A Deep Dive into Turnover Patterns and Risk Factors

February 24, 2025

1 Abstract

This analysis examines employee turnover patterns using HR data from 14,999 employees to identify key factors influencing departures and develop predictive insights for retention strategies. Through statistical analysis and logistic regression modeling, I identified three primary drivers of turnover: employee satisfaction (showing the strongest negative correlation at -0.39), years at company, and performance evaluation scores.

Key findings revealed distinctive patterns: employees with both very low (below 0.2) and very high (above 0.8) satisfaction levels showed increased turnover risk, with mid-range satisfaction correlating to higher retention. The analysis also uncovered concerning trends in the 3-5 year tenure range, where turnover rates peaked, particularly among high-performing employees working excessive hours (250+ monthly).

Using these insights, we developed a predictive model that can estimate individual turnover risk with reliable accuracy, enabling proactive retention interventions. The model categorizes employees into risk zones (Safe, Low, Medium, and High Risk), allowing HR to prioritize retention efforts effectively.

2 Dataset Overview

This dataset contains HR analytics data from a company with 14,999 employee records across 10 features, focusing on various factors potentially influencing employee turnover.

Key Features:

1. Employee Performance Metrics

- Satisfaction level (0-1 scale)
- Last performance evaluation (0-1 scale)
- Number of projects assigned
- Average monthly working hours

2. Career Development Indicators

- Time spent at company (in years)
- Promotion in last 5 years (binary: 0/1)
- Work accident history (binary: 0/1)

3. Employment Information

- Department (categorical: sales, technical, etc.)

- Salary level (categorical: low, medium, high)

Target Variable:

- Employee turnover status (binary: 0=stayed, 1=left)

3 Obtaining the Data

[6]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as matplot
import seaborn as sns
%matplotlib inline
```

[7]:
```
df=pd.read_csv('../employee turnover project/turnover.csv')
```

[8]:
```
df
```

[8]:

	satisfaction_level	last_evaluation	number_project	\
0	0.38	0.53	2	
1	0.80	0.86	5	
2	0.11	0.88	7	
3	0.72	0.87	5	
4	0.37	0.52	2	
…	…	…	…	
14994	0.40	0.57	2	
14995	0.37	0.48	2	
14996	0.37	0.53	2	
14997	0.11	0.96	6	
14998	0.37	0.52	2	

	average_montly_hours	time_spend_company	Work_accident	left	\
0	157	3	0	1	
1	262	6	0	1	
2	272	4	0	1	
3	223	5	0	1	
4	159	3	0	1	
…	…	…	…	…	
14994	151	3	0	1	
14995	160	3	0	1	
14996	143	3	0	1	
14997	280	4	0	1	
14998	158	3	0	1	

	promotion_last_5years	sales	salary
0	0	sales	low
1	0	sales	medium

```
2                         0    sales  medium
3                         0    sales     low
4                         0    sales     low
…                       …       …      …
14994                     0  support     low
14995                     0  support     low
14996                     0  support     low
14997                     0  support     low
14998                     0  support     low

[14999 rows x 10 columns]
```

[ ]:

4 Data Cleaning

[10]:
```
df.isnull().any()
```

[10]:
```
satisfaction_level       False
last_evaluation          False
number_project           False
average_montly_hours     False
time_spend_company       False
Work_accident            False
left                     False
promotion_last_5years    False
sales                    False
salary                   False
dtype: bool
```

[11]:
```
df.head()
```

[11]:
```
   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  sales  \
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
```

[12]:
```
df = df.rename(columns={'satisfaction_level': 'satisfaction',
                        'last_evaluation': 'evaluation',
                        'number_project': 'projectCount',
                        'average_montly_hours': 'averageMonthlyHours',
                        'time_spend_company': 'yearsAtCompany',
                        'Work_accident': 'workAccident',
                        'promotion_last_5years': 'promotion',
                        'sales' : 'department',
                        'left' : 'turnover'
                        })
```

[13]:
```
front = df['turnover']
df.drop(labels=['turnover'], axis=1,inplace = True)
df.insert(0, 'turnover', front)
df.head()
```

[13]:
```
   turnover  satisfaction  evaluation  projectCount  averageMonthlyHours  \
0         1          0.38        0.53             2                  157
1         1          0.80        0.86             5                  262
2         1          0.11        0.88             7                  272
3         1          0.72        0.87             5                  223
4         1          0.37        0.52             2                  159

   yearsAtCompany  workAccident  promotion department  salary
0               3             0          0      sales     low
1               6             0          0      sales  medium
2               4             0          0      sales  medium
3               5             0          0      sales     low
4               3             0          0      sales     low
```

5 Exploring the Data

5.1 Statistical Overview

The dataset has:

- About 15,000 employee observations and 10 features
- The company had a turnover rate of about 24%
- Mean satisfaction of employees is 0.61

[17]:
```
df.shape
```

[17]:
```
(14999, 10)
```

[18]:
```
df.dtypes
```

[18]:
```
turnover                 int64
satisfaction           float64
evaluation             float64
projectCount             int64
averageMonthlyHours      int64
yearsAtCompany           int64
workAccident             int64
promotion                int64
department              object
salary                  object
dtype: object
```

[19]:
```
# Looks like about 76% of employees stayed and 24% of employees left.

turnover_rate = df.turnover.value_counts() / len(df)
turnover_rate
```

[19]:
```
turnover
0    0.761917
1    0.238083
Name: count, dtype: float64
```

[20]:
```
df.describe()
```

[20]:

	turnover	satisfaction	evaluation	projectCount	averageMonthlyHours	yearsAtCompany	workAccident	promotion
count	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000
mean	0.238083	0.612834	0.716102	3.803054	201.050337	3.498233	0.144610	0.021268
std	0.425924	0.248631	0.171169	1.232592	49.943099	1.460136	0.351719	0.144281
min	0.000000	0.090000	0.360000	2.000000	96.000000	2.000000	0.000000	0.000000
25%	0.000000	0.440000	0.560000	3.000000	156.000000	3.000000	0.000000	0.000000
50%	0.000000	0.640000	0.720000	4.000000	200.000000	3.000000	0.000000	0.000000
75%	0.000000	0.820000	0.870000	5.000000	245.000000	4.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	7.000000	310.000000	10.000000	1.000000	1.000000

[21]:
```
# Overview of summary (Turnover V.S. Non-turnover)
turnover_Summary = df.groupby('turnover')
turnover_Summary.mean(numeric_only=True)
```

[21]:

turnover	satisfaction	evaluation	projectCount	averageMonthlyHours	yearsAtCompany	workAccident	promotion
0	0.666810	0.715473	3.786664	199.060203	3.380032	0.175009	0.026251
1	0.440098	0.718113	3.855503	207.419210	3.876505	0.047326	0.005321

5.2 Correlation Matrix & Heatmap

Moderate Positively Correlated Features:

- projectCount vs evaluation: 0.349333
- projectCount vs averageMonthlyHours: 0.417211
- averageMonthlyHours vs evaluation: 0.339742

Moderate Negatively Correlated Feature:

- satisfaction vs turnover: -0.388375

Stop and Think:

- What features affect our target variable the most (turnover)?
- What features have strong correlations with each other?
- Can we do a more in depth examination of these features?

Summary:

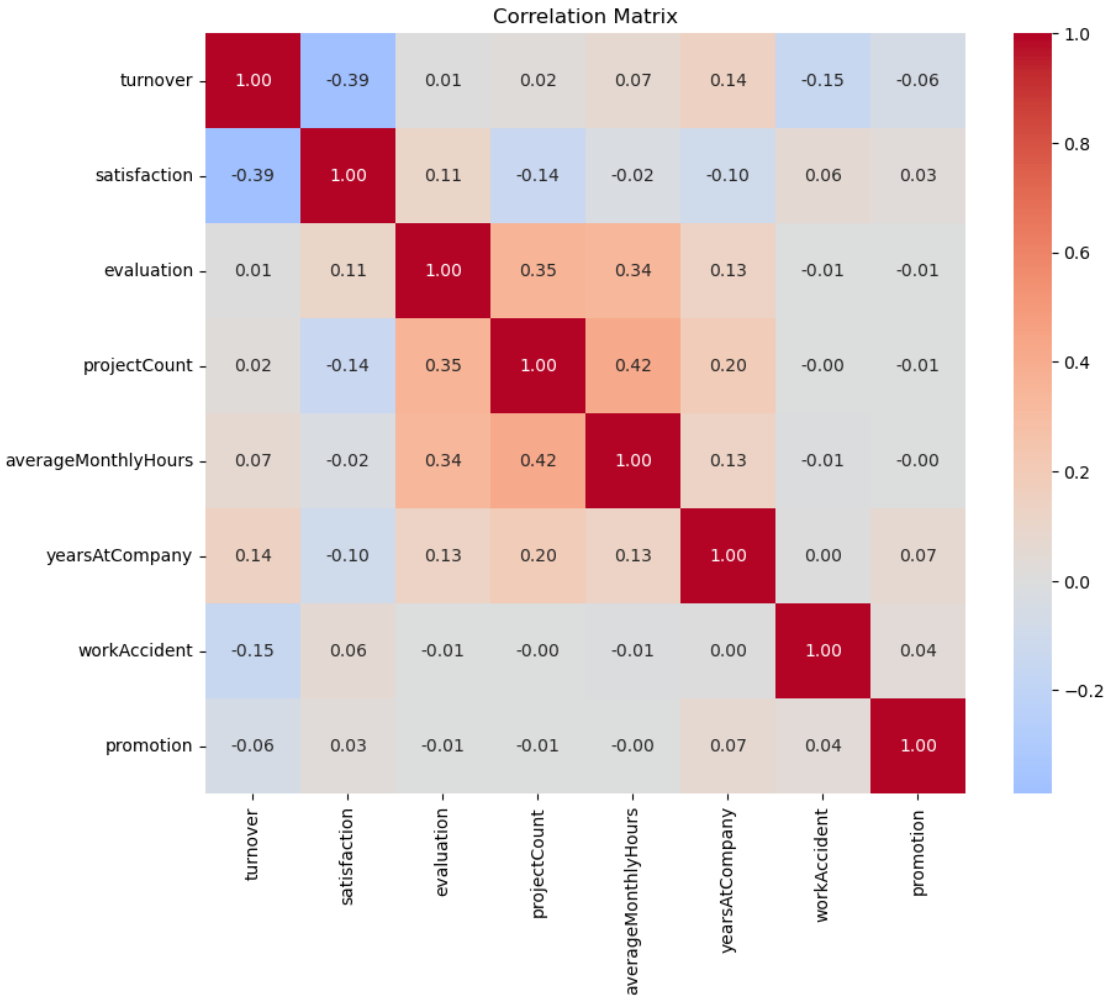
From the heatmap, there is a **positive(+)** correlation between projectCount, averageMonthlyHours, and evaluation. Which could mean that the employees who spent more hours and did more projects were evaluated highly.

For the **negative(-)** relationships, turnover and satisfaction are highly correlated. I'm assuming that people tend to leave a company more when they are less satisfied.

start from In [12]

[25]:
```
# Correlation Matrix

numeric_df = df.select_dtypes(include=['float64', 'int64'])
corr = numeric_df.corr()

plt.figure(figsize=(10, 8))
sns.heatmap(corr,
            xticklabels=corr.columns.values,
            yticklabels=corr.columns.values,
```

```
            annot=True,
            cmap='coolwarm',
            center=0,
            fmt='.2f',
            square=True)

plt.title('Correlation Matrix')
plt.show()
```



5.3 Statistical Test for Correlation

5.3.1 One-Sample T-Test (Measuring Satisfaction Level)¶

A one-sample t-test checks whether a sample mean differs from the population mean. Since satisfaction has the highest correlation with our dependent variable turnover, let's test to see whether the average satisfaction level of employees that had a turnover differs from those that had no turnover.

Hypothesis Testing: Is there significant difference in the **means of satisfaction level** between employees who had a turnover and temployees who had no turnover?

- **Null Hypothesis**: *(H0: pTS = pES)* The null hypothesis would be that there is no difference in satisfaction level between employees who did turnover and those who did not..
- **Alternate Hypothesis**: *(HA: pTS != pES)* The alternative hypothesis would be that there is a difference in satisfaction level between employees who did turnover and those who did not..

[28]:
```
# Compare average satisfaction between current employees and those who left
# Calculate mean satisfaction for current employees
current_emp_satisfaction = df['satisfaction'][df['turnover'] == 0].mean()

# Calculate mean satisfaction for employees who left
former_emp_satisfaction = df[df['turnover']==1]['satisfaction'].mean()


print(f'The mean satisfaction for current employees is:␣
 ↪{current_emp_satisfaction:.3f}')
print(f'The mean satisfaction for employees who left is:␣
 ↪{former_emp_satisfaction:.3f}')
```

```
The mean satisfaction for current employees is: 0.667
The mean satisfaction for employees who left is: 0.440
```

5.3.2 Conducting the Independent T-Test

Let's conduct an **independent t-test at 95% confidence level** to compare satisfaction levels between current employees and those who left. This test will help us determine if there is a statistically significant difference in satisfaction between these two groups. The null hypothesis states that there is no difference in satisfaction levels between current and former employees. To conduct this analysis, we'll use the **stats.ttest_ind()** function from scipy.

[30]:
```
current_emp_satisfaction = df['satisfaction'][df['turnover'] == 0].mean()
former_emp_satisfaction = df[df['turnover']==1]['satisfaction'].mean()


n_current = len(df[df['turnover'] == 0])
n_former = len(df[df['turnover'] == 1])

print(f'The mean satisfaction for current employees (n={n_current}) is:␣
 ↪{current_emp_satisfaction:.3f}')
print(f'The mean satisfaction for employees who left (n={n_former}) is:␣
 ↪{former_emp_satisfaction:.3f}')


from scipy import stats
```

```
t_stat, p_value = stats.ttest_ind(
    df[df['turnover'] == 0]['satisfaction'],  # Current employees group
    df[df['turnover'] == 1]['satisfaction']   # Former employees group
)

print(f'\nIndependent t-test results:')
print(f't-statistic: {t_stat:.3f}')
print(f'p-value: {p_value:.4f}')


if p_value < 0.05:
    print('\nThe difference in satisfaction between current and former␣
 ↪employees is statistically significant.')
else:
    print('\nNo statistically significant difference was found in satisfaction␣
 ↪between current and former employees.')
```

```
The mean satisfaction for current employees (n=11428) is: 0.667
The mean satisfaction for employees who left (n=3571) is: 0.440

Independent t-test results:
t-statistic: 51.613
p-value: 0.0000

The difference in satisfaction between current and former employees is
statistically significant.
```

Based on the independent t-test results (t=51.613, p<0.001), **we reject the null hypothesis**. The data provides strong statistical evidence that **there is a significant difference in satisfaction levels between current employees (M=0.667, n=11,428) and former employees (M=0.440, n=3,571)**.

The test result shows the test statistic "t" is equal to 51.613. This test statistic measures the difference in satisfaction between current and former employees, scaled by the variability in the data. Since our p-value (0.0000) is less than our significance level (0.05), we reject the null hypothesis. This indicates that the difference in satisfaction between current employees (M=0.667) and former employees (M=0.440) is statistically significant and not due to random chance.b

[ ]:

5.3.3 T-Test Quantile

[34]:
```
degree_freedom = n_current + n_former - 2  # n1 + n2 - 2 for independent t-test


LQ = stats.t.ppf(0.025, degree_freedom)
RQ = stats.t.ppf(0.975, degree_freedom)
```

```
print(f'The t-distribution left critical value is: {LQ:.3f}')
print(f'The t-distribution right critical value is: {RQ:.3f}')
```

```
The t-distribution left critical value is: -1.960
The t-distribution right critical value is: 1.960
```

5.3.4 Summary of Independent T-Test

To examine the relationship between employee satisfaction and turnover, we conducted an independent t-test comparing satisfaction levels between current and former employees. The analysis revealed significant differences between the two groups: Current employees (n=11,428) showed a higher mean satisfaction score (M=0.667) compared to former employees (n=3,571, M=0.440). The independent t-test results (t=51.613, p<0.0000) indicate that this difference is statistically significant at the 95% confidence level, leading us to **reject the null hypothesis** that there is no difference in satisfaction between the groups. The substantial difference in satisfaction scores (0.227) and the extremely low p-value suggest that employee satisfaction is strongly associated with turnover behavior. This finding has important implications for HR practices:

Employee satisfaction could serve as an early indicator of turnover risk Special attention should be paid to employees whose satisfaction scores fall below 0.440 Implementing targeted interventions for employees showing low satisfaction levels may help prevent future turnover

These results emphasize the importance of regularly monitoring and addressing employee satisfaction as part of a comprehensive retention strategy.

[ ]:
```

```

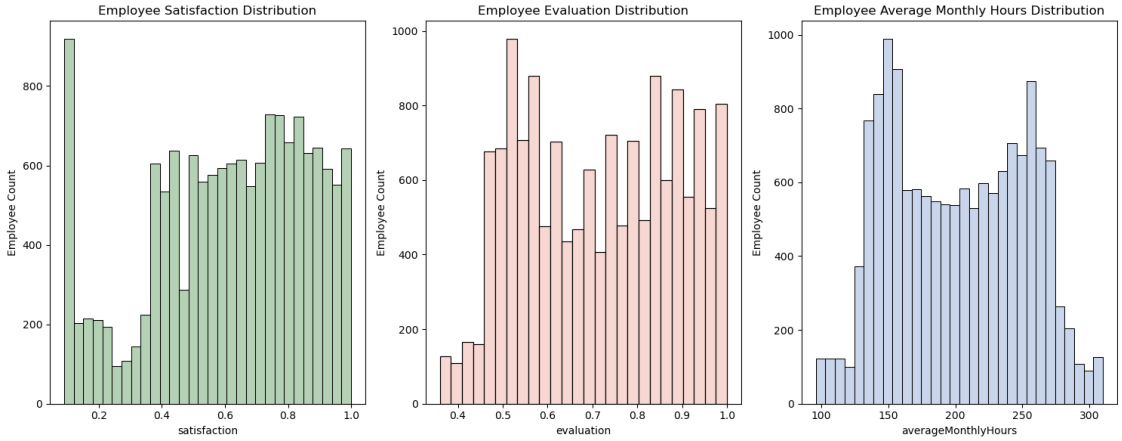
5.4 Distribution Plots (Satisfaction - Evaluation - AverageMonthlyHours)

Summary: Let's examine the distribution on some of the employee's features. Here's what I found:

- **Satisfaction** - There is a huge spike for employees with low satisfaction and high satisfaction.
- **Evaluation** - There is a bimodal distrubtion of employees for low evaluations (less than 0.6) and high evaluations (more than 0.8)
- **AverageMonthlyHours** - There is another bimodal distribution of employees with lower and higher average monthly hours (less than 150 hours & more than 250 hours)
- The evaluation and average monthly hour graphs both share a similar distribution.
- Employees with lower average monthly hours were evaluated less and vice versa.
- If you look back at the correlation matrix, the high correlation between evaluation and averageMonthlyHours does support this finding.

[38]:
```
f, axes = plt.subplots(ncols=3, figsize=(15, 6))


soft_green = "#9BC19C"
soft_pink = "#F5CAC3"
soft_blue = "#B8C7E4"
```

```
sns.histplot(data=df, x='satisfaction', color=soft_green, ax=axes[0])
axes[0].set_title('Employee Satisfaction Distribution')
axes[0].set_ylabel('Employee Count')


sns.histplot(data=df, x='evaluation', color=soft_pink, ax=axes[1])
axes[1].set_title('Employee Evaluation Distribution')
axes[1].set_ylabel('Employee Count')


sns.histplot(data=df, x='averageMonthlyHours', color=soft_blue, ax=axes[2])
axes[2].set_title('Employee Average Monthly Hours Distribution')
axes[2].set_ylabel('Employee Count')

plt.tight_layout()
plt.show()
```



Stop and Think:

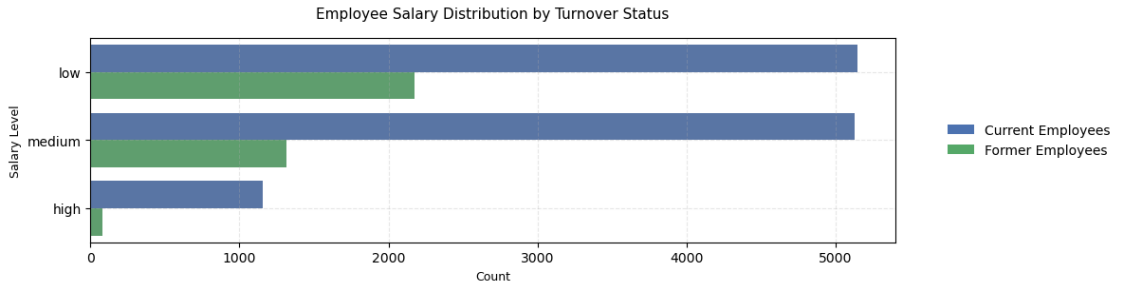
- Is there a reason for the high spike in low satisfaction of employees?
- Could employees be grouped in a way with these features?
- Is there a correlation between evaluation and averageMonthlyHours?

[ ]:

5.5 Salary and Turnover Analysis

[41]:
```
import matplotlib.pyplot as plt
import seaborn as sns


retained_color = '#4C72B0'  # Blue
```

```
left_color = '#55A868'      # Green


plt.figure(figsize=(12, 3.2))


ax = sns.countplot(y="salary",
                   hue='turnover',
                   data=df,
                   palette={0: retained_color, 1: left_color})


ax.set_title('Employee Salary Distribution by Turnover Status',
             pad=15, fontsize=11)
ax.set_xlabel('Count', fontsize=9)
ax.set_ylabel('Salary Level', fontsize=9)


ax.get_legend().remove()
legend_elements = [
    plt.Rectangle((0,0),1,1, facecolor=retained_color, label='Current␣
 ↪Employees'),
    plt.Rectangle((0,0),1,1, facecolor=left_color, label='Former Employees')
]
ax.legend(handles=legend_elements,
          bbox_to_anchor=(1.05, 0.5),
          loc='center left',
          frameon=False)


ax.grid(True, alpha=0.3, linestyle='--')
plt.tight_layout()
plt.show()
```



[ ]:

Summary: This is not unusual. Here's what I found:

- Majority of employees who left either had low or medium salary.
- Barely any employees left with high salary
- Employees with low to average salaries tend to leave the company.

Stop and Think:

- What is the work environmebnt like for low, medium, and high salaries?
- What made employees with high salaries to leave?

[ ]:

5.6 Department and Turnover Distribution

Summary: Let's see more information about the departments. Here's what I found:

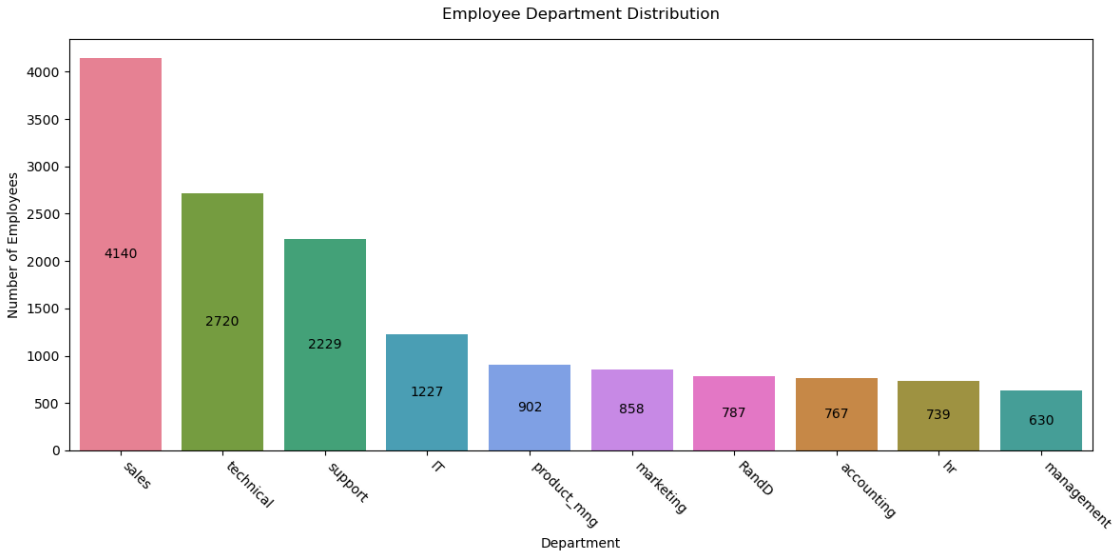
- The sales, technical, and support department were the top 3 departments to have employee turnover
- The management department had the smallest amount of turnover

Stop and Think:

- If we had more information on each department, can we pinpoint a more direct cause for employee turnover?

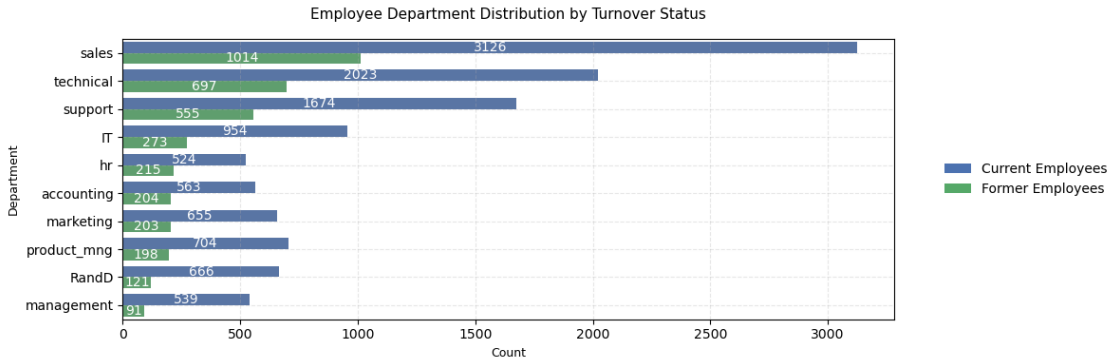
[45]:
```
plt.figure(figsize=(12, 6))


dept_order = df['department'].value_counts().index


n_departments = len(df['department'].unique())
color_types = sns.color_palette('husl', n_departments)


ax = sns.countplot(data=df,
                   x='department',
                   palette=color_types,
                   hue='department',
                   legend=False,
                   order=dept_order)


plt.title('Employee Department Distribution', pad=15, size=12)
plt.xlabel('Department', size=10)
plt.ylabel('Number of Employees', size=10)


for i in ax.containers:
    ax.bar_label(i, label_type='center')
```

```
plt.xticks(rotation=-45, ha='left')


plt.tight_layout()
plt.show()
```



[46]:
```
turnover_by_dept = df[df['turnover']==1]['department'].value_counts().index


f, ax = plt.subplots(figsize=(12, 4))


ax = sns.countplot(y="department",
                   hue='turnover',
                   data=df,
                   order=turnover_by_dept,
                   palette={0: retained_color, 1: left_color})


for container in ax.containers:
   ax.bar_label(container,
                label_type='center',
                color='white')


plt.title('Employee Department Distribution by Turnover Status',
```

```
        pad=15, size=11)
plt.xlabel('Count', size=9)
plt.ylabel('Department', size=9)


legend = ax.get_legend()
legend.remove()


from matplotlib.patches import Patch
legend_elements = [Patch(facecolor=retained_color, label='Current Employees'),
                   Patch(facecolor=left_color, label='Former Employees')]


plt.legend(handles=legend_elements,
          bbox_to_anchor=(1.05, 0.5),
          loc='center left',
          frameon=False)


plt.grid(True, alpha=0.3, linestyle='--')


plt.tight_layout()
plt.show()
```



[ ]:

5.7 Relationship between ProjectCount and Turnover

Summary: This graph is quite interesting as well. Here's what I found:

- More than half of the employees with 2,6, and 7 projects left the company
- Majority of the employees who did not leave the company had 3,4, and 5 projects

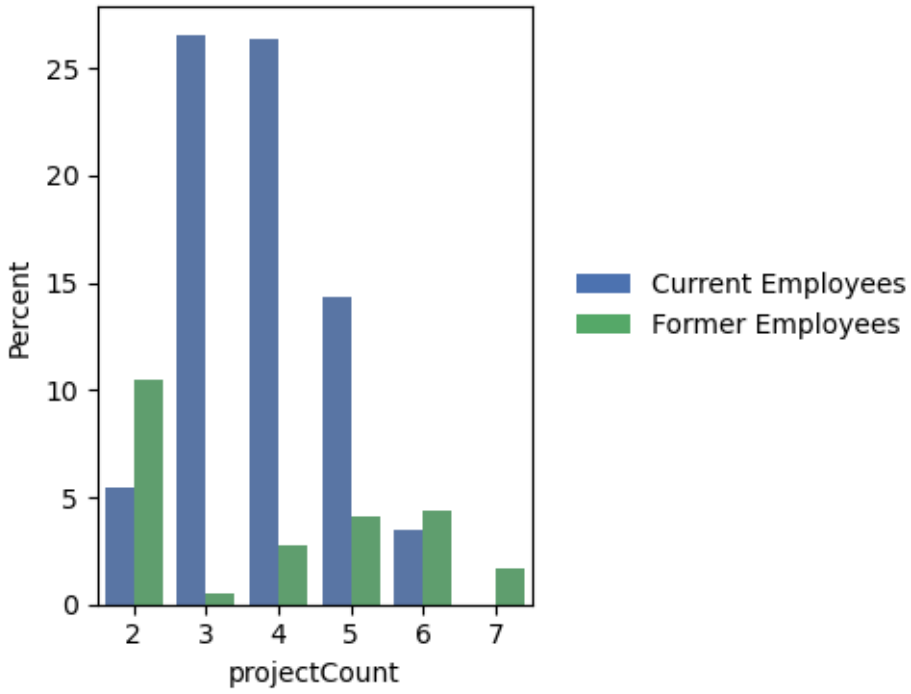
- All of the employees with 7 projects left the company
- There is an increase in employee turnover rate as project count increases

Stop and Think:

- Why are employees leaving at the lower/higher spectrum of project counts?
- Does this means that employees with project counts 2 or less are not worked hard enough or are not highly valued, thus leaving the company?
- Do employees with 6+ projects are getting overworked, thus leaving the company?

[ ]:

[49]:
```
retained_color = '#4C72B0'
left_color = '#55A868'

ax = sns.barplot(x="projectCount", y="projectCount", hue="turnover", data=df,
                 estimator=lambda x: len(x) / len(df) * 100,
                 palette={0: retained_color, 1: left_color})
ax.set(ylabel="Percent")


legend = ax.get_legend()
legend.remove()


from matplotlib.patches import Patch
legend_elements = [
    Patch(facecolor=retained_color, label='Current Employees'),
    Patch(facecolor=left_color, label='Former Employees')
]

plt.legend(handles=legend_elements,
           bbox_to_anchor=(1.05, 0.5),
           loc='center left',
           frameon=False)


plt.gcf().set_size_inches(plt.gcf().get_size_inches() * 0.8)

plt.tight_layout()
plt.show()
```



5.8 Evaluation's Impact on Turnover

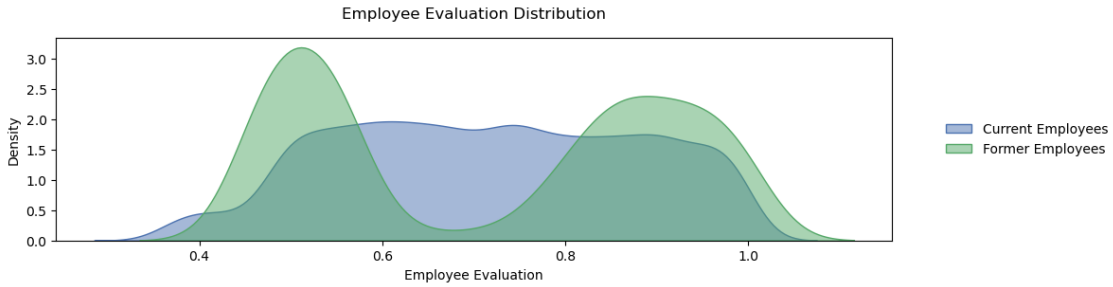
Summary:

- There is a biomodal distribution for those that had a turnover.
- Employees with **low** performance tend to leave the company more
- Employees with **high** performance tend to leave the company more
- The **sweet spot** for employees that stayed is within **0.6-0.8** evaluation

[ ]:

[52]:
```
fig = plt.figure(figsize=(12, 3.2))


ax = sns.kdeplot(data=df.loc[df['turnover'] == 0, 'evaluation'],
                 color=retained_color,  # Blue for current employees
                 fill=True,
                 alpha=0.5,
                 label='Current Employees')


ax = sns.kdeplot(data=df.loc[df['turnover'] == 1, 'evaluation'],
                 color=left_color,  # Green for former employees
                 fill=True,
                 alpha=0.5,
```

```
                label='Former Employees')


plt.title('Employee Evaluation Distribution',
          pad=15, size=12)
ax.set(xlabel='Employee Evaluation',
       ylabel='Density')


plt.legend(bbox_to_anchor=(1.05, 0.5),
           loc='center left',
           frameon=False)


plt.tight_layout()
plt.show()
```



[ ]:

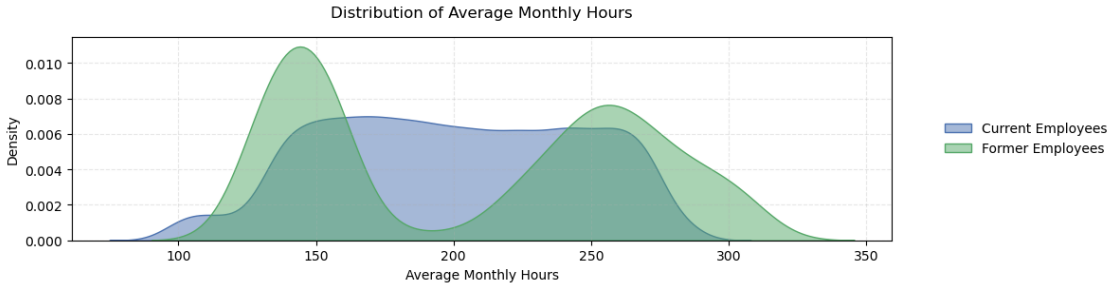
5.9 Analysis of Average Monthly Hours and Turnover

Summary:

- Another bi-modal distribution for employees that turnovered
- Employees who had less hours of work **(~150hours or less)** left the company more
- Employees who had too many hours of work **(~250 or more)** left the company
- Employees who left generally were **underworked** or **overworked**.

[55]:
```
fig, ax = plt.subplots(figsize=(12, 3.2))


retained_color = '#4C72B0'
left_color = '#55A868'

# Plot for retained employees
ax = sns.kdeplot(data=df.loc[df['turnover'] == 0, 'averageMonthlyHours'],
```

```
                color=retained_color,
                fill=True,
                alpha=0.5,
                label='Current Employees')

# Plot for employees who left
ax = sns.kdeplot(data=df.loc[df['turnover'] == 1, 'averageMonthlyHours'],
                color=left_color,
                fill=True,
                alpha=0.5,
                label='Former Employees')


plt.title('Distribution of Average Monthly Hours',
          pad=15, size=12)
ax.set(xlabel='Average Monthly Hours',
       ylabel='Density')


plt.grid(True, alpha=0.3, linestyle='--')

# Modify legend
plt.legend(bbox_to_anchor=(1.05, 0.5),
           loc='center left',
           frameon=False)


plt.tight_layout()
plt.show()
```



[ ]:

5.10 Satisfaction's Correlation with Turnover

[57]:
```
fig, ax = plt.subplots(figsize=(15, 4))


retained_color = '#4C72B0'
left_color = '#55A868'

# Plot for retained employees
ax = sns.kdeplot(data=df.loc[df['turnover'] == 0, 'satisfaction'],
                 color=retained_color,
                 fill=True,
                 alpha=0.5,
                 label='Retained')

# Plot for employees who left
ax = sns.kdeplot(data=df.loc[df['turnover'] == 1, 'satisfaction'],
                 color=left_color,
                 fill=True,
                 alpha=0.5,
                 label='Left')


plt.title('Employee Satisfaction Distribution\nRetained (Blue) vs. Left␣
 ↪(Green)',
          pad=15, size=12)
ax.set(xlabel='Satisfaction Level',
       ylabel='Density')


plt.grid(True, alpha=0.3, linestyle='--')


plt.text(0.02, 0.98, 'Blue = Current Employees',
         transform=ax.transAxes, color=retained_color,
         fontsize=10, va='top')
plt.text(0.02, 0.93, 'Green = Former Employees',
         transform=ax.transAxes, color=left_color,
         fontsize=10, va='top')


plt.tight_layout()
plt.show()
```

When analyzing the relationship between employee satisfaction and turnover, we observed an interesting tri-modal distribution pattern:

1. Low Satisfaction Group (0.2 or lower)
   - Likely represents employees who are extremely dissatisfied with their work environment
   - Recommendation: Conduct exit interviews to understand specific reasons and improve working conditions
2. Medium-Low Satisfaction Group (0.3-0.5)
   - Represents employees with specific but not extreme dissatisfaction
   - Recommendation: Implement regular satisfaction surveys to identify specific pain points
3. High Satisfaction Group (0.7 or higher)
   - This group's turnover deserves special attention, possible reasons include:
     – Better external opportunities
     – Career development bottlenecks
     – Competitive offers from other companies
   - Recommendation: Strengthen career development paths and key talent retention programs

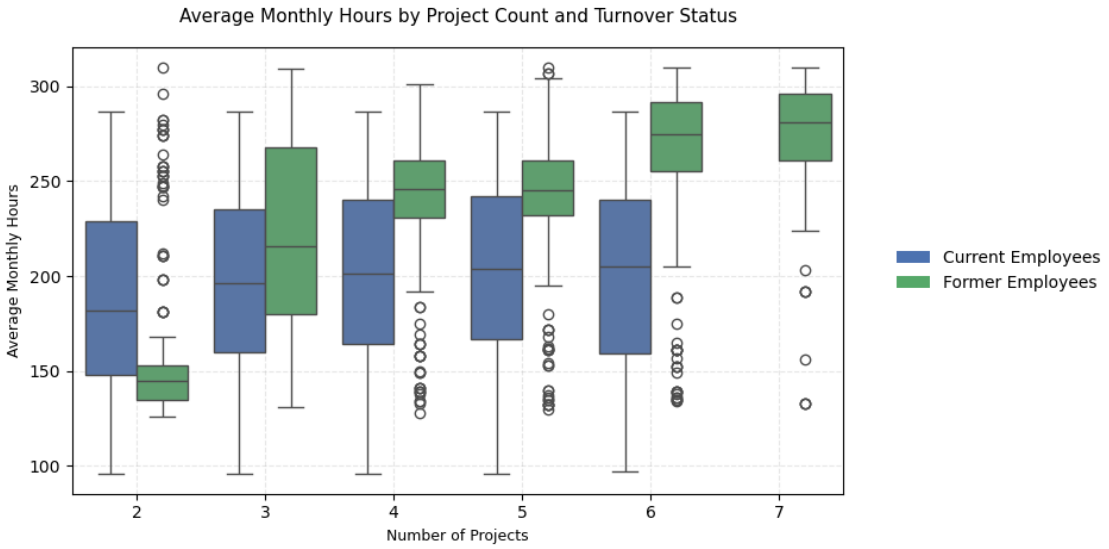
Action Items: - Develop differentiated retention strategies for different satisfaction groups - Establish early warning mechanisms to identify potential turnover risks - Optimize regular performance and satisfaction assessment mechanisms

Business Impact: - Cost savings through improved retention - Better talent management through targeted interventions - Enhanced employee engagement through demonstrated responsiveness

5.11 ProjectCount and Average Monthly Hours Correlation

[60]:
```
plt.figure(figsize=(9.6, 4.8))


ax = sns.boxplot(x="projectCount",
                 y="averageMonthlyHours",
                 hue="turnover",
                 data=df,
                 palette=[retained_color, left_color])
```

```
plt.title('Average Monthly Hours by Project Count and Turnover Status',
          pad=15, size=11)
plt.xlabel('Number of Projects', size=9)
plt.ylabel('Average Monthly Hours', size=9)


legend = ax.get_legend()


legend.remove()


from matplotlib.patches import Patch
legend_elements = [Patch(facecolor=retained_color, label='Current Employees'),
                   Patch(facecolor=left_color, label='Former Employees')]


plt.legend(handles=legend_elements,
           bbox_to_anchor=(1.05, 0.5),
           loc='center left',
           frameon=False)


plt.grid(True, alpha=0.3, linestyle='--')


plt.tight_layout()
plt.show()
```



Summary:

- As project count increased, so did average monthly hours
- Something weird about the boxplot graph is the difference in averageMonthlyHours between people who had a turnver and did not.
- Looks like employees who did not have a turnover had consistent averageMonthlyHours, despite the increase in projects
- In contrast, employees who did have a turnover had an increase in averageMonthlyHours with the increase in projects

Stop and Think:

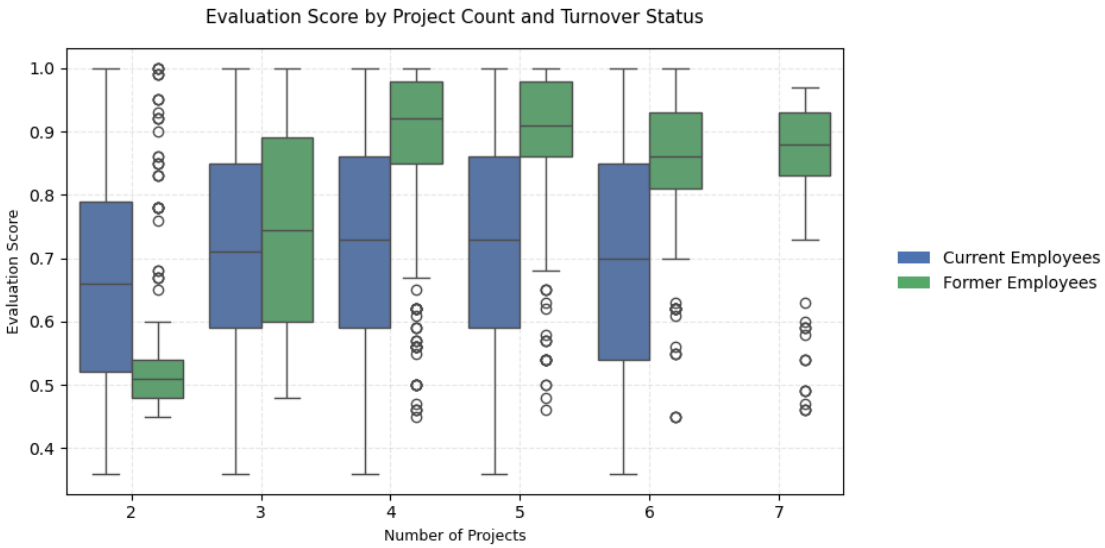
- What could be the meaning for this?
- Why is it that employees who left worked more hours than employees who didn't, even with the same project count?

[ ]:

5.12 ProjectCount and Evaluation Relationship

[63]:
```
plt.figure(figsize=(9.6, 4.8))


ax = sns.boxplot(x="projectCount",
                 y="evaluation",
                 hue="turnover",
                 data=df,
                 palette=[retained_color, left_color])


plt.title('Evaluation Score by Project Count and Turnover Status',
          pad=15, size=11)
plt.xlabel('Number of Projects', size=9)
plt.ylabel('Evaluation Score', size=9)


legend = ax.get_legend()


legend.remove()


from matplotlib.patches import Patch
legend_elements = [Patch(facecolor=retained_color, label='Current Employees'),
                   Patch(facecolor=left_color, label='Former Employees')]


plt.legend(handles=legend_elements,
           bbox_to_anchor=(1.05, 0.5),
```

```
          loc='center left',
          frameon=False)


plt.grid(True, alpha=0.3, linestyle='--')


plt.tight_layout()
plt.show()
```



Summary: This graph looks very similar to the graph above. What I find strange with this graph is with the turnover group. There is an increase in evaluation for employees who did more projects within the turnover group. But, again for the non-turnover group, employees here had a consistent evaluation score despite the increase in project counts.

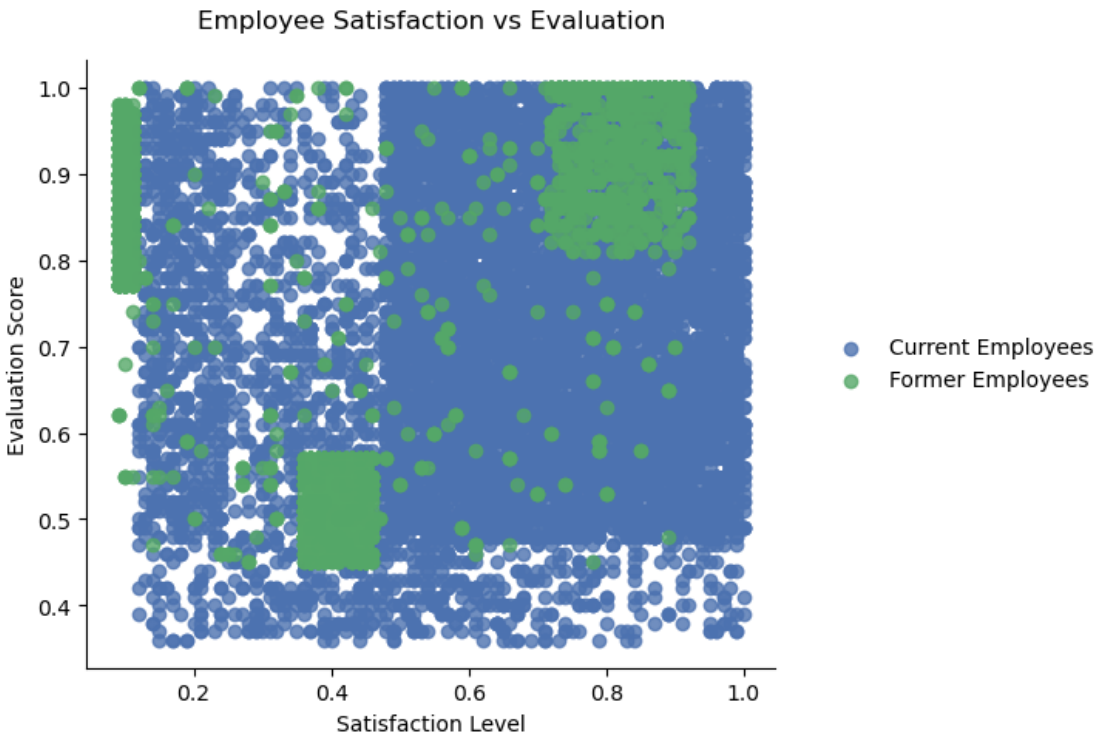
Questions to think about:

- Why is it that employees who left, had on average, a higher evaluation than employees who did not leave, even with an increase in project count?
- Shouldn't employees with lower evaluations tend to leave the company more?

5.13 Satisfaction and Evaluation Analysis

[67]:
```
plot = sns.lmplot(x='satisfaction',
                  y='evaluation',
                  data=df,
                  fit_reg=False,
                  hue='turnover',
                  palette={0: '#4C72B0', 1: '#55A868'},
                  height=5,
```

```
                aspect=1.5,
                legend=False)


plt.legend(['Current Employees', 'Former Employees'],
        bbox_to_anchor=(1.05, 0.5),
        loc='center left',
        frameon=False,
        fontsize=10)


plt.title('Employee Satisfaction vs Evaluation',
        pad=15,
        size=12)
plt.xlabel('Satisfaction Level', size=10)
plt.ylabel('Evaluation Score', size=10)


plt.tight_layout()
plt.show()
```



Summary: This is by far the most compelling graph. This is what I found:

- There are **3** distinct clusters for employees who left the company

Cluster 1 (Hard-working and Sad Employee): Satisfaction was below 0.2 and evaluations were greater than 0.75. Which could be a good indication that employees who left the company were good workers but felt horrible at their job.

- Question: What could be the reason for feeling so horrible when you are highly evaluated? Could it be working too hard? Could this cluster mean employees who are "overworked"?

Cluster 2 (Bad and Sad Employee): Satisfaction between about 0.35~0.45 and evaluations below ~0.58. This could be seen as employees who were badly evaluated and felt bad at work.

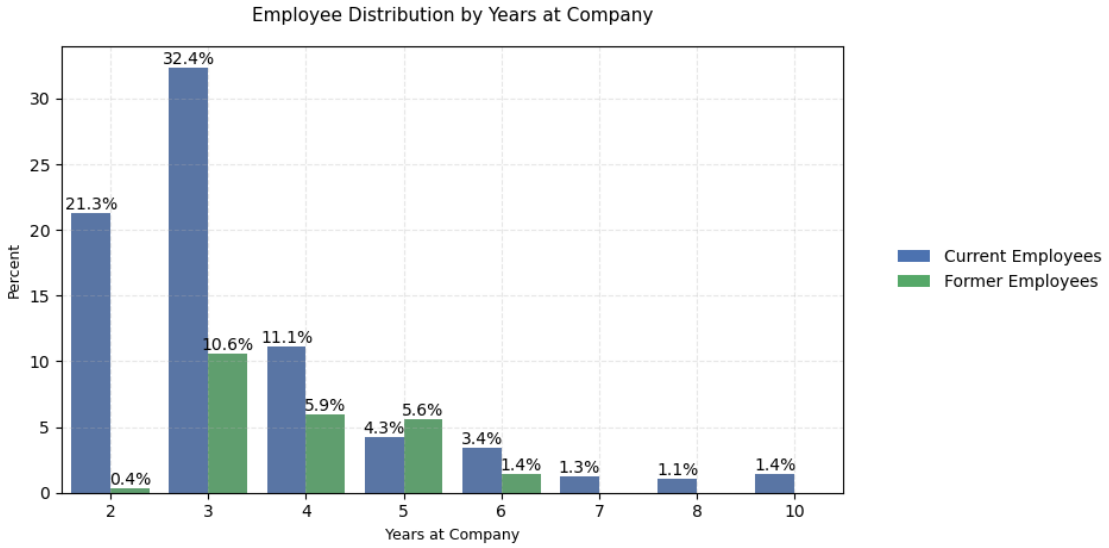
- Question: Could this cluster mean employees who "under-performed"?

Cluster 3 (Hard-working and Happy Employee): Satisfaction between 0.7~1.0 and evaluations were greater than 0.8. Which could mean that employees in this cluster were "ideal". They loved their work and were evaluated highly for their performance.

- Question: Could this cluser mean that employees left because they found another job opportunity?

[69]:
```
plt.figure(figsize=(9.6, 4.8))


ax = sns.barplot(x="yearsAtCompany",
                 y="yearsAtCompany",
                 hue="turnover",
                 data=df,
                 estimator=lambda x: len(x) / len(df) * 100,
                 palette={0: retained_color, 1: left_color})


plt.title('Employee Distribution by Years at Company',
          pad=15, size=11)
plt.xlabel('Years at Company', size=9)
plt.ylabel('Percent', size=9)


legend = ax.get_legend()


legend.remove()


from matplotlib.patches import Patch
legend_elements = [Patch(facecolor=retained_color, label='Current Employees'),
                   Patch(facecolor=left_color, label='Former Employees')]


plt.legend(handles=legend_elements,
           bbox_to_anchor=(1.05, 0.5),
           loc='center left',
```

```
        frameon=False)


plt.grid(True, alpha=0.3, linestyle='--')


for container in ax.containers:
    ax.bar_label(container, fmt='%.1f%%')


plt.tight_layout()
plt.show()
```



5.14 Years at Company and Turnover Patterns

Summary: Let's see if theres a point where employees start leaving the company. Here's what I found:

- More than half of the employees with **4 and 5** years left the company
- Employees with **5** years should highly be looked into

Stop and Think:

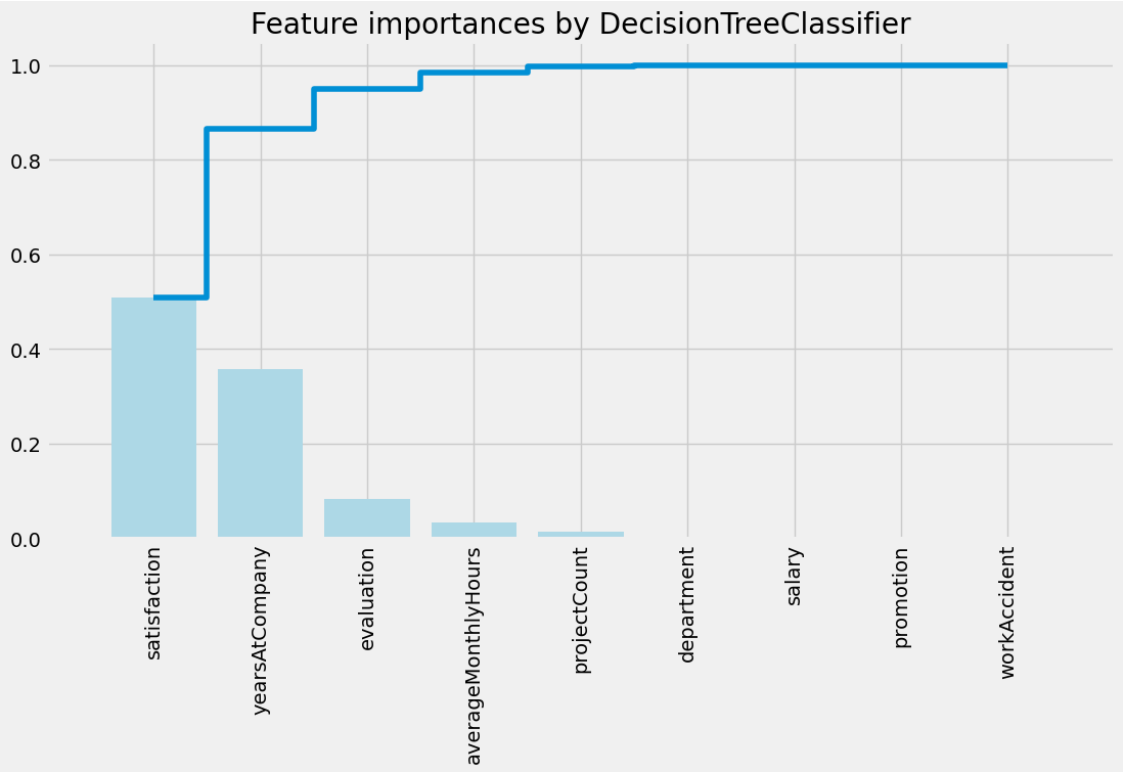
Why are employees leaving mostly at the **3-5** year range? Who are these employees that left? Are these employees part-time or contractors?

[ ]:

6 Feature Importance

[73]:
```
from sklearn import tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
plt.style.use('fivethirtyeight')
plt.rcParams['figure.figsize'] = (12,6)


df = df.rename(columns={'satisfaction_level': 'satisfaction',
                        'last_evaluation': 'evaluation',
                        'number_project': 'projectCount',
                        'average_montly_hours': 'averageMonthlyHours',
                        'time_spend_company': 'yearsAtCompany',
                        'Work_accident': 'workAccident',
                        'promotion_last_5years': 'promotion',
                        'sales' : 'department',
                        'left' : 'turnover'
                        })

# Convert these variables into categorical variables
df["department"] = df["department"].astype('category').cat.codes
df["salary"] = df["salary"].astype('category').cat.codes

# Create train and test splits
target_name = 'turnover'
X = df.drop('turnover', axis=1)


y=df[target_name]

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.15,␣
 ↪random_state=123, stratify=y)

dtree = tree.DecisionTreeClassifier(
    #max_depth=3,
    class_weight="balanced",
    min_weight_fraction_leaf=0.01
    )
dtree = dtree.fit(X_train,y_train)


importances = dtree.feature_importances_
feat_names = df.drop(['turnover'],axis=1).columns


indices = np.argsort(importances)[::-1]
```

```
plt.figure(figsize=(12,6))
plt.title("Feature importances by DecisionTreeClassifier")
plt.bar(range(len(indices)), importances[indices], color='lightblue', ␣
 ↪align="center")
plt.step(range(len(indices)), np.cumsum(importances[indices]), where='mid',␣
 ↪label='Cumulative')
plt.xticks(range(len(indices)), feat_names[indices],␣
 ↪rotation='vertical',fontsize=14)
plt.xlim([-1, len(indices)])
plt.show()
```



[ ]:

Summary:

By using a **decision tree classifier**, it could rank the features used for the prediction. The top three features were employee satisfaction, yearsAtCompany, and evaluation. This is helpful in creating our model for logistic regression because it'll be more interpretable to understand what goes into our model when we utilize less features.

Top 3 Features:

- Satisfaction
- YearsAtCompany

- Evaluation

7 Modelling the Data: Logistic Regression

[76]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as matplot
import seaborn as sns
%matplotlib inline
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,␣
 ↪precision_score, recall_score, confusion_matrix, precision_recall_curve
from sklearn.preprocessing import RobustScaler



df=pd.read_csv('../employee turnover project/turnover.csv')

df = df.rename(columns={
    'satisfaction_level': 'satisfaction',
    'last_evaluation': 'evaluation',
    'number_project': 'projectCount',
    'average_montly_hours': 'averageMonthlyHours',
    'time_spend_company': 'yearsAtCompany',
    'Work_accident': 'workAccident',
    'promotion_last_5years': 'promotion',
    'sales': 'department',
    'left': 'turnover'
})


df["department"] = df["department"].astype('category').cat.codes
df["salary"] = df["salary"].astype('category').cat.codes


front = df['turnover']
df.drop(labels=['turnover'], axis=1, inplace=True)
df.insert(0, 'turnover', front)


df['int'] = 1


indep_var = ['satisfaction', 'evaluation', 'yearsAtCompany', 'int', 'turnover']
```

```
df = df[indep_var]

# Create train and test splits
target_name = 'turnover'
X = df.drop('turnover', axis=1)
y = df[target_name]

X_train, X_test, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.15,
    random_state=123,
    stratify=y
)


print("Training Data Head:")
print(X_train.head())
```

```
Training Data Head:
       satisfaction  evaluation  yearsAtCompany  int
9003           0.59        1.00               3    1
5697           0.81        0.98               2    1
10691          1.00        0.93               2    1
1884           0.87        0.91               5    1
13572          0.87        0.48               3    1
```

NOTE: This will be an in-depth analysis of using logistic regression as a classifier. This is more of a use-case example of what can be done and explained to management in a company.

Logistic Regression commonly deals with the issue of how likely an observation is to belong to each group. This model is commonly used to predict the likelihood of an event occurring. In contrast to linear regression, the output of logistic regression is transformed with a logit function. This makes the output either 0 or 1. This is a useful model to take advantage of for this problem because we are interested in predicting whether an employee will leave (0) or stay (1).

Another reason for why logistic regression is the preferred model of choice is because of its interpretability. Logistic regression predicts the outcome of the response variable (turnover) through a set of other explanatory variables, also called predictors. In context of this domain, the value of our response variable is categorized into two forms: 0 (zero) or 1 (one). The value of 0 (zero) represents the probability of an employee not leaving the company and the value of 1 (one) represents the probability of an employee leaving the company.

Logistic Regression models the probability of 'success' as:

The equation above shows the relationship between, the dependent variable (success), denoted as ( ) and independent variables or predictor of event, denoted as xi. Where is the constant of the equation and, is the coefficient of the predictor variables

[ ]:

8 Using Logistic Regression Coefficients

[79]:
```
import statsmodels.api as sm
iv = ['satisfaction','evaluation','yearsAtCompany', 'int']
logReg = sm.Logit(y_train, X_train[iv])
answer = logReg.fit()

answer.summary
answer.params
```

```
Optimization terminated successfully.
         Current function value: 0.467233
         Iterations 6
```

[79]:
```
satisfaction     -3.769022
evaluation        0.207596
yearsAtCompany    0.170145
int               0.181896
dtype: float64
```

With the elimination of the other variables, I'll be using the three most important features to create our model: Satisfaction, Evaluation, and YearsAtCompany.

Following overall equation was developed:

Employee Turnover Score = Satisfaction(-3.769022) + Evaluation(0.207596) + YearsAtCompany*(0.170145) + 0.181896

9 Explanation of Coefficients

[82]:
```
coef = answer.params

def calculate_turnover_probability(coef, satisfaction, evaluation,␣
 ↪years_at_company):

   intercept = coef['int']
   sat_coef = coef['satisfaction']
   eval_coef = coef['evaluation']
   years_coef = coef['yearsAtCompany']


   logit = intercept + \
           sat_coef * satisfaction + \
           eval_coef * evaluation + \
           years_coef * years_at_company


   probability = np.exp(logit) / (1 + np.exp(logit))
```

```
    return probability

# Example: Employee with 0.7 satisfaction, 0.8 evaluation and 3 years tenure
turnover_prob = calculate_turnover_probability(coef, 0.7, 0.8, 3)
print(f"Probability of turnover: {turnover_prob:.2%}")
```

```
Probability of turnover: 14.43%
```

Employee Turnover Score = Satisfaction(-3.769022) + Evaluation(0.207596) + YearsAtCompany*(0.170145) + 0.181896

The values above are the coefficient assigned to each independent variable. The constant 0.181896 represents the effect of all uncontrollable variables.

10 Intepretation of Score - Example

If you were to use these employee values into the equation:

- Satisfaction: 0.7
- Evaluation: 0.8
- YearsAtCompany: 3 You would get:

Employee Turnover Score = (0.7)(-3.769022) + (0.8)(0.207596) + (3)(0.170145) + 0.181896 = 0.14431 = 14%

Result: This employee would have a 14% chance of leaving the company. This information can then be used to form our retention plan.

11 Retention Plan Using Logistic Regression

With the logistic regression model, we can now use our scores and evaluate the employees through different scoring metrics. Each zone is explain here:

- **Safe Zone (Green)** – Employees within this zone are considered safe.
- **Low Risk Zone (Yellow)** – Employees within this zone are too be taken into consideration of potential turnover. This is more of a long-term track.
- **Medium Risk Zone (Orange)** – Employees within this zone are at risk of turnover. Action should be taken and monitored accordingly.
- **High Risk Zone (Red)** – Employees within this zone are considered to have the highest chance of turnover. Action should be taken immediately.

So with our example above, the employee with a 14% turnover score will be in the safe zone.

12 Conclusion

This paper outlines the different analysis done on the employee dataset and the usage of a Logistic regression model to make predictive insights on the probability of an employee to turnover. This model can be applied throughout the various departments of the company and be used as an aid to

help make better decisions in employee retention. The model should be updated periodically and include additional features for it to make more accurate predictions.

[ ]: