ODISHA UNIVERSITY OF TECHNOLOGY AND RESEARCH

BHUBANESWAR

SCHOOL OF ELECTRONIC SCIENCE

DATA ANALYSIS LAB PROJECT

Topic: Employee Attrition Prediction using Machine Learning

SUBMITTED TO: -

Sidharth Das

Electronic and Communication Dept

SUBMITTED BY: -

NAME: Adya Bramha Samantroy

REGD NO: 24111061

DEPT: Electronics and Communication Engineering

SEM: 3rd

SEC: A

GROUP: A3

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who supported and guided me throughout the successful completion of my project titled **"Employee Attrition Prediction Using Machine Learning."** This work has been a valuable learning experience, and it would not have been possible without the encouragement, assistance, and contributions of several individuals.

I am thankful to all the faculty members of the department for providing a strong academic foundation, access to essential resources, and a motivating environment that encouraged analytical thinking and research-oriented learning. I sincerely appreciate the institution for offering the necessary infrastructure, laboratory facilities, and tools required to carry out this project effectively.

My gratitude extends to the authors, researchers, and online communities whose openly available datasets, research papers, and libraries contributed significantly to the development of this work. Their contributions provided valuable insights into employee behaviour, attrition patterns, and machine learning methodologies used in HR analytics.

Finally, I express my deepest gratitude to my family for their unconditional love, patience, and encouragement. Their continuous support and belief in my abilities have been my greatest source of strength and inspiration during this entire journey.

INDEX

CHAPTER 1

Introduction

1.1 Introduction

This project focuses on predicting whether an employee is likely to leave an organisation using historical HR data. The dataset used contains 1,470 rows and 31 columns describing employee demographics, job details and satisfaction levels. Employee attrition prediction helps HR understand who might stay longer and which employees are at higher risk of leaving.

1.2 Objectives

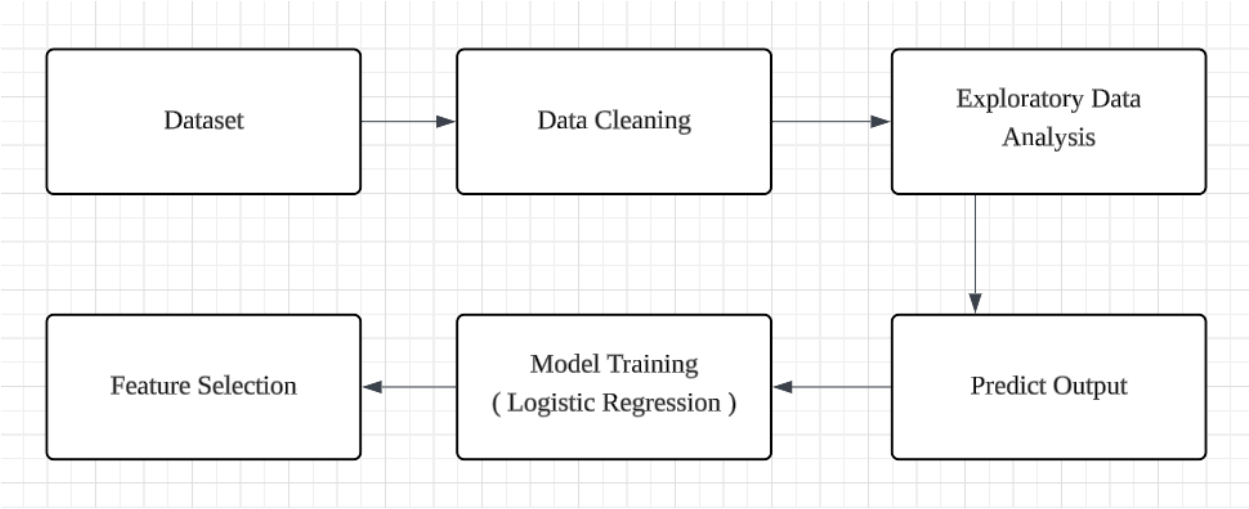
- To build a machine learning model that predicts the probability of employee attrition.
- To identify important numerical and categorical factors that significantly affect attrition using statistical tests and correlations
- To predict whether the employee will stay or not.

1.3 Motivation

High employee turnover increases recruitment cost, training cost and reduces productivity. Using data analytics, HR can proactively understand which profiles are more likely to leave and plan retention strategies. This motivates the use of Python, statistics and machine learning to analyse real-world HR data.

CHAPTER 2

Description of the System / Proposed System



2.1 Dataset Description

The dataset contains employee information such as Age, Department, JobRole, Marital Status, Gender, Monthly Income, Over Time, satisfaction scores and work-experience related fields. There are 1,470 records and 31 attributes including the target variable "Attrition" (Yes/No).

The dataset contains a mix of **numerical, categorical, and ordinal features**. Below is a summary of key columns:

Feature	Type	Description
Age	Numeric	Employee age in years
Department	Categorical	Sales, Human Resources
JobRole	Categorical	Employee's job role
Marital Status	Categorical	Single, Married, Divorced
Gender	Categorical	Male, Female
Monthly Income	Numeric	Employee's monthly salary
Overtime	Categorical	Yes/No
Job Satisfaction	Ordinal	Satisfaction score (1-4)
Environment Satisfaction	Ordinal	Satisfaction score (1-4)
Distance From Home	Numeric	Distance (in km) from home to workplace
Total Working Years	Numeric	Total years of work experience
Years At Company	Numeric	Number of years at the current company
Attrition	Categorical	Target Variable: Yes/No

Observations:

- Majority of employees are in **Research & Development** and **Sales**.
- Attrition (Yes) accounts for approximately **16–18%** of the dataset (imbalanced).
- OverTime is expected to be a **key factor** influencing attrition.

2.2 Data Preprocessing

The first step in data preprocessing involved checking the dataset for missing values. Each of the 31 columns was examined using standard pandas functions, and it was found that no missing values were present. This completeness ensures that statistical calculations and machine learning model training will not be affected by null entries, eliminating the need for imputation.

```
Dropped columns: ['EmployeeCount', 'Over18', 'StandardHours']

Updated column types:
Age                          int64
Attrition                 category
BusinessTravel              object
DailyRate                    int64
Department                  object
DistanceFromHome             int64
Education                    int64
EducationField              object
EnvironmentSatisfaction      int64
Gender                    category
HourlyRate                   int64
JobInvolvement               int64
JobLevel                     int64
JobRole                     object
JobSatisfaction              int64
MaritalStatus               object
MonthlyIncome                int64
MonthlyRate                  int64
NumCompaniesWorked           int64
OverTime                    object
PercentSalaryHike            int64
PerformanceRating            int64
RelationshipSatisfaction     int64
StockOptionLevel             int64
TotalWorkingYears            int64
TrainingTimesLastYear        int64
WorkLifeBalance              int64
YearsAtCompany               int64
YearsInCurrentRole           int64
YearsSinceLastPromotion      int64
YearsWithCurrManager         int64
dtype: object
```

Next, the dataset was inspected for duplicate rows to ensure that no employee records were counted more than once. Upon analysis, it was found that there were zero duplicate rows. This guarantees that the dataset accurately represents individual employee records without any repetition, which is crucial to prevent skewed distributions or biased model predictions.

Certain columns were then identified as non-informative and removed from the dataset. Specifically, EmployeeCount, Over18, and StandardHours were dropped. These columns were either constant across all records or redundant; for example, EmployeeCount is always 1, Over18 is true for all employees, and StandardHours is identical for every record.

Removing these columns reduces noise in the dataset and simplifies subsequent analysis, focusing only on features that could contribute meaningfully to predicting attrition.

The OverTime column initially contained inconsistent text entries such as yes, YES, no, and NO, occasionally with extra spaces. To standardize this important categorical feature, all values were normalized to Yes and No. This normalization is essential because inconsistent categorical labels could lead to incorrect counts, biased statistics, or errors during model training.

Finally, the Attrition and Gender columns were converted to categorical data types. This conversion enables efficient memory usage, proper handling of categorical features during analysis, and correct encoding when building machine learning models. Treating these columns as categorical ensures that they are appropriately processed in both statistical tests and predictive modeling.

CHAPTER 3

Techniques used / Proposed Methodology

3.1 Exploratory Data Analysis (EDA)

After preprocessing, the dataset of 1,470 records with 31 attributes was explored to understand its structure and characteristics. The dataset shape and first few rows were displayed to verify proper loading. Summary statistics for numerical features—mean, median, min, max, quartiles, and standard deviation—were computed to assess central tendency and variability, with boxplots and histograms (sns.boxplot, sns.histplot) providing visual confirmation of distributions and potential outliers.

Categorical variables such as Department, JobRole, Gender, MaritalStatus, BusinessTravel, OverTime, and Attrition were analyzed using value counts and countplots (sns.countplot) to visualize class distributions. Pearson correlation was calculated for numerical features like Age, MonthlyIncome, and YearsAtCompany, while Spearman correlation was used for ordinal features such as Education, JobLevel, WorkLifeBalance, and JobSatisfaction. These correlations were displayed using heatmaps (sns.heatmap) to highlight relationships and guide feature selection for modeling.

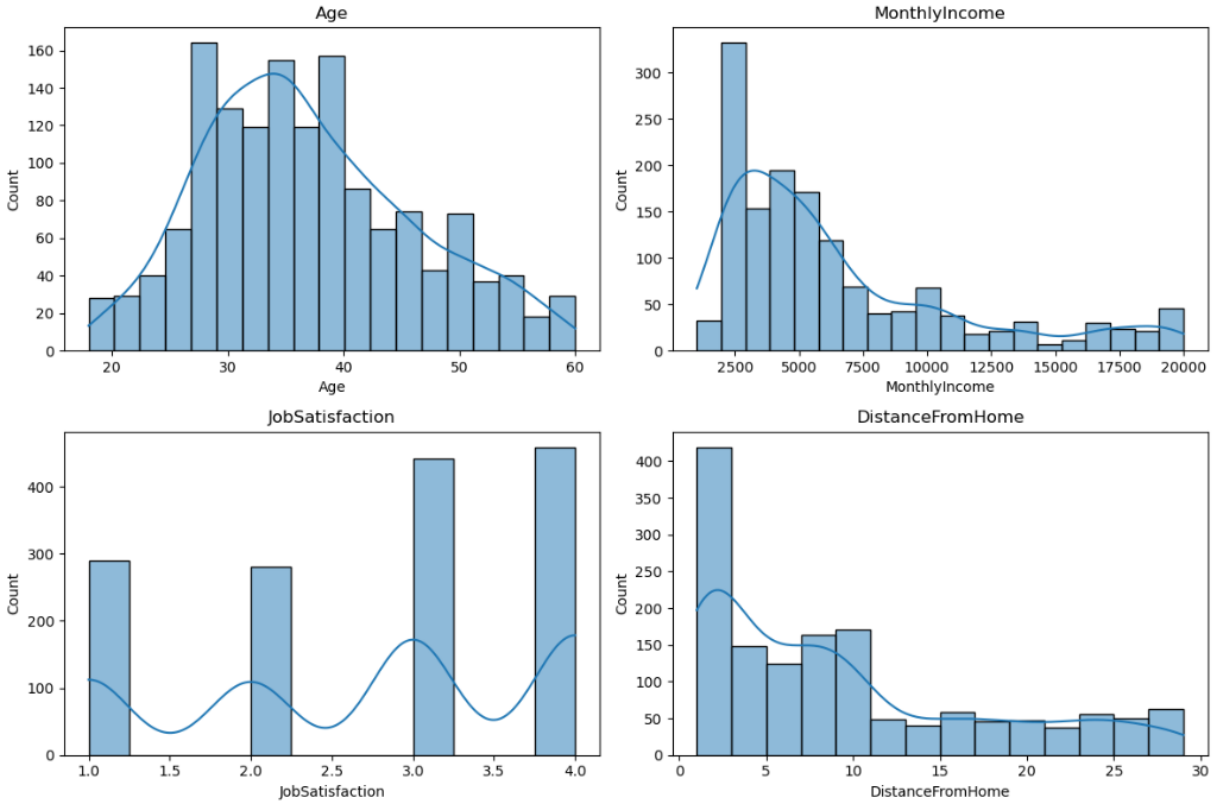
This analysis provides a clear overview of the dataset, identifies key features influencing attrition, and sets the foundation for predictive modeling, combining both statistical summaries and visual insights to support further analysis.

3.2 Statistic Analysis

To understand the relationships between employee attributes and attrition, statistical tests were conducted.

Chi-square tests were performed between the target variable Attrition and categorical features such as Gender, Department, BusinessTravel, JobRole, MaritalStatus, EducationField, and OverTime to evaluate whether these factors are significantly associated with attrition. Variables with p-values less than 0.05 were considered to have a significant relationship with attrition.

For numerical variables including MonthlyIncome, Age, DistanceFromHome, YearsAtCompany, and JobSatisfaction, independent t-tests were conducted to assess whether the means differ significantly between employees who left versus those who stayed. Again, p-values below 0.05 indicated a statistically significant difference between the “Yes” and “No” attrition groups.

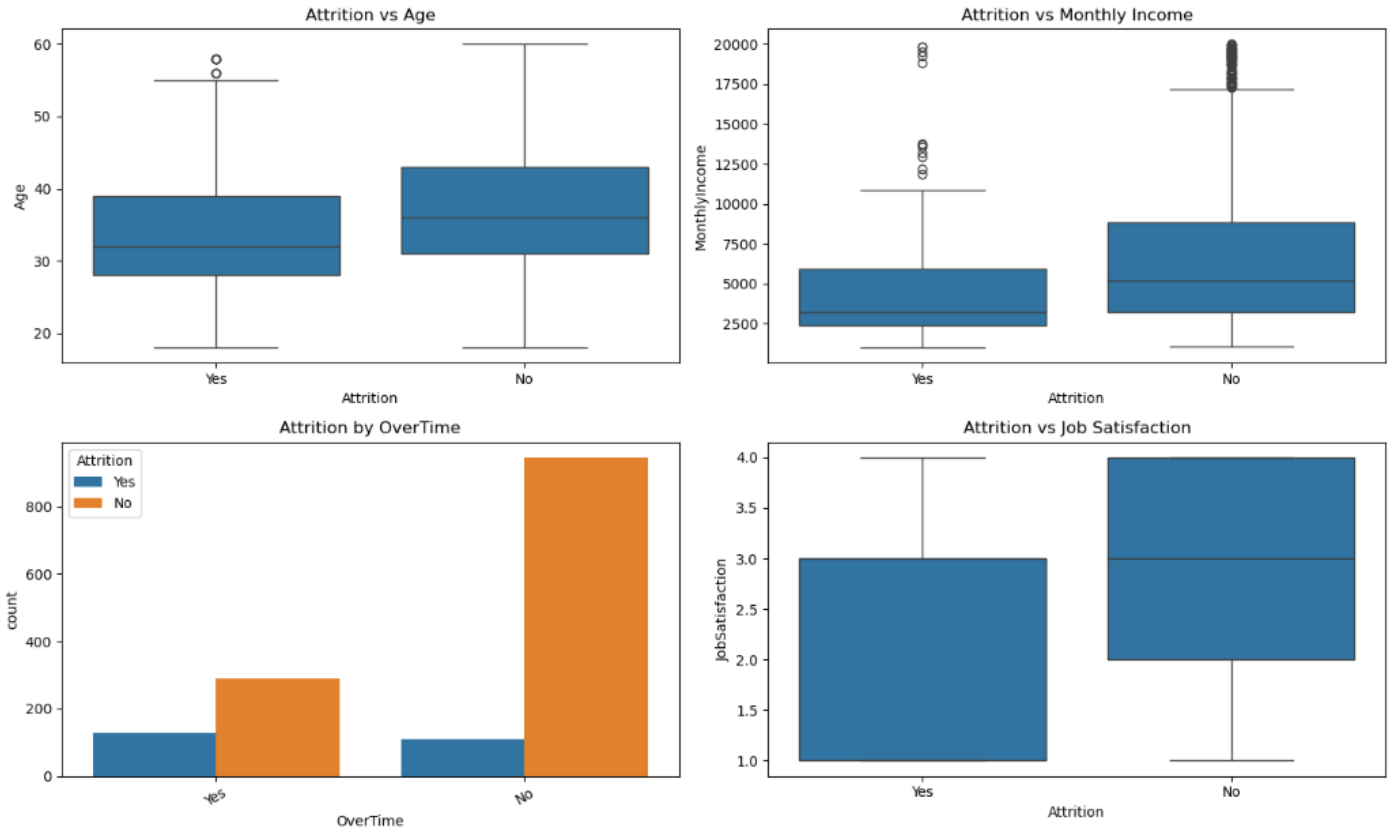


3.3 Feature Selection

For building the predictive model, feature selection was performed to identify the most influential variables related to attrition. Based on correlation analysis and statistical significance, the top predictors chosen were Age, MonthlyIncome, JobSatisfaction, OverTime, and DistanceFromHome. These features were selected as they demonstrated strong relationships with the target variable and are easily interpretable, which is important for HR decision-making.

3.4 Machine Learning Model (Logistic Regression)

A Logistic Regression model was employed to predict the binary attrition outcome. Categorical features such as OverTime were encoded using One-Hot Encoding to convert them into a numerical format suitable for the model. The dataset was split into training and testing sets with an 80:20 ratio, ensuring that the model could learn patterns from a large portion of data and be evaluated on unseen samples. The model was trained with a maximum of 300 iterations to ensure convergence. Model performance was assessed using a classification report, which included metrics such as precision, recall, and F1-score, providing a comprehensive view of its predictive ability.



CHAPTER 4

Result and Discussion

4.1 Model Performance

The Logistic Regression model was evaluated on a test set consisting of 294 employee records. Overall, the model achieved an accuracy of approximately 87%, indicating that it correctly predicted the attrition status for the majority of employees. Breaking this down by class, the model performed very well for employees who did not leave (Class 0), with a precision of 0.87, recall of 1.0, and F1-score of 0.93. This demonstrates the model's strong ability to identify employees who are likely to stay. However, for employees who left the company (Class 1), performance was notably lower, with a precision of 0.50, recall of 0.03, and F1-score of 0.05. This disparity highlights the challenge posed by the imbalanced dataset, where the number of "Yes" attrition cases is much smaller than "No" cases. While the model provides reliable predictions for retention, it underperforms in identifying potential attrition cases, which is a common limitation in classification tasks with class imbalance.

4.2 Sample Predictions and Insight

To illustrate the practical use of the model, sample predictions were performed for hypothetical employees. For instance, an employee aged 22, earning a monthly income of 5,000, working overtime, with a low job satisfaction score of 1 and living 25 km from the office, was predicted to have a high attrition probability of approximately 0.67, classifying them as likely to leave. By varying combinations of features such as age, income, overtime status, job satisfaction, and distance from home, the model can provide both a binary attrition prediction and a probability score. These predictions help identify employees at high risk of leaving, allowing HR to take proactive retention measures. Additionally, statistical tests conducted earlier support the selection of these features, confirming that variables like overtime, job role, business travel, marital status, and key numerical attributes (e.g., MonthlyIncome, Age, TotalWorkingYears, YearsAtCompany) have significant associations with attrition.

```
sample = pd.DataFrame({
    "Age": [22],
    "MonthlyIncome": [5000],
    "OverTime": ["Yes"],
    "JobSatisfaction": [1],
    "DistanceFromHome": [25]
})
prediction = model.predict(sample)[0]
prob = model.predict_proba(sample)[0][1]
print("Prediction =", prediction)
print("Probability of Attrition =", prob)
```

```
Prediction = 1
Probability of Attrition = 0.6709265976809173
```

CHAPTER 5

Advantages and Applications

5.1 Advantages

The Employee Attrition Prediction system provides HR teams with a quantitative estimate of each employee's likelihood of leaving, allowing for data-driven decision-making. By using a simple and interpretable logistic regression model, it focuses on key features such as age, monthly income, job satisfaction, overtime, and distance from home, which are easily understandable by non-technical stakeholders. The model's simplicity ensures transparency, and HR can justify retention strategies based on measurable factors. Moreover, the system reduces reliance on guesswork, helping prioritize interventions for employees who are at higher risk of attrition.

5.2 Applications

This predictive system can be used in multiple HR scenarios. During recruitment, it can screen candidates whose profiles align with employees likely to stay long-term, ensuring better workforce stability. For existing employees, it highlights those at risk of leaving, enabling timely interventions such as salary adjustments, reduced overtime, career development programs, or role changes. In addition, HR dashboards can incorporate the model's output to visualize attrition probabilities across departments, teams, or locations, making it easier to monitor overall workforce health and plan retention strategies strategically.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

The Employee Attrition Prediction project successfully demonstrates how logistic regression can be used to estimate the likelihood of employees leaving an organization. The model achieves reasonable accuracy in predicting overall attrition, particularly for employees who are likely to stay. Through statistical analysis and exploratory data analysis, key factors influencing attrition—such as overtime status, monthly income, job satisfaction, age, and distance from home—have been clearly identified, providing actionable insights for HR teams. However, a notable limitation of the current model is its low performance in predicting the minority class (employees who actually leave), which is primarily due to class imbalance in the dataset. Despite this limitation, the model offers a transparent and interpretable approach to workforce analytics, helping organizations make data-driven retention decisions.

6.2 Future Scope

Future improvements can focus on addressing the class imbalance to enhance prediction for employees at risk of leaving. Techniques such as SMOTE (Synthetic Minority Over-sampling Technique) or applying class weights in the logistic regression model can help mitigate this issue. Additionally, employing more advanced machine learning algorithms, including Random Forest, Gradient Boosting, or XGBoost, may capture non-linear relationships and improve predictive performance. Incorporating more HR-related features, such as historical performance evaluations, training records, and manager feedback, can further enhance model accuracy and provide a more holistic understanding of attrition drivers. Finally, deploying the model as an interactive web dashboard would allow HR personnel to input employee details and receive real-time attrition probability predictions, enabling proactive interventions and strategic workforce management.

APPENDIX

1. Exploratory Data Analysis (EDA) on Employee Attributes

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

df = pd.read_csv("C:/Users/abds0/Desktop/DAP project/dataset.csv")
print("Rows and Columns:", df.shape)
print("\nFirst 5 rows:")
print(df.head())
categorical_cols = [
    "Attrition", "BusinessTravel", "Department", "Gender",
    "JobRole", "MaritalStatus"]
for col in categorical_cols:
    print(f"\nValue counts for {col}:")
    print(df[col].value_counts())
```

```
Rows and Columns: (1470, 31)

First 5 rows:
   Age Attrition     BusinessTravel  DailyRate              Department  \
0   41       Yes      Travel_Rarely       1102                   Sales
1   49        No  Travel_Frequently        279  Research & Development
2   37       Yes      Travel_Rarely       1373  Research & Development
3   33        No  Travel_Frequently       1392  Research & Development
4   27        No      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]
```

```
Value counts for Attrition:
Attrition
No     1233
Yes     237
Name: count, dtype: int64

Value counts for Department:
Department
Research & Development    961
Sales                     446
Human Resources            63
Name: count, dtype: int64

Value counts for Gender:
Gender
Male      882
Female    588
Name: count, dtype: int64

Value counts for JobRole:
JobRole
Sales Executive              326
Research Scientist           292
Laboratory Technician        259
Manufacturing Director       145
Healthcare Representative    131
Manager                      102
Sales Representative          83
Research Director             80
Human Resources               52
Name: count, dtype: int64

Value counts for MaritalStatus:
MaritalStatus
Married     673
Single      470
Divorced    327
Name: count, dtype: int64
```

2. Data Cleaning

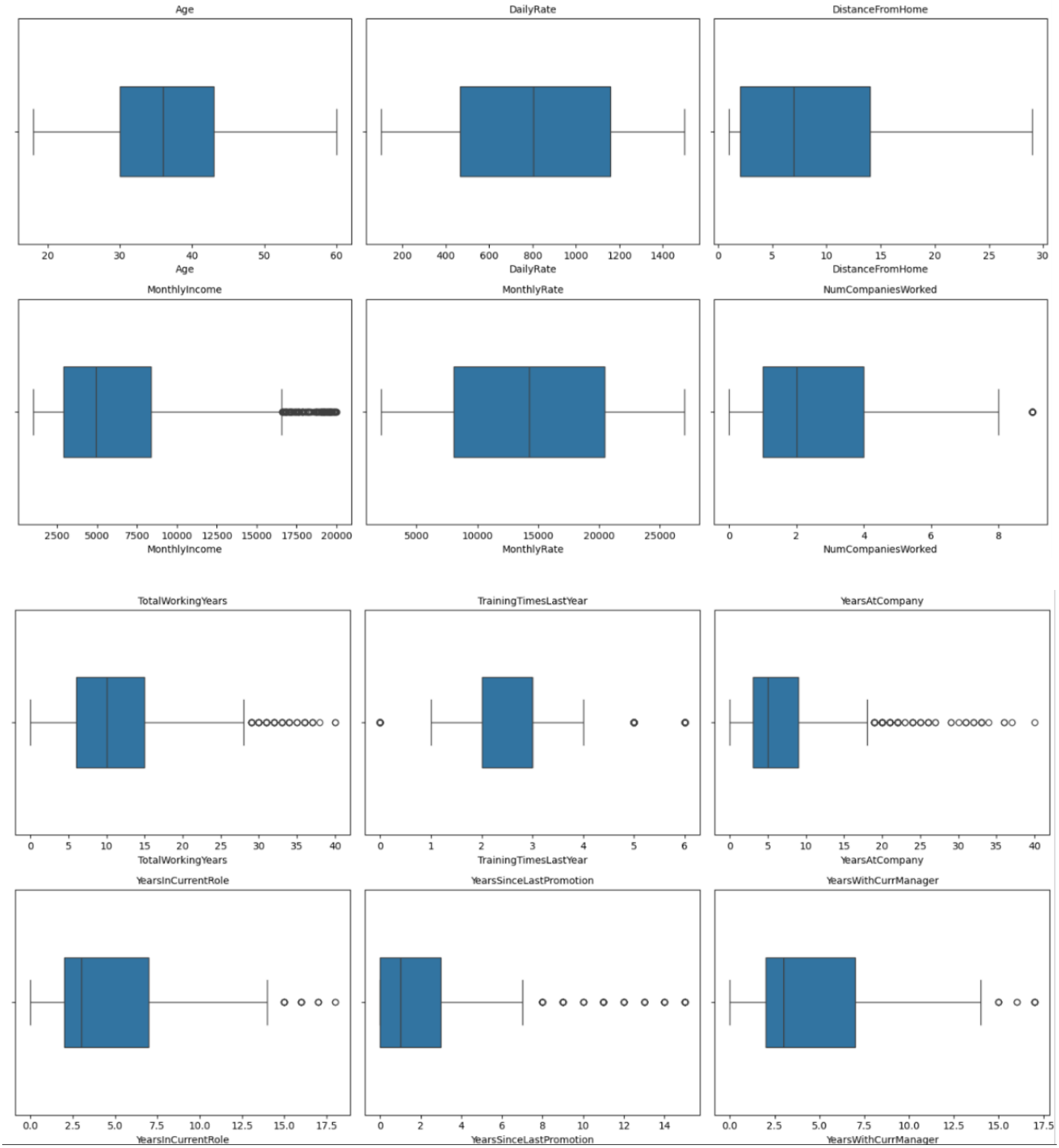
```python
import pandas as pd
df = pd.read_csv("C:/Users/abds0/Desktop/DAP project/dataset.csv")
print("\nMissing values in each column:")
print(df.isnull().sum())
print("\nDuplicate rows before:", df.duplicated().sum())
df = df.drop_duplicates()
print("Duplicate rows after:", df.duplicated().sum())
df['OverTime'] = df['OverTime'].str.strip()
df['OverTime'] = df['OverTime'].replace({
    'yes': 'Yes',
    'YES': 'Yes',
    'no': 'No',
    'NO': 'No'
})
remove_cols = ["EmployeeCount", "Over18", "StandardHours"]
df = df.drop(columns=["EmployeeCount", "Over18", "StandardHours"], errors='ignore')
print("\nDropped columns:", remove_cols)
df['Attrition'] = df['Attrition'].astype('category')
df['Gender'] = df['Gender'].astype('category')
print("\nUpdated column types:")
print(df.dtypes)
```

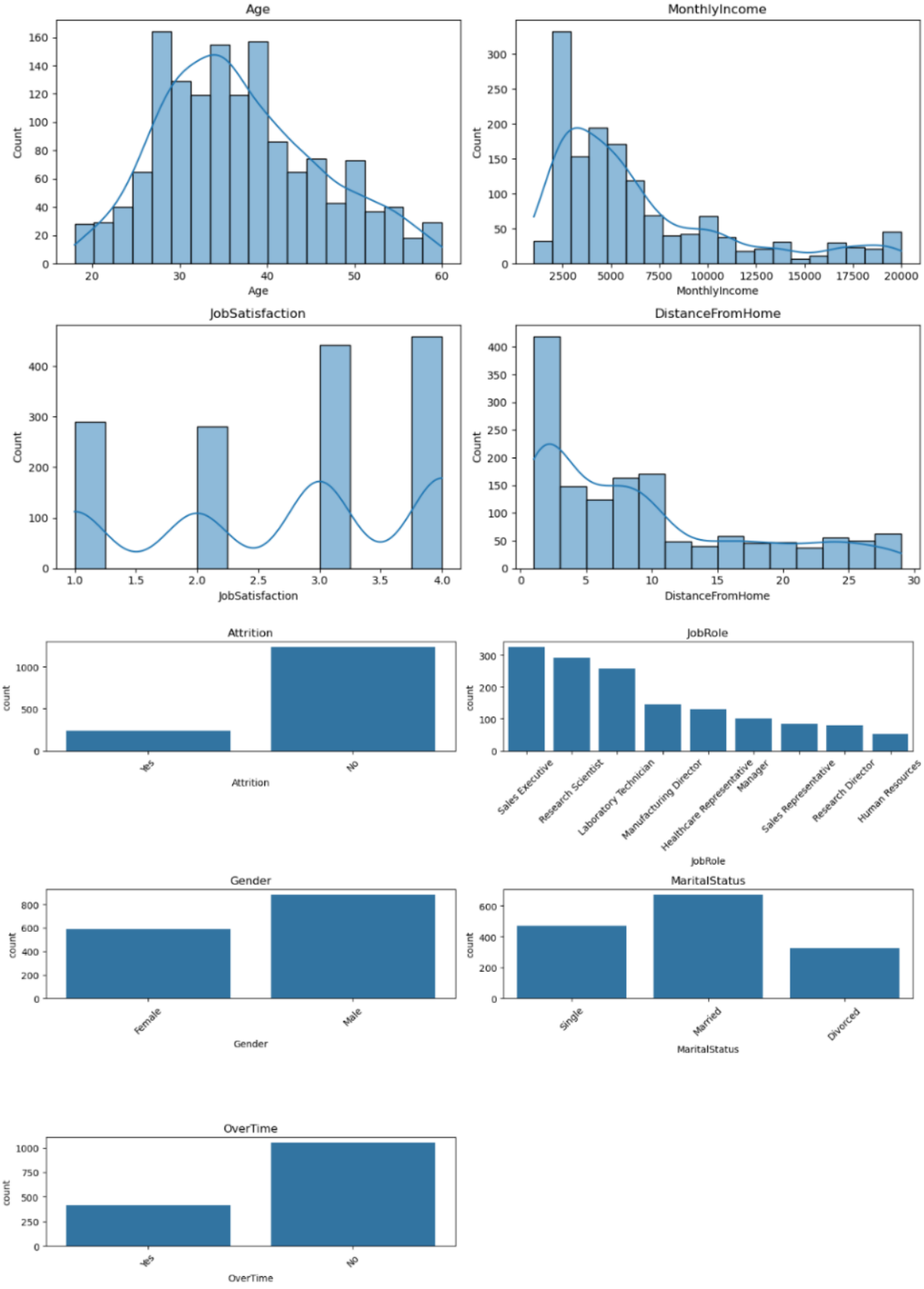
```
Missing values in each column:
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64

Duplicate rows before: 0
Duplicate rows after: 0
```

```
Dropped columns: ['EmployeeCount', 'Over18', 'StandardHours']

Updated column types:
Age                          int64
Attrition                 category
BusinessTravel              object
DailyRate                    int64
Department                  object
DistanceFromHome             int64
Education                    int64
EducationField              object
EnvironmentSatisfaction      int64
Gender                    category
HourlyRate                   int64
JobInvolvement               int64
JobLevel                     int64
JobRole                     object
JobSatisfaction              int64
MaritalStatus               object
MonthlyIncome                int64
MonthlyRate                  int64
NumCompaniesWorked           int64
OverTime                    object
PercentSalaryHike            int64
PerformanceRating            int64
RelationshipSatisfaction     int64
StockOptionLevel             int64
TotalWorkingYears            int64
TrainingTimesLastYear        int64
WorkLifeBalance              int64
YearsAtCompany               int64
YearsInCurrentRole           int64
YearsSinceLastPromotion      int64
YearsWithCurrManager         int64
dtype: object
```

3. Outlier Detection and Univariate, Bivariate & Multivariate Visualizations-

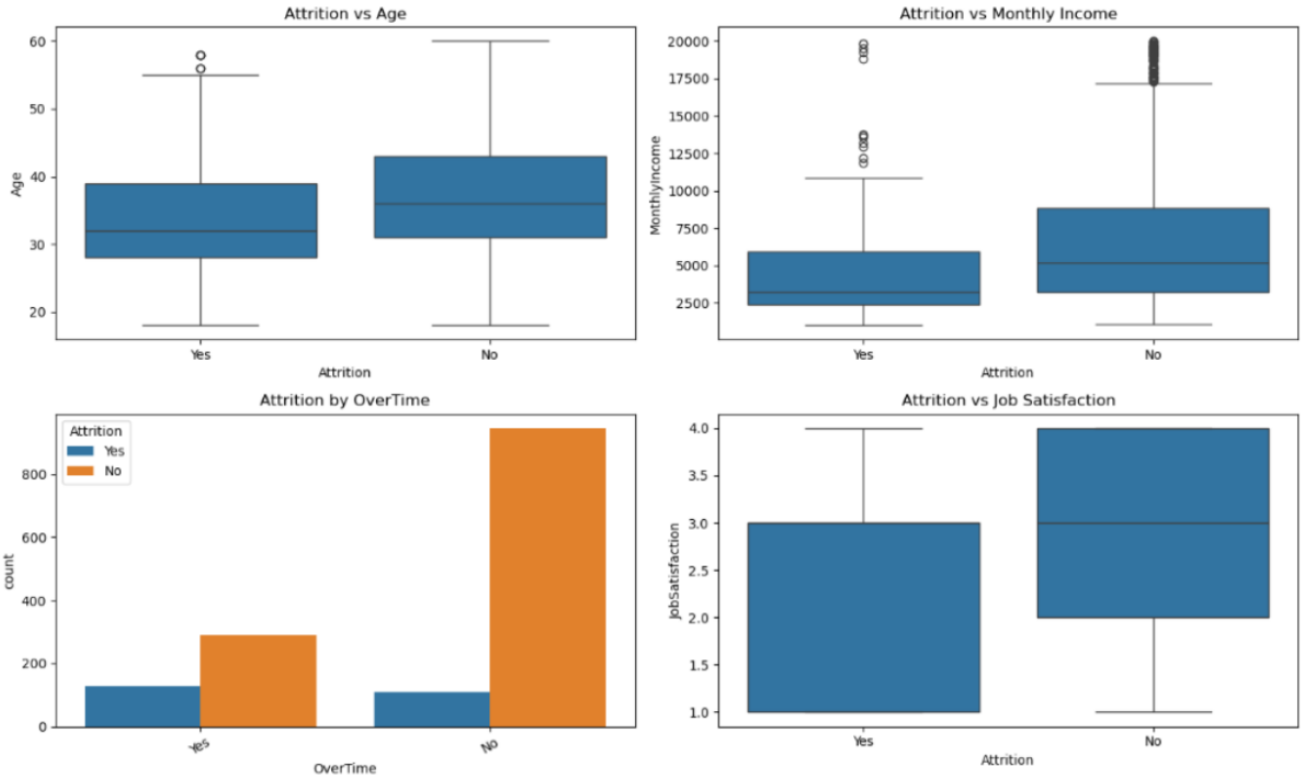
```python
import matplotlib.pyplot as plt
import seaborn as sns
import math
num_cols = [
    "Age", "DailyRate", "DistanceFromHome", "MonthlyIncome",
    "MonthlyRate", "NumCompaniesWorked", "TotalWorkingYears",
    "TrainingTimesLastYear", "YearsAtCompany", "YearsInCurrentRole",
    "YearsSinceLastPromotion", "YearsWithCurrManager"]
num_cols = [c for c in num_cols if c in df.columns]
n = len(num_cols)
rows = math.ceil(n / 3)   # grid with 3 columns
cols = 3
plt.figure(figsize=(15, 4 * rows))
for i, col in enumerate(num_cols, 1):
    plt.subplot(rows, cols, i)
    sns.boxplot(x=df[col], width=0.4)
    plt.title(col, fontsize=10)
    plt.tight_layout()
plt.suptitle("Boxplots of Numerical Features (Compact View)", fontsize=14, y=1.02)
plt.show()
```


Age
20 30 40 50 60
Age
DailyRate
200 400 600 800 1000 1200 1400
DailyRate
DistanceFromHome
0 5 10 15 20 25 30
DistanceFromHome
MonthlyIncome
2500 5000 7500 10000 12500 15000 17500 20000
MonthlyIncome
MonthlyRate
5000 10000 15000 20000 25000
MonthlyRate
NumCompaniesWorked
0 2 4 6 8
NumCompaniesWorked
TotalWorkingYears
0 5 10 15 20 25 30 35 40
TotalWorkingYears
TrainingTimesLastYear
0 1 2 3 4 5 6
TrainingTimesLastYear
YearsAtCompany
0 5 10 15 20 25 30 35 40
YearsAtCompany
YearsInCurrentRole
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5
YearsInCurrentRole
YearsSinceLastPromotion
0 2 4 6 8 10 12 14
YearsSinceLastPromotion
YearsWithCurrManager
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5
YearsWithCurrManager


Age
Count
Age
MonthlyIncome
Count
MonthlyIncome
JobSatisfaction
Count
JobSatisfaction
DistanceFromHome
Count
DistanceFromHome
Attrition
count
Yes
No
Attrition
JobRole
count
Sales Executive
Research Scientist
Laboratory Technician
Manufacturing Director
Healthcare Representative
Manager
Sales Representative
Research Director
Human Resources
JobRole
Gender
count
Female
Male
Gender
MaritalStatus
count
Single
Married
Divorced
MaritalStatus
OverTime
count
Yes
No
OverTime

4. Correlation Analysis

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

plt.figure(figsize=(14, 12))

# ---------- Attrition vs Age ----------
plt.subplot(3, 2, 1)
sns.boxplot(x='Attrition', y='Age', data=df)
plt.title("Attrition vs Age")

# ---------- Attrition vs MonthlyIncome ----------
plt.subplot(3, 2, 2)
sns.boxplot(x='Attrition', y='MonthlyIncome', data=df)
plt.title("Attrition vs Monthly Income")

# ---------- Attrition vs OverTime ----------
plt.subplot(3, 2, 3)
sns.countplot(x='OverTime', hue='Attrition', data=df)
plt.title("Attrition by OverTime")
plt.xticks(rotation=30)

# ---------- Attrition vs JobSatisfaction ----------
plt.subplot(3, 2, 4)
sns.boxplot(x='Attrition', y='JobSatisfaction', data=df)
plt.title("Attrition vs Job Satisfaction")

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



5. Hypothesis Testing

```
from scipy.stats import ttest_ind

print("\n\n🔹 t-TESTS (Numerical vs Attrition)\n")

df['Attrition_numeric'] = df['Attrition'].map({'Yes':1, 'No':0})

num_cols_to_test = [
    'MonthlyIncome','Age','DailyRate','HourlyRate',
    'MonthlyRate','TotalWorkingYears','DistanceFromHome',
    'YearsAtCompany','YearsInCurrentRole','YearsSinceLastPromotion'
]

for col in num_cols_to_test:
    yes = df[df['Attrition']=='Yes'][col].dropna()
    no = df[df['Attrition']=='No'][col].dropna()

    t_stat, p = ttest_ind(yes, no, equal_var=False)

    print(f"{col}: p-value = {p:.4f}  --> {'Significant' if p < 0.05 else 'Not Significant'}")
```

```
🔹 t-TESTS (Numerical vs Attrition)

MonthlyIncome: p-value = 0.0000  --> Significant
Age: p-value = 0.0000  --> Significant
DailyRate: p-value = 0.0300  --> Significant
HourlyRate: p-value = 0.7914  --> Not Significant
MonthlyRate: p-value = 0.5653  --> Not Significant
TotalWorkingYears: p-value = 0.0000  --> Significant
DistanceFromHome: p-value = 0.0041  --> Significant
YearsAtCompany: p-value = 0.0000  --> Significant
YearsInCurrentRole: p-value = 0.0000  --> Significant
YearsSinceLastPromotion: p-value = 0.1987  --> Not Significant
```

```
from scipy.stats import chi2_contingency
import pandas as pd

print("\n🔹 CHI-SQUARE TESTS (Categorical vs Categorical)\n")

cat_cols = [
    'Gender','Department','BusinessTravel','JobRole',
    'MaritalStatus','EducationField','OverTime'
]

chi_results = {}

for col in cat_cols:
    table = pd.crosstab(df['Attrition'], df[col])
    chi2, p, dof, exp = chi2_contingency(table)
    chi_results[col] = p
    print(f"{col}: p-value = {p:.4f}  --> {'Significant' if p < 0.05 else 'Not Significant'}")
```

```
🔹 CHI-SQUARE TESTS (Categorical vs Categorical)

Gender: p-value = 0.2906  --> Not Significant
Department: p-value = 0.0045  --> Significant
BusinessTravel: p-value = 0.0000  --> Significant
JobRole: p-value = 0.0000  --> Significant
MaritalStatus: p-value = 0.0000  --> Significant
EducationField: p-value = 0.0068  --> Significant
OverTime: p-value = 0.0000  --> Significant
```

6. Training Model

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report

df = pd.read_csv("C:/Users/abds0/Desktop/DAP project/dataset.csv")
df["Attrition"] = df["Attrition"].map({"Yes": 1, "No": 0})
simple_features = ["Age", "MonthlyIncome", "OverTime", "JobSatisfaction", "DistanceFromHome"]
X = df[simple_features]
y = df["Attrition"]
cat_cols = ["OverTime"]
num_cols = ["Age", "MonthlyIncome", "JobSatisfaction", "DistanceFromHome"]
preprocess = ColumnTransformer([
    ("cat", OneHotEncoder(drop='first'), cat_cols),
    ("num", "passthrough", num_cols)])
model = Pipeline([
    ("preprocess", preprocess),
    ("classifier", LogisticRegression(max_iter=300))])
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("\nCLASSIFICATION REPORT:")
print(classification_report(y_test, y_pred))
print("\nModel is trained successfully!")
```

7. Predicting whether the Employee will stay or not.

```
sample = pd.DataFrame({
    "Age": [22],
    "MonthlyIncome": [5000],
    "OverTime": ["Yes"],
    "JobSatisfaction": [1],
    "DistanceFromHome": [25]
})
prediction = model.predict(sample)[0]
prob = model.predict_proba(sample)[0][1]
print("Prediction =", prediction)
print("Probability of Attrition =", prob)
```

```
Prediction = 1
Probability of Attrition = 0.6709265976809173
```

```
new_data = pd.DataFrame({
    "Age": [25, 18],
    "MonthlyIncome": [20000, 7000],
    "OverTime": ["No", "Yes"],
    "JobSatisfaction": [4, 2],
    "DistanceFromHome": [5, 13]
})
preds = model.predict(new_data)
probs = model.predict_proba(new_data)[:,1]
print(preds)
print(probs)
```

```
[0 1]
[0.01970809 0.50760877]
```