In [171]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder,StandardScaler,MinMaxScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split,KFold,cross_val_score
from sklearn.metrics import accuracy_score
from sklearn.decomposition import PCA
```

In [172]:
```
data=pd.read_excel("../../Dataset/ML/Assignment/1673873196_hr_comma_sep.xlsx")
```

In [173]:
```
data.head()
```

Out[173]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
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

Checking null value

In [174]:
```
data.isna().sum()
```

Out[174]:
```
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
```

In [175]:
```
data.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   sales                  14999 non-null  object 
 9   salary                 14999 non-null  object 
dtypes: float64(2), int64(6), object(2)
memory usage: 1.1+ MB
```

EDA

In [176]:
```
sns.set()
```

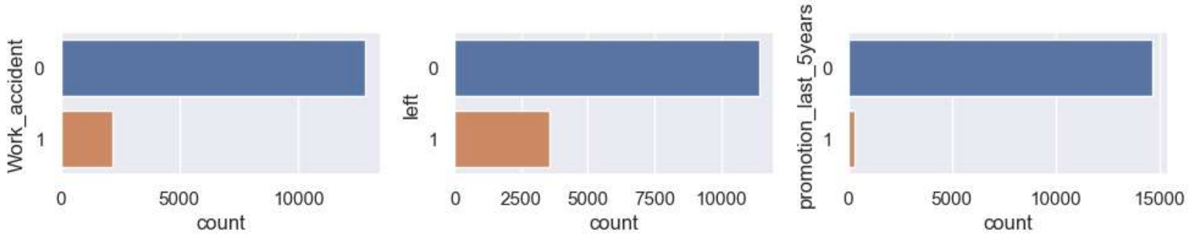
In [177]:
```
data.shape
```

Out[177]:
```
(14999, 10)
```

In [178]:
```
categorical=[x for x in data if data[x].dtypes=='O']
binary=[x for x in data if len(data[x].unique())<=2]
```

EDA of Binary Columns

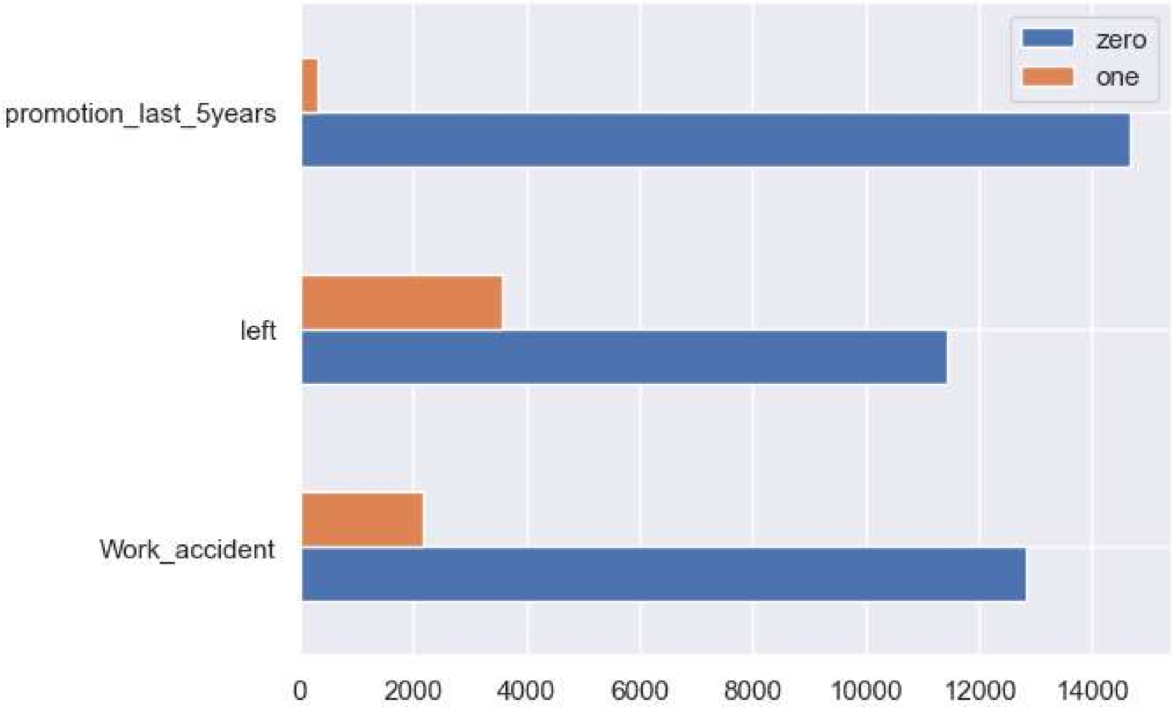
In [179]:
```
# Countplot for binary features
fig,ax=plt.subplots(1,3,figsize=(10,2))
for ax,col in zip(ax.flatten(),binary):
    sns.countplot(y=col,data=data,ax=ax)
    plt.tight_layout()
```



In [180]:
```
# Barplot for binary features
bin_bar=pd.DataFrame()
bin_bar['zero']=(data[binary]==0).sum()
bin_bar['one']=(data[binary]==1).sum()
```

In [181]:
```
bin_bar.plot.barh(y=['zero','one'])
plt.plot()
```

Out[181]:
```
[]
```



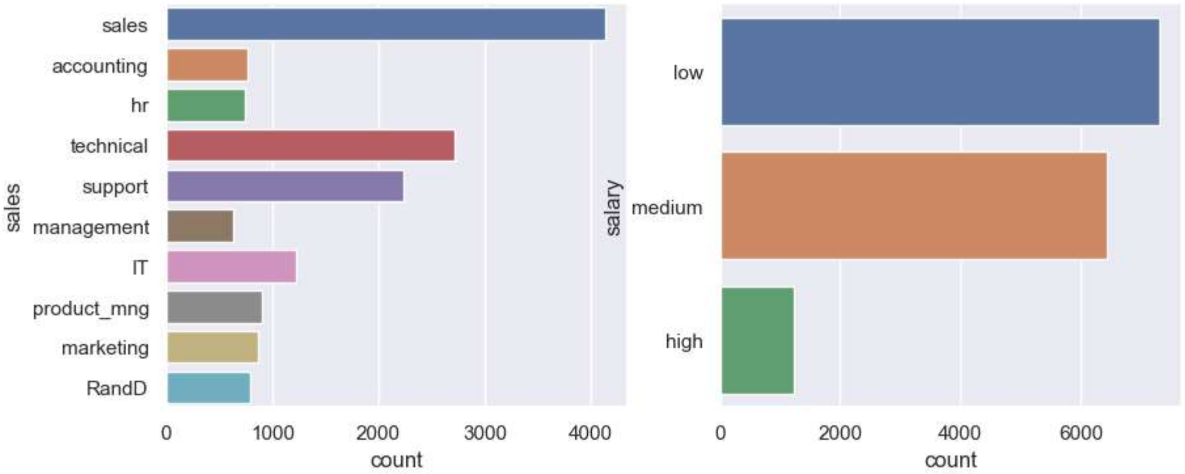
Findings

- We can see that the higher number of features have zero value and lower number of one value
- 30% of the features lies between 0 to 3000 values of 1
- There is less employee who is poromoted in the last 5 years
- Comparing left and stay of the employee, there is less employee who is left, we can say that 1 third of employee is left
- Less employee have occur work accident

EDA of Categorical features

In [182]:

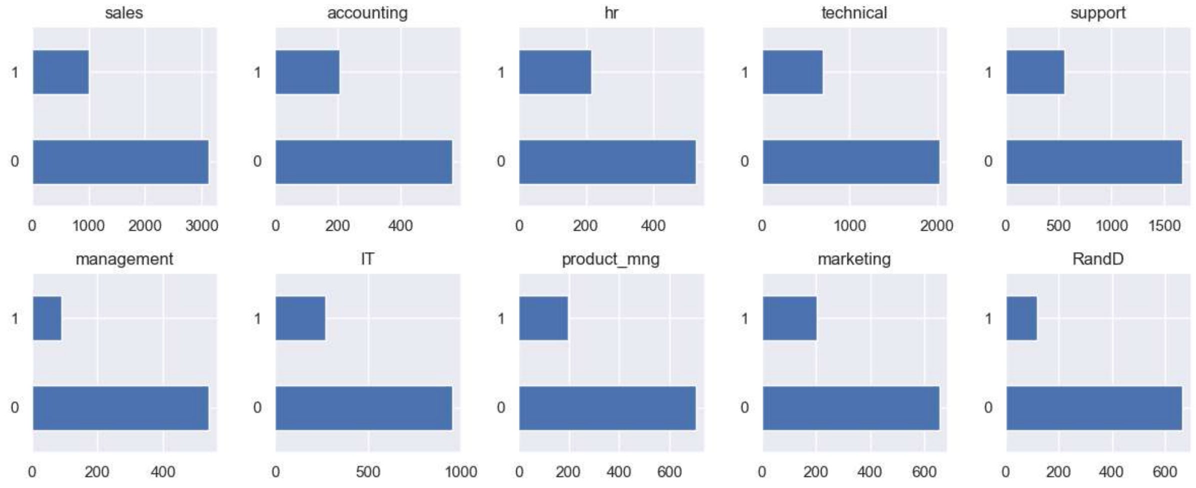
```
fig,ax=plt.subplots(1,2,figsize=(10,4))

for ax,cat in zip(ax.flatten(),categorical):
    sns.countplot(y=cat,data=data,ax=ax)
```



Checking which types of work has the most employee turn over

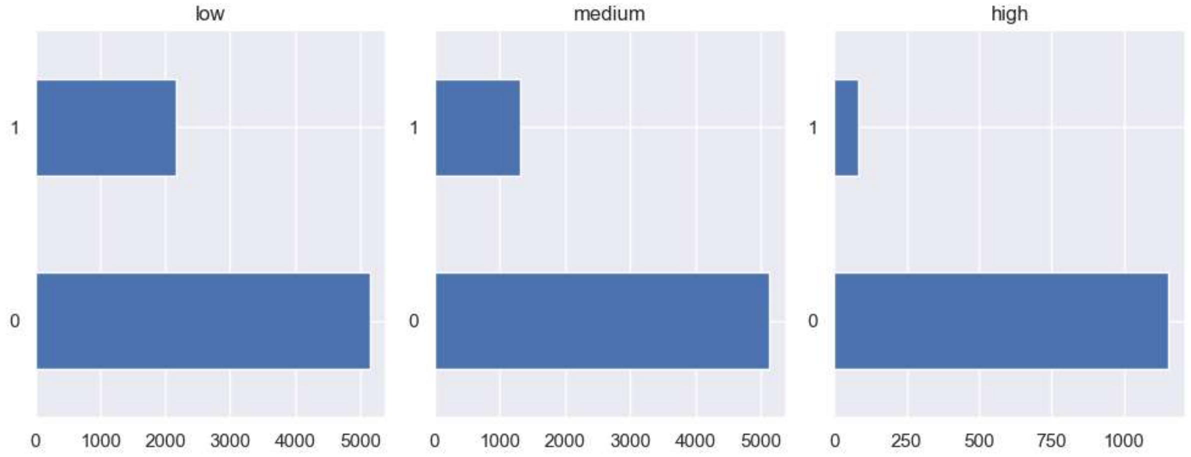
In [183]:

```
fig,ax=plt.subplots(2,5,figsize=(12,5))

work_type=data['sales'].unique()
for ax,work_ in zip(ax.flatten(),work_type):
    data.loc[data['sales']==work_]['left'].value_counts().plot.barh(ax=ax)
    ax.set_title(work_)
    plt.tight_layout()
```



In [184]:

```
fig,ax=plt.subplots(1,3,figsize=(10,4))

salary_type=data['salary'].unique()
for ax,work_ in zip(ax.flatten(),salary_type):
    data.loc[data['salary']==work_]['left'].value_counts().plot.barh(ax=ax)
    ax.set_title(work_)
    plt.tight_layout()
```



Findings

- We see that the hr job is have the most employee turnover followed by sales,marketing,technhical,accounting
- There is less employee turn over for the job of RanD,management,IT etc
- It is obvious that low salary employee have change to left the job

Feature Engineering

In [185]:

```
features=data.drop('left',axis=1)
y_=data['left']
```

In [186]:

```
features.head()
```

Out[186]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
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 [187]:

```
le=LabelEncoder()
```

In [188]:

```
features['sales_encoded']=le.fit_transform(data['sales'])
features['salary_encoded']=le.fit_transform(data['salary'])
```

In [189]:

```
def quadratic_encode(feature_,col1,col2):
    qe=feature_[col1]**2+5*feature_[col2]+8
    return qe
```

In [190]:

```
features['quadratic_encod1']=quadratic_encode(features,'Work_accident','promoti
```

In [191]:

```
features.head()
```

Out[191]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
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

Finding which features is responsible or employee turnover

In [192]:

```
features.head()
```

Out[192]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company
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 [193]:

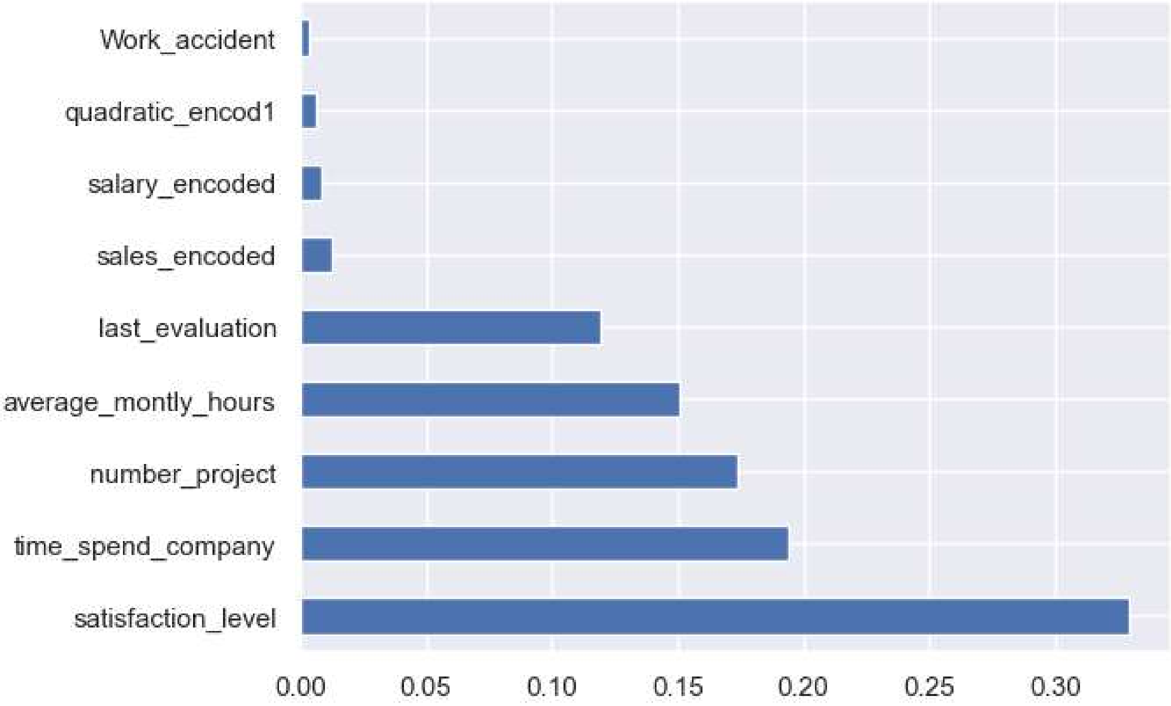
```
X=features.drop(['sales','salary'],axis=1)
```

In [194]:

```
rf=RandomForestClassifier().fit(X,y_)
rf_importances=pd.Series(rf.feature_importances_,index=X.columns)
rf_importances.nlargest(9).plot.barh()
```

Out[194]:

```
<AxesSubplot:>
```



Findings

- Satisfaction_level is huge responsible for employee turnover,followed by number_project,time_spend_company,average_montly_hours and last_evaluation

Modeling

In [195]:

```
selected=features.loc[data['left']==1][['satisfaction_level','last_evaluation']
```

In [196]:

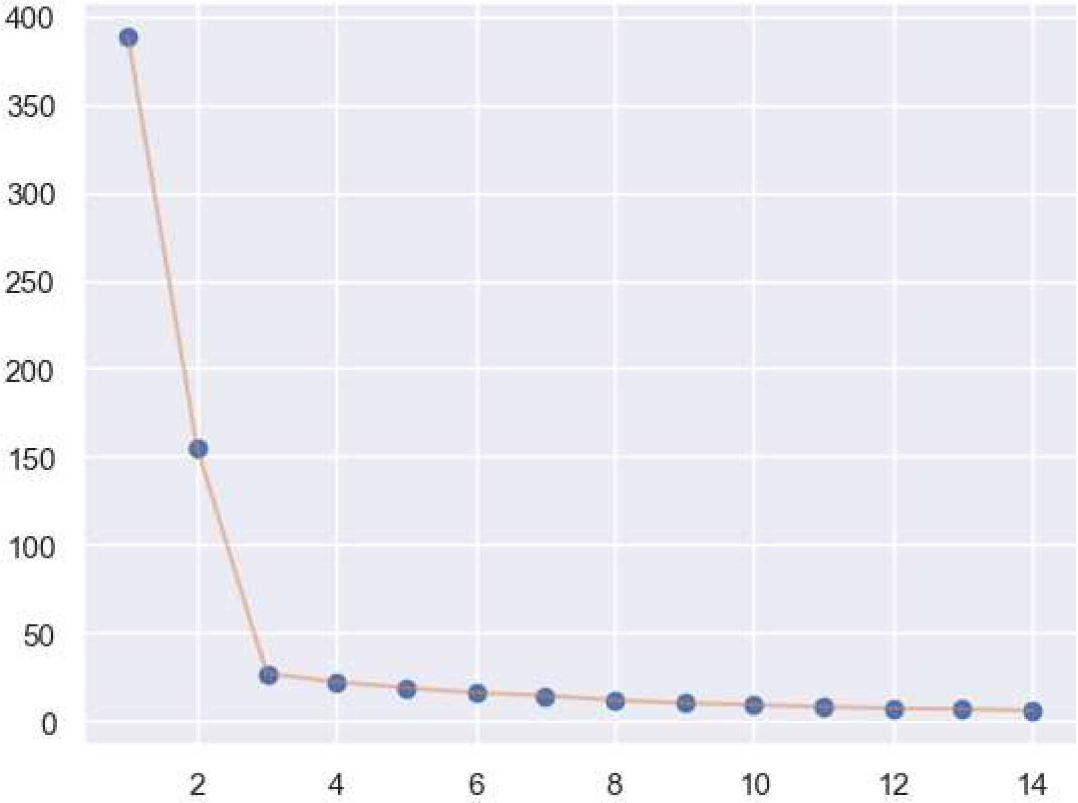
```
selected.head()
```

Out[196]:

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

In [197]:

```
# Find the number of cluster
wcss=[]

for i in range(1,15):
    kmeans=KMeans(n_clusters=i,init='k-means++',max_iter=300,n_init=10,random_s
    kmeans.fit(selected)
    wcss.append(kmeans.inertia_)

plt.plot(range(1,15),wcss,'o')
plt.plot(range(1,15),wcss,'-',alpha=0.5)
plt.show()
```



In [198]:

```
kmeans=KMeans(n_clusters=3,init='k-means++',random_state=10)
```

In [199]:

```
kmeans.fit(selected)
```

Out[199]:

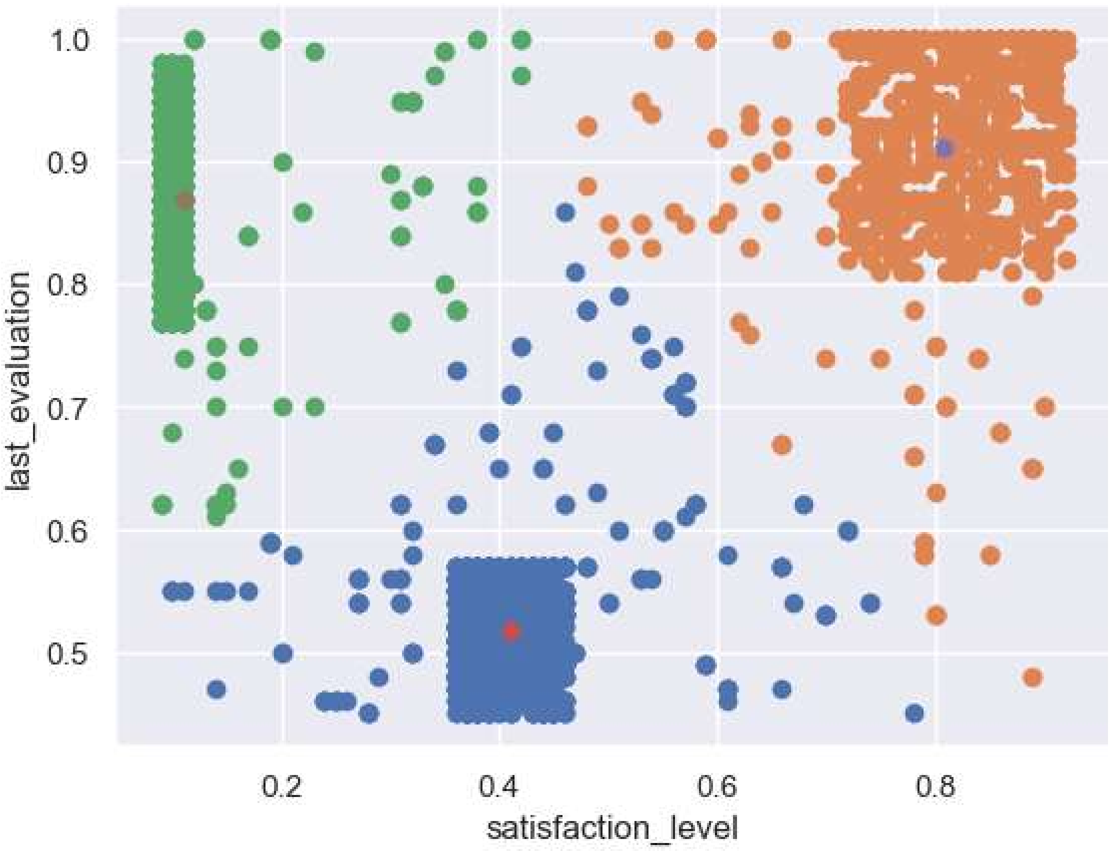
```
KMeans(n_clusters=3, random_state=10)
```

In [200]:

```
y_kmeans=kmeans.predict(selected)
```

In [201]:
```
plt.scatter(selected.values[y_kmeans==0,0],selected.values[y_kmeans==0,1],label
plt.scatter(selected.values[y_kmeans==1,0],selected.values[y_kmeans==1,1],label
plt.scatter(selected.values[y_kmeans==2,0],selected.values[y_kmeans==2,1],label

# Centroids
plt.scatter(kmeans.cluster_centers_[0][0],kmeans.cluster_centers_[0][1])
plt.scatter(kmeans.cluster_centers_[1][0],kmeans.cluster_centers_[1][1])
plt.scatter(kmeans.cluster_centers_[2][0],kmeans.cluster_centers_[2][1])
plt.xlabel('satisfaction_level')
plt.ylabel('last_evaluation')
```

Out[201]:
```
Text(0, 0.5, 'last_evaluation')
```



In [202]:
```
kmeans.cluster_centers_[0]
```

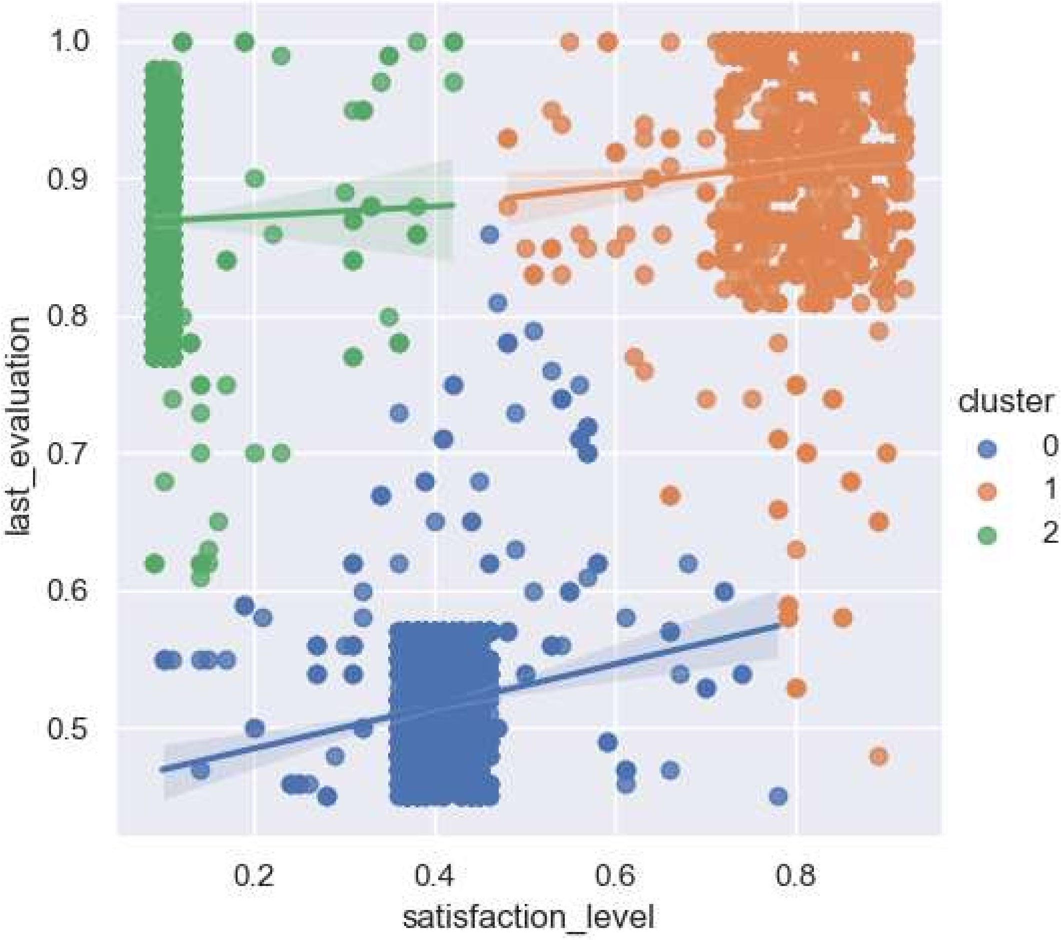
Out[202]:
```
array([0.41014545, 0.51698182])
```

In [203]:
```
selected['cluster']=y_kmeans
```

In [204]:
```
sns.lmplot('satisfaction_level','last_evaluation',data=selected,hue='cluster')
```

```
C:\Users\rajal\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureW
arning: Pass the following variables as keyword args: x, y. From version 0.1
2, the only valid positional argument will be `data`, and passing other argum
ents without an explicit keyword will result in an error or misinterpretatio
n.
  warnings.warn(
```

Out[204]:
```
<seaborn.axisgrid.FacetGrid at 0x21d0fc01100>
```



In [205]:
```
feature1=features.drop(['sales','salary'],axis=1)
target=data[['left']]
```

In [206]:
```
X_train,X_test,y_train,y_test=train_test_split(feature1,target,test_size=0.2,ra
```

In [207]:
```
Kf=KFold(n_splits=5,shuffle=False)
```

In [208]:
```
for train_set,test_set in Kf.split(X_train):
    print(f'TRAIN{train_set}')
    print(f'TEST{test_set}')
```

```
TRAIN[ 2400  2401  2402 ... 11996 11997 11998]
TEST[   0    1    2 ... 2397 2398 2399]
TRAIN[    0     1     2 ... 11996 11997 11998]
TEST[2400 2401 2402 ... 4797 4798 4799]
TRAIN[    0     1     2 ... 11996 11997 11998]
TEST[4800 4801 4802 ... 7197 7198 7199]
TRAIN[    0     1     2 ... 11996 11997 11998]
TEST[7200 7201 7202 ... 9597 9598 9599]
TRAIN[   0    1    2 ... 9597 9598 9599]
TEST[ 9600  9601  9602 ... 11996 11997 11998]
```

In [209]:
```
rf=RandomForestClassifier(class_weight={0:1,1:3},n_estimators=10)
```

In [210]:
```
score=cross_val_score(rf,X_train,y_train,scoring='accuracy',cv=5)
score*100
```

```
C:\Users\rajal\anaconda3\lib\site-packages\sklearn\model_selection\_validatio
n.py:680: DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples,), for example using
ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\rajal\anaconda3\lib\site-packages\sklearn\model_selection\_validatio
n.py:680: DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples,), for example using
ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\rajal\anaconda3\lib\site-packages\sklearn\model_selection\_validatio
n.py:680: DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples,), for example using
ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\rajal\anaconda3\lib\site-packages\sklearn\model_selection\_validatio
n.py:680: DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples,), for example using
ravel().
  estimator.fit(X_train, y_train, **fit_params)
C:\Users\rajal\anaconda3\lib\site-packages\sklearn\model_selection\_validatio
n.py:680: DataConversionWarning: A column-vector y was passed when a 1d array
was expected. Please change the shape of y to (n_samples,), for example using
ravel().
  estimator.fit(X_train, y_train, **fit_params)
```

Out[210]:
```
array([98.375     , 98.66666667, 99.25      , 98.75      , 98.74947895])
```

In [211]:
```
rf.fit(X_train,y_train)
```

```
C:\Users\rajal\AppData\Local\Temp\ipykernel_19092\1593328843.py:1: DataConver
sionWarning: A column-vector y was passed when a 1d array was expected. Pleas
e change the shape of y to (n_samples,), for example using ravel().
  rf.fit(X_train,y_train)
```

Out[211]:
```
RandomForestClassifier(class_weight={0: 1, 1: 3}, n_estimators=10)
```

In [212]:
```
pred=rf.predict(X_test)
```

In [213]:
```
accuracy_score(y_test,pred)*100
```

Out[213]:
```
98.56666666666666
```

In [214]:
```
y_test
```

Out[214]:
```
       left
13982     0
  822     1
13751     0
 9656     0
13497     0
  ...   ...
 3876     0
11504     0
 2435     0
 5161     0
 5184     0
```

3000 rows × 1 columns