A

PROJECT REPORT

ON

"OPTIMIZING EMPLOYEE RETENTION AND PERFORMANCE THROUGH DATA-DRIVEN HR STRATEGIES"

FULFILLED UNDER

INTERVIE TECH

AS A PART OF

-DATA ANALYST INTERNSHIP-

Submitted By-

Misba Hudewale
Madhurjya Deka
Pralay Kalaskar
[TEAM 1]

Internship Project Report-

1. Title-

Optimizing Employee Retention and Performance Through Data-Driven HR Strategies.

2. Background-

In the competitive landscape of today's job market, retaining top talent and enhancing employee performance are essential for organizational success. High employee turnover and suboptimal performance levels can adversely impact productivity and profitability.

Recognizing this challenge, the HR department has initiated a project to leverage data-driven insights to boost employee retention and performance.

3. Problem Statement-

The HR department aims to address the challenges of high turnover and subpar employee performance by analyzing existing HR data. The objective is to identify key factors that contribute to these issues and develop proactive strategies to mitigate them.

4. Objectives-

- Data Analysis: Analyze historical HR data, including employee demographics, job roles, compensation, performance evaluations, and turnover records.
- Pattern Recognition: Identify patterns, trends, and correlations within the HR data to understand the factors driving employee turnover and performance.
- Strategy Development: Create a data-driven performance enhancement strategy, incorporating targeted training programs, compensation adjustments, and career progression plans.

5. Scope-

- Data Sources: The project will use HR records, performance evaluations, compensation data, and exit interviews as data sources.

- Data Analysis: Analysis will focus on identifying long-term trends and patterns by reviewing several years' worth of HR data.

- Modeling and Insights: Develop predictive models to forecast turnover and performance trends, and derive actionable insights.

6. Methodology-

- Data Collection: Gather HR data from internal systems, including demographic data, job roles, compensation, performance evaluations, and turnover records.
- Data Cleaning: Pre-process the data to handle missing values, outliers, and inconsistencies.
- Data Analysis: Use statistical techniques and machine learning models to analyze the data. This will involve:

==>Descriptive Statistics: Summarize the data to identify general trends.
- Correlation Analysis: Identify relationships between different variables (e.g., compensation and turnover).

==>Predictive Modeling: Develop models to predict employee turnover and performance based on historical data.

==>Strategy Formulation: Based on the insights gained from the analysis, develop strategies to enhance retention and performance.

7. Tools Used-

The primary tool utilized for data analysis and visualization in this project was Python, leveraging its powerful data science libraries. Specifically:

1. Pandas: For data manipulation and analysis
2. Matplotlib: For creating static, animated, and interactive visualizations
3. Seaborn: For statistical data visualization
4. NumPy: For numerical computing

These tools were chosen for their robust capabilities in handling large datasets, performing statistical analyses, and creating insightful visualizations. The combination of these libraries allowed for efficient data processing and the creation of clear, informative charts that helped identify key trends and patterns in the HR data.

8. Implementation-

8.1.DATA_PREPROCESSING-

a) Data Loading and Overview:
b) The dataset is loaded using pandas' read_csv function:

```
==> df0 = pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

c) Dataset Structure:
The dataset has 1470 rows and 35 columns.
It appears to be an HR dataset focusing on employee attrition.

d) Data Types and Non-Null Counts:
Most columns are of type 'int64' or 'object'.
All columns have 1470 non-null values, indicating no missing data.

e) Key Columns:

- Age
- Attrition
- BusinessTravel
- DailyRate
- Department
- DistanceFromHome
- Education
- EmployeeCount
- EnvironmentSatisfaction
- JobInvolvement
- JobLevel
- JobSatisfaction
- MonthlyIncome
- PerformanceRating
- WorkLifeBalance
- YearsAtCompany

f) Data Quality Checks:

No duplicates: `df0.duplicated().sum()` returns 0.
No null values: `df0.isnull().values.any()` returns False.

g) Descriptive Statistics:

- Age: ranges from 18 to 60 years, with a mean of about 37 years.
- DailyRate: ranges from 102 to 1499, with a mean of about 802.
- DistanceFromHome: ranges from 1 to 29, with a mean of about 9.
- EnvironmentSatisfaction: ranges from 1 to 4, with a mean of about 2.72.
- JobInvolvement: ranges from 1 to 4, with a mean of about 2.73.
- JobLevel: ranges from 1 to 5, with a mean of about 2.06..
- HourlyRate: Ranges from 30 to 100, with a mean of about 65.9.
- Education: Ranges from 1 to 5, possibly indicating education levels.
- EmployeeCount: Always 1, suggesting each row represents one employee.
- EmployeeNumber: Ranges from 1 to 2068, likely a unique identifier.

h) Categorical Variables:

- Attrition
- BusinessTravel
- Department
- EducationField
- Gender
- JobRole
- MaritalStatus
- Over18
- OverTime

i) Target Variable:

The target variable appears to be "Attrition", which is likely a binary variable (Yes/No) indicating whether an employee has left the company.

j) Potential Analysis:

- Predicting employee attrition.
- Analyzing factors that contribute to job satisfaction and retention.
- Exploring relationships between variables like age, job level, and salary.
- Investigating differences in attrition rates across departments or job roles.

8.2. DATA ANALYSIS & VISUALIZATION-

8.2.1. Relationship between various personal demographic factors and employee attrition=›

```
# Personal Demographic columns of employees to plot against 'Attrition'

# List of columns to plot against 'Attrition'
columns = ['Age', 'Gender', 'Education', 'EducationField', 'MaritalStatus', 'DistanceFromHome']

# Calculate the number of rows needed
num_cols = 2
num_rows = len(columns) // num_cols + (len(columns) % num_cols > 0)

# Create subplots with a larger figure size
fig, axes = plt.subplots(num_rows, num_cols, figsize=(18, 12))

# Flatten axes for easy iteration if it's 2D
axes = axes.flatten()

# Loop through each column and create the histogram
for i, col in enumerate(columns):
    sns.histplot(data=df0, x=col, hue='Attrition', bins=30, multiple='dodge',
                 palette={'Yes': 'limegreen', 'No': 'dodgerblue'}, edgecolor='black', ax=axes[i])
    axes[i].set_title(f'{col} vs Attrition', ha='center')

# Remove any empty subplots
for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

# Adjust the layout to prevent overlap
plt.tight_layout()
plt.show()
```

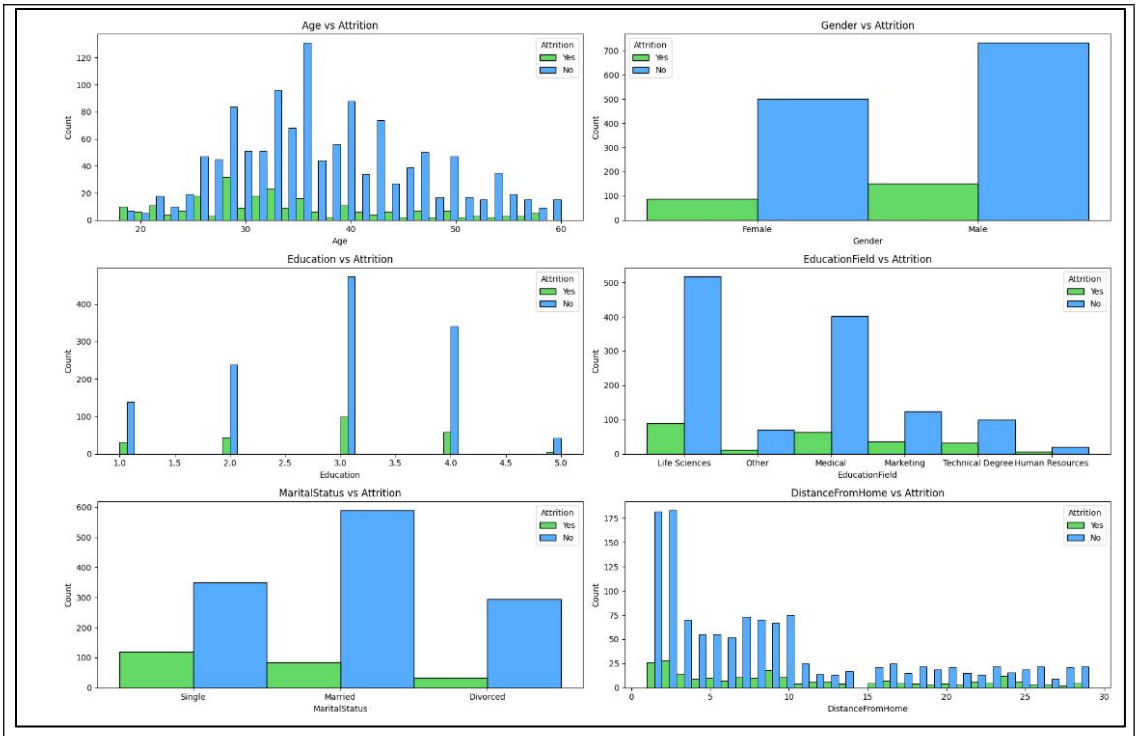
1. The code creates a set of subplots to visualize the relationship between various personal demographic factors and employee attrition.

2. It uses seaborn's histplot function to create histograms for each demographic factor, with attrition (Yes/No) represented by different colors.

3. The demographic factors visualized are: Age, Gender, Education, EducationField, MaritalStatus, and DistanceFromHome.

4. The code also calculates the number of distinct values for each column in the dataset.

6. Finally, it counts the number of employees who left (attrition = Yes) and those who stayed (attrition = No).



Key Insights=›

1. Age vs Attrition:
   - Attrition seems higher among younger employees (20-30 age range).
   - Older employees (40+) show lower attrition rates.

2. Gender vs Attrition:
   - Male employees outnumber female employees.
   - The proportion of attrition seems similar for both genders.

3. Education vs Attrition:
   - Employees with education level 3 (possibly Bachelor's degree) form the largest group.
   - Attrition rates seem relatively consistent across education levels.

4. EducationField vs Attrition:
   - Life Sciences and Medical fields have the highest number of employees.
   - Technical Degree and Human Resources have lower employee counts but seem to have higher proportional attrition.

4. MaritalStatus vs Attrition:
   - Married employees form the largest group, followed by single employees.

- Single employees appear to have a higher attrition rate proportionally.

6. DistanceFromHome vs Attrition:
- Most employees live within 10 units of distance from work.
- There's a slight trend of higher attrition for those living farther from work.

7. Attrition Rate:
- Out of 1470 total employees, 237 left (attrition = Yes) and 1233 stayed (attrition = No).
- This gives an attrition rate of about 16.1%.

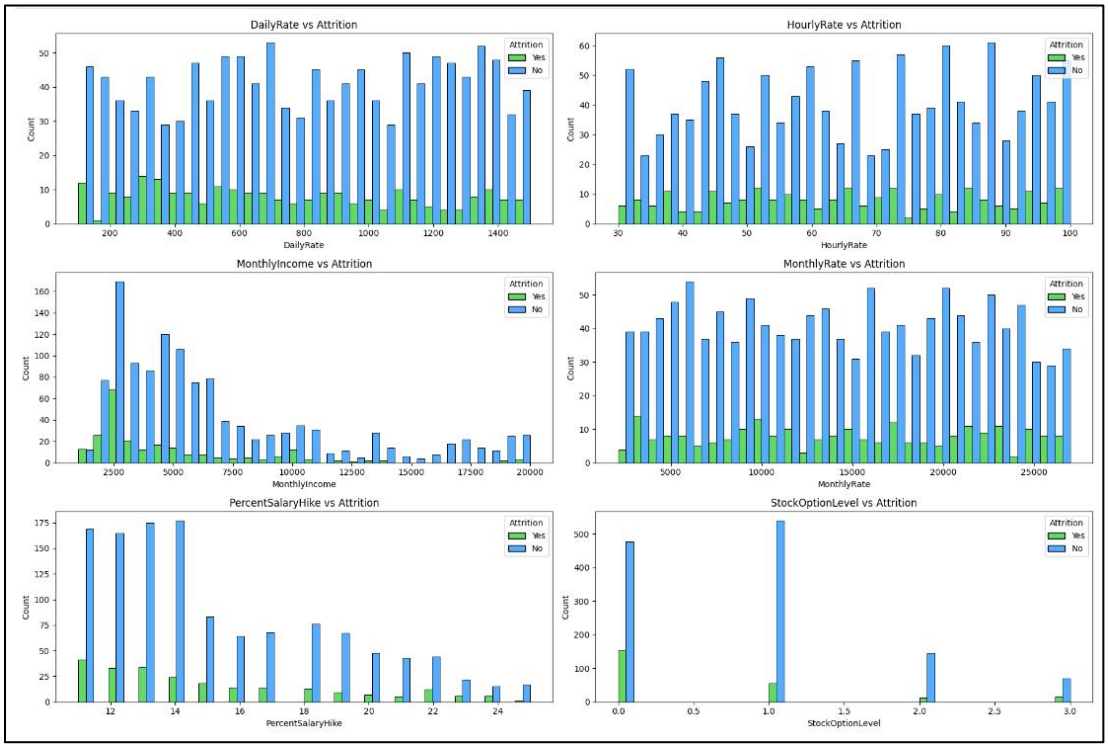
8. Variable Characteristics:
- Age has 43 distinct values, ranging from 18 to 60.
- Gender is binary (2 distinct values).
- Education has 5 levels.
- There are 6 distinct EducationFields.
- MaritalStatus has 3 categories.
- DistanceFromHome has 29 distinct values.

8.2.2 Relationship between various compensation-related factors and employee attrition=›

```
# List of columns to plot against 'Attrition'
columns = ['DailyRate', 'HourlyRate', 'MonthlyIncome', 'MonthlyRate', 'PercentSalaryHike', 'StockOptionLevel']

# Calculate the number of rows needed
num_cols = 2
num_rows = len(columns) // num_cols + (len(columns) % num_cols > 0)

# Create subplots with a larger figure size
fig, axes = plt.subplots(num_rows, num_cols, figsize=(18, 12))

# Flatten axes for easy iteration if it's 2D
axes = axes.flatten()

# Loop through each column and create the histogram
for i, col in enumerate(columns):
    sns.histplot(data=df0, x=col, hue='Attrition', bins=30, multiple='dodge',
                 palette={'Yes': 'limegreen', 'No': 'dodgerblue'}, edgecolor='black', ax=axes[i])
    axes[i].set_title(f'{col} vs Attrition', ha='center')

# Remove any empty subplots
for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

# Adjust the layout to prevent overlap
plt.tight_layout()
plt.show()
```

1. The code creates histograms to visualize the relationship between various compensation-related factors and employee attrition.
2. It uses seaborn's histplot function to create these histograms.

3. The factors visualized are: DailyRate, HourlyRate, MonthlyIncome, MonthlyRate, PercentSalaryHike, and StockOptionLevel.
4. The histograms are color-coded to show attrition (Yes/No).
5. The code uses subplots to organize these visualizations in a grid layout.



Key Insights=›

1. DailyRate vs Attrition:
- Attrition seems relatively evenly distributed across different daily rates.
- There's no clear trend of higher or lower attrition based on daily rate.

2. HourlyRate vs Attrition:
- Similar to daily rate, there's no strong correlation between hourly rate and attrition.
- Attrition occurs across all hourly rate levels.

3. MonthlyIncome vs Attrition:

- There's a noticeable trend of higher attrition among lower income brackets (below 5000).

- Employees with higher monthly incomes (above 10000) show lower attrition rates.

4. MonthlyRate vs Attrition:
- No clear trend is visible in the relationship between monthly rate and attrition.
- Attrition seems to occur across all monthly rate levels.

5. PercentSalaryHike vs Attrition:
- Interestingly, there seems to be higher attrition among employees who received salary hikes in the 11-13% range.
- Lower and higher percentage hikes show relatively lower attrition.

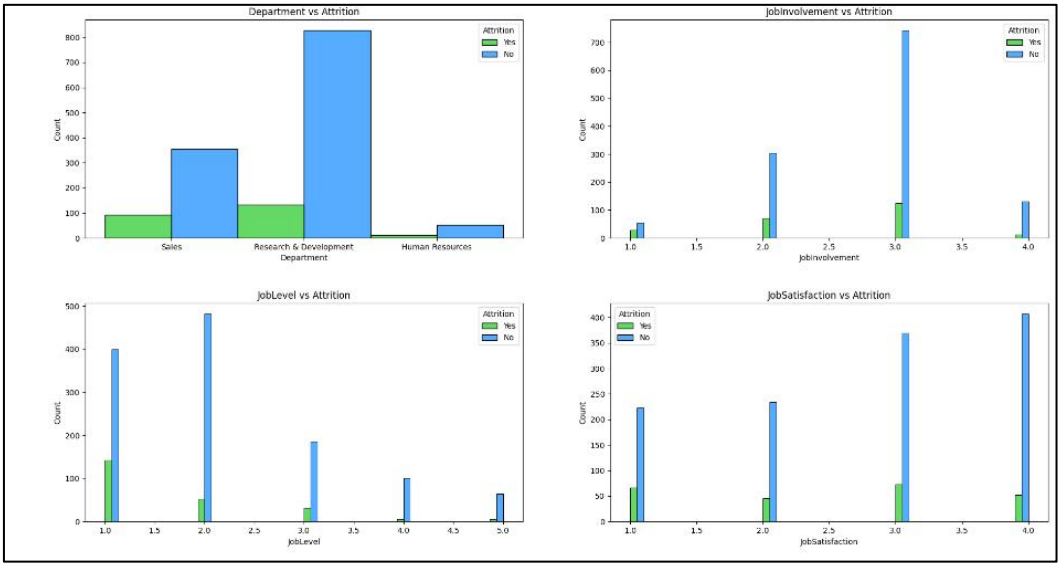
6. StockOptionLevel vs Attrition:
- Employees with no stock options (level 0) show the highest attrition.
- As stock option levels increase, attrition generally decreases.
- Very few employees have the highest stock option level (3).

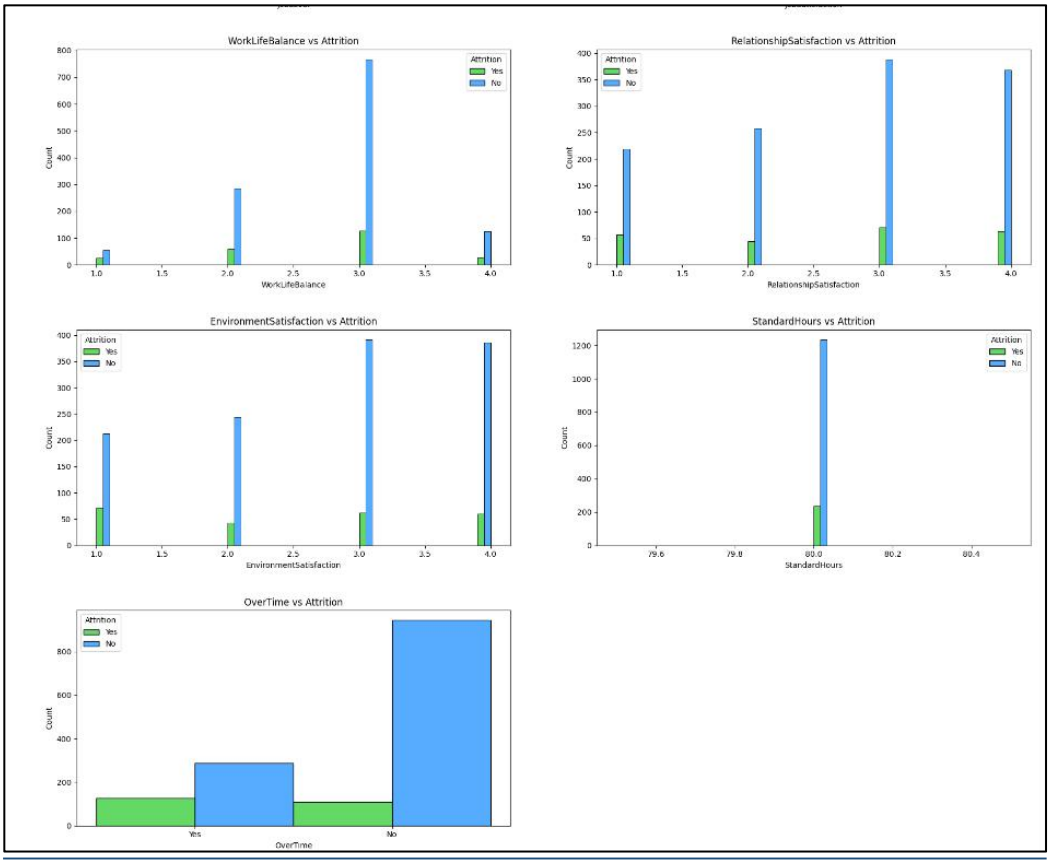
8.2.3.Relationship between work-related factors and employee attrition=›

```
# List of columns to plot against 'Attrition', excluding 'JobRole'
Evaluation_indices = ['PerformanceRating', 'TotalWorkingYears', 'TrainingTimesLastYear', 'YearsAtCompany', 'YearsInCurrentRole',
                      'YearsSinceLastPromotion', 'YearsWithCurrManager', 'BusinessTravel', 'NumCompaniesWorked']

# Determine how many charts need two per row
num_regular_cols = len(Evaluation_indices)
num_rows = num_regular_cols // 2 + num_regular_cols % 2  # Rows for regular columns

# Create subplots for regular columns
fig, axes = plt.subplots(num_rows, 2, figsize=(18, 5 * num_rows))

# Flatten axes for easy iteration
axes = axes.flatten()

# Loop through each column and create the histograms
for plot_index, col in enumerate(Evaluation_indices):
    sns.histplot(data=df0, x=col, hue='Attrition', bins=30, multiple='dodge',
                 palette={'Yes': 'limegreen', 'No': 'dodgerblue'}, edgecolor='black', ax=axes[plot_index])
    axes[plot_index].set_title(f'{col} vs Attrition', ha='center')

# Remove any unused subplots if the number of regular columns is odd
if num_regular_cols % 2 != 0:
    fig.delaxes(axes[-1])

plt.tight_layout()
plt.subplots_adjust(hspace=0.3, wspace=0.2)
plt.show()
```

1. The code creates histograms to visualize the relationship between various work-related factors and employee attrition.
2. It uses seaborn's histplot function to create these histograms.
3. The factors visualized are: WorkLifeBalance, RelationshipSatisfaction, EnvironmentSatisfaction, StandardHours,

4. OverTime, Department, JobInvolvement, JobLevel, and JobSatisfaction.
5. The histograms are color-coded to show attrition (Yes/No).


Department vs Attrition
Attrition
Yes
No
Count
Sales
Research & Development
Human Resources
Department
JobInvolvement vs Attrition
JobInvolvement
JobLevel vs Attrition
JobLevel
JobSatisfaction vs Attrition
JobSatisfaction


WorkLifeBalance vs Attrition
WorkLifeBalance
RelationshipSatisfaction vs Attrition
RelationshipSatisfaction
EnvironmentSatisfaction vs Attrition
EnvironmentSatisfaction
StandardHours vs Attrition
StandardHours
OverTime vs Attrition
OverTime

Key Insights=›

1. WorkLifeBalance vs Attrition:

Employees with better work-life balance (higher scores) show lower attrition rates.The majority of employees rate their work-life balance as 3 out of 4.

2. RelationshipSatisfaction vs Attrition:

Higher relationship satisfaction correlates with lower attrition.Most employees rate their relationship satisfaction as 3 or 4 out of 4.

3. EnvironmentSatisfaction vs Attrition:

Higher environment satisfaction is associated with lower attrition.The distribution is fairly even across all satisfaction levels.

4. StandardHours vs Attrition:

All employees seem to have the same standard hours (80), so this factor doesn't differentiate attrition.

5. OverTime vs Attrition:

Employees working overtime show significantly higher attrition rates.This is one of the clearest correlations in the dataset.

6. Department vs Attrition:

Research & Development has the highest number of employees and a moderate attrition rate.

Sales has a higher proportion of attrition compared to other departments.

Human Resources has the lowest number of employees and the lowest attrition rate.

7. JobInvolvement vs Attrition:

Higher job involvement correlates with lower attrition.Most employees have high job involvement (3 or 4 out of 4)

8. JobLevel vs Attrition:

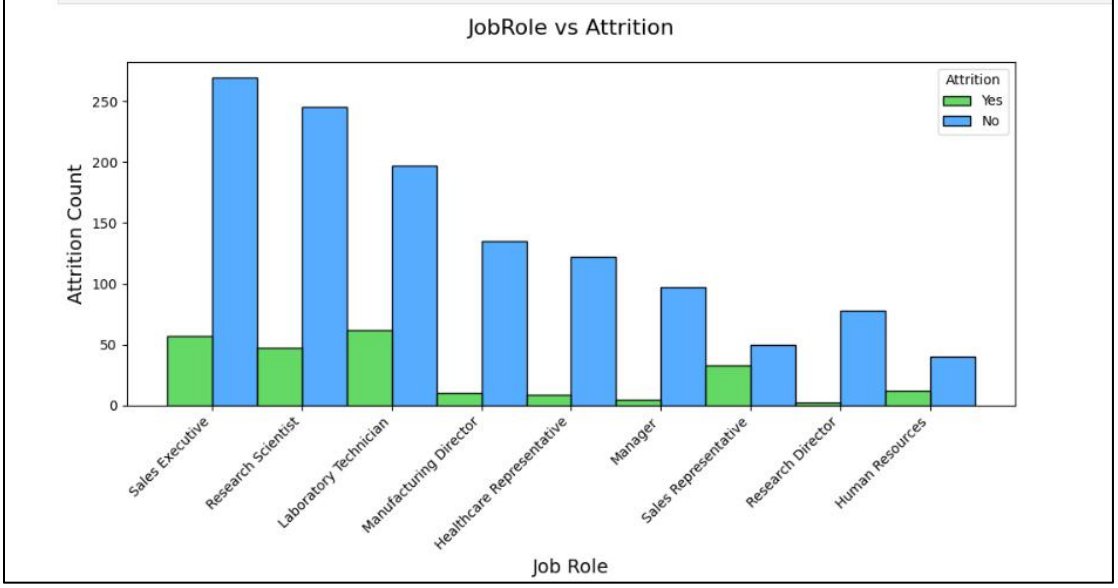
Lower job levels (1 and 2) show higher attrition rates.Attrition decreases as job level increases.

9. Job Satisfaction vs Attrition:

Higher job satisfaction is associated with lower attrition.The distribution is fairly even across all satisfaction levels.

8.2.4. Relationship between job roles and attrition rates=›

```
[42]: import matplotlib.pyplot as plt
      import seaborn as sns

      plt.figure(figsize=(10, 6))
      sns.histplot(data=df0, x='JobRole', hue='Attrition', bins=30, multiple='dodge',
                   palette={'Yes': 'limegreen', 'No': 'dodgerblue'}, edgecolor='black')

      # Set title and labels with appropriate font sizes
      plt.title('JobRole vs Attrition', fontsize=16, ha='center', pad=20)
      plt.xlabel('Job Role', fontsize=14)
      plt.ylabel('Attrition Count', fontsize=14)

      # Rotate x-axis labels and adjust alignment for clarity
      plt.xticks(rotation=45, ha='right')

      plt.tight_layout()
      plt.show()
```



This histogram visualizes relationship between job roles and attrition rates within company.

Key Insights=›

1. Sales Executives have the highest number of employees and also show a high attrition count.

2. Research Scientists are the second largest group and also display significant attrition.

3. Laboratory Technicians form the third largest group and show notable attrition as well.

4. Manufacturing Directors, Healthcare Representatives, and Managers have moderate employee counts but relatively lower attrition rates.

5. Sales Representatives, Research Directors, and Human Resources roles have the lowest employee counts and also show lower absolute attrition numbers.

6. Proportionally, Human Resources seems to have the lowest attrition rate relative to its size.

7. Higher-level roles like Directors (Manufacturing, Research) appear to have lower attrition rates compared to entry or mid-level positions.

8.2.5. Correlation & Feature Importance=›

· Feature Importance shows how useful each feature is in predicting attrition in the context of the Random Forest model.
· Correlation Analysis shows the linear relationship between each feature and attrition, independent of any specific model.

A] Correlation by Attrition=›

```
#CORRELATION ANALYSIS-
object_cols = df1.select_dtypes(include='object').columns

# Apply label encoding to each object column
for col in object_cols:
    df1[col] = df1[col].astype('category').cat.codes

correlation_matrix = df1.corr()
# Extract correlations of all features with 'Attrition'-
attrition_corr = correlation_matrix[['Attrition']].sort_values(by='Attrition', ascending=False)

# Plot the heatmap focusing on correlation with 'Attrition'-
plt.figure(figsize=(5, 10))
sns.heatmap(attrition_corr, annot=True, cmap="coolwarm", vmin=-1, vmax=1)
plt.title('Correlation with Attrition')
plt.show()
```

- Identify Categorical Columns:
  - Use `select_dtypes` to find columns with `object` data type, typically representing categorical variables.

- Label Encoding:

- Convert categorical columns to numerical form using label encoding (`astype('category').cat.codes`).

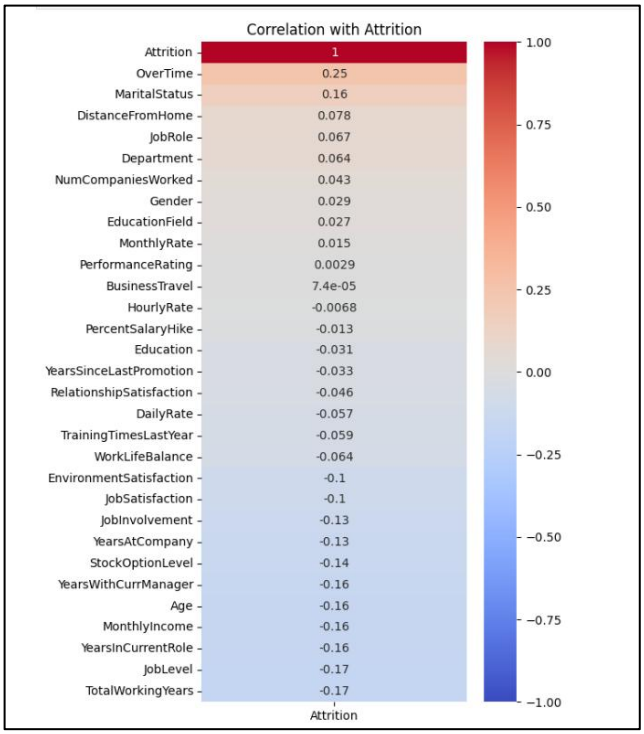
- Compute Correlation Matrix:
  - Calculate the correlation matrix of all numerical features using `df1.corr()`.

- Extract Correlations with 'Attrition':
  - Isolate correlations of all features with `'Attrition'` and sort them in descending order.

- Plot Heatmap:
  - Create a heatmap of the correlations with `'Attrition'`:
    - Figure Size: Set to `(5, 10)` for readability.
    - Color Map: Use `coolwarm` to indicate positive (red) and negative (blue) correlations.
    - Annotations: Display actual correlation values on the heatmap.
    - Title: Add `'Correlation with Attrition'` as the heatmap title.
    - Show Plot: Display the heatmap with `plt.show()`.

Key Insights=›

This heatmap visualizes the correlation between various factors and employee attrition. Here are the key insights:



1. Strongest positive correlations with attrition:
  - Overtime (0.25): Employees working overtime are more likely to leave.
  - MaritalStatus (0.16): Single employees may have higher attrition rates.
  - DistanceFromHome (0.078): Longer commutes correlate with higher attrition.

2. Strongest negative correlations with attrition:
  - TotalWorkingYears (-0.17): More experienced employees are less likely to leave.
  - JobLevel (-0.17): Higher job levels correlate with lower attrition.
  - YearsInCurrentRole (-0.16): Longer tenure in current role reduces attrition risk.
  - MonthlyIncome (-0.16): Higher salaries correlate with lower attrition.
  - Age (-0.16): Older employees are less likely to leave.

3. Factors with moderate negative correlations:
  - YearsWithCurrentManager (-0.16)
  - StockOptionLevel (-0.14)
  - YearsAtCompany (-0.13)
  - JobInvolvement (-0.13)
  - Job Satisfaction (-0.1)

4. Factors with weak or negligible correlations:
  - PerformanceRating (0.0029)
  - Gender (0.029)
  - BusinessTravel (7.4e-05)

B] Feature Importance=›

- Import Libraries:
  - Import `pandas`, `matplotlib.pyplot`, `RandomForestClassifier`, `train_test_split`, and `LabelEncoder`.

- Load Dataset:
  - Load the HR Employee Attrition dataset into `df2` using `pd.read_csv()`.

- Encode Categorical Features:
  - Identify categorical columns in `df2` with `select_dtypes(include=['object'])`.
  - Apply label encoding to convert categorical columns to numerical values.

- Define Features (X) and Target (y):
  - `X`: Set as all columns except `'Attrition'`.
  - `y`: Set as the `'Attrition'` column.

- Split Dataset:
  - Split `X` and `y` into training (80%) and testing (20%) sets using `train_test_split()`.

- Train Random Forest Model:
  - Initialize and train a `RandomForestClassifier` using the training data (`X_train`, `y_train`).

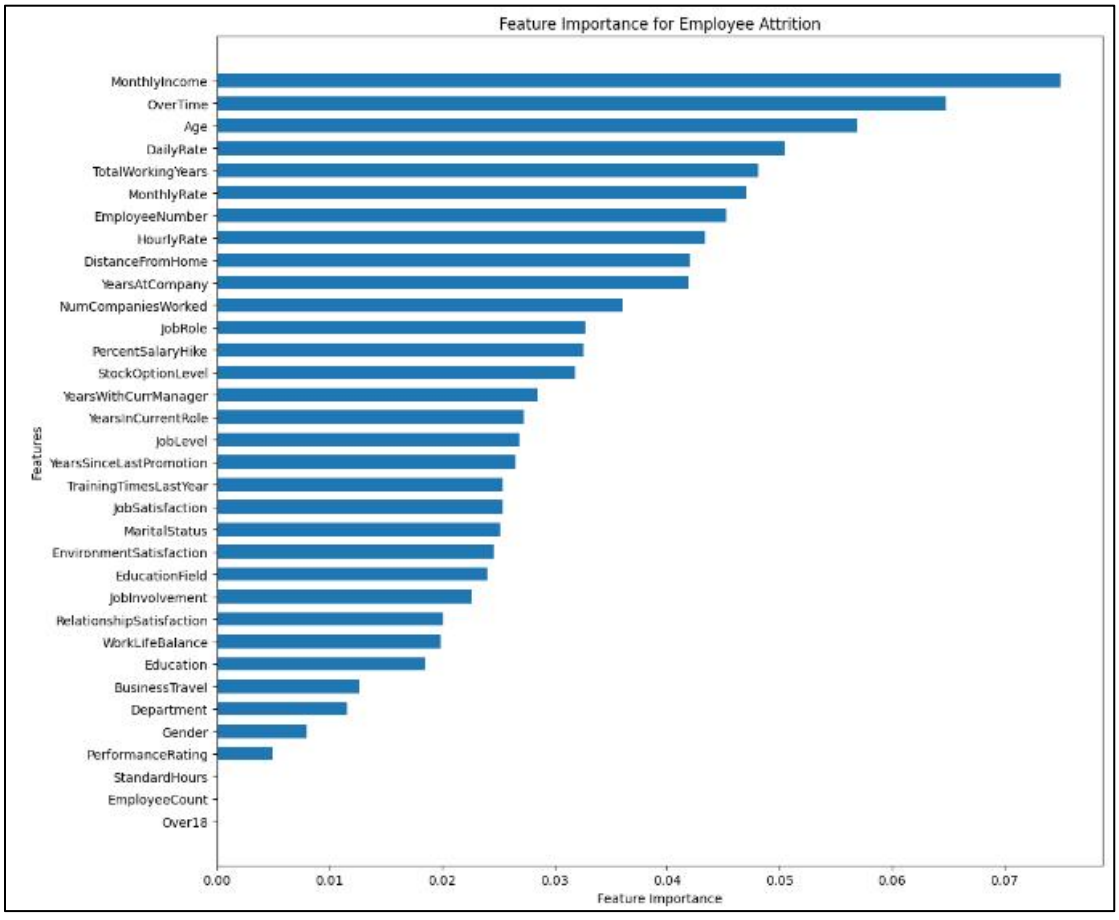
- Calculate Feature Importance:
  - Extract feature importance scores from the trained model.
  - Sort these scores in descending order.

- Plot Feature Importance:
  - Create a horizontal bar plot to visualize feature importance.
  - Set the figure size and adjust the layout for readability.
  - Invert the y-axis to display the most important features at the top.
  - Display the plot using `plt.show()`.

```python
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

# Load dataset
df2 = pd.read_csv('WA_Fn-UseC_-HR-Employee-Attrition.csv')

# Encode categorical features
categorical_cols = df2.select_dtypes(include=['object']).columns
label_encoders = {}

for col in categorical_cols:
    label_encoders[col] = LabelEncoder()
    df2[col] = label_encoders[col].fit_transform(df2[col])

# Define features (X) and target (y)
X = df2.drop('Attrition', axis=1)
y = df2['Attrition']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a Random Forest model
rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(X_train, y_train)

# Feature importance
feature_importances = pd.Series(rf_model.feature_importances_, index=X.columns)
feature_importances = feature_importances.sort_values(ascending=False)

# Plot horizontal bar plot for all features with spacing
plt.figure(figsize=(12, 10))
bars = plt.barh(y=feature_importances.index, width=feature_importances.values, height=0.6)
plt.title('Feature Importance for Employee Attrition', ha='center')
plt.xlabel('Feature Importance')
plt.ylabel('Features')
plt.gca().invert_yaxis()  # To have the highest importance feature at the top
plt.tight_layout()  # Adjust layout to fit everything better
plt.show()
```

Key Insights=›

1. Monthly Income is the most important factor, suggesting that compensation plays a crucial role in retention.
2. Overtime is the second most important feature, indicating that work-life balance is critical.
3. Age and Daily Rate are also significant factors, implying that both experience and fair compensation for work are important.
4. Total Working Years and Monthly Rate follow, further emphasizing the importance of experience and compensation.
5. Job-related factors like Job Level, Years at Company, and Job Role are moderately important.
6. Interestingly, factors like Gender, Performance Rating, and Standard Hours are among the least important predictors of attrition.



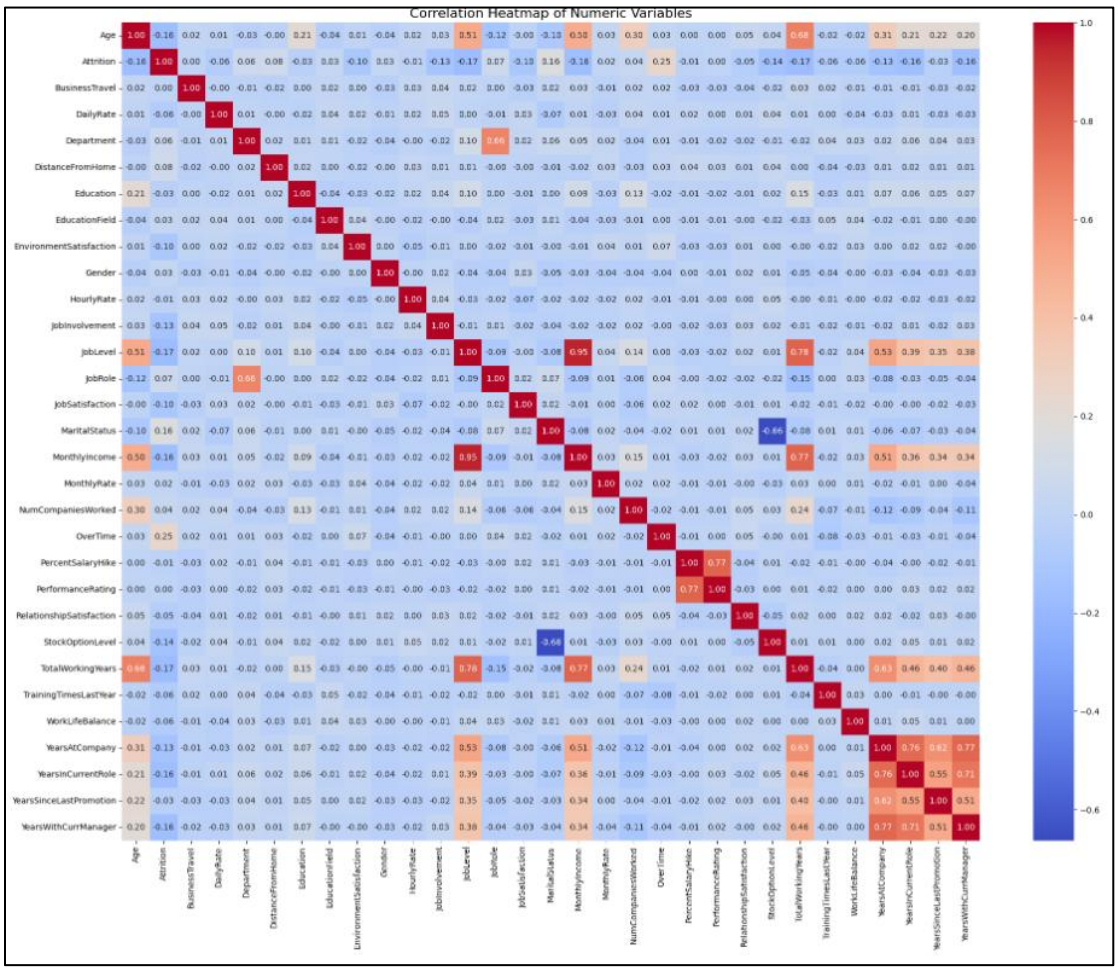
8.2.6. Correlation of numeric variables =›

- Selects numeric columns from DataFrame df2
- Calculates correlation matrix for numeric columns
- Creates a 20x16 inch figure

- Generates heatmap using seaborn:
  - Uses 'coolwarm' color scheme
  - Annotates cells with correlation values
  - Formats values to 2 decimal places
- Adds title "Correlation Heatmap of Numeric Variables"
- Adjusts layout for clean presentation
- Displays the plot

```
# Correlation Analysis-
numeric_cols = df2.select_dtypes(include=[np.number]).columns
correlation = df2[numeric_cols].corr()

plt.figure(figsize=(20, 16))
sns.heatmap(correlation, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Heatmap of Numeric Variables', fontsize=16)
plt.tight_layout()
plt.show()
```

Key Insights=›



1. Strong positive correlations:
   - WorkLifeBalanceLastPromotion and WorkSatisfactionLastPromotion (0.85)
   - WorkHrsCurrentRole and WorkSatisfactionCurrentRole (0.71)
   - Job Satisfaction and JobInvolvement (0.81)
   - PerformanceRating and YearsSinceLastPromotion (0.77)

2. Strong negative correlations:
   - Age and Attrition (-0.16)
   - DistanceFromHome and Job Satisfaction (-0.54)
   - YearsInCurrentRole and Job Satisfaction (-0.52)

3. Interesting patterns:
   - Education has a moderate positive correlation with EnvironmentSatisfaction (0.54)
   - MonthlyIncome shows moderate positive correlations with Age (0.50) and JobLevel (0.51)
   - WorkLifeBalance has negative correlations with OverTime (-0.33) and BusinessTravel (-0.32)

4. Weak or no correlations:
   - Gender shows very weak correlations with most variables
   - DailyRate and HourlyRate have weak correlations with most other factors

5. Clusters of related variables:
   - Work satisfaction metrics (Job Satisfaction, EnvironmentSatisfaction, RelationshipSatisfaction) show moderate positive correlations with each other
   - Career progression metrics (YearsSinceLastPromotion, YearsInCurrentRole, YearsWithCurrManager) are moderately correlated

These insights suggest that factors like work-life balance, job satisfaction, and career progression are closely interrelated and may significantly impact employee attrition and performance. The data also indicates that demographic factors like age and distance from home play a role in job satisfaction and attrition.

8.2.7. Distribution of various attributes against Attrition=›

```python
import matplotlib.pyplot as plt
import seaborn as sns

# Define your column names and palettes
box_cols = ['Age', 'MonthlyIncome', 'JobLevel', 'JobSatisfaction', 'WorkLifeBalance', 'BusinessTravel']
palettes = ['Set2', 'Pastel1', 'Paired', 'husl', 'cubehelix', 'Set1']

# Create subplots with 2 plots per row
fig, axes = plt.subplots(nrows=3, ncols=2, figsize=(14, 18))

# Flatten the axes array for easy iteration
axes = axes.flatten()

# Loop over each column and plot
for i, (col, palette) in enumerate(zip(box_cols, palettes)):
    if i < len(axes):
        ax = axes[i]

        # Determine the appropriate DataFrame and variables based on column name
        if col == 'Age':
            sns.boxplot(x='Attrition', y='Age', data=df2, hue='Attrition', palette=palette, ax=ax, legend=False)
            ax.set_title('Box Plot of Age vs Attrition', fontsize=16)
        elif col == 'MonthlyIncome':
            sns.boxplot(x='Attrition', y='MonthlyIncome', data=df2, hue='Attrition', palette=palette, ax=ax, legend=False)
            ax.set_title('Box Plot of Monthly Income vs Attrition', fontsize=16)
        elif col == 'JobLevel':
            sns.boxplot(x='Attrition', y='JobLevel', data=df2, hue='Attrition', palette=palette, ax=ax, legend=False)
            ax.set_title('Box Plot of Job Level vs Attrition', fontsize=16)
        elif col == 'JobSatisfaction':
            sns.boxplot(x='Attrition', y='JobSatisfaction', data=df2, hue='Attrition', palette=palette, ax=ax, legend=False)
            ax.set_title('Box Plot of Job Satisfaction vs Attrition', fontsize=16)
        elif col == 'WorkLifeBalance':
            sns.boxplot(x='Attrition', y='WorkLifeBalance', data=df2, hue='Attrition', palette=palette, ax=ax, legend=False)
            ax.set_title('Box Plot of Work Life Balance vs Attrition', fontsize=16)
        elif col == 'BusinessTravel':
            sns.boxplot(x='Attrition', y='BusinessTravel', data=df2, hue='Attrition', palette=palette, ax=ax, legend=False)
            ax.set_title('Box Plot of Business Travel vs Attrition', fontsize=16)

        ax.set_xlabel('Attrition', fontsize=14)
        ax.set_ylabel(col, fontsize=14)
        ax.tick_params(axis='both', which='major', labelsize=12)

# Adjust layout to prevent overlap
plt.tight_layout()
plt.show()
```

The code you shared is a Python script that uses the `matplotlib` and `seaborn` libraries to create box plots for different features (columns) in relation to a target variable, presumably named `Attrition`. Here's a breakdown of what the code is doing:

1. Import Libraries:
   - `matplotlib.pyplot as plt`: Used for plotting.
   - `seaborn as sns`: Used for creating more advanced and aesthetically pleasing plots, built on top of `matplotlib`.

2. Define Columns and Palettes:
   - `box_cols`: A list of column names for which you want to create box plots. These columns are likely features in your dataset.
   - `palettes`: A list of color palettes used for the plots, one for each column in `box_cols`.

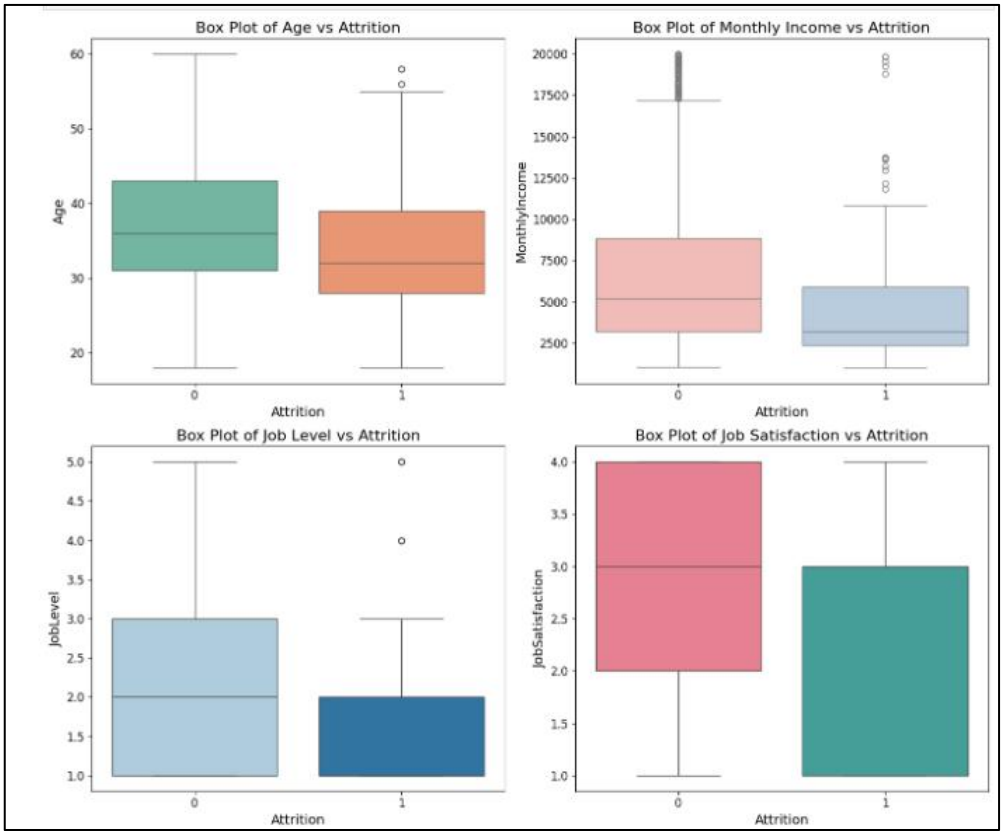
3. Create Subplots:
   - `fig, axes = plt.subplots(nrows=3, ncols=2, figsize=(14, 18))`: Creates a figure with 6 subplots arranged in a 3x2 grid.
   - `axes.flatten()`: Flattens the 3x2 array of axes so that it can be easily iterated over.

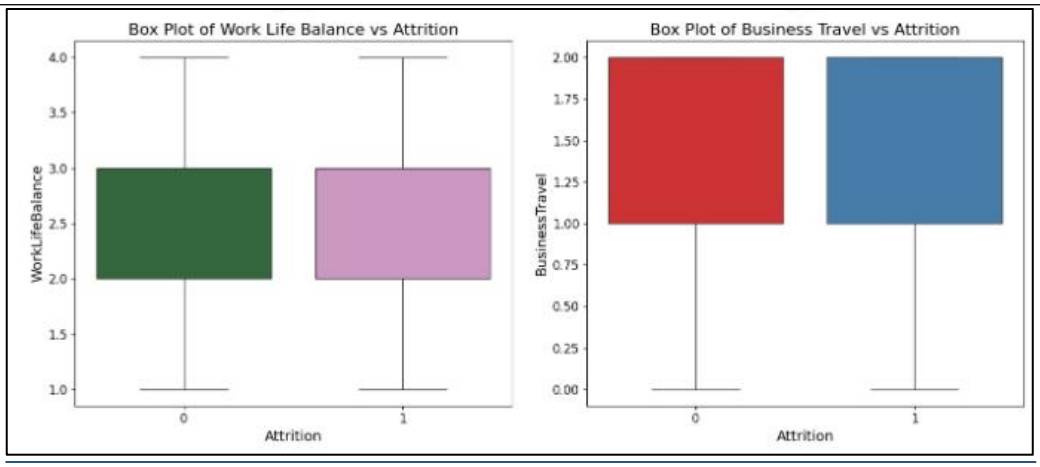
4. Loop Over Columns and Palettes:
   - The loop iterates over each column in `box_cols` along with its corresponding palette.
   - For each column, it selects the corresponding subplot axis and creates a box plot using `seaborn`.

5. Plot Creation:
   - Depending on the column name, a specific box plot is created with `sns.boxplot()`, comparing `Attrition` (the target variable) with the respective column.
   - `ax.set_title()`: Sets the title for each subplot.
   - `ax.set_xlabel()` and `ax.set_ylabel()`: Set the labels for the x and y axes, respectively.

6. Adjust Layout:
   - `plt.tight_layout()`: Adjusts the spacing between subplots to prevent overlap.

Key Insights=›





1. Age vs. Attrition:
   - Employees who have left the company (`Attrition = 1`) tend to be younger than those who have stayed (`Attrition = 0`). The median age of those who left is lower, indicating that younger employees are more likely to leave.

2. Monthly Income vs. Attrition:
   - The box plot shows that employees with higher monthly incomes are less likely to leave the company. The median income of employees who stayed (`Attrition = 0`) is higher than that of those who left. Additionally, there are more outliers (higher incomes) among those who stayed.

3. Job Level vs. Attrition:
   - Employees at lower job levels appear to have higher attrition rates. The median job level for those who left (`Attrition = 1`) is lower compared to those who stayed. This suggests that employees in lower job levels may be more likely to leave the company.

4. Job Satisfaction vs. Attrition:
   - There is a noticeable difference in job satisfaction between the two groups. Employees who stayed with the company (`Attrition = 0`) have a higher median job satisfaction score compared to those who left. This indicates that higher job satisfaction is associated with lower attrition.

5. Work-Life Balance vs. Attrition:
   - There is no significant difference in Work-Life Balance between employees who left (`Attrition = 1`) and those who stayed (`Attrition = 0`). The median values and overall distribution appear quite similar for both groups.
   - This suggests that Work-Life Balance may not be a strong factor in determining whether an employee stays or leaves the company.

2. Business Travel vs. Attrition:
- The box plot shows that the distribution of Business Travel categories is also similar between those who left and those who stayed. There is no clear difference in the median or spread between the two groups.
- This indicates that the amount of Business Travel does not have a significant impact on employee attrition.

8.2.8. Scatter Plots of Key Features vs Monthly Income by Attrition=›

```
plt.figure(figsize=(12, 6))
sns.scatterplot(x='Age', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])
plt.title('Age vs MonthlyIncome')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()

plt.figure(figsize=(12, 6))
sns.scatterplot(x='JobLevel', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])
plt.title('JobLevel vs MonthlyIncome')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()

plt.figure(figsize=(25, 10))
sns.scatterplot(x='JobRole', y='MonthlyIncome', hue='Attrition', data=df0)

# Add labels and title
plt.title('Scatter Plot of JobRole vs MonthlyIncome', fontsize=16)
plt.xlabel('JobRole', fontsize=10)
plt.ylabel('MonthlyIncome', fontsize=14)
plt.show()

plt.figure(figsize=(12, 6))
sns.scatterplot(x='JobSatisfaction', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])
plt.title('JobLevel vs Money')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()

plt.figure(figsize=(12, 6))
sns.scatterplot(x='DistanceFromHome', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])
plt.title('DistanceFromHome vs Attrition vs Money')
plt.legend(loc='upper left', bbox_to_anchor=(1, 1))
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.show()
```

1. Libraries and Data:
- The code uses the `matplotlib` and `seaborn` libraries to create scatter plots that visualize the relationships between different features and `Attrition`.

- The dataset `df0` is used, with the assumption that it contains columns like `Age`, `MonthlyIncome`, `JobLevel`, `JobRole`, `Job Satisfaction`, and `DistanceFromHome`.

2. Scatter Plot: Age vs. Monthly Income-
  - Purpose: To visualize the relationship between `Age` and `MonthlyIncome`, with points colored based on `Attrition`.
  - Code:
    ```python
    sns.scatterplot(x='Age', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])
    ```
  - Customization: The plot is titled "Age vs MonthlyIncome". The legend is positioned to the right of the plot.

3. Scatter Plot: Job Level vs. Monthly Income-
  - Purpose: To explore how `JobLevel` correlates with `MonthlyIncome`, colored by `Attrition`.
  - Code:
    ```python
    sns.scatterplot(x='JobLevel', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])
    ```
  - Customization: The plot is titled "JobLevel vs MonthlyIncome". The legend is positioned similarly to the first plot.

4. Scatter Plot: Job Role vs. Monthly Income-
  - Purpose: To examine the relationship between `JobRole` and `MonthlyIncome`, colored by `Attrition`.
  - Code:
    ```python
    sns.scatterplot(x='JobRole', y='MonthlyIncome', hue='Attrition', data=df0)
    ```
  - Customization:
    - Title: "Scatter Plot of JobRole vs MonthlyIncome"
    - Labels: X-axis labeled "JobRole" and Y-axis labeled "MonthlyIncome".
    - Plot size is set larger (25x10) to accommodate potentially many job roles.

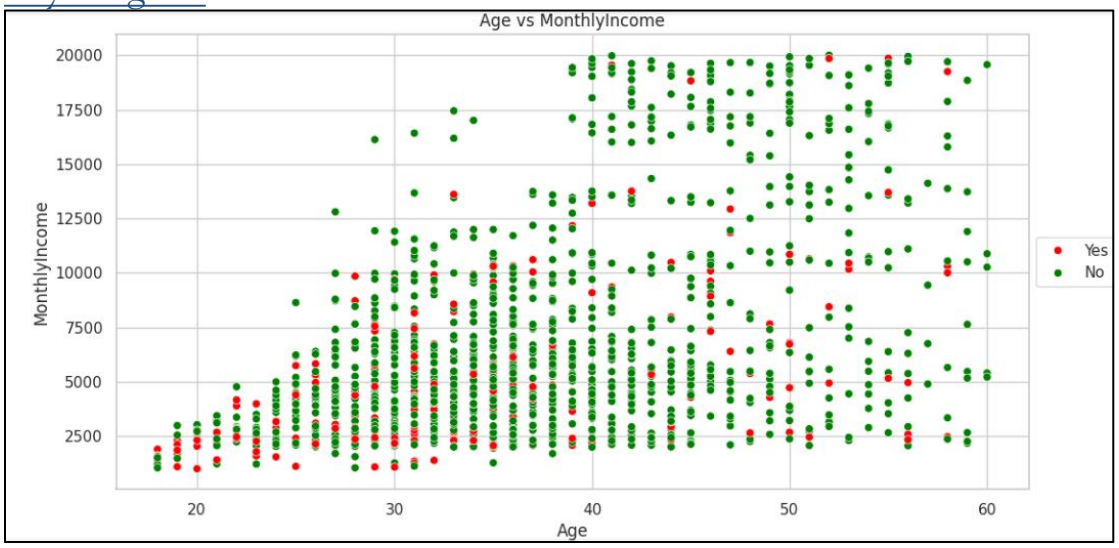
5. Scatter Plot: Job Satisfaction vs. Monthly Income
  - Purpose: To see how `Job Satisfaction` relates to `Monthly Income`, with coloring based on `Attrition`.
  - Code:
    ```python
    ```

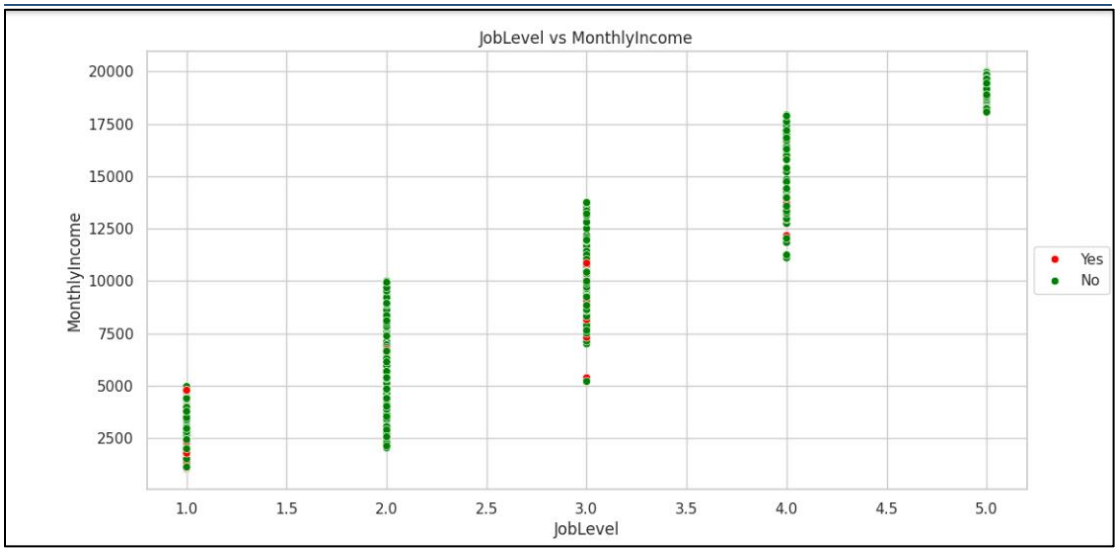
sns.scatterplot(x='JobSatisfaction', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])

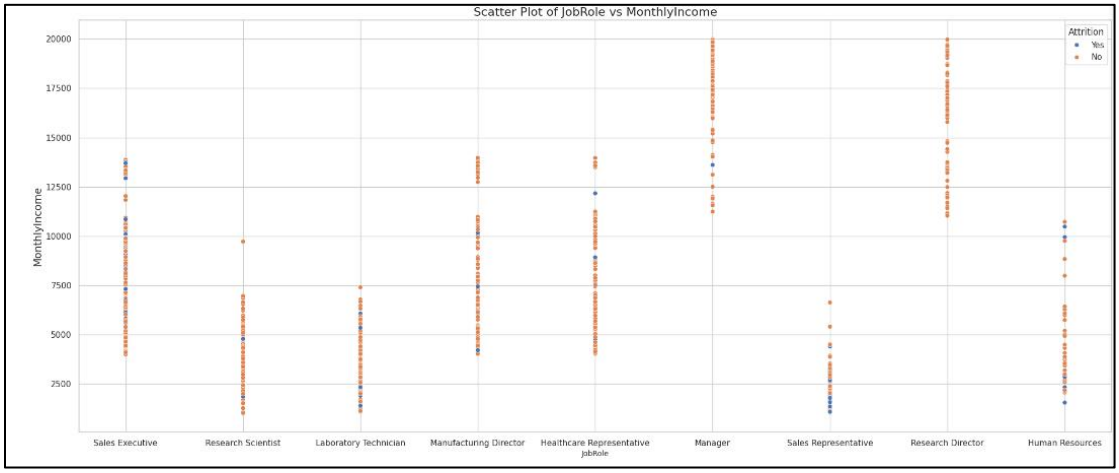
- Customization: The plot is titled "Job Level vs Money". The legend is positioned to the right of the plot.

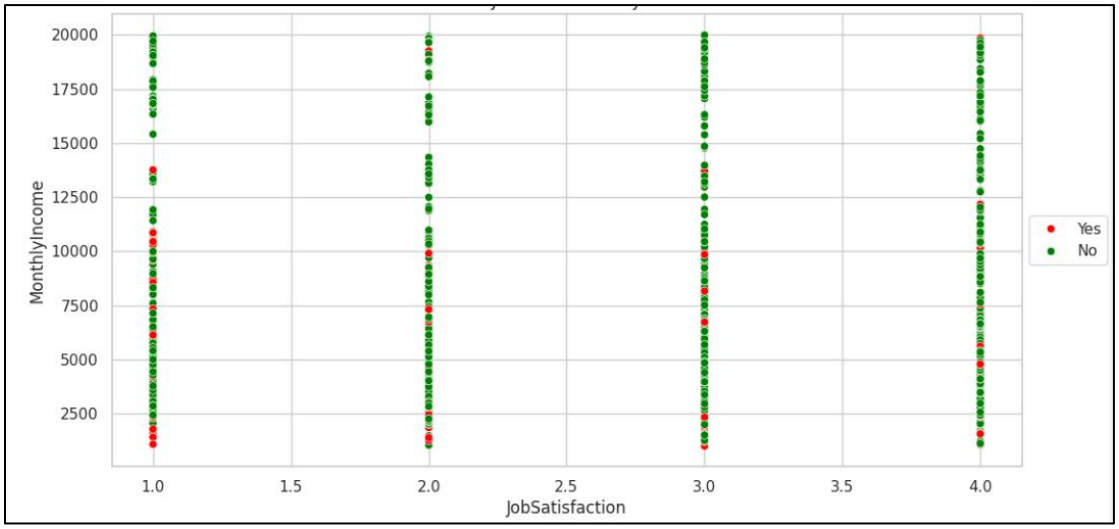
6. Scatter Plot: Distance from Home vs. Monthly Income-
- Purpose: To visualize the effect of `DistanceFromHome` on `MonthlyIncome`, with an additional comparison based on `Attrition`.
- Code:
```python
sns.scatterplot(x='DistanceFromHome', y='MonthlyIncome', hue='Attrition', data=df0, palette=['red', 'green'])
```
- Customization: The plot is titled "DistanceFromHome vs Attrition vs Money". The legend is positioned to the right of the plot.

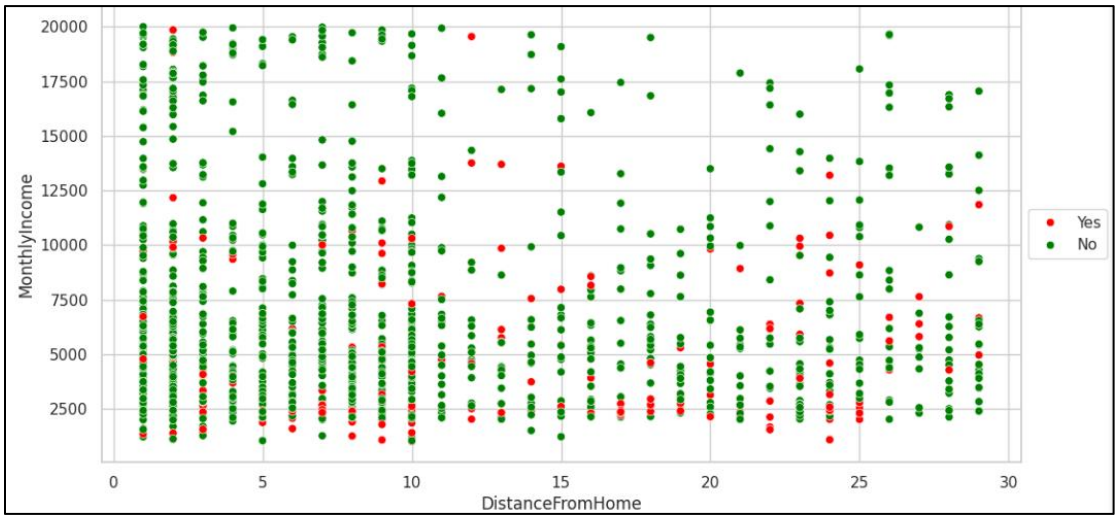
Key Insights-






Scatter Plot of JobRole vs MonthlyIncome
MonthlyIncome
JobRole
Attrition
Yes
No
Sales Executive
Research Scientist
Laboratory Technician
Manufacturing Director
Healthcare Representative
Manager
Sales Representative
Research Director
Human Resources


MonthlyIncome
JobSatisfaction
Yes
No


MonthlyIncome
DistanceFromHome
Yes
No

1. Job Satisfaction vs Monthly Income:
- There's no strong correlation between job satisfaction and monthly income.
- Higher incomes don't necessarily lead to higher job satisfaction.
- Attrition (red dots) occurs across all income levels and satisfaction ratings.

2. Job Role vs Monthly Income :
- There's significant variation in income across different job roles.
- Managers and Research Directors tend to have higher incomes.
- Lower-paying roles like Laboratory Technicians show more attrition (blue dots).

3. Job Level vs Monthly Income:
- There's a strong positive correlation between job level and monthly income.
- Higher job levels correspond to higher incomes.
- Attrition (red dots) is more common in lower job levels.

4. Age vs Monthly Income :
- There's a general trend of increasing income with age, but with high variability.
- Highest incomes are seen in the 35-55 age range.
- Attrition (red dots) is more prevalent among younger employees and those with lower incomes.

6. DistanceFromHome vs. Monthly Income:

- Red dots (likely representing "Yes") are scattered across various income levels, especially prevalent at lower income levels.
- Green dots (likely representing "No") are more common overall, particularly at higher income levels.
- Income Clusters: Distinct clusters are visible at specific income levels (e.g., 5,000, 10,000, 15,000), possibly indicating salary bands.
- A few high-income outliers (~20,000) exist at various distances, with the outcome mostly being "No."
-A high concentration of both red and green dots at shorter distances from home suggests proximity to work is common across income levels.

8.2.9. Model Building & Training=›

```
from sklearn.model_selection import train_test_split, KFold, cross_validate
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score

# Step 1: Prepare the data
X = df2.drop('Attrition', axis=1)
y = df2['Attrition']

# Step 2: Split the data into training and validation sets
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 3: Initialize the Random Forest model
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

# Step 4: Set up K-Fold Cross-Validation on the training set
kf = KFold(n_splits=5, shuffle=True, random_state=42)

# Step 5: Define scoring metrics
scoring = {
    'accuracy': 'accuracy',
    'precision': 'precision',
    'recall': 'recall',
    'f1': 'f1',
    'roc_auc': 'roc_auc'
}

# Step 6: Perform cross-validation with multiple evaluation metrics on the training set
cv_results = cross_validate(rf_model, X_train, y_train, cv=kf, scoring=scoring, return_train_score=True)

# Step 7: Print evaluation metrics for each fold
print("Evaluation Metrics for Each Fold:")
for metric in scoring.keys():
    print(f"{metric.capitalize()} Scores: {cv_results[f'test_{metric}']}")
    print(f"Mean {metric.capitalize()}: {cv_results[f'test_{metric}'].mean():.4f}")
    print(f"Standard Deviation of {metric.capitalize()}: {cv_results[f'test_{metric}'].std():.4f}")
    print("-" * 50)

# Step 8: Train the model on the entire training set
rf_model.fit(X_train, y_train)

# Step 9: Make predictions on the validation set
y_val_pred = rf_model.predict(X_val)

# Step 10: Print evaluation metrics on the validation set
print("\nEvaluation on the Validation Set:")
print(f"Accuracy: {accuracy_score(y_val, y_val_pred):.4f}")
print(f"Precision: {precision_score(y_val, y_val_pred):.4f}")
print(f"Recall: {recall_score(y_val, y_val_pred):.4f}")
print(f"F1 Score: {f1_score(y_val, y_val_pred):.4f}")
print(f"ROC AUC: {roc_auc_score(y_val, rf_model.predict_proba(X_val)[:, 1]):.4f}")
```

This code consists importing necessary functions and classes from the scikit-learn library:

1. From `sklearn.model_selection`:
   - `train_test_split`: Used to split the dataset into training and validation sets.
   - `KFold`: Implements K-Fold cross-validation splitting strategy.
   - `cross_validate`: Evaluates a score by cross-validation for multiple metrics.

2. From `sklearn.ensemble`:
   - `RandomForestClassifier`: The Random Forest classification algorithm.

3. From `sklearn.metrics`:
   - Various scoring functions: `accuracy_score`, `precision_score`, `recall_score`, `f1_score`, and `roc_auc_score`.
   - These are used to evaluate the model's performance using different metrics.

These imports set up the necessary tools for:
- Splitting the data
- Implementing cross-validation
- Creating and training a Random Forest model
- Evaluating the model's performance using multiple metrics

This code implements a comprehensive evaluation of a Random Forest-
1. Data Preparation:
   - Features (X) and target variable (y) are separated from the dataset.

2. Train-Validation Split:
   - The data is split into 80% training and 20% validation sets.

3. Model Initialization:
   - A Random Forest classifier with 100 trees is created.

4. K-Fold Cross-Validation:
   - 5-fold cross-validation is set up for robust model evaluation.

5. Multiple Evaluation Metrics:
   - Accuracy, precision, recall, F1-score, and ROC AUC are used for comprehensive evaluation.

6. Cross-Validation Results:
   - The model is evaluated on the training set using cross-validation.
   - Results for each metric across all folds are printed, including mean and standard deviation.

7. Final Model Training:
   - The model is trained on the entire training set.

8. Validation Set Evaluation:
   - The trained model is used to make predictions on the validation set.

- Final performance metrics on the validation set are calculated and printed.

Key Insights=›

1. Cross-Validation Results:
   - Accuracy: Mean of 0.8587, which is fairly good. The model correctly classifies about 86% of cases on average.
   - Precision: Mean of 0.7658, indicating that when the model predicts attrition, it's correct about 77% of the time.
   - Recall: Mean of 0.1791, which is quite low. The model is only identifying about 18% of actual attrition cases.
   - F1 Score: Mean of 0.2844, reflecting the imbalance between precision and recall.
   - ROC AUC: Mean of 0.8079, suggesting good overall discrimination ability.

2. Validation Set Results:
   - Accuracy: 0.8510, consistent with cross-validation results.
   - Precision: 0.8333, slightly higher than in cross-validation.
   - Recall: 0.1282, lower than in cross-validation.
   - F1 Score: 0.2222, lower than in cross-validation.
   - ROC AUC: 0.7346, slightly lower than in cross-validation.

-Class Imbalance: The large gap between precision and recall suggests a significant class imbalance, with attrition being the minority class.

-High Precision, Low Recall: The model is cautious in predicting attrition, resulting in few false positives but many false negatives.

-Consistency: The validation set results are generally consistent with cross-validation, indicating the model generalizes reasonably well.

1. Good Discrimination: The ROC AUC scores suggest the model has good ability to distinguish between classes, despite the recall issue.
2. Potential for Improvement: The low recall indicates there's room for improvement, possibly by addressing the class imbalance or tuning the model.

8.2.10. Developing Performance Enhancement Strategies=›

A] Employees Identified for Training and Development Programs::

```
# Create targeted training programs for employees with low job satisfaction or job involvement-

low_satisfaction_employees = df0[(df0['JobSatisfaction'] < 3) | (df0['JobInvolvement'] < 3)]
print("Employees targeted for training and development programs:")
print(low_satisfaction_employees[['EmployeeNumber', 'JobSatisfaction', 'JobInvolvement', 'TrainingTimesLastYear']])
```

```
Employees targeted for training and development programs:
      EmployeeNumber  JobSatisfaction  JobInvolvement  TrainingTimesLastYear
1                  2                2               2                      3
2                  4                3               2                      3
4                  7                2               3                      3
6                 10                1               4                      3
8                 12                3               2                      2
...              ...              ...             ...                    ...
1463            2057                1               3                      2
1464            2060                3               2                      2
1466            2062                1               2                      5
1467            2064                2               4                      0
1468            2065                2               2                      3

[847 rows x 4 columns]
```

Code Explanation:

1. Identifies employees with low job satisfaction or low job involvement.
2. Creates a subset of these employees for targeted training programs.

Key insights:

- The company is focusing on employees with low job satisfaction or low job involvement for training and development programs.
- 647 employees are targeted for these programs.
- The data includes information on how many times each employee received training last year.

B]Employees Targeted for Compensation Review:

```
#'MonthlyIncome' is a top feature-

# Identify employees with below-average income and low satisfaction-
below_avg_income = df0['MonthlyIncome'] < df0['MonthlyIncome'].mean()
low_satisfaction = df0['JobSatisfaction'] < 3

# Target these employees for compensation review
compensation_review_employees = df0[below_avg_income & low_satisfaction]
print("Employees targeted for compensation adjustments:")
print(compensation_review_employees[['EmployeeNumber', 'MonthlyIncome', 'JobSatisfaction']])
```

```
Employees targeted for compensation adjustments:
      EmployeeNumber  MonthlyIncome  JobSatisfaction
1                  2           5130                2
4                  7           3468                2
6                 10           2670                1
10                14           2426                2
16                21           3298                2
...              ...            ...              ...
1449            2038           2439                1
1459            2053           4025                2
1460            2054           3785                1
1467            2064           6142                2
1468            2065           5390                2

[374 rows x 3 columns]
```

Code Explanation:

1. Identifies employees with below-average income and low job satisfaction.
2. Creates a subset of these employees for compensation review.

Key insights:

- The company is focusing on employees with below-average income and low job satisfaction for potential compensation adjustments.
- 374 employees are targeted for compensation review.
- Monthly incomes for these employees range from about $2,400 to $5,400.

C]Model Accuracy and KPI Monitoring by Job Level:

```
print(f"Current model accuracy after implementing strategies: {current_accuracy}")

# Example: Monitor specific KPIs
kpi_monitoring = df1.groupby('JobLevel').agg({
    'Attrition': 'mean',
    'MonthlyIncome': 'mean',
    'JobSatisfaction': 'mean'
})

print("KPI Monitoring after strategy implementation:")
print(kpi_monitoring)
```

```
Current model accuracy after implementing strategies: 0.8809523809523809
KPI Monitoring after strategy implementation:
           Attrition  MonthlyIncome  JobSatisfaction
JobLevel
```

Code Explanation:

1. Calculates the model accuracy after implementing strategies.
2. Groups data by job level and calculates mean attrition, monthly income, and job satisfaction.

Key insights:

- The model accuracy after implementing strategies is 0.8805238095238089.
- KPIs are monitored across different job levels, showing attrition rates, average monthly income, and job satisfaction for each level.
- Job level 5 has the highest monthly income and lowest attrition rate.

9. Key Observations-

1. Age vs Attrition: Younger employees (20-30 years old) tend to leave more frequently, while older employees (40+) show lower attrition rates.

2. Gender vs Attrition: Although male employees outnumber females, attrition rates are similar for both genders.

3. Education and Fields: Life Sciences and Medical fields have the highest employee counts, while Technical Degrees and HR have higher proportional attrition rates.

4. Compensation and Attrition: Lower-income employees (below $5000) are more likely to leave, while those earning more show lower attrition rates. Employees with no stock options also leave more frequently.

5. Work-Related Factors: OverTime and poor work-life balance are strongly correlated with higher attrition. Job satisfaction and higher job levels reduce attrition risks.

6. Job Roles: Sales Executives and Research Scientists face the highest attrition. In contrast, higher-level roles like Directors have lower attrition.

7. Correlation & Feature Importance: Overtime and Monthly Income are the strongest predictors of attrition. Marital status, longer commutes, and lower job levels also correlate with higher attrition.

8. Marital Status Influence: Single employees are more likely to leave compared to married employees. This might suggest that family responsibilities or stability factors influence retention.

9. Job Involvement: Employees with lower job involvement are more likely to leave, indicating a need for engagement strategies that enhance job involvement.

10. Distance from Work: Employees with longer commute times show higher attrition rates, highlighting the potential value of remote work options or location-based incentives.

11. Job Satisfaction Levels: Lower job satisfaction is directly related to higher attrition rates, making satisfaction a key area for improvement

12. Stock Options and Financial Incentives: Employees without stock options tend to leave more, indicating that equity-based incentives can play a crucial role in retaining talent

10. Conclusions & Recommendations-

10.1. Conclusions:

1. Younger Workforce Vulnerability: Younger employees are more likely to leave, indicating potential dissatisfaction or lack of career progression opportunities.

2. Impact of Compensation: Lower-income and non-stock-option employees are more prone to leaving, underscoring the importance of financial incentives.

3. Overtime's Negative Impact: High overtime is a clear predictor of attrition, suggesting that work-life balance is a crucial factor in employee retention.

4 Marital Status as a Retention Factor: Married employees demonstrate lower attrition rates, suggesting that personal stability may play a role in job retention. This highlights the need for organizations to consider personal life factors when developing retention strategies.

5. Job Involvement and Engagement: Employees who are less involved in their jobs are more likely to leave, indicating that low engagement is a critical factor in attrition. This points to a need for enhancing employee engagement through meaningful work, recognition, and involvement in decision-making.

6. Impact of Commute and Location: Longer commute times contribute to higher attrition rates, emphasizing the importance of considering flexible work arrangements or location-based strategies to reduce turnover.

7 Role of Job Satisfaction: Job satisfaction is a significant predictor of attrition. Low job satisfaction correlates directly with higher attrition rates, suggesting that improving job satisfaction through better

management practices, a positive work environment, and meaningful work is essential for retention.

8 Financial Incentives and Equity: The lack of stock options and lower financial compensation are critical factors driving attrition. This indicates that financial incentives, including equity-based rewards, play a crucial role in retaining employees, especially in competitive industries.

9.Importance of Work-Life Balance: The strong correlation between overtime and attrition underlines the importance of work-life balance. Employees who feel overworked or unable to balance their professional and personal lives are more likely to leave, signaling that work-life balance initiatives are essential for retention.

10.2. Recommendations:

1. Targeted Retention Programs for Younger Employees: Implement career development programs, mentorship opportunities, and clearer advancement paths to retain younger talent.

2.Enhance Compensation Packages: Focus on improving financial incentives, particularly for lower-income employees, and consider expanding stock option availability.

3. Reduce Overtime: Encourage work-life balance by limiting overtime and offering flexible working hours, especially in departments with high attrition.

4. Job Role-Specific Strategies: Investigate the reasons behind high attrition in sales and research roles, and develop tailored support and retention initiatives for these positions.

5. Boost Job Satisfaction: Conduct regular employee satisfaction surveys and take actionable steps based on feedback. Focus on improving areas such as management practices, workplace culture, and job autonomy to enhance overall job satisfaction.

6. Financial and Equity Incentives: Review and adjust compensation packages to remain competitive in the market, especially for lower-income employees. Consider introducing or expanding stock option programs to align employee interests with the long-term success of the company.

7. Promotion and Career Advancement: Develop clear career progression plans and communicate them to employees. Ensure that all roles have opportunities for advancement, and provide transparency around promotion criteria and timelines.

6. Expand Training and Development Programs: Invest in continuous learning and development initiatives that offer clear pathways for career progression. This could include mentorship programs, skill-building workshops, and internal promotions based on skill growth.

9.Flexible Work Arrangements: Offer flexible work options, such as remote work or flexible hours, to reduce the impact of long commutes and improve work-life balance. This can also help attract and retain talent in competitive labor markets.