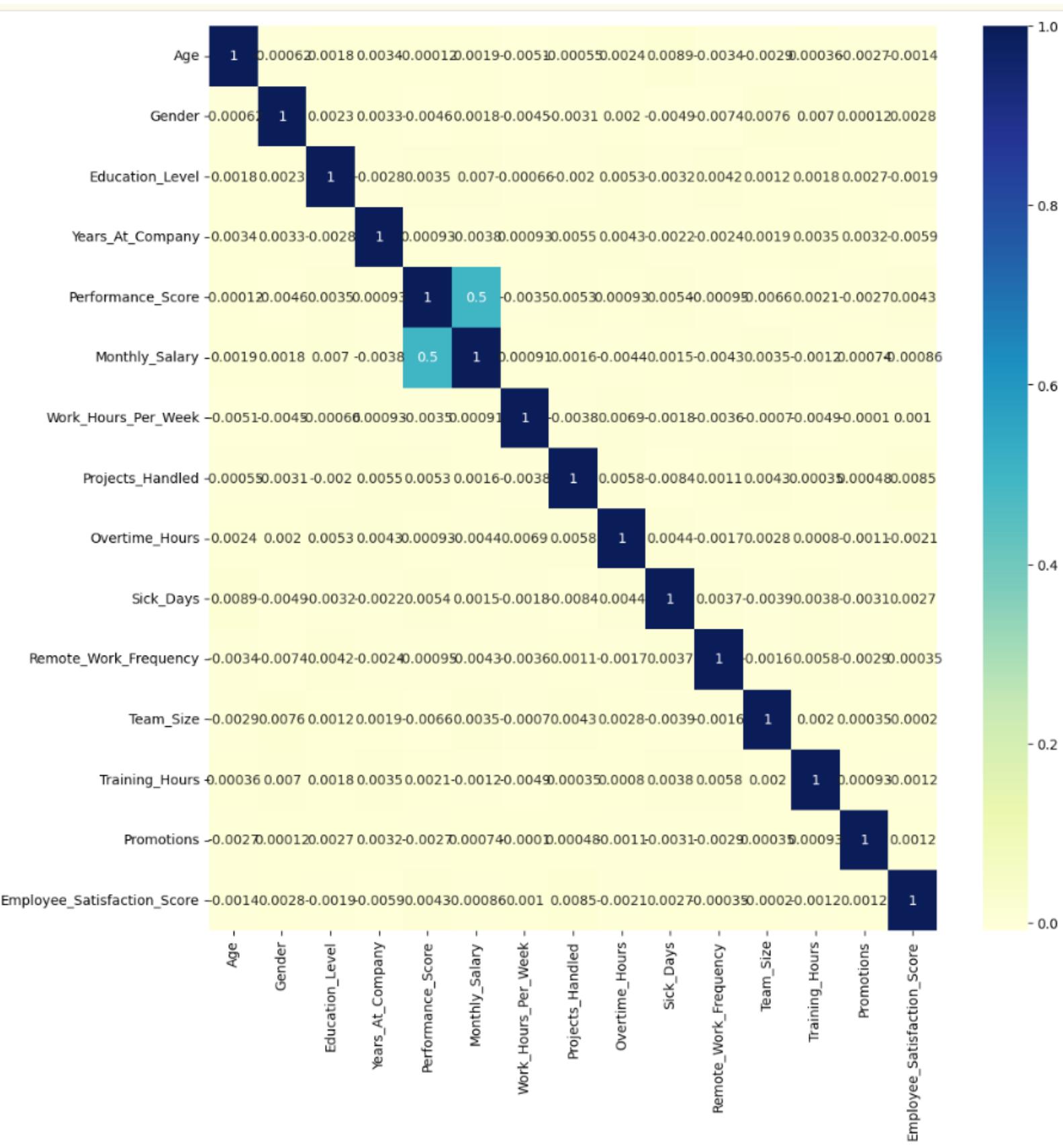
## Correlación entre variables

A continuación vamos a presentar la matriz de correlación junto con el mapa de calor para describir que es lo que observamos y sus características más importantes.

	Age	Gender	Education_Level	Years_At_Company	Performance_Score	Monthly_Salary	Work_Hours_Per_Week
Age	1.000000	-0.000620	0.001804	0.003394	-0.000124	0.001897	-0.005122
Gender	-0.000620	1.000000	0.002324	0.003339	-0.004635	0.001790	-0.004490
Education_Level	0.001804	0.002324	1.000000	-0.002786	0.003525	0.007023	-0.000663
Years_At_Company	0.003394	0.003339	-0.002786	1.000000	0.000930	-0.003835	0.000931
Performance_Score	-0.000124	-0.004635	0.003525	0.000930	1.000000	0.495948	-0.003543
Monthly_Salary	0.001897	0.001790	0.007023	-0.003835	0.495948	1.000000	0.000911
Work_Hours_Per_Week	-0.005122	-0.004490	-0.000663	0.000931	-0.003543	0.000911	1.000000
Projects_Handled	-0.000554	-0.003119	-0.002024	0.005456	0.005280	0.001607	-0.003818
Overtime_Hours	0.002410	0.001972	0.005262	0.004293	0.000930	-0.004396	0.006913
Sick_Days	0.008941	-0.004905	-0.003230	-0.002231	0.005395	0.001534	-0.001822
Remote_Work_Frequency	-0.003374	-0.007447	0.004157	-0.002438	-0.000953	-0.004341	-0.003643
Team_Size	-0.002889	0.007620	0.001209	0.001930	-0.006617	0.003510	-0.000698
Training_Hours	0.000362	0.007022	0.001768	0.003457	0.002078	-0.001236	-0.004874
Promotions	-0.002651	0.000116	0.002715	0.003163	-0.002653	0.000742	-0.000101
Employee_Satisfaction_Score	-0.001405	0.002828	-0.001901	-0.005902	0.004260	-0.000863	0.001002

Projects_Handled	Overtime_Hours	Sick_Days	Remote_Work_Frequency	Team_Size	Training_Hours	Promotions	Employee_Satisfaction_Score	Actions
-0.000554	0.002410	0.008941	-0.003374	-0.002889	0.000362	-0.002651	-0.001405	
-0.003119	0.001972	-0.004905	-0.007447	0.007620	0.007022	0.000116	0.002828	
-0.002024	0.005262	-0.003230	0.004157	0.001209	0.001768	0.002715	-0.001901	
0.005456	0.004293	-0.002231	-0.002438	0.001930	0.003457	0.003163	-0.005902	
0.005280	0.000930	0.005395	-0.000953	-0.006617	0.002078	-0.002653	0.004260	
0.001607	-0.004396	0.001534	-0.004341	0.003510	-0.001236	0.000742	-0.000863	
-0.003818	0.006913	-0.001822	-0.003643	-0.000698	-0.004874	-0.000101	0.001002	
1.000000	0.005762	-0.008380	0.001123	0.004296	0.000348	0.000479	0.008484	
0.005762	1.000000	0.004389	-0.001708	0.002763	0.000801	-0.001126	-0.002060	
-0.008380	0.004389	1.000000	0.003668	-0.003915	0.003845	-0.003129	0.002683	
0.001123	-0.001708	0.003668	1.000000	-0.001600	0.005813	-0.002915	-0.000353	
0.004296	0.002763	-0.003915	-0.001600	1.000000	0.001985	0.000352	-0.000205	
0.000348	0.000801	0.003845	0.005813	0.001985	1.000000	0.000931	-0.001211	
0.000479	-0.001126	-0.003129	-0.002915	0.000352	0.000931	1.000000	0.001242	
0.008484	-0.002060	0.002683	-0.000353	-0.000205	-0.001211	0.001242	1.000000	

# Proyecto Final



## Correlaciones Relevantes

### 1- Performance\_Score:

#### Con Monthly\_Salary: 0.495948

Hay una correlación moderada positiva entre Performance\_Score y Monthly\_Salary. Esto sugiere que un mejor rendimiento en el trabajo puede estar relacionado con salarios más altos. Es importante tener en cuenta esta relación, ya que podría indicar que los empleados con mejor desempeño tienen mayores salarios, lo cual es un patrón común en muchas organizaciones.

#### Con Projects\_Handled: 0.005280

La correlación con la cantidad de proyectos gestionados es muy baja (cercana a 0). Esto sugiere que no hay una relación significativa entre el rendimiento y el número de proyectos gestionados por un empleado en este conjunto de datos.

#### Con Employee\_Satisfaction\_Score: 0.004260

La correlación con la satisfacción del empleado es muy baja, lo que sugiere que el rendimiento y la satisfacción no están fuertemente relacionados en este conjunto de datos, lo cual es interesante porque normalmente una alta satisfacción podría esperarse que se traduzca en mejor rendimiento.

### 2- Monthly\_Salary:

#### Con Years\_At\_Company: -0.003835

La correlación negativa entre Monthly\_Salary y Years\_At\_Company es bastante débil. Esto podría indicar que no hay una relación clara entre la duración de un empleado en la empresa y su salario. En muchas organizaciones, los salarios podrían aumentar con la antigüedad, pero en este conjunto de datos no parece ser el caso.

## Proyecto Final

### **Con Work\_Hours\_Per\_Week: 0.000911**

La correlación entre las horas de trabajo semanales y el salario es muy baja. Esto indica que no hay una relación clara entre la cantidad de horas trabajadas por semana y el salario mensual. Sin embargo, esto podría cambiar dependiendo del sector y el tipo de trabajo, pero para este conjunto de datos, no parece que haya una relación lineal significativa.

### **3- Training\_Hours:**

### **Con Employee\_Satisfaction\_Score: 0.005813**

La correlación entre las horas de capacitación y la satisfacción del empleado es baja, lo que sugiere que las horas de capacitación no tienen un impacto importante en la satisfacción de los empleados, al menos desde una perspectiva lineal.

### **Con Performance\_Score: 0.002078**

La relación entre las horas de capacitación y el rendimiento también es baja. Esto podría indicar que, a pesar de que la capacitación es importante, no necesariamente se traduce directamente en una mejora del rendimiento de los empleados en este conjunto de datos.

### **4- Promotions:**

### **Con Employee\_Satisfaction\_Score: 0.001242**

La correlación entre promociones y satisfacción del empleado es muy baja. Esto sugiere que el número de promociones de un empleado no está estrechamente relacionado con su satisfacción en el trabajo.

## **Correlaciones Débiles o Irrelevantes:**

### **Gender y otras variables:**

La mayoría de las correlaciones entre el género y las otras variables son extremadamente bajas (cerca de 0). Esto indica que el género no tiene una influencia significativa sobre el rendimiento, el salario, la satisfacción o cualquier otra variable en este conjunto de datos. Esto puede ser una señal de que el modelo no debería incluir el género como una característica para predecir el rendimiento, ya que no aporta valor significativo.

### **Age y Performance\_Score:**

La correlación entre la edad y el rendimiento es muy baja (casi 0), lo que sugiere que la edad no está directamente relacionada con el rendimiento en este caso. Es posible que, en este conjunto de datos, no haya una tendencia clara de que los empleados más jóvenes o mayores tengan un mejor o peor desempeño.

### **Education\_Level y otras variables:**

Las correlaciones entre nivel educativo y otras variables como rendimiento, salario o satisfacción son bajas. Esto podría indicar que el nivel educativo por sí solo no tiene un impacto significativo en el desempeño del empleado. Sin embargo, esto podría depender de la naturaleza de las industrias y el tipo de trabajo que se esté analizando.

## Proyecto Final

### **Implicaciones para el Modelo:**

#### **Variables a Mantener:**

#### **Monthly\_Salary y Performance\_Score:**

Dado que la correlación entre Performance\_Score y Monthly\_Salary es moderada, estas dos variables podrían ser útiles en el modelo. Los salarios más altos tienden a estar relacionados con un mejor rendimiento, lo que hace que ambas variables sean importantes para predecir el rendimiento del empleado.

#### **Projects\_Handled y Team\_Size:**

Aunque la correlación no es muy fuerte, las variables relacionadas con el trabajo en equipo y la cantidad de proyectos gestionados podrían ser de interés en un análisis más detallado. Si hay algún impacto indirecto que no se refleja bien en una relación lineal directa, podría ser interesante explorar interacciones entre estas variables.

#### **Variables a Evaluar con Precaución:**

#### **Gender y Education\_Level:**

Las correlaciones con estas variables son muy débiles, por lo que no parecen ser predictores importantes para Performance\_Score. Si bien es importante tratar el género de manera ética y equitativa, parece que no tiene una relación directa con el rendimiento en este conjunto de datos. El nivel educativo tampoco tiene una correlación fuerte con el desempeño, por lo que podría no ser esencial para el modelo.

Con lo anterior descrito podemos tomar varias decisiones sobre el camino en el que queramos que vaya nuestro modelo y si queremos realizarle ciertos ajustes al mismo, los mencionamos a continuación.

## Proyecto Final

### **Posibles Ajustes en el Modelo:**

Eliminación de Variables con Correlación Baja: Las variables que tienen una correlación muy baja con el rendimiento o el salario (como Gender, Age, Promotions) pueden ser consideradas para ser eliminadas o combinadas con otras variables si no aportan valor predictivo.

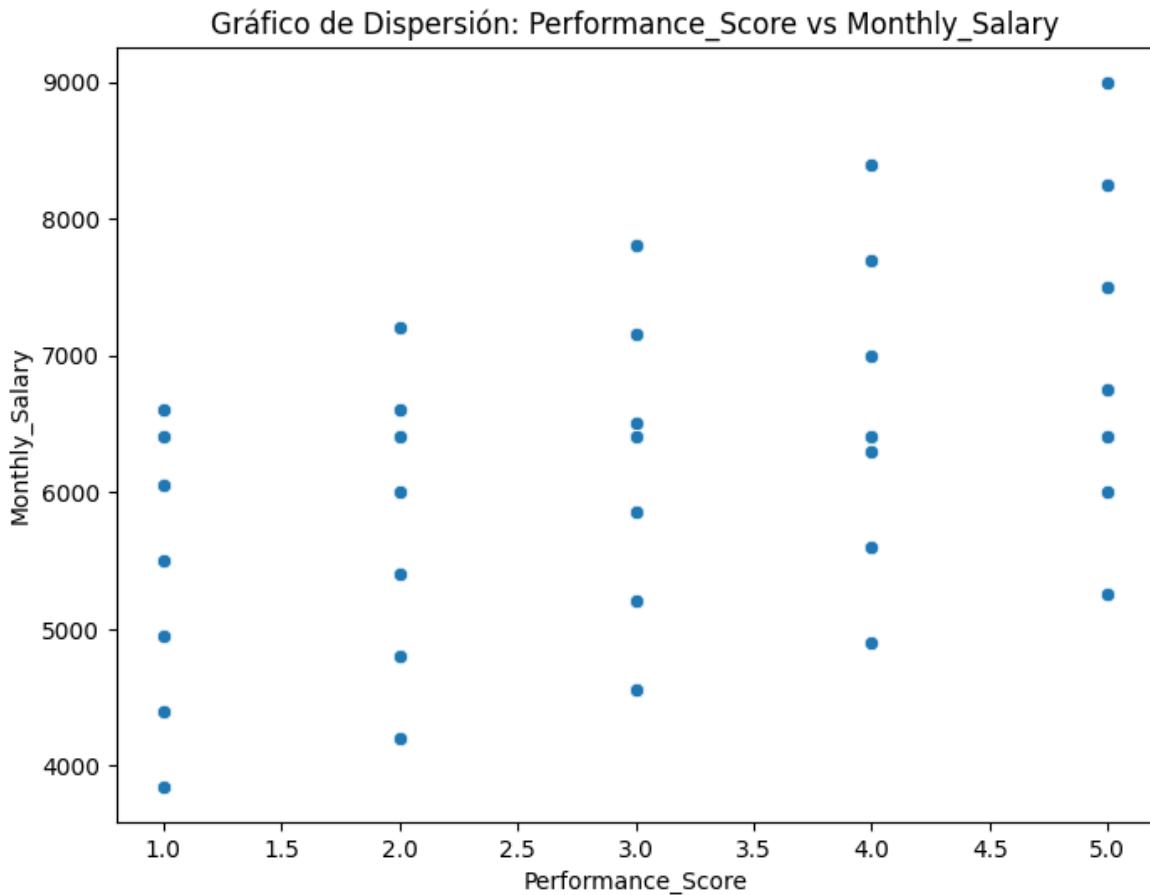
En general, las variables como Performance\_Score, Monthly\_Salary, y algunas relacionadas con la satisfacción y capacitación son las que parecen tener un mayor impacto en el rendimiento de los empleados. El modelo deberá considerar estas variables, mientras que variables como Gender, Age y Education\_Level tienen una correlación débil con el rendimiento y podrían no ser relevantes.

## Correlación entre Variables: Parejas de Variables

All identificar parejas de variables con correlaciones fuertes o moderadas, podemos investigar cómo estas relaciones podrían influir en la predicción de la variable objetivo (en este caso, Performance\_Score).

Para visualizar estas relaciones, se utilizan gráficos de dispersión, que permiten observar si existe alguna tendencia o patrón entre las variables. Los gráficos de dispersión son herramientas poderosas para detectar patrones no lineales o incluso relaciones débiles que podrían ser importantes para el análisis.

### Gráfico de dispersión de "Performance\_Score" y 'Monthly\_Salary'



# Proyecto Final

## **Patrones interesantes :**

### **Agrupación por niveles de Performance\_Score:**

Los datos están claramente segmentados por los niveles discretos de Performance\_Score (1, 2, 3, 4, 5), lo que facilita identificar diferencias de comportamiento entre estos grupos. Dentro de cada nivel, el rango de salarios (eje Y) es amplio, lo que indica que no todos los empleados con el mismo puntaje de desempeño tienen el mismo salario.

### **Tendencia general creciente:**

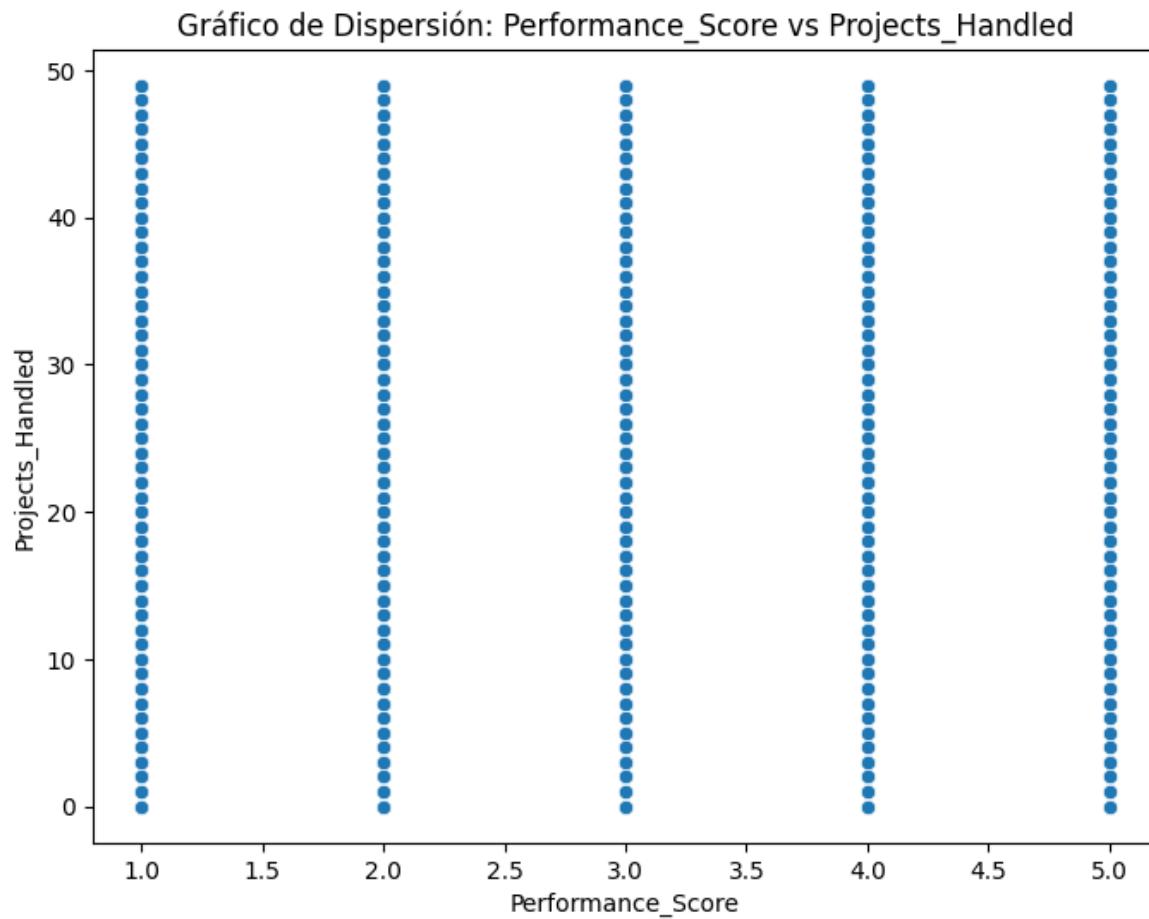
A medida que el Performance\_Score aumenta, los salarios más altos tienden a concentrarse en los niveles superiores (4 y 5). Sin embargo, existe una considerable superposición entre los rangos salariales de los distintos niveles de Performance\_Score. Por ejemplo: Algunos empleados con un puntaje bajo (1 o 2) tienen salarios similares a aquellos con puntajes altos (4 o 5). Esto sugiere que hay otras variables además del desempeño que explican el salario (como experiencia, antigüedad o puesto).

### **Desigualdad en la dispersión:**

En los niveles más bajos de Performance\_Score (1 y 2), los salarios tienden a concentrarse más cerca del rango inferior (4000-6000). Para los niveles más altos (4 y 5), los salarios muestran mayor dispersión, extendiéndose hasta los niveles más altos (>8000).

# Proyecto Final

## Gráfico de dispersión entre 'Performance\_Score' y 'Projects\_Handled'



**Patrones interesantes:**

**Segmentación por Performance\_Score:**

Los datos están agrupados en niveles bien definidos de Performance\_Score (1, 2, 3, 4, 5), como se esperaba debido a la naturaleza discreta de esta variable.

**Rango constante de Projects\_Handled:**

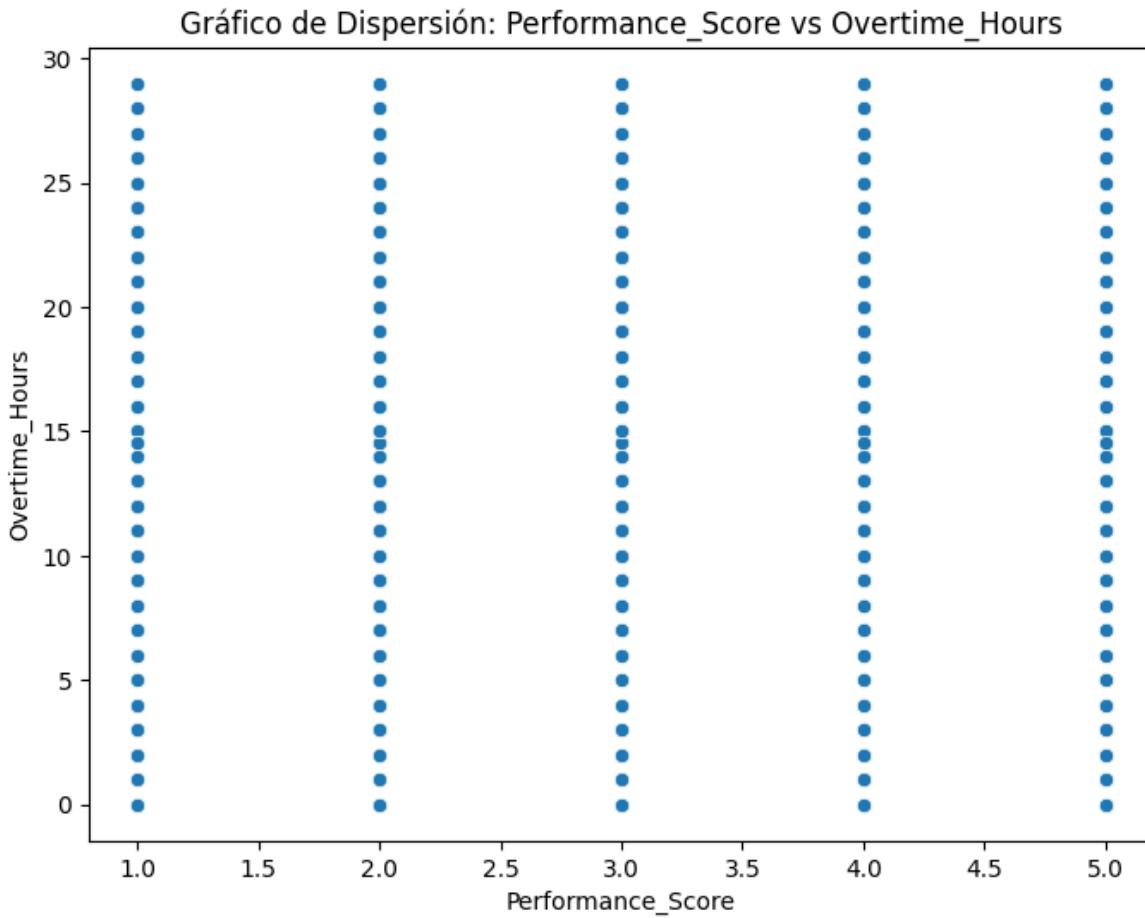
Dentro de cada nivel de Performance\_Score, hay una dispersión uniforme en la cantidad de proyectos manejados, desde 0 hasta 50. Esto sugiere que el número de proyectos no está correlacionado directamente con el puntaje de desempeño.

## Proyecto Final

### Ausencia de una tendencia clara:

No se observa un patrón creciente o decreciente claro entre Performance\_Score y Projects\_Handled. Esto implica que manejar más o menos proyectos no necesariamente afecta el puntaje de desempeño.

### Gráfico de dispersión entre 'Performance\_Score' y 'Overtime\_Hours'



### Distribución uniforme en Overtime\_Hours:

Para cada valor de Performance\_Score (de 1 a 5), los datos están dispersos uniformemente en el rango de horas extra (de 0 a 30). Esto sugiere que no parece haber una relación evidente entre el puntaje de desempeño y la cantidad de horas extra trabajadas.

## Proyecto Final

Ausencia de agrupamientos o tendencias:

No se identifican agrupamientos destacados ni tendencias ascendentes o descendentes. Esto indica que las dos variables son, posiblemente, independientes entre sí.

### **Implicaciones para el modelo:**

Las gráficas sugieren que la relación entre las variables analizadas y el Performance\_Score no es muy fuerte, especialmente para Projects\_Handled. Sin embargo, Monthly\_Salary tiene un patrón débil que podría complementarse con otras variables. Por lo tanto, ambas variables pueden incluirse en el modelo inicialmente, pero se debe prestar atención a su importancia predictiva durante el entrenamiento para decidir si conservarlas o descartarlas.

Monthly\_Salary podría ser una de las variables utilizadas para dividir los datos en ciertos niveles, especialmente en el rango medio o alto de desempeño, sin embargo, su alta variabilidad dentro de cada nivel de Performance\_Score implica que no será suficiente como predictor por sí solo.

En el caso de Performance\_Score Dado que no hay una relación directa clara, es probable que tenga baja importancia predictiva para Performance\_Score. Si cruzamos esta variable con otras, como Overtime\_Hours o Training\_Hours, podría tener un efecto de mayor impacto. Podría ser relevante en ciertos subgrupos. Por ejemplo, empleados con alto desempeño y muchos proyectos podrían requerir entrenamiento o tener características distintas.

## Proyecto Final

### Análisis de Valores Atípicos (Outliers):

Primero procedemos a dar una breve descripción de los valores con los que estamos tratando. Los outliers son datos que se desvían significativamente de los valores esperados o de la mayoría de los datos en un conjunto. Representan puntos inusuales o extremos que pueden ser causados por errores, variaciones naturales o eventos atípicos. Identificarlos es importante porque pueden influir en los resultados de los análisis estadísticos o modelos.

```
[70] columna_numerica = [
    'Age', 'Years_At_Company', 'Performance_Score', 'Monthly_Salary',
    'Work_Hours_Per_Week', 'Projects_Handled', 'Overtime_Hours', 'Sick_Days',
    'Team_Size', 'Training_Hours', 'Promotions', 'Employee_Satisfaction_Score'
]

# Calcular el IQR y detectar outliers para cada columna numérica
outliers = {}
for column in columna_numerica:
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Filtrar outliers
    outliers[column] = df[(df[column] < lower_bound) | (df[column] > upper_bound)]

    # Imprimir la cantidad de outliers detectados por columna
print(f'Outliers en {column}: {outliers[column].shape[0]}')

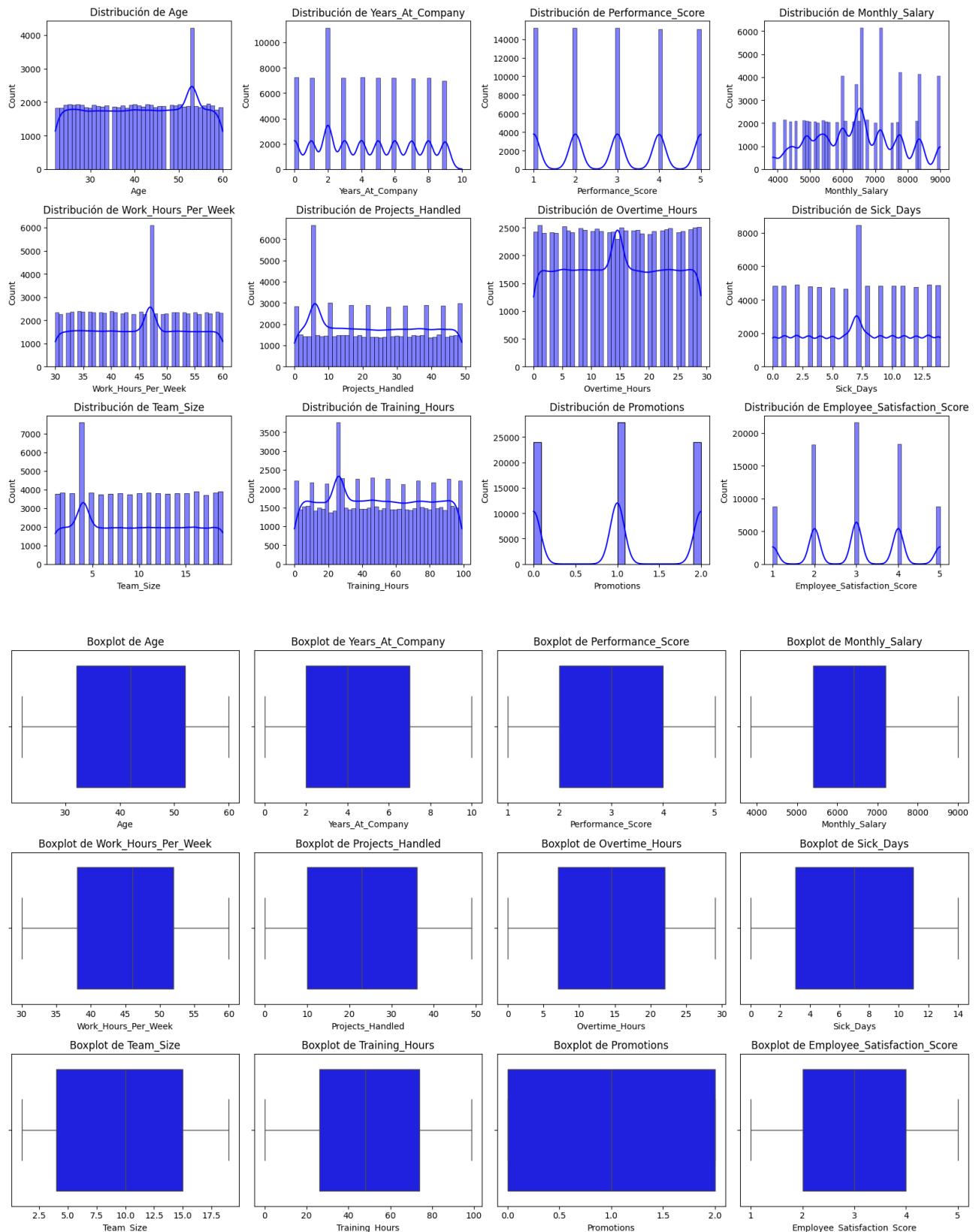
→ Outliers en Age: 0
Outliers en Years_At_Company: 0
Outliers en Performance_Score: 0
Outliers en Monthly_Salary: 0
Outliers en Work_Hours_Per_Week: 0
Outliers en Projects_Handled: 0
Outliers en Overtime_Hours: 0
Outliers en Sick_Days: 0
Outliers en Team_Size: 0
Outliers en Training_Hours: 0
Outliers en Promotions: 0
Outliers en Employee_Satisfaction_Score: 0

[71] # Usar un multiplicador de 3 en lugar de 1.5 para el IQR para ser más permisivo
outliers = {}
for column in columna_numerica:
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 3 * IQR # Multiplicador de 3
    upper_bound = Q3 + 3 * IQR
    outliers[column] = df[(df[column] < lower_bound) | (df[column] > upper_bound)]

    print(f'Outliers en {column}: {outliers[column].shape[0]}')

→ Outliers en Age: 0
Outliers en Years_At_Company: 0
Outliers en Performance_Score: 0
Outliers en Monthly_Salary: 0
Outliers en Work_Hours_Per_Week: 0
Outliers en Projects_Handled: 0
Outliers en Overtime_Hours: 0
Outliers en Sick_Days: 0
Outliers en Team_Size: 0
Outliers en Training_Hours: 0
Outliers en Promotions: 0
Outliers en Employee_Satisfaction_Score: 0
```

# Proyecto Final



# Proyecto Final

## Identificación de Outliers

Para el análisis de valores atípicos, se utilizó un enfoque basado en el método estadístico del rango intercuartílico (IQR). Este método considera como outliers aquellos datos que están por debajo de  $Q1 - 1.5 \times IQR$  o por encima de  $Q3 + 1.5 \times IQR$ . Este enfoque es ampliamente aceptado y permite identificar valores extremos sin depender de la distribución de los datos. Además, se realizaron verificaciones visuales utilizando gráficos de dispersión y boxplots para respaldar las observaciones.

## Resultados del Análisis

Después de aplicar este método al conjunto de datos, no se identificaron valores fuera de los rangos establecidos para ninguna de las columnas numéricas analizadas. Esto indica que los datos están bien distribuidos dentro de sus rangos naturales y no presentan valores extremos que puedan considerarse atípicos.

## Tratamiento de Outliers

Dado que no se detectaron outliers, no fue necesario implementar estrategias de tratamiento como eliminación, transformación o imputación. Este resultado asegura que los datos analizados son representativos y no están influenciados por valores extremos que puedan sesgar los análisis o modelos posteriores.

La ausencia de valores atípicos refuerza la calidad del conjunto de datos y asegura la validez de los análisis estadísticos y modelos que se desarrollen con base en este. Esto permite avanzar en el proyecto con confianza.

# Proyecto Final

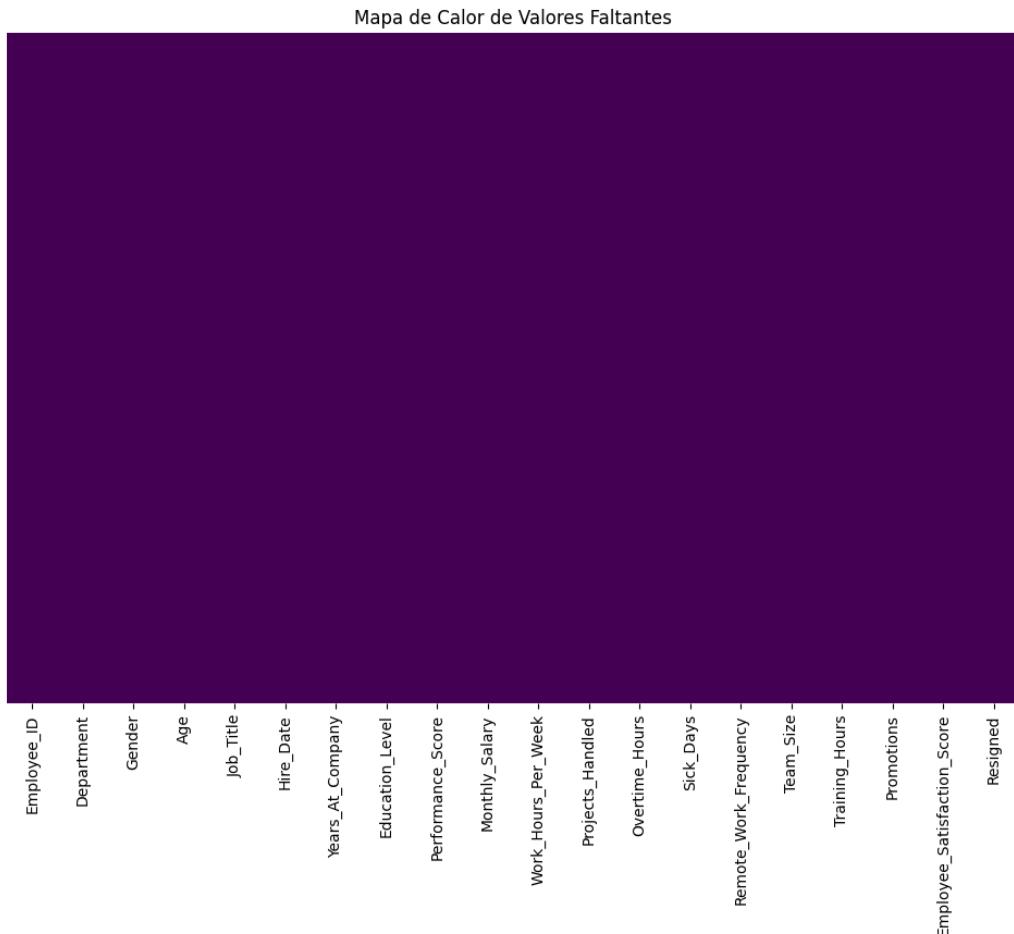
## Análisis de Valores Faltantes

Los valores faltantes son datos ausentes en un conjunto de datos, representados comúnmente como espacios vacíos, nulos o marcados con etiquetas como "NaN" o "NULL". Estos pueden surgir por errores en la recopilación de datos, problemas técnicos o porque la información simplemente no estaba disponible. Identificarlos y tratarlos adecuadamente es crucial, ya que pueden afectar la precisión de los análisis y modelos predictivos.

```
# Ver la cantidad de datos faltantes por columna
missing_data = df.isnull().sum()
missing_percentage = (missing_data / len(df)) * 100
missing_info = pd.DataFrame({'Missing Values': missing_data, 'Percentage': missing_percentage})

# Mostrar las columnas con valores faltantes
print("Datos faltantes por columna:")
print(missing_info[missing_info['Missing Values'] > 0])

# Visualizar los valores faltantes con un mapa de calor
plt.figure(figsize=(12, 8))
sns.heatmap(df.isnull(), cbar=False, cmap='viridis', yticklabels=False)
plt.title("Mapa de Calor de Valores Faltantes")
plt.show()
```



# Proyecto Final

## **Identificación de Datos Faltantes**

Se realizó un análisis exhaustivo para identificar la presencia de valores faltantes en el conjunto de datos, utilizando herramientas estadísticas y visualizaciones como un mapa de calor. En la visualización presentada, se observa que no hay áreas destacadas que representen datos ausentes, lo que indica que todas las columnas y filas contienen información completa.

## **Estrategia de Imputación o Eliminación**

Dado que no se encontraron valores faltantes en el análisis, no fue necesario implementar estrategias para su tratamiento. Esto implica que no se realizó imputación (rellenar con la media, mediana u otros valores) ni eliminación de filas o columnas.

## **Justificación**

**Calidad y completitud del conjunto de datos:** La ausencia de valores faltantes garantiza que el dataset es completo, lo cual simplifica el análisis y asegura que toda la información está disponible para realizar modelos y evaluaciones.

**Evitar sesgos en los datos:** No tener que imputar o eliminar valores asegura que no se introduzcan sesgos artificiales derivados del tratamiento de datos faltantes, lo que preserva la integridad de los resultados.

**Eficiencia en el procesamiento:** Sin valores faltantes, se elimina la necesidad de aplicar técnicas adicionales para gestionar datos incompletos, lo que permite un flujo de trabajo más eficiente y directo.

El hecho de contar con un conjunto de datos sin valores faltantes fortalece la confianza en el análisis y las conclusiones que se deriven de este. Esto refleja una recopilación de datos robusta y asegura que no se comprometerá la calidad de los resultados por problemas de datos incompletos.

# Proyecto Final

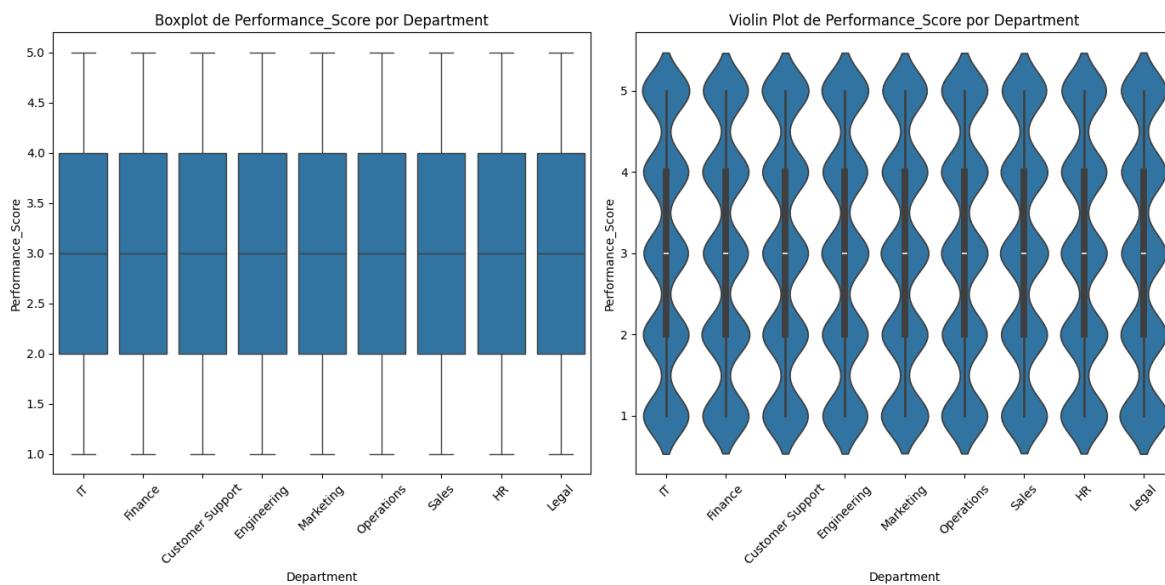
## Relación entre Variables categóricas y numéricas

Realizaremos un análisis comparativo para explorar cómo se distribuye una variable numérica (como Performance\_Score) dentro de diferentes categorías (por ejemplo, Department o Gender). Usaremos gráficos de caja (boxplots) o gráficos de violín para:

Visualizar la distribución de datos en cada categoría: Estos gráficos muestran valores como la mediana, el rango intercuartílico y posibles valores atípicos.

Comparar diferencias entre grupos: Podremos identificar si existen variaciones significativas en la variable numérica según la categoría, lo que puede revelar patrones, desigualdades o áreas de interés.

Este análisis nos ayuda a entender la relación entre variables categóricas y numéricas y a encontrar diferencias clave que podrían ser importantes para la interpretación o el modelado.



# Proyecto Final

## Relación entre Department y Performance Score:

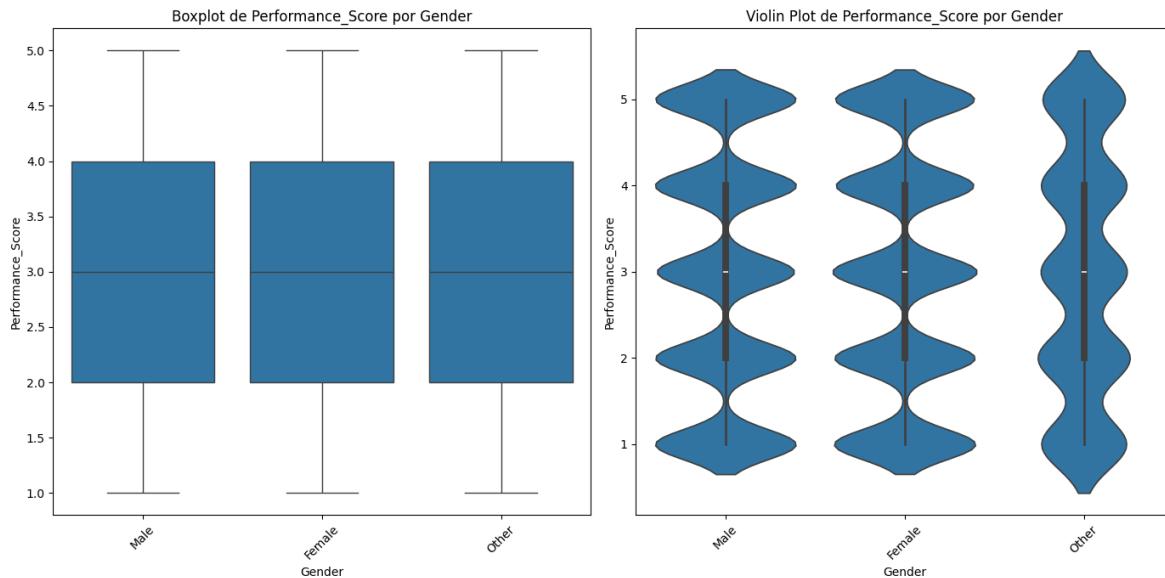
### Boxplot:

La mediana de las puntuaciones de desempeño es consistente en todos los departamentos (3). Algunos departamentos muestran una mayor dispersión en las puntuaciones (por ejemplo, Engineering y Operations), mientras que otros son más uniformes.

### Violin Plot:

La densidad de puntuaciones es alta alrededor de la mediana en todos los departamentos. Departamentos como Engineering tienen una distribución más amplia, lo que sugiere una mayor variabilidad en las puntuaciones.

No hay diferencias significativas en las puntuaciones de desempeño entre departamentos, aunque la dispersión varía ligeramente según el departamento



# Proyecto Final

## Relación entre Gender y Performance Score:

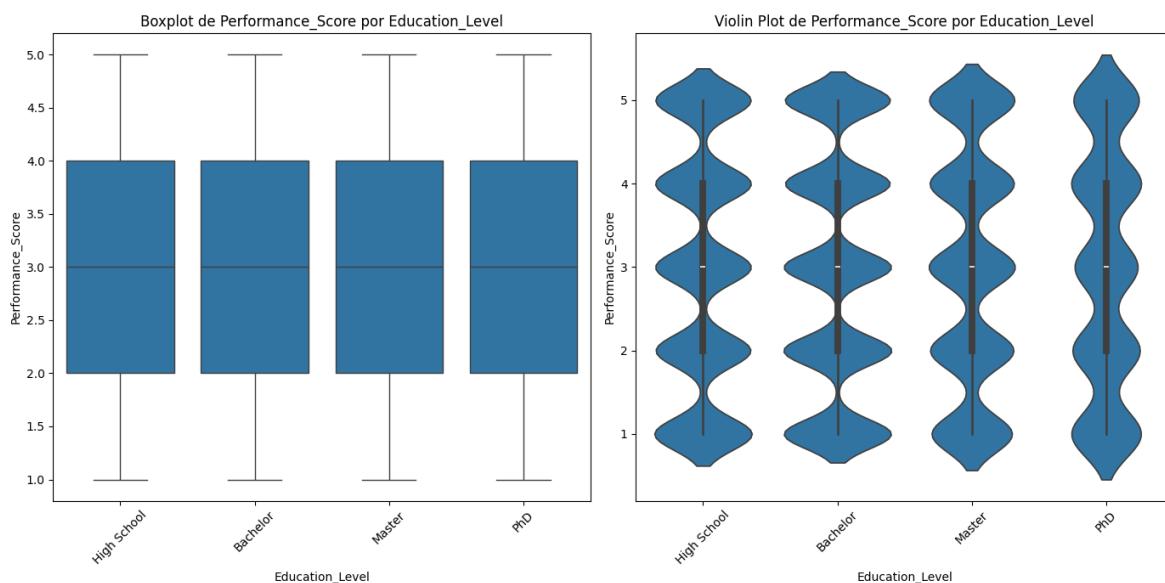
### Boxplot:

La mediana de las puntuaciones es similar entre géneros (3). La dispersión de las puntuaciones es casi idéntica para todos los géneros, con un rango ligeramente mayor en el grupo "Other".

### Violin Plot:

La densidad de las puntuaciones se concentra fuertemente alrededor de la mediana para todos los géneros. Las formas de los violines no muestran diferencias sustanciales entre los géneros.

No se observan diferencias significativas en las puntuaciones de desempeño entre géneros. Esto sugiere que el género no influye en el desempeño según los datos.



## Relación entre Education Level y Performance Score:

### Boxplot:

La mediana (3) y la dispersión de las puntuaciones son similares para todos los niveles educativos (*High School*, *Bachelor*, *Master*, y *PhD*).

## Proyecto Final

### **Violin Plot:**

Las distribuciones son muy similares, con densidades altas cerca de la mediana en todos los niveles educativos. No se observan distribuciones anómalas o diferenciación significativa entre categorías.

El nivel educativo no parece estar relacionado con las puntuaciones de desempeño. Esto puede sugerir que otros factores (habilidades prácticas, experiencia, etc.) son más relevantes para determinar el desempeño.

### Analisis general

Las tres variables categóricas (Department, Gender, y Education Level) muestran distribuciones de desempeño muy similares en términos de mediana, dispersión y densidad. No hay indicios de diferencias significativas entre grupos en cuanto al Performance Score. Esto podría implicar que las evaluaciones de desempeño están estandarizadas o que las puntuaciones son influenciadas por otros factores no representados en estas gráficas.

## Observaciones y Hallazgos Importantes

### Variable Objetivo y Variables Predictoras

La variable objetivo que se busca predecir es **Performance\_Score**, que representa el desempeño de los empleados. Para identificar las variables que más afectan a esta variable, se utilizó un análisis de correlación y visualizaciones.

#### 1. Heatmap y Coeficientes de Correlación:

Un heatmap de correlación mostró que:

Monthly\_Salary tiene una correlación moderada positiva con Performance\_Score, lo que sugiere que los empleados con salarios más altos tienden a obtener mejores evaluaciones.

Variables como Sick\_Days, Projects\_Handled, y Employee\_Satisfaction\_Score tienen correlaciones extremadamente bajas con Performance\_Score, indicando una relación mínima lineal.

Age y Gender presentan una correlación negativa débil, lo que implica que su influencia directa es insignificante.

### Hallazgos Clave

#### Relaciones Relevantes:

Monthly\_Salary es la variable más relevante debido a su correlación positiva moderada con Performance\_Score.

Las variables categóricas como Department y Education\_Level presentan distribuciones homogéneas de desempeño, lo que sugiere que las evaluaciones están estandarizadas entre estos grupos.

#### Relaciones Débiles o Inesperadas:

Sick\_Days y Projects\_Handled tienen correlaciones muy bajas con el desempeño, lo que indica que probablemente no sean buenos predictores individuales.

## Proyecto Final

Employee\_Satisfaction\_Score tiene una relación lineal baja con Performance\_Score, pero podría influir de manera no lineal o a través de interacciones con otras variables.

### Anomalías y Patrones Interesantes:

No se detectaron valores atípicos significativos ni distribuciones anómalas en las variables analizadas. Existe una ligera variabilidad en las puntuaciones de desempeño entre los departamentos, pero no es suficiente para considerarla una diferencia significativa.

## Implicaciones para el Modelo

### Selección de Variables:

Monthly\_Salary: Por su correlación moderada positiva, es un predictor clave para el desempeño.

Variables categóricas: Como Department y Education\_Level, ya que los árboles de decisión pueden manejar fácilmente variables categóricas.

Employee\_Satisfaction\_Score: Aunque tiene una correlación lineal baja, podría aportar valor a través de relaciones no lineales o interacciones con otras variables.

Variables con menor relevancia inicial: Sick\_Days, Team\_Size, y Projects\_Handled: Estas variables tienen correlaciones extremadamente bajas con el desempeño, pero podrían considerarse en modelos más complejos si se identifican interacciones significativas.

### 3. Modelo de machine Learning

#### Descripción del modelo y justificación:

##### Árbol de decisiones

Decidimos usar un árbol de decisiones porque este modelo es altamente interpretativo y eficiente para capturar relaciones no lineales entre variables, lo que lo hace ideal para explorar cómo diversos factores, tanto numéricos como categóricos, afectan al **Performance\_Score**. Además, los árboles no requieren una preparación de datos extensa, como la normalización o codificación, y permiten identificar fácilmente las variables más relevantes, proporcionando reglas claras que pueden ser interpretadas y aplicadas en un contexto organizacional. Esto facilita no solo la predicción del desempeño, sino también la generación de insights accionables.

La importancia de las características calculada por el árbol de decisión servirá para confirmar qué variables son predictoras clave y cuáles tienen un impacto menor. Las reglas generadas por el árbol ofrecerán ideas sobre cómo las variables categóricas y numéricas interactúan para afectar el desempeño.

El análisis exploratorio de datos muestra que `Monthly_Salary` es la variable más relevante para predecir `Performance_Score`, mientras que variables como `Sick_Days` y `Projects_Handled` tienen poca relación directa con el desempeño. Sin embargo, el uso de un árbol de decisión permitirá explorar interacciones complejas y relaciones no lineales que podrían no ser evidentes en el análisis de correlación inicial.

#### Implementación y Entrenamiento:

El proceso de implementación y entrenamiento del modelo de Random Forest para predecir `Performance_Score` se divide en varias fases clave: preparación de los datos, implementación del modelo, evaluación y ajustes.

# Proyecto Final

## División de los Datos:

Antes de entrenar el modelo, es importante dividir los datos en dos conjuntos:

**Conjunto de Entrenamiento:** Este conjunto se utiliza para entrenar el modelo, es decir, para ajustar los parámetros internos del modelo a partir de las relaciones que existen entre las características predictoras (X) y la variable objetivo (Performance\_Score).

**Conjunto de Prueba:** Este conjunto se utiliza para evaluar el rendimiento del modelo una vez que ha sido entrenado. Es crucial que el conjunto de prueba no se utilice durante el proceso de entrenamiento para garantizar que el modelo generalice correctamente a datos no vistos.

```
[56] #Variable sobre la que vamos a trabajar Performance_Score  
#Variables que nos ayudarán, todas menos Employee_ID  
  
[57] from sklearn.model_selection import train_test_split  
  
# Definir las variables predictoras (X) y la variable objetivo (y)  
X = df2.drop(columns=['Performance_Score', 'Job_Title', 'Hire_Date', 'Department', 'Employee_ID'])  
y = df2['Performance_Score']  
  
# Dividir los datos en entrenamiento y prueba (80% entrenamiento, 20% prueba)  
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

En el código, se elimina Performance\_Score, Job\_Title, Hire\_Date, Department y Employee\_ID de las características predictoras. Performance\_Score es la variable objetivo (lo que queremos predecir), y las otras columnas se eliminan para evitar que interfieran con el modelo.

## Entrenamiento del modelo:

El modelo utilizado es un Random Forest Regressor, que es un conjunto de árboles de decisión que, mediante un proceso de bagging, mejora la capacidad predictiva y reduce el riesgo de sobreajuste.

```
[58] from sklearn.ensemble import RandomForestRegressor  
  
# Inicializar el modelo de Random Forest  
model = RandomForestRegressor(random_state=42)  
  
# Entrenar el modelo con los datos de entrenamiento  
model.fit(X_train, y_train)
```

En este paso:

## Proyecto Final

Inicializamos el modelo de Random Forest con el parámetro `random_state=42` (El número 42 es simplemente un número arbitrario utilizado comúnmente como semilla en muchos proyectos de programación y machine learning.) para garantizar la reproducibilidad de los resultados y utilizamos el método `.fit()` para entrenar el modelo con los datos de entrenamiento (`X_train` y `y_train`).

### Evaluación del modelo:

Una vez entrenado el modelo, es crucial evaluar su rendimiento utilizando métricas que nos indiquen qué tan bien está prediciendo el `Performance_Score`.

#### Error Cuadrático Medio (MSE):

El MSE mide la diferencia promedio entre las predicciones del modelo y los valores reales. Mientras más bajo sea el MSE, mejor será el modelo. Es importante que el MSE sea bajo porque indica que las predicciones están cerca de los valores reales.

#### Coeficiente de Determinación ( $R^2$ ):

El  $R^2$  nos indica qué porcentaje de la variabilidad en `Performance_Score` ha sido explicada por el modelo. Un valor cercano a 1 significa que el modelo ha explicado la mayoría de la variabilidad.

```
[59] from sklearn.metrics import mean_squared_error, r2_score

# Hacer predicciones sobre el conjunto de prueba
y_pred = model.predict(X_test)

# Evaluar el modelo con el Error Cuadrático Medio (MSE)
mse = mean_squared_error(y_test, y_pred)
print(f"Error cuadrático medio (MSE): {mse:.2f}")

# Evaluar el modelo con el Coeficiente de Determinación (R²)
r2 = r2_score(y_test, y_pred)
print(f"Coeficiente de determinación (R²): {r2:.2f}")

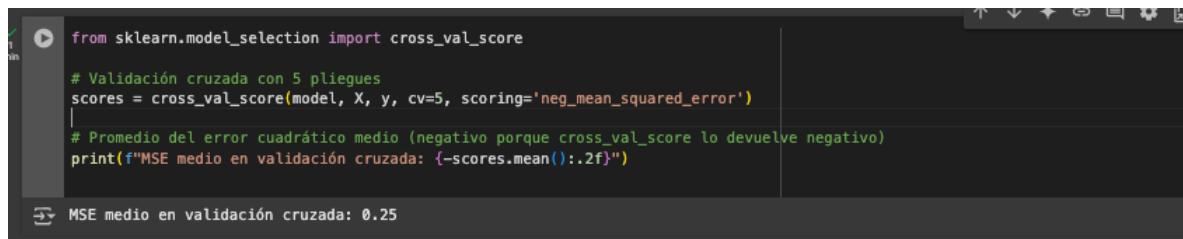
Error cuadrático medio (MSE): 0.24
Coeficiente de determinación (R²): 0.88
```

#### Validación Cruzada:

Además de evaluar el modelo con un conjunto de prueba, es recomendable usar validación cruzada para obtener una evaluación más detallada del rendimiento del modelo. La validación cruzada permite entrenar y evaluar el modelo en diferentes

## Proyecto Final

subconjuntos de los datos, lo que ayuda a reducir el riesgo de sobreajuste.



```
from sklearn.model_selection import cross_val_score
# Validación cruzada con 5 pliegues
scores = cross_val_score(model, X, y, cv=5, scoring='neg_mean_squared_error')
# Promedio del error cuadrático medio (negativo porque cross_val_score lo devuelve negativo)
print(f"MSE medio en validación cruzada: {-scores.mean():.2f}")

MSE medio en validación cruzada: 0.25
```

### Resultados:

MSE en conjunto de prueba: 0.24

R<sup>2</sup> en conjunto de prueba: 0.88

MSE medio en validación cruzada: 0.25

Estos resultados indican que el modelo tiene un buen desempeño, explicando un 88% de la variabilidad de Performance\_Score.

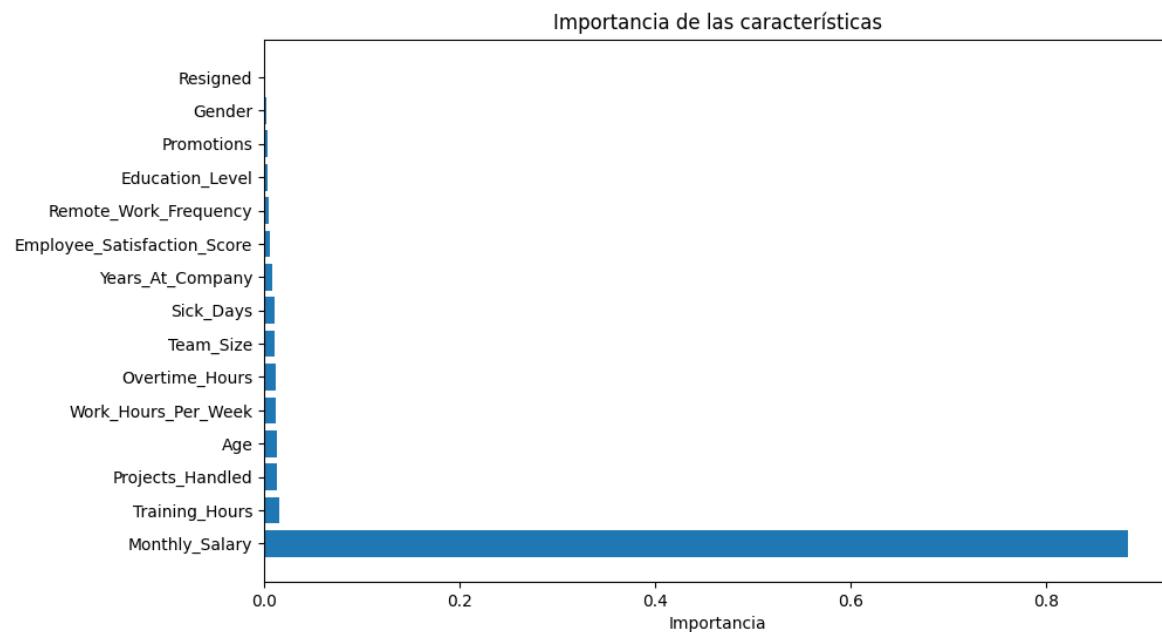
Generamos una grafica de barras para ver más a detalle la relacion directa de las variables predictoras y su importancia en el modelo de predicción.

El análisis de la **importancia de las características** mediante el gráfico de barras horizontal nos ha permitido identificar las variables más influyentes en la predicción del **Performance\_Score** utilizando el modelo de **Random Forest**. La visualización muestra de manera clara cuáles son las características que aportan más valor al modelo y cuáles tienen una contribución menor.

A su vez este análisis ayuda a **interpretar el modelo** de manera más fácil, ya que revela qué factores están impulsando la predicción del desempeño de los empleados. Esto puede ser útil no solo para la mejora del modelo, sino también para ofrecer una **perspectiva clara a los responsables de la toma de decisiones en la empresa**, quienes pueden utilizar esta información para enfocarse en áreas clave como el salario, la capacitación y la gestión de proyectos.

La visualización de la **importancia de las características** permite mejorar la comprensión del modelo, optimizar la selección de variables y guiar las decisiones estratégicas en torno a los factores que más influyen en el rendimiento de los empleados.

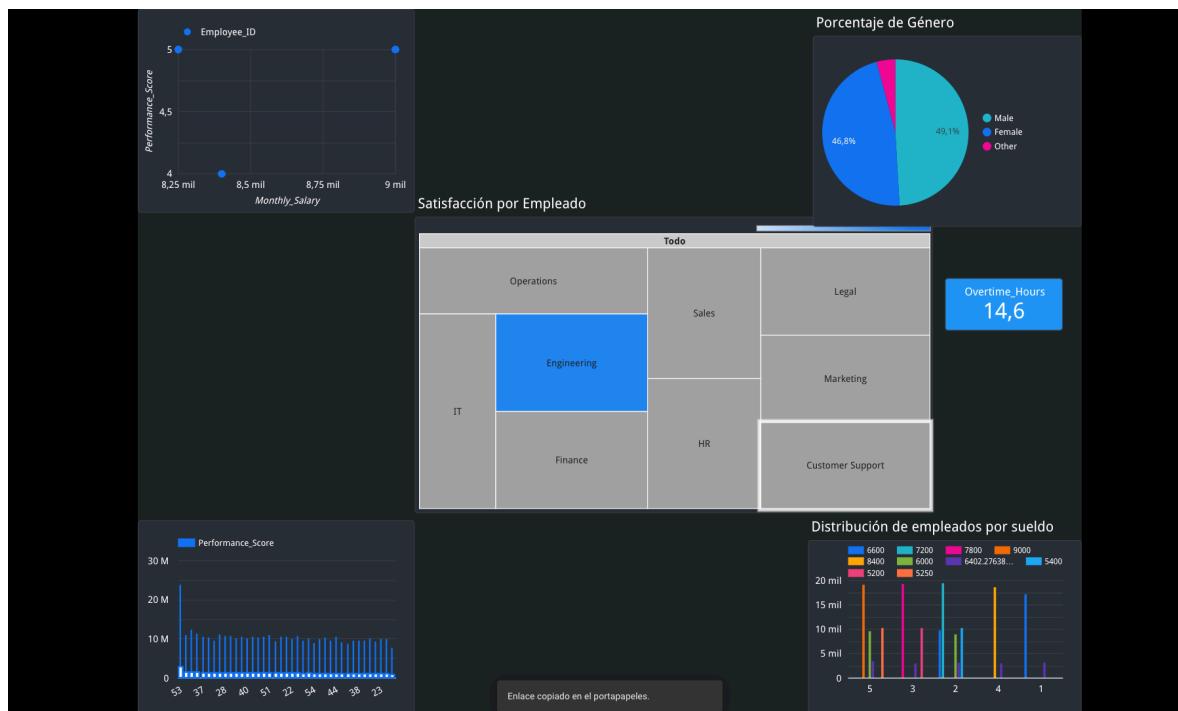
# Proyecto Final



# Proyecto Final

## 5. Dashboard

Es una herramienta visual que organiza y presenta de manera clara y concisa datos relevantes, métricas y KPIs (Indicadores Clave de Desempeño) para facilitar la toma de decisiones. Suele estar compuesto por gráficos, tablas y otros elementos visuales que permiten a los usuarios analizar el estado de un sistema o proceso en tiempo real.



### Porcentaje de Género:

Este gráfico circular proporciona una distribución clara de los empleados según su género (masculino, femenino y otros). Permite identificar rápidamente la composición de género dentro de la empresa.

### Satisfacción del Empleado:

Este gráfico muestra los niveles de satisfacción de los empleados en relación con diferentes factores, como el salario o las condiciones de trabajo.

### Distribución por Departamentos:

El diagrama de árbol muestra la estructura organizacional de la empresa y la cantidad de empleados en cada departamento.

## Proyecto Final

### **Rendimiento:**

El gráfico de líneas muestra el rendimiento general de los empleados a lo largo del tiempo. Podría estar relacionado con metas, objetivos o indicadores clave de desempeño (KPIs).

### **Distribución de Empleados por Sueldo:**

Este gráfico de barras múltiples compara los rangos salariales entre diferentes categorías de empleados. Permite identificar brechas salariales y desigualdades.

### **Uso y beneficios:**

El dashboard facilita el análisis y la toma de decisiones basadas en datos sobre el Performance\_Score de los empleados. Ayuda a visualizar de manera clara cómo las distintas variables influyen en el rendimiento, permite detectar patrones y anomalías, segmenta el desempeño según diferentes categorías y proporciona una herramienta eficiente para la presentación de resultados. En última instancia, un dashboard mejora la toma de decisiones estratégicas y operativas dentro de la organización, optimizando la gestión de recursos humanos y mejorando el desempeño organizacional.

## 6. Conclusiones y futuras líneas de trabajo

### Resumen de los Hallazgos Principales

#### Análisis Exploratorio de Datos (EDA):

Distribución equilibrada: La mayoría de las variables numéricas y categóricas presentan distribuciones relativamente equilibradas, sin sesgos significativos.

Correlaciones débiles: La mayoría de las correlaciones entre variables son débiles, lo que sugiere que las relaciones entre ellas son complejas y no lineales.

Monthly\_Salary como mejor predictor: El salario mensual muestra una correlación moderada positiva con el desempeño, lo que indica que los empleados con salarios más altos tienden a tener mejor desempeño.

Ausencia de valores atípicos: No se encontraron valores atípicos significativos, lo que indica una buena calidad de los datos.

#### Modelo de Machine Learning (Random Forest):

Buen desempeño: El modelo de Random Forest logró explicar un 88% de la variabilidad en el desempeño de los empleados, lo que indica un buen ajuste a los datos.

Importancia de las variables: El modelo destacó la importancia de Monthly\_Salary como el predictor más significativo, seguido de otras variables como YearsAtCompany y TrainingHours.

Interpretabilidad: Los árboles de decisión son altamente interpretables, lo que permite entender las reglas de decisión y la lógica detrás de las predicciones.

## **Cumplimiento de los Objetivos Iniciales**

El análisis realizado ha cumplido con los objetivos iniciales de comprender la distribución de los datos: Se han identificado las características principales de los datos, como la distribución de los empleados por departamento, género y nivel educativo.

Identificar relaciones entre variables: Se han explorado las correlaciones entre las variables, especialmente entre el desempeño y otras variables como el salario y la experiencia.

Construir un modelo predictivo: Se ha desarrollado un modelo de Random Forest que puede predecir el desempeño de los empleados con una precisión razonable.

Interpretar los resultados: El modelo y las visualizaciones permiten comprender qué factores influyen en el desempeño de los empleados y cómo se relacionan entre sí.

## **Posibles Mejoras y Direcciones Futuras**

Mayor profundidad en el análisis de variables categóricas: Explorar interacciones entre variables categóricas y numéricas mediante técnicas como la codificación one-hot y la creación de variables dummy.

Incorporar variables temporales: Analizar cómo el desempeño evoluciona a lo largo del tiempo y si existen patrones estacionales o tendencias a largo plazo.

En resumen, el análisis realizado proporciona una base sólida para comprender los factores que influyen en el desempeño de los empleados. Sin embargo, hay oportunidades para profundizar en el análisis y obtener una comprensión más completa de los datos. Al

## Proyecto Final

implementar las recomendaciones mencionadas, se pueden obtener resultados aún más precisos y relevantes para la toma de decisiones en la organización.

## 6. Referencias

- Kaggle. (2021). *Employee performance data* [Conjunto de datos]. Kaggle. <https://www.kaggle.com/datasets/xyz/employee-performance-data>
- Tukey, J. W. (1977). *Exploratory data analysis*. Addison-Wesley.
- Kohavi, R. (1995). *A study of cross-validation and bootstrap for accuracy estimation and model selection*. Proceedings of the 14th International Joint Conference on Artificial Intelligence, 1137–1143.
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>

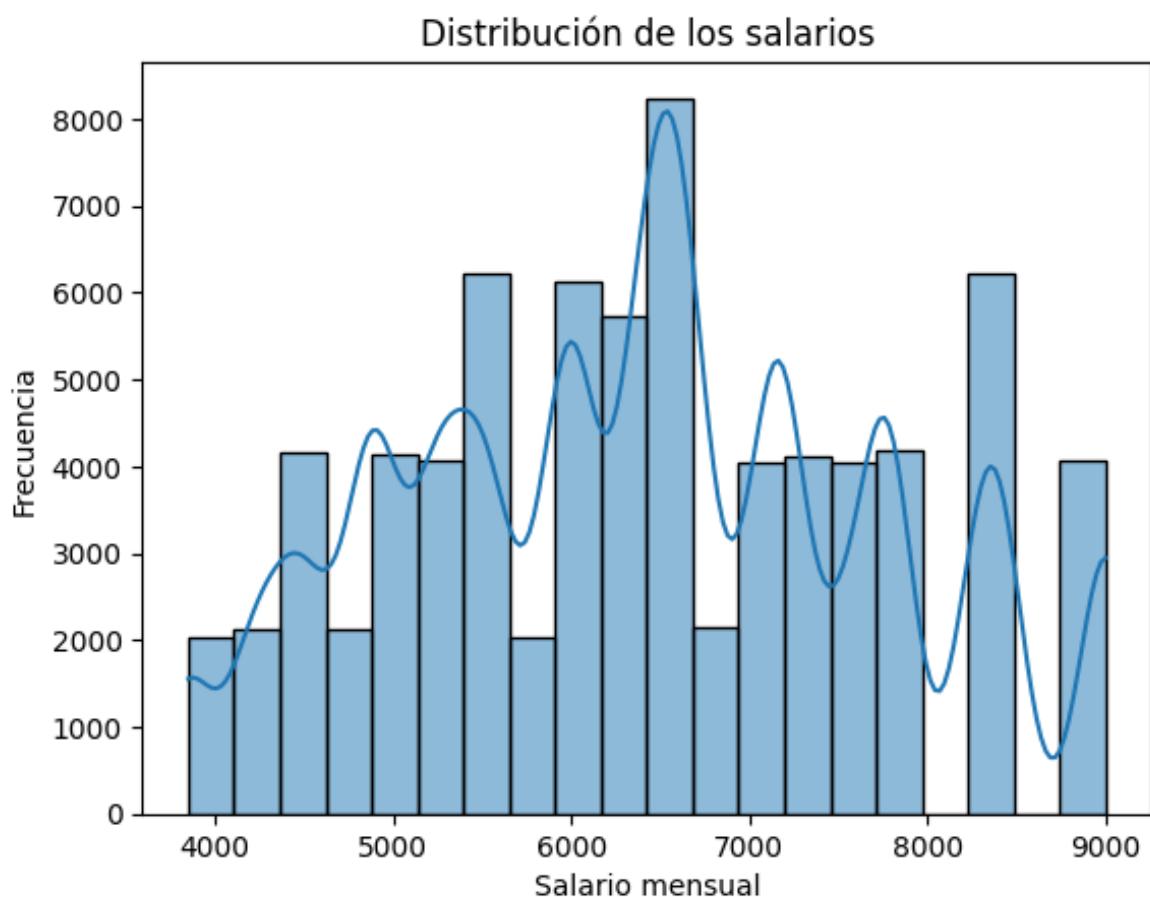
## Proyecto Final

### Anexos

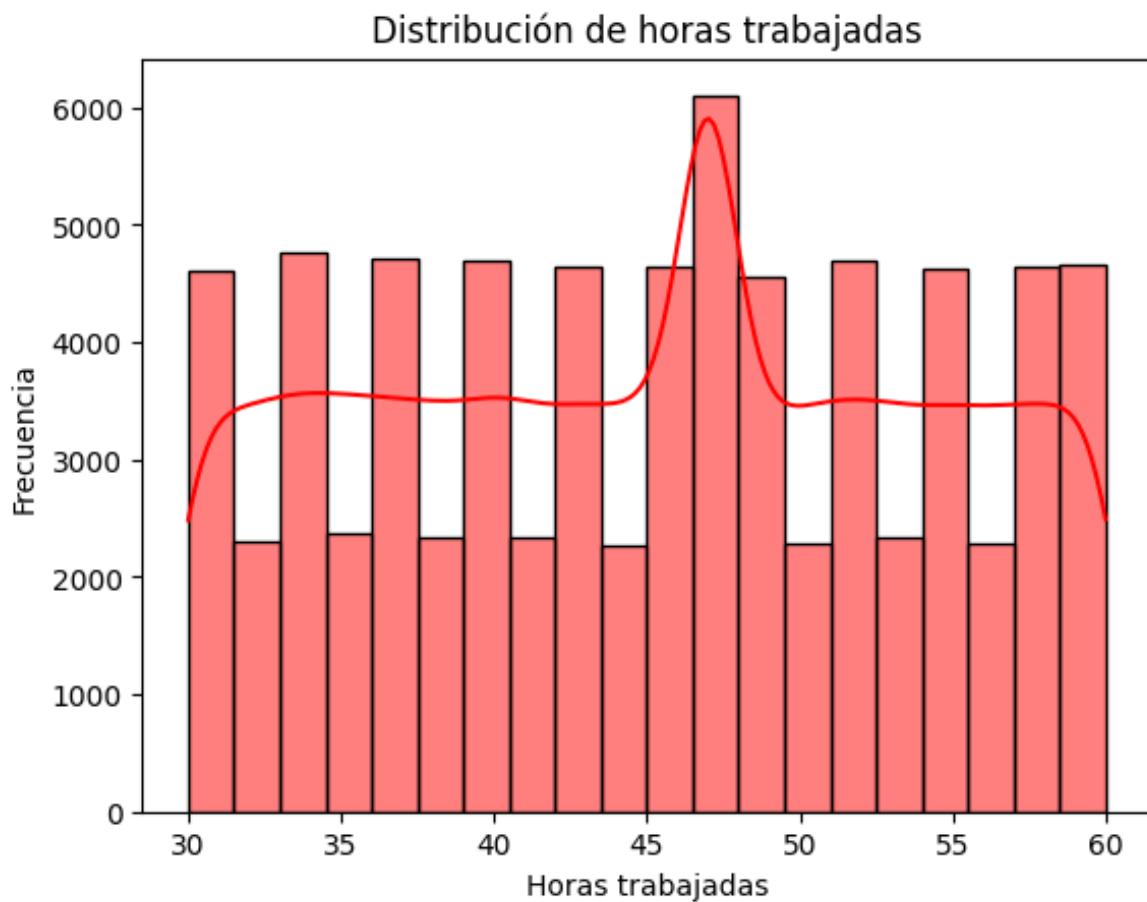
Código Fuente para la implementacion del modelo o limpieza de datos

[https://colab.research.google.com/drive/1FgQV\\_Nq-Own9BiQMVsJdsEj14dIUpBNq?usp=sharing](https://colab.research.google.com/drive/1FgQV_Nq-Own9BiQMVsJdsEj14dIUpBNq?usp=sharing)

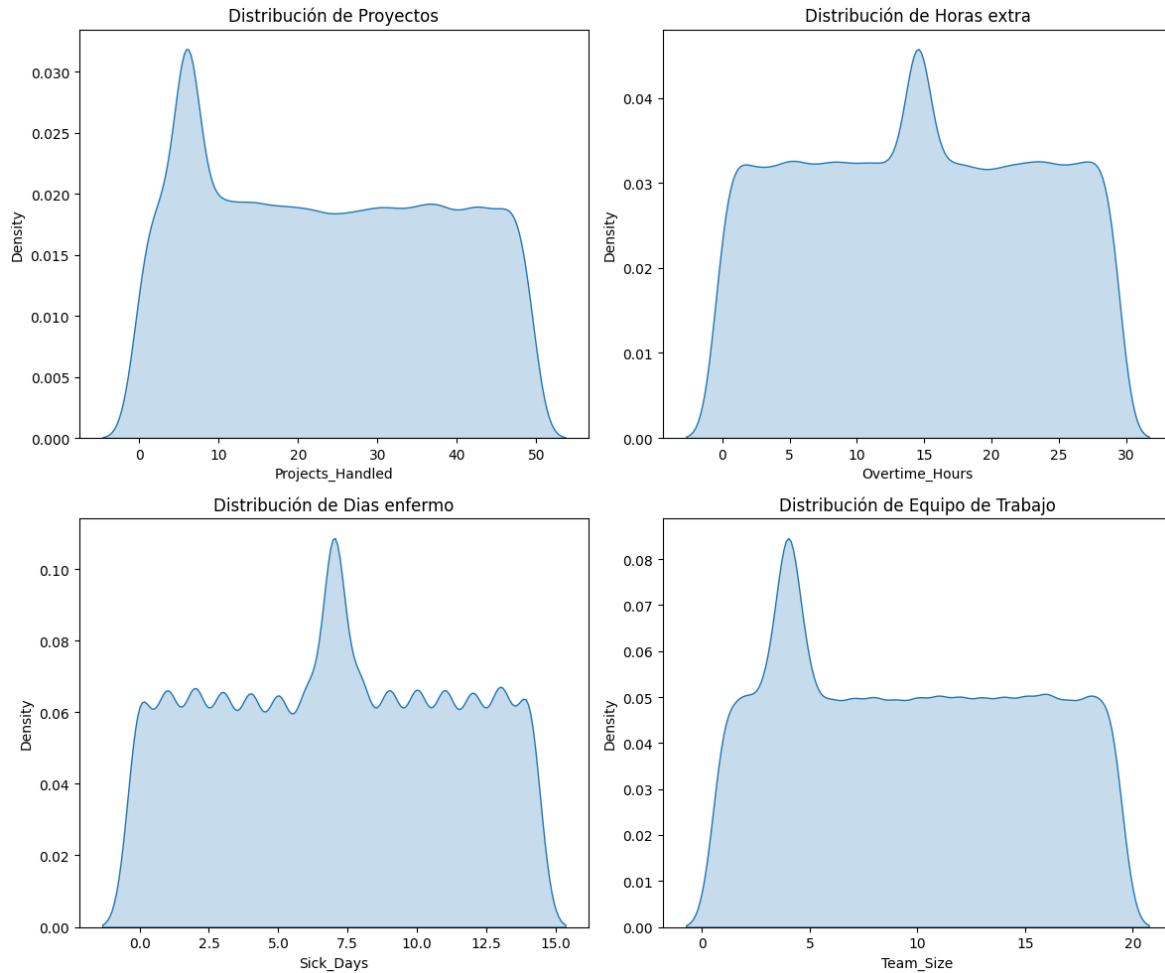
Graficos adicionales que no se incluyeron



## Proyecto Final



# Proyecto Final



Basé de datos limpia que se utilizó

```
[3] df=pd.read_csv('/content/drive/Mydrive/Bases de datos /Base_limpia_Pract2.csv')
df
```

	Employee_ID	Department	Gender	Age	Job_Title	Hire_Date	Years_At_Company	Education_Level	Performance_Score	Monthly_Salary	Work_Hours_Per_Week	Projects_Handled	Overtime_Hours	Sick_Days
0	1.0	IT	Male	55.0	Specialist	2022-01-19	2.0	High School	5.0	6750.0	33.0	32.0	22.0	2.0
1	2.0	Finance	Male	29.0	Developer	2024-04-18	0.0	High School	5.0	7500.0	34.0	34.0	13.0	14.0
2	4.0	Customer Support	Female	48.0	Analyst	2016-10-22	7.0	Bachelor	2.0	4800.0	52.0	6.0	28.0	12.0
3	5.0	Engineering	Female	36.0	Analyst	2021-07-23	3.0	Bachelor	2.0	4800.0	38.0	11.0	29.0	13.0
4	7.0	IT	Male	37.0	Technician	2023-06-28	1.0	Bachelor	5.0	5250.0	55.0	20.0	29.0	2.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
75827	86208.0	Operations	Male	57.0	Analyst	2016-07-10	8.0	PhD	1.0	4400.0	44.0	42.0	5.0	7.0
75828	91011.0	Legal	Female	38.0	Specialist	2019-06-15	6.0	High School	3.0	5850.0	51.0	12.0	18.0	5.0
75829	98318.0	Customer Support	Female	32.0	Developer	2015-04-06	9.0	High School	5.0	7500.0	30.0	6.0	1.0	0.0
75830	48253.0	Finance	Female	42.0	Engineer	2016-09-27	7.0	High School	5.0	9000.0	34.0	41.0	19.0	7.0
75831	85783.0	Sales	Female	54.0	Specialist	2022-01-02	2.0	Bachelor	2.0	5400.0	57.0	6.0	28.0	9.0

75832 rows × 20 columns

Pasos siguientes: Generar código con df | Ver gráficos recomendados | New interactive sheet

```
[4] df.info()
```

## Proyecto Final

Remote_Work_Frequency	Team_Size	Training_Hours	Promotions	Employee_Satisfaction_Score	Resigned	Actions
0.0	14.0	66.0	0.999653	2.63000	False	  
100.0	12.0	61.0	2.000000	1.72000	False	  
100.0	10.0	0.0	1.000000	1.86000	False	  
100.0	15.0	9.0	1.000000	1.25000	False	  
0.0	16.0	27.0	0.000000	4.46000	False	  
...	...	...	...	...	...	  
50.0	18.0	88.0	0.000000	3.00211	False	  
0.0	8.0	13.0	2.000000	3.81000	False	  
100.0	2.0	86.0	2.000000	2.74000	False	  
50.0	18.0	18.0	1.000000	4.40000	False	  
75.0	19.0	84.0	1.000000	3.00211	False	  