In [1]:
```
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report,accuracy_score
from sklearn.preprocessing import StandardScaler
```

In [2]:
```
# Load the dataset
df = pd.read_csv('HR_comma_sep.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work
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 [4]:
```
df.shape
```

Out[4]:
```
(14999, 10)
```

In [5]:
```
df.tail()
```

Out[5]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
14994	0.40	0.57	2	151	3
14995	0.37	0.48	2	160	3
14996	0.37	0.53	2	143	3
14997	0.11	0.96	6	280	4
14998	0.37	0.52	2	158	3

In [6]:
```
df.info()
```

Out[6]:
```
<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
```

In [7]:
```
df.describe()
```

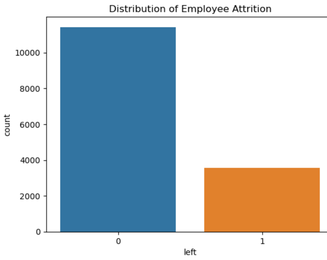
Out[7]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
count	14999.000000	14999.000000	14999.000000	14999.000000	14999.000000
mean	0.612834	0.716102	3.803054	201.050337	3.498233
std	0.248631	0.171169	1.232592	49.943099	1.460136
min	0.090000	0.360000	2.000000	96.000000	2.000000
25%	0.440000	0.560000	3.000000	156.000000	3.000000
50%	0.640000	0.720000	4.000000	200.000000	3.000000
75%	0.820000	0.870000	5.000000	245.000000	4.000000
max	1.000000	1.000000	7.000000	310.000000	10.000000

In [8]:
```
# Check for missing values
print(df.isnull().sum())
#no missing values
```

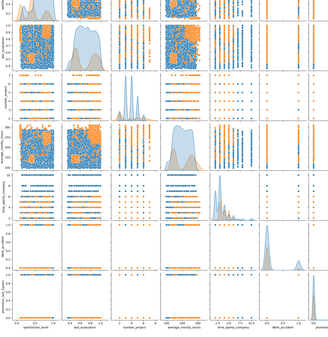
Out[8]:
```
satisfaction_level       0
last_evaluation          0
number_project           0
average_montly_hours     0
time_spend_company       0
Work_accident            0
left                     0
promotion_last_5years    0
Department               0
salary                   0
dtype: int64
```

In [9]:
```
# Visualize the distribution of the target variable
sns.countplot(x='left', data=df)
plt.title('Distribution of Employee Attrition')
plt.show()
```

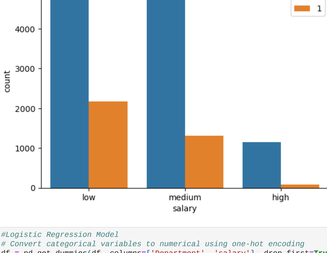


In [10]:
```
# Explore relationships between variables
sns.pairplot(df, hue='left')
plt.show()
#used to create a grid of scatterplots for all pairs of numeric features
#in the DataFrame.
#each scatterplot represents the relationship between two variables.
```

Out[10]:
```
C:\Users\Vaishnavi\anaconda3\lib\site-packages\seaborn\axisgrid.py:118: UserWarnin
g: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



In [11]:
```
# Explore categorical variables
sns.countplot(x='salary', hue='left', data=df)
plt.title('Attrition by Salary Level')
plt.show()
```



In [12]:
```
#Logistic Regression Model
# Convert categorical variables to numerical using one-hot encoding
df = pd.get_dummies(df, columns=['Department', 'salary'], drop_first=True)
```

In [13]:
```
# Split the data into features (X) and target variable (y)
X = df.drop('left', axis=1)
y = df['left']
```

In [14]:
```
#Split Data into Training and Testing
xtrain,xtest,ytrain,ytest=train_test_split(X,y,test_size=0.3,random_state=1)
```

In [15]:
```
#Build and Train Logistic Regression Model
lr= LogisticRegression()
lr.fit(xtrain,ytrain)
```

Out[15]:
```
C:\Users\Vaishnavi\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:4
60: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

▾ LogisticRegression

LogisticRegression()

In [16]:
```
#Make Predictions:
ypred=lr.predict(xtest)
```

In [18]:
```
#Evaluation
print(confusion_matrix(ytest,ypred))
print(classification_report(ytest,ypred))
print('Accuracy: ', accuracy_score(ytest,ypred))
```

Out[18]:
```
[[3125  291]
 [ 645  439]]
              precision    recall  f1-score   support

           0       0.83      0.91      0.87      3416
           1       0.60      0.40      0.48      1084

    accuracy                           0.79      4500
   macro avg       0.72      0.66      0.68      4500
weighted avg       0.77      0.79      0.78      4500

Accuracy:  0.792
```

In [ ]:
```

```

In [ ]:
```

```