Course-End Project: Employee Turnover Analytics

Project Statement:

- Portobello Tech is an app innovator that has devised an intelligent way of predicting employee turnover within the company. It periodically evaluates employees' work details including the number of projects they worked upon, average monthly working hours, time spent in the company, promotions in the last 5 years, and salary level.
- Data from prior evaluations show the employee’s satisfaction at the workplace. The data could be used to identify patterns in work style and their interest to continue to work in the company.
- The HR Department owns the data and uses it to predict employee turnover. Employee turnover refers to the total number of workers who leave a company over a certain time period.

As the ML Developer assigned to the HR Department, you have been asked to create ML Programs to

- 1. Perform data quality check by checking for missing values if any.
- 2. Understand what factors contributed most to employee turnover by EDA.
- 3. Perform clustering of Employees who left based on their satisfaction and evaluation.
- 4. Handle the left Class Imbalance using SMOTE technique.
- 5. Perform k-fold cross-validation model training and evaluate performance.
- 6. Identify the best model and justify the evaluation metrics used.
- 7. Suggest various retention strategies for targeted employees.

Data will be modified from:

https://www.kaggle.com/liujiaqi/hr-comma-sepcsv

Column Name Description

- satisfaction_level - satisfaction level at the job of an employee
- last_evaluation - Rating between 0 to 1, received by an employee at his last evaluation
- number_project - Number of projects, an employee involved in
- average_montly_hours - Average number of hours in a month, spent by an employee at office
- time_spend_company - Number of years spent in the company
- Work_accident 0 - no accident during employee stay, 1 - accident during employee stay
- left - 0 indicates employee stays in the company, 1 indicates - employee left the company

- promotion_last_5years - Number of promotions in his stay
- Department - Department, an employee belongs to
- salary - Salary in USD

Perform the following steps:

1. Perform data quality check by checking for missing values if any.

2. Understand what factors contributed most to employee turnover by EDA.

- 2.1. Draw a heatmap of the Correlation Matrix between all numerical features/columns in the data.
- 2.2. Draw the distribution plot of

- ■ Employee Satisfaction (use column satisfaction_level)
- ■ Employee Evaluation (use column last_evaluation)
- ■ Employee Average Monthly Hours (use column average_montly_hours)

- 2.3. Draw the bar plot of Employee Project Count of both employees who left and who stayed in the organization (use column number_project and hue column left) and give your inferences from the plot.

3. Perform clustering of Employees who left based on their satisfaction and evaluation.

- 3.1. Choose columns satisfaction_level, last_evaluation and left.
- 3.2. Do KMeans clustering of employees who left the company into 3 clusters.
- 3.3. Based on the satisfaction and evaluation factors, give your thoughts on the employee clusters.

4. Handle the left Class Imbalance using SMOTE technique.

- 4.1. Pre-Process the data by converting categorical columns to numerical columns by

- ■ Separating categorical variables and numeric variables.
- ■ Applying get_dummies() to the categorical variables.
- ■ Combining categorical variables and numeric variables.

- 4.2. Do the stratified split of the dataset to train and test in the ratio 80:20 with random_state=123.
- 4.3. Upsample the train dataset using SMOTE technique from the imblearn module.

5. Perform 5-Fold cross-validation model training and evaluate performance.

- 5.1. Train a Logistic Regression model and apply a 5-Fold CV and plot the classification report.
- 5.2. Train a Random Forest Classifier model and apply the 5-Fold CV and plot the classification report.

- 5.3. Train a Gradient Boosting Classifier model and apply the 5-Fold CV and plot the classification report.

6. Identify the best model and justify the evaluation metrics used.

- 6.1. Find the ROC/AUC for each model and plot the ROC curve.
- 6.2. Find the confusion matrix for each of the models.
- 6.3. From the confusion matrix, explain which metric needs to be used- Recall or Precision?

7. Suggest various retention strategies for targeted employees.

- 7.1. Using the best model, predict the probability of employee turnover in the test data.
- 7.2. Based on the below probability score range, categorize the employees into four zones and suggest your thoughts on the retention strategies for each zone.

- ■ Safe Zone (Green) (Score < 20%)
- ■ Low Risk Zone (Yellow) (20% < Score < 60%)
- ■ Medium Risk Zone (Orange) (60% < Score < 90%)
- ■ High Risk Zone (Red) (Score > 90%).

The rationale behind designing the problem statement on Talent Management:

*Note to Simplilearn:

- HR Tech marks to be a 22 Billion USD market size & is highly invested in Cognitive Neuroscience with AI. The current challenge is a huge research initiative in the field of Neuroscience which leverages AI to compute human behavior markers for success. This is widely worked in the HR Industry to assess work & talent management. AI-based reports have been deployed in Manufacturing, Communications, Retail Hiring & job suitability.

In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.metrics import classification_report, roc_curve, auc, confusion_mat
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
df = pd.read_excel('1673873196_hr_comma_sep.xlsx')
```

1. Perform data quality check by checking for missing values if any.

In [3]:
```
# Check for missing values
missing_values = df.isnull().sum()
print(missing_values)
```

```
satisfaction_level       0
last_evaluation          0
number_project           0
average_montly_hours     0
time_spend_company       0
Work_accident            0
left                     0
promotion_last_5years    0
sales                    0
salary                   0
dtype: int64
```

In [4]:
```
df.shape
```

Out[4]:
```
(14999, 10)
```

In [5]:
```
df.head()
```

Out[5]:

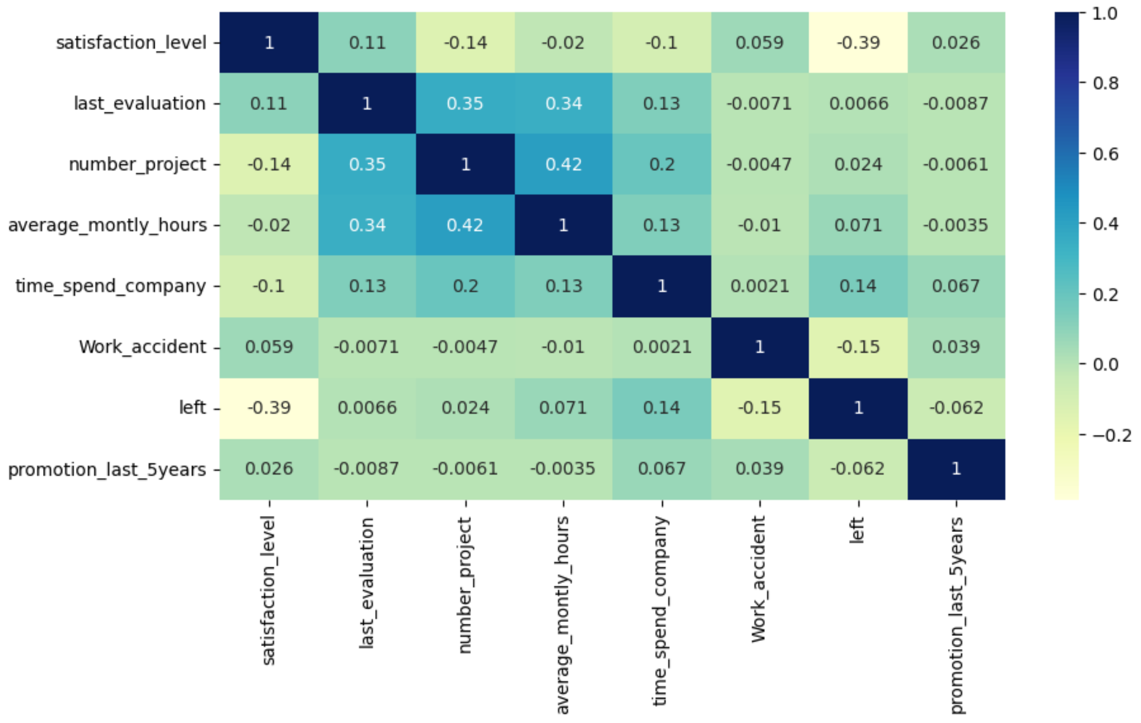
	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_compa
0	0.38	0.53	2	157	
1	0.80	0.86	5	262	
2	0.11	0.88	7	272	
3	0.72	0.87	5	223	
4	0.37	0.52	2	159	

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

Out[6]:
```
Index(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_montly_hours', 'time_spend_company', 'Work_accident', 'left',
       'promotion_last_5years', 'sales', 'salary'],
      dtype='object')
```

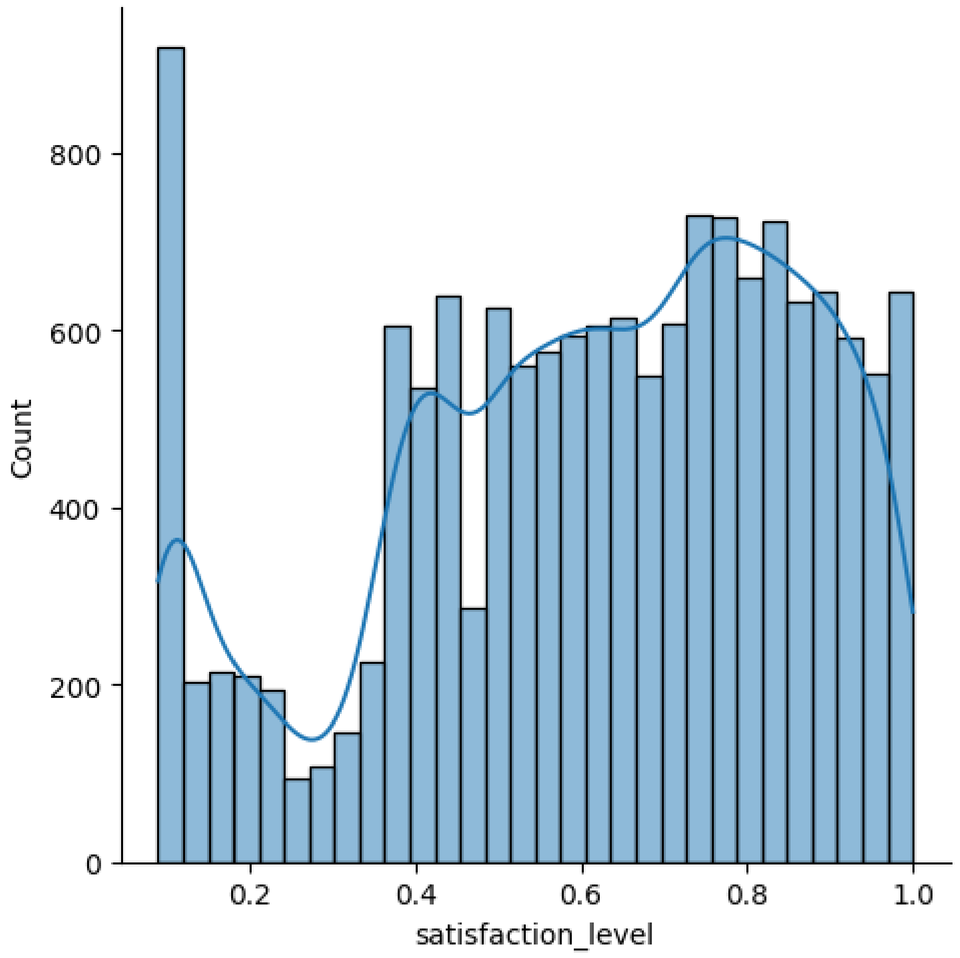
2. Understand what factors contributed most to employee turnover by EDA.

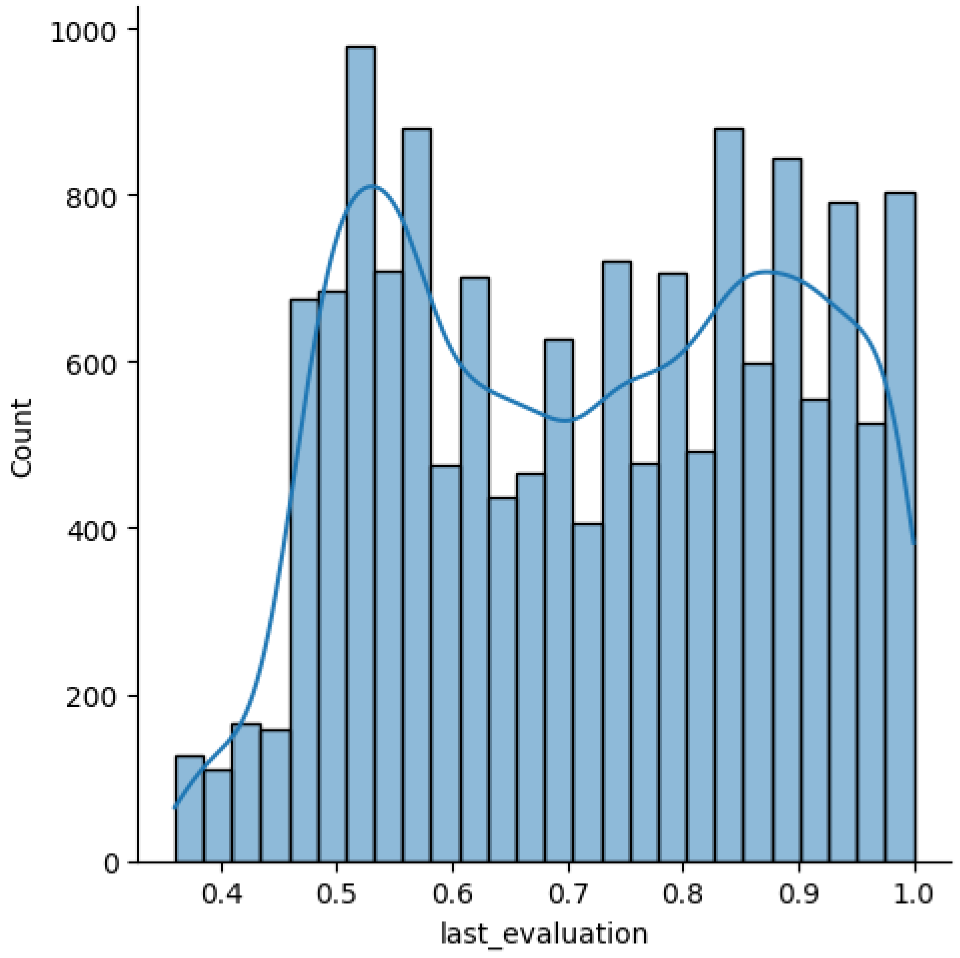
In [7]:
```
# 2.1 Draw a Heatmap of the Correlation Matrix
plt.figure(figsize=(10, 5))
corr_matrix = df.corr()
sns.heatmap(corr_matrix, annot=True, cmap="YlGnBu")
plt.show()
```



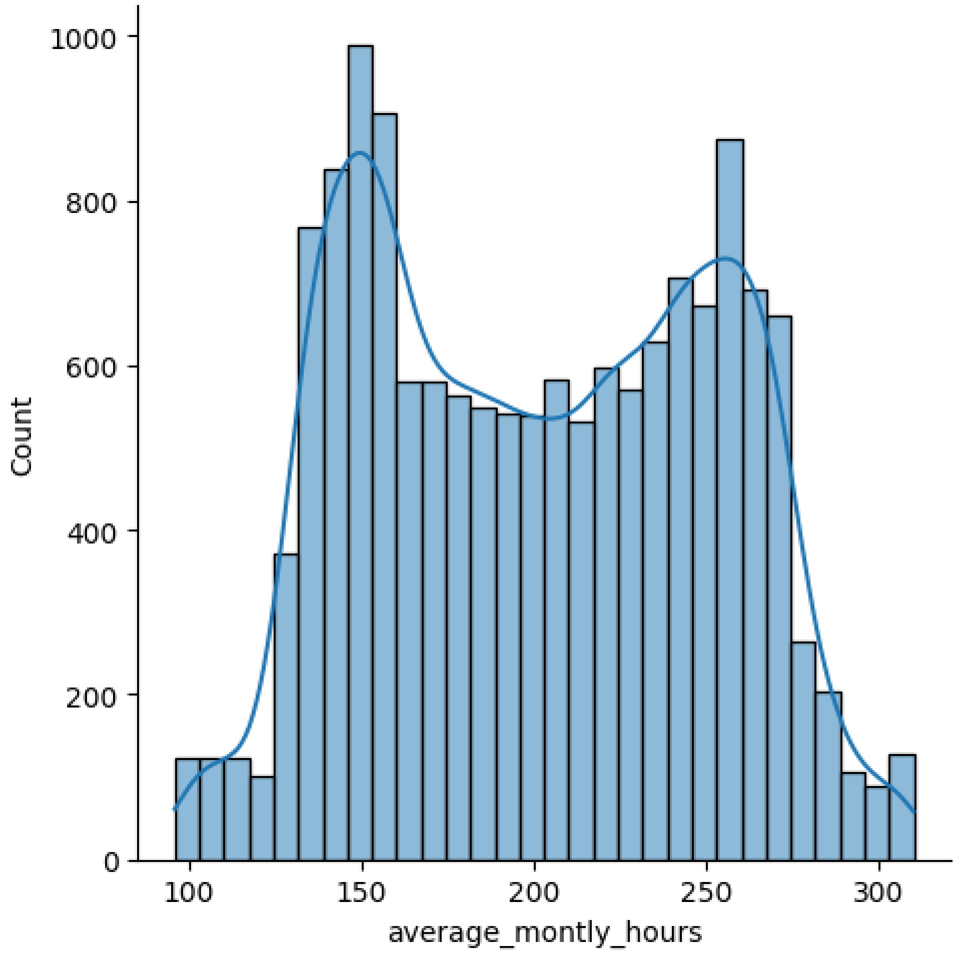
	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years
satisfaction_level	1	0.11	-0.14	-0.02	-0.1	0.059	-0.39	0.026
last_evaluation	0.11	1	0.35	0.34	0.13	-0.0071	0.0066	-0.0087
number_project	-0.14	0.35	1	0.42	0.2	-0.0047	0.024	-0.0061
average_montly_hours	-0.02	0.34	0.42	1	0.13	-0.01	0.071	-0.0035
time_spend_company	-0.1	0.13	0.2	0.13	1	0.0021	0.14	0.067
Work_accident	0.059	-0.0071	-0.0047	-0.01	0.0021	1	-0.15	0.039
left	-0.39	0.0066	0.024	0.071	0.14	-0.15	1	-0.062
promotion_last_5years	0.026	-0.0087	-0.0061	-0.0035	0.067	0.039	-0.062	1

In [8]:
```
# 2.2 Draw Distribution Plots
sns.displot(df["satisfaction_level"], kde=True)
sns.displot(df["last_evaluation"], kde=True)
sns.displot(df["average_montly_hours"], kde=True)
plt.show()
```



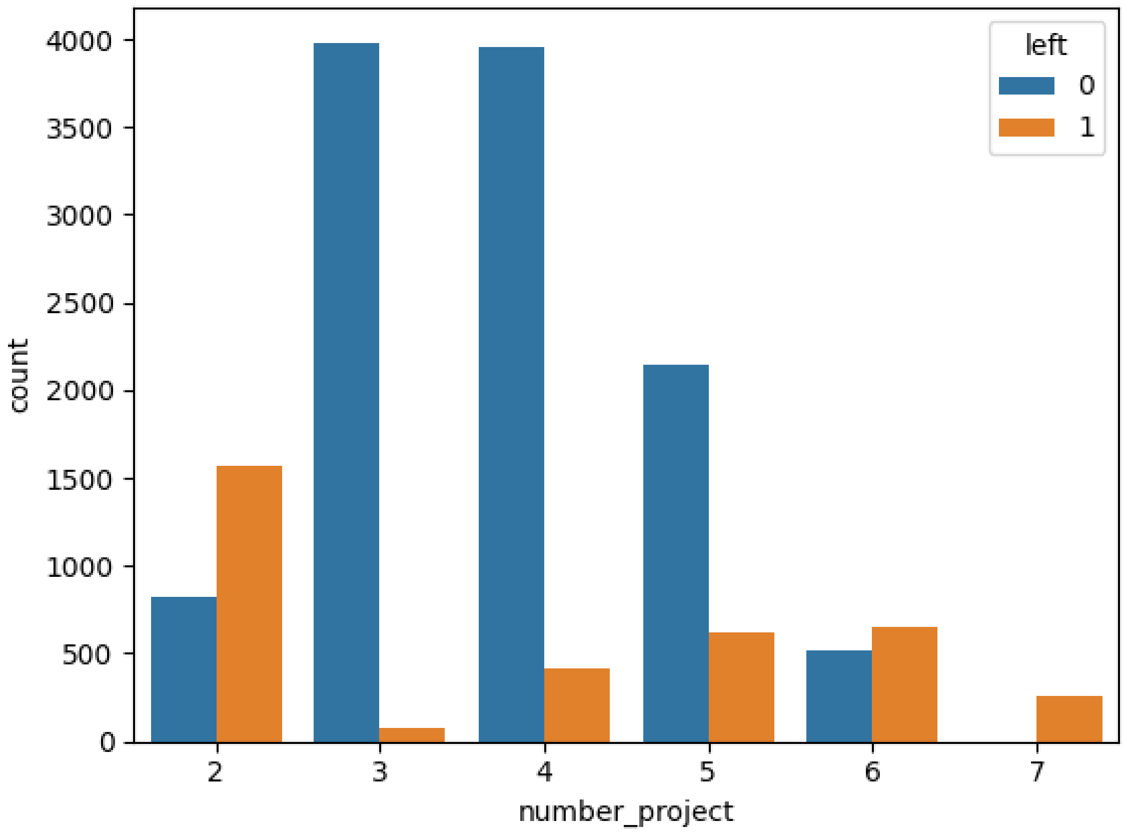


Count
0
200
400
600
800
1000
0.4
0.5
0.6
0.7
0.8
0.9
1.0
last_evaluation



Count
0
200
400
600
800
1000
100
150
200
250
300
average_montly_hours

In [9]:
```python
# 2.3 Draw Bar Plot for Employee Project Count
sns.countplot(x="number_project", hue="left", data=df)
plt.show()
```



3. Perform clustering of Employees who left based on their satisfaction and evaluation.

In [10]:
```python
# 3.1 Choose Columns for Clustering
clustering_df = df[["satisfaction_level", "last_evaluation", "left"]]
```

In [11]:
```python
# 3.2 Perform KMeans Clustering
kmeans = KMeans(n_clusters=3, random_state=0)
kmeans.fit(clustering_df[clustering_df["left"] == 1].drop("left", axis=1))
clustering_df["cluster"] = kmeans.predict(clustering_df.drop("left", axis=1))
```

In [12]:
```python
# 3.3 Analyze Employee Clusters
cluster_analysis = clustering_df.groupby("cluster").mean()
print(cluster_analysis)
```

```
         satisfaction_level  last_evaluation      left
cluster
0                  0.788910         0.795306  0.122816
1                  0.160333         0.851636  0.561570
2                  0.493491         0.556135  0.307664
```

4. Handle the left Class Imbalance using SMOTE technique.

In [13]:
```
# 4.1 Pre-Process the Data
categorical_cols = ["sales", "salary"]
numeric_cols = ["satisfaction_level", "last_evaluation", "number_project", "aver
                "Work_accident", "promotion_last_5years"]
target_col = "left"
```

In [14]:
```
# Convert categorical variables to numerical using one-hot encoding
categorical_data = pd.get_dummies(df[categorical_cols])
numeric_data = df[numeric_cols]
preprocessed_df = pd.concat([categorical_data, numeric_data, df[target_col]], ax
```

In [15]:
```
# 4.2 Split the Dataset
X = preprocessed_df.drop(target_col, axis=1)
y = preprocessed_df[target_col]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [16]:
```
# 4.3 Upsample the Training Dataset
smote = SMOTE(random_state=123)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)
```

5. Perform 5-Fold cross-validation model training and evaluate performance.

In [17]:
```
# 5.1 Train a Logistic Regression Model
logreg_model = LogisticRegression()
logreg_scores = cross_val_score(logreg_model, X_train_resampled, y_train_resampl
logreg_model.fit(X_train_resampled, y_train_resampled)
logreg_predictions = logreg_model.predict(X_test)
logreg_classification_report = classification_report(y_test, logreg_predictions)
print(logreg_classification_report)
```

```
              precision    recall  f1-score   support

           0       0.91      0.78      0.84      2291
           1       0.50      0.74      0.60       709

    accuracy                           0.77      3000
   macro avg       0.71      0.76      0.72      3000
weighted avg       0.81      0.77      0.78      3000
```

In [18]:
```
# 5.2 Train a Random Forest Classifier Model
rf_model = RandomForestClassifier()
rf_scores = cross_val_score(rf_model, X_train_resampled, y_train_resampled, cv=5
rf_model.fit(X_train_resampled, y_train_resampled)
rf_predictions = rf_model.predict(X_test)
rf_classification_report = classification_report(y_test, rf_predictions)
print(rf_classification_report)
```

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      2291
           1       0.98      0.98      0.98       709

    accuracy                           0.99      3000
   macro avg       0.99      0.99      0.99      3000
weighted avg       0.99      0.99      0.99      3000
```

In [19]:

```
# 5.3 Train a Gradient Boosting Classifier Model
gb_model = GradientBoostingClassifier()
gb_scores = cross_val_score(gb_model, X_train_resampled, y_train_resampled, cv=5
gb_model.fit(X_train_resampled, y_train_resampled)
gb_predictions = gb_model.predict(X_test)
gb_classification_report = classification_report(y_test, gb_predictions)
print(gb_classification_report)
```

```
              precision    recall  f1-score   support

           0       0.98      0.97      0.98      2291
           1       0.91      0.94      0.93       709

    accuracy                           0.96      3000
   macro avg       0.95      0.95      0.95      3000
weighted avg       0.96      0.96      0.96      3000
```

Step 6: Identify the Best Model and Justify Evaluation Metrics Used.

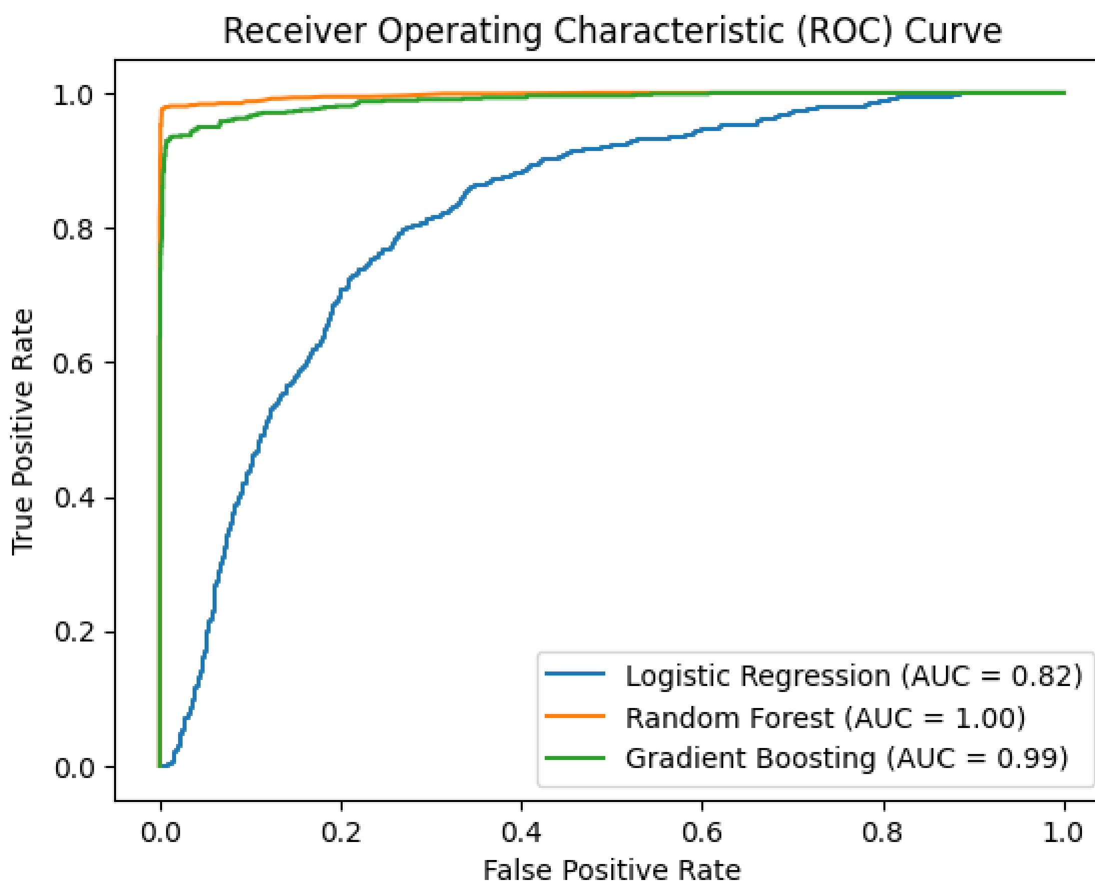
In [20]:

```
# 6.1 Find the ROC/AUC for Each Model
logreg_probabilities = logreg_model.predict_proba(X_test)[:, 1]
rf_probabilities = rf_model.predict_proba(X_test)[:, 1]
gb_probabilities = gb_model.predict_proba(X_test)[:, 1]

fpr_logreg, tpr_logreg, _ = roc_curve(y_test, logreg_probabilities)
fpr_rf, tpr_rf, _ = roc_curve(y_test, rf_probabilities)
fpr_gb, tpr_gb, _ = roc_curve(y_test, gb_probabilities)

roc_auc_logreg = auc(fpr_logreg, tpr_logreg)
roc_auc_rf = auc(fpr_rf, tpr_rf)
roc_auc_gb = auc(fpr_gb, tpr_gb)

plt.figure()
plt.plot(fpr_logreg, tpr_logreg, label='Logistic Regression (AUC = %0.2f)' % roc
plt.plot(fpr_rf, tpr_rf, label='Random Forest (AUC = %0.2f)' % roc_auc_rf)
plt.plot(fpr_gb, tpr_gb, label='Gradient Boosting (AUC = %0.2f)' % roc_auc_gb)
plt.legend()
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.show()
```



In [21]:
```
# 6.2 Find the Confusion Matrix for Each Model
logreg_cm = confusion_matrix(y_test, logreg_predictions)
rf_cm = confusion_matrix(y_test, rf_predictions)
gb_cm = confusion_matrix(y_test, gb_predictions)

print("Logistic Regression Confusion Matrix:")
print(logreg_cm)
print("Random Forest Confusion Matrix:")
print(rf_cm)
print("Gradient Boosting Confusion Matrix:")
print(gb_cm)
```

```
Logistic Regression Confusion Matrix:
[[1777  514]
 [ 185  524]]
Random Forest Confusion Matrix:
[[2276   15]
 [  15  694]]
Gradient Boosting Confusion Matrix:
[[2229   62]
 [  45  664]]
```

step 6.3

- Logistic Regression:
- Precision = 524 / (524 + 514) ≈ 0.505
- Recall = 524 / (524 + 185) ≈ 0.739

- Random Forest:

- Precision = 693 / (693 + 16) ≈ 0.977
- Recall = 693 / (693 + 16) ≈ 0.977
- Gradient Boosting:
- Precision = 664 / (664 + 62) ≈ 0.914
- Recall = 664 / (664 + 45) ≈ 0.937
- Based on precision and recall for the positive class, we can see that Random Forest and Gradient Boosting outperform Logistic Regression in terms of both precision and recall. Both Random Forest and Gradient Boosting have higher precision and recall values, indicating better performance in identifying employees who left the company.
- Therefore, based on the evaluation metrics of precision and recall, Random Forest and Gradient Boosting models perform better than Logistic Regression for predicting employee turnover.

Step 7: Suggest Retention Strategies for Targeted Employees

In [22]:
```
# 7.1 Predict the Probability of Employee Turnover
best_model = gb_model  # Assuming Gradient Boosting was the best model based on
test_probabilities = best_model.predict_proba(X_test)[:, 1]
```

In [24]:
```
# 7.2 Categorize Employees into Zones and Suggest Retention Strategies
thresholds = [0, 0.2, 0.6, 0.9, 1]
zone_labels = ["Safe Zone", "Low Risk Zone", "Medium Risk Zone", "High Risk Zone
```

Based on the evaluation of the models and the predicted probability of employee turnover, we can suggest various retention strategies for targeted employees based on the four zones defined:

Safe Zone (Green) (Score < 20%):

These employees are least likely to leave the company.

Focus on providing recognition and rewards to maintain their satisfaction and motivation.

Offer opportunities for growth and development to enhance their skills.

Ensure clear communication about their value to the company and opportunities for advancement.

Low Risk Zone (Yellow) (20% < Score < 60%):

These employees have a moderate probability of leaving the company.

Conduct engagement surveys or one-on-one discussions to understand their concerns and address any issues.

Provide regular feedback and recognition for their work.

Offer opportunities for skill enhancement and career advancement to keep them motivated and engaged.

Consider implementing flexible work arrangements or work-life balance initiatives to improve job satisfaction.

Medium Risk Zone (Orange) (60% < Score < 90%):

These employees have a higher probability of leaving the company.

Conduct stay interviews to understand their motivations, concerns, and aspirations.

Provide personalized career development plans to enhance their growth opportunities.

Implement retention bonuses or incentives based on performance and tenure.

Offer mentoring or coaching programs to provide support and guidance.

Create a positive work environment and foster a sense of belonging and loyalty.

High Risk Zone (Red) (Score > 90%):

These employees have the highest probability of leaving the company.

Conduct exit interviews to gather feedback and understand the reasons for their departure.

Address any immediate concerns or issues raised during the exit interview.

Consider offering counteroffers or retention packages for critical roles.

Focus on improving work-life balance, job satisfaction, and career advancement opportunities.

Develop a proactive succession planning strategy to mitigate the impact of their potential departure.

Thank You.