Activity_ Course 7 Salifort Motors project lab

March 6, 2024

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

[32]:
```
# Import packages
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import sklearn.metrics as metrics

from sklearn.metrics import 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.

[3]:
```
# 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()
```

[3]:
```
   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

[4]:

```
# Gather basic information about the data
print(df0.shape)
print(df0.info())
print(df0.isna().sum())
```

```
(14999, 10)
<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
None
satisfaction_level       0
last_evaluation          0
number_project           0
average_montly_hours     0
time_spend_company       0
```

```
Work_accident            0
left                     0
promotion_last_5years    0
Department               0
salary                   0
dtype: int64
```

2.3.2 Gather descriptive statistics about the data

[5]:
```python
# Gather descriptive statistics about the data
df0.describe()
```

[5]:
```
       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.

[6]:
```
# Display all column names
df0.columns.values
```

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

[7]:
```
# Rename columns as needed
df0.rename(columns = {'Work_accident' : 'work_accident' , 'Department' :␣
↪'department'}, inplace=True)


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

[7]:
```
array(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_montly_hours', 'time_spend_company', 'work_accident',
       'left', 'promotion_last_5years', 'department', 'salary'],
      dtype=object)
```

2.3.4 Check missing values

Check for any missing values in the data.

[8]:
```
# Check for missing values
df0.isnull().sum()
```

[8]:
```
satisfaction_level       0
last_evaluation          0
number_project           0
average_montly_hours     0
time_spend_company       0
work_accident            0
left                     0
promotion_last_5years    0
department               0
salary                   0
dtype: int64
```

2.3.5 Check duplicates

Check for any duplicate entries in the data.

[9]:
```
# Check for duplicates
df0.duplicated()
```

[9]:
```
0        False
1        False
2        False
3        False
4        False
         …
14994     True
14995     True
14996     True
14997     True
14998     True
Length: 14999, dtype: bool
```

[10]:
```
# Inspect some rows containing duplicates as needed
df0.duplicated(subset = ['satisfaction_level', 'last_evaluation',␣
 ↪'number_project'])
```

[10]:
```
0        False
1        False
2        False
3        False
4        False
         …
14994     True
14995     True
14996     True
14997     True
14998     True
Length: 14999, dtype: bool
```

[11]:
```
# Drop duplicates and save resulting dataframe in a new variable as needed
df_dropped = df0.drop_duplicates()


# Display first few rows of new dataframe as needed
df_dropped.head()
```

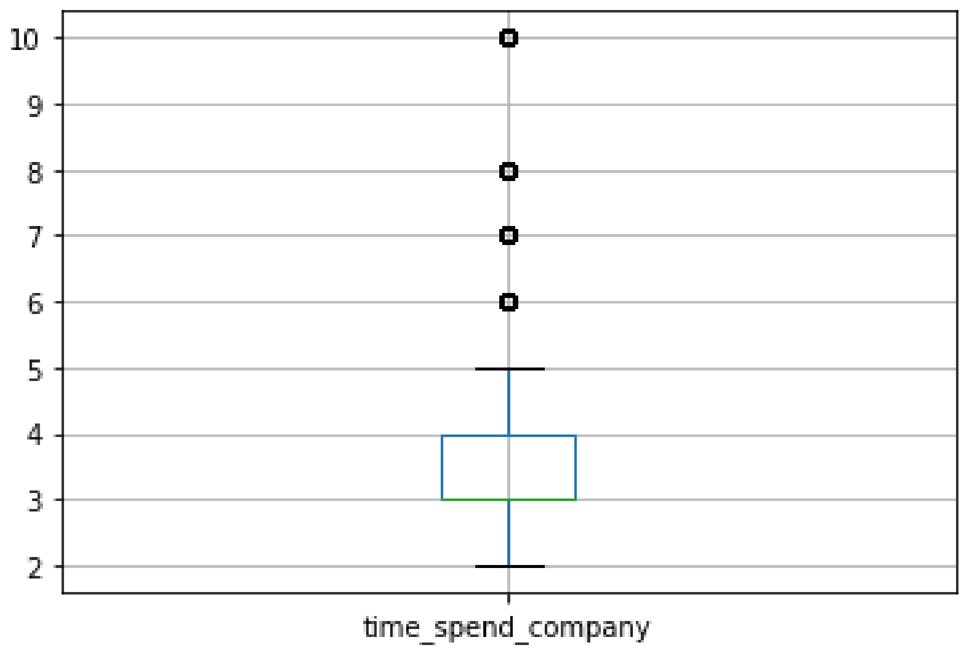
[11]:
```
   satisfaction_level  last_evaluation  number_project  average_montly_hours  \
0                0.38             0.53               2                   157
1                0.80             0.86               5                   262
2                0.11             0.88               7                   272
3                0.72             0.87               5                   223
4                0.37             0.52               2                   159
```

```
   time_spend_company  work_accident  left  promotion_last_5years 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
```

2.3.6 Check outliers

Check for outliers in the data.

[12]:
```
# Create a boxplot to visualize distribution of `tenure` and detect any outliers

boxplot = df_dropped.boxplot(column= ['time_spend_company'])
```



[13]:
```
#from above boxplot we can conclude that lower limit is = (2 - 1.5*1 )= 0.5 and␣
↪upper limit is 5+ 1.5*1=6.5
# Determine the number of rows containing outliers
```

```
df_outlier = df_dropped[df_dropped['time_spend_company'] > 5.5]
df_outlier.shape
```

[13]:
```
(824, 10)
```

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?

[Double-click to enter your responses here.]

3.1 Step 2. Data Exploration (Continue EDA)

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

[14]:
```
# Get numbers of people who left vs. stayed
df_dropped['left'].value_counts()

# Get percentages of people who left vs. stayed
df_dropped['left'].value_counts(normalize=True)
```

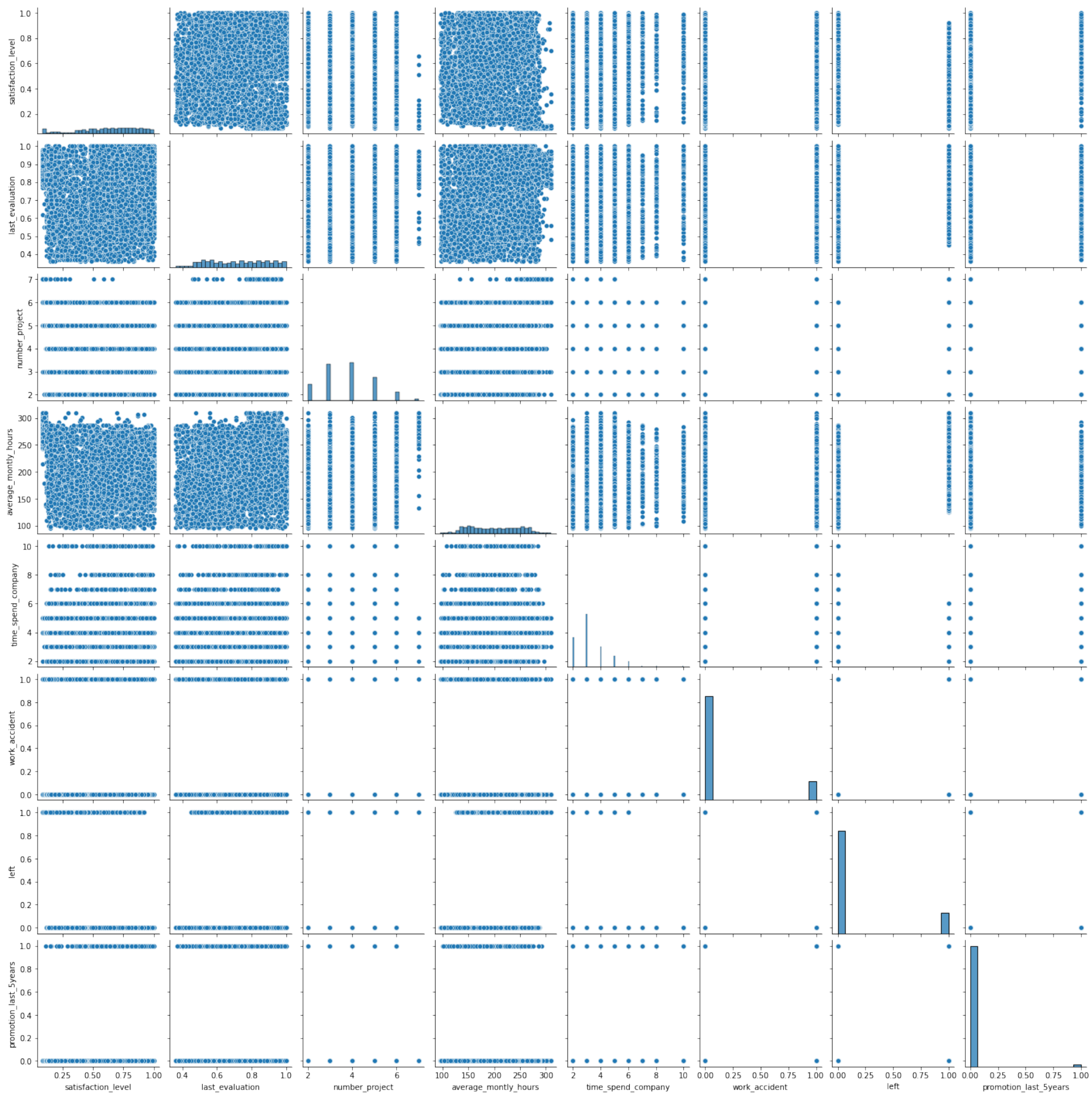
[14]:
```
0    0.833959
1    0.166041
Name: left, dtype: float64
```

3.1.1 Data visualizations

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

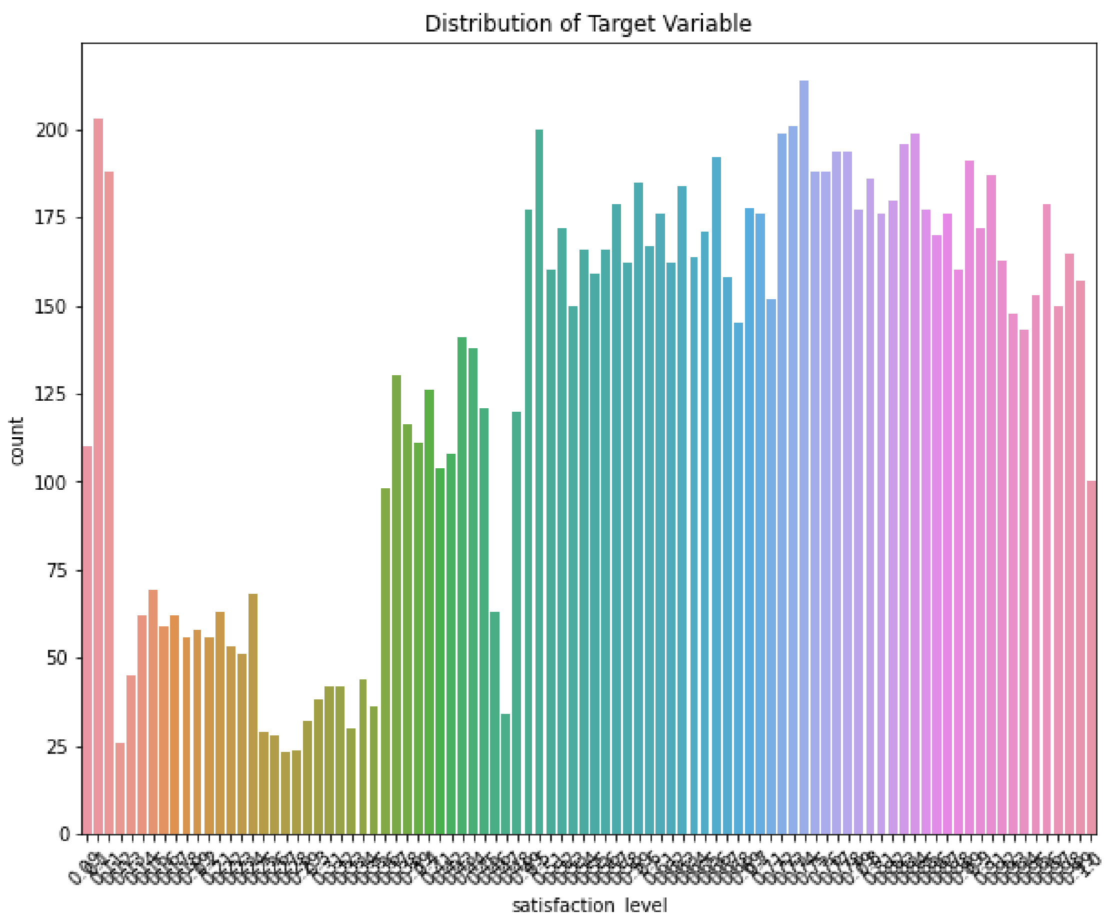
[15]:
```
# Create a plot as needed
sns.pairplot(df_dropped)
```

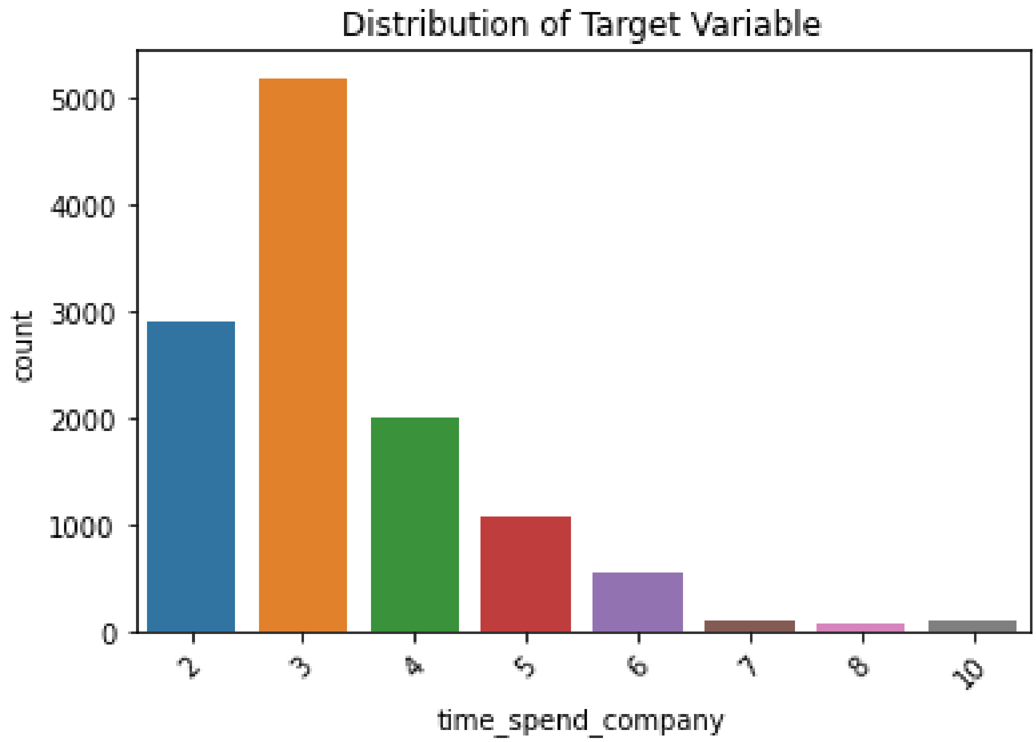
[15]:
```
<seaborn.axisgrid.PairGrid at 0x7f5f37d838d0>
```

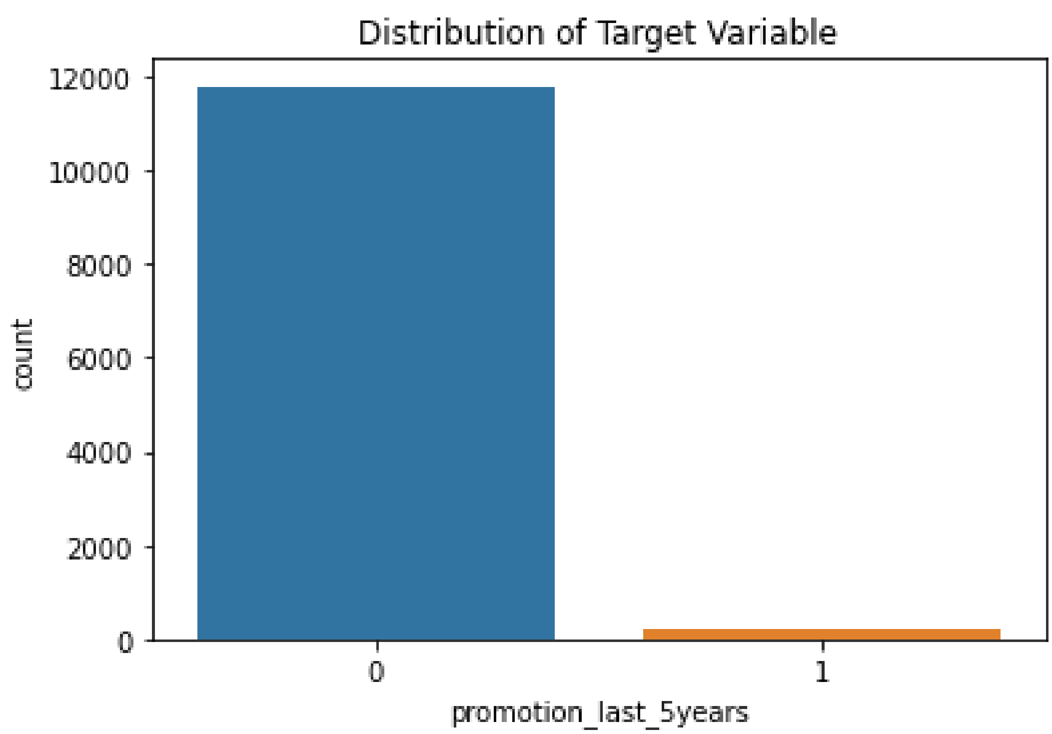


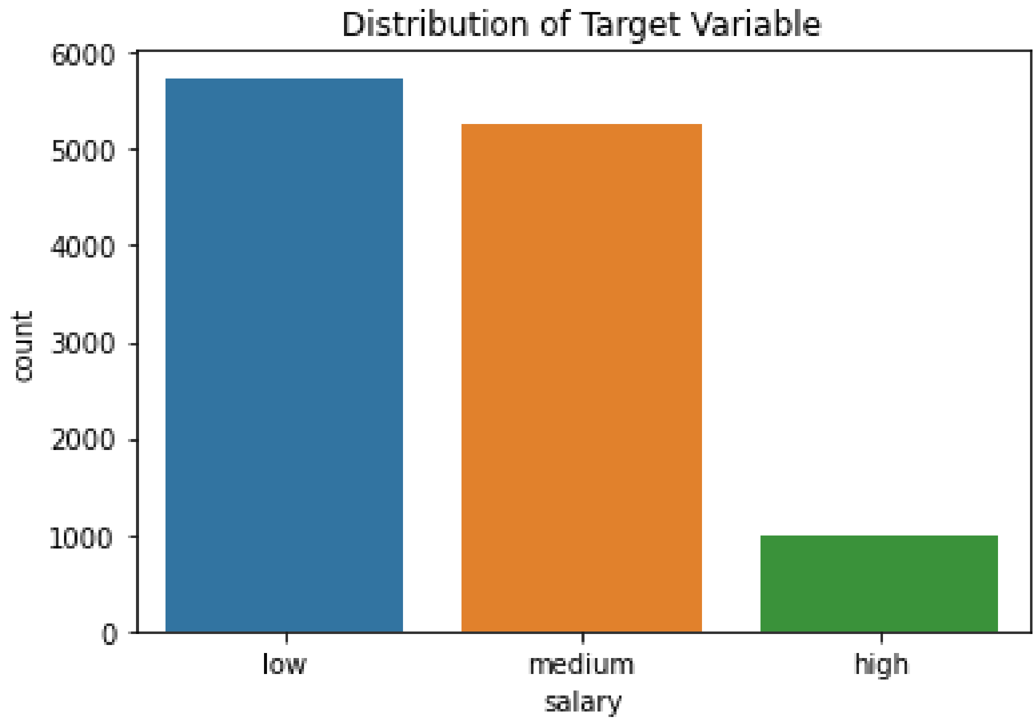
[16]:
```
# Create a plot as needed
plt.figure(figsize=(10, 8))
sns.countplot(x= 'satisfaction_level', data=df_dropped)
plt.title('Distribution of Target Variable')
plt.xticks(rotation=45)
plt.show()

plt.xticks(rotation=45)
```

```python
sns.countplot(x= 'time_spend_company', data=df_dropped)
plt.title('Distribution of Target Variable')
plt.show()

sns.countplot(x= 'promotion_last_5years', data=df_dropped)
plt.title('Distribution of Target Variable')
plt.show()

sns.countplot(x= 'salary', data=df_dropped)
plt.title('Distribution of Target Variable')
plt.show()
```



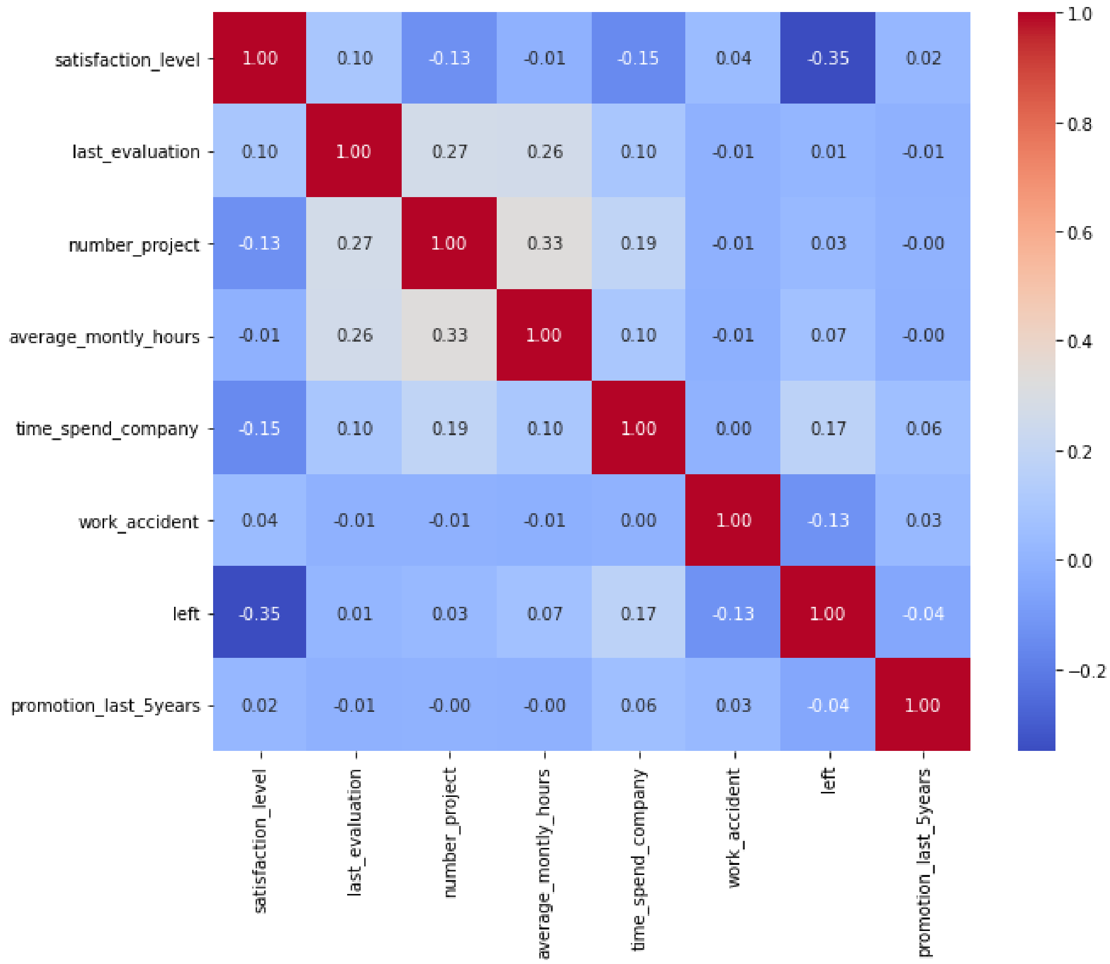

Distribution of Target Variable
count
5000
4000
3000
2000
1000
0
2
3
4
5
6
7
8
10
time_spend_company


Distribution of Target Variable
count
12000
10000
8000
6000
4000
2000
0
0
1
promotion_last_5years

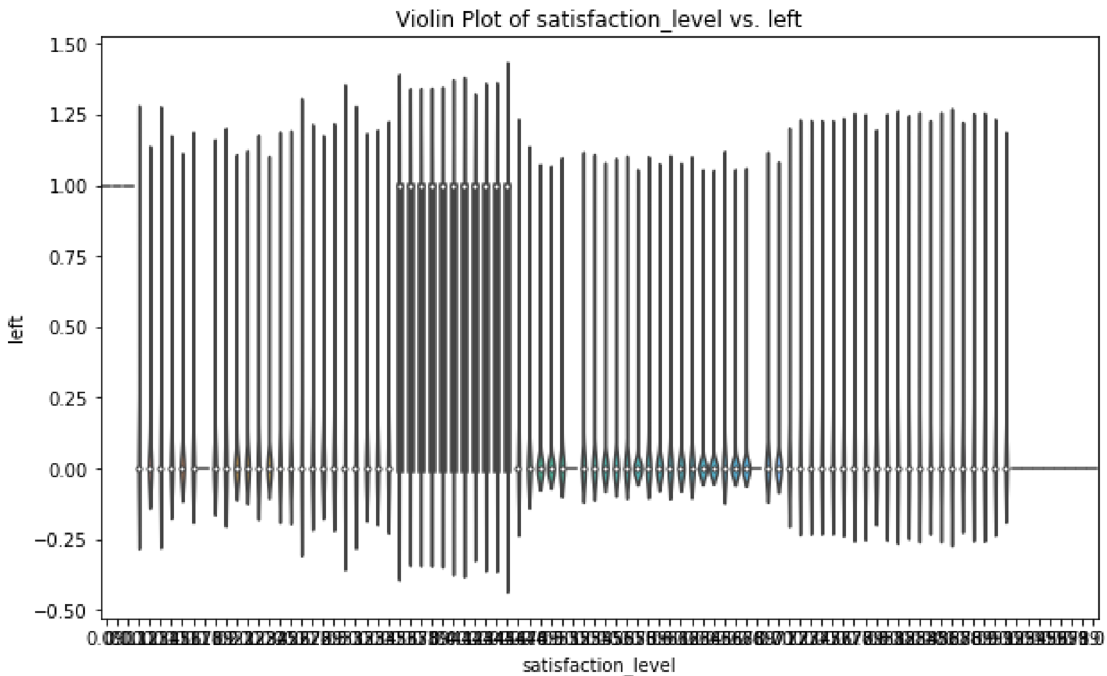


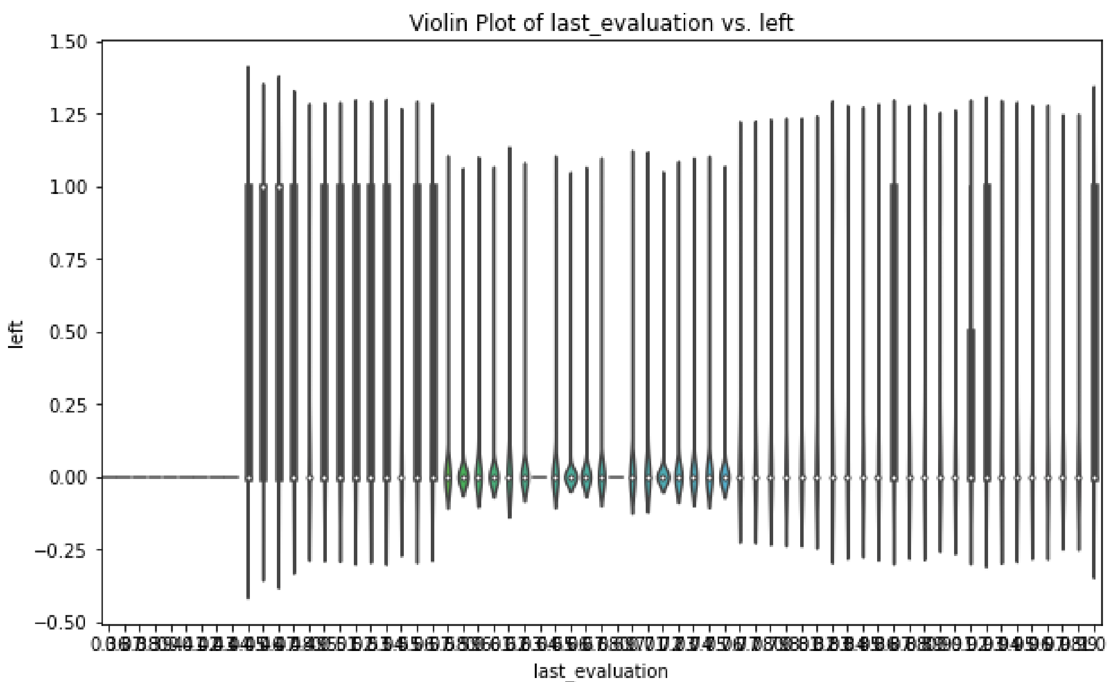
[17]:

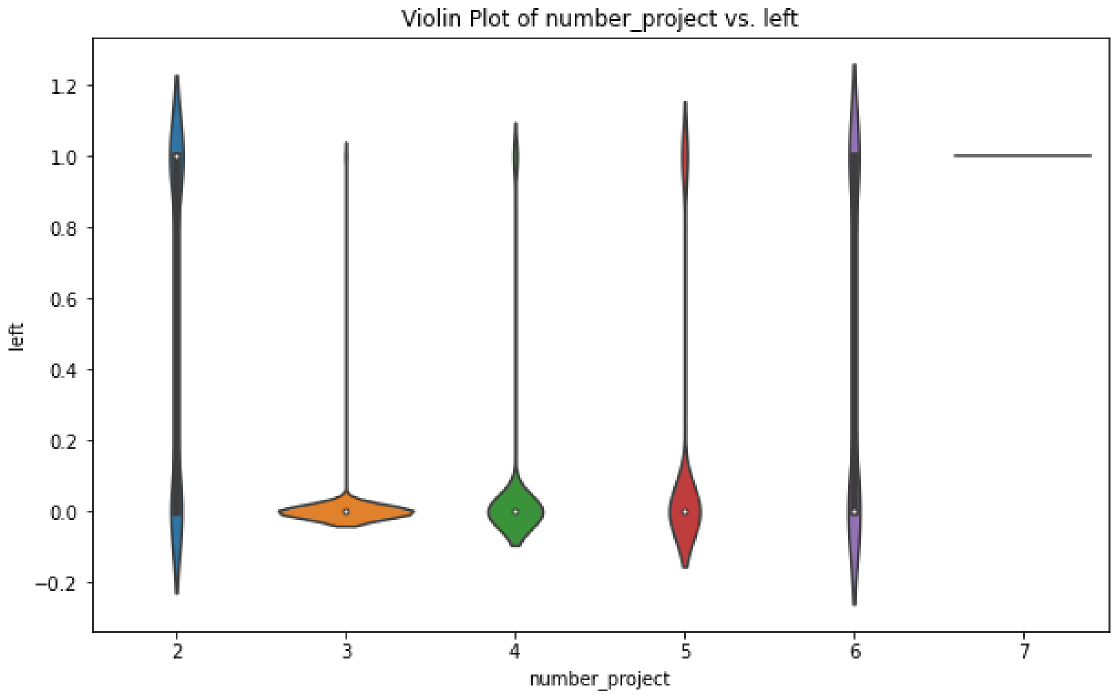
```
# Create a plot as needed
corr = df_dropped.corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.show()
```

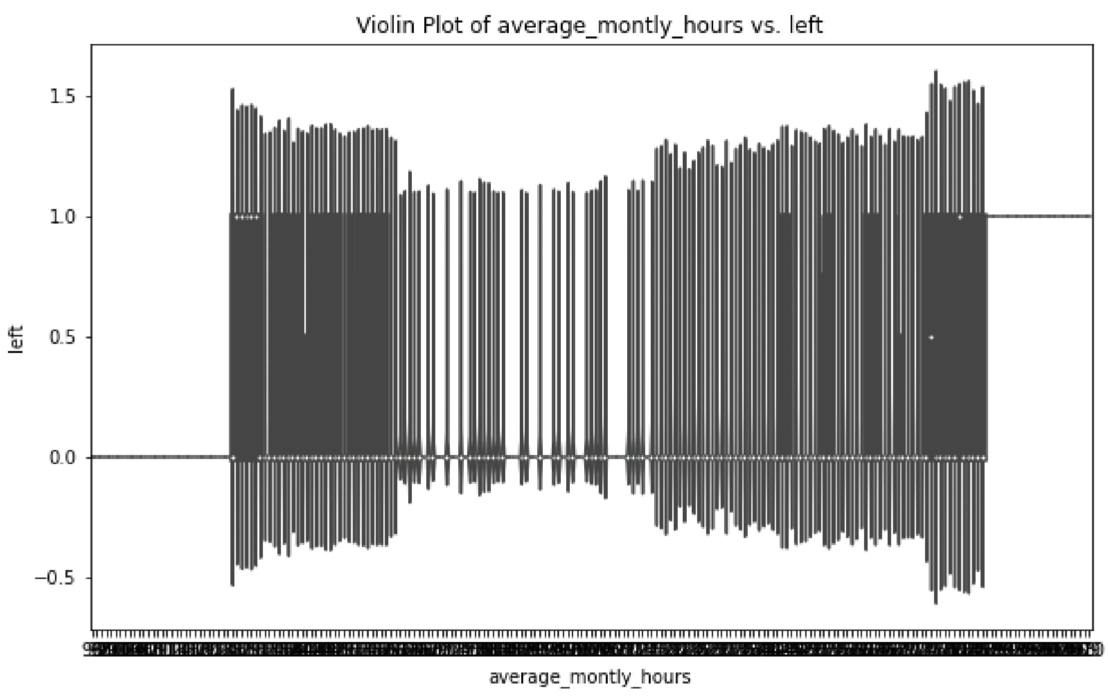


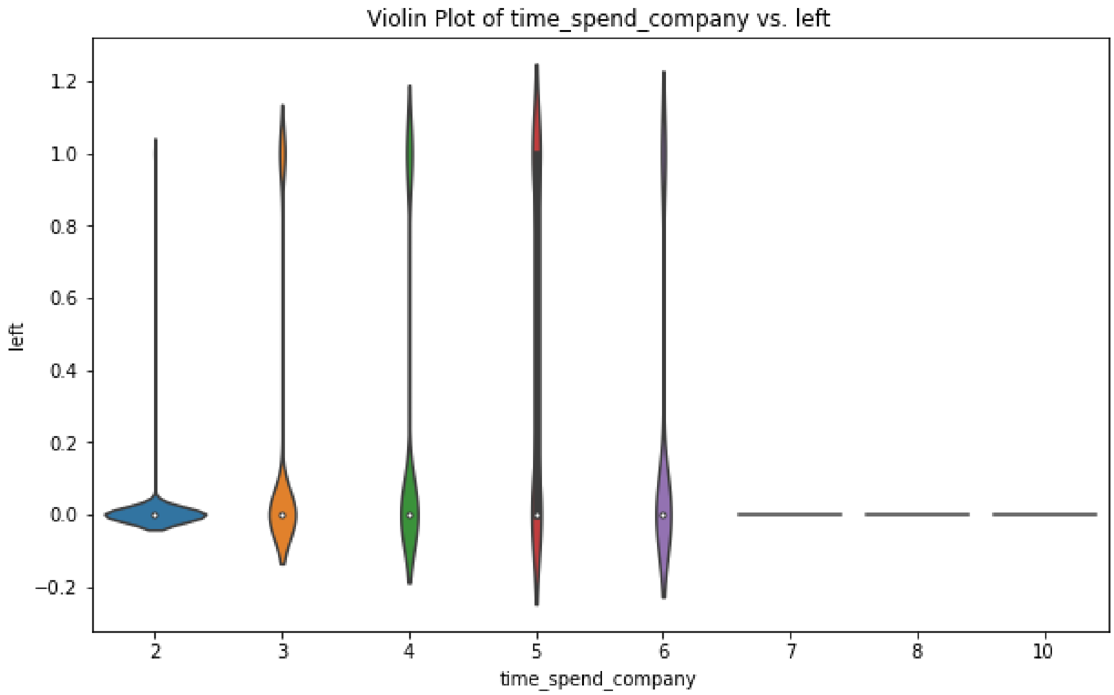
[18]:
```python
# List of variables
variables = ['satisfaction_level', 'last_evaluation', 'number_project',
             'average_montly_hours', 'time_spend_company']

# Target variable
target_var = 'left'

# Loop through each variable for violin plot
for var in variables:
    plt.figure(figsize=(10, 6))  # Adjust figsize as needed
    sns.violinplot(x=var, y=target_var, data=df_dropped)
    plt.xlabel(var)  # Set xlabel based on the variable
    plt.ylabel(target_var)  # Set ylabel based on the target variable
    plt.title(f'Violin Plot of {var} vs. {target_var}')
    plt.show()
```


Violin Plot of satisfaction_level vs. left
left
satisfaction_level


Violin Plot of last_evaluation vs. left
left
last_evaluation


Violin Plot of number_project vs. left
left
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
2
3
4
5
6
7
number_project

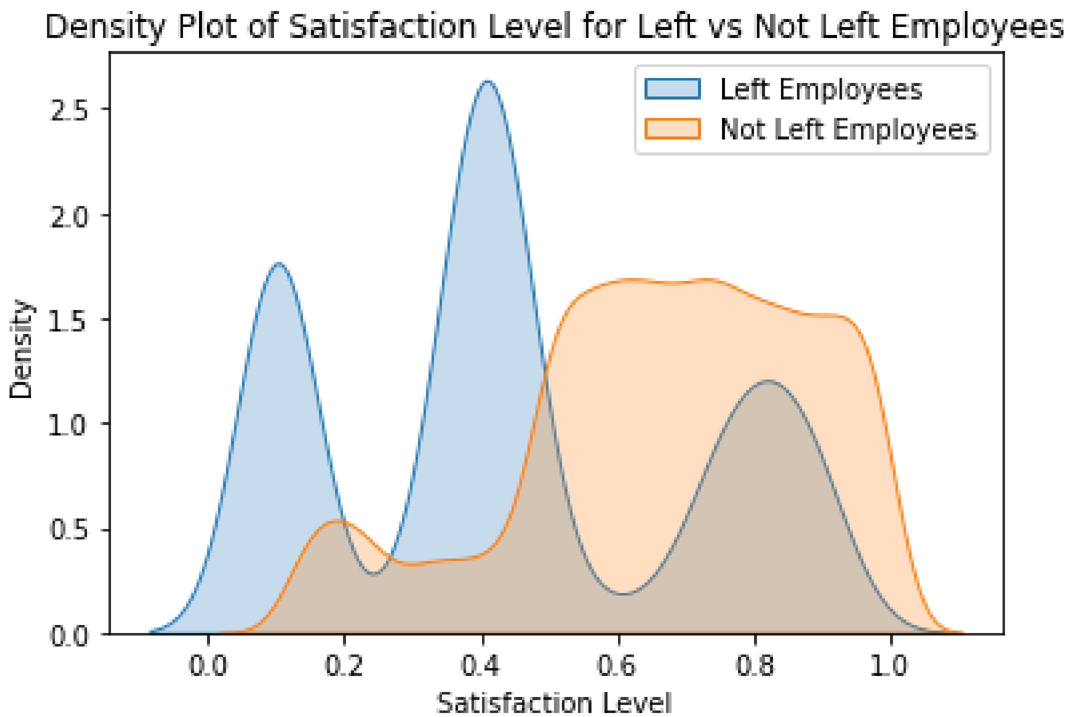

Violin Plot of average_montly_hours vs. left
left
1.5
1.0
0.5
0.0
−0.5
average_montly_hours

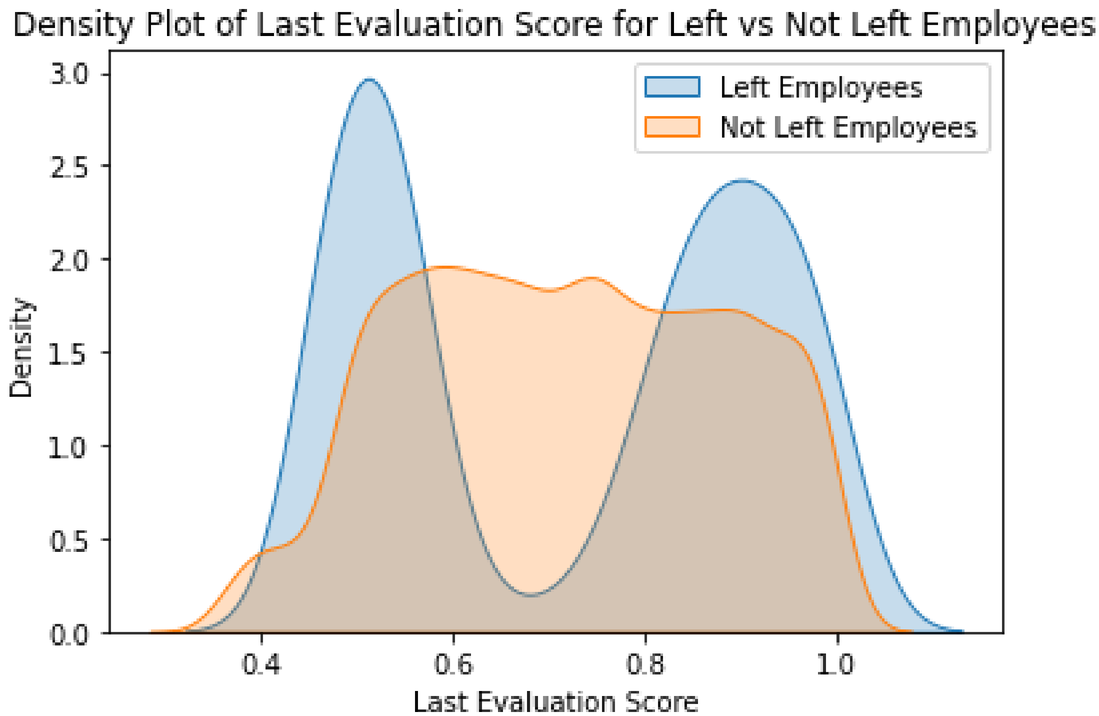


[19]:

```
# Create a plot as needed


# Create data subsets
left_employees = df_dropped[df_dropped['left'] == 1]['satisfaction_level']
not_left_employees = df_dropped[df_dropped['left'] == 0]['satisfaction_level']

# Plot density plot
sns.kdeplot(left_employees, label='Left Employees', shade=True)
sns.kdeplot(not_left_employees, label='Not Left Employees', shade=True)
plt.xlabel('Satisfaction Level')
plt.ylabel('Density')
plt.title('Density Plot of Satisfaction Level for Left vs Not Left Employees')
plt.legend()
plt.show()




# Create data subsets
left_employees1 = df_dropped[df_dropped['left'] == 1]['last_evaluation']
not_left_employees1 = df_dropped[df_dropped['left'] == 0]['last_evaluation']

# Plot density plot
sns.kdeplot(left_employees1, label='Left Employees', shade=True)
```

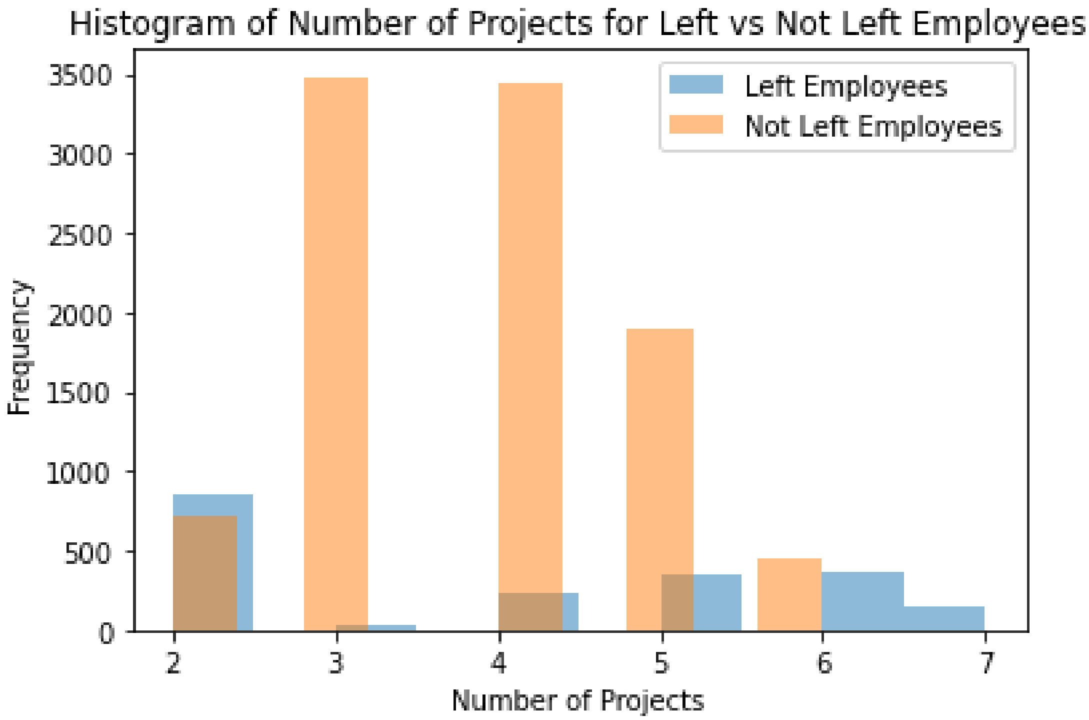
```
sns.kdeplot(not_left_employees1, label='Not Left Employees', shade=True)
plt.xlabel('Last Evaluation Score')
plt.ylabel('Density')
plt.title('Density Plot of Last Evaluation Score for Left vs Not Left␣
 ↪Employees')
plt.legend()
plt.show()
```

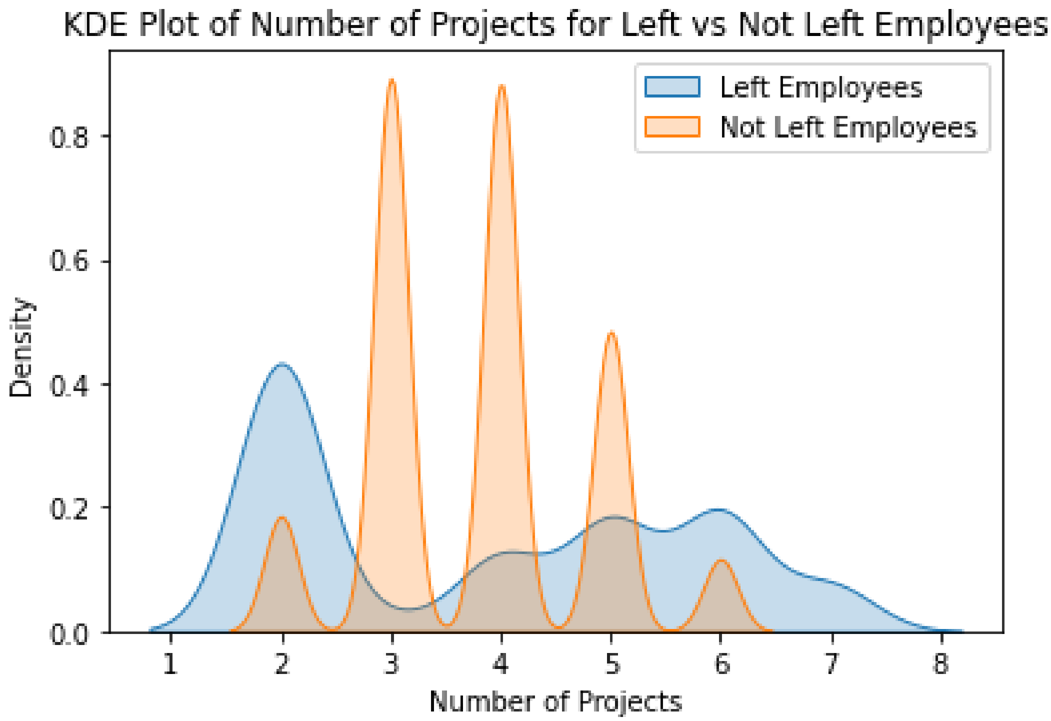




[20]:
```
# Create a plot as needed


# Create data subsets
left_projects = df_dropped[df_dropped['left'] == 1]['number_project']
not_left_projects = df_dropped[df_dropped['left'] == 0]['number_project']

# Plot histograms
plt.hist(left_projects, bins=10, alpha=0.5, label='Left Employees')
plt.hist(not_left_projects, bins=10, alpha=0.5, label='Not Left Employees')
plt.xlabel('Number of Projects')
plt.ylabel('Frequency')
plt.title('Histogram of Number of Projects for Left vs Not Left Employees')
plt.legend()
plt.show()


# Create data subsets
left_projects = df_dropped[df_dropped['left'] == 1]['number_project']
not_left_projects = df_dropped[df_dropped['left'] == 0]['number_project']

# Plot KDE plots
sns.kdeplot(left_projects, label='Left Employees', shade=True)
```

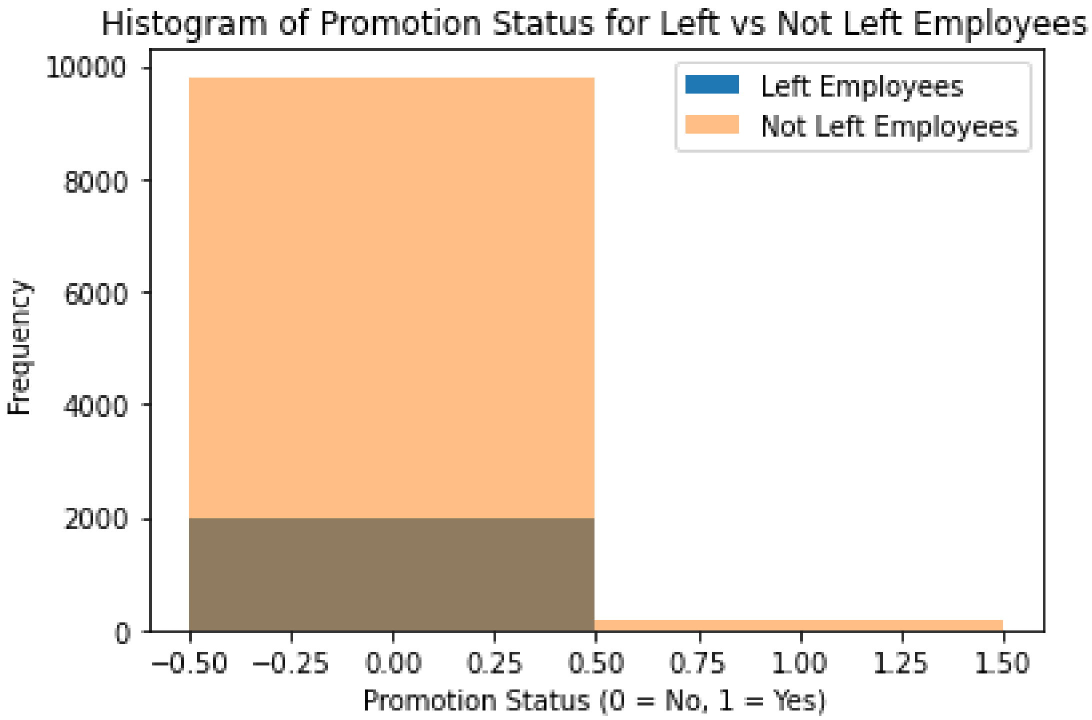
```
sns.kdeplot(not_left_projects, label='Not Left Employees', shade=True)
plt.xlabel('Number of Projects')
plt.ylabel('Density')
plt.title('KDE Plot of Number of Projects for Left vs Not Left Employees')
plt.legend()
plt.show()
```

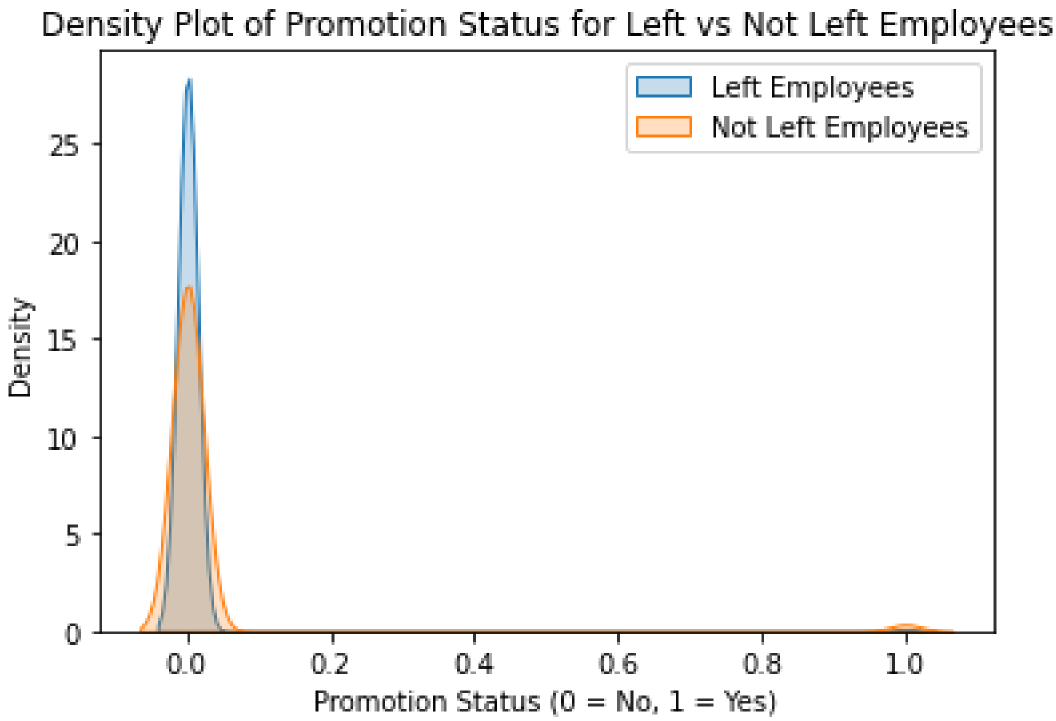




[21]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Create data subsets
left_promotion = df_dropped[df_dropped['left'] == 1]['promotion_last_5years']
not_left_promotion = df_dropped[df_dropped['left'] ==␣
 ↪0]['promotion_last_5years']

# Plot histograms
plt.hist(left_promotion, bins=[-0.5, 0.5, 1.5], alpha=1, label='Left Employees')
plt.hist(not_left_promotion, bins=[-0.5, 0.5, 1.5], alpha=0.5, label='Not Left␣
 ↪Employees')
plt.xlabel('Promotion Status (0 = No, 1 = Yes)')
plt.ylabel('Frequency')
plt.title('Histogram of Promotion Status for Left vs Not Left Employees')
plt.legend()
plt.show()

# Plot KDE plots
sns.kdeplot(left_promotion, label='Left Employees', shade=True)
sns.kdeplot(not_left_promotion, label='Not Left Employees', shade=True)
plt.xlabel('Promotion Status (0 = No, 1 = Yes)')
plt.ylabel('Density')
```

```python
plt.title('Density Plot of Promotion Status for Left vs Not Left Employees')
plt.legend()
plt.show()
```





[22]:

```
# Create a plot as needed
### YOUR CODE HERE ###
```

3.1.2 Insights

[What insights can you gather from the plots you created to visualize the data? Double-click to enter your responses here.]

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?

[Sure.]

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.

[Response variable is categorical.]

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

[Since the output is categorical we can use Binomial Logistic Regression or Tree based models]

4.1.3 Modeling

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

[23]:
```
df_enc = df_dropped.copy()

# Encode the `salary` column as an ordinal numeric category
df_enc['salary'] = (
    df_enc['salary'].astype('category')
    .cat.set_categories(['low', 'medium', 'high'])
    .cat.codes
)

# Dummy encode the `department` column
df_enc = pd.get_dummies(df_enc, drop_first=False)

# Display the new dataframe
df_enc.head()
```

[23]:

```
   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  salary  \
0                   3              0     1                      0       0
1                   6              0     1                      0       1
2                   4              0     1                      0       1
3                   5              0     1                      0       0
4                   3              0     1                      0       0

   department_IT  department_RandD  department_accounting  department_hr  \
0              0                 0                      0              0
1              0                 0                      0              0
2              0                 0                      0              0
3              0                 0                      0              0
4              0                 0                      0              0

   department_management  department_marketing  department_product_mng  \
0                      0                     0                       0
1                      0                     0                       0
2                      0                     0                       0
3                      0                     0                       0
4                      0                     0                       0

   department_sales  department_support  department_technical
0                 1                   0                     0
1                 1                   0                     0
2                 1                   0                     0
3                 1                   0                     0
4                 1                   0                     0
```

[25]:

```python
# Select rows without outliers in `tenure` and save resulting dataframe in a␣
↪new variable
df_logreg = df_enc[(df_enc['time_spend_company'] >= 1.5) &␣
↪(df_enc['time_spend_company'] <= 5.5)]

# Display first few rows of new dataframe
df_logreg.head()
```

[25]:

```
   satisfaction_level  last_evaluation  number_project  average_montly_hours  \
0                0.38             0.53               2                   157
2                0.11             0.88               7                   272
3                0.72             0.87               5                   223
```

```
4                0.37             0.52              2                 159
5                0.41             0.50              2                 153

   time_spend_company  work_accident  left  promotion_last_5years  salary  \
0                   3              0     1                      0       0
2                   4              0     1                      0       1
3                   5              0     1                      0       0
4                   3              0     1                      0       0
5                   3              0     1                      0       0

   department_IT  department_RandD  department_accounting  department_hr  \
0              0                 0                      0              0
2              0                 0                      0              0
3              0                 0                      0              0
4              0                 0                      0              0
5              0                 0                      0              0

   department_management  department_marketing  department_product_mng  \
0                      0                     0                       0
2                      0                     0                       0
3                      0                     0                       0
4                      0                     0                       0
5                      0                     0                       0

   department_sales  department_support  department_technical
0                 1                   0                     0
2                 1                   0                     0
3                 1                   0                     0
4                 1                   0                     0
5                 1                   0                     0
```

[26]:
```python
y = df_logreg['left']
y.head()
```

[26]:
```
0    1
2    1
3    1
4    1
5    1
Name: left, dtype: int64
```

[27]:
```python
X= df_logreg.drop('left',axis =1)
X.head()
```

[27]:
```
   satisfaction_level  last_evaluation  number_project  average_montly_hours  \
0                0.38             0.53               2                   157
2                0.11             0.88               7                   272
```

```
3                0.72             0.87               5                 223
4                0.37             0.52               2                 159
5                0.41             0.50               2                 153

   time_spend_company  work_accident  promotion_last_5years  salary  \
0                   3              0                      0       0
2                   4              0                      0       1
3                   5              0                      0       0
4                   3              0                      0       0
5                   3              0                      0       0

   department_IT  department_RandD  department_accounting  department_hr  \
0              0                 0                      0              0
2              0                 0                      0              0
3              0                 0                      0              0
4              0                 0                      0              0
5              0                 0                      0              0

   department_management  department_marketing  department_product_mng  \
0                      0                     0                       0
2                      0                     0                       0
3                      0                     0                       0
4                      0                     0                       0
5                      0                     0                       0

   department_sales  department_support  department_technical
0                 1                   0                     0
2                 1                   0                     0
3                 1                   0                     0
4                 1                   0                     0
5                 1                   0                     0
```

[28]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,␣
↪stratify=y, random_state=42)

log_clf = LogisticRegression(random_state=42, max_iter=500).fit(X_train,␣
↪y_train)
```

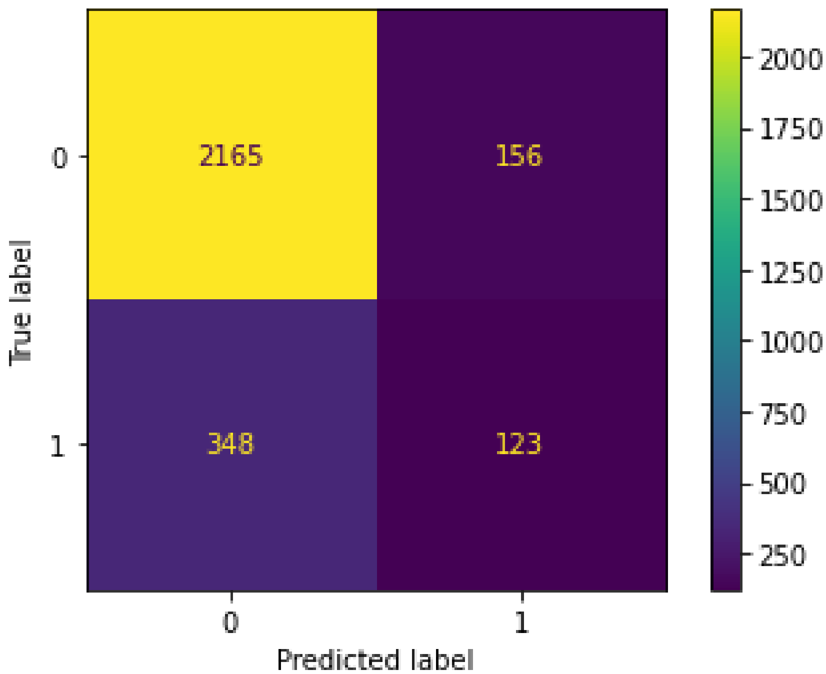
[29]:

```
y_pred = log_clf.predict(X_test)
```

[31]:

```
from sklearn.metrics import accuracy_score, precision_score, recall_score,\
f1_score, confusion_matrix, ConfusionMatrixDisplay, classification_report

# Compute values for confusion matrix
log_cm = confusion_matrix(y_test, y_pred, labels=log_clf.classes_)

# Create display of confusion matrix
```

```
log_disp = ConfusionMatrixDisplay(confusion_matrix=log_cm,
                                  display_labels=log_clf.classes_)

# Plot confusion matrix
log_disp.plot(values_format='')

# Display plot
plt.show()
```



5 pacE: Execute Stage

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

[33]:
```
target_names = ['Predicted would not leave', 'Predicted would leave']
print(classification_report(y_test, y_pred, target_names=target_names))
```

```
                           precision    recall  f1-score   support

Predicted would not leave       0.86      0.93      0.90      2321
    Predicted would leave       0.44      0.26      0.33       471

                 accuracy                           0.82      2792
```

```
   macro avg       0.65      0.60      0.61      2792
weighted avg       0.79      0.82      0.80      2792
```

[ ]:

[ ]:

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?

Double-click to enter your responses here.

5.1 Step 4. Results and Evaluation

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

5.1.1 Summary of model results

[Double-click to enter your summary here.]

5.1.2 Conclusion, Recommendations, Next Steps

[Double-click to enter your conclusion, recommendations, and next steps here.]

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.