```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files HR_comma_sep.csv

- **HR_comma_sep.csv**(text/csv) - 551785 bytes, last modified: 2/23/2020 - 100% done

```
Saving HR_comma_sep.csv to HR_comma_sep.csv
```

```
df=pd.read_csv("HR_comma_sep.csv")
```

```
df
```

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
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

Classification Types

1)Binary classification

simple ans in yes or no

2)Multiclass Classification

more than two catagories is multiclass like congress,bjp,aap

```
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
```

```
df.left.unique()
```

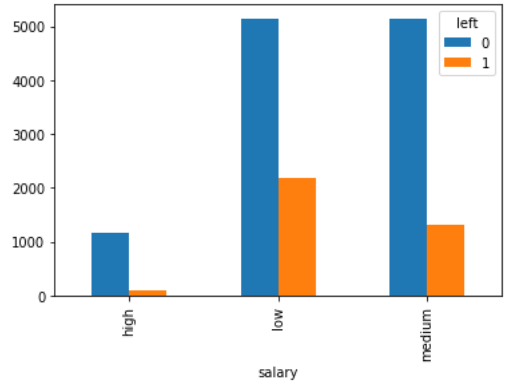
```
array([1, 0])
```

```
df.describe()
```

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
count	14999.000000	14999.000000	14999.000000	14999.000000	149
mean	0.612834	0.716102	3.803054	201.050337	
std	0.248631	0.171169	1.232592	49.943099	
min	0.090000	0.360000	2.000000	96.000000	
25%	0.440000	0.560000	3.000000	156.000000	
50%	0.640000	0.720000	4.000000	200.000000	
75%	0.820000	0.870000	5.000000	245.000000	
max	1.000000	1.000000	7.000000	310.000000	

```
pd.crosstab(df.salary,df.left).plot(kind='bar')
```

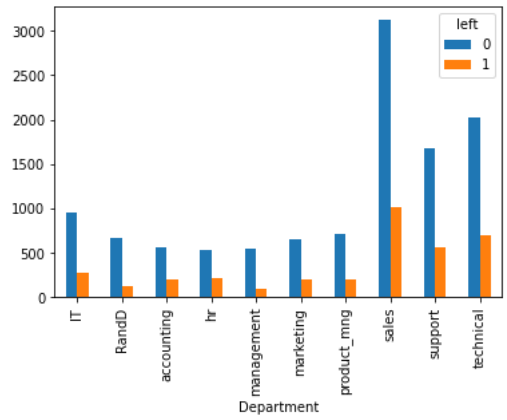
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6cb5b8da00>
```



higher salary employees as leaving less as compared to others

```
pd.crosstab(df.Department,df.left).plot(kind='bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6cb5c12790>
```



mostly sales employees are leaving the most of all departments

```
newdf=df
```

```
newdf
```

	satisfaction_level	last_evaluation	number_project	average_montly_hours	time_spen
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	

```
newdf = df[['satisfaction_level','average_montly_hours','promotion_last_5years','salary']]
```

```
newdf
```

	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
...	...	...	...	...
14994	0.40	151	0	low
14995	0.37	160	0	low
14996	0.37	143	0	low
14997	0.11	280	0	low
14998	0.37	158	0	low

14999 rows × 4 columns

It needs to be converted to numbers and we will use dummy variable for that.

```
dummies = pd.get_dummies(newdf.salary, prefix="salary")
```

prefix is used to append the column name of that dataframe

```
df2 = pd.concat([newdf,dummies],axis='columns')
```

concat means joining two datasets

```
df2
```

	satisfaction_level	average_montly_hours	promotion_last_5years	salary	salary_high
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
...	...	...	...	...	...
14994	0.40	151	0	low	0
14995	0.37	160	0	low	0
14996	0.37	143	0	low	0
14997	0.11	280	0	low	0
14998	0.37	158	0	low	0

14999 rows × 7 columns

```
df2.drop('salary',axis='columns',inplace=True)
```

```
df2
```

	satisfaction_level	average_montly_hours	promotion_last_5years	salary_high	salary
0	0.38	157	0	0	
1	0.80	262	0	0	
2	0.11	272	0	0	
3	0.72	223	0	0	
4	0.37	159	0	0	
...	...	...	...	...	
14994	0.40	151	0	0	
14995	0.37	160	0	0	
14996	0.37	143	0	0	
14997	0.11	280	0	0	
14998	0.37	158	0	0	

14999 rows × 6 columns

```
x=df2
x
```

	satisfaction_level	average_montly_hours	promotion_last_5years	salary_high	salary
0	0.38	157	0	0	
1	0.80	262	0	0	
2	0.11	272	0	0	
3	0.72	223	0	0	
4	0.37	159	0	0	
...	...	...	...	...	
14994	0.40	151	0	0	
14995	0.37	160	0	0	
14996	0.37	143	0	0	
14997	0.11	280	0	0	
14998	0.37	158	0	0	

14999 rows × 6 columns

```
y=df.left
y
```

```
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
```

```
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest=train_test_split(x,y,test_size=0.25)
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(xtrain,ytrain)
model.predict(xtest)
```

```
array([0, 0, 0, ..., 0, 0, 0])
```

```
##Accuracy
model.score(xtest,ytest)
```

```
0.7784
```

0s completed at 1:17 PM