# Employee Turnover Analytics. Course-end Project 4

September 20, 2023

[1]:
```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')
```

[3]:
```
from sklearn.model_selection import StratifiedShuffleSplit
from imblearn.over_sampling import SMOTE
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.model_selection import cross_val_score, cross_val_predict
from sklearn.metrics import classification_report, confusion_matrix,␣
↪ConfusionMatrixDisplay
```

[4]:
```
df = pd.read_excel('1673873196_hr_comma_sep.xlsx')
```

[5]:
```
df.head()
```

[5]:

| | satisfaction_level | last_evaluation | number_project | average_montly_hours | time_spend_company | Work_accident | left | promotion_last_5years | sales |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0.38 | 0.53 | 2 | 157 | 3 | 0 | 1 | 0 | sales |
| 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 | | | | | |

```
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
```

[6]:
```
df.tail()
```

[6]:
```
       satisfaction_level  last_evaluation  number_project  \
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  \
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
14994                      0  support    low
14995                      0  support    low
14996                      0  support    low
14997                      0  support    low
14998                      0  support    low
```

[7]:
```
df.shape
```

[7]:
```
(14999, 10)
```

Perform data quality check by checking for missing values if any.

[8]:
```
df.isna().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
sales                    0
salary                   0
dtype: int64
```

[10]:
```
df.describe().T
```

[10]:
```
                         count        mean        std    min     25%     50%  \
satisfaction_level     14999.0    0.612834   0.248631   0.09    0.44    0.64
last_evaluation        14999.0    0.716102   0.171169   0.36    0.56    0.72
number_project         14999.0    3.803054   1.232592   2.00    3.00    4.00
average_montly_hours   14999.0  201.050337  49.943099  96.00  156.00  200.00
time_spend_company     14999.0    3.498233   1.460136   2.00    3.00    3.00
Work_accident          14999.0    0.144610   0.351719   0.00    0.00    0.00
left                   14999.0    0.238083   0.425924   0.00    0.00    0.00
promotion_last_5years  14999.0    0.021268   0.144281   0.00    0.00    0.00

                          75%    max
satisfaction_level       0.82    1.0
last_evaluation          0.87    1.0
number_project           5.00    7.0
average_montly_hours   245.00  310.0
time_spend_company       4.00   10.0
Work_accident            0.00    1.0
left                     0.00    1.0
promotion_last_5years    0.00    1.0
```

[11]:
```
df['sales'].value_counts()
```

[11]:
```
sales          4140
technical      2720
support        2229
IT             1227
product_mng     902
marketing       858
RandD           787
accounting      767
hr              739
management      630
Name: sales, dtype: int64
```

[12]:
```
df.rename(columns={'sales':'department'},inplace=True)
df.columns
```

[12]:
```
Index(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_montly_hours', 'time_spend_company', 'Work_accident', 'left',
       'promotion_last_5years', 'department', 'salary'],
      dtype='object')
```

[13]:
```
df['salary'].value_counts()
```

[13]:
```
low       7316
medium    6446
high      1237
Name: salary, dtype: int64
```

Understand what factors contributed most to employee turnover by EDA.
Draw a heatmap of the Correlation Matrix between all numerical features/columns in the data.

[14]:
```
corr = df.corr()
corr
```

[14]:

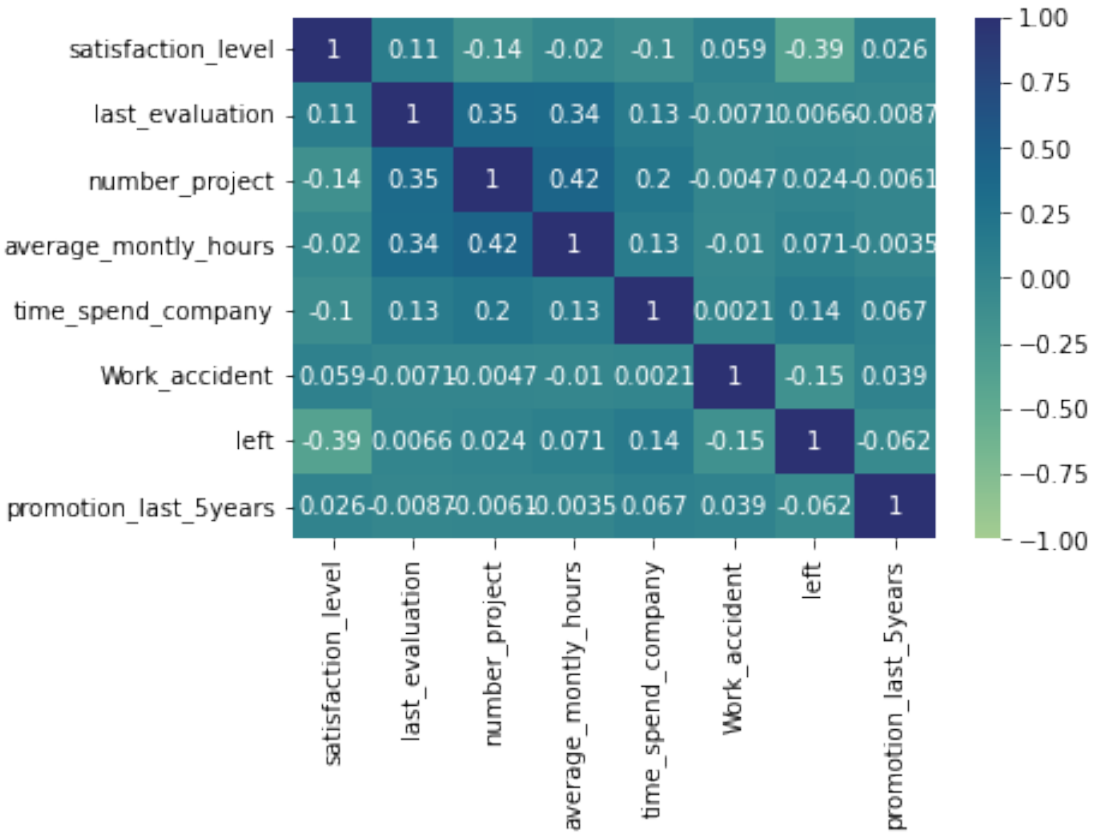
| | satisfaction_level | last_evaluation | number_project | \ |
|---|---|---|---|---|
| satisfaction_level | 1.000000 | 0.105021 | -0.142970 | |
| last_evaluation | 0.105021 | 1.000000 | 0.349333 | |
| number_project | -0.142970 | 0.349333 | 1.000000 | |
| average_montly_hours | -0.020048 | 0.339742 | 0.417211 | |
| time_spend_company | -0.100866 | 0.131591 | 0.196786 | |
| Work_accident | 0.058697 | -0.007104 | -0.004741 | |
| left | -0.388375 | 0.006567 | 0.023787 | |
| promotion_last_5years | 0.025605 | -0.008684 | -0.006064 | |

| | average_montly_hours | time_spend_company | \ |
|---|---|---|---|
| satisfaction_level | -0.020048 | -0.100866 | |
| last_evaluation | 0.339742 | 0.131591 | |
| number_project | 0.417211 | 0.196786 | |
| average_montly_hours | 1.000000 | 0.127755 | |
| time_spend_company | 0.127755 | 1.000000 | |
| Work_accident | -0.010143 | 0.002120 | |
| left | 0.071287 | 0.144822 | |
| promotion_last_5years | -0.003544 | 0.067433 | |

| | Work_accident | left | promotion_last_5years |
|---|---|---|---|
| satisfaction_level | 0.058697 | -0.388375 | 0.025605 |
| last_evaluation | -0.007104 | 0.006567 | -0.008684 |
| number_project | -0.004741 | 0.023787 | -0.006064 |
| average_montly_hours | -0.010143 | 0.071287 | -0.003544 |
| time_spend_company | 0.002120 | 0.144822 | 0.067433 |
| Work_accident | 1.000000 | -0.154622 | 0.039245 |
| left | -0.154622 | 1.000000 | -0.061788 |
| promotion_last_5years | 0.039245 | -0.061788 | 1.000000 |

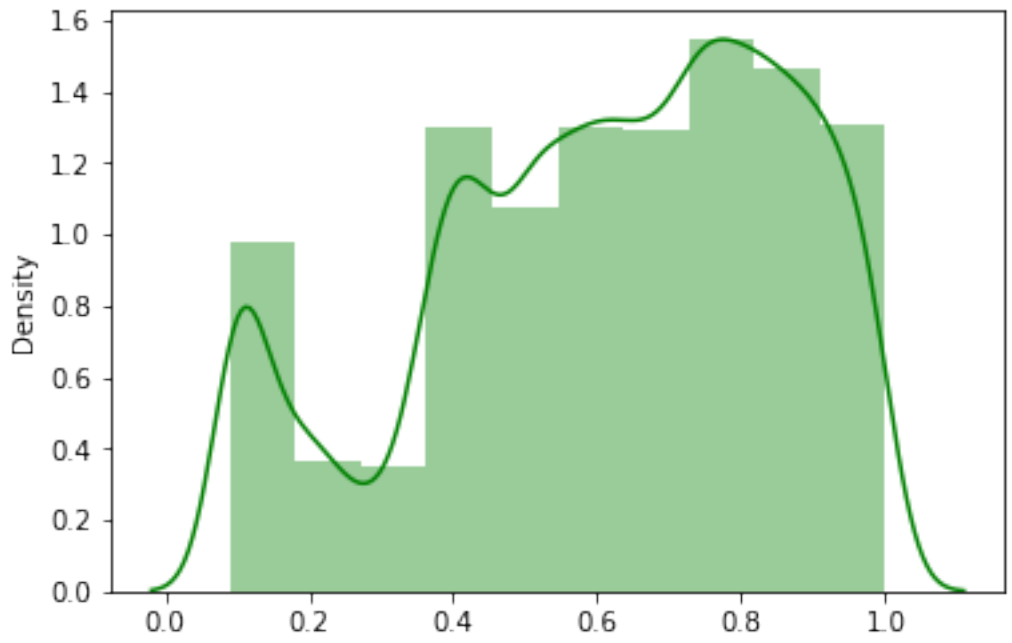
[15]:
```
sns.heatmap(corr, annot=True, vmin=-1,vmax=+1,cmap="crest")
```

[15]: <AxesSubplot: >



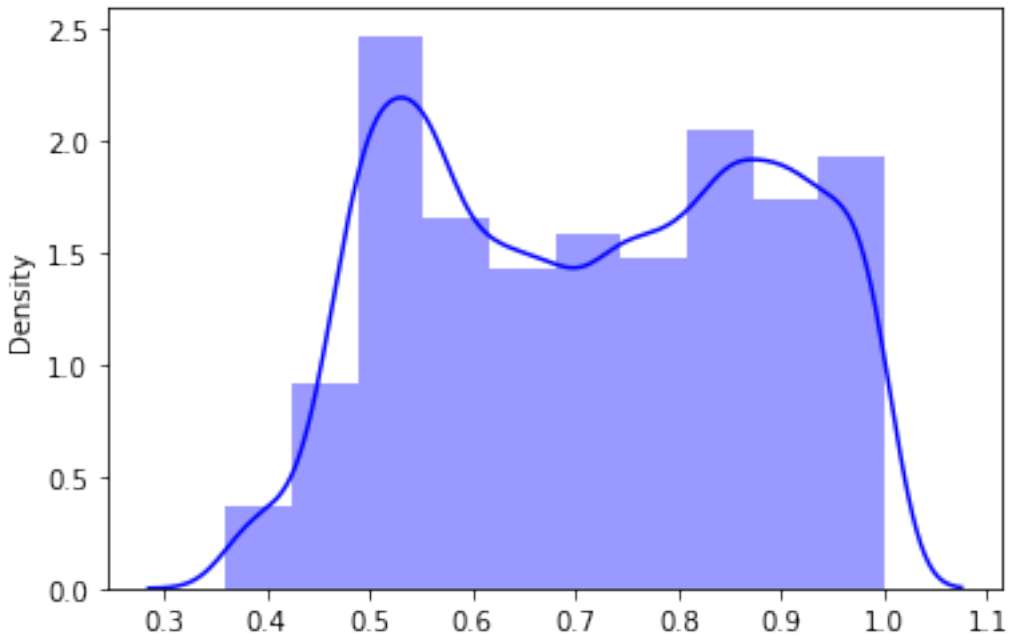

[16]:
```
sns.distplot(x=df['satisfaction_level'], hist=True,bins=10 ,color='green')
```

[16]: <AxesSubplot: ylabel='Density'>



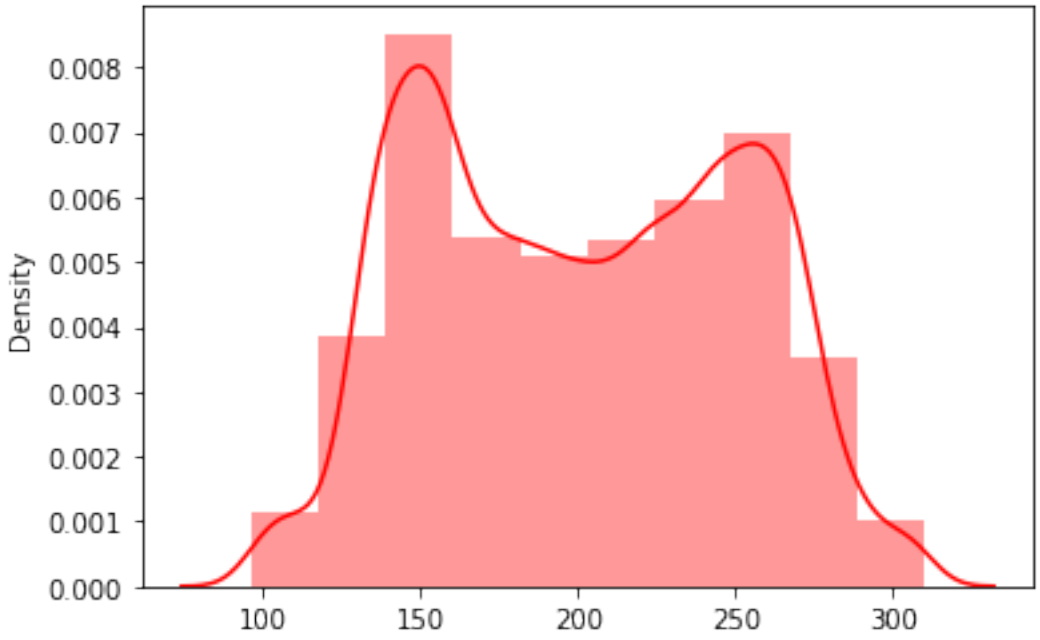

[17]:
```
sns.distplot(x=df['last_evaluation'], hist=True,bins=10 ,color='blue')
```

[17]: <AxesSubplot: ylabel='Density'>



[18]:
```
sns.distplot(x=df['average_montly_hours'], hist=True,bins=10 ,color='red')
```

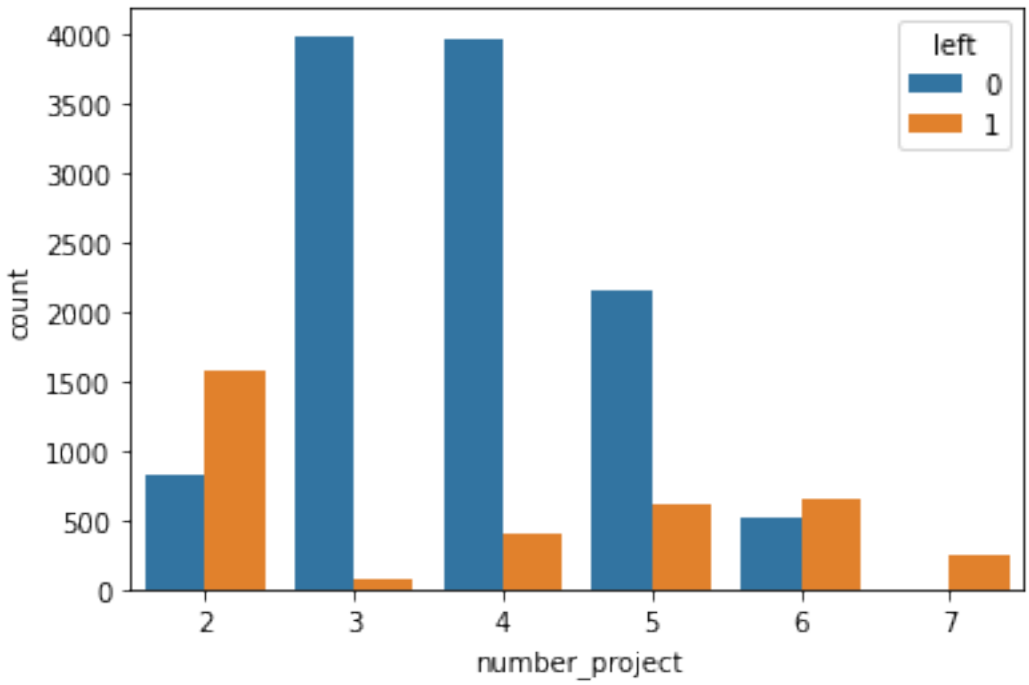
[18]:
```
<AxesSubplot: ylabel='Density'>
```




Draw the bar plot of Employee Project Count of both employees who left and who stayed in the organization

[19]:
```
sns.countplot(data=df, x='number_project', hue='left')
```

[19]:
```
<AxesSubplot: xlabel='number_project', ylabel='count'>
```




Perform clustering of Employees who left based on their satisfaction and evaluation.

[20]:
```
cluster = df[['satisfaction_level', 'last_evaluation', 'left']].copy()
cluster.head()
```

[20]:

| | satisfaction_level | last_evaluation | left |
|---|---|---|---|
| 0 | 0.38 | 0.53 | 1 |
| 1 | 0.80 | 0.86 | 1 |
| 2 | 0.11 | 0.88 | 1 |
| 3 | 0.72 | 0.87 | 1 |
| 4 | 0.37 | 0.52 | 1 |

[21]:
```
cluster['left'].value_counts()
```

[21]:
```
0    11428
1     3571
Name: left, dtype: int64
```

[22]:
```
cluster.shape
```

[22]:
```
(14999, 3)
```

[23]:
```
from sklearn.cluster import KMeans
kmc = KMeans(n_clusters=3, random_state=42)
y_predicted = kmc.fit_predict(cluster)
```

```
y_predicted
```

[23]:
```
array([1, 1, 1, …, 1, 1, 1], dtype=int32)
```

[24]:
```
cluster['cluster']=y_predicted
cluster.sample(5)
```

[24]:

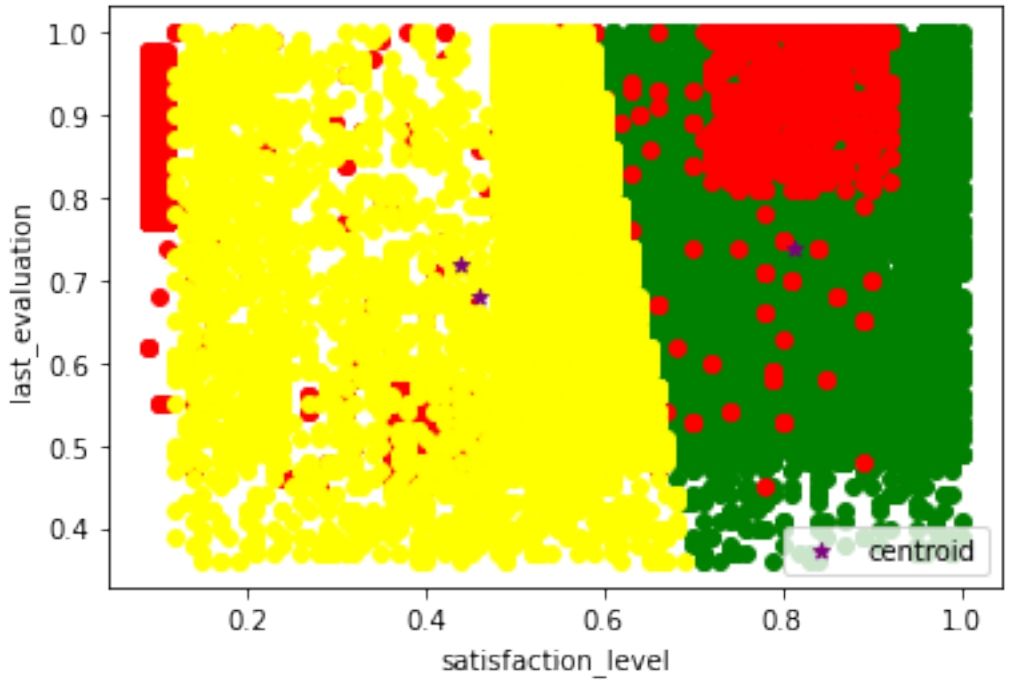
| | satisfaction_level | last_evaluation | left | cluster |
|---|---|---|---|---|
| 11417 | 0.16 | 0.46 | 0 | 2 |
| 14901 | 0.11 | 0.93 | 1 | 1 |
| 1327 | 0.42 | 0.47 | 1 | 1 |
| 14425 | 0.40 | 0.53 | 1 | 1 |
| 11813 | 0.39 | 0.91 | 0 | 2 |

[25]:
```
kmc.cluster_centers_
```

[25]:
```
array([[8.13757657e-01, 7.40231585e-01, 1.62647673e-14],
       [4.40098012e-01, 7.18112574e-01, 1.00000000e+00],
       [4.59096093e-01, 6.80477297e-01, 1.25455202e-14]])
```

[26]:
```
cluster1 = cluster[cluster.cluster==0]
cluster2 = cluster[cluster.cluster==1]
cluster3 = cluster[cluster.cluster==2]
plt.
 ↪scatter(cluster1['satisfaction_level'],cluster1['last_evaluation'],color='green')
plt.
 ↪scatter(cluster2['satisfaction_level'],cluster2['last_evaluation'],color='red')
plt.
 ↪scatter(cluster3['satisfaction_level'],cluster3['last_evaluation'],color='yellow')
plt.scatter(kmc.cluster_centers_[:,0],kmc.cluster_centers_[:
 ↪,1],color='purple',marker='*',label='centroid')
plt.xlabel('satisfaction_level')
plt.ylabel('last_evaluation')
plt.legend()
```

[26]:
```
<matplotlib.legend.Legend at 0x7f902a116f50>
```




[27]:
```
df.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
```

Pre-Process the data by converting categorical columns to numerical columns by Applying get_dummies() to the categorical variables.

[28]:
```
df.head()
```

[28]:
```
   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
```

[29]:
```
df.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
```

[30]:
```
df['department'].value_counts()
```

[30]:
```
sales          4140
technical      2720
support        2229
IT             1227
```

```
product_mng     902
marketing       858
RandD           787
accounting      767
hr              739
management      630
Name: department, dtype: int64
```

[31]:
```
department = pd.get_dummies(df['department'], prefix='department',␣
 ↪prefix_sep='_', drop_first=True)
department
```

[31]:

| | department_RandD | department_accounting | department_hr | \ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | |
| 1 | 0 | 0 | 0 | |
| 2 | 0 | 0 | 0 | |
| 3 | 0 | 0 | 0 | |
| 4 | 0 | 0 | 0 | |
| … | … | … | … | |
| 14994 | 0 | 0 | 0 | |
| 14995 | 0 | 0 | 0 | |
| 14996 | 0 | 0 | 0 | |
| 14997 | 0 | 0 | 0 | |
| 14998 | 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 | |
| … | … | … | … | |
| 14994 | 0 | 0 | 0 | |
| 14995 | 0 | 0 | 0 | |
| 14996 | 0 | 0 | 0 | |
| 14997 | 0 | 0 | 0 | |
| 14998 | 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 |
| … | … | … | … |
| 14994 | 0 | 1 | 0 |
| 14995 | 0 | 1 | 0 |

```
14996                0                    1                    0
14997                0                    1                    0
14998                0                    1                    0

[14999 rows x 9 columns]
```

[32]:
```
df['salary'].value_counts()
```

[32]:
```
low       7316
medium    6446
high      1237
Name: salary, dtype: int64
```

[33]:
```
salary =  pd.get_dummies(df['salary'], prefix='salary', prefix_sep='_',␣
 ↪drop_first=True)
salary
```

[33]:
```
       salary_low  salary_medium
0               1              0
1               0              1
2               0              1
3               1              0
4               1              0
…             …              …
14994           1              0
14995           1              0
14996           1              0
14997           1              0
14998           1              0

[14999 rows x 2 columns]
```

[34]:
```
data = pd.concat([df,department,salary], axis=1)
data.head()
```

[34]:
```
   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  …  department_accounting  department_hr  department_management  \
0     low  …                      0              0                      0
1  medium  …                      0              0                      0
2  medium  …                      0              0                      0
3     low  …                      0              0                      0
4     low  …                      0              0                      0

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

   department_support  department_technical  salary_low  salary_medium
0                   0                     0           1              0
1                   0                     0           0              1
2                   0                     0           0              1
3                   0                     0           1              0
4                   0                     0           1              0

[5 rows x 21 columns]
```

[35]:
```
data = data.drop(['department','salary'],axis=1)
data.shape
```

[35]:
```
(14999, 19)
```

[36]:
```
data.head()
```

[36]:
```
   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  \
0                   3              0     1                      0
1                   6              0     1                      0
2                   4              0     1                      0
3                   5              0     1                      0
4                   3              0     1                      0

   department_RandD  department_accounting  department_hr  \
```

```
0                  0                     0             0
1                  0                     0             0
2                  0                     0             0
3                  0                     0             0
4                  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  salary_low  \
0                 1                   0                     0           1
1                 1                   0                     0           0
2                 1                   0                     0           0
3                 1                   0                     0           1
4                 1                   0                     0           1

   salary_medium
0              0
1              1
2              1
3              0
4              0
```

[37]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 19 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_RandD       14999 non-null  uint8
 9   department_accounting  14999 non-null  uint8
 10  department_hr          14999 non-null  uint8
 11  department_management  14999 non-null  uint8
 12  department_marketing   14999 non-null  uint8
```

```
 13  department_product_mng  14999 non-null  uint8
 14  department_sales        14999 non-null  uint8
 15  department_support      14999 non-null  uint8
 16  department_technical    14999 non-null  uint8
 17  salary_low              14999 non-null  uint8
 18  salary_medium           14999 non-null  uint8
dtypes: float64(2), int64(6), uint8(11)
memory usage: 1.1 MB
```

Do the stratified split of the dataset to train and test.

[38]:
```
X = data.drop(['left'],axis=1)
y = data['left']
```

[39]:
```
print(X.shape)
print(y.shape)
```

```
(14999, 18)
(14999,)
```

[40]:
```
sss = StratifiedShuffleSplit(n_splits=5, test_size=0.2, random_state=123)
sss.get_n_splits(X, y)
```

[40]: 5

[41]:
```
for train,test in sss.split(X,y):     #this will splits the index
    X_train = X.iloc[train]
    y_train = y.iloc[train]
    X_test = X.iloc[test]
    y_test = y.iloc[test]
print(y_train.value_counts())
print(y_test.value_counts())
```

```
0    9142
1    2857
Name: left, dtype: int64
0    2286
1     714
Name: left, dtype: int64
```

[42]:
```
print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
(11999, 18)
(11999,)
(3000, 18)
(3000,)
```

Upsample the train dataset using SMOTE technique from the imblearn module. que.

[43]:
```
smote = SMOTE(random_state = 11)
X_train, y_train = smote.fit_resample(X_train, y_train)
```

[44]:
```
print(X_train.shape)
print(y_train.shape)
```

```
(18284, 18)
(18284,)
```

Train a Logistic Regression model and apply a 5-Fold CV and plot the classification report.

[45]:
```
lr = LogisticRegression(solver='liblinear',multi_class='ovr')
score_lr=cross_val_score(lr,X_train, y_train, cv=5)
print(score_lr)
print("Avg :",np.average(score_lr))
```

```
[0.74651354 0.77796008 0.81104731 0.81596937 0.80661926]
Avg : 0.7916219097214119
```

[46]:
```
pred_lr = cross_val_predict(lr, X_test, y_test, cv=5)
pred_lr
```

[46]:
```
array([0, 0, 0, …, 0, 1, 0])
```

[47]:
```
print(classification_report(y_test,pred_lr))
```

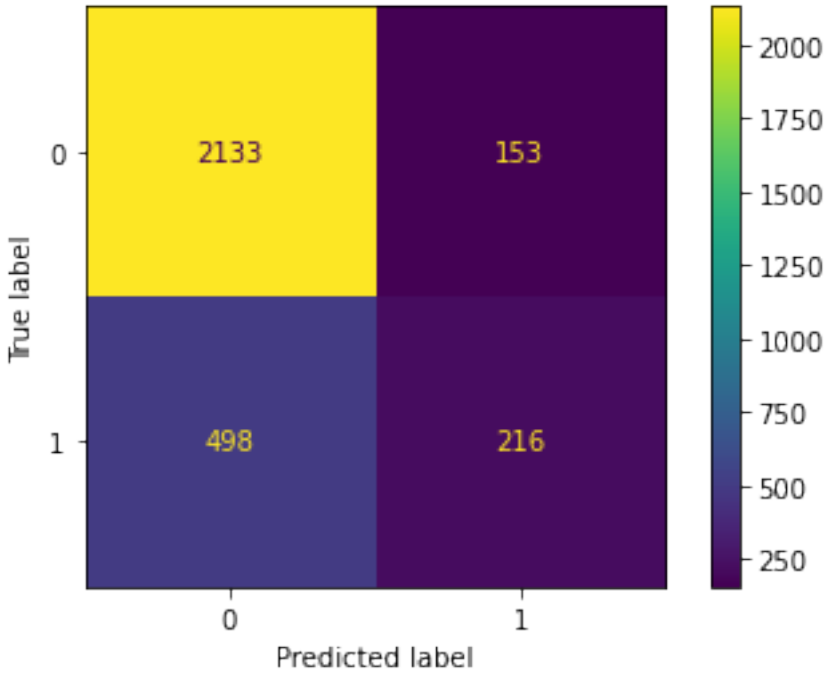
```
              precision    recall  f1-score   support

           0       0.81      0.93      0.87      2286
           1       0.59      0.30      0.40       714

    accuracy                           0.78      3000
   macro avg       0.70      0.62      0.63      3000
weighted avg       0.76      0.78      0.76      3000
```

[48]:
```
cm_lr = confusion_matrix(y_test, pred_lr)
disp = ConfusionMatrixDisplay(cm_lr)
disp.plot()
```

[48]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x7f902977b820>
```




Train a Random Forest Classifier model and apply the 5-Fold CV and plot the classification report.

[49]:
```
rfc = RandomForestClassifier(n_estimators=40)
score_rfc=cross_val_score(rfc,X_train, y_train, cv=5)
print(score_rfc)
print("Avg :",np.average(score_rfc))
```

```
[0.98222587 0.98222587 0.97757725 0.98249932 0.97893873]
Avg : 0.9806934065480368
```

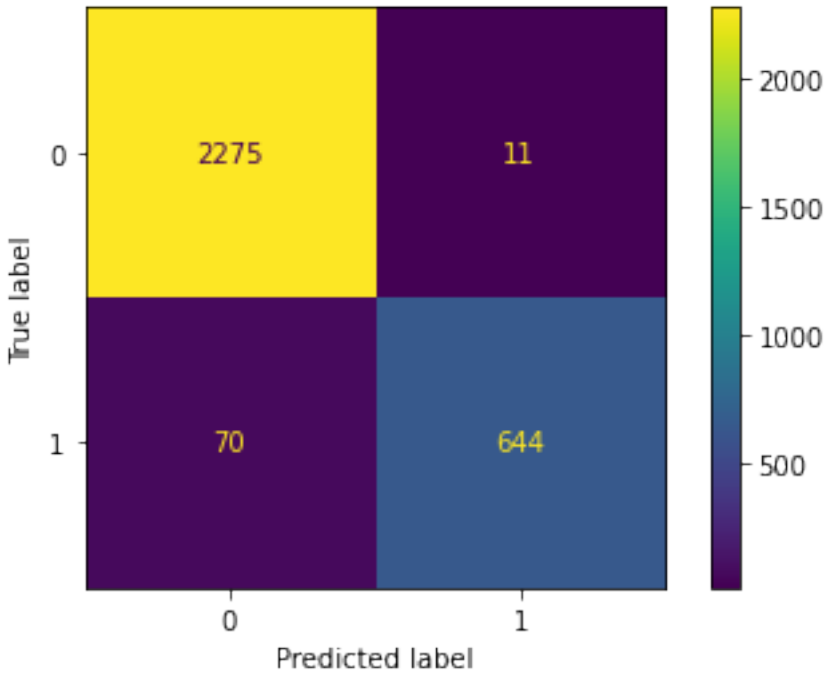
[50]:
```
pred_rfc = cross_val_predict(rfc, X_test, y_test, cv=5)
print(classification_report(y_test,pred_rfc))
cm_rfc = confusion_matrix(y_test, pred_rfc)
disp = ConfusionMatrixDisplay(cm_rfc)
disp.plot()
```

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98      2286
           1       0.98      0.90      0.94       714

    accuracy                           0.97      3000
   macro avg       0.98      0.95      0.96      3000
weighted avg       0.97      0.97      0.97      3000
```

[50]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f90297a1870>




Train a Gradient Boosting Classifier model and apply the 5-Fold CV and plot the classification report.

[51]:
```
gbc = GradientBoostingClassifier(n_estimators=100, learning_rate=1.
 ↪0,max_depth=1, random_state=0)
score_gbc=cross_val_score(gbc,X_train, y_train, cv=5)
print(score_gbc)
print("Avg :",np.average(score_gbc))
```

```
[0.95023243 0.94476347 0.9472245  0.94749795 0.95350109]
Avg : 0.9486438884929773
```

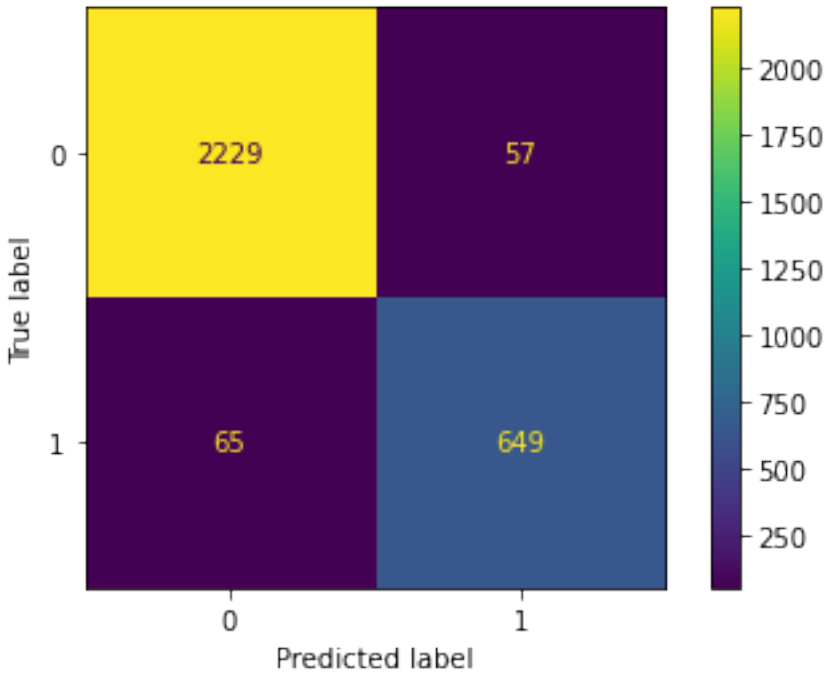
[52]:
```
pred_gbc = cross_val_predict(gbc, X_test, y_test, cv=5)
print(classification_report(y_test,pred_gbc))
cm_gbc = confusion_matrix(y_test, pred_gbc)
disp = ConfusionMatrixDisplay(cm_gbc)
disp.plot()
```

```
              precision    recall  f1-score   support

           0       0.97      0.98      0.97      2286
           1       0.92      0.91      0.91       714
```

```
              accuracy                           0.96      3000
             macro avg       0.95      0.94      0.94      3000
          weighted avg       0.96      0.96      0.96      3000
```

[52]:
```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at
0x7f902e4fc5b0>
```




Using the best model, predict the probability of employee turnover in the test data.

[54]:
```
test_predictions = pd.DataFrame(data=pred_rfc)
test_predictions
```

[54]:
```
        0
0       1
1       1
2       0
3       0
4       0
…      ..
2995    0
2996    1
2997    0
2998    0
```

```
2999  1

[3000 rows x 1 columns]
```

[55]:
```
test_predictions.rename(columns={0:'predictions'},inplace=True)
test_predictions.head()
```

[55]:
```
   predictions
0            1
1            1
2            0
3            0
4            0
```

[56]:
```
prob = cross_val_predict(rfc, X_test, y_test, cv=5, method='predict_proba')
# keep probabilities for the positive outcome only
prob = prob[:, 1]
prob
```

[56]:
```
array([1.   , 0.95 , 0.025, …, 0.05 , 0.05 , 0.95 ])
```

[57]:
```
probability = pd.DataFrame(data=prob)
probability.head()
```

[57]:
```
       0
0  1.000
1  0.950
2  0.025
3  0.000
4  0.000
```

[58]:
```
probability.rename(columns={0:'probability'},inplace=True)
probability.head()
```

[58]:
```
   probability
0        1.000
1        0.950
2        0.025
3        0.000
4        0.000
```

[59]:
```
len(probability)
```

[59]:
```
3000
```

Based on the below probability score range, categorize the employees into four zones and suggest your thoughts on the retention strategies for each zone.

Safe Zone (Green) (Score $< 20\%$)

Low Risk Zone (Yellow) (20% < Score < 60%)
Medium Risk Zone (Orange) (60% < Score < 90%)
High Risk Zone (Red) (Score > 90%).

[60]:
```
# create a list of our conditions
conditions = [
    (probability['probability'] <= 0.2),
    (probability['probability'] > 0.2) & (probability['probability'] <= 0.6),
    (probability['probability'] > 0.6) & (probability['probability'] <= 0.9),
    (probability['probability'] > 0.9)
    ]

# create a list of the values we want to assign for each condition
values = ['Safe Zone (Green)', 'Low Risk Zone (Yellow)', 'Medium Risk Zone␣
 ↪(Orange)', 'High Risk Zone (Red)']

# create a new column and use np.select to assign values to it using our lists␣
 ↪as arguments
probability['zone'] = np.select(conditions, values)

# display updated DataFrame
probability.head()
```

[60]:

| | probability | zone |
|---|---|---|
| 0 | 1.000 | High Risk Zone (Red) |
| 1 | 0.950 | High Risk Zone (Red) |
| 2 | 0.025 | Safe Zone (Green) |
| 3 | 0.000 | Safe Zone (Green) |
| 4 | 0.000 | Safe Zone (Green) |

[61]:
```
print(X_test.shape)
print(test_predictions.shape)
print(probability.shape)
```

```
(3000, 18)
(3000, 1)
(3000, 2)
```

[62]:
```
X_test = X_test.reset_index()
```

[63]:
```
new_test_df = pd.concat([X_test,test_predictions,probability], axis=1)
new_test_df.head()
```

[63]:

| | index | satisfaction_level | last_evaluation | number_project | \ |
|---|---|---|---|---|---|
| 0 | 439 | 0.41 | 0.52 | 2 | |
| 1 | 649 | 0.46 | 0.50 | 2 | |
| 2 | 8478 | 0.58 | 0.63 | 5 | |

```
3  13225                0.52              0.89                3
4   7962                0.74              0.54                4

   average_montly_hours  time_spend_company  Work_accident  \
0                   136                   3              0
1                   156                   3              0
2                   191                   3              1
3                   188                   6              0
4                   167                   2              0

   promotion_last_5years  department_RandD  department_accounting  …  \
0                      0                 0                      0  …
1                      0                 0                      0  …
2                      0                 0                      0  …
3                      0                 0                      0  …
4                      0                 0                      0  …

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

   department_support  department_technical  salary_low  salary_medium  \
0                   0                     1           1              0
1                   0                     1           0              1
2                   0                     1           0              0
3                   0                     0           0              1
4                   0                     0           0              1

   predictions  probability                 zone
0            1        1.000  High Risk Zone (Red)
1            1        0.950  High Risk Zone (Red)
2            0        0.025     Safe Zone (Green)
3            0        0.000     Safe Zone (Green)
4            0        0.000     Safe Zone (Green)

[5 rows x 22 columns]
```

[64]:
```
new_test_df['zone'].value_counts()
```

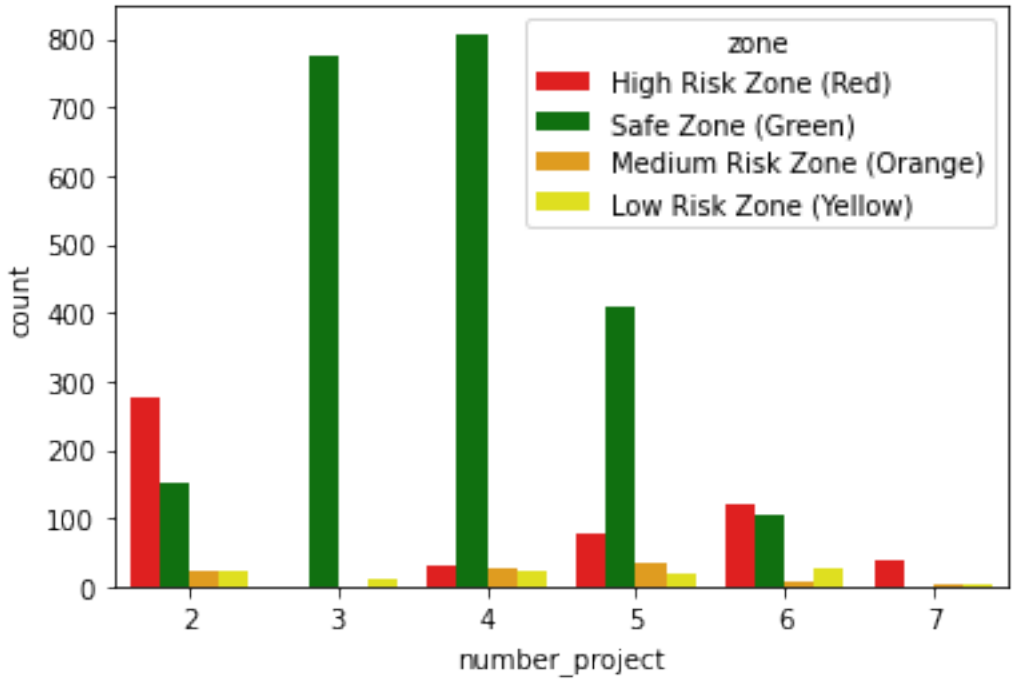
[64]:
```
Safe Zone (Green)            2249
High Risk Zone (Red)          543
Low Risk Zone (Yellow)        109
Medium Risk Zone (Orange)      99
Name: zone, dtype: int64
```

[65]:
```
new_test_df['zone'].value_counts()/len(new_test_df)
```

[65]:
```
Safe Zone (Green)            0.749667
High Risk Zone (Red)         0.181000
Low Risk Zone (Yellow)       0.036333
Medium Risk Zone (Orange)    0.033000
Name: zone, dtype: float64
```

[66]:
```
colors = {'High Risk Zone (Red)':'red', 'Safe Zone (Green)':'green', 'Medium␣
↪Risk Zone (Orange)':'orange', 'Low Risk Zone (Yellow)':'yellow'}
sns.countplot(data=new_test_df, x='number_project', hue='zone', palette= colors)
```

[66]:
```
<AxesSubplot: xlabel='number_project', ylabel='count'>
```




Overview based on 'number_project' :

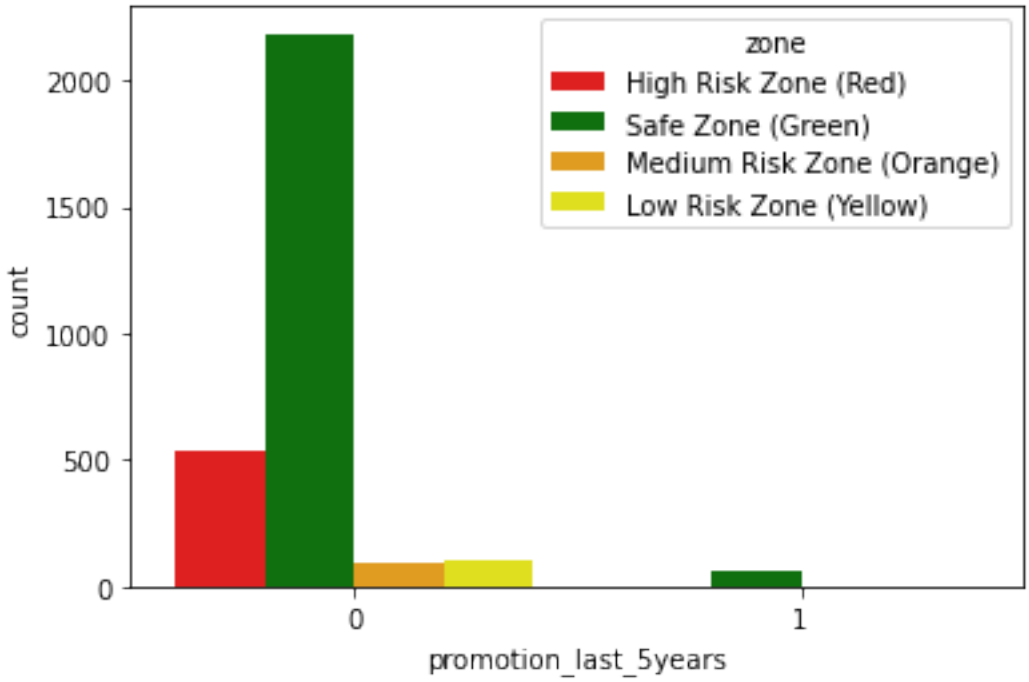
above plot, we can clearly say that employees who were involved in less 2 or less projects are most likely to leave the company. Also, employees who were part of 6 or more projects were also more prone to leave the company.

[67]:
```
new_test_df['promotion_last_5years'].value_counts()
```

[67]:
```
0    2929
1      71
Name: promotion_last_5years, dtype: int64
```

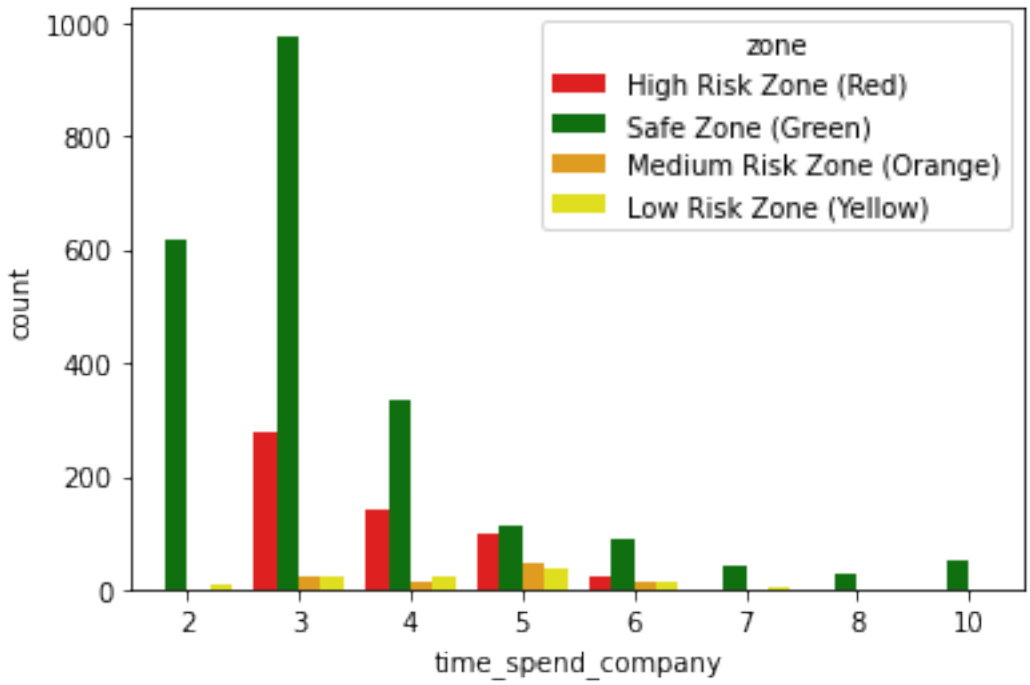
[68]:
```
sns.countplot(data=new_test_df, x='promotion_last_5years', hue='zone', palette=␣
↪colors)
```

[68]: <AxesSubplot: xlabel='promotion_last_5years', ylabel='count'>




[69]:
```
sns.countplot(data=new_test_df, x='time_spend_company', hue='zone', palette=␣
↪colors)
```

[69]: <AxesSubplot: xlabel='time_spend_company', ylabel='count'>

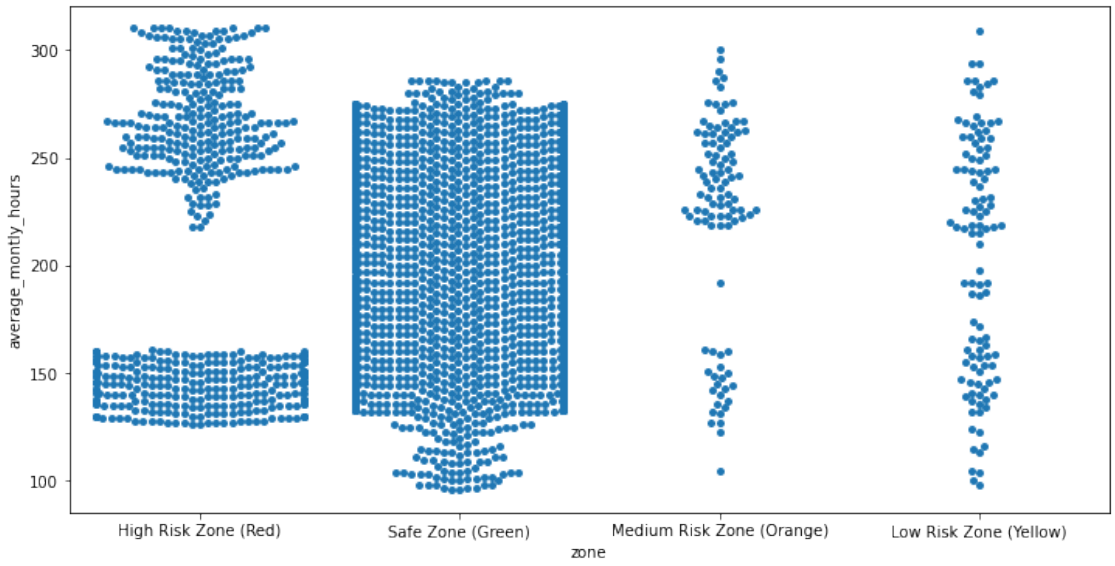



Overview based on 'time_spend_company' :

From above plot, we can clearly say that employees with experience between 3 to 5 yrs are likely to leave.

[70]:
```
# average_montly_hours
plt.figure(figsize=(12,6))
sns.swarmplot(data=new_test_df, x='zone', y='average_montly_hours')
```

[70]: <AxesSubplot: xlabel='zone', ylabel='average_montly_hours'>




[71]:
```
plt.figure(figsize=(12,6))
sns.violinplot(x="zone",y="average_montly_hours",data=new_test_df, inner="box",␣
↪palette=colors, cut=2, linewidth=1)
```

[71]: <AxesSubplot: xlabel='zone', ylabel='average_montly_hours'>

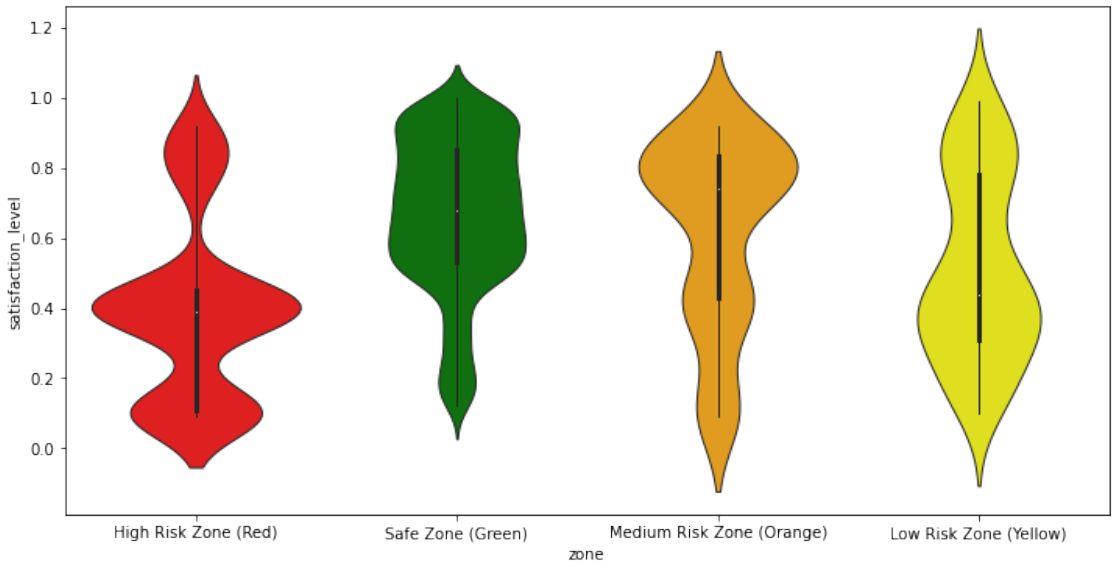

Overview based on 'average_montly_hours' :

From above plot, we can clearly say that employees who worked less than 160 Hrs were most likely to quit. Also, the employees who worked more than 220 Hrs on an average were at medium risk of

quiting the company. However, Employees who worked 200 Hrs mothly on an average were pretty happy and did not quit from the company.

[72]:
```
plt.figure(figsize=(12,6))
sns.violinplot(x="zone",y="satisfaction_level",data=new_test_df, inner="box",␣
↪palette=colors, cut=2, linewidth=1)
```

[72]: <AxesSubplot: xlabel='zone', ylabel='satisfaction_level'>

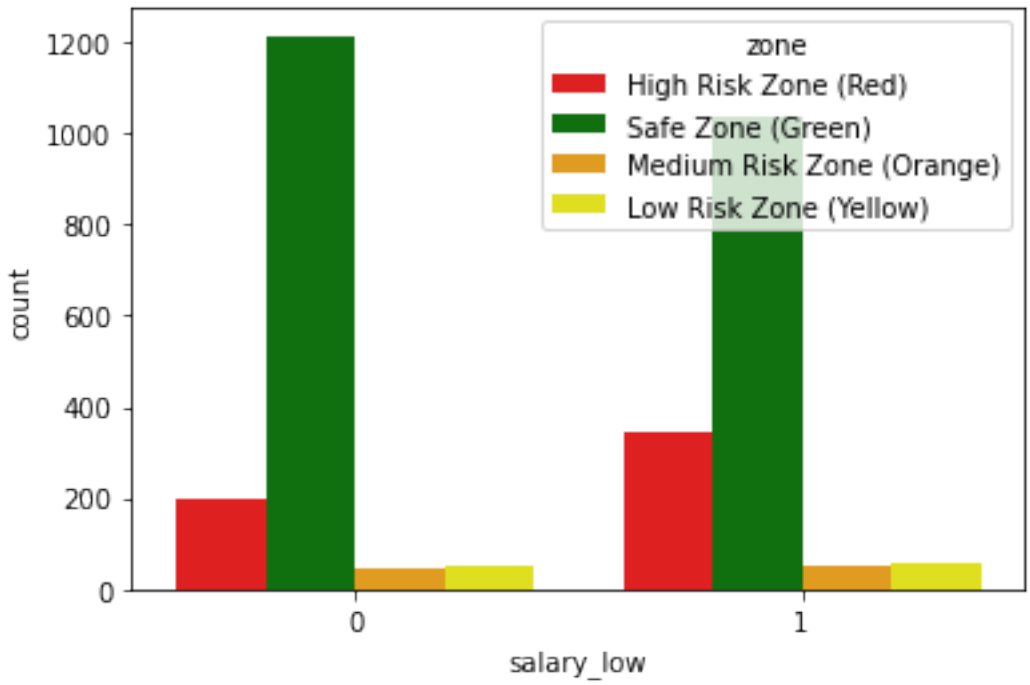



Overview based on 'satisfaction_level' :

From above plot, we can clearly say that employees with satisfaction level less than 0.4 were most likely to leave the company. Also, employees with satisfaction level of 0.6 were happy at the company and will not quit. But also, employees with satisfaction level of 0.8 and higher had some chance of quiting.

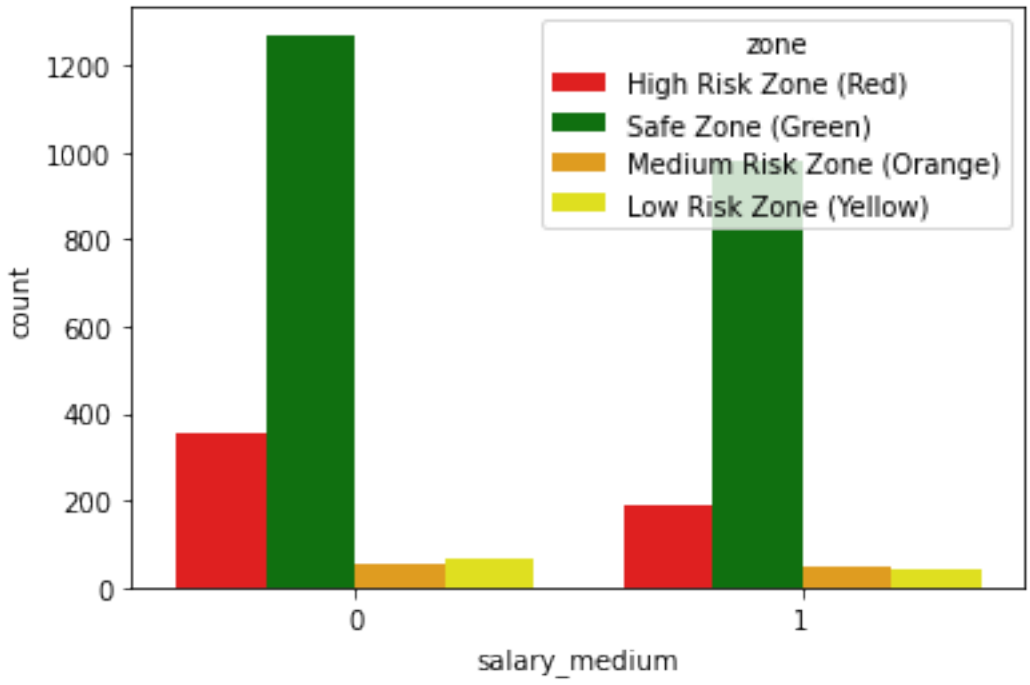
[73]:
```
sns.countplot(data=new_test_df, x='salary_low', hue='zone', palette= colors)
```

[73]: <AxesSubplot: xlabel='salary_low', ylabel='count'>



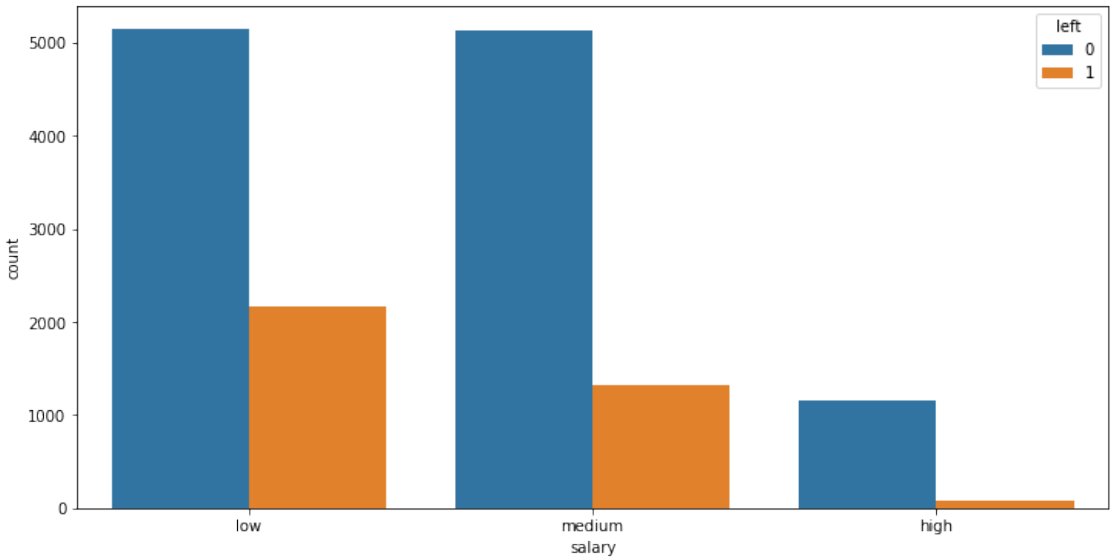

[74]:
```
sns.countplot(data=new_test_df, x='salary_medium', hue='zone', palette= colors)
```

[74]: <AxesSubplot: xlabel='salary_medium', ylabel='count'>



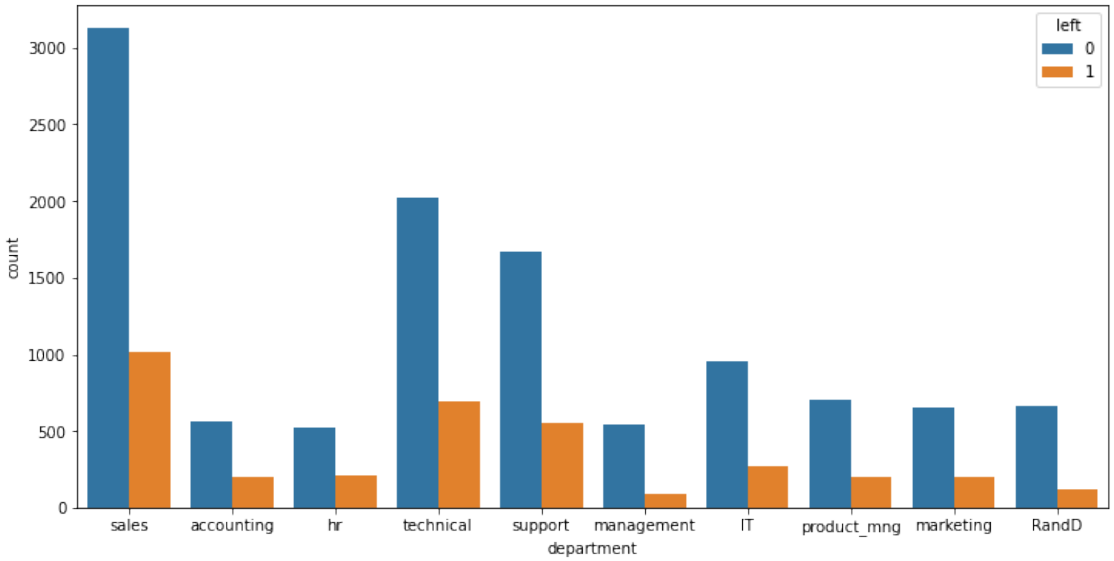
[75]:
```
plt.figure(figsize=(12,6))
sns.countplot(data=df, x='salary', hue='left')
```

[75]: <AxesSubplot: xlabel='salary', ylabel='count'>



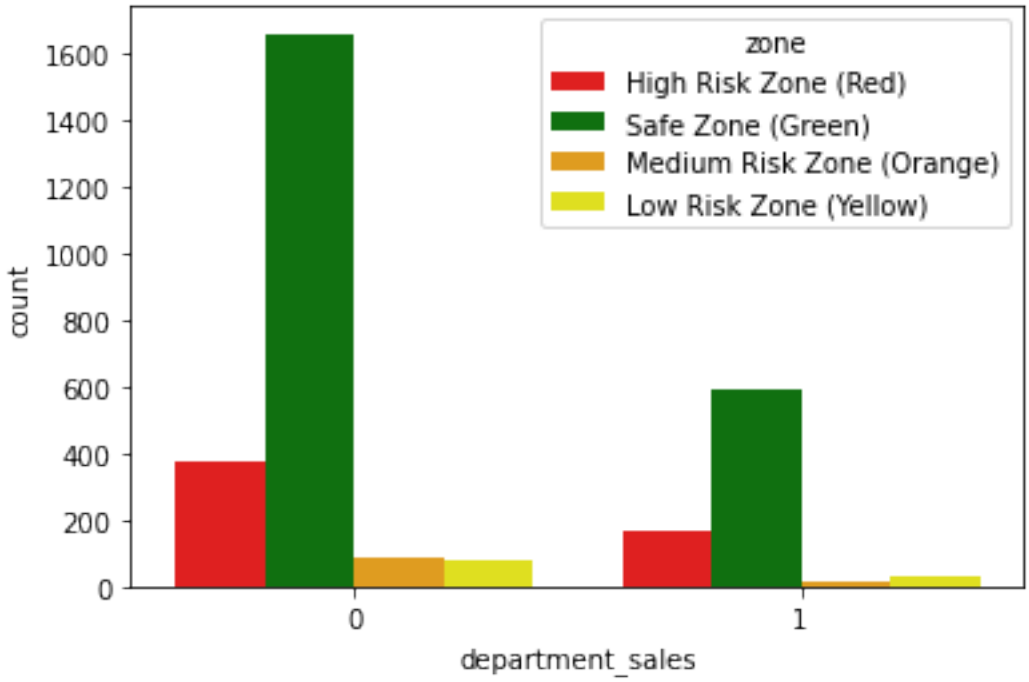

[76]:
```
plt.figure(figsize=(12,6))
sns.countplot(data=df, x='department', hue='left')
```

[76]: <AxesSubplot: xlabel='department', ylabel='count'>



[77]:
```
sns.countplot(data=new_test_df, x='department_sales', hue='zone', palette=␣
↪colors)
```

[77]: <AxesSubplot: xlabel='department_sales', ylabel='count'>




Overview based on 'salary' and 'department' :

Both, these categories don't provide clear pattern on employee action. Although, employees with low and medium salary had more chance of quitting the others. Similarly, employees who belongs to sales, technical and support departments are more prone to leaving the company.