Capstone project: Providing data-driven suggestions for HR

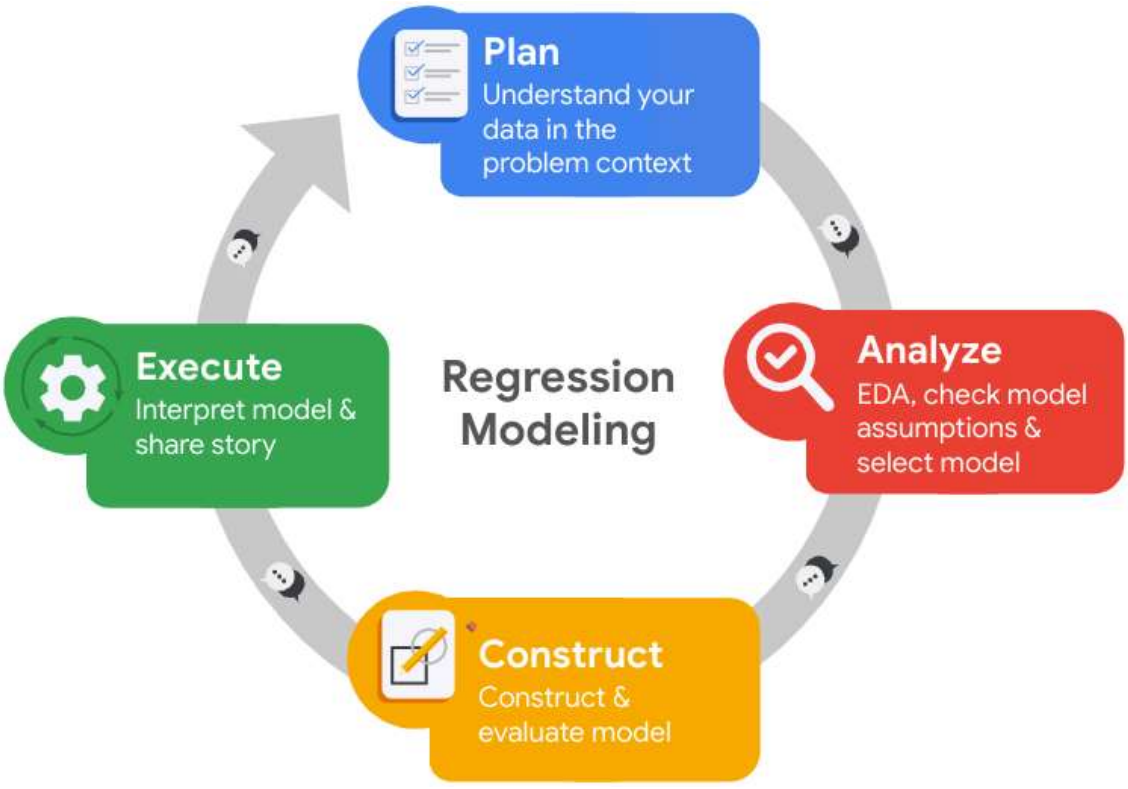
Description and deliverables

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.

PACE stages



Pace: Plan

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

In this stage, consider the following:

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.

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 (https://www.kaggle.com/datasets/mfaisalqureshi/hr-analytics-and-job-prediction?select=HR_comma_sep.csv).

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.]

Step 1. Imports

- Import packages
- Load dataset

Import packages

In [1]:
```
# Import packages
### YOUR CODE HERE ###

import numpy as np
import pandas as pd


import matplotlib.pyplot as plt
import seaborn as sns


pd.set_option('display.max_columns', None)


from xgboost import XGBClassifier
from xgboost import XGBRegressor
from xgboost import plot_importance

from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier


from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.metrics import accuracy_score, precision_score, recall_score,\
f1_score, confusion_matrix, ConfusionMatrixDisplay, classification_report
from sklearn.metrics import roc_auc_score, roc_curve
from sklearn.tree import plot_tree


import pickle
```

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.

In [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
### YOUR CODE HERE ###
df0.head()
```

Out[2]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
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	

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

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

Gather basic information about the data

In [3]:
```
# Gather basic information about the data
### YOUR CODE HERE ###
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
```

Gather descriptive statistics about the data

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

Out[4]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_com
count	14999.000000	14999.000000	14999.000000	14999.000000	14999.0(
mean	0.612834	0.716102	3.803054	201.050337	3.4(
std	0.248631	0.171169	1.232592	49.943099	1.4(
min	0.090000	0.360000	2.000000	96.000000	2.0(
25%	0.440000	0.560000	3.000000	156.000000	3.0(
50%	0.640000	0.720000	4.000000	200.000000	3.0(
75%	0.820000	0.870000	5.000000	245.000000	4.0(
max	1.000000	1.000000	7.000000	310.000000	10.0(

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.

In [5]:
```
# Display all column names
### YOUR CODE HERE ###
df0.columns
```

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

In [6]:
```
# Rename columns as needed
### YOUR CODE HERE ###
df0 = df0.rename(columns={'Work_accident': 'work_accident',
                          'average_montly_hours': 'average_monthly_hours',
                          'time_spend_company': 'tenure',
                          'Department': 'department'})


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

Out[6]:
```
Index(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_monthly_hours', 'tenure', 'work_accident', 'left',
       'promotion_last_5years', 'department', 'salary'],
      dtype='object')
```

Check missing values

Check for any missing values in the data.

In [7]:
```
# Check for missing values
### YOUR CODE HERE ###
df0.isna().sum()
```

Out[7]:
```
satisfaction_level       0
last_evaluation          0
number_project           0
average_monthly_hours    0
tenure                   0
work_accident            0
left                     0
promotion_last_5years    0
department               0
salary                   0
dtype: int64
```

Check duplicates

Check for any duplicate entries in the data.

In [8]:
```
# Check for duplicates
### YOUR CODE HERE ###
df0.duplicated().sum()
```

Out[8]:
```
3008
```

In [9]:
```
# Inspect some rows containing duplicates as needed
### YOUR CODE HERE ###
df0[df0.duplicated()].head()
```

Out[9]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	tenure	work_ac
396	0.46	0.57	2	139	3	
866	0.41	0.46	2	128	3	
1317	0.37	0.51	2	127	3	
1368	0.41	0.52	2	132	3	
1461	0.42	0.53	2	142	3	

In [10]:
```
# Drop duplicates and save resulting dataframe in a new variable as needed
### YOUR CODE HERE ###
df1 = df0.drop_duplicates(keep='first')

# Display first few rows of new dataframe as needed
### YOUR CODE HERE ###
df1.head()
```

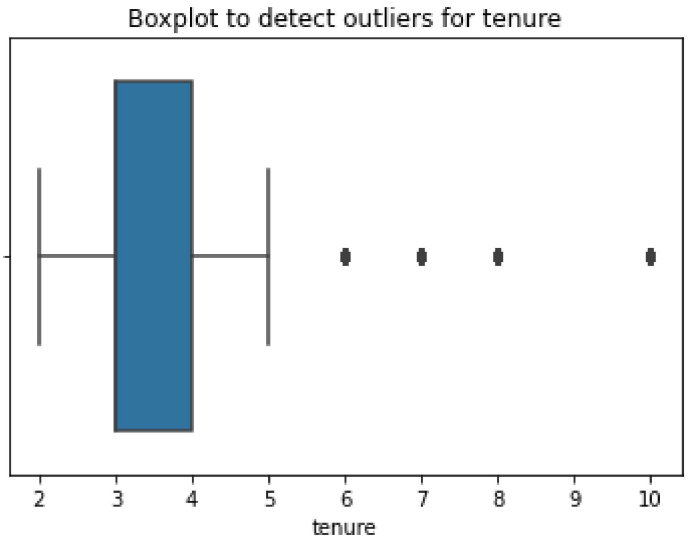
Out[10]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	tenure	work_accid
0	0.38	0.53	2	157	3	
1	0.80	0.86	5	262	6	
2	0.11	0.88	7	272	4	
3	0.72	0.87	5	223	5	
4	0.37	0.52	2	159	3	

Check outliers

Check for outliers in the data.

In [14]:
```
# Create a boxplot to visualize distribution of `tenure` and detect any outlie
### YOUR CODE HERE ###
plt.figure
plt.title('Boxplot to detect outliers for tenure', fontsize=12)
plt.xticks
plt.yticks
sns.boxplot(x=df1['tenure'])
plt.show()
```



In [17]:
```
# Determine the number of rows containing outliers
### YOUR CODE HERE ###
percentile25 = df1['tenure'].quantile(0.25)
percentile75 = df1['tenure'].quantile(0.75)
iqr = percentile75 - percentile25

upper_limit = percentile75 + 1.5 * iqr
lower_limit = percentile25 - 1.5 * iqr
print("Lower limit:", lower_limit)
print("Upper limit:", upper_limit)

outliers = df1[(df1['tenure'] > upper_limit) | (df1['tenure'] < lower_limit)]

print("Number of rows in the data containing outliers in `tenure`:", len(outli
```

```
Lower limit: 1.5
Upper limit: 5.5
Number of rows in the data containing outliers in `tenure`: 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.

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.]

Step 2. Data Exploration (Continue EDA)

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

In [18]:
```
# Get numbers of people who left vs. stayed
### YOUR CODE HERE ###
print(df1['left'].value_counts())
print()

# Get percentages of people who left vs. stayed
### YOUR CODE HERE ###
print(df1['left'].value_counts(normalize=True))
```

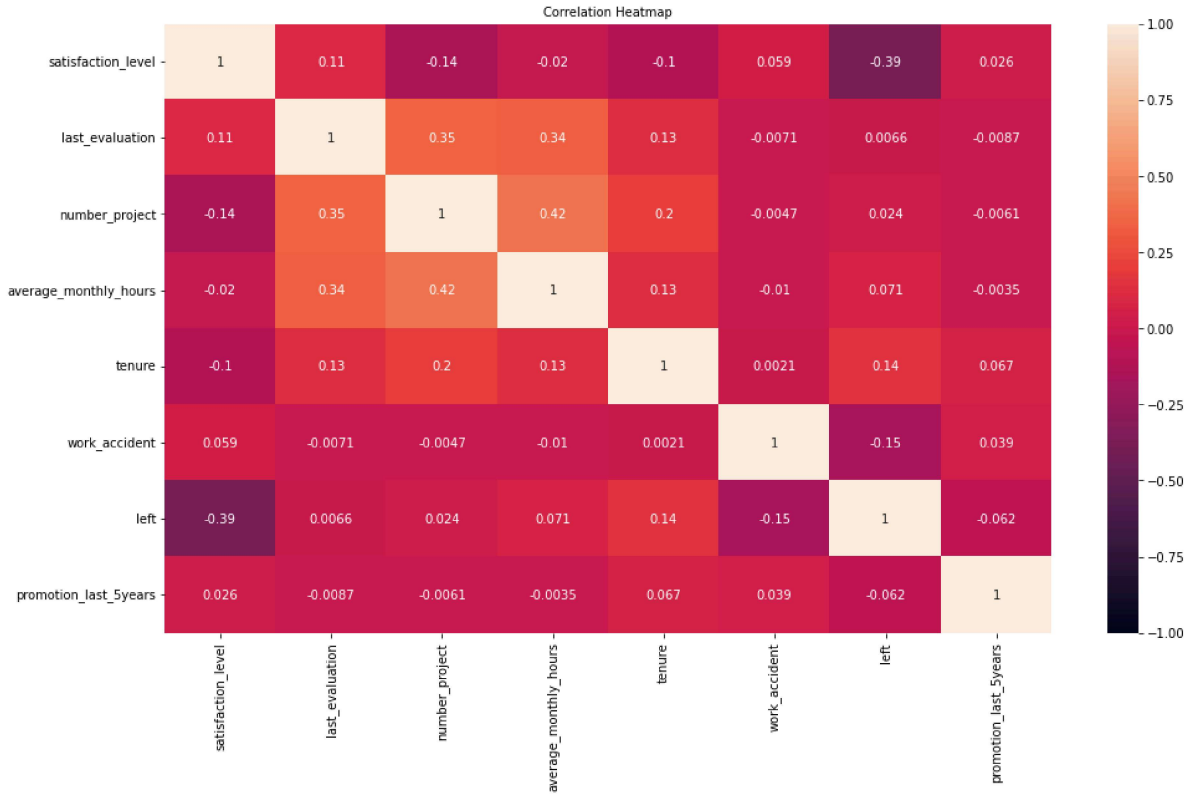
```
0    10000
1     1991
Name: left, dtype: int64

0    0.833959
1    0.166041
Name: left, dtype: float64
```

Data visualizations

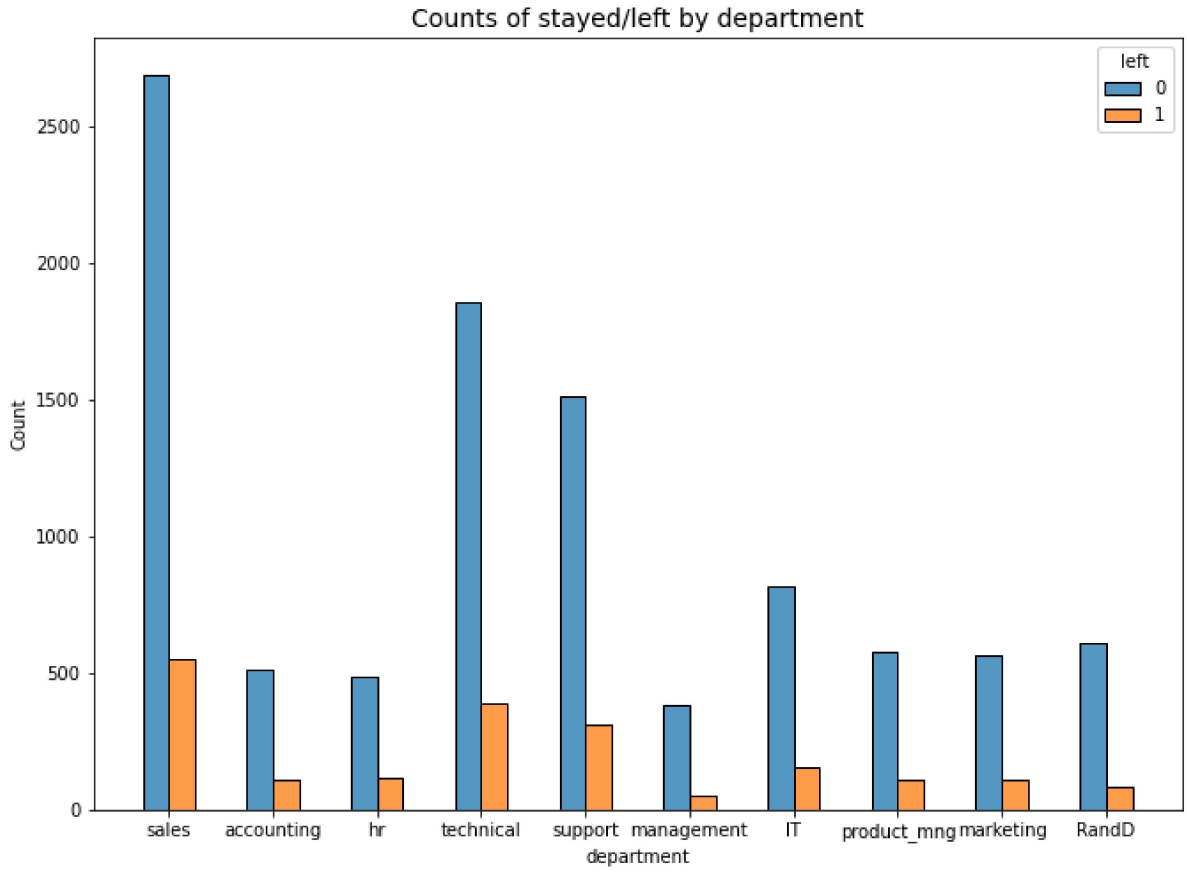
Now, examine variables that you're interested in, and create plots to visualize relationships between variables in the data.

In [31]:
```
# Create a plot as needed
### YOUR CODE HERE ###
plt.figure(figsize=(16, 9))
heatmap = sns.heatmap(df0.corr(), vmin=-1 ,annot=True)
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':10})
```

Out[31]: Text(0.5, 1.0, 'Correlation Heatmap')

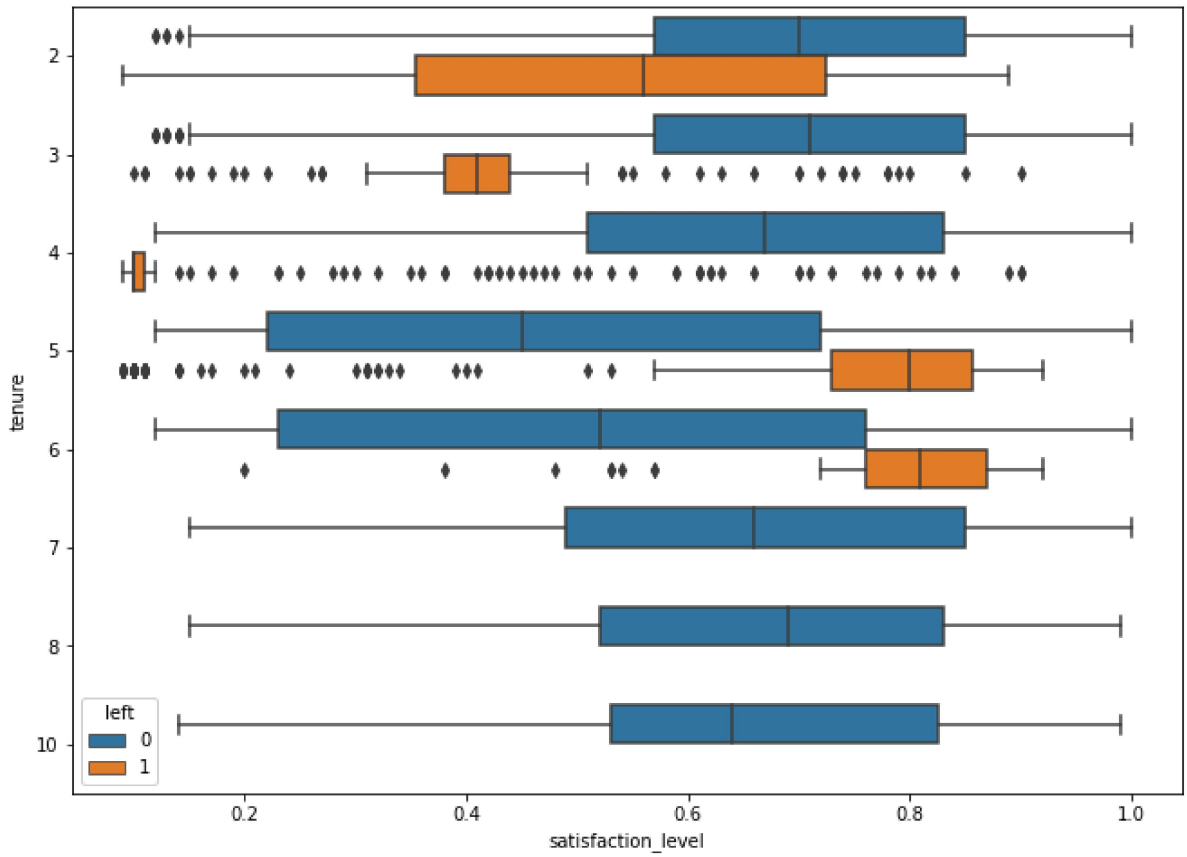


In [52]:
```
# Create a plot as needed
### YOUR CODE HERE ###
plt.figure(figsize = (11,8))
sns.histplot(data=df1, x='department',hue = 'left',multiple = 'dodge',shrink=0

plt.title('Counts of stayed/left by department', fontsize=14);
```



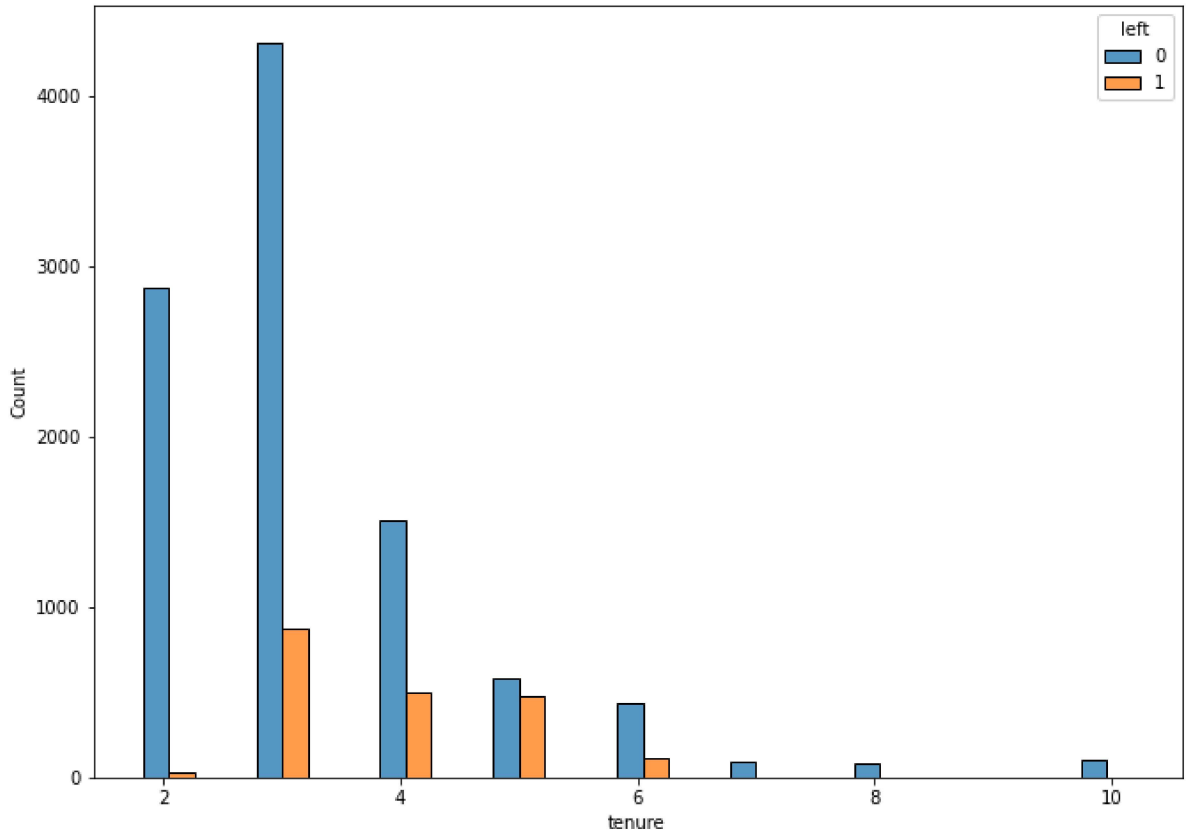
In [56]:
```
# Create a plot as needed
### YOUR CODE HERE ###
plt.figure(figsize = (11,8))
sns.boxplot(data=df1, x='satisfaction_level', y='tenure', hue='left', orient="
```

Out[56]: <matplotlib.axes._subplots.AxesSubplot at 0x7703a59bd590>



In [58]:
```
# Create a plot as needed
### YOUR CODE HERE ###
plt.figure(figsize = (11,8))
sns.histplot(data=df1, x='tenure', hue='left', multiple='dodge', shrink=5)
```

Out[58]: <matplotlib.axes._subplots.AxesSubplot at 0x7703a5657bd0>



Insights

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

1. People leaving had negative co relation with satisfaction level showing that lower satisfaction level in workplace made people leave.
2. Workplace accidents also had same negative co relation.
3. No evidence as to any particular department had many eployees leaving.
4. People leaving are usually with less tenure.
5. People with high tenure are very less maybe due to their high positions.

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?

[Double-click to enter your responses here.]

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

Identify the type of prediction task.

[Double-click to enter your responses here.]

Identify the types of models most appropriate for this task.

[Double-click to enter your responses here.]

Modeling

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

In [60]:
```
### YOUR CODE HERE ###
df_enc = df1.copy()

df_enc['salary'] = (
    df_enc['salary'].astype('category')
    .cat.set_categories(['low', 'medium', 'high'])
    .cat.codes)

df_enc = pd.get_dummies(df_enc, drop_first=False)

df_enc.head()
```

Out[60]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	tenure	work_accid
0	0.38	0.53	2	157	3	
1	0.80	0.86	5	262	6	
2	0.11	0.88	7	272	4	
3	0.72	0.87	5	223	5	
4	0.37	0.52	2	159	3	

In [62]:
```
df_lgrg = df_enc[(df_enc['tenure'] >= lower_limit) & (df_enc['tenure'] <= uppe
df_lgrg.head()
```

Out[62]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	tenure	work_accid
0	0.38	0.53	2	157	3	
2	0.11	0.88	7	272	4	
3	0.72	0.87	5	223	5	
4	0.37	0.52	2	159	3	
5	0.41	0.50	2	153	3	

In [64]:
```
y = df_lgrg['left']
y
```

Out[64]:
```
0        1
2        1
3        1
4        1
5        1
        ..
11985    0
11986    0
11987    0
11992    0
11999    0
Name: left, Length: 11167, dtype: int64
```

In [67]:
```
X = df_lgrg.drop('left', axis=1)
X.head()
```

Out[67]:

	satisfaction_level	last_evaluation	number_project	average_monthly_hours	tenure	work_accid
0	0.38	0.53	2	157	3	
2	0.11	0.88	7	272	4	
3	0.72	0.87	5	223	5	
4	0.37	0.52	2	159	3	
5	0.41	0.50	2	153	3	

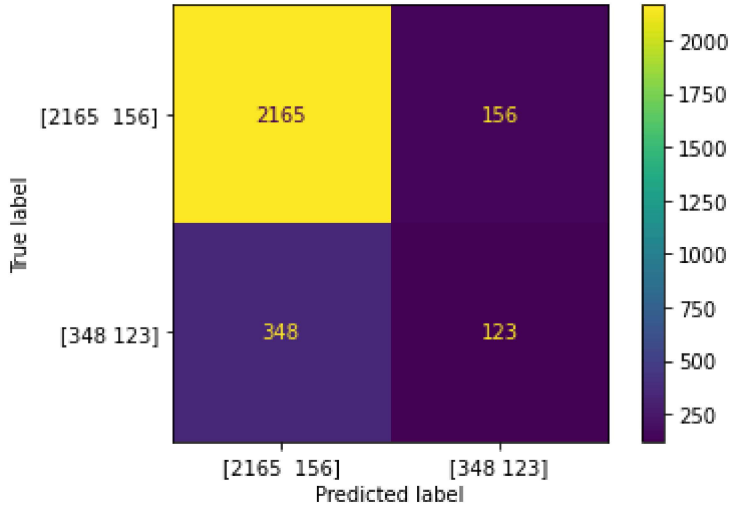
In [68]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, stra
```

In [69]:
```
log_reg = LogisticRegression(random_state=42, max_iter=500).fit(X_train, y_tra
```

In [71]:
```
y_pred = log_reg.predict(X_test)
```

In [73]:
```
log_cmtr = confusion_matrix(y_test, y_pred, labels=log_reg.classes_)
```

In [81]:
```
log_disp = ConfusionMatrixDisplay(confusion_matrix=log_cmtr, display_labels=lo
log_disp.plot(values_format='')
plt.show()
```



In [82]:
```
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
```

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?

Double-click to enter your responses here.

1. My model has high recall and f1 scores for the people who would not leave the company.
2. Workplace enviornment and work hours should be balanced.
3. Promote deserving people based on their workng hours.
4. Yes maybe the model can be improved by a decision tree.
5. What reasons have the satisfaction level drop in last 4 years.
6. The course material itself.
7. Nope.

Step 4. Results and Evaluation

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

Summary of model results

The logistic regression model achieved precision of 80%, recall of 83%, f1-score of 80% , and accuracy of 83%.

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.