In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from ydata_profiling import ProfileReport
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.preprocessing import LabelEncoder
from imblearn.over_sampling import SMOTE
from collections import Counter
import warnings
warnings.simplefilter(action = 'ignore')
```

In [2]:
```
df = pd.read_csv(r"C:\Users\HP\Downloads\Indicino project.xlsx - Attrition_data.csv
df.head()
```

Out[2]:

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education
0	41	Yes	Travel_Rarely	1102	Sales	1	2
1	49	No	Travel_Frequently	279	Research & Development	8	1
2	37	Yes	Travel_Rarely	1373	Research & Development	2	2
3	33	No	Travel_Frequently	1392	Research & Development	3	4
4	27	No	Travel_Rarely	591	Research & Development	2	1

5 rows × 35 columns

In [3]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1470 entries, 0 to 1469
Data columns (total 35 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   Age                       1470 non-null   int64 
 1   Attrition                 1470 non-null   object
 2   BusinessTravel            1470 non-null   object
 3   DailyRate                 1470 non-null   int64 
 4   Department                1470 non-null   object
 5   DistanceFromHome          1470 non-null   int64 
 6   Education                 1470 non-null   int64 
 7   EducationField            1470 non-null   object
 8   EmployeeCount             1470 non-null   int64 
 9   EmployeeNumber            1470 non-null   int64 
 10  EnvironmentSatisfaction   1470 non-null   int64 
 11  Gender                    1470 non-null   object
 12  HourlyRate                1470 non-null   int64 
 13   JobInvolvement           1470 non-null   object
 14  JobLevel                  1470 non-null   int64 
 15  JobRole                   1470 non-null   object
 16  JobSatisfaction           1470 non-null   int64 
 17  MaritalStatus             1470 non-null   object
 18  MonthlyIncome             1470 non-null   int64 
 19  MonthlyRate               1470 non-null   int64 
 20  NumCompaniesWorked        1470 non-null   int64 
 21  Over18                    1470 non-null   object
 22  OverTime                  1470 non-null   object
 23  PercentSalaryHike         1470 non-null   int64 
 24  PerformanceRating         1470 non-null   int64 
 25  RelationshipSatisfaction  1470 non-null   int64 
 26  StandardHours             1470 non-null   int64 
 27  StockOptionLevel          1470 non-null   int64 
 28  TotalWorkingYears         1470 non-null   int64 
 29  TrainingTimesLastYear     1470 non-null   int64 
 30  WorkLifeBalance           1470 non-null   int64 
 31  YearsAtCompany            1470 non-null   int64 
 32  YearsInCurrentRole        1470 non-null   int64 
 33  YearsSinceLastPromotion   1470 non-null   int64 
 34  YearsWithCurrManager      1470 non-null   int64 
dtypes: int64(25), object(10)
memory usage: 402.1+ KB
```

In [4]:
```
pip install ydata-profiling
```

```
Requirement already satisfied: ydata-profiling in c:\users\hp\anaconda3\lib\site-pac
kages (4.12.2)
Requirement already satisfied: scipy<1.16,>=1.4.1 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (1.13.1)
Requirement already satisfied: pandas!=1.4.0,<3,>1.1 in c:\users\hp\anaconda3\lib\si
te-packages (from ydata-profiling) (2.2.2)
Requirement already satisfied: matplotlib>=3.5 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (3.9.2)
Requirement already satisfied: pydantic>=2 in c:\users\hp\anaconda3\lib\site-package
s (from ydata-profiling) (2.8.2)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (6.0.1)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in c:\users\hp\anaconda3\lib\site
-packages (from ydata-profiling) (3.1.4)
Requirement already satisfied: visions<0.8.0,>=0.7.5 in c:\users\hp\anaconda3\lib\si
te-packages (from visions[type_image_path]<0.8.0,>=0.7.5->ydata-profiling) (0.7.6)
Requirement already satisfied: numpy<2.2,>=1.16.0 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (1.26.4)
Requirement already satisfied: htmlmin==0.1.12 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (0.1.12)
Requirement already satisfied: phik<0.13,>=0.11.1 in c:\users\hp\anaconda3\lib\site-
packages (from ydata-profiling) (0.12.4)
Requirement already satisfied: requests<3,>=2.24.0 in c:\users\hp\anaconda3\lib\site
-packages (from ydata-profiling) (2.32.3)
Requirement already satisfied: tqdm<5,>=4.48.2 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (4.66.5)
Requirement already satisfied: seaborn<0.14,>=0.10.1 in c:\users\hp\anaconda3\lib\si
te-packages (from ydata-profiling) (0.13.2)
Requirement already satisfied: multimethod<2,>=1.4 in c:\users\hp\anaconda3\lib\site
-packages (from ydata-profiling) (1.12)
Requirement already satisfied: statsmodels<1,>=0.13.2 in c:\users\hp\anaconda3\lib\s
ite-packages (from ydata-profiling) (0.14.2)
Requirement already satisfied: typeguard<5,>=3 in c:\users\hp\anaconda3\lib\site-pac
kages (from ydata-profiling) (4.4.1)
Requirement already satisfied: imagehash==4.3.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from ydata-profiling) (4.3.1)
Requirement already satisfied: wordcloud>=1.9.3 in c:\users\hp\anaconda3\lib\site-pa
ckages (from ydata-profiling) (1.9.4)
Requirement already satisfied: dacite>=1.8 in c:\users\hp\anaconda3\lib\site-package
s (from ydata-profiling) (1.8.1)
Requirement already satisfied: PyWavelets in c:\users\hp\anaconda3\lib\site-packages
(from imagehash==4.3.1->ydata-profiling) (1.7.0)
Requirement already satisfied: pillow in c:\users\hp\anaconda3\lib\site-packages (fr
om imagehash==4.3.1->ydata-profiling) (10.4.0)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\hp\anaconda3\lib\site-pac
kages (from jinja2<3.2,>=2.11.1->ydata-profiling) (2.1.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from matplotlib>=3.5->ydata-profiling) (1.2.0)
Requirement already satisfied: cycler>=0.10 in c:\users\hp\anaconda3\lib\site-packag
es (from matplotlib>=3.5->ydata-profiling) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\hp\anaconda3\lib\site-p
ackages (from matplotlib>=3.5->ydata-profiling) (4.51.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\hp\anaconda3\lib\site-p
ackages (from matplotlib>=3.5->ydata-profiling) (1.4.4)
Requirement already satisfied: packaging>=20.0 in c:\users\hp\anaconda3\lib\site-pac
kages (from matplotlib>=3.5->ydata-profiling) (24.1)
```

```
Requirement already satisfied: pyparsing>=2.3.1 in c:\users\hp\anaconda3\lib\site-pa
ckages (from matplotlib>=3.5->ydata-profiling) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in c:\users\hp\anaconda3\lib\sit
e-packages (from matplotlib>=3.5->ydata-profiling) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\users\hp\anaconda3\lib\site-packag
es (from pandas!=1.4.0,<3,>1.1->ydata-profiling) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in c:\users\hp\anaconda3\lib\site-pack
ages (from pandas!=1.4.0,<3,>1.1->ydata-profiling) (2023.3)
Requirement already satisfied: joblib>=0.14.1 in c:\users\hp\anaconda3\lib\site-pack
ages (from phik<0.13,>=0.11.1->ydata-profiling) (1.4.2)
Requirement already satisfied: annotated-types>=0.4.0 in c:\users\hp\anaconda3\lib\s
ite-packages (from pydantic>=2->ydata-profiling) (0.6.0)
Requirement already satisfied: pydantic-core==2.20.1 in c:\users\hp\anaconda3\lib\si
te-packages (from pydantic>=2->ydata-profiling) (2.20.1)
Requirement already satisfied: typing-extensions>=4.6.1 in c:\users\hp\anaconda3\lib
\site-packages (from pydantic>=2->ydata-profiling) (4.11.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\hp\anaconda3\lib
\site-packages (from requests<3,>=2.24.0->ydata-profiling) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\hp\anaconda3\lib\site-packag
es (from requests<3,>=2.24.0->ydata-profiling) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\hp\anaconda3\lib\site-
packages (from requests<3,>=2.24.0->ydata-profiling) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\hp\anaconda3\lib\site-
packages (from requests<3,>=2.24.0->ydata-profiling) (2024.12.14)
Requirement already satisfied: patsy>=0.5.6 in c:\users\hp\anaconda3\lib\site-packag
es (from statsmodels<1,>=0.13.2->ydata-profiling) (0.5.6)
Requirement already satisfied: colorama in c:\users\hp\anaconda3\lib\site-packages
(from tqdm<5,>=4.48.2->ydata-profiling) (0.4.6)
Requirement already satisfied: attrs>=19.3.0 in c:\users\hp\anaconda3\lib\site-packa
ges (from visions<0.8.0,>=0.7.5->visions[type_image_path]<0.8.0,>=0.7.5->ydata-profi
ling) (23.1.0)
Requirement already satisfied: networkx>=2.4 in c:\users\hp\anaconda3\lib\site-packa
ges (from visions<0.8.0,>=0.7.5->visions[type_image_path]<0.8.0,>=0.7.5->ydata-profi
ling) (3.3)
Requirement already satisfied: six in c:\users\hp\anaconda3\lib\site-packages (from
patsy>=0.5.6->statsmodels<1,>=0.13.2->ydata-profiling) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

In [5]:
```
profile = ProfileReport(df, explorative=True)
profile.to_notebook_iframe()
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Pandas Profiling Report

Overview

Brought to you by YData

Overview | Alerts 25 | Reproduction

Dataset statistics

Number of variables	35
Number of observations	1470
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	1.1 MiB
Average record size in memory	770.8 B

Variable types

Numeric	15
Boolean	3
Categorical	17

Variables

Select Columns

In [6]:
```
df.shape
```

Out[6]:
```
(1470, 35)
```

In [7]:
```
# Handling outliers in numerical columns using z-scores method

# upon examining our dataset, we noticed that all numerical columns indicate no out
```

In [8]:
```
category = df.select_dtypes(include = ['object', 'category']).columns

print(category)
```

```
Index(['Attrition', 'BusinessTravel', 'Department', 'EducationField', 'Gender',
       ' JobInvolvement ', 'JobRole', 'MaritalStatus', 'Over18', 'OverTime'],
      dtype='object')
```

In [9]:
```
encoder = LabelEncoder()
df['Attrition'] = encoder.fit_transform(df['Attrition'])


categorical_cols = ['BusinessTravel', 'Department', 'EducationField', 'Gender', ' J

for cols in categorical_cols:
    df[cols] = encoder.fit_transform(df[cols])
```

In [10]:
```
X = df.drop(columns = ['Attrition', 'StandardHours', 'Over18', 'EmployeeNumber', 'E
```

In [11]:
```
y=df['Attrition']
```

In [12]:
```
y.head()
```

Out[12]:
```
0    1
1    0
2    1
3    0
4    0
Name: Attrition, dtype: int32
```

In [13]:
```
#  (X = features, y = target variable)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
```

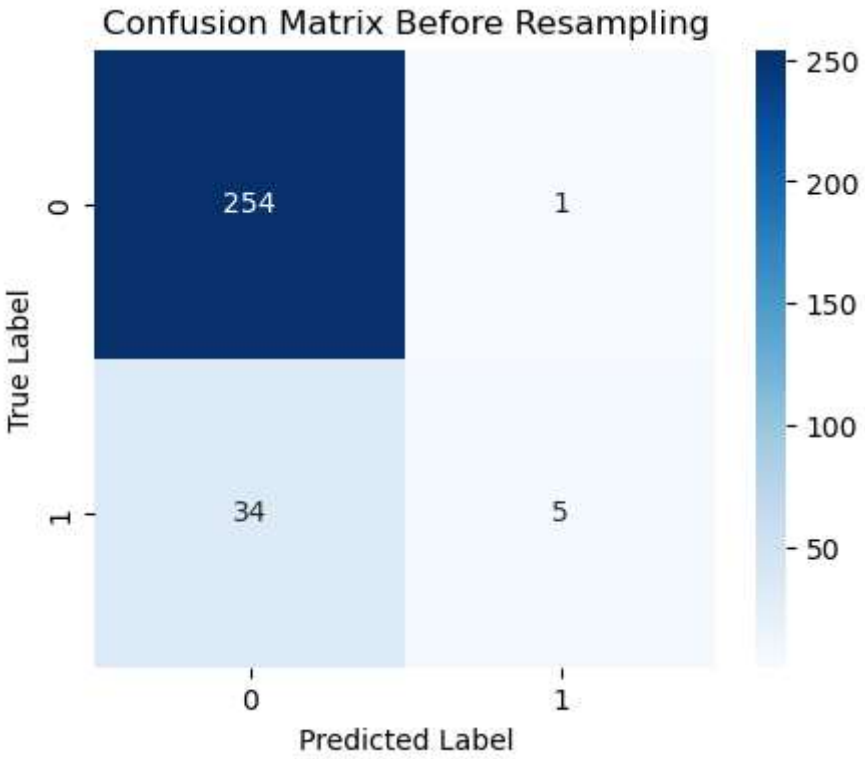
In [14]:
```
model = RandomForestClassifier(max_depth = 15, min_samples_split = 15, criterion =

model.fit(X_train, y_train)
```

Out[14]:
```
RandomForestClassifier(max_depth=15, min_samples_split=15, random_state=4
2)
```

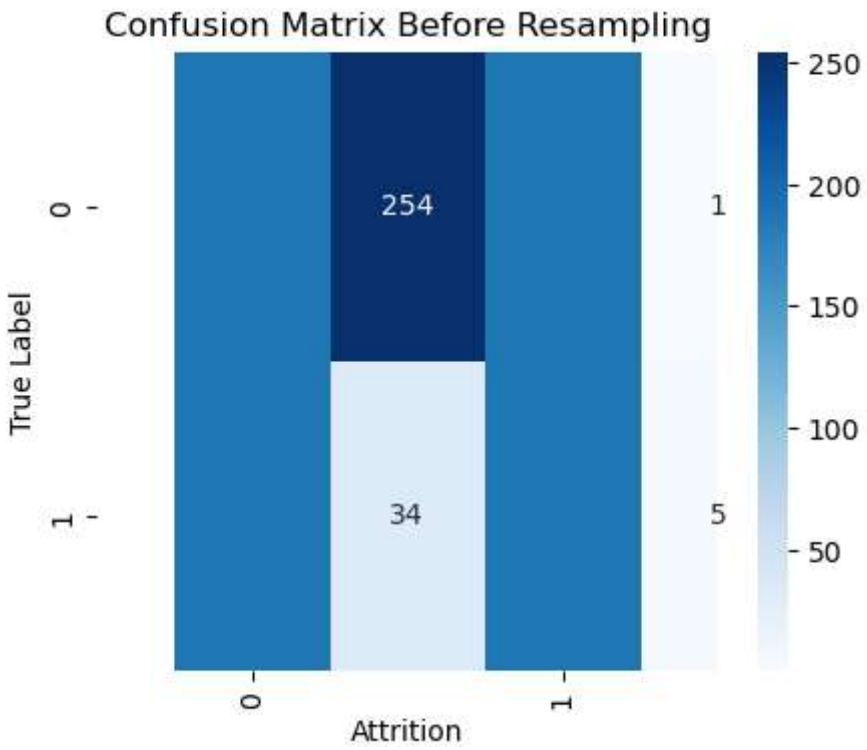
In [15]:
```
y_pred = model.predict(X_test)
```

In [37]:
```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Compute confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Visualize it
plt.figure(figsize=(5, 4))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix Before Resampling")
plt.show()
```



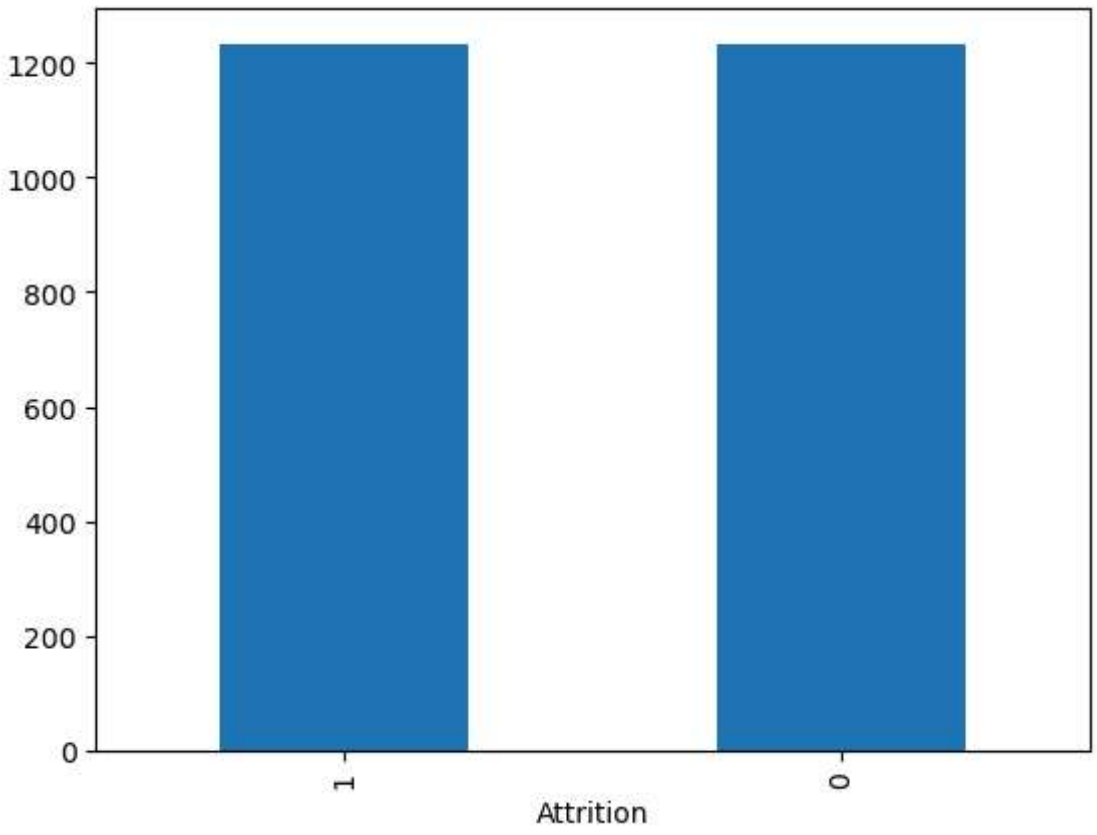
In [17]:
```
%matplotlib inline
import matplotlib.pyplot as plt

y.value_counts().plot(kind='bar')
plt.show()
```



In [18]:
```
# Apply SMOTE to balance the dataset
smote = SMOTE(random_state=42)


X__resampled, y_resampled = smote.fit_resample(X, y)


print("Before SMOTE:", y.value_counts())  # Class distribution before SMOTE
print("After SMOTE:", y_resampled.value_counts())  # Class distribution after S
```

```
Before SMOTE: Attrition
0    1233
1     237
Name: count, dtype: int64
After SMOTE: Attrition
1    1233
0    1233
Name: count, dtype: int64
```

In [19]:
```
y_resampled.value_counts().plot(kind='bar')
plt.show()
```



In [20]:
```
X_train, X_test, y_train, y_test = train_test_split(X__resampled, y_resampled, test
```

In [21]:
```
model.fit(X_train, y_train)
```

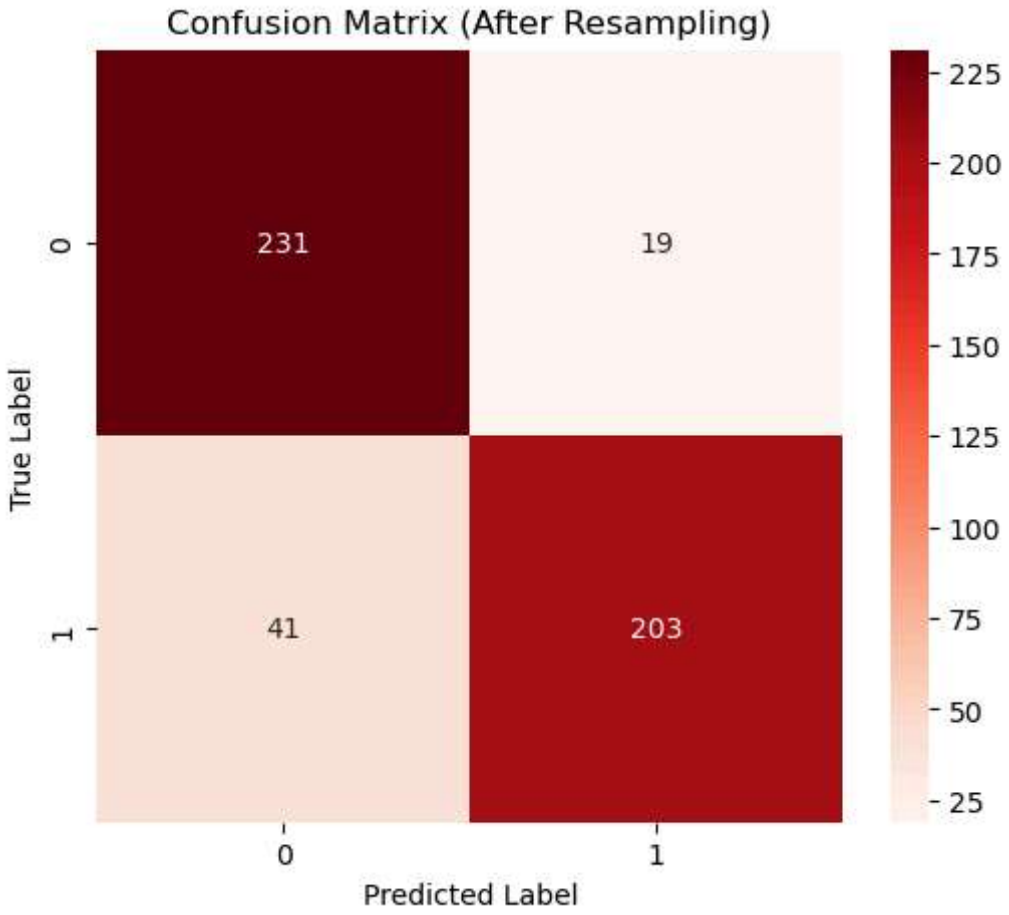
Out[21]:

RandomForestClassifier

```
RandomForestClassifier(max_depth=15, min_samples_split=15, random_state=4
2)
```

In [22]:
```
y_pred_resampled = model.predict(X_test)
```

In [23]:
```
# Compute the confusion matrix
cm_resampled = confusion_matrix(y_test, y_pred_resampled)

# Plot the confusion matrix
plt.figure(figsize=(6, 5))
sns.heatmap(cm_resampled, annot=True, fmt="d", cmap="Reds", xticklabels=[0, 1], yti
plt.title('Confusion Matrix (After Resampling)')
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()
```



In [24]:
```
print(f"y_pred shape: {y_pred.shape}")
```

```
y_pred shape: (294,)
```

In [25]:
```
# Resampling changes only the training data (e.g., X_train_resampled, y_train_resam
# Some models, like Decision Trees, can still favor the majority class even after r
```

In [26]:
```
print(y_test.shape, y_pred.shape)
```

```
(494,) (294,)
```

In [27]:
```
# Classification Accuracy

from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

# Cross-validation accuracy
scores = cross_val_score(model, X, y, cv=5, scoring="accuracy")
print(f'Cross-Validation Accuracy: {scores.mean():.2f}')


accuracy = accuracy_score(y_test, y_pred_resampled)
precision = precision_score(y_test, y_pred_resampled)
recall = recall_score(y_test, y_pred_resampled)
f1 = f1_score(y_test, y_pred_resampled)

# Compute ROC-AUC using predicted probabilities
y_pred_probs = model.predict_proba(X_test)[:, 1]  # Get probabilities for class 1
```

```
roc_auc = roc_auc_score(y_test, y_pred_probs)

# Print results
print(f'Classification Accuracy: {accuracy:.2f}')
print(f'Precision: {precision:.2f}')
print(f'Recall: {recall:.2f}')
print(f'F1 Score: {f1:.2f}')
print(f'ROC_AUC: {roc_auc:.2f}')
```

```
Cross-Validation Accuracy: 0.86
Classification Accuracy: 0.88
Precision: 0.91
Recall: 0.83
F1 Score: 0.87
ROC_AUC: 0.95
```

In [28]:
```
print("Unique values in y_resampled:", set(y_resampled))
```

```
Unique values in y_resampled: {0, 1}
```

In [29]:
```
print("Type of y_test:", type(y_test))
print("Type of y_pred:", type(y_pred))
```

```
Type of y_test: <class 'pandas.core.series.Series'>
Type of y_pred: <class 'numpy.ndarray'>
```

In [30]:
```
train_accuracy = model.score(X_train, y_train)

print(f"Train Accuracy: {train_accuracy:.2%}")
```

```
Train Accuracy: 97.77%
```

In [31]:
```
print(f'Test Accuracy: {accuracy:.2%}')
```

```
Test Accuracy: 87.85%
```

In [32]:
```
print(f"X_test shape: {X_test}")
print(f"y_test shape: {y_test}")
```

```
X_test shape:       Age  BusinessTravel  DailyRate  Department  DistanceFromHome  Ed
ucation  \
1078   44               2         136           1                28          3
1281   35               2         303           2                27          3
621    36               2         928           2                 1          2
1508   29               1         336           1                15          1
1975   43               0         741           0                27          3
...   ...             ...         ...         ...               ...        ...
2099   47               2        1073           1                 7          3
1836   29               0         477           1                23          2
303    31               2         218           2                 7          3
1864   25               1        1123           1                 3          3
141    45               2        1316           1                29          3

      EducationField  EnvironmentSatisfaction  Gender  HourlyRate  ...  \
1078               1                        4       1          32  ...
1281               1                        3       1          84  ...
621                1                        2       1          56  ...
1508               3                        3       0          81  ...
1975               0                        1       0          77  ...
...              ...                      ...     ...         ...  ...
2099               4                        2       0          62  ...
1836               1                        4       1          73  ...
303                5                        2       1         100  ...
1864               3                        3       0          80  ...
141                3                        3       1          83  ...

      PerformanceRating  RelationshipSatisfaction  StockOptionLevel  \
1078                  3                         3                 1
1281                  3                         4                 0
621                   3                         4                 1
1508                  3                         1                 0
1975                  3                         3                 1
...                 ...                       ...               ...
2099                  3                         4                 0
1836                  4                         2                 0
303                   3                         2                 1
1864                  3                         3                 0
141                   3                         2                 0

      TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  \
1078                 24                      1                4
1281                 10                      2                3
621                  18                      1                2
1508                  9                      1                2
1975                 10                      2                2
...                 ...                    ...              ...
2099                 23                      0                3
1836                  1                      2                2
303                  10                      3                2
1864                  4                      4                3
141                   9                      2                2

      YearsAtCompany  YearsInCurrentRole  YearsSinceLastPromotion  \
1078              20                   6                       14
1281              10                   7                        7
```

```
621              18                  14                          4
1508              7                   6                          6
1975              8                   5                          6
...             ...                 ...                        ...
2099             14                   9                          8
1836              1                   0                          1
303               8                   7                          7
1864              3                   1                          0
141               6                   5                          0

      YearsWithCurrManager
1078                    17
1281                     7
621                     11
1508                     6
1975                     7
...                    ...
2099                     6
1836                     0
303                      7
1864                     2
141                      3

[494 rows x 30 columns]
y_test shape: 1078    0
1281    1
621     0
1508    1
1975    1
       ..
2099    1
1836    1
303     0
1864    1
141     0
Name: Attrition, Length: 494, dtype: int32
```

HYPERPARAMETER TUNING

In [34]:

```
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

# Define the model
model = RandomForestClassifier(random_state=42)

# Define the hyperparameter grid
param_grid = {
    'n_estimators': [50, 100, 200],          # Number of trees in the forest
    'max_depth': [None, 10, 20, 30],         # Depth of each tree
    'min_samples_split': [2, 5, 10],         # Minimum samples required to split a
    'min_samples_leaf': [1, 2, 4],           # Minimum samples required at a leaf n
    'max_features': ['sqrt', 'log2'],        # Number of features to consider at ea
    'class_weight': ['balanced', None]       # Handle class imbalance
```

```
}

# Initialize GridSearchCV (Exhaustive Search)
random_search = RandomizedSearchCV(model, param_grid, n_iter=50, cv=5, scoring='f1'
# Fit on training data
random_search.fit(X_resampled, y_resampled)


# Evaluate the best model
best_model = random_search.best_estimator_
y_pred = best_model.predict(X_test)

# Print classification report
print(classification_report(y_test, y_pred))
```

```
Fitting 5 folds for each of 50 candidates, totalling 250 fits
              precision    recall  f1-score   support

           0       0.99      0.98      0.99       250
           1       0.98      0.99      0.99       244

    accuracy                           0.99       494
   macro avg       0.99      0.99      0.99       494
weighted avg       0.99      0.99      0.99       494
```

In [35]:
```
print("Best Hyperparameters:", random_search.best_params_)
print("Best Score:", random_search.best_score_)
```

```
Best Hyperparameters: {'n_estimators': 200, 'min_samples_split': 2, 'min_samples_lea
f': 1, 'max_features': 'log2', 'max_depth': 10, 'class_weight': None}
Best Score: 0.8892839127223503
```

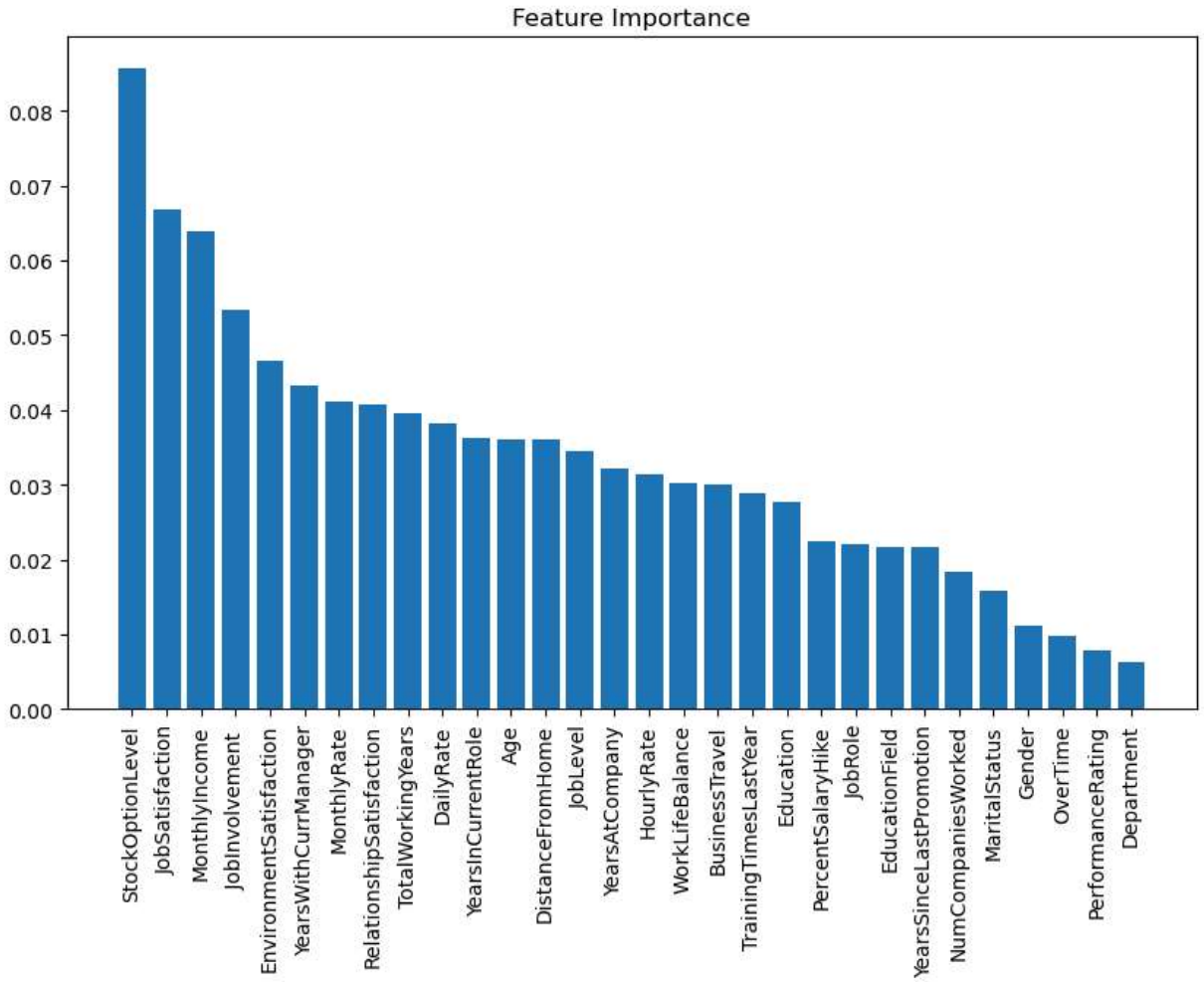
In [36]:
```
best_model = random_search.best_estimator_
y_pred = best_model.predict(X_test)
```

In [37]:
```
from sklearn.metrics import classification_report, confusion_matrix

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Confusion Matrix:
 [[245   5]
 [  2 242]]
Classification Report:
               precision    recall  f1-score   support

           0       0.99      0.98      0.99       250
           1       0.98      0.99      0.99       244

    accuracy                           0.99       494
   macro avg       0.99      0.99      0.99       494
weighted avg       0.99      0.99      0.99       494
```

In [38]:
```
import matplotlib.pyplot as plt
import numpy as np
```

```
feature_importances = best_model.feature_importances_
sorted_indices = np.argsort(feature_importances)[::-1]  # Sort in descending order

plt.figure(figsize=(10, 6))
plt.bar(range(len(feature_importances)), feature_importances[sorted_indices])
plt.xticks(range(len(feature_importances)), X_train.columns[sorted_indices], rotati
plt.title("Feature Importance")
plt.show()
```



In [39]:
```
print("Train Accuracy:", best_model.score(X_train, y_train))
print("Test Accuracy:", best_model.score(X_test, y_test))
```

```
Train Accuracy: 0.9939148073022313
Test Accuracy: 0.9858299595141701
```

In [40]:
```
pd.set_option('display.max_columns', None)
print(X.head())
```

	Age	BusinessTravel	DailyRate	Department	DistanceFromHome	Education
0	41	2	1102	2	1	2
1	49	1	279	1	8	1
2	37	2	1373	1	2	2
3	33	1	1392	1	3	4
4	27	2	591	1	2	1

	EducationField	EnvironmentSatisfaction	Gender	HourlyRate
0	1	2	0	94
1	1	3	1	61
2	4	4	1	92
3	1	4	0	56
4	3	1	1	40

	JobInvolvement	JobLevel	JobRole	JobSatisfaction	MaritalStatus
0	2	2	7	4	2
1	1	2	6	2	1
2	1	1	2	3	2
3	2	1	6	3	1
4	2	1	2	2	1

	MonthlyIncome	MonthlyRate	NumCompaniesWorked	OverTime
0	5993	19479	8	1
1	5130	24907	1	0
2	2090	2396	6	1
3	2909	23159	1	1
4	3468	16632	9	0

	PercentSalaryHike	PerformanceRating	RelationshipSatisfaction
0	11	3	1
1	23	4	4
2	15	3	2
3	11	3	3
4	12	3	4

	StockOptionLevel	TotalWorkingYears	TrainingTimesLastYear
0	0	8	0
1	1	10	3
2	0	7	3
3	0	8	3
4	1	6	3

	WorkLifeBalance	YearsAtCompany	YearsInCurrentRole
0	1	6	4
1	3	10	7
2	3	0	0
3	3	8	7
4	3	2	2

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

In [95]:
```
new_data = [[36, 2, 1973, 1, 2, 1, 4, 4, 1, 95, 1, 2, 3, 3, 3, 2190, 2596, 6, 4, 15
predictions = best_model.predict(new_data)
print("🔮 Predictions:", predictions)
```

```
🔮 Predictions: [0]
```

In [ ]:
```

```