In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```python
emp=pd.read_excel('employee.xlsx')
```

In [3]:
```python
emp.head()
```

Out[3]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	sales	salary
0	0.38	0.53	2	157	3	0	1	0	sales	low
1	0.80	0.86	5	262	6	0	1	0	sales	medium
2	0.11	0.88	7	272	4	0	1	0	sales	medium
3	0.72	0.87	5	223	5	0	1	0	sales	low
4	0.37	0.52	2	159	3	0	1	0	sales	low

In [4]:
```python
emp.isnull().any()
```

Out[4]:
```
satisfaction_level       False
last_evaluation          False
number_project           False
average_montly_hours     False
time_spend_company       False
Work_accident            False
left                     False
promotion_last_5years    False
sales                    False
salary                   False
dtype: bool
```

In [5]:
```python
import seaborn as sns
```

In [6]:
```python
emp.columns
```

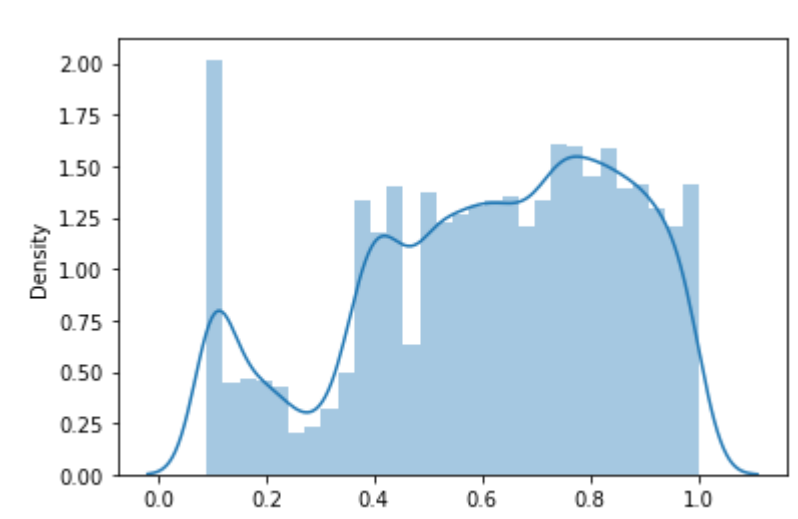
Out[6]:
```
Index(['satisfaction_level', 'last_evaluation', 'number_project',
       'average_montly_hours', 'time_spend_company', 'Work_accident', 'left',
       'promotion_last_5years', 'sales', 'salary'],
      dtype='object')
```

In [7]:
```python
#sns.heatmap(emp[['satisfaction_level', 'last_evaluation', 'number_project',
      # 'average_montly_hours', 'time_spend_company', 'Work_accident', 'left',
       #'promotion_last_5years']],annot=True)
```

In [8]:
```python
sns.distplot(emp.satisfaction_level)
```

```
C:\Users\sutharsan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

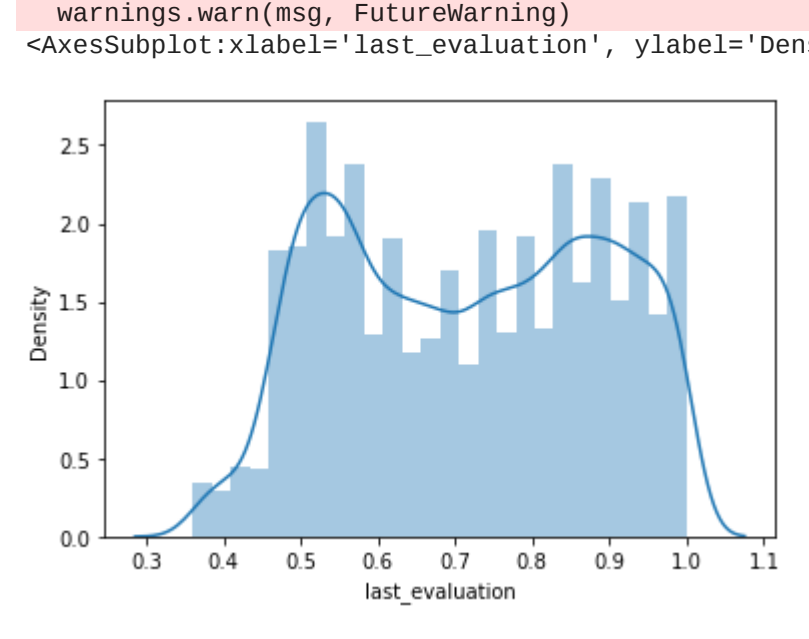
Out[8]:
```
<AxesSubplot:xlabel='satisfaction_level', ylabel='Density'>
```



In [9]:
```python
sns.distplot(emp.last_evaluation)
```

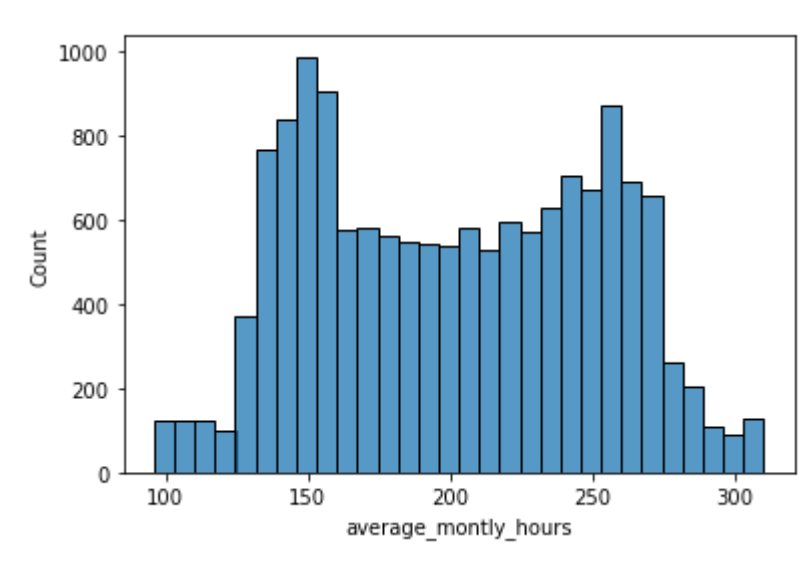
```
C:\Users\sutharsan\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[9]:
```
<AxesSubplot:xlabel='last_evaluation', ylabel='Density'>
```



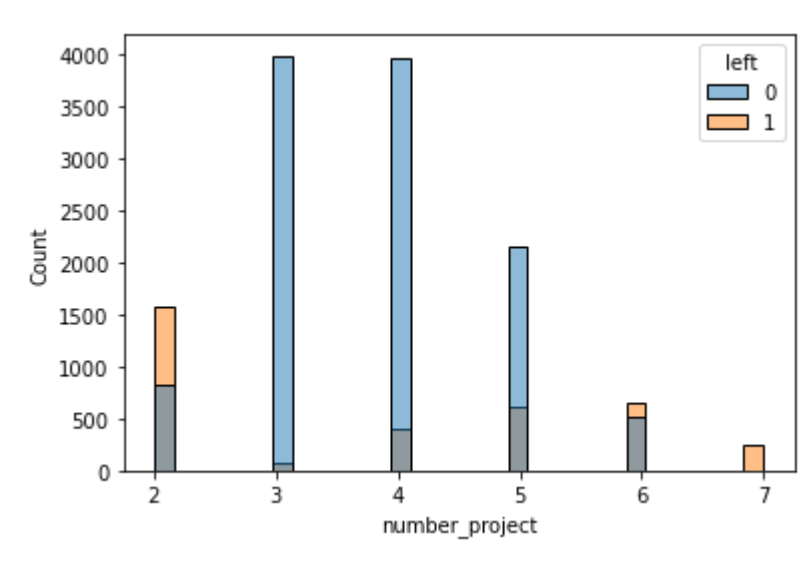
In [10]:
```python
sns.histplot(emp.average_montly_hours)
```

Out[10]:
```
<AxesSubplot:xlabel='average_montly_hours', ylabel='Count'>
```



In [11]:
```python
sns.histplot(data=emp,x='number_project',hue='left')
```

Out[11]:
```
<AxesSubplot:xlabel='number_project', ylabel='Count'>
```



In [12]:
```python
from sklearn.cluster import KMeans
```

In [14]:
```python
x=emp.loc[:,['satisfaction_level', 'last_evaluation','left']]
x.head()
```

Out[14]:

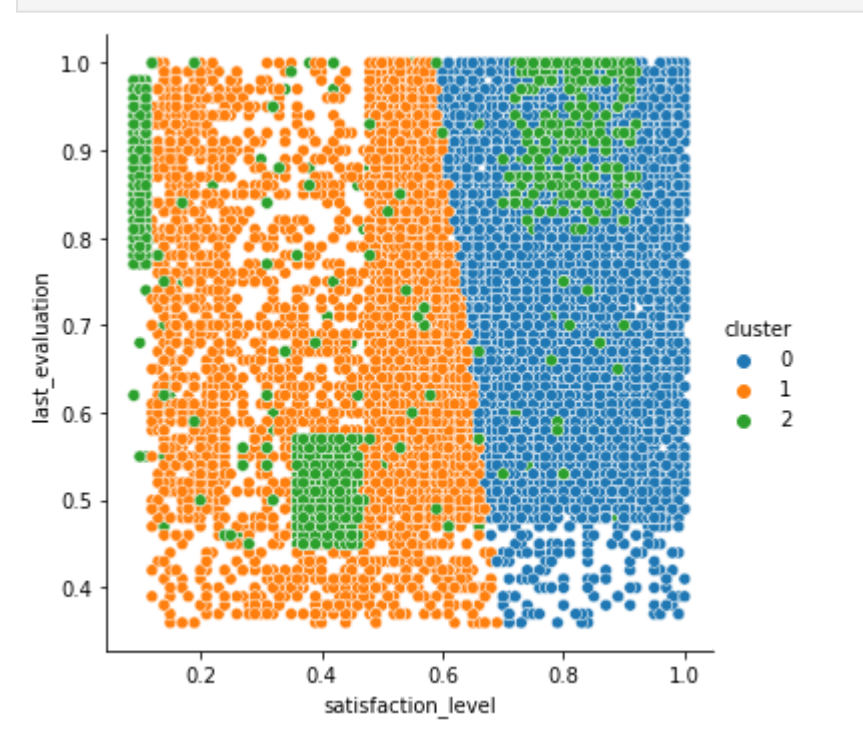
	satisfaction_level	last_evaluation	left
0	0.38	0.53	1
1	0.80	0.86	1
2	0.11	0.88	1
3	0.72	0.87	1
4	0.37	0.52	1

In [16]:
```python
km=KMeans(n_clusters=3)
x['cluster']=km.fit_predict(x)
x["cluster"] = x["cluster"].astype("category")
x.head()
```

Out[16]:

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

In [18]:
```python
sns.relplot(x='satisfaction_level',y='last_evaluation',hue='cluster',data=x);
```



In [21]:
```python
emp.dtypes
```

Out[21]:
```
satisfaction_level       float64
last_evaluation          float64
number_project             int64
average_montly_hours       int64
time_spend_company         int64
Work_accident              int64
left                       int64
promotion_last_5years      int64
sales                     object
salary                    object
dtype: object
```

In [23]:
```python
emp=pd.get_dummies(emp,columns=['sales', 'salary'])
```

In [24]:
```python
emp.head(2)
```

Out[24]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_company	Work_accident	left	promotion_last_5years	sales_IT	sales_RandD	...	sales_hr	sales_management
0	0.38	0.53	2	157	3	0	1	0	0	0	...	0	0
1	0.80	0.86	5	262	6	0	1	0	0	0	...	0	0

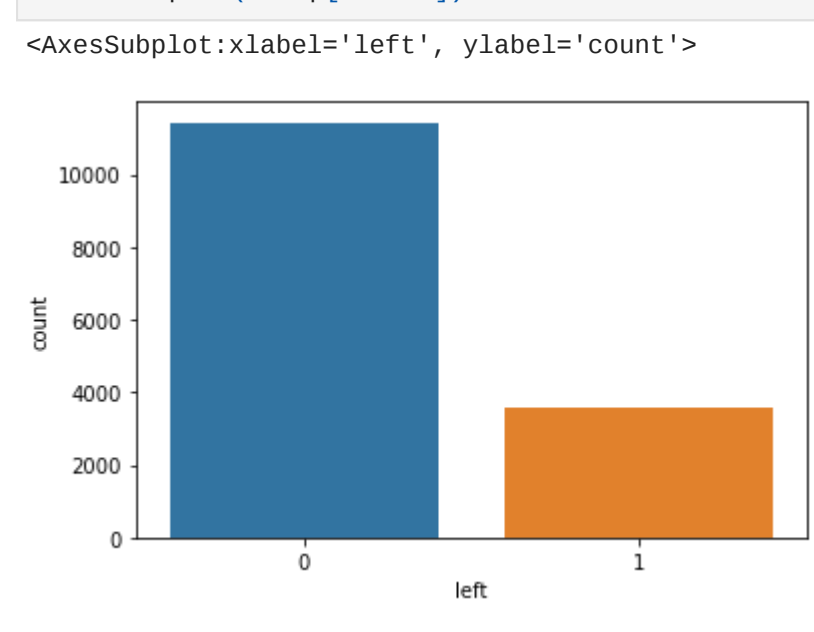
2 rows × 21 columns

In [26]:
```python
from sklearn.model_selection import train_test_split
y=emp['left']
x=emp.drop(['left'],axis=1)
```

In [27]:
```python
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=123)
```

In [33]:
```python
sns.countplot(x=emp['left'])        # data imbalance
```

Out[33]:
```
<AxesSubplot:xlabel='left', ylabel='count'>
```



In [35]:
```python
from imblearn.over_sampling import SMOTE
sm=SMOTE()
x_train_sm,y_train_sm=sm.fit_resample(x_train,y_train)
```

In [41]:
```python
y_train.value_counts()
```

Out[41]:
```
0    9137
1    2862
Name: left, dtype: int64
```

In [42]:
```python
y_train_sm.value_counts()
```

Out[42]:
```
0    9137
1    9137
Name: left, dtype: int64
```

In [43]:
```python
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report,accuracy_score
lr=LogisticRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
print('LLR: %2.2f' % accuracy_score(y_test,y_pred))
```

```
LLR: 0.79
```

```
C:\Users\sutharsan\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: 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(
```

In [56]:
```python
from sklearn import model_selection
kfold=model_selection.KFold(n_splits=5)
score='roc_auc'
lr_result=model_selection.cross_val_score(lr,x_train,y_train,cv=kfold,scoring=score)
```

```
C:\Users\sutharsan\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: 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(
C:\Users\sutharsan\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: 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(
C:\Users\sutharsan\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: 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(
C:\Users\sutharsan\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: 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(
C:\Users\sutharsan\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:814: 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(
```

In [57]:
```python
lr_result
```

Out[57]:
```
array([0.81935754, 0.81148639, 0.83558535, 0.81581868, 0.80930144])
```

In [60]:
```python
from sklearn.metrics import roc_auc_score
lr_score=roc_auc_score(y_test,y_pred)
print ("Logistic Regression AUC = %2.2f" % lr_score)
```

```
Logistic Regression AUC = 0.62
```

In [62]:
```python
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.81      0.93      0.87      2291
           1       0.59      0.31      0.41       709

    accuracy                           0.79      3000
   macro avg       0.70      0.62      0.64      3000
weighted avg       0.76      0.79      0.76      3000
```

In [65]:
```python
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
y_pred_rf=rf.predict(x_test)
print('rf: %2.2f' % accuracy_score(y_test,y_pred_rf))
```

```
rf: 0.99
```

In [67]:
```python
from sklearn import model_selection
kfold=model_selection.KFold(n_splits=5)
score='roc_auc'
rf_result=model_selection.cross_val_score(rf,x_train,y_train,cv=kfold,scoring=score)
```

In [68]:
```python
rf_result
```

Out[68]:
```
array([0.99072612, 0.99383242, 0.9930144 , 0.99000321, 0.98848025])
```

In [71]:
```python
from sklearn.metrics import roc_auc_score
rf_score=roc_auc_score(y_test,y_pred_rf)
print ("random forest AUC = %2.2f" % rf_score)
```

```
random forest AUC = 0.99
```

In [73]:
```python
print(classification_report(y_test, y_pred_rf))
```

```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      2291
           1       0.99      0.98      0.98       709

    accuracy                           0.99      3000
   macro avg       0.99      0.99      0.99      3000
weighted avg       0.99      0.99      0.99      3000
```

In [74]:
```python
from sklearn.ensemble import GradientBoostingClassifier
gb=GradientBoostingClassifier()
```

In [75]:
```python
gb.fit(x_train,y_train)
y_pred_gb=gb.predict(x_test)
print('GB: %2.2f' % accuracy_score(y_test,y_pred_gb))
```

```
GB: 0.98
```

In [76]:
```python
from sklearn import model_selection
kfold=model_selection.KFold(n_splits=5)
score='roc_auc'
gb_result=model_selection.cross_val_score(gb,x_train,y_train,cv=kfold,scoring=score)
```

In [77]:
```python
gb_result
```

Out[77]:
```
array([0.98567755, 0.9882766 , 0.98964867, 0.98890654, 0.98790006])
```

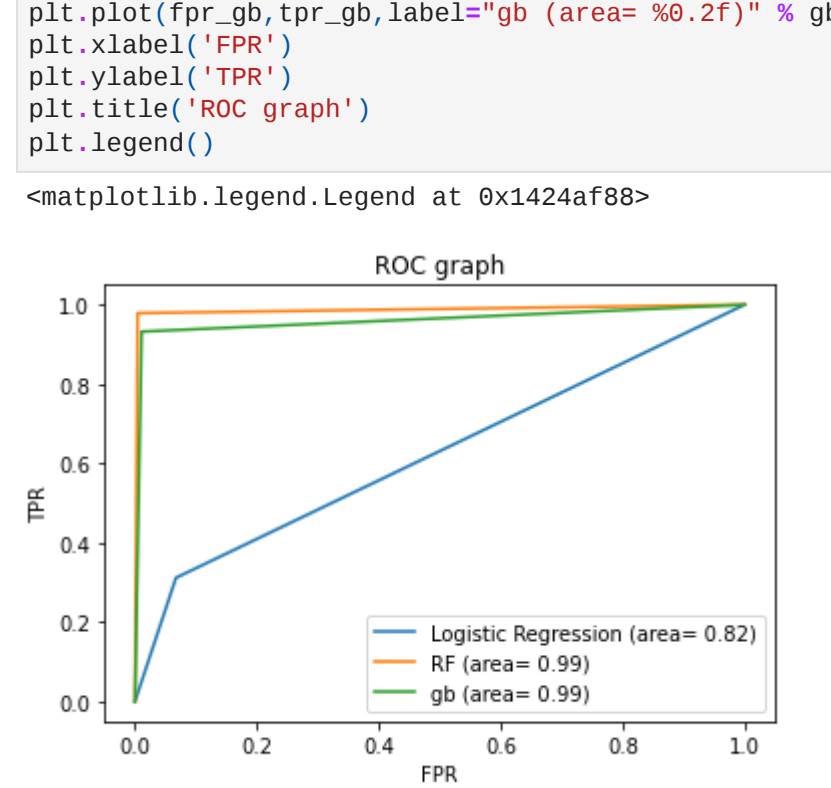
In [78]:
```python
from sklearn.metrics import roc_auc_score
gb_score=roc_auc_score(y_test,y_pred_gb)
print ("random forest AUC = %2.2f" % gb_score)
```

```
random forest AUC = 0.96
```

In [88]:
```python
from sklearn.metrics import roc_curve
fpr,tpr,thresholds=roc_curve(y_test,y_pred)
fpr_rf,tpr_rf,rf_thresholds=roc_curve(y_test,y_pred_rf)
fpr_gb,tpr_gb,gb_thresholds=roc_curve(y_test,y_pred_gb)
plt.figure()

plt.plot(fpr,tpr,label='Logistic Regression (area=% 0.2f)' % lr_result.mean())
plt.plot(fpr_rf,tpr_rf,label="RF (area=% 0.2f)" % rf_result.mean())
plt.plot(fpr_gb,tpr_gb,label="gb (area= %0.2f)" % gb_result.mean())
plt.xlabel('FPR')
plt.ylabel('TPR')
plt.title('ROC graph')
plt.legend()
```

Out[88]:
```
<matplotlib.legend.Legend at 0x1424af88>
```



In [92]:
```python
# rf got good accuracy_score
fm=rf.feature_importances_
```

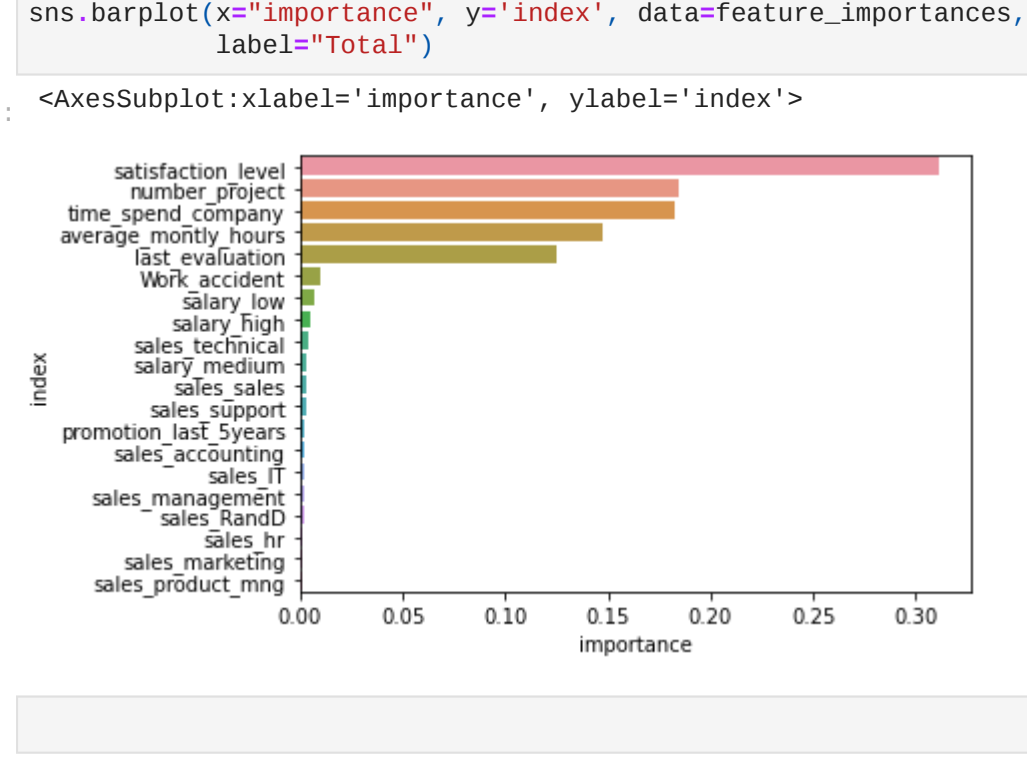
In [96]:
```python
feature_importances = pd.DataFrame(rf.feature_importances_,
                                   index = x_train.columns,
                                   columns=['importance']).sort_values('importance', ascending=False)
feature_importances = feature_importances.reset_index()
feature_importances
```

Out[96]:

	index	importance
0	satisfaction_level	0.310950
1	number_project	0.184559
2	time_spend_company	0.182799
3	average_montly_hours	0.147687
4	last_evaluation	0.125214
5	Work_accident	0.010133
6	salary_low	0.006879
7	salary_high	0.005224
8	sales_technical	0.003689
9	salary_medium	0.003482
10	sales_sales	0.003230
11	sales_support	0.002907
12	promotion_last_5years	0.001911
13	sales_accounting	0.001873
14	sales_IT	0.001869
15	sales_management	0.001827
16	sales_RandD	0.001717
17	sales_hr	0.001509
18	sales_marketing	0.001300
19	sales_product_mng	0.001241

In [100…
```python
sns.barplot(x="importance", y='index', data=feature_importances,
            label="Total")
```

Out[100]:
```
<AxesSubplot:xlabel='importance', ylabel='index'>
```



In [ ]: