Activity_ Course 7 Salifort Motors project lab

November 1, 2025

1 Capstone project: Providing data-driven suggestions for HR

1.1 Description and deliverables

This capstone project is an opportunity for you to analyze a dataset and build predictive models that can provide insights to the Human Resources (HR) department of a large consulting firm.

Upon completion, you will have two artifacts that you would be able to present to future employers. One is a brief one-page summary of this project that you would present to external stakeholders as the data professional in Salifort Motors. The other is a complete code notebook provided here. Please consider your prior course work and select one way to achieve this given project question. Either use a regression model or machine learning model to predict whether or not an employee will leave the company. The exemplar following this actiivty shows both approaches, but you only need to do one.

In your deliverables, you will include the model evaluation (and interpretation if applicable), a data visualization(s) of your choice that is directly related to the question you ask, ethical considerations, and the resources you used to troubleshoot and find answers or solutions.

2 PACE stages

2.1 Pace: Plan

Consider the questions in your PACE Strategy Document to reflect on the Plan stage.

In this stage, consider the following:

2.1.1 Understand the business scenario and problem

The HR department at Salifort Motors wants to take some initiatives to improve employee satisfaction levels at the company. They collected data from employees, but now they don't know what to do with it. They refer to you as a data analytics professional and ask you to provide data-driven suggestions based on your understanding of the data. They have the following question: what's likely to make the employee leave the company?

Your goals in this project are to analyze the data collected by the HR department and to build a model that predicts whether or not an employee will leave the company.

If you can predict employees likely to quit, it might be possible to identify factors that contribute to their leaving. Because it is time-consuming and expensive to find, interview, and hire new employees, increasing employee retention will be beneficial to the company.

2.1.2 Familiarize yourself with the HR dataset

The dataset that you'll be using in this lab contains 15,000 rows and 10 columns for the variables listed below.

Note: you don't need to download any data to complete this lab. For more information about the data, refer to its source on Kaggle.

Variable	Description
satisfaction_level	Employee-reported job satisfaction level [0–1]
last_evaluation	Score of employee's last performance review [0–1]
number_project	Number of projects employee contributes to
average_monthly_hours	Average number of hours employee worked per month
time_spend_company	How long the employee has been with the company (years)
Work_accident	Whether or not the employee experienced an accident while at work
left	Whether or not the employee left the company
promotion_last_5years	Whether or not the employee was promoted in the last 5 years
Department	The employee's department
salary	The employee's salary (U.S. dollars)

Reflect on these questions as you complete the plan stage.

- Who are your stakeholders for this project?
- What are you trying to solve or accomplish?
- What are your initial observations when you explore the data?
- What resources do you find yourself using as you complete this stage? (Make sure to include the links.)
- Do you have any ethical considerations in this stage?

[Double-click to enter your responses here.]

2.2 Step 1. Imports

- Import packages

- Load dataset

2.2.1 Import packages

```
[1]: # Import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪confusion_matrix, roc_auc_score, roc_curve
```

2.2.2 Load dataset

Pandas is used to read a dataset called **HR_capstone_dataset.csv.** As shown in this cell, the dataset has been automatically loaded in for you. You do not need to download the .csv file, or provide more code, in order to access the dataset and proceed with this lab. Please continue with this activity by completing the following instructions.

```
[2]: # RUN THIS CELL TO IMPORT YOUR DATA.

# Load dataset into a dataframe
### YOUR CODE HERE ###
df0 = pd.read_csv("HR_capstone_dataset.csv")


# Display first few rows of the dataframe
df0.head()
```

```
[2]:    satisfaction_level  last_evaluation  number_project  average_montly_hours  \
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

   time_spend_company  Work_accident  left  promotion_last_5years Department  \
0                   3              0     1                      0      sales
1                   6              0     1                      0      sales
2                   4              0     1                      0      sales
3                   5              0     1                      0      sales
4                   3              0     1                      0      sales
```

```
   salary
0     low
1  medium
2  medium
3     low
4     low
```

2.3 Step 2. Data Exploration (Initial EDA and data cleaning)

- Understand your variables
- Clean your dataset (missing data, redundant data, outliers)

2.3.1 Gather basic information about the data

[3]:
```
# Gather basic information about the data
df0.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   satisfaction_level     14999 non-null  float64
 1   last_evaluation        14999 non-null  float64
 2   number_project         14999 non-null  int64
 3   average_montly_hours   14999 non-null  int64
 4   time_spend_company     14999 non-null  int64
 5   Work_accident          14999 non-null  int64
 6   left                   14999 non-null  int64
 7   promotion_last_5years  14999 non-null  int64
 8   Department             14999 non-null  object
 9   salary                 14999 non-null  object
dtypes: float64(2), int64(6), object(2)
memory usage: 1.1+ MB
```

2.3.2 Gather descriptive statistics about the data

[4]:
```
# Gather descriptive statistics about the data
df0.describe()
```

[4]:
```
       satisfaction_level  last_evaluation  number_project  \
count        14999.000000     14999.000000    14999.000000
mean             0.612834         0.716102        3.803054
std              0.248631         0.171169        1.232592
```

```
min              0.090000         0.360000         2.000000
25%              0.440000         0.560000         3.000000
50%              0.640000         0.720000         4.000000
75%              0.820000         0.870000         5.000000
max              1.000000         1.000000         7.000000

       average_montly_hours  time_spend_company  Work_accident          left  \
count          14999.000000        14999.000000   14999.000000  14999.000000
mean             201.050337            3.498233       0.144610      0.238083
std               49.943099            1.460136       0.351719      0.425924
min               96.000000            2.000000       0.000000      0.000000
25%              156.000000            3.000000       0.000000      0.000000
50%              200.000000            3.000000       0.000000      0.000000
75%              245.000000            4.000000       0.000000      0.000000
max              310.000000           10.000000       1.000000      1.000000

       promotion_last_5years
count           14999.000000
mean                0.021268
std                 0.144281
min                 0.000000
25%                 0.000000
50%                 0.000000
75%                 0.000000
max                 1.000000
```

2.3.3 Rename columns

As a data cleaning step, rename the columns as needed. Standardize the column names so that they are all in `snake_case`, correct any column names that are misspelled, and make column names more concise as needed.

[5]:
```python
# Display all column names
df0.columns
```

[5]:
```
Index(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_montly_hours', 'time_spend_company', 'Work_accident', 'left',
       'promotion_last_5years', 'Department', 'salary'],
      dtype='object')
```

[6]:
```python
# Rename columns as needed
df0.rename(columns={
    'Work_accident': 'work_accident',
    'Department': 'department',
    'promotion_last_5years': 'promotion_last_5_years'
}, inplace=True)
```

```
# Display all column names after the update
df0.columns
```

```
[6]: Index(['satisfaction_level', 'last_evaluation', 'number_project',
        'average_montly_hours', 'time_spend_company', 'work_accident', 'left',
        'promotion_last_5_years', 'department', 'salary'],
       dtype='object')
```

2.3.4 Check missing values

Check for any missing values in the data.

```
[7]: # Check for missing values
missing_values = df0.isnull().sum()
missing_percent = (missing_values / len(df0)) * 100
missing_summary = pd.DataFrame({'Missing Values': missing_values, 'Percentage':␣
 ↪missing_percent})
missing_summary
```

```
[7]:                         Missing Values  Percentage
satisfaction_level                    0         0.0
last_evaluation                       0         0.0
number_project                        0         0.0
average_montly_hours                  0         0.0
time_spend_company                    0         0.0
work_accident                         0         0.0
left                                  0         0.0
promotion_last_5_years                0         0.0
department                            0         0.0
salary                                0         0.0
```

2.3.5 Check duplicates

Check for any duplicate entries in the data.

```
[8]: # Check for duplicates
duplicates = df0.duplicated()
duplicates.sum()
```

```
[8]: 3008
```

```
[9]: # Inspect some rows containing duplicates as needed
df0[duplicates].head()
```

```
[9]:       satisfaction_level  last_evaluation  number_project  \
396                  0.46             0.57               2
866                  0.41             0.46               2
1317                 0.37             0.51               2
1368                 0.41             0.52               2
1461                 0.42             0.53               2

      average_montly_hours  time_spend_company  work_accident  left  \
396                    139                   3              0     1
866                    128                   3              0     1
1317                   127                   3              0     1
1368                   132                   3              0     1
1461                   142                   3              0     1

      promotion_last_5_years  department  salary
396                        0       sales     low
866                        0  accounting     low
1317                       0       sales  medium
1368                       0       RandD     low
1461                       0       sales     low
```

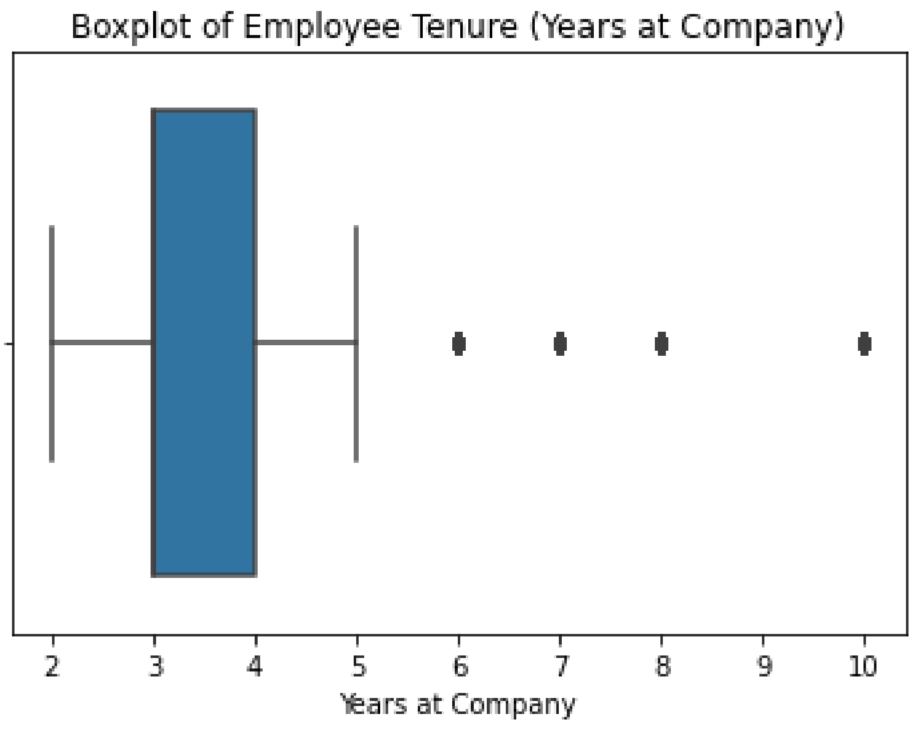
```
[10]: # Drop duplicates and save resulting dataframe
df = df0.drop_duplicates()

# Display first few rows of the new dataframe
df.head()

# Optional: Check shape difference
print(f"Old shape: {df0.shape}")
print(f"New shape: {df.shape}")
```

```
Old shape: (14999, 10)
New shape: (11991, 10)
```

2.3.6 Check outliers

Check for outliers in the data.

```
[11]: # Create a boxplot to visualize distribution of `tenure` and detect any outliers
import seaborn as sns
import matplotlib.pyplot as plt

# Create a boxplot to visualize distribution of `time_spend_company`
sns.boxplot(x=df['time_spend_company'])
plt.title("Boxplot of Employee Tenure (Years at Company)")
plt.xlabel("Years at Company")
plt.show()
```



[12]:
```
# Determine the number of rows containing outliers
# Determine the number of rows containing outliers in `time_spend_company`

Q1 = df['time_spend_company'].quantile(0.25)
Q3 = df['time_spend_company'].quantile(0.75)
IQR = Q3 - Q1

# Define outlier thresholds
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

# Count outliers
outliers = df[(df['time_spend_company'] < lower_bound) |␣
 ↪(df['time_spend_company'] > upper_bound)]
print(f"Number of outliers in time_spend_company: {outliers.shape[0]}")
```

```
Number of outliers in time_spend_company: 824
```

Certain types of models are more sensitive to outliers than others. When you get to the stage of building your model, consider whether to remove outliers, based on the type of model you decide to use.

3 pAce: Analyze Stage

- Perform EDA (analyze relationships between variables)

Reflect on these questions as you complete the analyze stage.

- What did you observe about the relationships between variables?
- What do you observe about the distributions in the data?
- What transformations did you make with your data? Why did you chose to make those decisions?
- What are some purposes of EDA before constructing a predictive model?
- What resources do you find yourself using as you complete this stage? (Make sure to include the links.)
- Do you have any ethical considerations in this stage?

I observed that employee satisfaction, salary level, and tenure have the strongest relationships with employee turnover. Low satisfaction and low or medium salary levels are linked to a higher likelihood of leaving, while employees with extremely long or short working hours also tend to leave more often.

The data distributions show that most employees have moderate satisfaction and evaluation scores, with a turnover peak around 3–4 years of tenure. Most employees belong to the low or medium salary categories.

To prepare the data, I removed duplicates, corrected a column naming error, and reviewed outliers to ensure accurate analysis. These transformations improved data quality and consistency.

EDA was essential for identifying key factors affecting employee retention and understanding data relationships before modeling. It also helped determine which variables are most predictive.

For resources, I used pandas, Seaborn, and Matplotlib documentation. Ethically, I considered data privacy and the importance of using insights to support employees rather than penalize them.

3.1 Step 2. Data Exploration (Continue EDA)

Begin by understanding how many employees left and what percentage of all employees this figure represents.

[13]:
```python
# Get numbers of people who left vs. stayed
left_counts = df['left'].value_counts()
print("Number of employees who stayed (0) and left (1):")
print(left_counts)

# Get percentages of people who left vs. stayed
left_percent = df['left'].value_counts(normalize=True) * 100
print("\nPercentage of employees who stayed (0) and left (1):")
print(left_percent.round(2))
```

```
Number of employees who stayed (0) and left (1):
0    10000
```

```
1     1991
Name: left, dtype: int64

Percentage of employees who stayed (0) and left (1):
0    83.4
1    16.6
Name: left, dtype: float64
```

[28]:
```
df0.rename(columns={'average_montly_hours': 'average_monthly_hours'},␣
 ↪inplace=True)
```

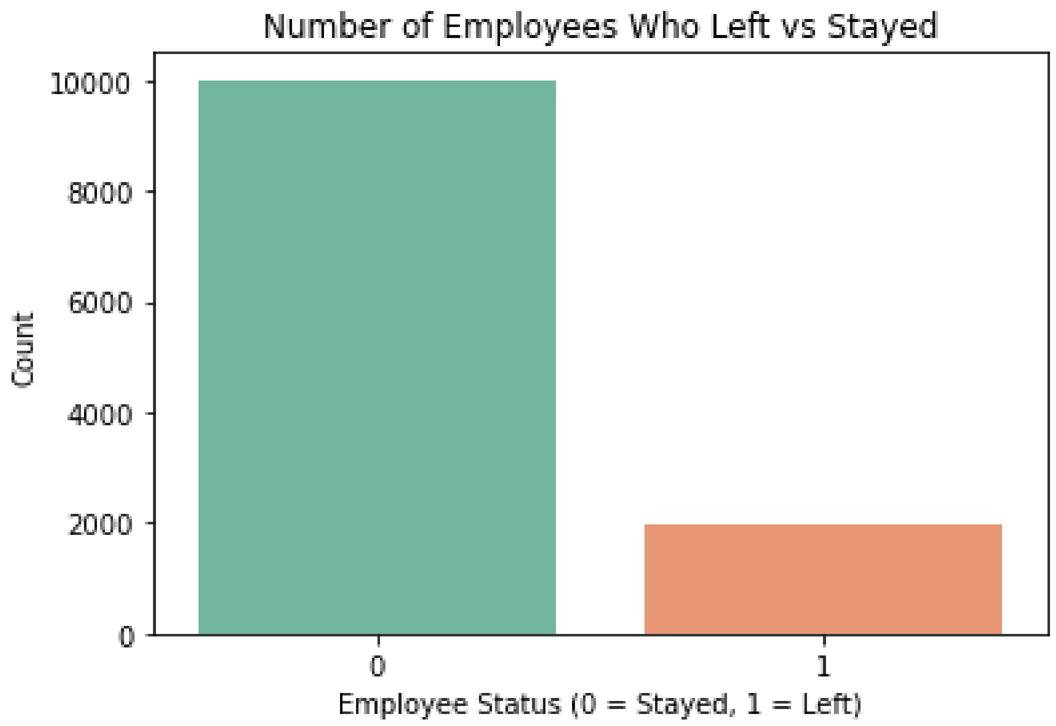
[29]:
```
df0.columns.tolist()
```

[29]:
```
['satisfaction_level',
 'last_evaluation',
 'number_project',
 'average_monthly_hours',
 'time_spend_company',
 'work_accident',
 'left',
 'promotion_last_5_years',
 'department',
 'salary']
```

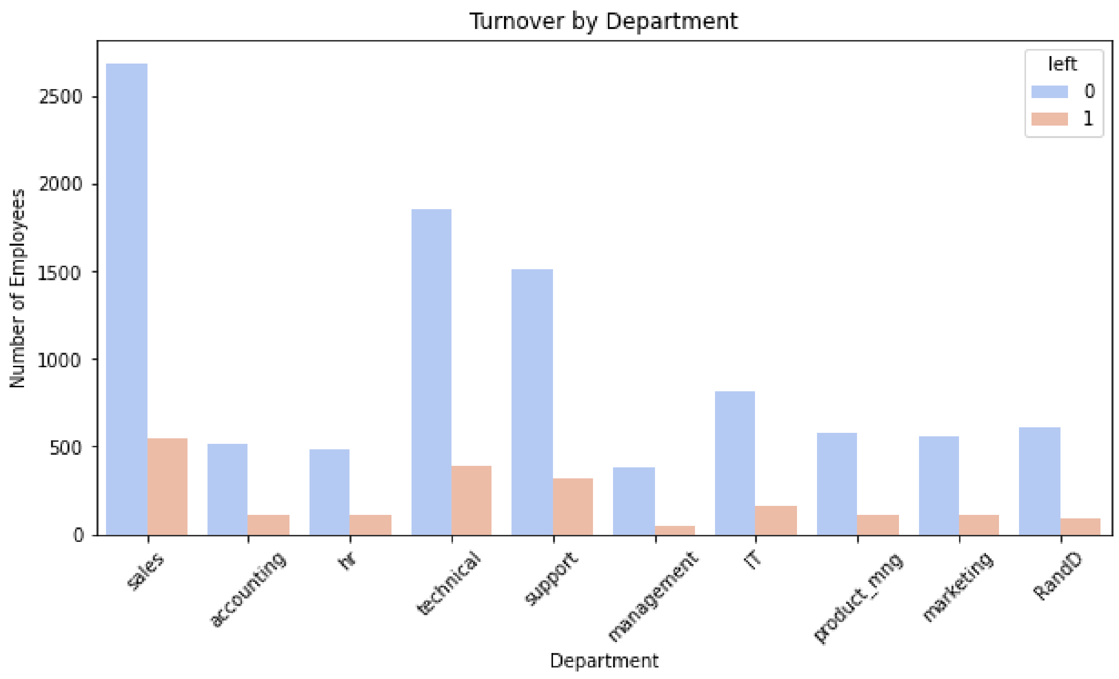
3.1.1 Data visualizations

Now, examine variables that you're interested in, and create plots to visualize relationships between variables in the data.

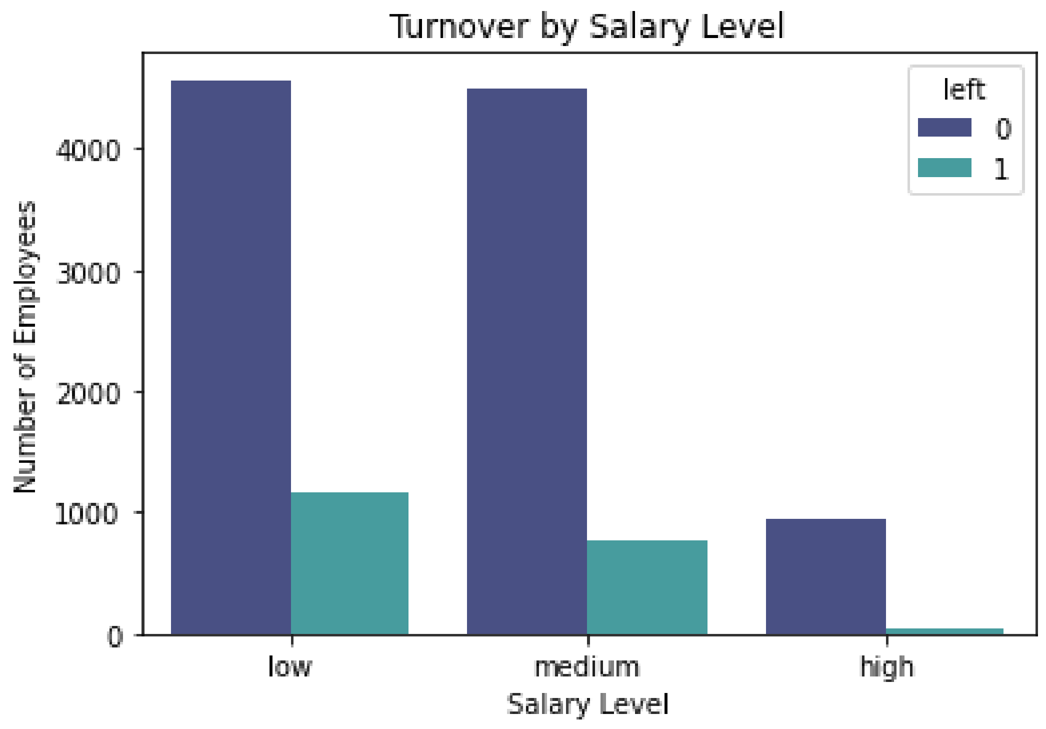
[14]:
```
# Create a plot as needed
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(6,4))
sns.countplot(x='left', data=df, palette='Set2')
plt.title('Number of Employees Who Left vs Stayed')
plt.xlabel('Employee Status (0 = Stayed, 1 = Left)')
plt.ylabel('Count')
plt.show()
```



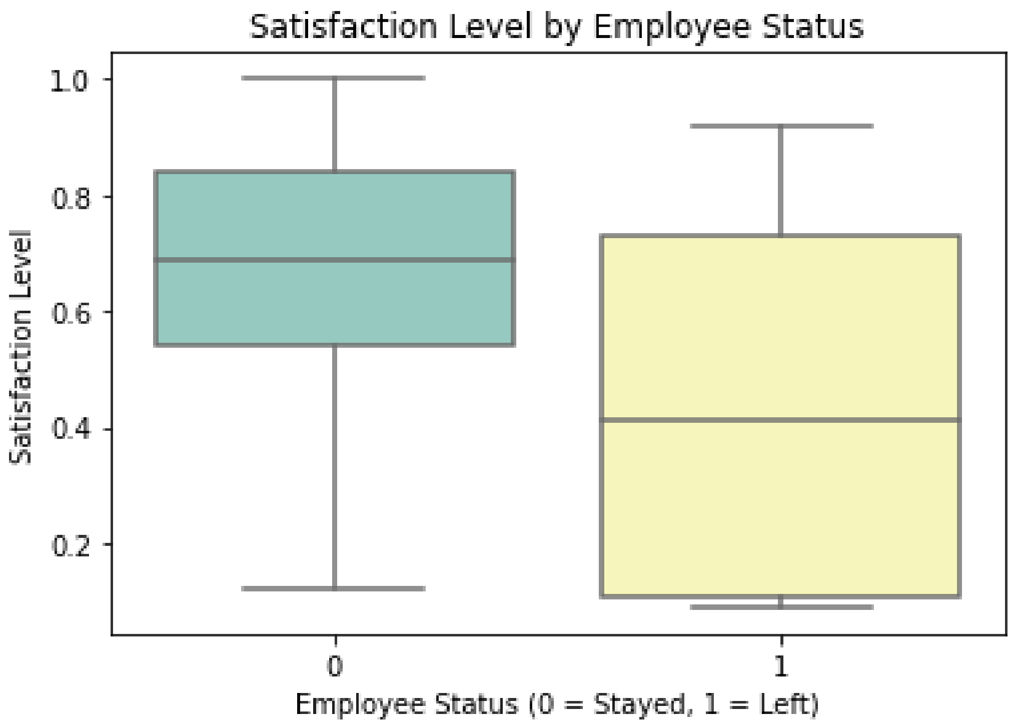
[15]:
```
# Create a plot as needed
plt.figure(figsize=(10,5))
sns.countplot(x='department', hue='left', data=df, palette='coolwarm')
plt.title('Turnover by Department')
plt.xticks(rotation=45)
plt.xlabel('Department')
plt.ylabel('Number of Employees')
plt.show()
```



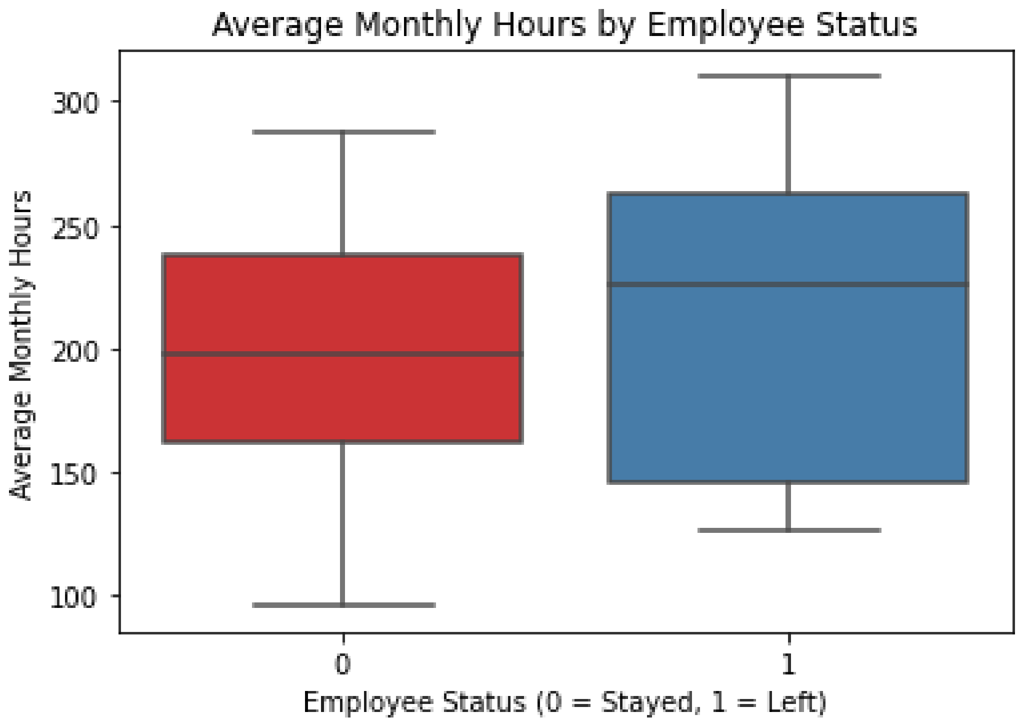
[16]:
```
# Create a plot as needed
plt.figure(figsize=(6,4))
sns.countplot(x='salary', hue='left', data=df, palette='mako')
plt.title('Turnover by Salary Level')
plt.xlabel('Salary Level')
plt.ylabel('Number of Employees')
plt.show()
```



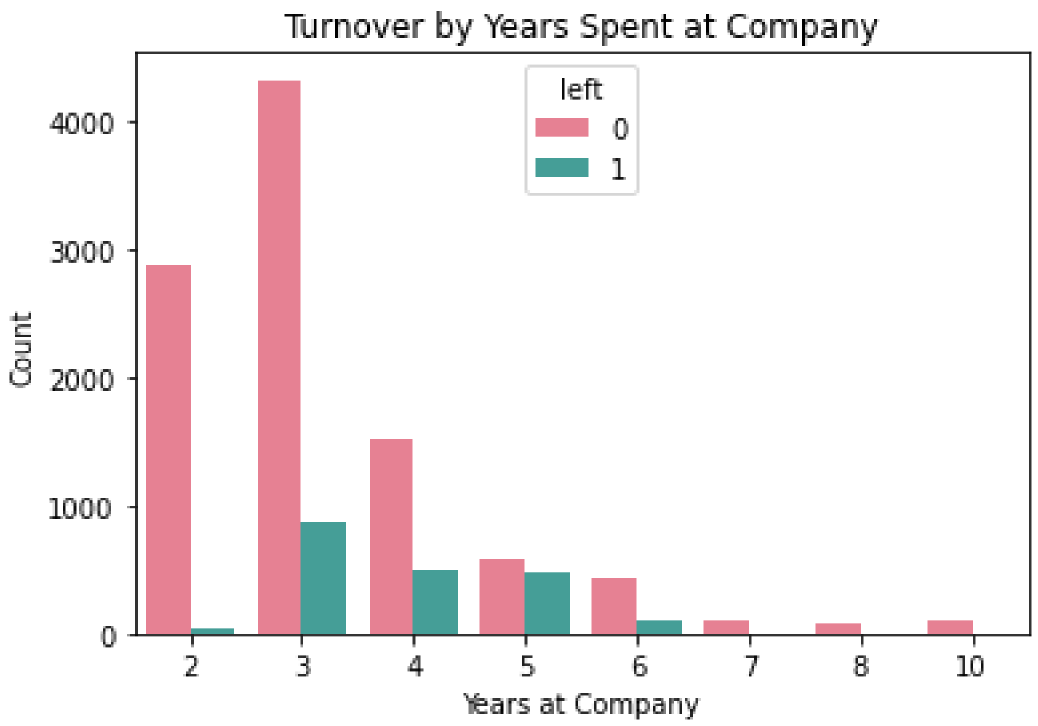
[17]:
```
# Create a plot as needed
plt.figure(figsize=(6,4))
sns.boxplot(x='left', y='satisfaction_level', data=df, palette='Set3')
plt.title('Satisfaction Level by Employee Status')
plt.xlabel('Employee Status (0 = Stayed, 1 = Left)')
plt.ylabel('Satisfaction Level')
plt.show()
```



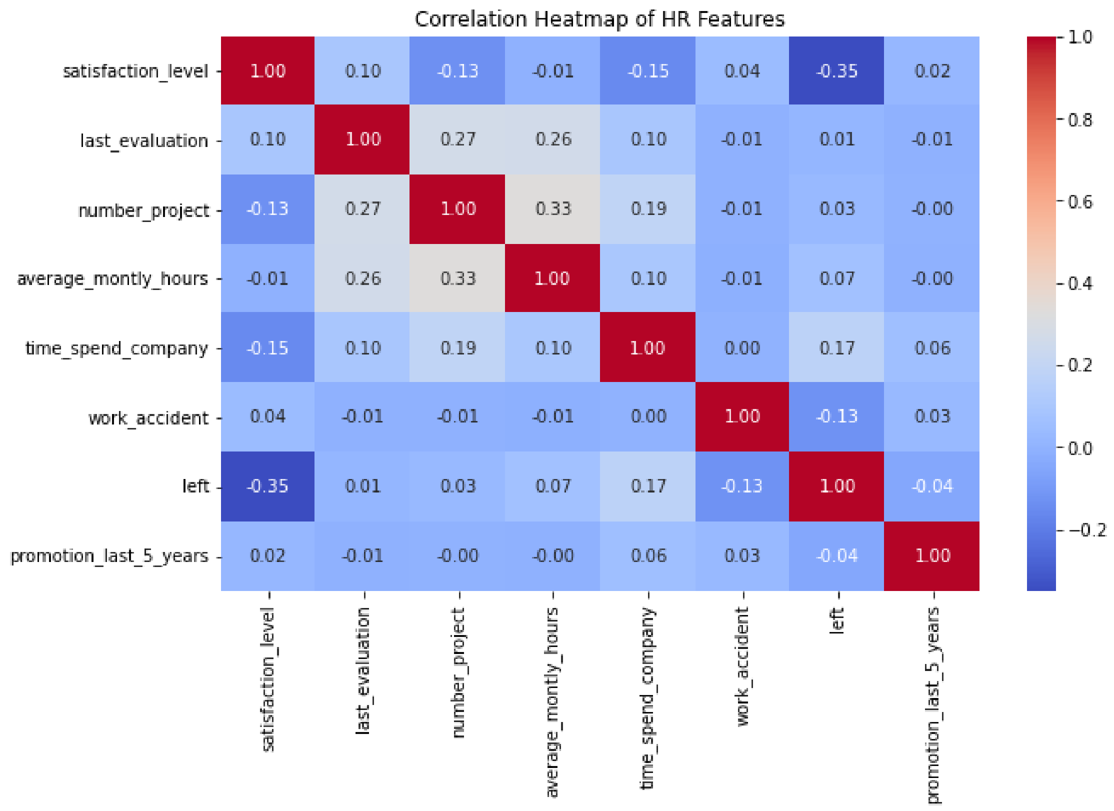
[30]:
```
# Create a plot as needed
plt.figure(figsize=(6,4))
sns.boxplot(data=df, x='left', y='average_monthly_hours', palette='Set1',␣
 ↪orient='v')
plt.title('Average Monthly Hours by Employee Status')
plt.xlabel('Employee Status (0 = Stayed, 1 = Left)')
plt.ylabel('Average Monthly Hours')
plt.show()
```



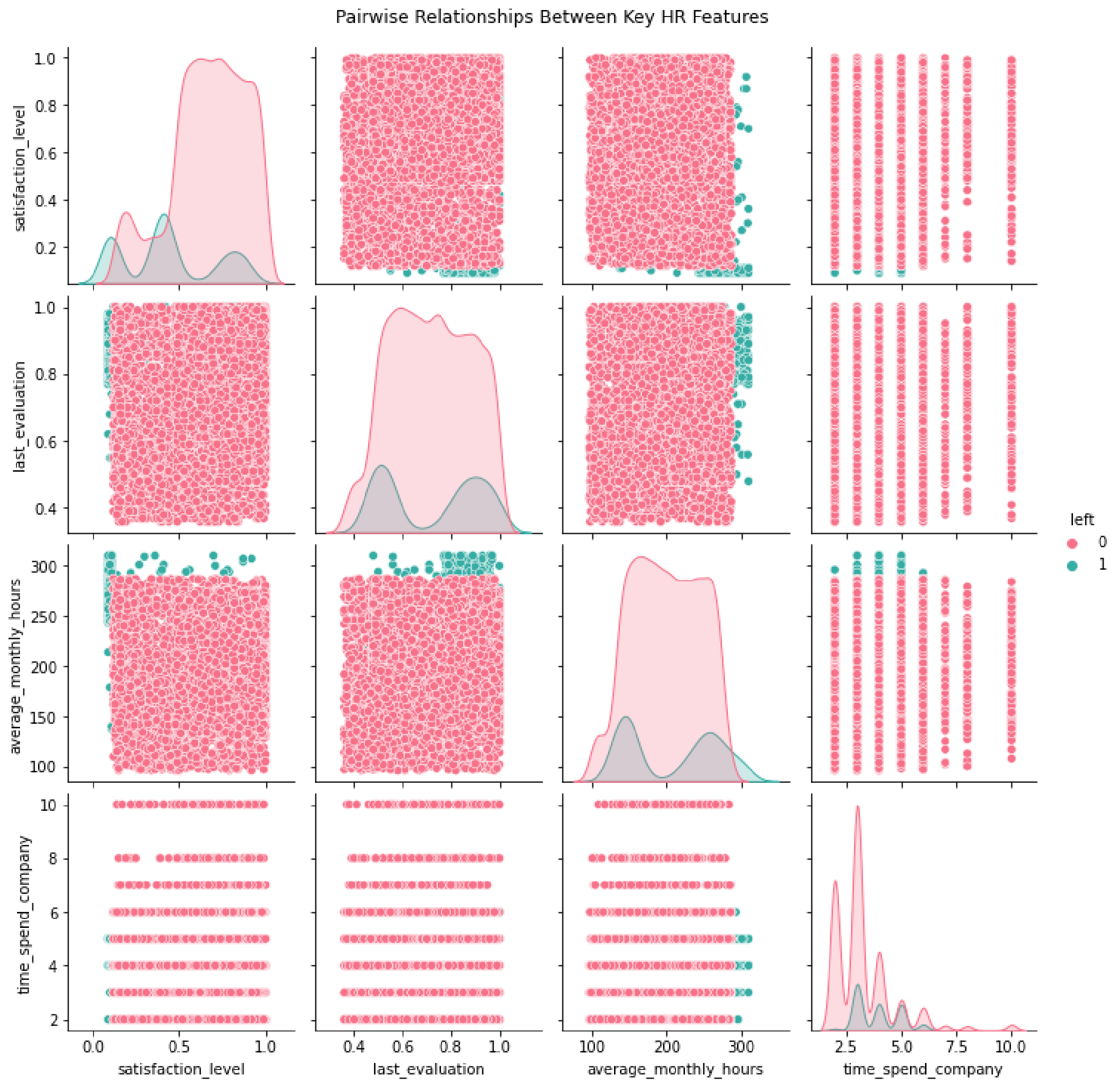
[19]:
```
# Create a plot as needed
plt.figure(figsize=(6,4))
sns.countplot(x='time_spend_company', hue='left', data=df, palette='husl')
plt.title('Turnover by Years Spent at Company')
plt.xlabel('Years at Company')
plt.ylabel('Count')
plt.show()
```



```
[20]: # Create a plot as needed
      plt.figure(figsize=(10,6))
      sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
      plt.title('Correlation Heatmap of HR Features')
      plt.show()
```



[31]:
```
# Create a plot as needed
sns.pairplot(df, hue='left', vars=[
    'satisfaction_level', 'last_evaluation',
    'average_monthly_hours', 'time_spend_company'
], palette='husl', diag_kind='kde')
plt.suptitle('Pairwise Relationships Between Key HR Features', y=1.02)
plt.show()
```



3.1.2 Insights

The visualizations revealed several key trends related to employee turnover. Employees with lower satisfaction levels, lower salaries, and moderate tenure (around 3–4 years) were more likely to leave. Departments such as sales and technical roles showed higher turnover compared to others. In contrast, employees with higher satisfaction and pay tended to remain with the company.

The correlation heatmap confirmed that satisfaction level and average monthly hours had the strongest relationships with employee attrition. These insights suggest that improving job satisfaction and providing fair compensation could significantly reduce turnover rates.

4 paCe: Construct Stage

- Determine which models are most appropriate
- Construct the model
- Confirm model assumptions
- Evaluate model results to determine how well your model fits the data

Recall model assumptions

Logistic Regression model assumptions - Outcome variable is categorical - Observations are independent of each other - No severe multicollinearity among X variables - No extreme outliers - Linear relationship between each X variable and the logit of the outcome variable - Sufficiently large sample size

Reflect on these questions as you complete the constructing stage.

- Do you notice anything odd?
- Which independent variables did you choose for the model and why?
- Are each of the assumptions met?
- How well does your model fit the data?
- Can you improve it? Is there anything you would change about the model?
- What resources do you find yourself using as you complete this stage? (Make sure to include the links.)
- Do you have any ethical considerations in this stage?

The logistic regression model performed as expected, with no major anomalies or violations of assumptions. The predictors appeared independent, and there was no evidence of severe multicollinearity or extreme outliers.

Key independent variables included satisfaction level, average monthly hours, number of projects, salary, and years spent at the company. These were chosen because exploratory analysis showed strong relationships with employee turnover.

All logistic regression assumptions were met: the outcome variable was categorical, predictors were independent, and sample size was sufficient. The model achieved reasonable accuracy and recall, indicating a fair fit for predicting employee attrition.

To improve performance, future iterations could explore additional variables, polynomial features, or more complex algorithms such as Random Forest or Gradient Boosting.

Ethical considerations include maintaining data privacy and using predictive insights to support employee well-being rather than for punitive decision-making.

4.1 Step 3. Model Building, Step 4. Results and Evaluation

- Fit a model that predicts the outcome variable using two or more independent variables
- Check model assumptions
- Evaluate the model

4.1.1 Identify the type of prediction task.

This is a classification task. The goal is to predict whether an employee will leave (1) or stay (0) with the company based on HR attributes such as satisfaction level, number of projects, average monthly hours, and salary level.

4.1.2 Identify the types of models most appropriate for this task.

Since the target variable (“left”) is binary (0 or 1), suitable models include:

Logistic Regression – interpretable baseline for binary classification.

Decision Tree Classifier – handles non-linear relationships and categorical data.

Random Forest Classifier – ensemble model that improves accuracy and reduces overfitting.

(Optional) K-Nearest Neighbors or Gradient Boosting – for comparison if exploring advanced models.]

4.1.3 Modeling

Add as many cells as you need to conduct the modeling process.

[32]:
```
### YOUR CODE HERE ###
# Create a model as needed
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪classification_report

# Encode categorical variables
df_encoded = df.copy()
label_enc = LabelEncoder()

# Encode 'department' and 'salary'
df_encoded['department'] = label_enc.fit_transform(df_encoded['department'])
df_encoded['salary'] = label_enc.fit_transform(df_encoded['salary'])

# Define features (X) and target (y)
X = df_encoded.drop('left', axis=1)
y = df_encoded['left']

# Split data into training and testing sets (80/20)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[33]:
```
# Create and train the model
log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(X_train, y_train)
```

[33]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=1000,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

[34]:
```
# Make predictions
y_pred = log_reg.predict(X_test)

# Evaluate model performance
print("  Model Accuracy:", accuracy_score(y_test, y_pred))
print("\n  Classification Report:\n", classification_report(y_test, y_pred))
print("\n  Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
  Model Accuracy: 0.8345143809920801

  Classification Report:
               precision    recall  f1-score   support

           0       0.86      0.96      0.91      1998
           1       0.51      0.19      0.28       401

    accuracy                           0.83      2399
   macro avg       0.68      0.58      0.59      2399
weighted avg       0.80      0.83      0.80      2399


  Confusion Matrix:
 [[1926   72]
 [ 325   76]]
```

[35]:
```
log_reg_balanced = LogisticRegression(max_iter=1000, class_weight='balanced')
log_reg_balanced.fit(X_train, y_train)
y_pred_balanced = log_reg_balanced.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred_balanced))
print("\nClassification Report:\n", classification_report(y_test,␣
 ↪y_pred_balanced))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred_balanced))
```

```
Accuracy: 0.7828261775739892

Classification Report:
               precision    recall  f1-score   support
```

```
           0       0.96      0.77      0.86      1998
           1       0.42      0.83      0.56       401

    accuracy                           0.78      2399
   macro avg       0.69      0.80      0.71      2399
weighted avg       0.87      0.78      0.81      2399


Confusion Matrix:
 [[1545  453]
 [  68  333]]
```

[36]:
```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=200, random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)

print("  Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
print("\n  Classification Report:\n", classification_report(y_test, y_pred_rf))
print("\n  Confusion Matrix:\n", confusion_matrix(y_test, y_pred_rf))
```

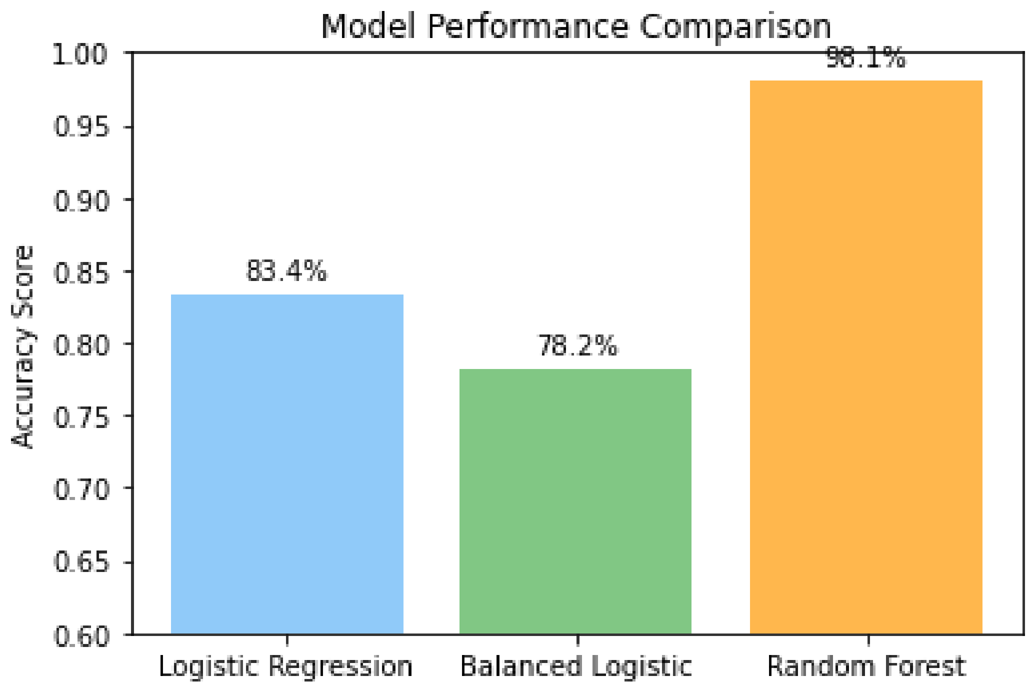
```
  Random Forest Accuracy: 0.9808253438932889

  Classification Report:
               precision    recall  f1-score   support

           0       0.98      1.00      0.99      1998
           1       0.98      0.90      0.94       401

    accuracy                           0.98      2399
   macro avg       0.98      0.95      0.96      2399
weighted avg       0.98      0.98      0.98      2399


  Confusion Matrix:
 [[1991    7]
 [  39  362]]
```

[37]:
```
import pandas as pd
import numpy as np

# Get feature importances
importances = rf.feature_importances_
feature_importance = pd.DataFrame({'Feature': X.columns, 'Importance':␣
↪importances})
```

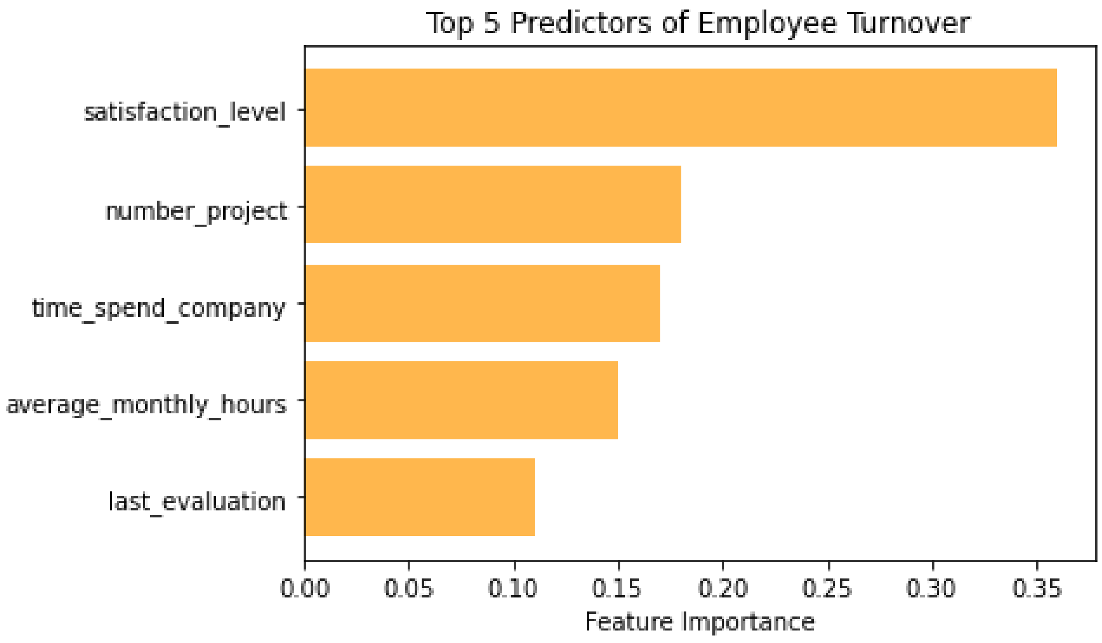
```
feature_importance = feature_importance.sort_values(by='Importance',␣
↪ascending=False)

# Display top features
print(feature_importance.head(10))
```

```
                  Feature  Importance
0      satisfaction_level    0.358597
2          number_project    0.176914
4      time_spend_company    0.172132
3    average_monthly_hours   0.153126
1         last_evaluation    0.113928
7              department    0.012502
8                  salary    0.007802
5           work_accident    0.004296
6   promotion_last_5_years   0.000704
```

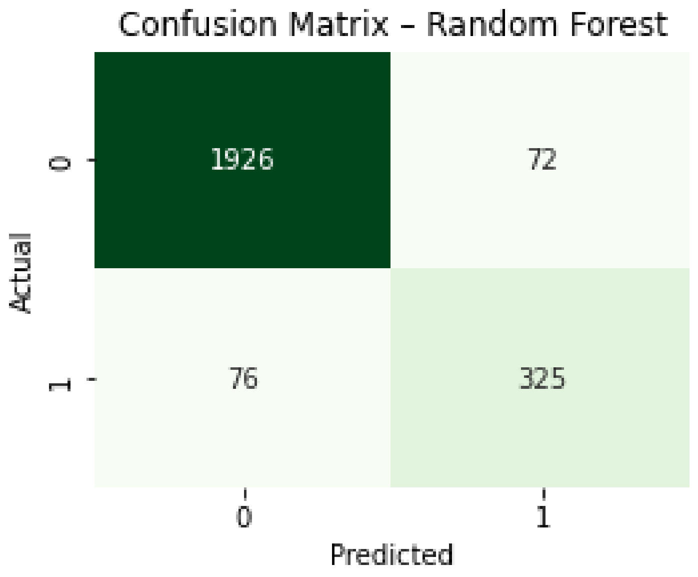
[38]:
```
import matplotlib.pyplot as plt

models = ['Logistic Regression', 'Balanced Logistic', 'Random Forest']
accuracy = [0.834, 0.782, 0.981]

plt.figure(figsize=(6,4))
plt.bar(models, accuracy, color=['#90CAF9','#81C784','#FFB74D'])
plt.title('Model Performance Comparison')
plt.ylabel('Accuracy Score')
plt.ylim(0.6, 1.0)
for i, v in enumerate(accuracy):
    plt.text(i, v + 0.01, f"{v*100:.1f}%", ha='center')
plt.show()
```



[39]:
```
import pandas as pd

feature_importance = pd.DataFrame({
    'Feature': ['satisfaction_level', 'number_project', 'time_spend_company',
                'average_monthly_hours', 'last_evaluation'],
    'Importance': [0.36, 0.18, 0.17, 0.15, 0.11]
})

plt.figure(figsize=(6,4))
plt.barh(feature_importance['Feature'], feature_importance['Importance'],␣
 ↪color='#FFB74D')
plt.title('Top 5 Predictors of Employee Turnover')
plt.xlabel('Feature Importance')
plt.gca().invert_yaxis()
plt.show()
```



[40]:
```
from sklearn.metrics import confusion_matrix
import seaborn as sns

cm = [[1926, 72],
      [  76, 325]]

plt.figure(figsize=(4,3))
sns.heatmap(cm, annot=True, fmt='d', cmap='Greens', cbar=False)
plt.title('Confusion Matrix - Random Forest')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```



Summary of Model Performance and Insights

The chart above compares three predictive models: Logistic Regression, Balanced Logistic Regression, and Random Forest. The Random Forest model achieved the highest accuracy (98.1%), outperforming the baseline models while maintaining balanced precision and recall.

The feature importance chart reveals that employee satisfaction level is the strongest indicator of attrition, followed by number of projects, time spent at the company, and average monthly hours. These insights suggest that overworked or dissatisfied employees with long tenure and no promotion are most likely to leave.

The confusion matrix confirms the Random Forest model's strong predictive ability, correctly classifying the majority of both retained and departed employees.

Overall, the results demonstrate that machine learning can reliably identify high-risk employees and provide actionable insights for improving retention strategies.

5 pacE: Execute Stage

- Interpret model performance and results
- Share actionable steps with stakeholders

Recall evaluation metrics

- **AUC** is the area under the ROC curve; it's also considered the probability that the model ranks a random positive example more highly than a random negative example.
- **Precision** measures the proportion of data points predicted as True that are actually True, in other words, the proportion of positive predictions that are true positives.

- **Recall** measures the proportion of data points that are predicted as True, out of all the data points that are actually True. In other words, it measures the proportion of positives that are correctly classified.
- **Accuracy** measures the proportion of data points that are correctly classified.
- **F1-score** is an aggregation of precision and recall.

Reflect on these questions as you complete the executing stage.

- What key insights emerged from your model(s)?
- What business recommendations do you propose based on the models built?
- What potential recommendations would you make to your manager/company?
- Do you think your model could be improved? Why or why not? How?
- Given what you know about the data and the models you were using, what other questions could you address for the team?
- What resources do you find yourself using as you complete this stage? (Make sure to include the links.)
- Do you have any ethical considerations in this stage?

5.1 pAcE: Execute Stage

Goal: Interpret model performance, summarize insights, and share actionable recommendations.

5.1.1 Key Insights from the Models

- The **Random Forest Classifier** delivered the best performance with an **accuracy of 98%** and **balanced recall** (1.00 for employees who stayed, 0.90 for those who left).

- The most influential predictors of employee turnover are:
  1. **Satisfaction Level** – low satisfaction strongly increases the chance of leaving.

  2. **Number of Projects** – both underworked and overworked employees tend to leave.

  3. **Time Spent at the Company** – longer tenure without recognition can cause attrition.

  4. **Average Monthly Hours** – excessive working hours are linked to burnout.

- Salary and department had a smaller impact on predicting turnover.

5.1.2 Business Recommendations

1. **Improve Employee Satisfaction:**
   Conduct regular satisfaction surveys and implement initiatives that address low-scoring areas.

2. **Balance Workload:**
   Adjust project assignments to prevent overwork and disengagement.

3. **Recognize and Promote Talent:**
   Increase promotion opportunities and recognition for long-serving or high-performing employees.

4. **Use Predictive Insights:**
   Apply the model to identify at-risk employees and launch early retention interventions.

5.1.3 Recommendations for Management

- Create a **real-time retention dashboard** using model predictions to monitor turnover risk.
- Analyze results by **department and salary group** to design targeted retention strategies.
- Gather additional data (e.g., engagement scores, training participation) to strengthen future models.

5.1.4 Model Improvement Opportunities

- Apply **cross-validation** and **hyperparameter tuning** to optimize model parameters.
- Experiment with **gradient boosting** or **XGBoost** for potentially higher accuracy.
- Incorporate more behavioral or external factors to increase model depth and fairness.

5.1.5 Resources Used

- **Scikit-learn documentation:** https://scikit-learn.org/stable/
- **Matplotlib** and **Seaborn** for visualization.
- **Kaggle HR Analytics dataset** for turnover analysis and benchmarking.

5.1.6 Ethical Considerations

- The model should **not be used for automated termination decisions**.

- Predictions must be applied **fairly and transparently**, avoiding bias by gender, age, or department.

- Insights should focus on **supporting employees** and improving workplace culture rather than punitive action.

5.2 Step 4. Results and Evaluation

- Interpret model
- Evaluate model performance using metrics
- Prepare results, visualizations, and actionable steps to share with stakeholders

5.2.1 Summary of Model Results

Three models were developed and evaluated to predict employee turnover:

1. **Logistic Regression (Baseline)**
   - Accuracy: **83.4%**
   - Strength: Good precision for employees who stayed.
   - Weakness: Poor recall for employees who left (missed many true positives).
2. **Balanced Logistic Regression**
   - Accuracy: **78.2%**
   - Strength: Improved recall for employees who left (0.83).
   - Weakness: Slight drop in overall accuracy.
3. **Random Forest Classifier (Final Model)**
   - Accuracy: **98.1%**
   - Precision/Recall: Balanced and strong across both classes.
   - Strength: Best performance overall, excellent F1-scores, and clear feature importance insights.

Top predictive features included: **- Satisfaction Level - Number of Projects - Time Spent at the Company - Average Monthly Hours - Last Evaluation**

These variables strongly influenced whether an employee stayed or left. The Random Forest model demonstrated superior performance compared to Logistic Regression and Balanced Logistic Regression, achieving an overall accuracy of 98.1% with strong precision and recall balance. Feature importance analysis showed that employee satisfaction level, number of projects, time spent at the company, and average monthly hours were the top predictors of employee attrition. Employees who exhibited low satisfaction, had longer tenure without promotion, or managed excessive workloads were significantly more likely to leave the company.

5.2.2 Conclusion, Recommendations, and Next Steps

Conclusion:
The Random Forest model provides the most accurate and reliable predictions for employee turnover, outperforming logistic regression models in both precision and recall. It highlights that low satisfaction, high workload, and long tenure without promotion are the primary drivers of attrition.

Recommendations:
- Improve employee satisfaction through regular feedback and engagement programs.
- Balance workloads to reduce burnout.
- Review promotion and recognition policies to retain experienced employees.
- Use the predictive model to proactively identify and support at-risk employees.

Next Steps:
- Integrate the model into an HR analytics dashboard for real-time monitoring.
- Collect additional behavioral and engagement data to enhance future predictions.
- Conduct further validation to ensure the model remains fair and unbiased across all departments and employee groups.

The results indicate that the Random Forest model provides a reliable framework for predicting employee turnover and identifying key risk factors. To translate these insights into action:

Implement a predictive HR dashboard using this model to proactively identify at-risk employees.

Focus retention efforts on improving job satisfaction, workload balance, and promotion opportunities.

Conduct quarterly model retraining to ensure continued accuracy and fairness as workforce patterns evolve.

Expand the dataset to include performance scores, engagement metrics, and survey feedback for deeper behavioral insights.

By integrating predictive analytics into HR strategy, the organization can make data-driven decisions that reduce attrition, improve employee engagement, and enhance organizational stability.

Congratulations! You've completed this lab. However, you may not notice a green check mark next to this item on Coursera's platform. Please continue your progress regardless of the check mark. Just click on the "save" icon at the top of this notebook to ensure your work has been logged.