

DA76

Vo Tran Hoang Chau

Employee Promotion Analysis



1. INTRODUCTION

01

Problem Statement

The company is selecting outstanding employees for promotion to motivate staff to strive for excellence in their work. The Board of Directors (BOD) is evaluating which department will be nominated for promotions this time.

Context:

As a Data Analyst at company, you are responsible for reporting to the Board of Directors (BOD) on which key factors influencing employee promotion decisions and predict whether nominating employees from that department would be reasonable.

About Dataset

Dataset contains information about employees from various departments of a company. It includes demographic details, education, work experience, performance ratings, and training records.

Data has **155846 records and 14 columns**.



1. INTRODUCTION

02

Data Dictionary

employee_id	Unique identifier for each employee
department	Department where the employee works
region	Geographic region of the employee
education	Employee's highest education level
gender	Gender of the employee (m = male, f = female)
recruitment_channel	Source of recruitment (sourcing, other, referrer)
no_of_trainings	Number of training programs attended
age	Age of the employee
previous_year_rating	Performance rating from the previous year (1 to 5)
length_of_service	Number of years the employee has worked in the company
KPIs_met >80%	Whether the employee met >80% of Key Performance Indicators (1 = Yes, 0 = No)
awards_won?	Whether the employee has won an award (1 = Yes, 0 = No)
avg_training_score	Average training score of the employee
is_promoted	Whether the employee was promoted (1 = Yes, 0 = No)



1. INTRODUCTION

03

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 155846 entries, 0 to 155845
Data columns (total 14 columns):
#   Column                Non-Null Count  Dtype
---  -
0   employee_id           155846 non-null int64
1   department            155846 non-null object
2   region                155846 non-null object
3   education             155846 non-null object
4   gender                155846 non-null object
5   recruitment_channel    155846 non-null object
6   no_of_trainings       155846 non-null int64
7   age                   155846 non-null int64
8   previous_year_rating  142491 non-null float64
9   length_of_service     155846 non-null int64
10  KPIs_met >80%         155846 non-null int64
11  awards_won?           155846 non-null int64
12  avg_training_score     155846 non-null int64
13  is_promoted           155846 non-null int64
dtypes: float64(1), int64(8), object(5)
memory usage: 16.6+ MB
```

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

	0
employee_id	0
department	0
region	0
education	5330
gender	0
recruitment_channel	0
no_of_trainings	0
age	0
previous_year_rating	13355
length_of_service	0
KPIs_met >80%	0
awards_won?	0
avg_training_score	0
is_promoted	0

dtype: int64

```
data.duplicated().sum()
```

```
np.int64(0)
```

```
data['education'].fillna('Unknown', inplace=True)
```

Using Machine Learning to
predict the missing value of
previous_year_rating



2. EXPLORATORY DATA ANALYSIS

04

```
df = data.copy() # Tạo bản sao tránh ảnh hưởng dữ liệu gốc

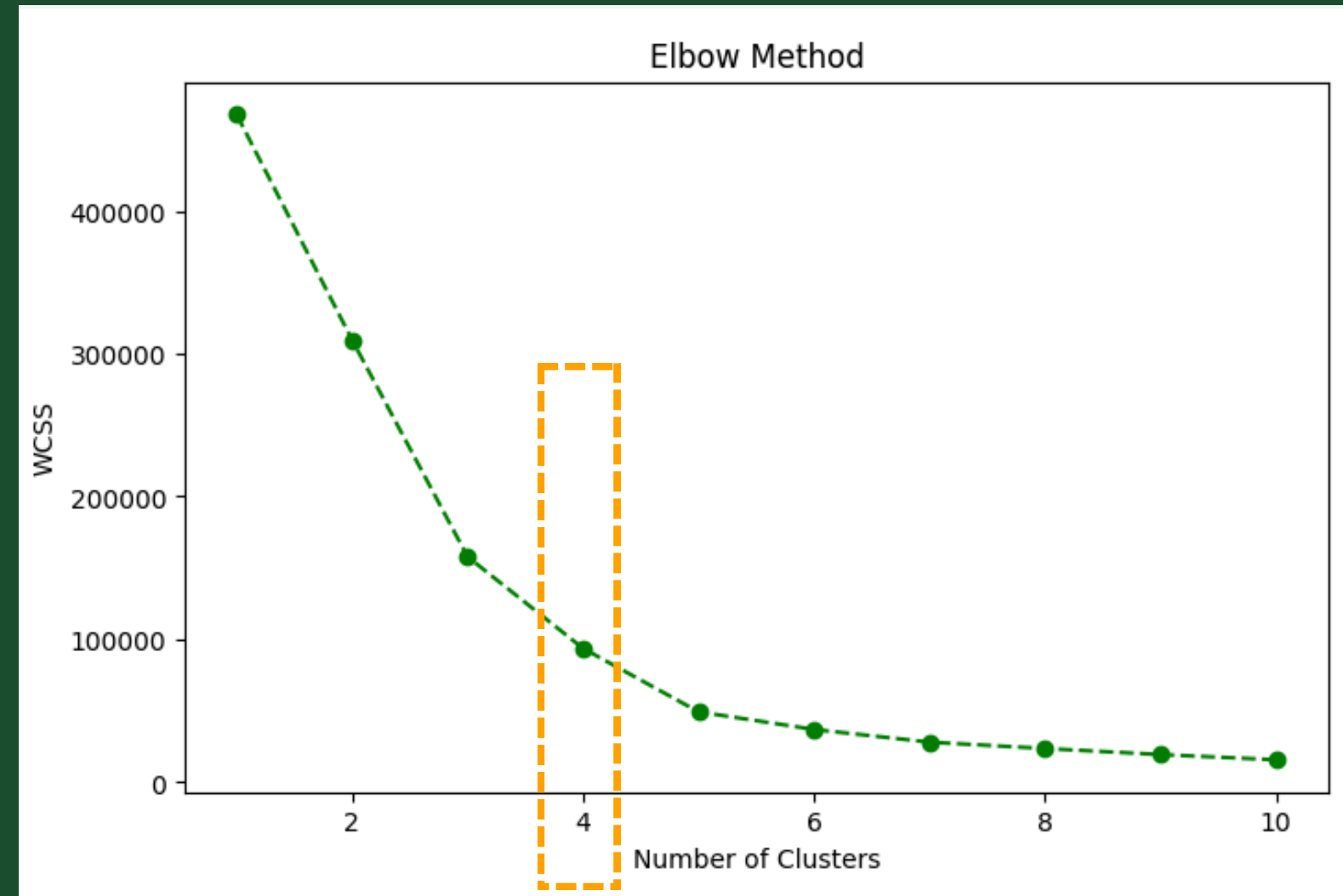
# Chia dữ liệu thành 2 phần:
df_train = df.dropna(subset=['previous_year_rating'])
df_missing = df[df['previous_year_rating'].isna()]

# Chỉ dùng các biến liên quan để phân cụm
features = ['avg_training_score', 'KPIs_met >80%', 'awards_won?']

# Chuẩn hóa dữ liệu trước khi phân cụm
scaler = StandardScaler()
df_scaled = pd.DataFrame(scaler.fit_transform(df[features]), columns=features)

# Xác định số cụm tối ưu bằng phương pháp Elbow
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, random_state=42, n_init=10)
    kmeans.fit(df_scaled)
    wcss.append(kmeans.inertia_)

plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), wcss, marker='o', linestyle='--')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.title('Elbow Method')
plt.show()
```



- K-Means was chosen for handling missing values in previous_year_rating because it effectively groups employees with similar performance characteristics
- Using K-Means to cluster employees into groups based on 3 factors: **avg_training_score** , **KPI met >80%** , **awards_won** to ensuring a more accurate imputation. It is a fast and efficient method that maintains the integrity of the data while avoiding bias
- Using the Elbow method to select the optimal k parameter (**k=4**)

2. EXPLORATORY DATA ANALYSIS

05

```
# Áp dụng thuật toán K-Means với k=4
kmeans = KMeans(n_clusters=4, random_state=42, n_init=10)
df_scaled['Cluster'] = kmeans.fit_predict(df_scaled)

# Gán lại kết quả phân cụm vào DataFrame gốc
df['Cluster'] = df_scaled['Cluster']
df_train['Cluster'] = df_scaled['Cluster']
```

- Apply k=4 in K-Means to cluster employees based on performance related features
- Fill the missing values in “previous_year_rating” by using **the average rating of each cluster**
- After filling the missing values, verify the data to confirm that no missing values remain

```
# Tính trung bình previous_year_rating theo từng cụm
cluster_means = df_train.groupby('Cluster')['previous_year_rating'].mean()

# Điền giá trị thiếu bằng trung bình cụm tương ứng
df.loc[df['previous_year_rating'].isna(), 'previous_year_rating'] = df.loc[df['previous_year_rating'].isna(), 'Cluster'].map(cluster_means)
df['previous_year_rating'] = df['previous_year_rating'].fillna(cluster_means)
df['previous_year_rating'] = df['previous_year_rating'].round().astype(int)

df.isnull().sum()
```

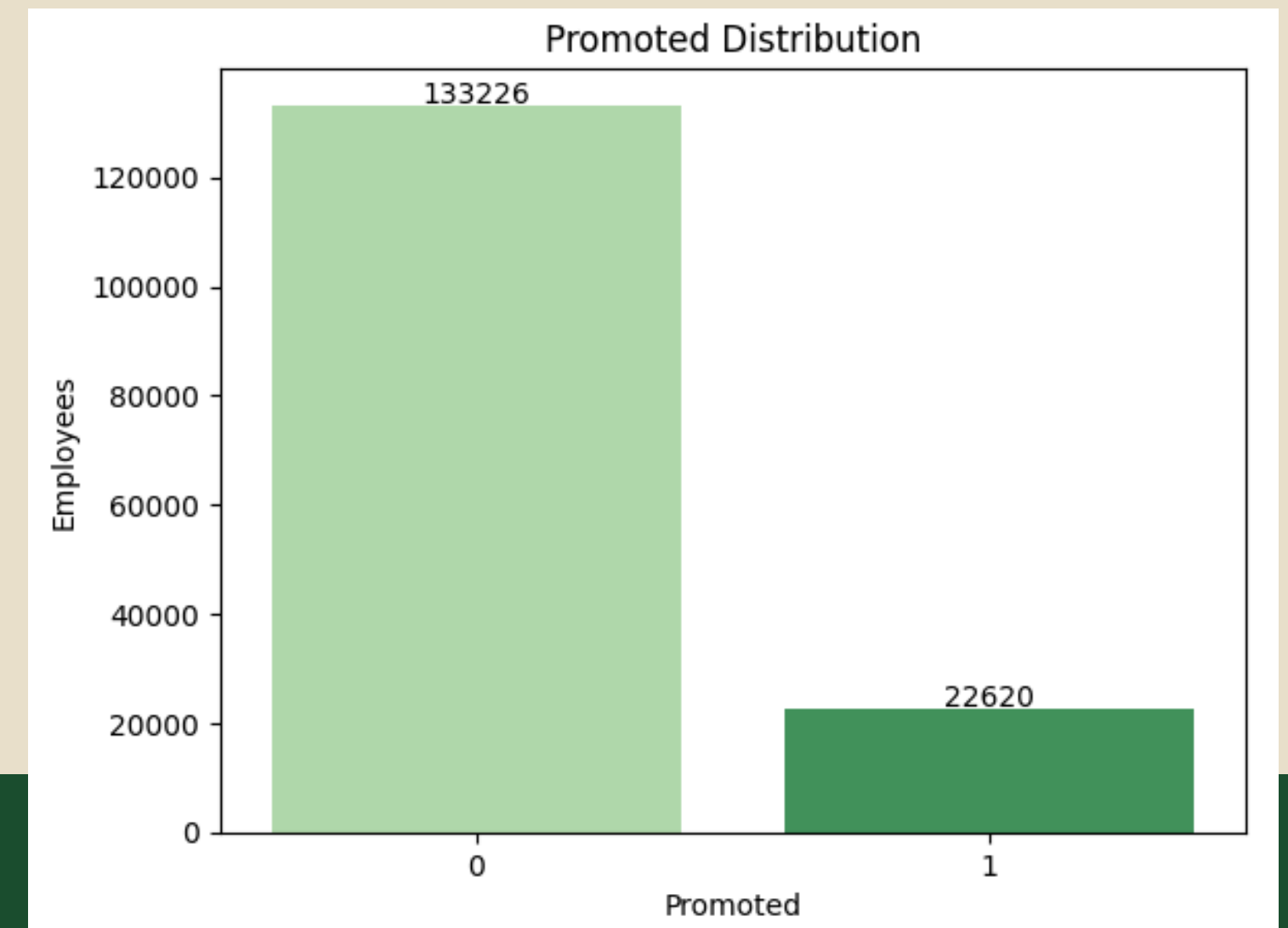
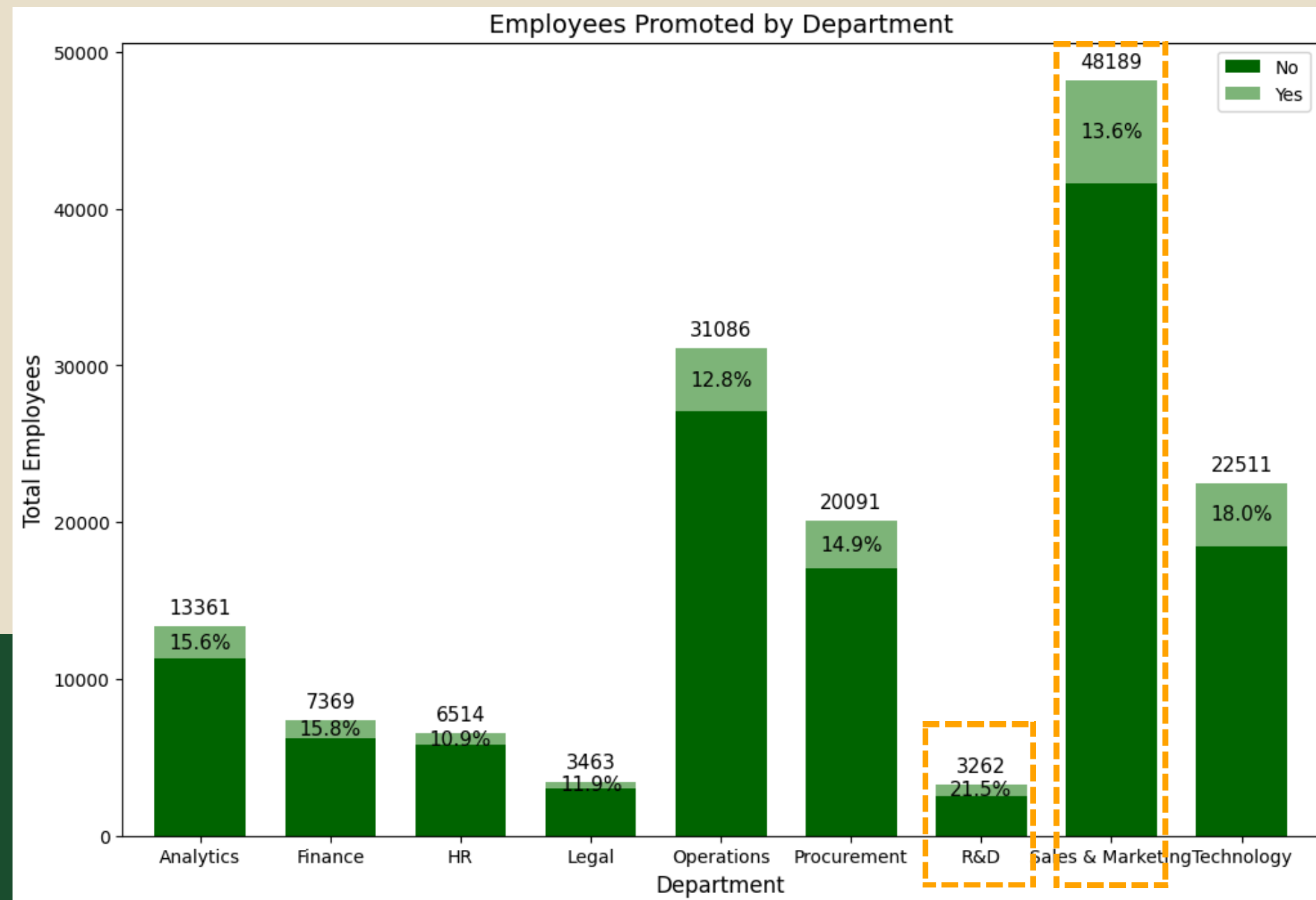
	0
employee_id	0
department	0
region	0
education	0
gender	0
recruitment_channel	0
no_of_trainings	0
age	0
previous_year_rating	0
length_of_service	0
KPIs_met >80%	0
awards_won?	0
avg_training_score	0
is_promoted	0
Cluster	0

dtype: int64



2. EXPLORATORY DATA ANALYSIS

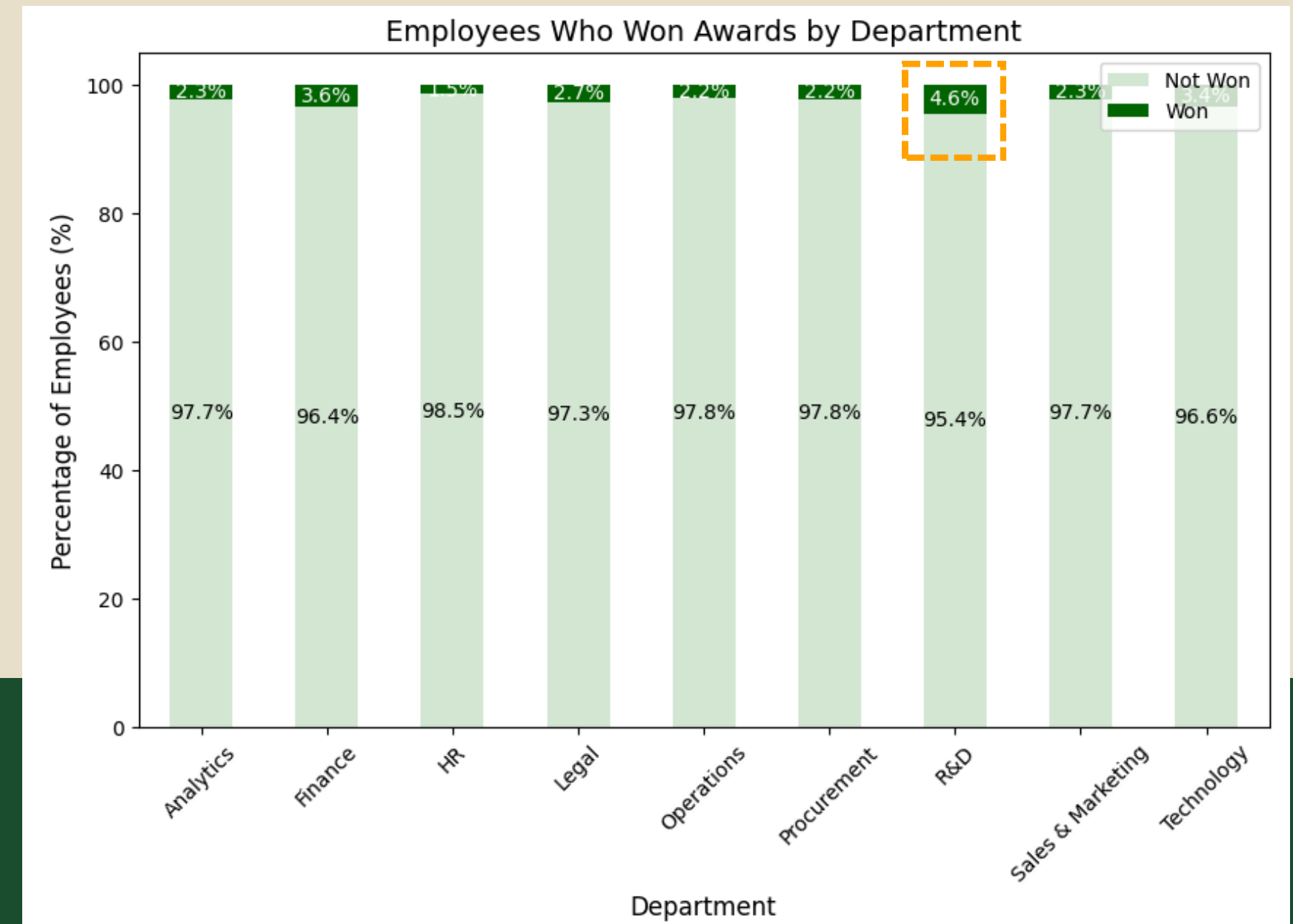
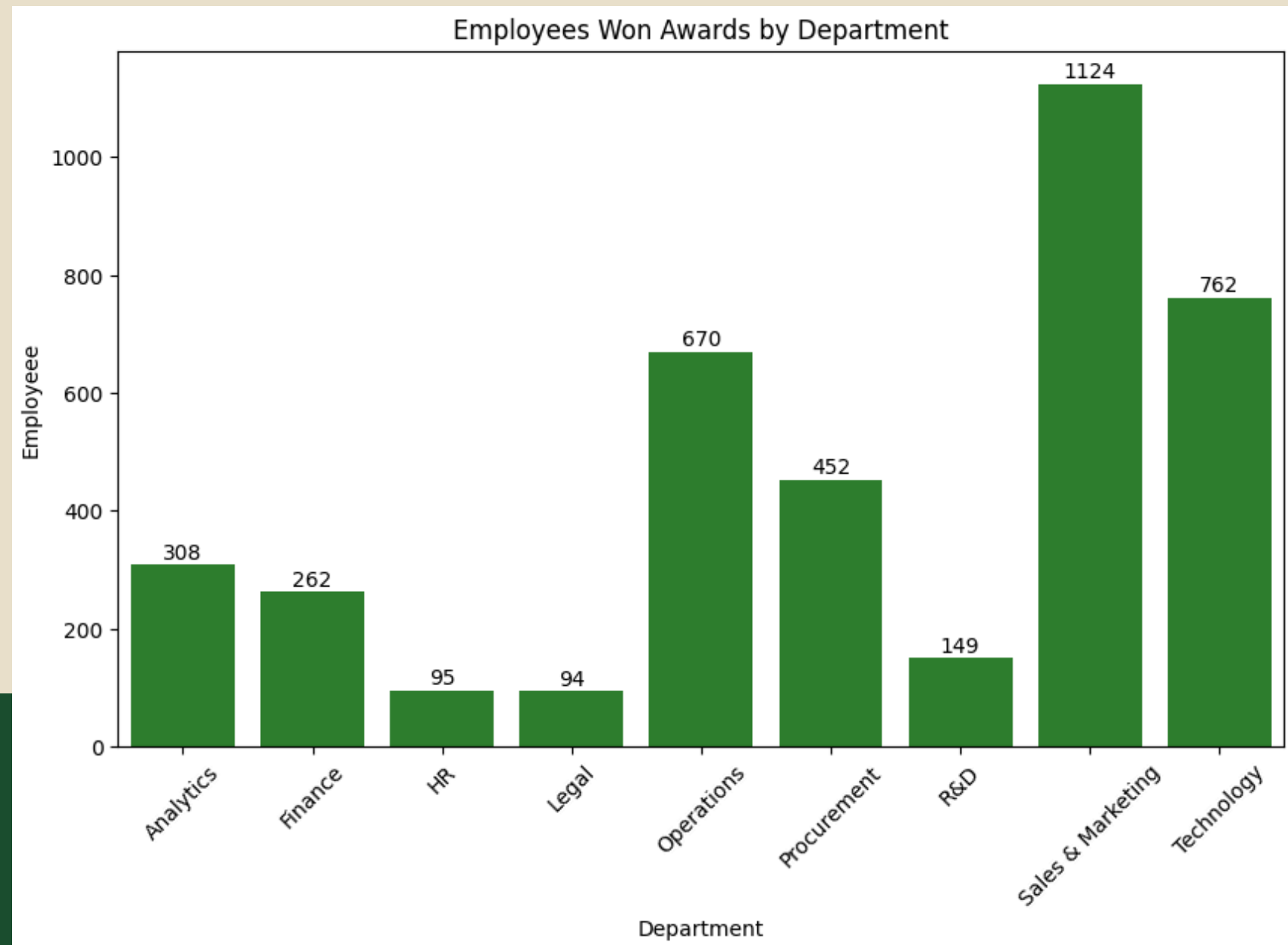
06



- Only 22,620 employees were promoted, while 133,226 employees were not, indicating a low overall promotion rate
- **Sales & Marketing is the largest department** in terms of employee count (48,189 employees) but the promotion rate is only 13.6%
- **R&D has the fewest employees** (3,262), however it has the **highest promotion rate (21.5%)**, significantly higher than other departments

2. EXPLORATORY DATA ANALYSIS

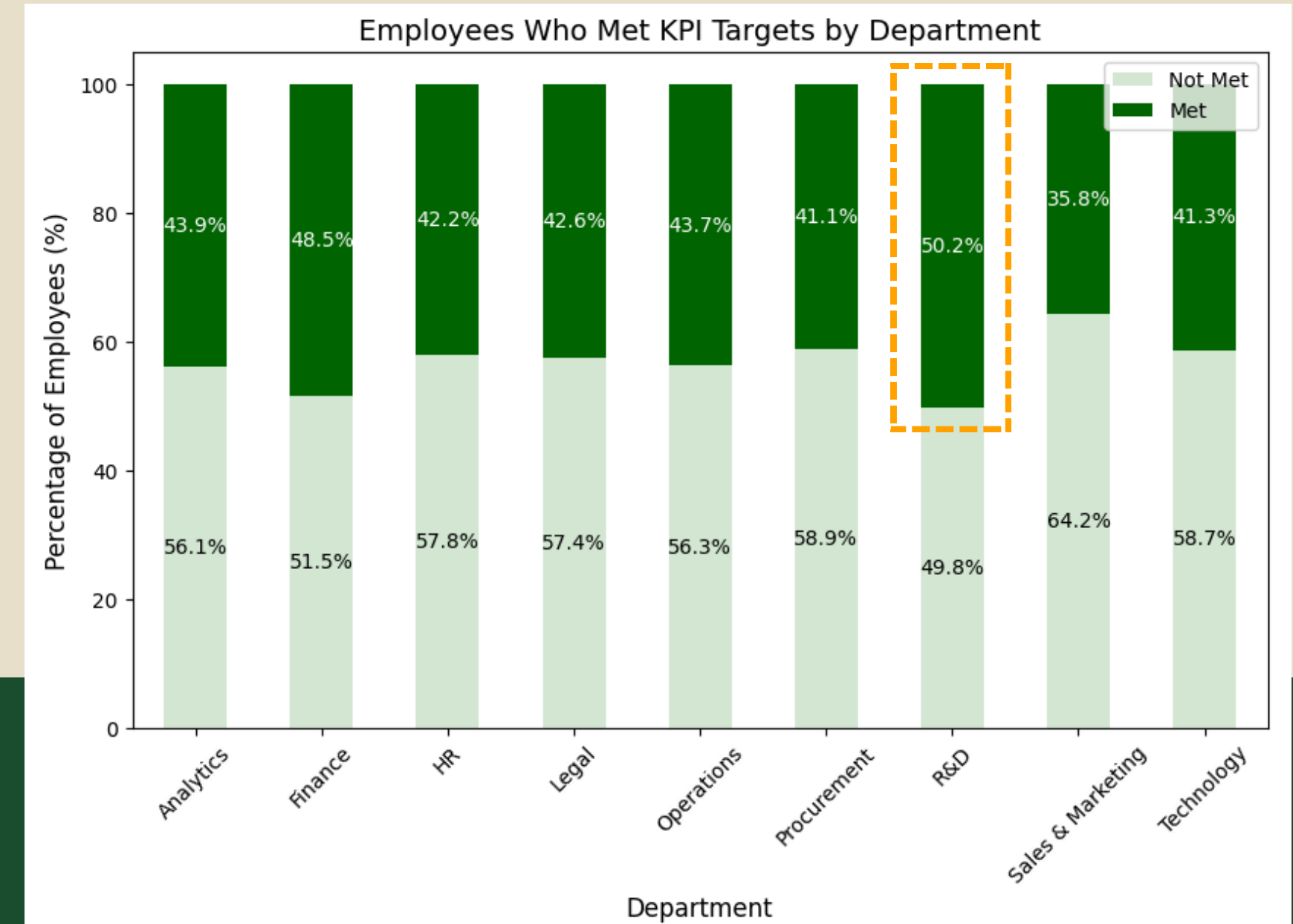
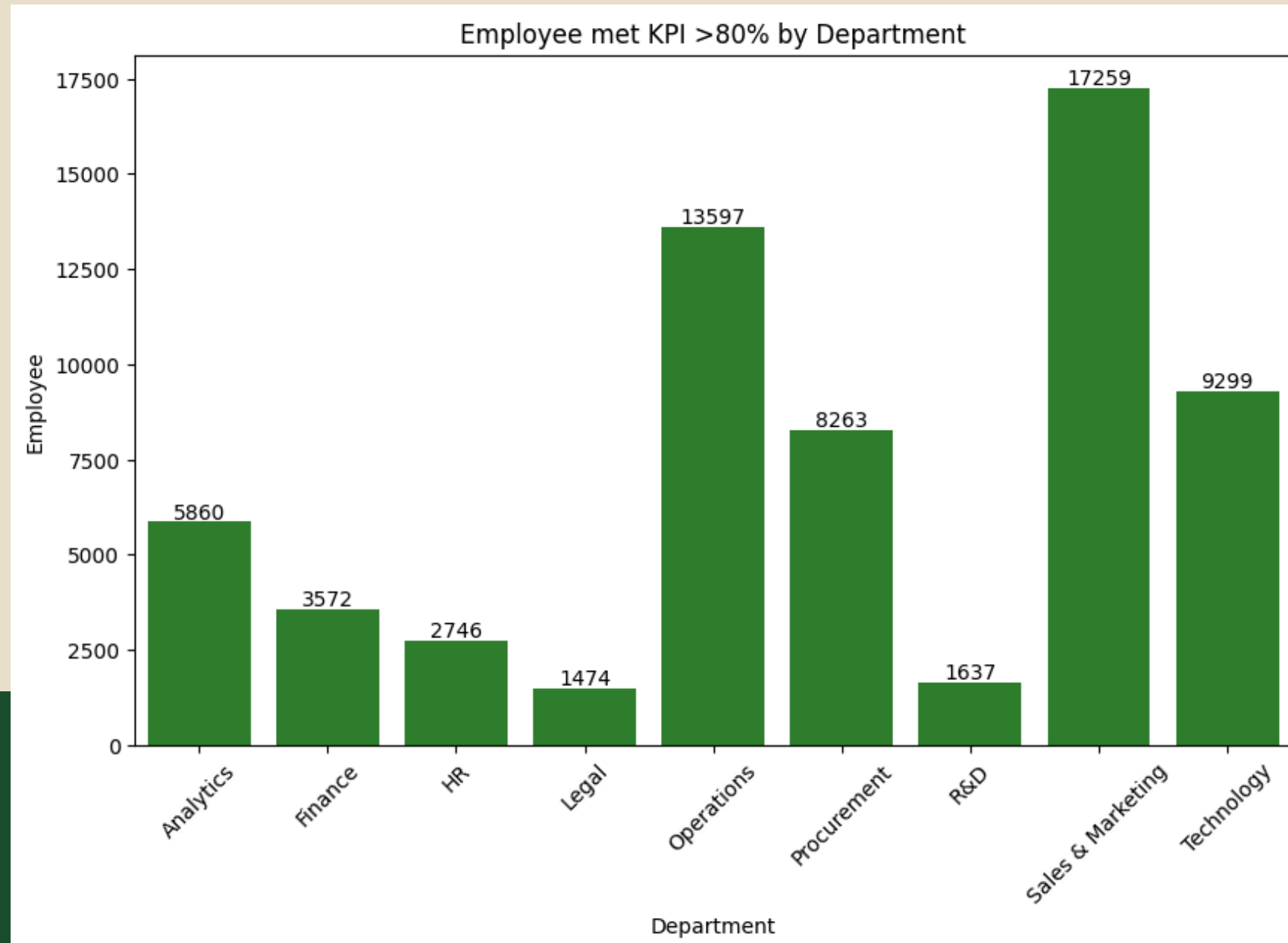
07



- **R&D** has the smallest number of employees, but it has the **highest award-winning rate at 4.6%**
- Finance and Technology have relatively fewer employees, but their award-winning rates are quite high, at 3.6% and 3.4%, respectively
- **Sales & Marketing** has the highest number of employees, which explains why it also has the most award winners (1,124 employees). However, award recipients **only make up 2.3%** of the department

2. EXPLORATORY DATA ANALYSIS

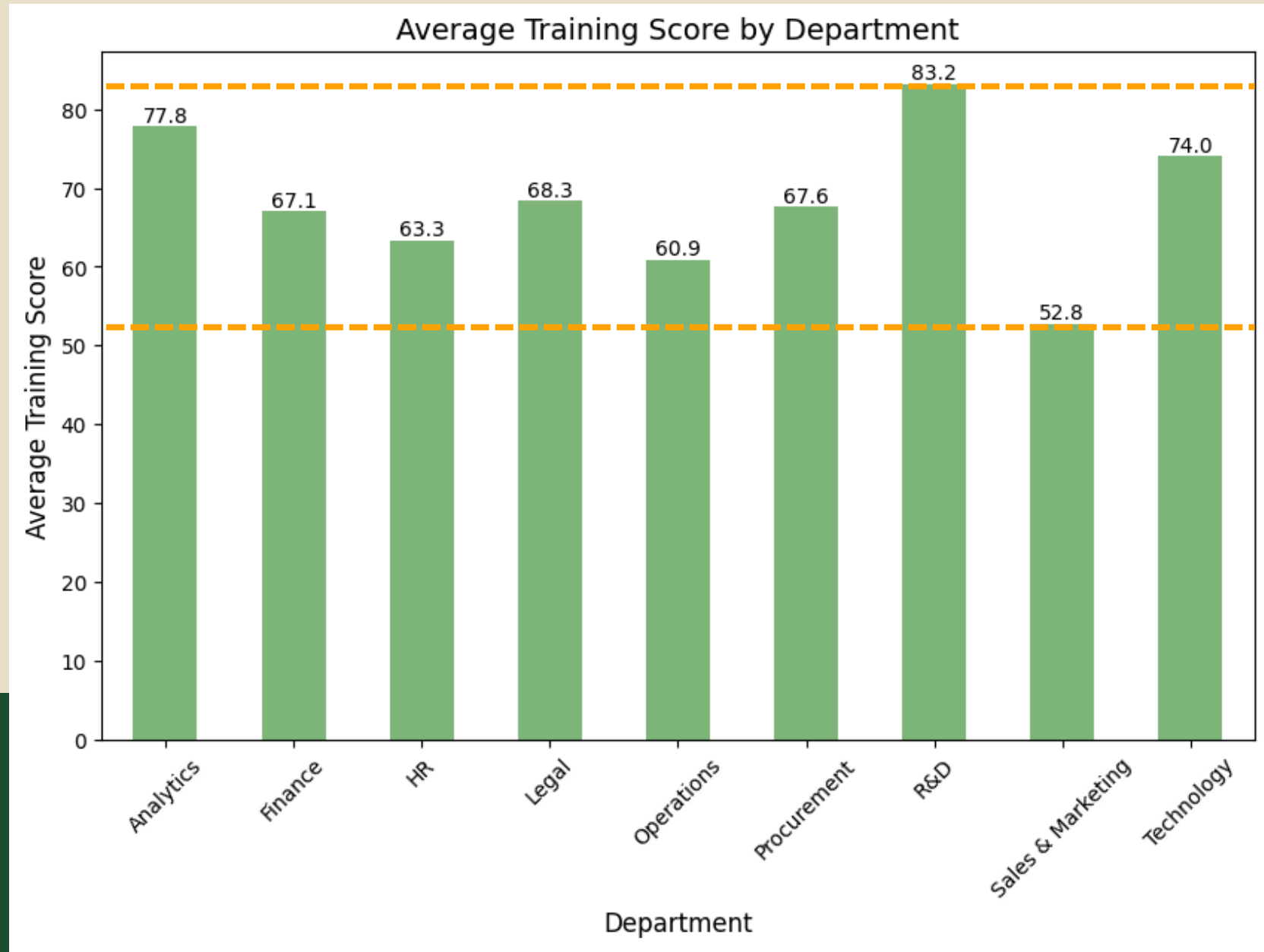
08



- **R&D** has the **highest KPI achievement rate (50.2%)**, meaning most employees in this department meet their targets despite being a small team
- Finance has the second-highest KPI achievement rate at 48.5%, just behind R&D (50.2%)
- The remaining departments have a relatively consistent KPI achievement rate, fluctuating around 42-43%
- **Sales & Marketing** department has the **lowest KPI achievement rate, at only 35.8%**

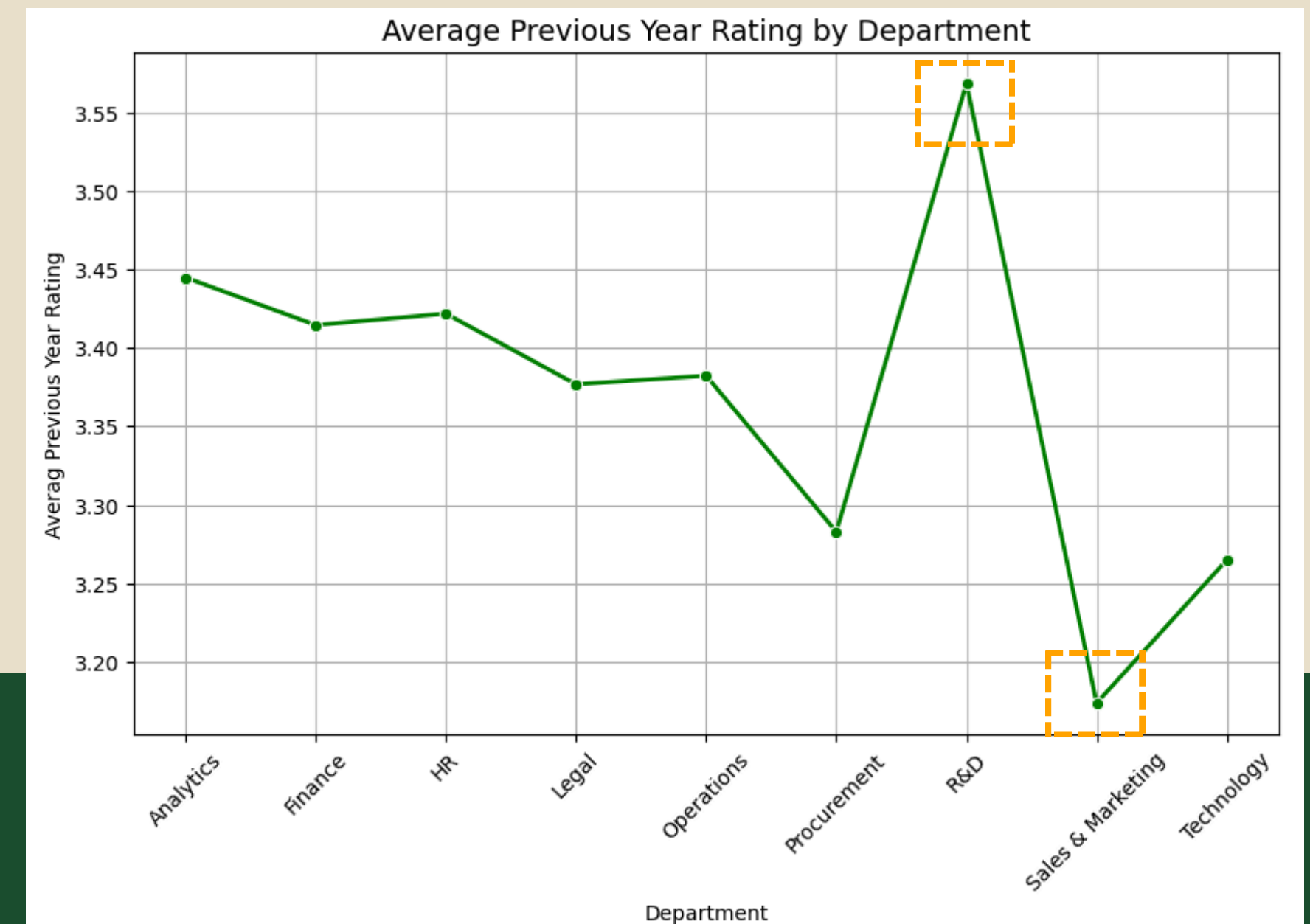
2. EXPLORATORY DATA ANALYSIS

09



R&D has the highest average training score (83.2), while Sales & Marketing has the lowest (52.8).

=> This suggests a significant gap in training levels across departments



R&D received the highest average rating from the previous year, whereas Sales & Marketing had the lowest.

=> This aligns with their training scores and may indicate a correlation between training and performance.

2. EXPLORATORY DATA ANALYSIS



After analyzing the factors related to promotion by department, **R&D** and **Sales & Marketing** were selected for the prediction model because they show the opposite results

R&D:

- Highest promotion rate (21.5%) despite having the fewest employees
- Highest average training score (83.2)
- Highest previous year rating, possibly reflecting strong performance

Sales & Marketing:

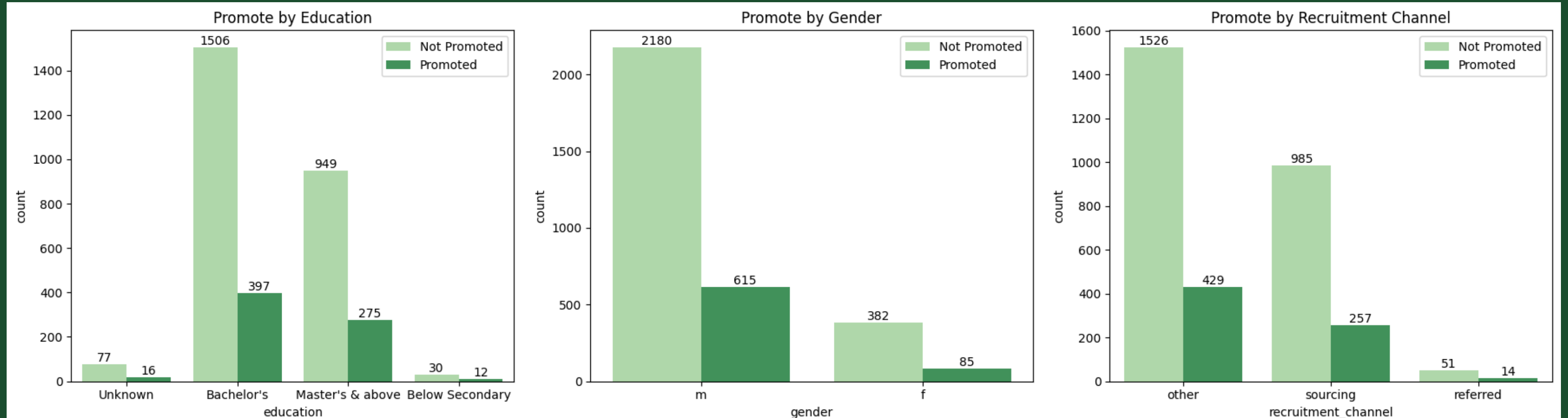
- Largest department but low promotion rate (13.6%)
- Lowest average training score (52.8)
- Lowest previous year rating -> may impact promotion opportunities
- Lowest KPI completion rate (35%), which could be a key factor

=> This selection helps create a clear comparison and validates the key factors influencing the prediction

3. PROMOTION ANALYSIS

R&D Department

11

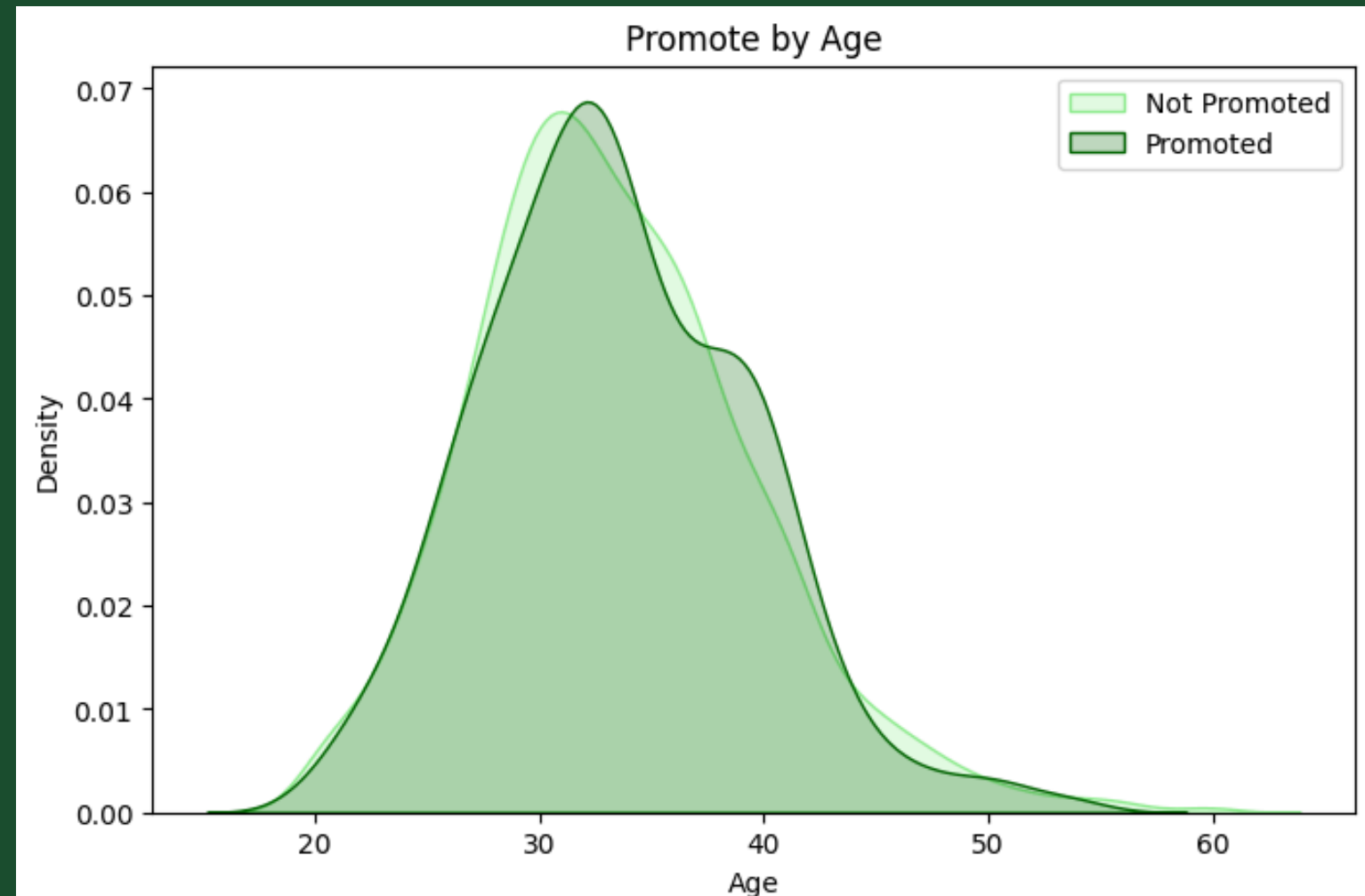


- Employees with **"Bachelor's" and "Master's & above" degrees have higher promotion**
- Men (2,180 not promoted, 615 promoted) have a higher promotion rate compared to women (382 not promoted, 85 promoted)
- The "Other" recruitment channel shows the highest number of promotions (429), suggesting it may attract more promising candidates.

3. PROMOTION ANALYSIS

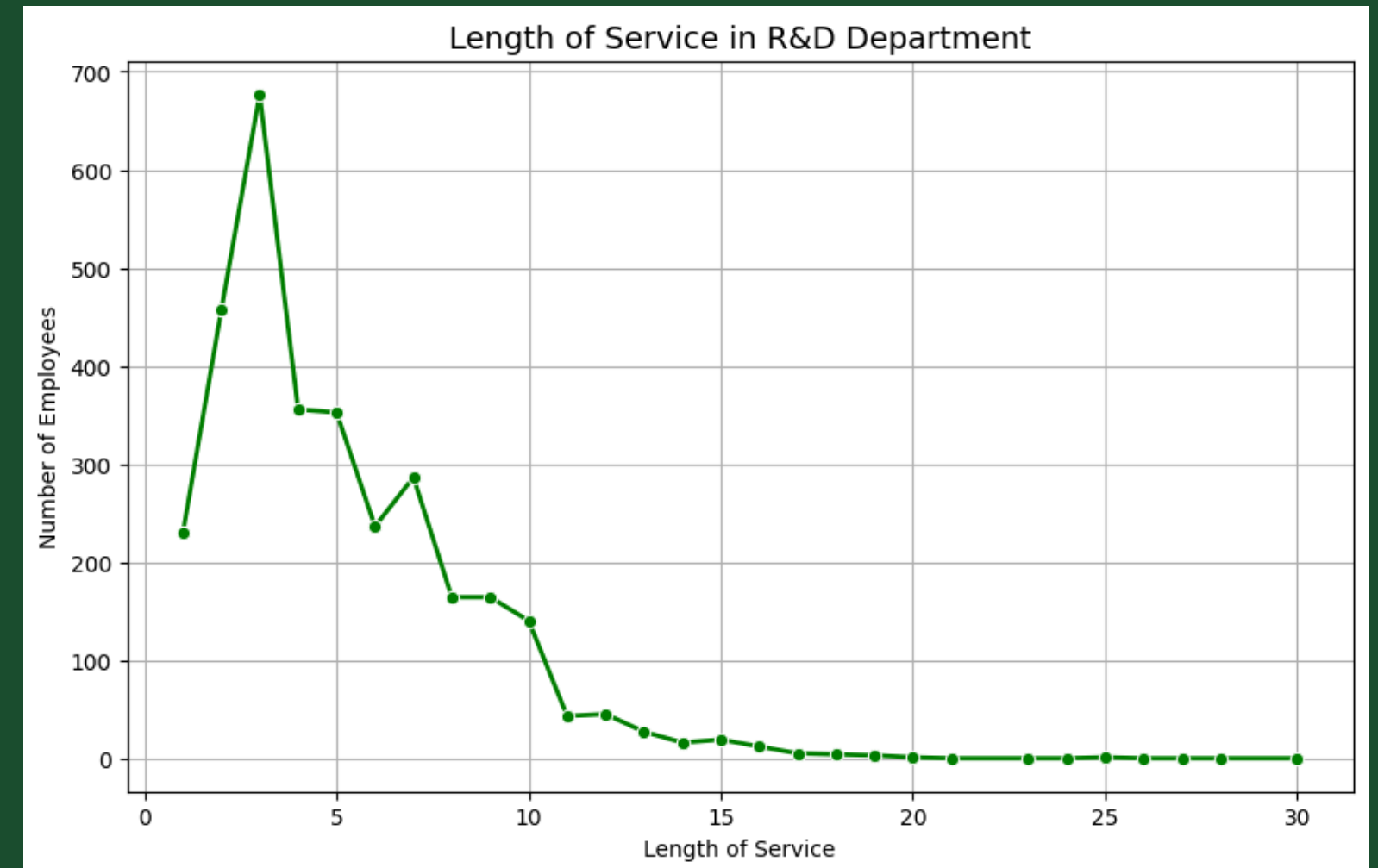
R&D Department

12



Both promoted and not promoted employees share a **similar peak density around the age of 30.**

=> However, this may indicate that mid-career employees are most active in seeking promotions.

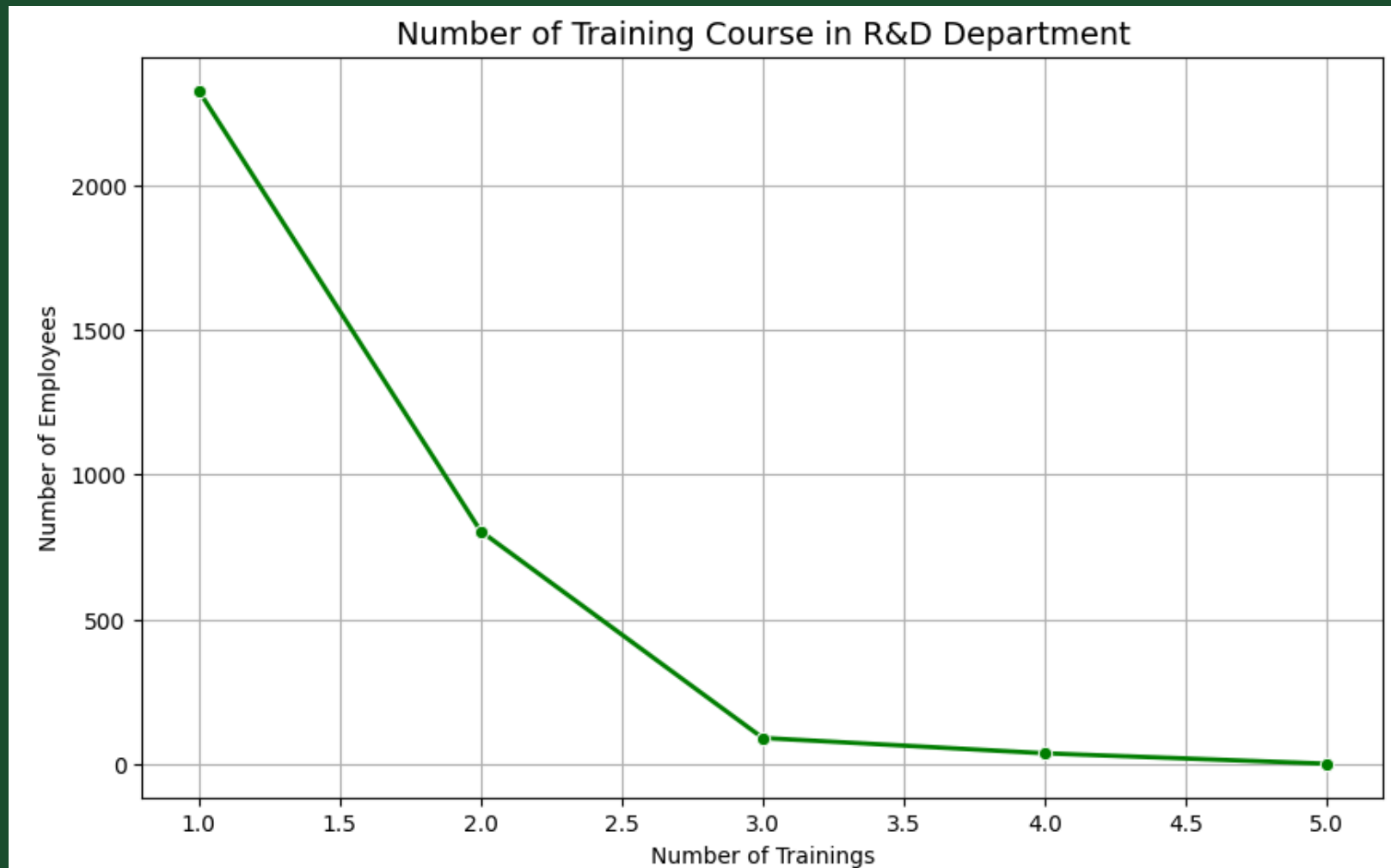


- Employees with **2 years of service in the R&D department dominate the distribution**, suggesting that the early career phase is critical for this department
- A steep decline after 2 years indicates challenges in retaining employees long term, or fewer opportunities in later years

3. PROMOTION ANALYSIS

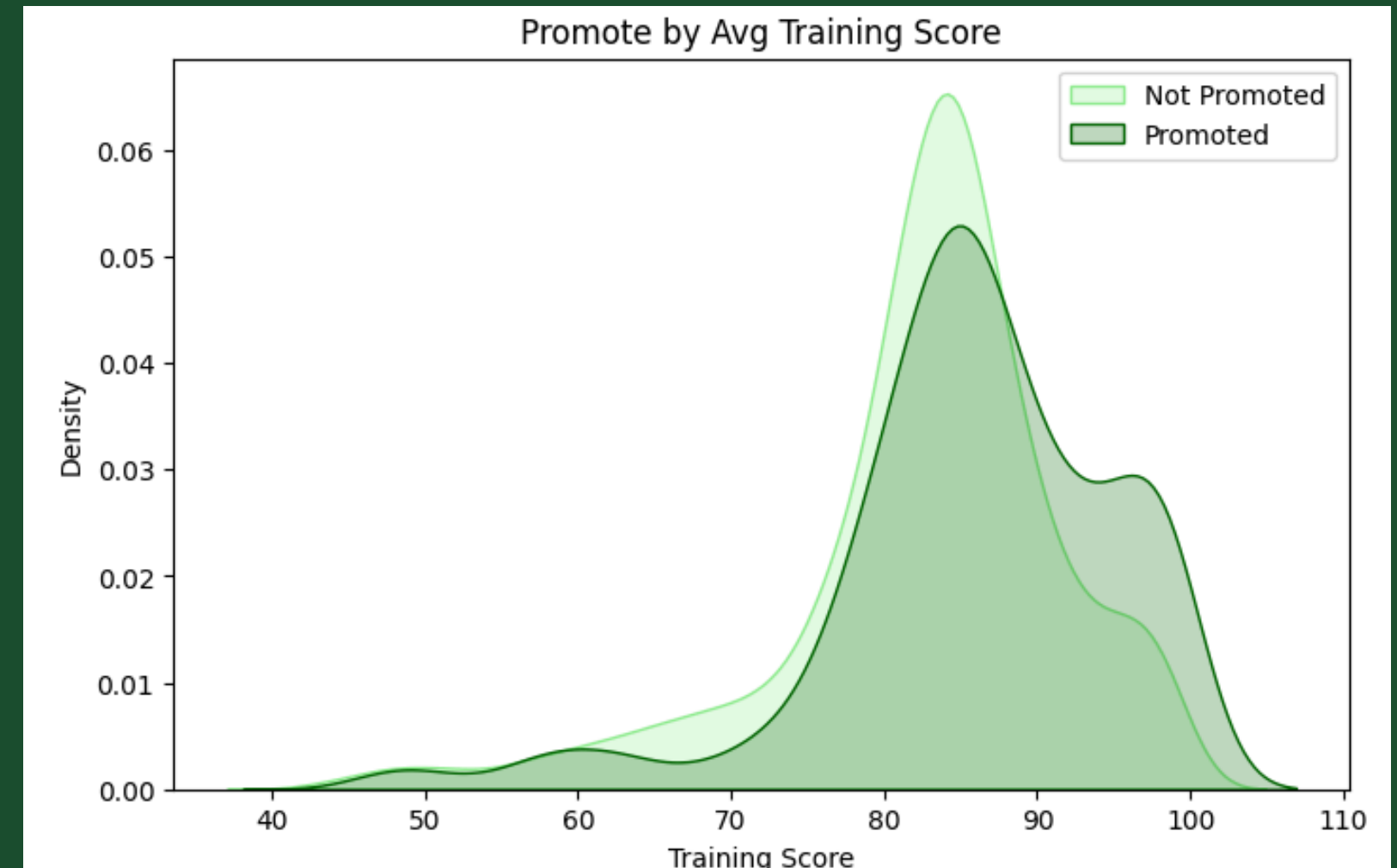
R&D Department

13



The green line graph shows that **most employees have only undergone 1 training course**

=> This may imply that employees receive limited training opportunities or that many employees prefer fewer sessions

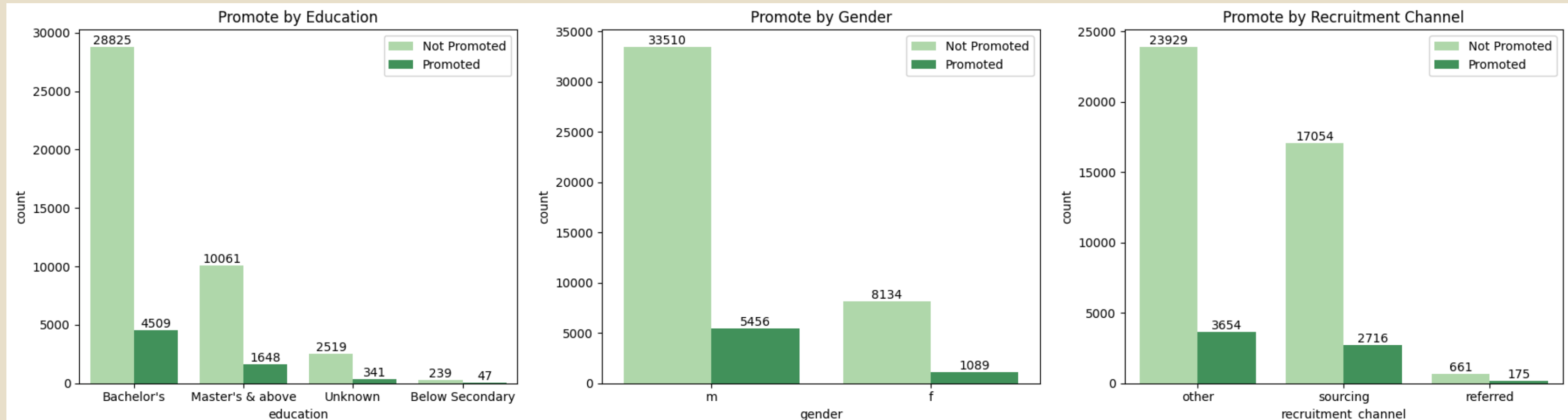


- The density plot indicates that **employees with higher training scores are more likely to be promoted.**
- The promoted distribution peaks at higher training scores compared to the not promoted distribution

3. PROMOTION ANALYSIS

Sales & Marketing Department

14

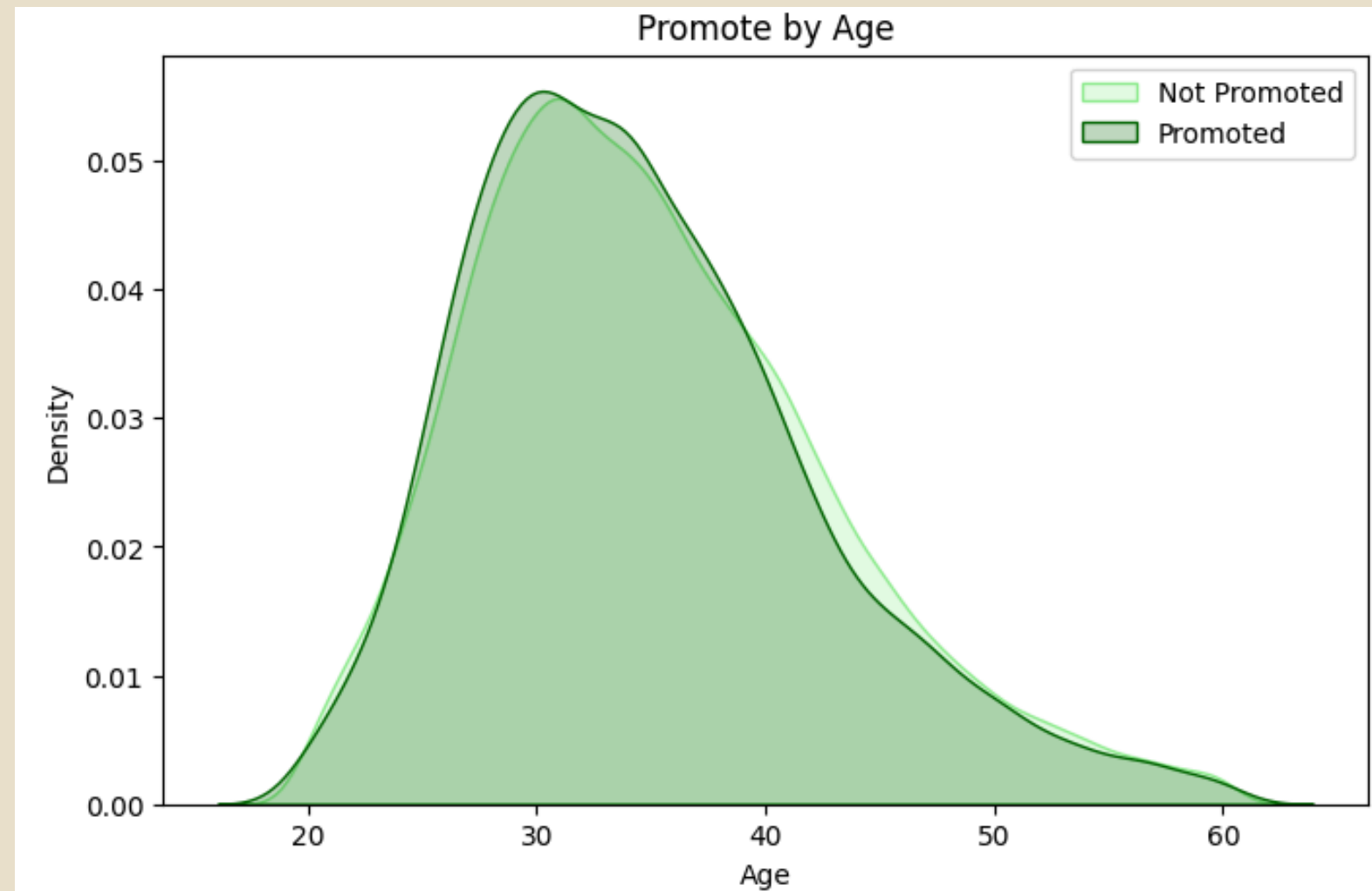


- **The highest number of promoted employees holds a Bachelor's degree (4,509)**
- Male employees form the majority of both promoted (5,456) and not promoted (33,510) categories, indicating a gender imbalance
- The "Other" recruitment channel contributed the most promoted employees (3,654) similar to R&D department, indicating its effectiveness in attracting talent with higher promotion potential

3. PROMOTION ANALYSIS

Sales & Marketing Department

15



The density distribution highlights a prominent **peak around the age of 30** for both promoted and not-promoted employees **similar to R&D department**
=> This suggests that individuals in their early 30s are highly active in their career progression or opportunities around that age are abundant

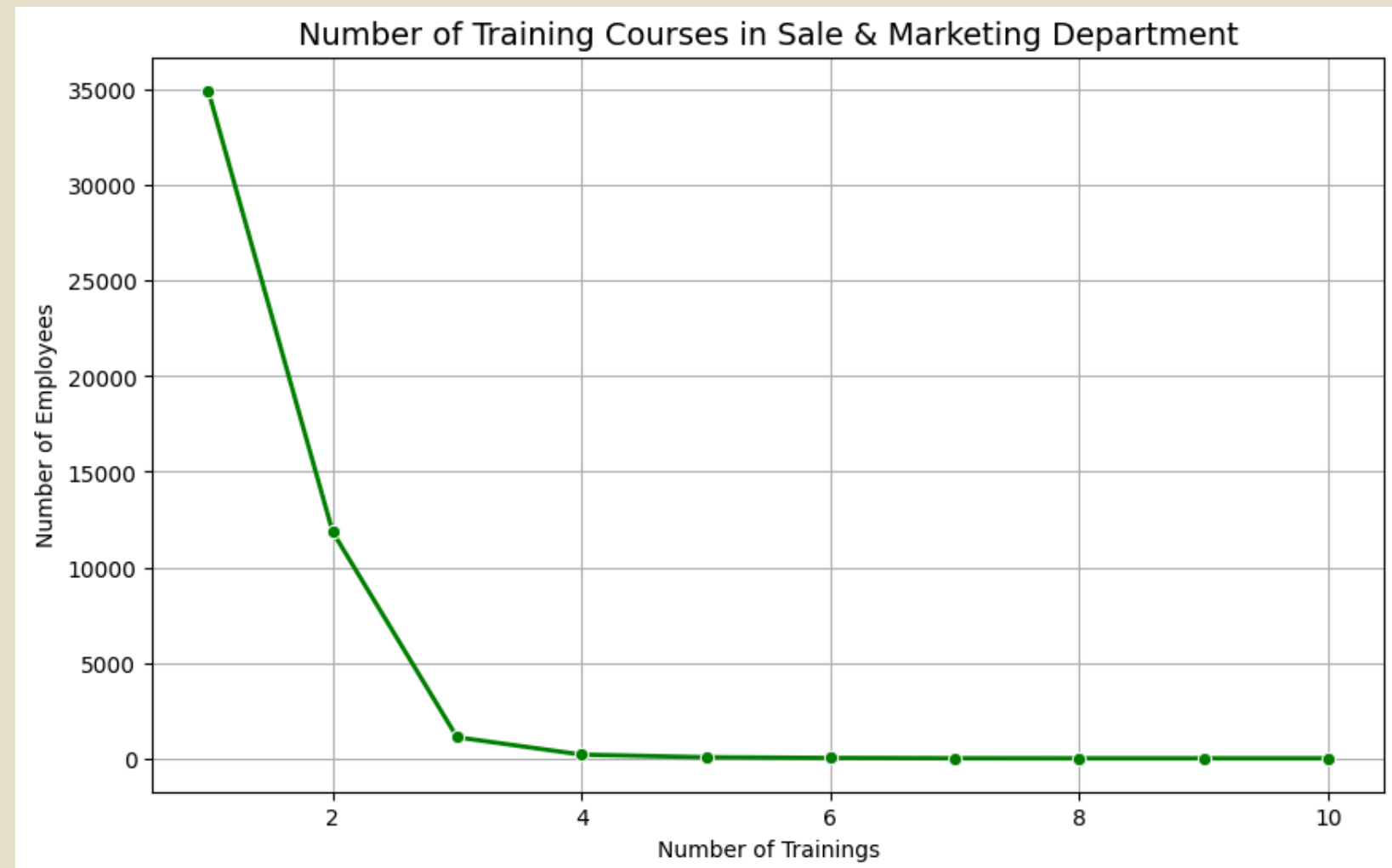


- The data shows that the highest concentration of employees falls in the **0-2 years** of service range
- Beyond 2 years, the curve slopes gently, suggesting that employees who remain past this initial phase tend to stay longer-term but in smaller numbers

3. PROMOTION ANALYSIS

Sales & Marketing Department

16



- Similar to R&D department, **most of employees have completed 1 training course**
- Beyond 4 courses, the number of employees plateaus at a very low level, suggesting that only a small fraction of employees engage in extensive training



- Employees with average training scores **around 50 are most common** in both promoted and not promoted groups
- However, **the promoted distribution appears taller and narrower**, implying that higher training scores are slightly associated with a greater likelihood of promotion

4. PROMOTE PREDICTION

R&D Department

17

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
le = LabelEncoder()
df_rd['education'] = le.fit_transform(df_rd['education'])
df_rd['gender'] = le.fit_transform(df_rd['gender'])
df_rd['recruitment_channel'] = le.fit_transform(df_rd['recruitment_channel'])
```



Encoding

```
# Xác định feature và target
X = df_rd[['education', 'gender', 'recruitment_channel', 'no_of_trainings', 'age', 'previous_year_rating', 'length_of_service', 'KPIs_met >80%', 'awards_won?', 'avg_training_score']]
y = df_rd['is_promoted']
```

```
# Chuẩn hóa dữ liệu
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X) # Fit và transform toàn bộ dữ liệu

# Chia dữ liệu thành tập train và test
X_scaled_train, X_scaled_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42, stratify=y)

# Áp dụng SMOTE trên tập train
smote = SMOTE(sampling_strategy=0.5, random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_scaled_train, y_train)

# Kiểm tra kích thước tập dữ liệu sau khi xử lý
print("Kích thước tập train trước SMOTE:", X_scaled_train.shape)
print("Kích thước tập train sau SMOTE:", X_train_resampled.shape)
```



Split data into train
and test, then apply
SMOTE on train

```
Kích thước tập train trước SMOTE: (2283, 10)
Kích thước tập train sau SMOTE: (2689, 10)
```

4. PROMOTE PREDICTION - Random Forest

18

```
# Random Forest
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train_resampled, y_train_resampled)
```

RandomForestClassifier

```
RandomForestClassifier(random_state=42)
```

```
# Dự đoán trên tập test
rf_pred = rf_model.predict(X_scaled_test)
```

```
# Đánh giá mô hình
print("Accuracy Score:", accuracy_score(y_test, rf_pred))
print("\nClassification Report:\n", classification_report(y_test, rf_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, rf_pred))
```

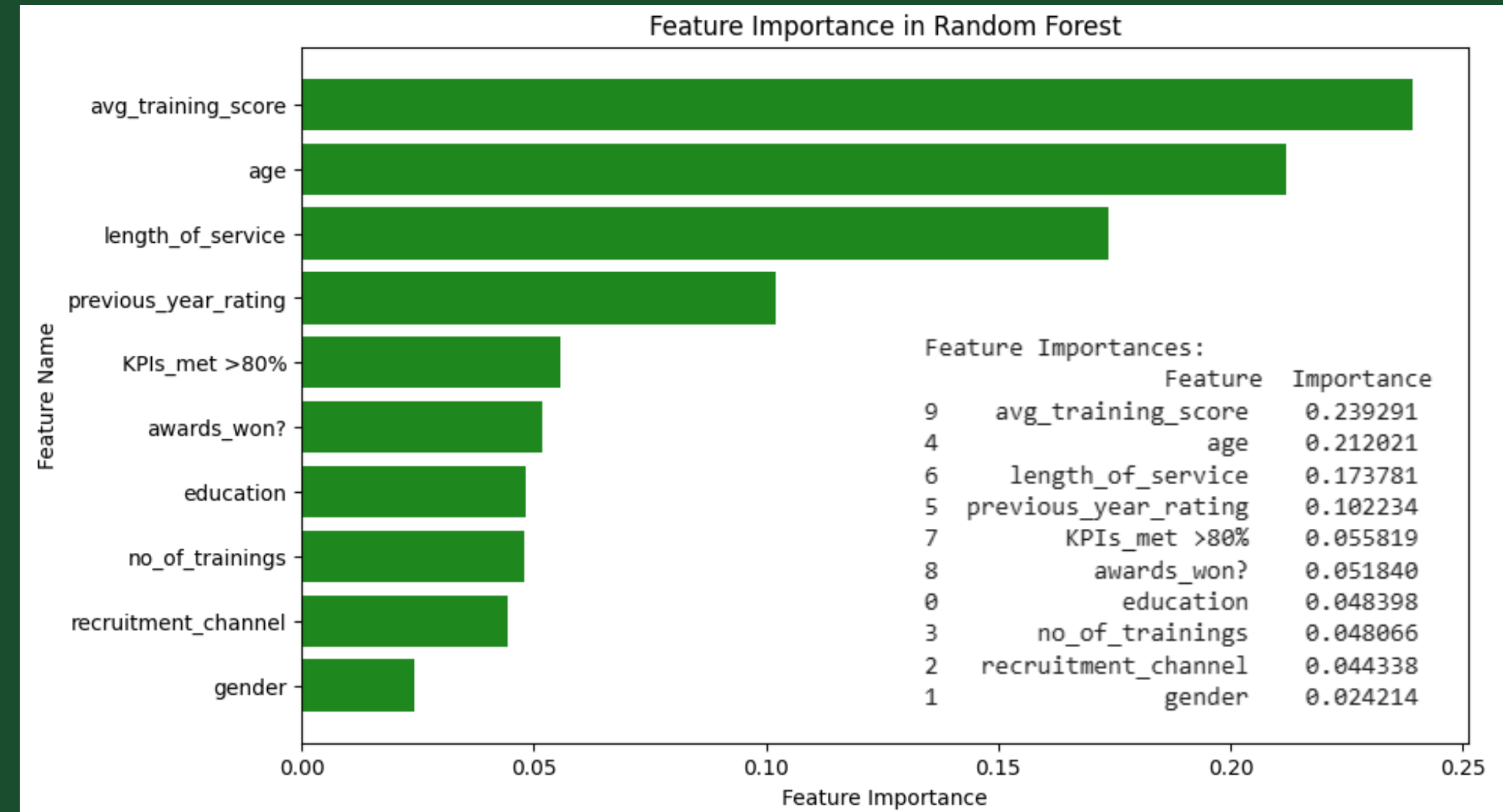
Accuracy Score: 0.8008171603677222

Classification Report:

	precision	recall	f1-score	support
0	0.84	0.92	0.88	769
1	0.56	0.36	0.43	210
accuracy			0.80	979
macro avg	0.70	0.64	0.66	979
weighted avg	0.78	0.80	0.78	979

Confusion Matrix:

```
[[709  60]
 [135  75]]
```



- The **Accuracy Score is 80%**, indicating that the model performs well overall
- Low **Precision for the "Promote" class (0.56)**, this means that only 56% of employees predicted to be promoted were actually promoted
- The most important features for prediction include average training score, age, and length of service

4. PROMOTE PREDICTION - Random Forest

19

```
# Grid Search
rf = RandomForestClassifier()
grid_space = {'max_depth':[5, 10, 15, 20],
              'n_estimators':[10, 100, 10],
              'criterion':['gini','entropy']}
rf_grid = GridSearchCV(rf, param_grid= grid_space, scoring='accuracy')
model_grid = rf_grid.fit(X_train_resampled, y_train_resampled)
# Best hyperparameters
print('Best hyperparameters: '+str(model_grid.best_params_))
print('Best score: '+str(model_grid.best_score_))

Best hyperparameters: {'criterion': 'gini', 'max_depth': 20, 'n_estimators': 100}
Best score:0.7954926515890983

best_rf = RandomForestClassifier(
    criterion='gini',
    max_depth=10,
    n_estimators=30,
    random_state=42)
best_rf.fit(X_train_resampled, y_train_resampled)
```

```
RandomForestClassifier
RandomForestClassifier(max_depth=10, n_estimators=30, random_state=42)
```

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Dự đoán trên tập test
rf_best = best_rf.predict(X_scaled_test)

# Đánh giá hiệu suất
print("Accuracy Score:", accuracy_score(y_test, rf_best))
print("\nClassification Report:\n", classification_report(y_test, rf_best))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, rf_best))
```

Accuracy Score: 0.8110316649642493

Classification Report:

	precision	recall	f1-score	support
0	0.84	0.94	0.89	769
1	0.61	0.33	0.43	210
accuracy			0.81	979
macro avg	0.72	0.64	0.66	979
weighted avg	0.79	0.81	0.79	979

Confusion Matrix:

```
[[725  44]
 [141  69]]
```

- Using GridSearchCV, the model reached an **Accuracy of 81.01%**, indicating a strong ability to predict employees promotion (increase from the previous 80%)
 - The **Precision increased to 0.61**, meaning the model is more accurate in identifying employees for promotion
- => **Using GridSearchCV in Random Forest get higher accuracy and precision than previous**

4. PROMOTE PREDICTION - Gradient Boosting

20

```
# Gradient Boosting
gb_model = GradientBoostingClassifier()
gb_model.fit(X_train_resampled, y_train_resampled)

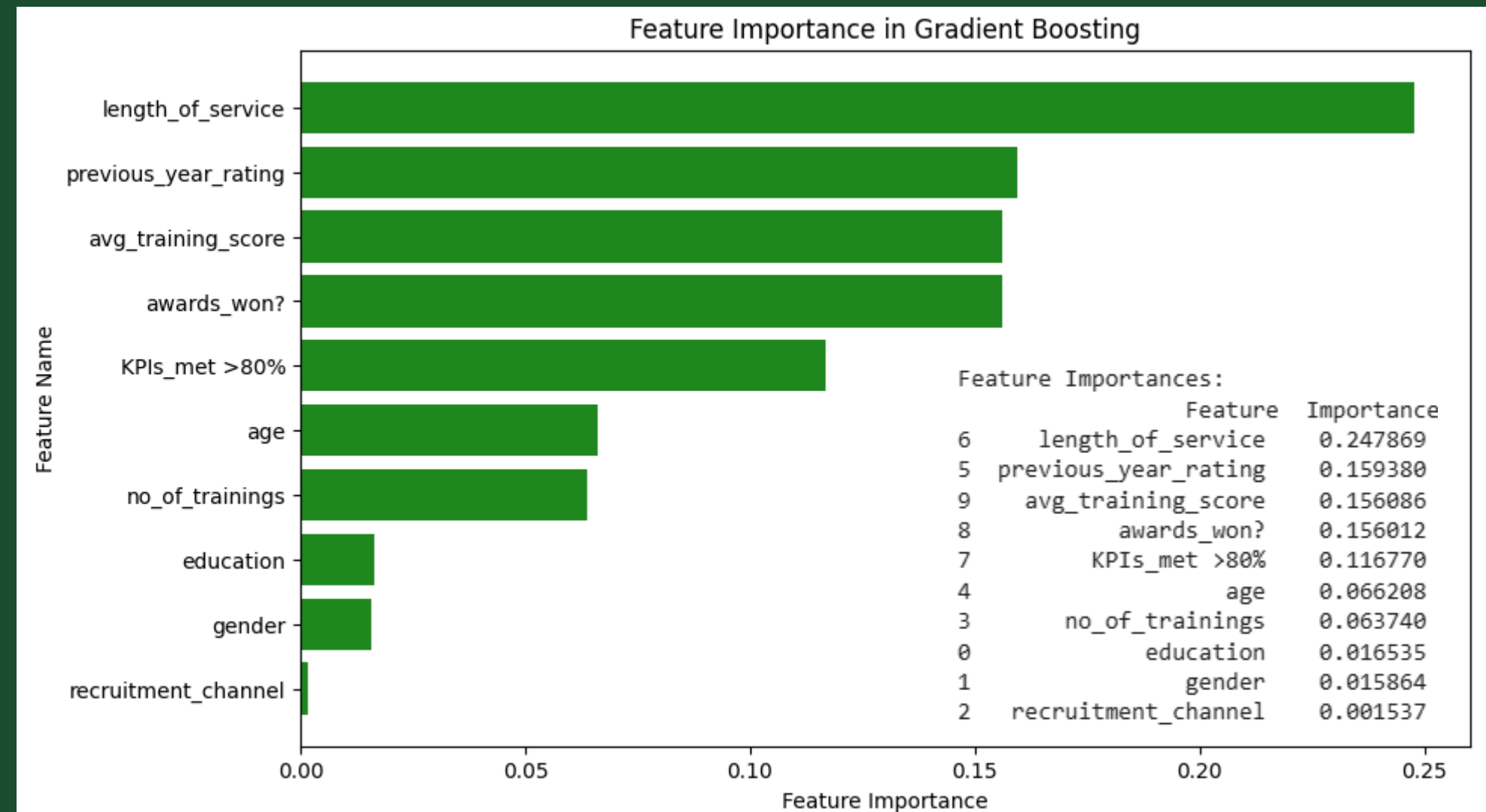
# Dự đoán trên tập test
gb_pred = gb_model.predict(X_scaled_test)

# Đánh giá mô hình
print("Accuracy Score:", accuracy_score(y_test, gb_pred))
print("\nClassification Report:\n", classification_report(y_test, gb_pred))
```

Accuracy Score: 0.8140960163432074

Classification Report:

	precision	recall	f1-score	support
0	0.83	0.96	0.89	769
1	0.65	0.29	0.40	210
accuracy			0.81	979
macro avg	0.74	0.62	0.65	979
weighted avg	0.79	0.81	0.79	979



- **Accuracy Score: 81.5%**, slightly higher than Random Forest (81.01%), this indicates that the Gradient Boosting model performs slightly better overall
- **Precision 0.65** => the model correctly identifies more employees who are actually promoted
- "Length of service" is the most important feature, followed by "Previous year rating" and "Avg training score"

4. PROMOTE PREDICTION - Gradient Boosting

21

```
# Grid Search
gb = GradientBoostingClassifier()

parameters = {
    'loss': ['deviance', 'exponential'],
    'learning_rate': [0.001, 0.1, 1, 10],
    'n_estimators': [100, 150, 180, 200]
}

grid_search = GridSearchCV(gb, parameters, cv=5, n_jobs=-1, verbose=1)
grid_search.fit(X_train_resampled, y_train_resampled)
print(grid_search.best_params_)
```

```
# Lấy bộ tham số tối ưu từ Grid Search
best_params = grid_search.best_params_

# Huấn luyện lại mô hình với tham số tối ưu và dữ liệu sau SMOTE
best_gb = GradientBoostingClassifier(**best_params)
best_gb.fit(X_train_resampled, y_train_resampled)

# Dự đoán trên tập test
gb_best = best_gb.predict(X_scaled_test)

# Đánh giá mô hình
print("Accuracy Score:", accuracy_score(y_test, gb_best))
print("\nClassification Report:\n", classification_report(y_test, gb_best))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, gb_best))
```

Accuracy Score: 0.8161389172625128

Classification Report:

	precision	recall	f1-score	support
0	0.84	0.95	0.89	769
1	0.64	0.32	0.43	210
accuracy			0.82	979
macro avg	0.74	0.64	0.66	979
weighted avg	0.80	0.82	0.79	979

Confusion Matrix:

```
[[732  37]
 [143  67]]
```

- Using GridSearchCV results higher **Accuracy Score: 81.63%** than the previous Gradient Boosting model (81.5%)
 - **Precision is 0.64**, slightly lower than the previous 0.65 but still better than Random Forest.
- => **Using GridSearchCV in Gradient Boosting get similar results with Random Forest**

4. PROMOTE PREDICTION

Sales & Marketing Department

22

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
le = LabelEncoder()
df_sm['education'] = le.fit_transform(df_sm['education'])
df_sm['gender'] = le.fit_transform(df_sm['gender'])
df_sm['recruitment_channel'] = le.fit_transform(df_sm['recruitment_channel'])
```



Encoding

```
# Xác định feature và target
X1 = df_sm[['education', 'gender', 'recruitment_channel', 'no_of_trainings', 'age', 'previous_year_rating', 'length_of_service', 'KPIs_met >80%', 'awards_won?', 'avg_training_score']]
y1 = df_sm['is_promoted']
```

```
# Chuẩn hóa dữ liệu
scaler = StandardScaler()
X1_scaled = scaler.fit_transform(X1) # Fit và transform toàn bộ dữ liệu

# Chia dữ liệu thành tập train và test
X1_scaled_train, X1_scaled_test, y1_train, y1_test = train_test_split(X1_scaled, y1, test_size=0.3, random_state=42, stratify=y1)

# Áp dụng SMOTE trên tập train
smote = SMOTE(sampling_strategy=0.5, random_state=42)
X1_train_resampled, y1_train_resampled = smote.fit_resample(X1_scaled_train, y1_train)

# Kiểm tra kích thước tập dữ liệu sau khi xử lý
print("Kích thước tập train trước SMOTE:", X1_scaled_train.shape)
print("Kích thước tập train sau SMOTE:", X1_train_resampled.shape)
```



Split data into train
and test, then apply
SMOTE on train

```
Kích thước tập train trước SMOTE: (33732, 10)
Kích thước tập train sau SMOTE: (43726, 10)
```

4. PROMOTE PREDICTION - Random Forest

```
# Random Forest
rf_model1 = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model1.fit(X1_train_resampled, y1_train_resampled)

# Dự đoán trên tập test
rf1_pred = rf_model1.predict(X1_scaled_test)

# Đánh giá mô hình
print("Accuracy Score:", accuracy_score(y1_test, rf1_pred))
print("\nClassification Report:\n", classification_report(y1_test, rf1_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y1_test, rf1_pred))
```

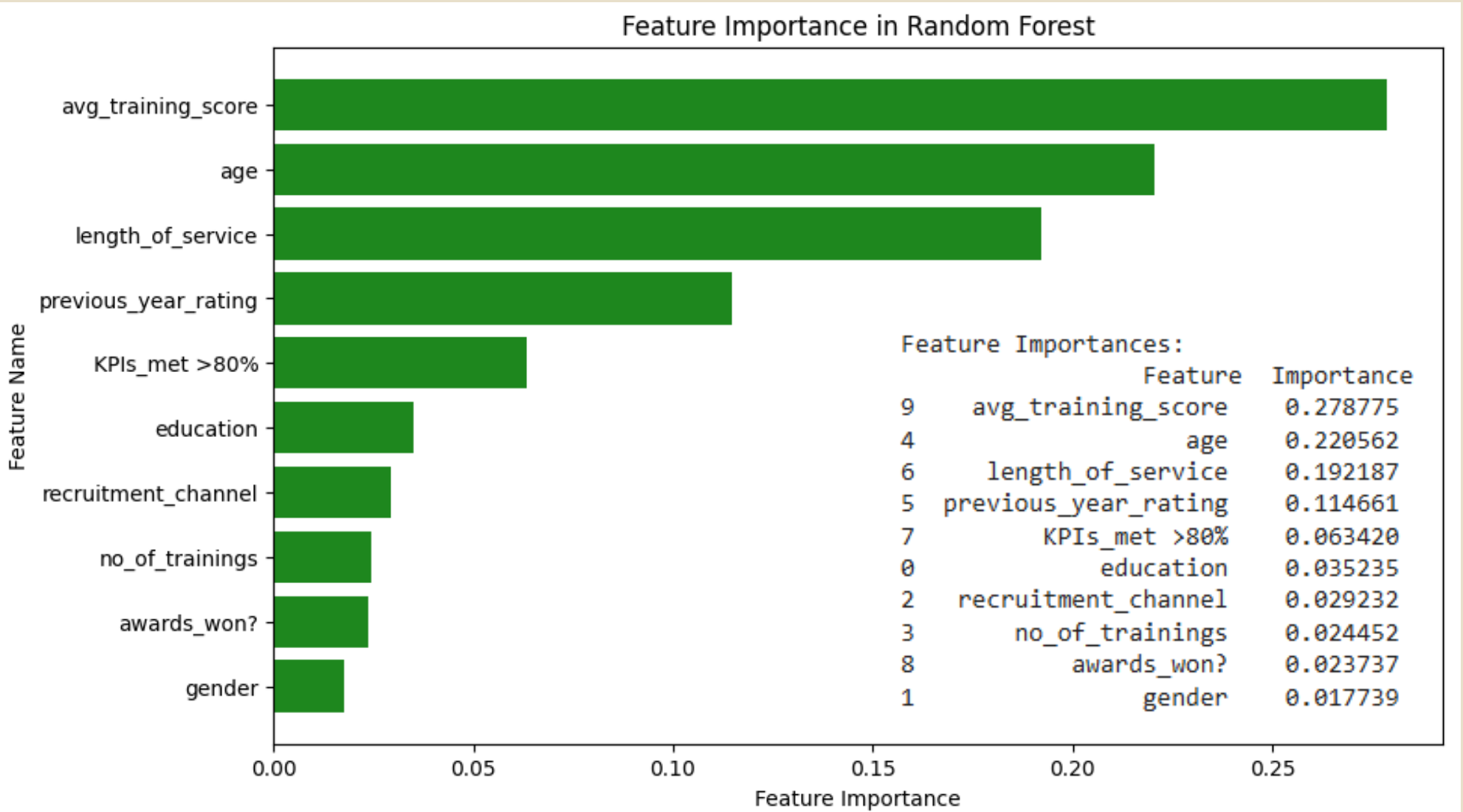
Accuracy Score: 0.8449885868437436

Classification Report:

	precision	recall	f1-score	support
0	0.88	0.94	0.91	12493
1	0.38	0.22	0.27	1964
accuracy			0.84	14457
macro avg	0.63	0.58	0.59	14457
weighted avg	0.82	0.84	0.83	14457

Confusion Matrix:

[[11791	702]
[1539	425]]



- The **Accuracy Score is 84.4%**, indicating that the model performs well overall
- Low **Precision for the "Promote" class (0.38)**, means that the model incorrectly promoted too many employees, leading to unrealistic expectations and possible unfair decision-making
- The most important features similar to R&D department include average training score, age, and length of service

4. PROMOTE PREDICTION - Random Forest

24

```
# Grid Search
rf1 = RandomForestClassifier()
grid_space = {'max_depth':[5, 10, 15, 20],
              'n_estimators':[10, 100, 10],
              'criterion':['gini','entropy']}
rf_grid1 = GridSearchCV(rf1, param_grid= grid_space, scoring='accuracy')
model_grid1 = rf_grid1.fit(X1_train_resampled, y1_train_resampled)
# Best hyperparameters
print('Best hyperparameters: '+str(model_grid1.best_params_))
print('Best score: '+str(model_grid1.best_score_))

Best hyperparameters: {'criterion': 'gini', 'max_depth': 20, 'n_estimators': 100}
Best score:0.7954926515890983

best_rf1 = RandomForestClassifier(
    criterion='gini',
    max_depth=10,
    n_estimators=30,
    random_state=42)
best_rf1.fit(X1_scaled_train, y1_train)
```

RandomForestClassifier

RandomForestClassifier(max_depth=10, n_estimators=30, random_state=42)

```
# Dự đoán trên tập test
rf1_best = best_rf1.predict(X1_scaled_test)

# Đánh giá hiệu suất
print("Accuracy Score:", accuracy_score(y1_test, rf1_best))
print("\nClassification Report:\n", classification_report(y1_test, rf1_best))
print("\nConfusion Matrix:\n", confusion_matrix(y1_test, rf1_best))
```

Accuracy Score: 0.8732793802310299

Classification Report:

	precision	recall	f1-score	support
0	0.88	0.99	0.93	12493
1	0.69	0.12	0.21	1964
accuracy			0.87	14457
macro avg	0.79	0.56	0.57	14457
weighted avg	0.85	0.87	0.83	14457

Confusion Matrix:

```
[[12388  105]
 [ 1727  237]]
```

- Using GridSearchCV, the model reached an **Accuracy Score: 87.3%**, which is higher than the previous model's 84.4% suggests that Grid Search optimization has improved the model's predictive performance.
 - The **Precision increased to 0.69**, means that the model makes fewer mistakes when predicting promotions, leading to more accurate and fair decision-making, making the model more reliable for promotion decisions, ensuring that promotions are given to deserving employees.
- => **Using GridSearchCV in Random Forest has significantly improved the model's performance**

4. PROMOTE PREDICTION - Gradient Boosting

25

```
# Gradient Boosting
gb_model1 = GradientBoostingClassifier()
gb_model1.fit(X1_train_resampled, y1_train_resampled)

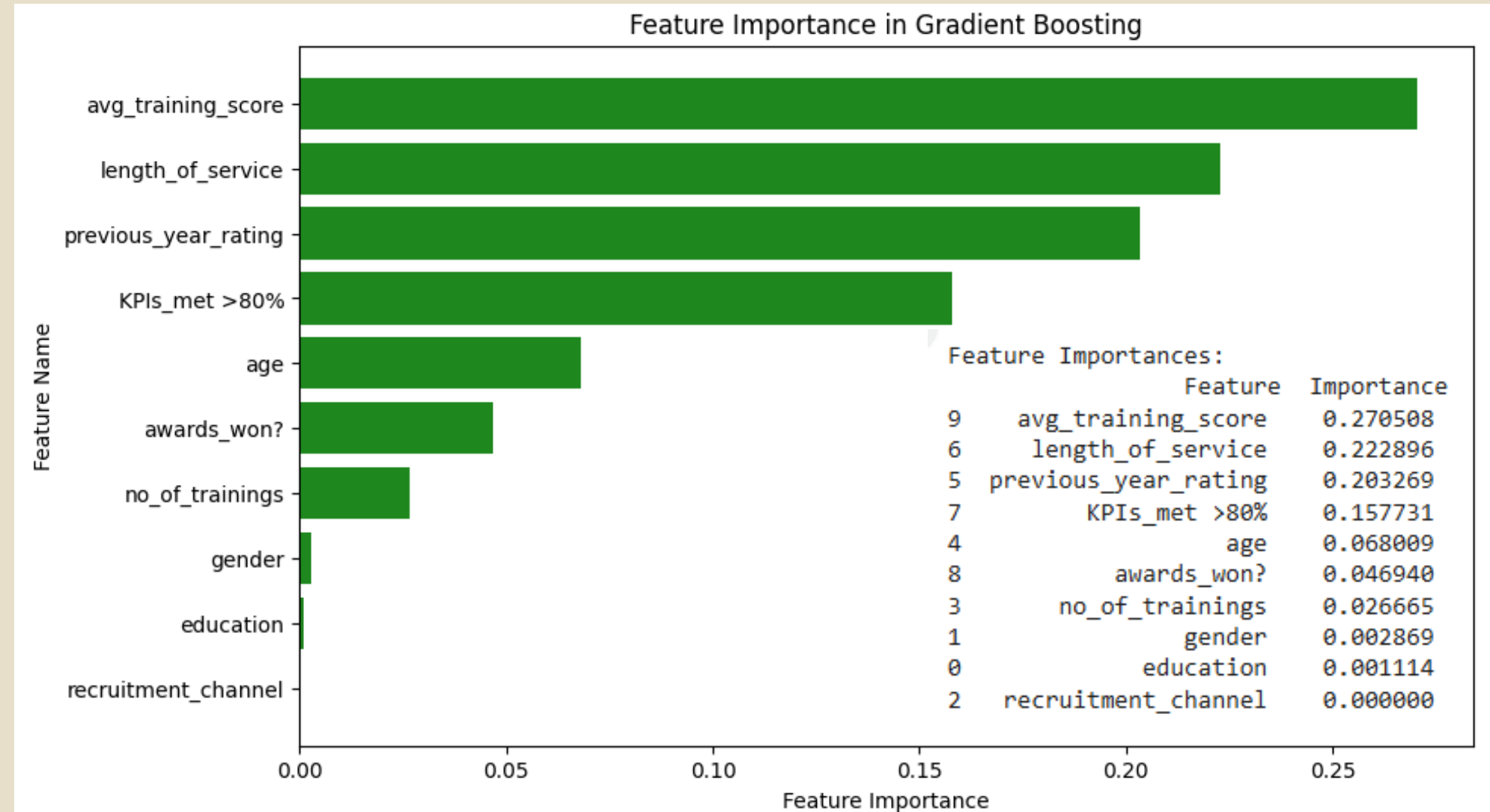
# Dự đoán trên tập test
gb1_pred = gb_model.predict(X1_scaled_test)

# Đánh giá mô hình
print("Accuracy Score:", accuracy_score(y1_test, gb1_pred))
print("\nClassification Report:\n", classification_report(y1_test, gb1_pred))
```

Accuracy Score: 0.2382236978626271

Classification Report:

	precision	recall	f1-score	support
0	0.89	0.14	0.24	12493
1	0.14	0.89	0.24	1964
accuracy			0.24	14457
macro avg	0.51	0.51	0.24	14457
weighted avg	0.78	0.24	0.24	14457



- The Gradient Boosting model performed very poorly in Sales & Marketing department.
- Very **Low Accuracy (23.8%)** => The model is making incorrect predictions most of the time.
- **Precison 0.14** => Only 14% of employees predicted for promotion were actually promoted, which means that the model is predicting a lot of employees who will be promoted but are predicting the promotion incorrectly
- But the **Recall 0.89** => The model correctly identifies most of the actual promotions

4. PROMOTE PREDICTION - Gradient Boosting

26

```
# Grid Search
gb = GradientBoostingClassifier()

parameters = {
    'loss': ['deviance', 'exponential'],
    'learning_rate': [0.001, 0.1, 1, 10],
    'n_estimators': [100, 150, 180, 200]
}

grid_search1 = GridSearchCV(gb, parameters, cv=5, n_jobs=-1, verbose=1)
grid_search1.fit(X1_train_resampled, y1_train_resampled)
print(grid_search1.best_params_)
```

- Using GridSearchCV, the Gradient Boosting model still performs poorly.
- **Accuracy is only 14.66%** => The model is incorrect most of the time, meaning it struggles to generalize well.
- **Precision 0.14** => Only 14% of predicted promotions were correct, which means that the model is predicting a lot of employees who will be promoted but are predicting the promotion incorrectly
- But **Recall 0.90** => The model correctly identifies most of the actual promotions, but this comes at the cost of high false-positive rates.

```
# Lấy bộ tham số tối ưu từ Grid Search
best_params = grid_search.best_params_

# Huấn luyện lại mô hình với tham số tối ưu và dữ liệu sau SMOTE
if 'colsample_bytree' in best_params:
    del best_params['colsample_bytree']
best_gb1 = GradientBoostingClassifier(**best_params)
best_gb1.fit(X1_train_resampled, y1_train_resampled)

# Dự đoán trên tập test
gb1_best = best_gb.predict(X1_scaled_test)

# Đánh giá mô hình
print("Accuracy Score:", accuracy_score(y1_test, gb1_best))
print("\nClassification Report:\n", classification_report(y1_test, gb1_best))
print("\nConfusion Matrix:\n", confusion_matrix(y1_test, gb1_best))
```

Accuracy Score: 0.16469530331327384

Classification Report:

	precision	recall	f1-score	support
0	0.86	0.04	0.08	12493
1	0.14	0.96	0.24	1964
accuracy			0.16	14457
macro avg	0.50	0.50	0.16	14457
weighted avg	0.76	0.16	0.10	14457

Confusion Matrix:

```
[[ 499 11994]
 [  82  1882]]
```

Similarities

- **Education:** Employees with "**Bachelor's**" and "**Master's & above**" degrees have higher promotion for both departments
- **Age:** Both departments have the age **peak around of 30** for both promoted and not-promoted employees => This suggests that individuals in their early 30s are highly active in their career progression or opportunities around that age are abundant
- **Length of service:** Most employees work for 2 years, then the number gradually decreases and drops sharply after 5 years
- **Recruitment:** The "**Other**" **recruitment channel** contributed the most promoted employees
- **Training:** Most of employees have completed 1 training course, employee with higher scores are tend to promote

Differences

- The significant difference between the two departments shows that training and job performance have a major impact on promotion opportunities
- R&D employees have a high promotion rate due to their capabilities, strong training, and high performance
- Sales & Marketing needs to improve training and job performance to increase promotion opportunities for employees



5. CONCLUSION

Kết quả dự đoán R&D Department:

	Model	Accuracy	Precision	Recall	F1-Score
0	Random Forest	0.800817	0.555556	0.357143	0.434783
1	Random Forest GridSearchCV	0.811032	0.610619	0.328571	0.427245
2	Gradient Boosting	0.814096	0.648936	0.290476	0.401316
3	Gradient Boosting GridSearchCV	0.816139	0.644231	0.319048	0.426752

Kết quả dự đoán Sales & Marketing Department:

	Model	Accuracy	Precision	Recall	F1-Score
0	Random Forest	0.844989	0.377107	0.216395	0.274992
1	Random Forest GridSearchCV	0.873279	0.692982	0.120672	0.205551
2	Gradient Boosting	0.238224	0.139223	0.889002	0.240745
3	Gradient Boosting GridSearchCV	0.164695	0.135630	0.958248	0.237626

- **R&D Department:** Models show relatively balanced performance, with Gradient Boosting GridSearchCV achieving the highest accuracy (0.816). However, precision do not differ significantly between the models
- **Sales & Marketing Department:** Performance is inconsistent. Random Forest GridSearchCV has the highest accuracy (0.873) and improve precision (0.69), but Gradient Boosting GridSearchCV achieves very high recall (0.958) at the cost of lower precision and accuracy

The model performs differently across departments



DA76

Vo Tran Hoang Chau



Thank You