In [2]:
```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
%matplotlib inline
```

In [3]:
```
df=pd.read_csv('hranalytics.csv')
df
```

Out[3]:

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

14999 rows × 10 columns

In [4]:
```
left=df[df.left==1]
left.shape
```

Out[4]:
```
(3571, 10)
```

In [5]:
```
retained=df[df.left==0]
retained.shape
```

Out[5]:
```
(11428, 10)
```

In [6]:
```
df.groupby('left').mean()
```

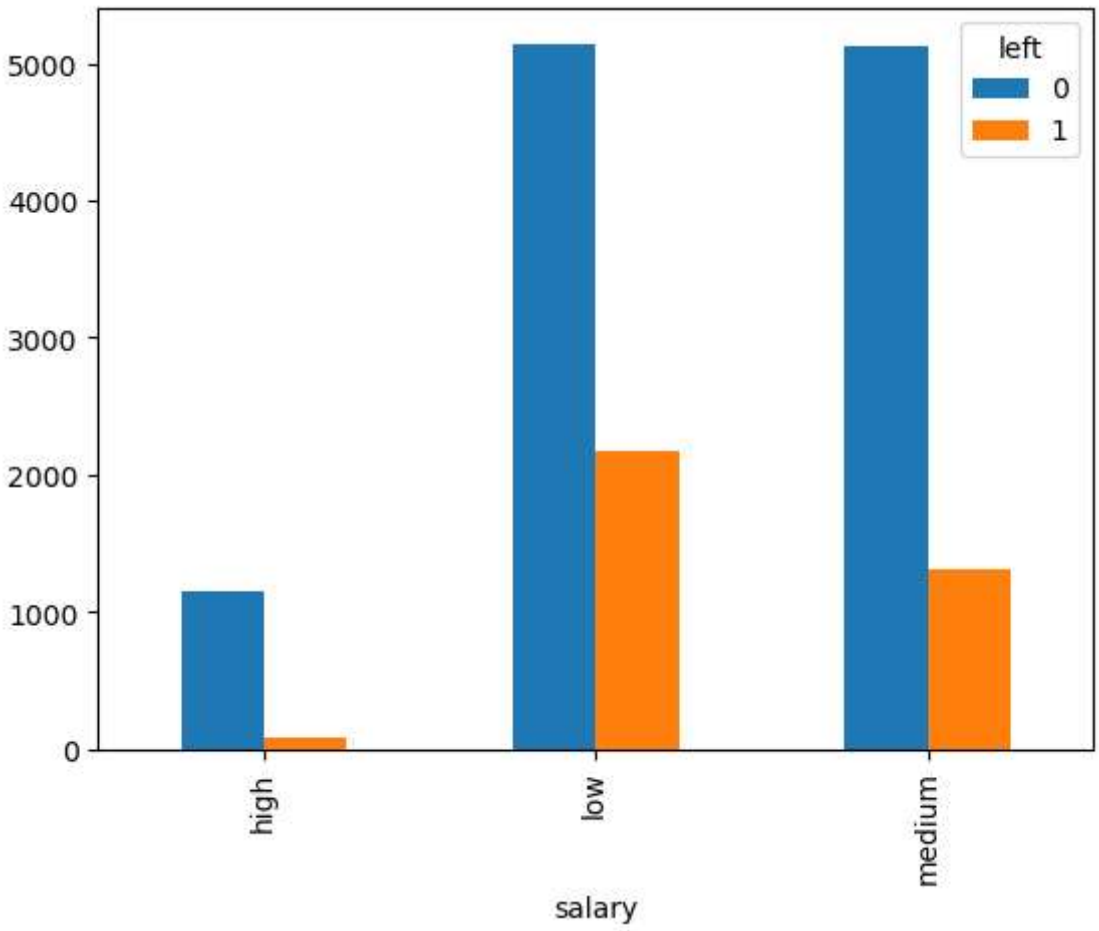
```
C:\Users\atul\AppData\Local\Temp\ipykernel_1780\588011459.py:1: FutureWarn
ing: The default value of numeric_only in DataFrameGroupBy.mean is depreca
ted. In a future version, numeric_only will default to False. Either speci
fy numeric_only or select only columns which should be valid for the funct
ion.
  df.groupby('left').mean()
```

Out[6]:

left	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spend_cc
0	0.666810	0.715473	3.786664	199.060203	3.
1	0.440098	0.718113	3.855503	207.419210	3.

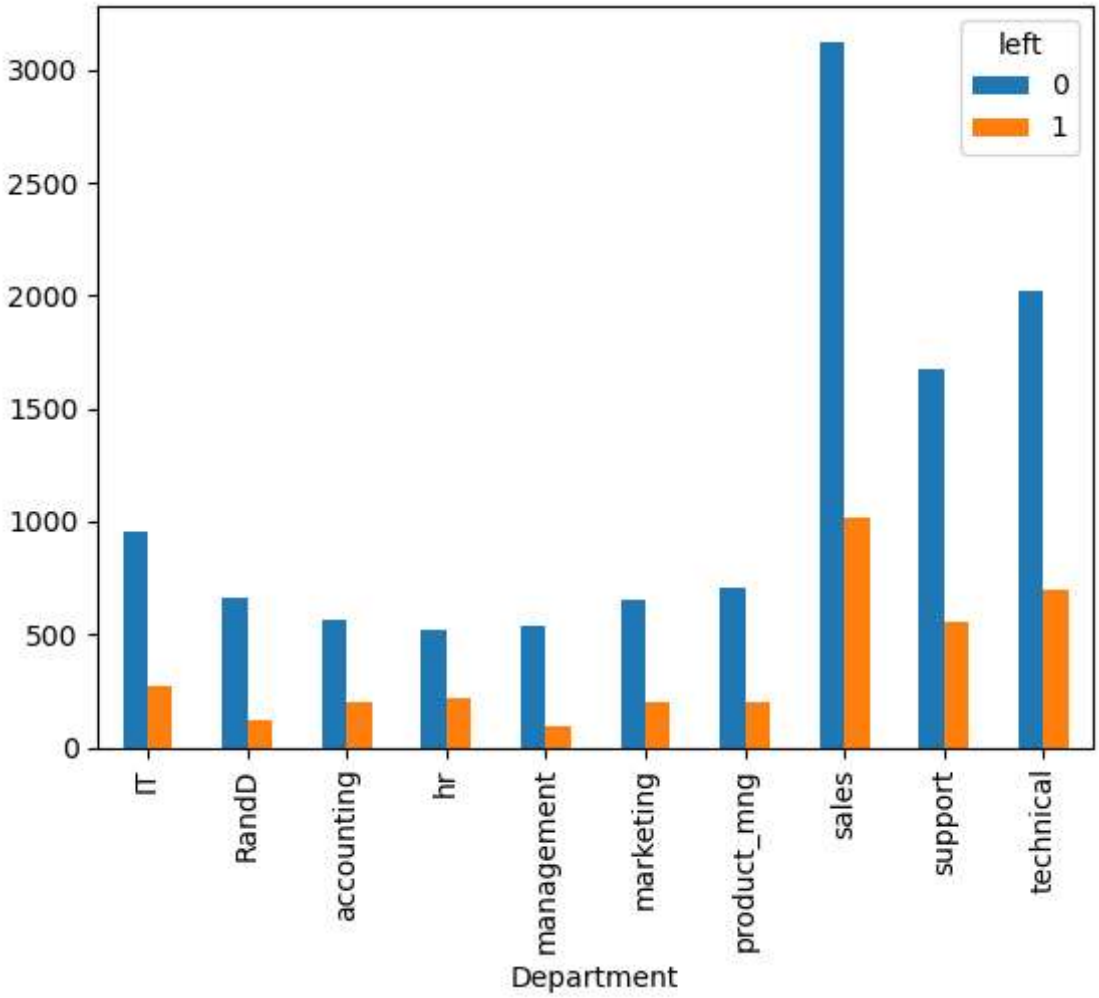
In [7]:
```
pd.crosstab(df.salary,df.left).plot(kind='bar')
```

Out[7]: <Axes: xlabel='salary'>



In [8]:
```
pd.crosstab(df.Department,df.left).plot(kind='bar')
```

Out[8]:
```
<Axes: xlabel='Department'>
```



In [9]:
```
subdf = df[['satisfaction_level','average_montly_hours','promotion_last_5ye
subdf.head()
```

Out[9]:

	satisfaction_level	average_montly_hours	promotion_last_5years	salary
0	0.38	157	0	low
1	0.80	262	0	medium
2	0.11	272	0	medium
3	0.72	223	0	low
4	0.37	159	0	low

In [10]:
```
salary_dummies = pd.get_dummies(subdf.salary, prefix="salary")
```

In [11]:
```
df_with_dummies = pd.concat([subdf,salary_dummies],axis='columns')
df_with_dummies.head()
```

Out[11]:

	satisfaction_level	average_montly_hours	promotion_last_5years	salary	salary_high	sala
0	0.38	157	0	low	0	
1	0.80	262	0	medium	0	
2	0.11	272	0	medium	0	
3	0.72	223	0	low	0	
4	0.37	159	0	low	0	

In [12]:
```
df_with_dummies.drop('salary',axis='columns',inplace=True)
df_with_dummies.head()
```

Out[12]:

	satisfaction_level	average_montly_hours	promotion_last_5years	salary_high	salary_low	s
0	0.38	157	0	0	1	
1	0.80	262	0	0	0	
2	0.11	272	0	0	0	
3	0.72	223	0	0	1	
4	0.37	159	0	0	1	

In [13]:
```
x = df_with_dummies
x.head()
```

Out[13]:

	satisfaction_level	average_montly_hours	promotion_last_5years	salary_high	salary_low	s
0	0.38	157	0	0	1	
1	0.80	262	0	0	0	
2	0.11	272	0	0	0	
3	0.72	223	0	0	1	
4	0.37	159	0	0	1	

In [14]:
```
y=df.left
y
```

Out[14]:
```
0        1
1        1
2        1
3        1
4        1
        ..
14994    1
14995    1
14996    1
14997    1
14998    1
Name: left, Length: 14999, dtype: int64
```

In [15]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,train_size=0.3)
```

In [16]:
```
from sklearn.linear_model import LogisticRegression
model=LogisticRegression()
```

In [17]:
```
model.fit(x_train,y_train)
```

Out[17]:
```
LogisticRegression
LogisticRegression()
```

In [20]:
```
model.predict(x_test)
```

Out[20]:
```
array([0, 0, 0, ..., 0, 0, 1], dtype=int64)
```

In [19]:
```
model.score(x_train,y_train)
```

Out[19]:
```
0.7781729273171816
```

In [ ]:
```

```