```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

```
data = pd.read_csv('Employee-Attrition.csv')
```

```
data.isnull().sum()
```

```
Age                       0
Attrition                 0
BusinessTravel            0
DailyRate                 0
Department                0
DistanceFromHome          0
Education                 0
EducationField            0
EmployeeCount             0
EmployeeNumber            0
EnvironmentSatisfaction   0
Gender                    0
HourlyRate                0
JobInvolvement            0
JobLevel                  0
JobRole                   0
JobSatisfaction           0
MaritalStatus             0
MonthlyIncome             0
MonthlyRate               0
NumCompaniesWorked        0
Over18                    0
OverTime                  0
PercentSalaryHike         0
PerformanceRating         0
RelationshipSatisfaction  0
StandardHours             0
StockOptionLevel          0
TotalWorkingYears         0
TrainingTimesLastYear     0
WorkLifeBalance           0
YearsAtCompany            0
YearsInCurrentRole        0
YearsSinceLastPromotion   0
YearsWithCurrManager      0
dtype: int64
```

```
label_encoder = LabelEncoder()
categorical_columns = ['EducationField', 'JobInvolvement', 'JobSatisfaction', 'Gender',
                       'JobRole', 'Over18', 'OverTime', 'MaritalStatus', 'BusinessTravel', 'Department']
for col in categorical_columns:
    data[col] = label_encoder.fit_transform(data[col])

data.head(10)
```

| | Age | Attrition | BusinessTravel | DailyRate | Department | DistanceFromHome | Education |
|---|---|---|---|---|---|---|---|
| **0** | 41 | Yes | 2 | 1102 | 2 | 1 | 1 |
| **1** | 49 | No | 1 | 279 | 1 | 8 | 0 |
| **2** | 37 | Yes | 2 | 1373 | 1 | 2 | 1 |
| **3** | 33 | No | 1 | 1392 | 1 | 3 | 3 |
| **4** | 27 | No | 2 | 591 | 1 | 2 | 0 |
| **5** | 32 | No | 1 | 1005 | 1 | 2 | 1 |
| **6** | 59 | No | 2 | 1324 | 1 | 3 | 2 |
| **7** | 30 | No | 2 | 1358 | 1 | 24 | 0 |
| **8** | 38 | No | 1 | 216 | 1 | 23 | 2 |
| **9** | 36 | No | 2 | 1299 | 1 | 27 | 2 |

10 rows × 35 columns

```
categorical_columns = ['EducationField', 'JobInvolvement', 'JobSatisfaction', 'Gender',
                       'JobRole', 'MaritalStatus', 'Over18', 'OverTime', 'BusinessTravel',
```

```
# Use one-hot encoding for the remaining categorical columns
data_encoded = pd.get_dummies(data, columns=categorical_columns, drop_first=True)

data.head(10)
```

| | Age | Attrition | BusinessTravel | DailyRate | Department | DistanceFromHome | Education |
|---|---|---|---|---|---|---|---|
| **0** | 41 | Yes | 2 | 1102 | 2 | 1 | 1 |
| **1** | 49 | No | 1 | 279 | 1 | 8 | 0 |
| **2** | 37 | Yes | 2 | 1373 | 1 | 2 | 1 |
| **3** | 33 | No | 1 | 1392 | 1 | 3 | 3 |
| **4** | 27 | No | 2 | 591 | 1 | 2 | 0 |
| **5** | 32 | No | 1 | 1005 | 1 | 2 | 1 |
| **6** | 59 | No | 2 | 1324 | 1 | 3 | 2 |
| **7** | 30 | No | 2 | 1358 | 1 | 24 | 0 |
| **8** | 38 | No | 1 | 216 | 1 | 23 | 2 |
| **9** | 36 | No | 2 | 1299 | 1 | 27 | 2 |

10 rows × 35 columns

```
X = data_encoded.drop(['Attrition'], axis=1)
y = data_encoded['Attrition']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
logistic_model = LogisticRegression(random_state=42, max_iter=1500)
logistic_model.fit(X_train, y_train)

# Predict using the Logistic Regression model
logistic_predictions = logistic_model.predict(X_test)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

```
logistic_accuracy = accuracy_score(y_test, logistic_predictions)
logistic_classification_report = classification_report(y_test, logistic_predictions, zero_division=1)
logistic_confusion_matrix = confusion_matrix(y_test, logistic_predictions)
```

```
decision_tree_model = DecisionTreeClassifier(random_state=42)
decision_tree_model.fit(X_train, y_train)
```

```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

```
decision_tree_predictions = decision_tree_model.predict(X_test)

# Calculate performance metrics for Decision Tree
decision_tree_accuracy = accuracy_score(y_test, decision_tree_predictions)
decision_tree_classification_report = classification_report(y_test, decision_tree_predictions)
decision_tree_confusion_matrix = confusion_matrix(y_test, decision_tree_predictions)
```

```
print("Logistic Regression Metrics:")
print(f"Accuracy: {logistic_accuracy:.2f}")
print("Classification Report:")
print(logistic_classification_report)
print("Confusion Matrix:")
print(logistic_confusion_matrix)

print("\nDecision Tree Metrics:")
print(f"Accuracy: {decision_tree_accuracy:.2f}")
print("Classification Report:")
print(decision_tree_classification_report)
```

```
print("Confusion Matrix:")
print(decision_tree_confusion_matrix)
```

```
Logistic Regression Metrics:
Accuracy: 0.87
Classification Report:
              precision    recall  f1-score   support

          No       0.88      0.98      0.93       255
         Yes       0.56      0.13      0.21        39

    accuracy                           0.87       294
   macro avg       0.72      0.56      0.57       294
weighted avg       0.84      0.87      0.83       294

Confusion Matrix:
[[251   4]
 [ 34   5]]

Decision Tree Metrics:
Accuracy: 0.77
Classification Report:
              precision    recall  f1-score   support

          No       0.89      0.85      0.87       255
         Yes       0.22      0.28      0.25        39

    accuracy                           0.77       294
   macro avg       0.55      0.56      0.56       294
weighted avg       0.80      0.77      0.78       294

Confusion Matrix:
[[216  39]
 [ 28  11]]
```