In [3]:
```
##Human Resource Analytics, the objective of this project is to analyze 'why a
re the best and most experienced employees
##leaving an organization prematurely'. It is a problem from Kaggel data scien
ce challenge

#Import basic packages
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
#Output plots in notebook
%matplotlib inline

#Read data
data = pd.read_csv('HR_comma_sep.csv')
data.head()
```

Out[3]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
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]:
```
#check data types
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14999 entries, 0 to 14998
Data columns (total 10 columns):
satisfaction_level       14999 non-null float64
last_evaluation          14999 non-null float64
number_project           14999 non-null int64
average_montly_hours     14999 non-null int64
time_spend_company       14999 non-null int64
Work_accident            14999 non-null int64
left                     14999 non-null int64
promotion_last_5years    14999 non-null int64
sales                    14999 non-null object
salary                   14999 non-null object
dtypes: float64(2), int64(6), object(2)
memory usage: 1.1+ MB
```

In [7]:
```
#basic statistics of the attributes
data.describe()
```

Out[7]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_
count	14999.000000	14999.000000	14999.000000	14999.000000	14999
mean	0.612834	0.716102	3.803054	201.050337	3.498
std	0.248631	0.171169	1.232592	49.943099	1.460
min	0.090000	0.360000	2.000000	96.000000	2.000
25%	0.440000	0.560000	3.000000	156.000000	3.000
50%	0.640000	0.720000	4.000000	200.000000	3.000
75%	0.820000	0.870000	5.000000	245.000000	4.000
max	1.000000	1.000000	7.000000	310.000000	10.00

In [8]:
```
data.dtypes
```

Out[8]:
```
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 [9]:
```
#finding unique data
data['sales'].unique()
```

Out[9]:
```
array(['sales', 'accounting', 'hr', 'technical', 'support', 'management',
       'IT', 'product_mng', 'marketing', 'RandD'], dtype=object)
```

In [10]:
```
data['salary'].unique()
```

Out[10]:
```
array(['low', 'medium', 'high'], dtype=object)
```

In [11]:
```
data['sales'].replace(['sales', 'accounting', 'hr', 'technical', 'support', 'm
anagement', 'IT','product_mng', 'marketing', 'RandD'],[0,1,2,3,4,5,6,7,8,9], i
nplace=True)
```

In [41]:
```
data['salary'].replace([], [], inplace = True)
```

In [45]:
```
data['sales']
```

```
Out[45]: 0        0
         1        0
         2        0
         3        0
         4        0
         5        0
         6        0
         7        0
         8        0
         9        0
         10       0
         11       0
         12       0
         13       0
         14       0
         15       0
         16       0
         17       0
         18       0
         19       0
         20       0
         21       0
         22       0
         23       0
         24       0
         25       0
         26       0
         27       0
         28       1
         29       1
                 ..
         14969    0
         14970    0
         14971    0
         14972    1
         14973    1
         14974    1
         14975    2
         14976    2
         14977    2
         14978    2
         14979    3
         14980    3
         14981    3
         14982    3
         14983    3
         14984    3
         14985    3
         14986    3
         14987    3
         14988    3
         14989    3
         14990    4
         14991    4
         14992    4
         14993    4
         14994    4
```

```
14995    4
14996    4
14997    4
14998    4
Name: sales, Length: 14999, dtype: int64
```

In [46]:
```
data['salary'].replace(['low', 'medium', 'high'], [0,1,2], inplace = True)
```

In [52]:
```
#find correlations
corr= data.corr()

corr
```

Out[52]:

	satisfaction_level	last_evaluation	number_project	average_mor
satisfaction_level	1.000000	0.105021	-0.142970	-0.020048
last_evaluation	0.105021	1.000000	0.349333	0.339742
number_project	-0.142970	0.349333	1.000000	0.417211
average_montly_hours	-0.020048	0.339742	0.417211	1.000000
time_spend_company	-0.100866	0.131591	0.196786	0.127755
Work_accident	0.058697	-0.007104	-0.004741	-0.010143
left	-0.388375	0.006567	0.023787	0.071287
promotion_last_5years	0.025605	-0.008684	-0.006064	-0.003544
sales	0.015413	0.011855	0.005577	-0.002387
salary	0.050022	-0.013002	-0.001803	-0.002242

In [53]:
```
corr=(corr)
```

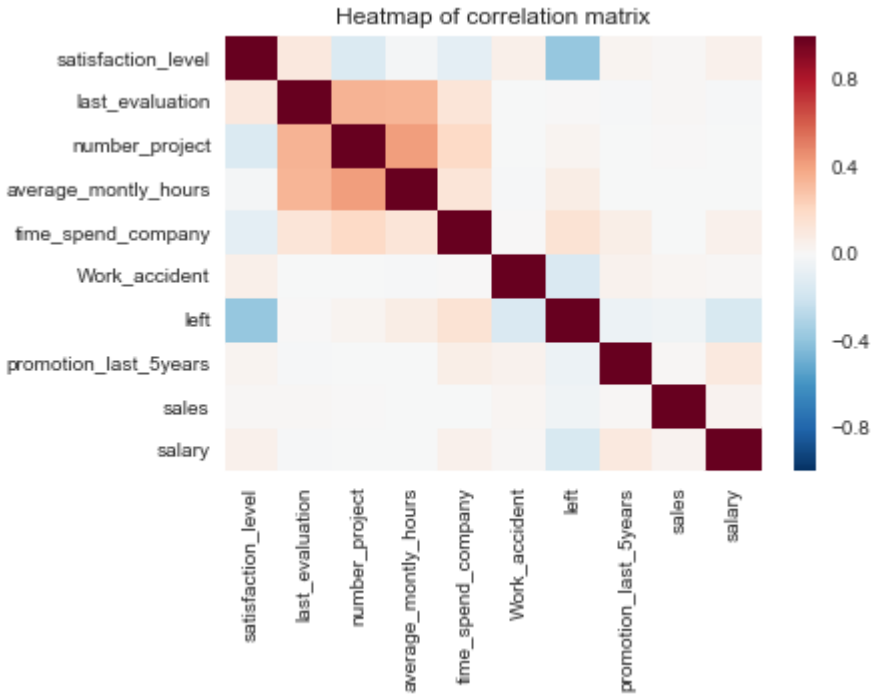
In [54]:
```
corr
```

Out[54]:

	satisfaction_level	last_evaluation	number_project	average_mor
satisfaction_level	1.000000	0.105021	-0.142970	-0.020048
last_evaluation	0.105021	1.000000	0.349333	0.339742
number_project	-0.142970	0.349333	1.000000	0.417211
average_montly_hours	-0.020048	0.339742	0.417211	1.000000
time_spend_company	-0.100866	0.131591	0.196786	0.127755
Work_accident	0.058697	-0.007104	-0.004741	-0.010143
left	-0.388375	0.006567	0.023787	0.071287
promotion_last_5years	0.025605	-0.008684	-0.006064	-0.003544
sales	0.015413	0.011855	0.005577	-0.002387
salary	0.050022	-0.013002	-0.001803	-0.002242

In [55]:
```
sns.heatmap(corr, xticklabels=corr.columns.values, yticklabels=corr.columns.va
lues)
sns.plt.title('Heatmap of correlation matrix')
```

Out[55]: <matplotlib.text.Text at 0xba6d780>



In [67]:
```
#extract column with name 'left'
corr_left = pd.DataFrame(corr['left'].drop('left'))
```

In [68]:
```
corr_left.sort_values(by ='left', ascending = False)
```

Out[68]:

	left
time_spend_company	0.144822
average_montly_hours	0.071287
number_project	0.023787
last_evaluation	0.006567
sales	-0.043814
promotion_last_5years	-0.061788
Work_accident	-0.154622
salary	-0.157898
satisfaction_level	-0.388375

In [12]:
```
data.head()
```

Out[12]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
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 [72]:
```
data['avg_hour_project'] = (data['average_montly_hours'] * 12) /data['number_p
roject']
data['avg_hour_project_range'] = pd.cut(data['avg_hour_project'], 3)
data[['avg_hour_project_range' , 'left']].groupby(['avg_hour_project_range']).
mean()
```

Out[72]:

	left
avg_hour_project_range	
(192.334, 749.333]	0.186359
(749.333, 1304.667]	0.340725
(1304.667, 1860.0]	0.098940

In [83]:
```
data['avg_hour_project'] = (data['average_montly_hours'] * 12) /data['number_p
roject']
data['avg_hour_project_range'] = pd.cut(data['avg_hour_project'], 3)
data[['avg_hour_project_range' , 'left']].groupby(['avg_hour_project_range']).
mean()
```

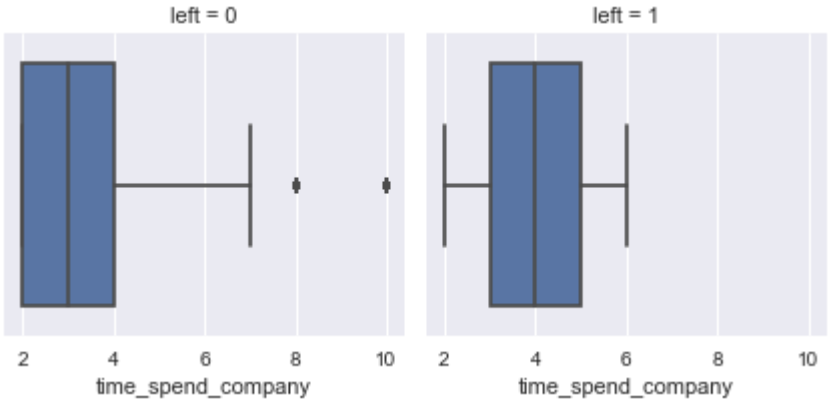
Out[83]:

	left
avg_hour_project_range	
(192.334, 749.333]	0.186359
(749.333, 1304.667]	0.340725
(1304.667, 1860.0]	0.098940

In [84]:
```
data.loc[data['avg_hour_project'] <= 749.333, 'avg_hour_project'] = 0
data.loc[(data['avg_hour_project'] <= 1304.667) & (data['avg_hour_project'] >
749.333), 'avg_hour_project'] = 1
data.loc[(data['avg_hour_project'] <= 1860.00) & (data['avg_hour_project'] > 1
304.667), 'avg_hour_project'] = 2
data.drop(['avg_hour_project_range'], axis = 1, inplace = True)
```

In [87]:
```
#some EDA on data
g = sns.FacetGrid(data, col='left')
g.map(sns.boxplot, 'time_spend_company')
```

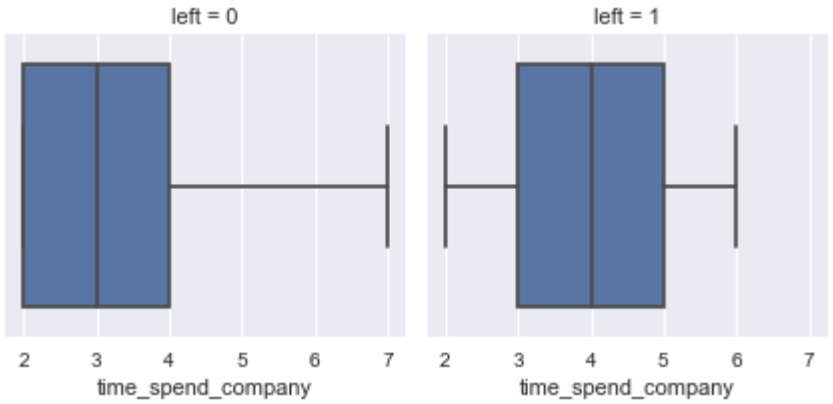
Out[87]: <seaborn.axisgrid.FacetGrid at 0xbec75c0>



In [90]:
```
dropdata = data[data['time_spend_company']>=8]
data.drop(dropdata.index, inplace = True)
```

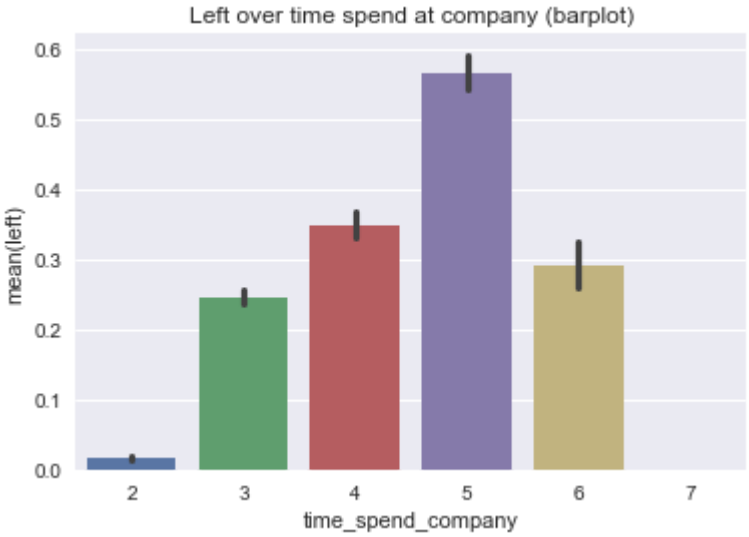
In [91]:
```
g = sns.FacetGrid(data, col='left')
g.map(sns.boxplot, 'time_spend_company')
```

Out[91]: <seaborn.axisgrid.FacetGrid at 0xc1e9978>



In [92]:
```
sns.barplot(x = 'time_spend_company', y= 'left', data = data)
sns.plt.title('Left over time spend at company (barplot)')
```

Out[92]: <matplotlib.text.Text at 0xd476ef0>

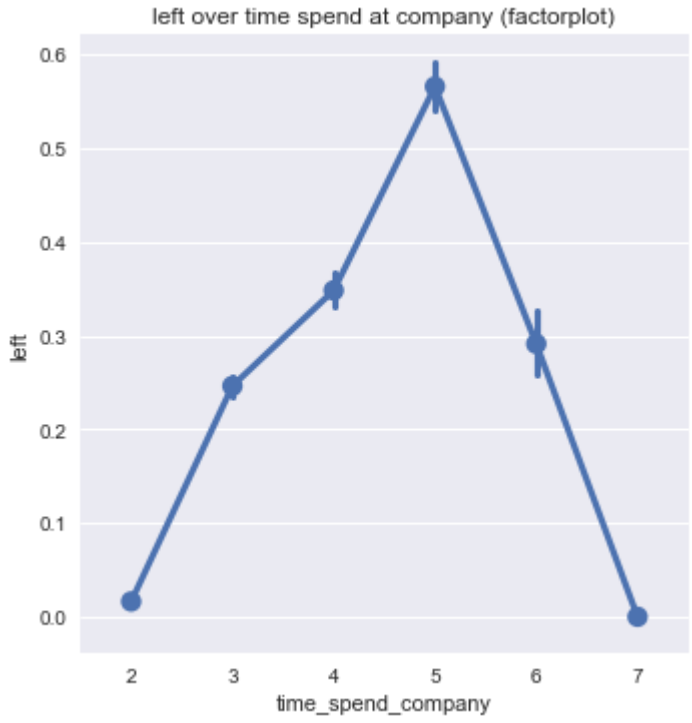


In [97]:
```
sns.factorplot(x = 'time_spend_company', y = 'left', data = data)
sns.plt.title('left over time spend at company (factorplot)')
```

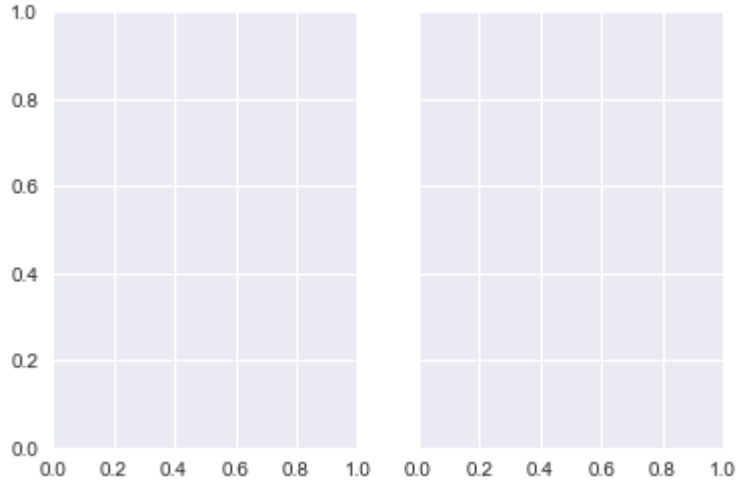
Out[97]: <matplotlib.text.Text at 0xdad1dd8>

In [98]:
```
sns.factorplot(x = 'time_spend_company', y = 'left', data = data, size  = 5)
sns.plt.title('left over time spend at company (factorplot)')
```

Out[98]: <matplotlib.text.Text at 0xdb8a4e0>

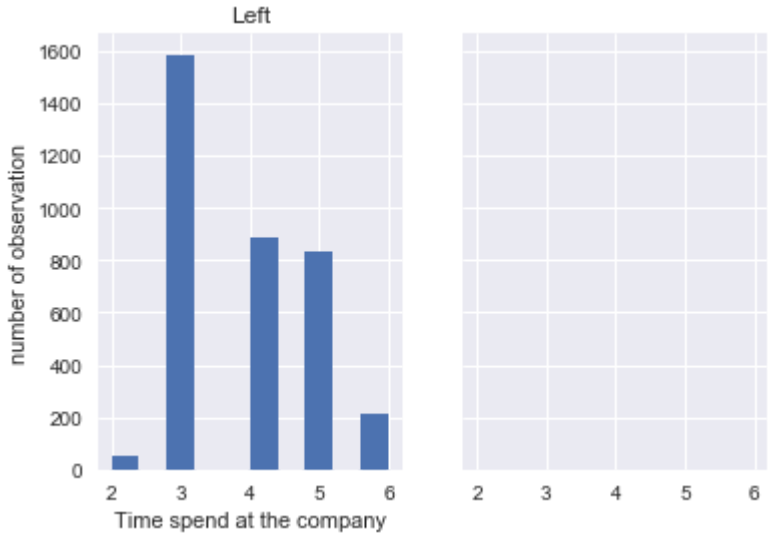


In [99]:
```
left = data[data['left'] == 1]
not_left = data[data['left'] == 0]
f, axrrr = plt.subplots(1, 2, sharey = True, sharex = True)
```



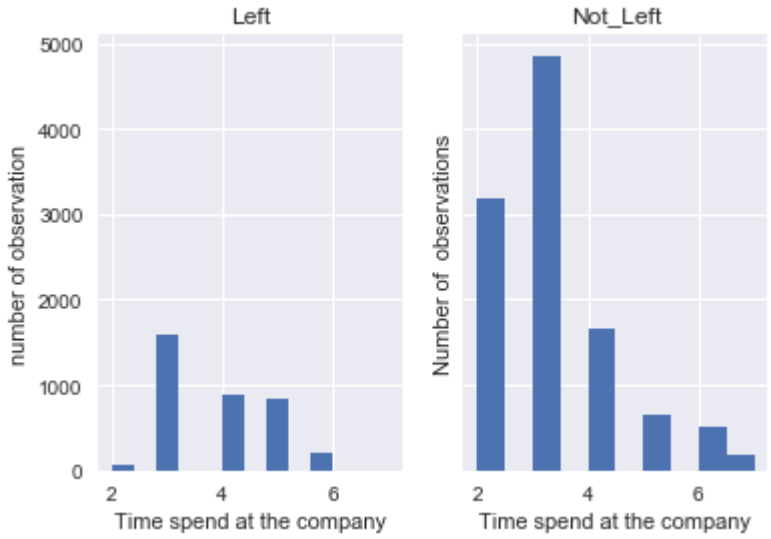
In [103]:
```
left = data[data['left'] == 1]
not_left = data[data['left'] == 0]
f, axrrr = plt.subplots(1, 2, sharey = True, sharex = True)
axrrr[0].hist('time_spend_company', data = left, bins = 10)
axrrr[0].set_title('Left')
axrrr[0].set_xlabel('Time spend at the company')
axrrr[0].set_ylabel('number of observation')
```

Out[103]: <matplotlib.text.Text at 0xde915f8>



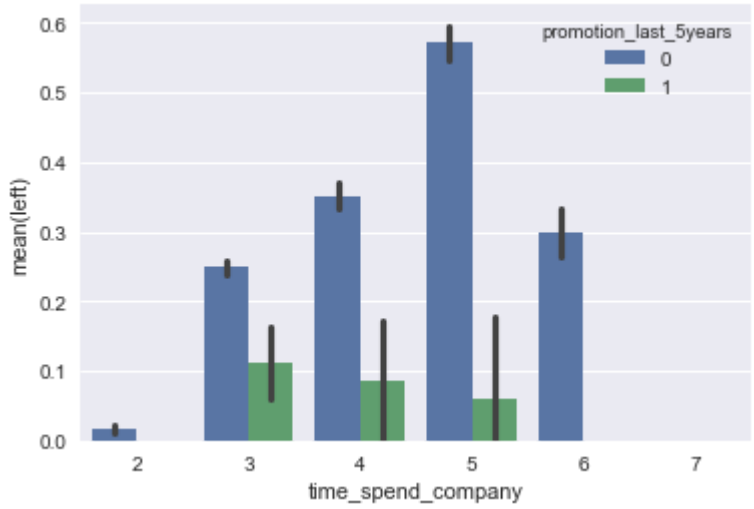
In [112]:
```
left = data[data['left'] == 1]
not_left = data[data['left'] == 0]
f, axrrr = plt.subplots(1, 2, sharey = True, sharex = True)

axrrr[0].hist('time_spend_company', data = left, bins = 10)
axrrr[0].set_title('Left')
axrrr[0].set_xlabel('Time spend at the company')
axrrr[0].set_ylabel('number of observation')

axrrr[1].hist('time_spend_company', data = not_left, bins = 10)
axrrr[1].set_title('Not_Left')
axrrr[1].set_xlabel('Time spend at the company')
axrrr[1].set_ylabel('Number of  observations')
```

Out[112]: <matplotlib.text.Text at 0xf9a2128>



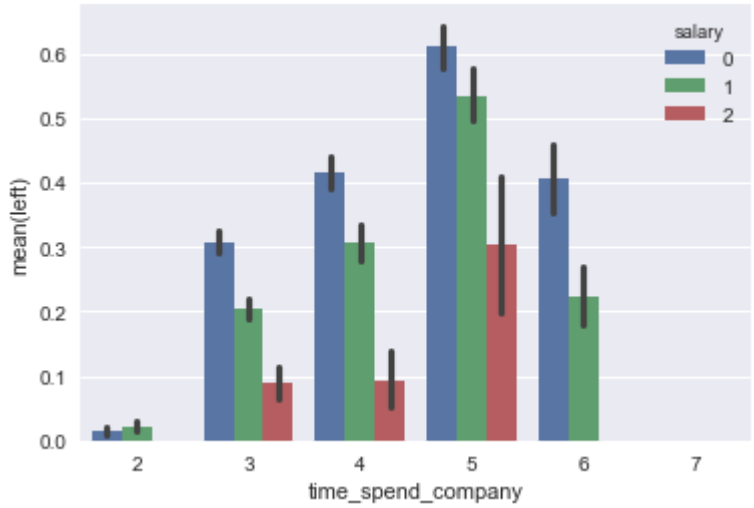
In [116]:
```
sns.barplot(x ='time_spend_company', y = 'left', hue = 'promotion_last_5years'
,  data = data)
```

Out[116]: <matplotlib.axes._subplots.AxesSubplot at 0xfba9588>



In [117]:
```
sns.barplot(x ='time_spend_company', y = 'left', hue = 'salary',  data = data)
```

Out[117]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xfd40588>
```



In [118]:
```
data.dtypes
```

Out[118]:
```
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                      int64
salary                     int64
avg_hour_project         float64
dtype: object
```

In [119]:
```
data.describe()
```

Out[119]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_
count	14623.000000	14623.000000	14623.000000	14623.000000	14623
mean	0.611633	0.715922	3.805102	201.157355	3.353
std	0.249324	0.171453	1.238614	49.984234	1.150
min	0.090000	0.360000	2.000000	96.000000	2.000
25%	0.440000	0.560000	3.000000	156.000000	3.000
50%	0.640000	0.720000	4.000000	200.000000	3.000
75%	0.820000	0.870000	5.000000	245.000000	4.000
max	1.000000	1.000000	7.000000	310.000000	7.000

In [120]:
```
data.head()
```

Out[120]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
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 [121]:
```
data['salary'].unique()
```

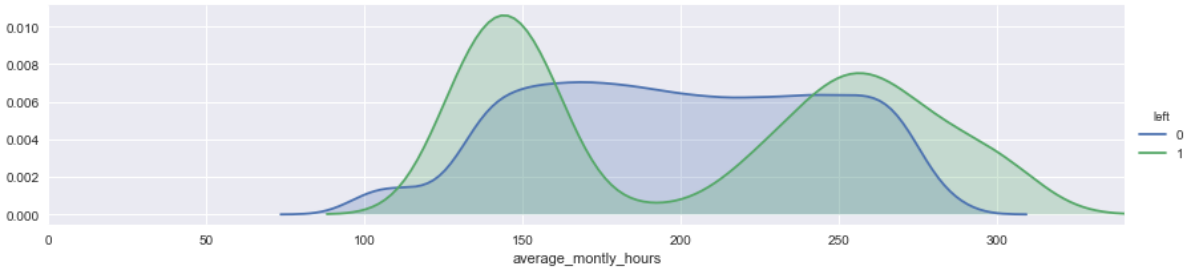
Out[121]:
```
array([0, 1, 2], dtype=int64)
```

In [122]:
```
data['promotion_last_5years'].unique()
```

Out[122]:
```
array([0, 1], dtype=int64)
```

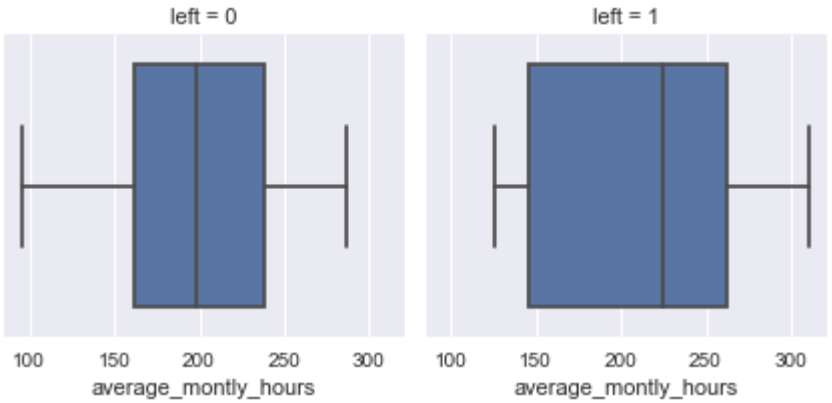
In [136]:
```
#average monthly hours
g = sns.FacetGrid(data, hue = "left", aspect = 4)
g.map(sns.kdeplot,'average_montly_hours',shade = True)
g.set(xlim=(0, 1.099*data['average_montly_hours'].max()))
g.add_legend()
```

Out[136]:
```
<seaborn.axisgrid.FacetGrid at 0x1143d978>
```



In [142]:
```
#Boxplot
g = sns.FacetGrid(data, col = 'left')
g.map(sns.boxplot, 'average_montly_hours')
np.mean(data[data['left'] ==1]['average_montly_hours']), np.mean(data[data['le
ft'] == 0] ['average_montly_hours'])
```

Out[142]: (207.41921030523662, 199.13409337676438)



In [156]:
```
#continuous variables are not good for classification. So convert them to cate
gorical ones.
#continuous to categorical
#create range using pandas
data['avg_mon_hours_range'] = pd.cut(data['average_montly_hours'], 3)
cf = data[['avg_mon_hours_range', 'left']].groupby(['avg_mon_hours_range']).me
an()
cf.sort_values(by = 'left', ascending = True)
```

Out[156]:

	left
avg_mon_hours_range	
(167.333, 238.667]	0.072888
(95.786, 167.333]	0.334932
(238.667, 310.0]	0.363340

In [157]:
```
cf
```

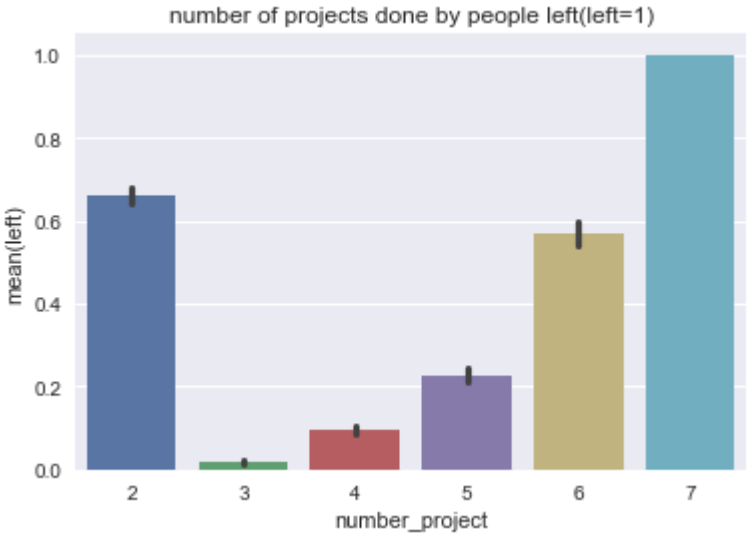
Out[157]:

	left
avg_mon_hours_range	
(95.786, 167.333]	0.334932
(167.333, 238.667]	0.072888
(238.667, 310.0]	0.363340

In [160]:
```
#replace continous values by categorical ones
data.loc[data['average_montly_hours'] <= 167.333, 'average_montly_hours'] = 0
data.loc[(data['average_montly_hours'] > 167.333) & (data['average_montly_hour
s'] <= 238.667), 'average_montly_hours'] = 1
data.loc[(data['average_montly_hours'] > 238.667) & (data['average_montly_hour
s'] <= 310.0), 'average_montly_hours'] = 2
data.drop(['avg_mon_hours_range'], axis = 1, inplace = True)
```

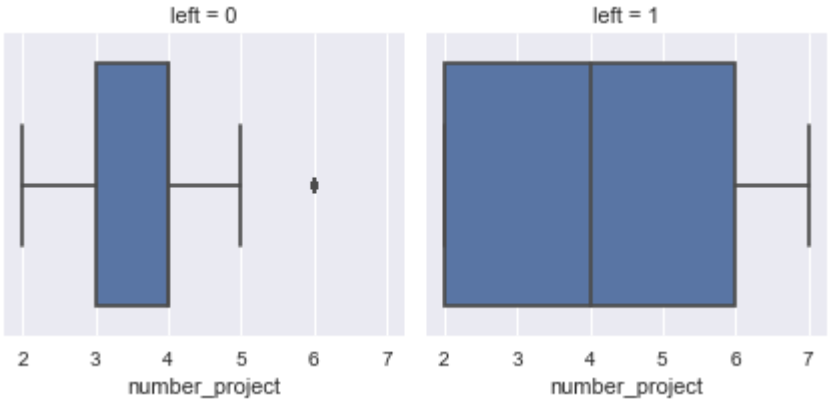
In [166]:
```
# number project
sns.barplot(x = 'number_project', y = 'left', data=data)
sns.plt.title('number of projects done by people left(left=1)')
```

Out[166]: <matplotlib.text.Text at 0x11576a90>



In [168]:
```
#boxplot
g = sns.FacetGrid(data, col='left')
g.map(sns.boxplot, 'number_project')
print('left_median : ', np.median(data[data['left'] == 1] ['number_project']))
print('not_left_median : ', np.median(data[data['left'] == 0] ['number_projec
t']))
# this indicates number of projects done by a person is not a good estimate, b
ecause it is distributed equally between left=0 and left = 1
```

```
left_median :  4.0
not_left_median :  4.0
```



In [169]:
```
# outliers dp exit, dro those observations
dropdata= data[(data['number_project'] ==8) & (data['left']==0)]
data.drop(dropdata.index, inplace= True)
```

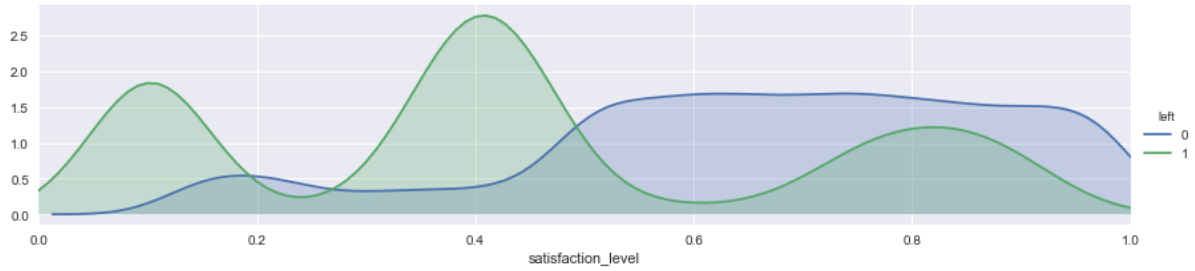
In [188]:
```
data.head()
```

Out[188]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
0	0.38	0.53	2	0	3
1	0.80	0.86	5	2	6
2	0.11	0.88	7	2	4
3	0.72	0.87	5	1	5
4	0.37	0.52	2	0	3

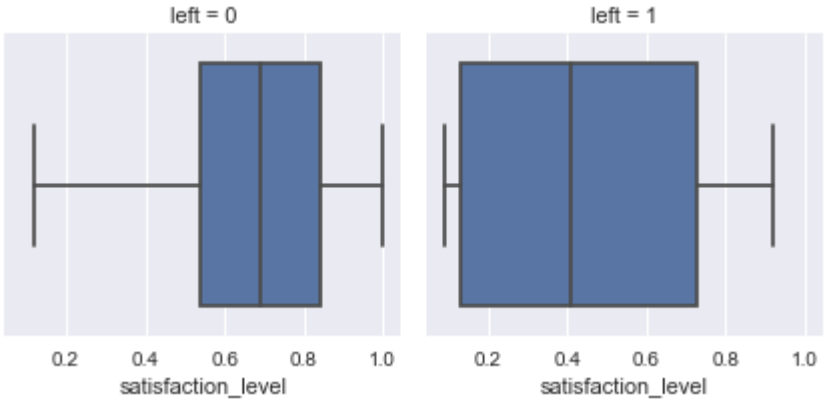
In [178]:
```
#satisfaction_level
g=sns.FacetGrid(data, hue='left', aspect=4)
g.map(sns.kdeplot, 'satisfaction_level', shade=True)
g.set(xlim=(0, data['satisfaction_level'].max()))
g.add_legend()
```

Out[178]: <seaborn.axisgrid.FacetGrid at 0x131667f0>



In [180]:
```
#boxplot
g= sns.FacetGrid(data, col= 'left')
g.map(sns.boxplot, 'satisfaction_level')
```

Out[180]: <seaborn.axisgrid.FacetGrid at 0x13351240>



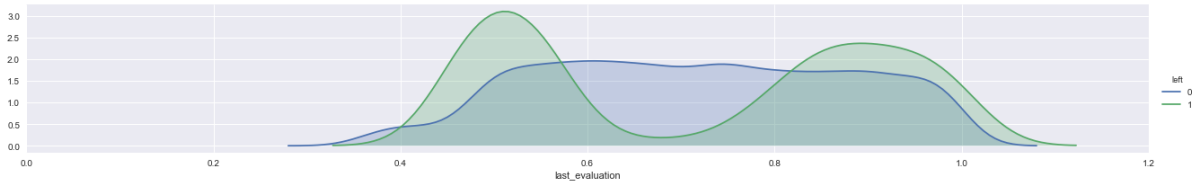
In [194]:
```
# continous to categorical
data['satisfaction_range'] = pd.cut(data['satisfaction_level'], 3)
data[['satisfaction_range', 'left']].groupby(['satisfaction_range']).mean()
```

Out[194]:

	left
satisfaction_range	
(-0.003, 1.0]	0.545455
(1.0, 2.0]	0.196339
(2.0, 3.0]	0.148668

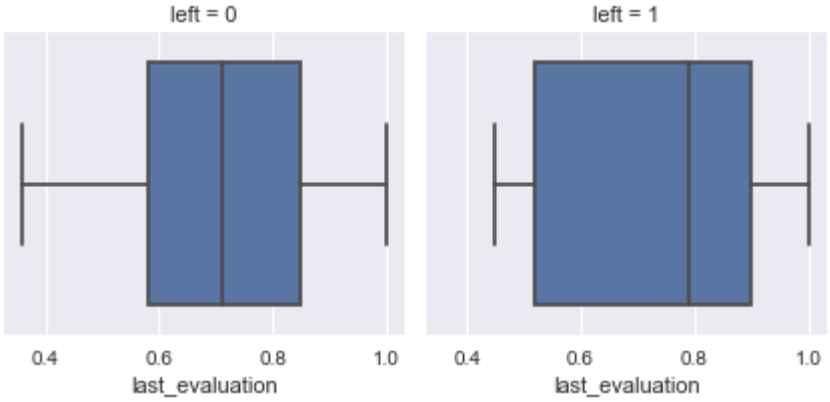
In [185]:
```
#last_evaluation
g=sns.FacetGrid(data, hue='left', aspect=6)
g.map(sns.kdeplot, 'last_evaluation', shade=True)
g.set(xlim=(0, 1.2 * data['last_evaluation'].max()))
g.add_legend()
```

Out[185]: <seaborn.axisgrid.FacetGrid at 0x135400b8>



In [187]:
```
#boxplot
g=sns.FacetGrid(data, col='left')
g.map(sns.boxplot, 'last_evaluation')
```

Out[187]: <seaborn.axisgrid.FacetGrid at 0x1415eb70>



In [198]:
```
data.loc[(data['satisfaction_level']==2), 'satisfaction_level']=1
data.loc[(data['satisfaction_level']==3), 'satisfaction_level']=2
```

In [191]:
```
# same process for evaluation_range
data['evaluation_range'] = pd.cut(data['last_evaluation'], 3)
data[['evaluation_range', 'left']].groupby(['evaluation_range']).mean()

data.drop(['satisfaction_range'], axis = 1, inplace = True)
```

Out[191]:

	left
evaluation_range	
(0.359, 0.573]	0.380430
(0.573, 0.787]	0.041594
(0.787, 1.0]	0.305770

In [193]:
```
data.loc[(data['last_evaluation']> 0.787) & (data['last_evaluation']<=1.0), 'l
ast_evaluation']=2
data.loc[(data['last_evaluation']> 0.573) & (data['last_evaluation']<=0.787),
'last_evaluation']=1
data.loc[(data['last_evaluation']<= 0.573), 'last_evaluation']=0

data.drop(['evaluation_range'], axis = 1, inplace = True)
```

In [201]:
```
data.head()
```

Out[201]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
0	0.0	0.0	2	0	3
1	2.0	2.0	5	2	6
2	0.0	2.0	7	2	4
3	2.0	2.0	5	1	5
4	0.0	0.0	2	0	3

In [200]:
```
data.drop(['satisfaction_range'], axis = 1, inplace = True)
```

In [202]:
```
data.dtypes
```

Out[202]:
```
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                      int64
salary                     int64
avg_hour_project         float64
dtype: object
```

In [203]:
```
#Train-Test split
from sklearn.model_selection import train_test_split
label = data.pop('left')
data_train, data_test, label_train, label_test = train_test_split(data, label,
 test_size = 0.2, random_state = 42)
```

In [204]:
```
#Logistic Regression
from sklearn.linear_model import LogisticRegression
logis = LogisticRegression()
logis.fit(data_train, label_train)
logis_score_train = logis.score(data_train, label_train)
print("Training score: ",logis_score_train)
logis_score_test = logis.score(data_test, label_test)
print("Testing score: ",logis_score_test)
```

```
Training score:  0.766113865618
Testing score:  0.775726495726
```

In [206]:
```
data.head()
```

Out[206]:

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
0	0.0	0.0	2	0	3
1	2.0	2.0	5	2	6
2	0.0	2.0	7	2	4
3	2.0	2.0	5	1	5
4	0.0	0.0	2	0	3

In [208]:
```
#SVM
from sklearn.svm import SVC
svm=SVC()
svm.fit(data_train, label_train)
trainingscore=svm.score(data_train, label_train)
print("training score: ", trainingscore)
testingscore=svm.score(data_test, label_test)
print("testing score: ", testingscore)
```

```
training score:  0.964780304326
testing score:  0.964444444444
```

In [ ]:
```
```