Data Load and Cleanin

```
# STEP 1: Data Load and Cleaning

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


# Load dataset
df = pd.read_csv("/content/WA_Fn-UseC_-HR-Employee-Attrition.csv")

df.head()
```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber	.
0	41	Yes	Travel_Rarely	1102	Sales	1	2	Life Sciences	1	1	
1	49	No	Travel_Frequently	279	Research & Development	8	1	Life Sciences	1	2	
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2	Other	1	4	
3	33	No	Travel_Frequently	1392	Research & Development	3	4	Life Sciences	1	5	
4	27	No	Travel_Rarely	591	Research & Development	2	1	Medical	1	7	

5 rows × 35 columns

```
df.describe()
```

	Age	DailyRate	DistanceFromHome	Education	EmployeeCount	EmployeeNumber	EnvironmentSatisfaction	HourlyRate	JobIn
count	1470.000000	1470.000000	1470.000000	1470.000000	1470.0	1470.000000	1470.000000	1470.000000	14
mean	36.923810	802.485714	9.192517	2.912925	1.0	1024.865306	2.721769	65.891156	
std	9.135373	403.509100	8.106864	1.024165	0.0	602.024335	1.093082	20.329428	
min	18.000000	102.000000	1.000000	1.000000	1.0	1.000000	1.000000	30.000000	
25%	30.000000	465.000000	2.000000	2.000000	1.0	491.250000	2.000000	48.000000	
50%	36.000000	802.000000	7.000000	3.000000	1.0	1020.500000	3.000000	66.000000	
75%	43.000000	1157.000000	14.000000	4.000000	1.0	1555.750000	4.000000	83.750000	
max	60.000000	1499.000000	29.000000	5.000000	1.0	2068.000000	4.000000	100.000000	

8 rows × 26 columns

```
# Drop unnecessary columns
columns_to_drop = ['EmployeeCount', 'Over18', 'StandardHours', 'EmployeeNumber']
df.drop(columns=columns_to_drop, axis=1, inplace=True)


# Convert 'Attrition' to binary: Yes → 1, No → 0
df['Attrition'] = df['Attrition'].map({'Yes': 1, 'No': 0})


# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values[missing_values > 0])
```

```
Missing Values:
 Series([], dtype: int64)
```

```
# Shape and Column Info
print("\nDataset Shape after Cleaning:", df.shape)
print("\nRemaining Columns:\n", df.columns)
```

What can I help you build?

```
Dataset Shape after Cleaning: (1470, 31)

Remaining Columns:
 Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
       'DistanceFromHome', 'Education', 'EducationField',
       'EnvironmentSatisfaction', 'Gender', 'HourlyRate', 'JobInvolvement',
       'JobLevel', 'JobRole', 'JobSatisfaction', 'MaritalStatus',
       'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked', 'OverTime',
       'PercentSalaryHike', 'PerformanceRating', 'RelationshipSatisfaction',
       'StockOptionLevel', 'TotalWorkingYears', 'TrainingTimesLastYear',
       'WorkLifeBalance', 'YearsAtCompany', 'YearsInCurrentRole',
       'YearsSinceLastPromotion', 'YearsWithCurrManager'],
      dtype='object')
```

```
# Show first 5 rows of cleaned data
print("\nSample Data:\n", df.head())
```

```
Sample Data:
    Age  Attrition     BusinessTravel  DailyRate              Department  \
0   41          1      Travel_Rarely       1102                   Sales
1   49          0  Travel_Frequently        279  Research & Development
2   37          1      Travel_Rarely       1373  Research & Development
3   33          0  Travel_Frequently       1392  Research & Development
4   27          0      Travel_Rarely        591  Research & Development

   DistanceFromHome  Education EducationField  EnvironmentSatisfaction  \
0                 1          2  Life Sciences                        2
1                 8          1  Life Sciences                        3
2                 2          2          Other                        4
3                 3          4  Life Sciences                        4
4                 2          1        Medical                        1

   Gender  ...  PerformanceRating  RelationshipSatisfaction  StockOptionLevel  \
0  Female  ...                  3                         1                 0
1    Male  ...                  4                         4                 1
2    Male  ...                  3                         2                 0
3  Female  ...                  3                         3                 0
4    Male  ...                  3                         4                 1

  TotalWorkingYears  TrainingTimesLastYear WorkLifeBalance  YearsAtCompany  \
0                 8                      0               1               6
1                10                      3               3              10
2                 7                      3               3               0
3                 8                      3               3               8
4                 6                      3               3               2

   YearsInCurrentRole  YearsSinceLastPromotion YearsWithCurrManager
0                   4                        0                   5
1                   7                        1                   7
2                   0                        0                   0
3                   7                        3                   0
4                   2                        2                   2

[5 rows x 31 columns]
```

```
df_cleaned = df.copy()
df_cleaned.to_csv("Clean_HR_Attrition.csv", index=False)
print("✅ Clean CSV saved!")
```

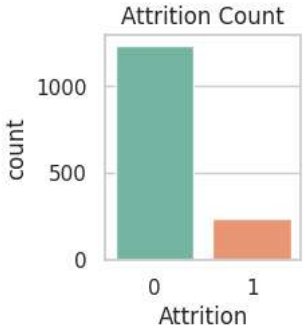
```
✅ Clean CSV saved!
```

```
# Set style
sns.set(style="whitegrid")
plt.figure(figsize=(12, 6))
```

```
<Figure size 1200x600 with 0 Axes>
<Figure size 1200x600 with 0 Axes>
```

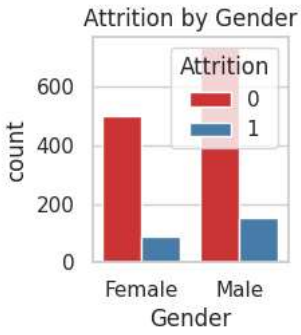
```
# 1. Count Plot - Attrition
plt.subplot(2, 3, 1)
sns.countplot(data=df, x='Attrition', palette='Set2')
plt.title("Attrition Count")
```

```
/tmp/ipython-input-14-4181848654.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.countplot(data=df, x='Attrition', palette='Set2')
Text(0.5, 1.0, 'Attrition Count')
```



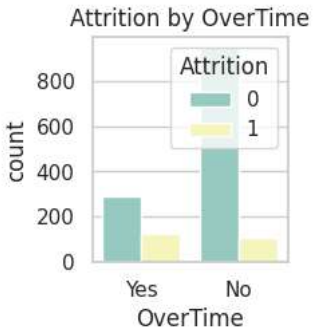

```
# 2. Attrition by Gender
plt.subplot(2, 3, 2)
sns.countplot(data=df, x='Gender', hue='Attrition', palette='Set1')
plt.title("Attrition by Gender")
```

```
Text(0.5, 1.0, 'Attrition by Gender')
```



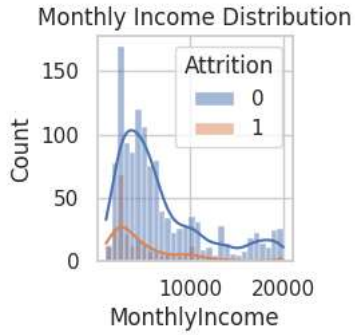

```
# 3. Attrition by OverTime
plt.subplot(2, 3, 3)
sns.countplot(data=df, x='OverTime', hue='Attrition', palette='Set3')
plt.title("Attrition by OverTime")
```

```
Text(0.5, 1.0, 'Attrition by OverTime')
```



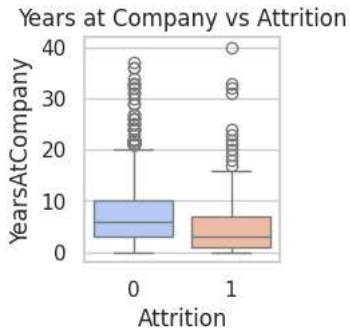

```
# 4. Monthly Income Distribution
plt.subplot(2, 3, 4)
sns.histplot(data=df, x='MonthlyIncome', hue='Attrition', kde=True, bins=30)
plt.title("Monthly Income Distribution")
```

```
Text(0.5, 1.0, 'Monthly Income Distribution')
```




```
# 5. Years at Company vs Attrition
plt.subplot(2, 3, 5)
sns.boxplot(data=df, x='Attrition', y='YearsAtCompany', palette='coolwarm')
plt.title("Years at Company vs Attrition")
```

```
/tmp/ipython-input-18-3834869003.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.boxplot(data=df, x='Attrition', y='YearsAtCompany', palette='coolwarm')
Text(0.5, 1.0, 'Years at Company vs Attrition')
```




```
# 6. Department-wise Attrition
plt.subplot(2, 3, 6)
sns.countplot(data=df, x='Department', hue='Attrition', palette='Accent')
plt.title("Attrition by Department")
plt.xticks(rotation=15)
plt.tight_layout()
plt.show()
```

** Label Encoding**

```
from sklearn.preprocessing import LabelEncoder

# Create a copy of the dataset
df_encoded = df.copy()
```

```
# Label encode all categorical columns
label_encoder = LabelEncoder()

# Identify categorical columns
categorical_cols = df_encoded.select_dtypes(include=['object']).columns

# Apply Label Encoding
for col in categorical_cols:
    df_encoded[col] = label_encoder.fit_transform(df_encoded[col])

# Show the encoded columns and head
print("Encoded Columns:\n", categorical_cols.tolist())
print("\nEncoded Dataset Sample:\n", df_encoded.head())
```

```
Encoded Columns:
 ['BusinessTravel', 'Department', 'EducationField', 'Gender', 'JobRole', 'MaritalStatus', 'OverTime']

Encoded Dataset Sample:
    Age  Attrition  BusinessTravel  DailyRate  Department  DistanceFromHome  \
0   41          1               2       1102           2                 1
1   49          0               1        279           1                 8
2   37          1               2       1373           1                 2
3   33          0               1       1392           1                 3
4   27          0               2        591           1                 2

   Education  EducationField  EnvironmentSatisfaction  Gender  ...  \
0          2               1                        2       0  ...
1          1               1                        3       1  ...
2          2               4                        4       1  ...
3          4               1                        4       0  ...
4          1               3                        1       1  ...

   PerformanceRating  RelationshipSatisfaction  StockOptionLevel  \
0                  3                         1                 0
1                  4                         4                 1
2                  3                         2                 0
3                  3                         3                 0
4                  3                         4                 1

   TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
0                  8                      0                1               6
1                 10                      3                3              10
2                  7                      3                3               0
3                  8                      3                3               8
4                  6                      3                3               2

   YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                   4                        0                     5
1                   7                        1                     7
2                   0                        0                     0
3                   7                        3                     0
4                   2                        2                     2

[5 rows x 31 columns]
```

Train-Test Split

```
from sklearn.model_selection import train_test_split

# Features (X) and Target (y)
X = df_encoded.drop('Attrition', axis=1)
y = df_encoded['Attrition']

# Split into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y)

# Print data shapes
print("✅ Train shape:", X_train.shape)
print("✅ Test shape:", X_test.shape)
```

```
✅ Train shape: (1176, 30)
✅ Test shape: (294, 30)
```

🔍 Interpretation Tips: Accuracy: Overall correctness of the model.

Confusion Matrix:

True Positives (TP): Attrition predicted correctly

False Positives (FP): Attrition predicted but didn't occur

False Negatives (FN): Attrition missed by model

Precision/Recall:

High Precision = low false positives

High Recall = low false negatives

F1 Score balances both

```
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
import seaborn as sns
import matplotlib.pyplot as plt

# Accuracy Score
accuracy = accuracy_score(y_test, y_pred)
print(f"✅ Model Accuracy: {accuracy:.2f}")

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
print("\nConfusion Matrix:\n", cm)

# Plot Confusion Matrix
plt.figure(figsize=(5,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix - Logistic Regression')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()

# Classification Report
print("\nClassification Report:\n")
print(classification_report(y_test, y_pred))
```

```
Model Accuracy: 0.86

Confusion Matrix:
 [[245   2]
 [ 39   8]]
```