```
import numpy as np
import pandas as pd
df=pd.read_csv("sample_data/dataset.csv")
df.head()
```

	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects	Perc in Co Archit
0	69	63	78	87	94	94	
1	78	62	73	60	71	70	
2	71	86	91	87	61	81	
3	76	87	60	84	89	73	
4	92	62	90	67	71	89	

5 rows × 39 columns

> #data preprocessing
>
> #checking null values

```
df.isna()
```

	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects	P in Arc
0	False	False	False	False	False	False	
1	False	False	False	False	False	False	
2	False	False	False	False	False	False	
3	False	False	False	False	False	False	
4	False	False	False	False	False	False	

```
df=df.dropna()
df.isna().sum().sum()
```

```
0
```

```
df.duplicated()
df.drop_duplicates()
```

	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects	P in Arc
0	69	63	78	87	94	94	
1	78	62	73	60	71	70	

```
df.dtypes
```

```
Acedamic percentage in Operating Systems     int64
percentage in Algorithms                     int64
Percentage in Programming Concepts           int64
Percentage in Software Engineering           int64
Percentage in Computer Networks              int64
Percentage in Electronics Subjects           int64
Percentage in Computer Architecture          int64
Percentage in Mathematics                    int64
Percentage in Communication skills           int64
Hours working per day                        int64
Logical quotient rating                      int64
hackathons                                   int64
coding skills rating                         int64
public speaking points                       int64
can work long time before system?           object
self-learning capability?                   object
Extra-courses did                           object
certifications                              object
workshops                                   object
talenttests taken?                          object
olympiads                                   object
reading and writing skills                  object
memory capability score                     object
Interested subjects                         object
interested career area                      object
Job/Higher Studies?                         object
Type of company want to settle in?          object
Taken inputs from seniors or elders         object
interested in games                         object
Interested Type of Books                    object
Salary Range Expected                       object
In a Realtionship?                          object
Gentle or Tuff behaviour?                   object
Management or Technical                     object
Salary/work                                 object
hard/smart worker                           object
worked in teams ever?                       object
Introvert                                   object
Suggested Job Role                          object
dtype: object
```

```
df['can work long time before system?']
```

```
0      yes
1      yes
```

```
2       yes
3        no
4        no
        ...
4280     no
4281    yes
4282     no
4283     no
4284    yes
Name: can work long time before system?, Length: 4285, dtype: object
```

```
df['can work long time before system?']=df['can work long time before system?'].astype('ca
df['can work long time before system?']=df['can work long time before system?'].cat.codes
df['can work long time before system?']
```

```
0       1
1       1
2       1
3       0
4       0
       ..
4280    0
4281    1
4282    0
4283    0
4284    1
Name: can work long time before system?, Length: 4285, dtype: int8
```

```
for col in df.select_dtypes(include=['object']).columns:
    df[col] = df[col].astype('category')
    df[col] = df[col].cat.codes
df.dtypes
```

```
Acedamic percentage in Operating Systems    int64
percentage in Algorithms                    int64
Percentage in Programming Concepts          int64
Percentage in Software Engineering          int64
Percentage in Computer Networks             int64
Percentage in Electronics Subjects          int64
Percentage in Computer Architecture         int64
Percentage in Mathematics                   int64
Percentage in Communication skills          int64
Hours working per day                       int64
Logical quotient rating                     int64
hackathons                                  int64
coding skills rating                        int64
public speaking points                      int64
can work long time before system?            int8
self-learning capability?                    int8
Extra-courses did                            int8
certifications                               int8
workshops                                    int8
talenttests taken?                           int8
olympiads                                    int8
reading and writing skills                   int8
memory capability score                      int8
Interested subjects                          int8
```

```
interested career area                    int8
Job/Higher Studies?                       int8
Type of company want to settle in?        int8
Taken inputs from seniors or elders       int8
interested in games                       int8
Interested Type of Books                  int8
Salary Range Expected                     int8
In a Realtionship?                        int8
Gentle or Tuff behaviour?                 int8
Management or Technical                   int8
Salary/work                               int8
hard/smart worker                         int8
worked in teams ever?                     int8
Introvert                                 int8
Suggested Job Role                        int8
dtype: object
```

```
# Studing dataset: Descriptive Analysis
```

```
df.describe()
```

	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects
count	4285.000000	4285.000000	4285.000000	4285.000000	4285.000000	4285.000000
mean	76.824504	76.890782	77.028705	77.070478	77.151459	76.930222
std	10.024829	10.092972	10.210363	10.142135	10.025024	10.187781
min	60.000000	60.000000	60.000000	60.000000	60.000000	60.000000
25%	68.000000	68.000000	68.000000	68.000000	69.000000	68.000000
50%	77.000000	77.000000	77.000000	77.000000	77.000000	77.000000
75%	85.000000	86.000000	86.000000	86.000000	86.000000	86.000000
max	94.000000	94.000000	94.000000	94.000000	94.000000	94.000000

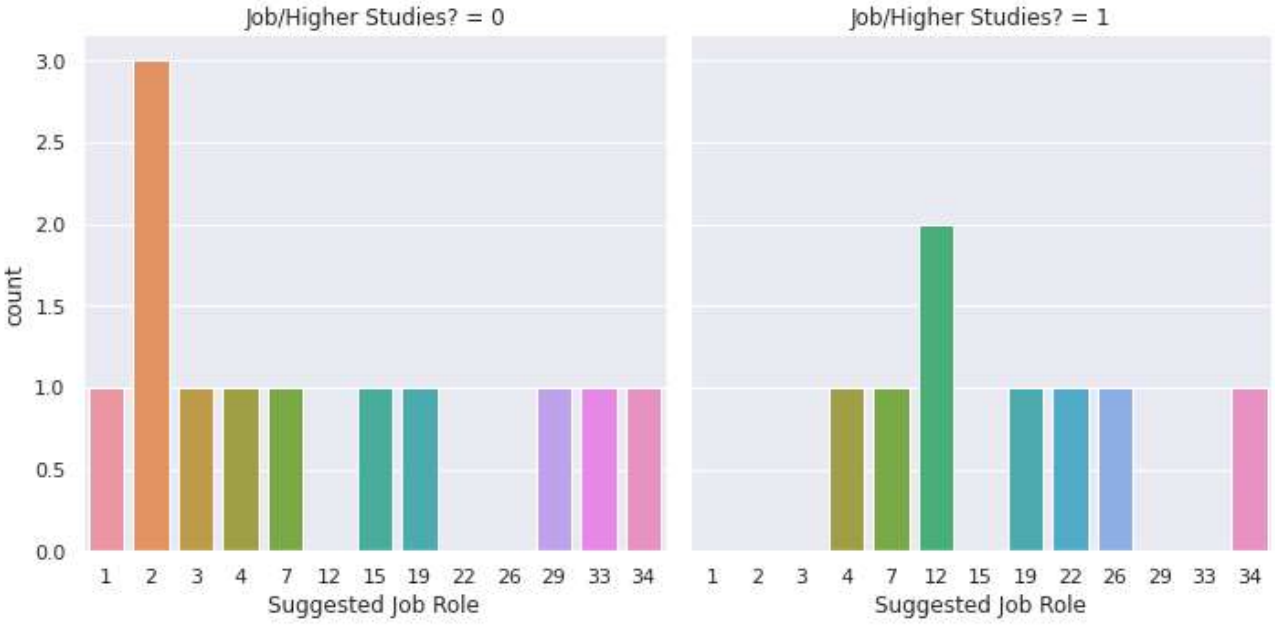
8 rows × 39 columns

```
left = df.groupby('Suggested Job Role')
left.mean()
```

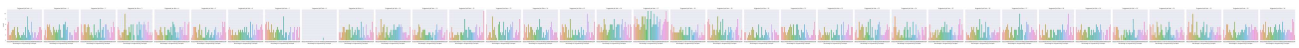
Suggested Job Role	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects
0	75.837607	77.965812	76.709402	75.222222	76.452991	76.538462
1	77.365079	78.873016	76.476190	78.150794	75.865079	77.285714
2	76.068182	77.537879	76.568182	76.833333	77.462121	76.469697
3	76.091603	76.824427	77.351145	76.427481	76.916031	76.961832
4	75.834646	75.488189	78.307087	76.078740	76.496063	77.653543
5	76.045872	78.394495	77.807339	79.302752	75.853211	77.504587
6	78.480000	75.080000	78.112000	77.960000	74.960000	76.592000
7	75.404762	77.119048	75.039683	77.206349	78.023810	78.079365
8	82.000000	66.000000	81.000000	88.000000	86.000000	66.000000
9	77.558333	77.983333	76.866667	77.291667	76.650000	76.425000
10	77.273438	75.765625	77.070312	77.515625	76.250000	76.578125
11	76.980769	76.750000	76.567308	77.625000	77.663462	77.701923
12	77.443478	77.121739	79.200000	77.704348	76.408696	75.904348
13	76.268293	76.934959	77.016260	76.276423	76.934959	78.650407
14	78.585366	76.731707	76.170732	76.601626	76.869919	77.292683
15	76.308943	76.943089	77.154472	77.560976	77.756098	77.333333
16	76.816568	77.325444	76.940828	76.745562	77.124260	76.769231
17	76.579832	76.647059	77.142857	76.399160	77.941176	77.075630
18	78.157895	76.745614	75.929825	76.263158	76.701754	74.517544
19	75.654135	77.000000	74.909774	76.323308	78.172932	76.240602
20	77.363636	76.784091	77.102273	78.511364	77.272727	77.306818
21	76.360656	75.983607	75.606557	76.270492	77.975410	77.762295
22	75.451923	76.663462	78.567308	76.625000	76.826923	77.067308
23	76.379845	77.240310	77.178295	77.844961	78.209302	76.139535
24	76.767241	76.543103	76.836207	77.241379	77.250000	77.689655
25	78.606557	76.081967	78.024590	76.245902	78.614754	77.278689
26	75.800000	77.043478	77.556522	77.278261	77.173913	76.339130
27	76.566176	76.948529	77.235294	77.404412	76.720588	76.183824
28	77.592000	76.128000	78.536000	76.416000	77.768000	77.168000

29	76.893443	77.688525	76.475410	76.934426	77.303279	76.860656
30	76.212963	77.203704	77.759259	77.435185	78.768519	78.222222
31	76.008475	76.271186	75.779661	76.245763	76.567797	75.805085
32	78.716535	77.464567	75.866142	78.527559	77.370079	76.023622
33	77.224806	77.023256	78.062016	76.868217	76.596899	76.720930
34	77.421429	76.357143	77.385714	78.157143	77.400000	77.750000

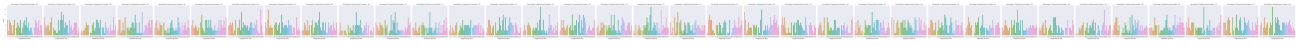
```
import seaborn as sns
sns.set(style="darkgrid")
g = sns.catplot(x="Suggested Job Role",col="Job/Higher Studies?",
                data=df.head(20), kind="count"
                );
```




```
g = sns.catplot(x="Percentage in Programming Concepts", col="Suggested Job Role",
                data=df, kind="count"
                );
```



```
g = sns.catplot(x="Suggested Job Role",col="Percentage in Programming Concepts",
                data=df, kind="count"
                );
```



```
data = df.iloc[:,:-1].values
label = df.iloc[:,-1]
```

```
#Label Encoding: COnverting To Numeric values
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder = LabelEncoder()

for i in range(14,38):
    data[:,i] = labelencoder.fit_transform(data[:,i])

#Normalizing the data
from sklearn.preprocessing import Normalizer
data1=data[:,:14]
normalized_data = Normalizer().fit_transform(data1)

data2=data[:,14:]
df1 = np.append(normalized_data,data2,axis=1)


#Combining into a dataset
df2=df.iloc[:,:-1]
dataset = pd.DataFrame(df1,columns=df2.columns)
dataset
```

urs ing day	...	interested in games	Interested Type of Books	Salary Range Expected	In a Realtionship?	Gentle or Tuff behaviour?	Management or Technical	Sal
186	...	0.0	21.0	1.0	0.0	1.0	0.0	
843	...	1.0	5.0	1.0	1.0	0.0	1.0	
706	...	1.0	29.0	0.0	0.0	1.0	0.0	
213	...	0.0	23.0	0.0	1.0	0.0	0.0	
268	...	1.0	7.0	1.0	0.0	1.0	0.0	
...	...	...	...	...	...	...	...	
244	...	1.0	0.0	0.0	1.0	1.0	1.0	
895	...	1.0	16.0	0.0	1.0	0.0	0.0	
738	...	0.0	20.0	0.0	1.0	0.0	0.0	
152	...	1.0	27.0	1.0	0.0	0.0	1.0	
421	...	0.0	1.0	0.0	1.0	1.0	0.0	

```
# For label
label = df.iloc[:,-1]
original=label.unique()
label=label.values
```

```
label=label.values
label2 = labelencoder.fit_transform(label)
y=pd.DataFrame(label2,columns=["Suggested Job Role"])
numeric=y["Suggested Job Role"].unique()
Y = pd.DataFrame({'Suggested Job Role':original, 'Associated Number':numeric})


Y
```

	Suggested Job Role	Associated Number
0	7	7
1	19	19
2	29	29
3	2	2
4	26	26
5	1	1
6	4	4

```
dataset = pd.read_csv("sample_data/dataset.csv")
print(np.shape(dataset))
dataset.head()
```

```
(20000, 39)
```

	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects	Perc in Co Archit
0	69	63	78	87	94	94	
1	78	62	73	60	71	70	
2	71	86	91	87	61	81	
3	76	87	60	84	89	73	
4	92	62	90	67	71	89	

5 rows × 39 columns

```
data = dataset.iloc[:,:-1].values
label = dataset.iloc[:,-1].values
len(data[0])
```

```
38
```

```
dataset.iloc[:,14:38]
```

	can work long time before system?	self-learning capability?	Extra-courses did	certifications	workshops	talenttests taken?	olym
0	yes	yes	yes	shell programming	cloud computing	no	
1	yes	no	yes	machine learning	database security	no	
2	yes	no	yes	app development	web technologies	no	
3	no	yes	no	python	data science	yes	
4	no	no	no	app development	cloud computing	no	
...	...	...	...	...	...	...	
19995	yes	no	no	app development	cloud computing	yes	
19996	yes	no	no	full stack	game development	no	
19997	yes	yes	yes	information security	database security	yes	
19998	no	no	no	full stack	cloud computing	no	
19999	yes	yes	yes	app development	database security	no	

20000 rows × 24 columns

```
dataset.iloc[:,:14]
```

	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects	i Ar
0	69	63	78	87	94	94	
1	78	62	73	60	71	70	
2	71	86	91	87	61	81	
3	76	87	60	84	89	73	
4	92	62	90	67	71	89	
...	...	...	...	...	...	...	
19996	80	69	83	87	82	66	

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
```

```
labelencoder = LabelEncoder()
```

```
for i in range(14,38):
    data[:,i] = labelencoder.fit_transform(data[:,i])
data[:5]
```

```
array([[69, 63, 78, 87, 94, 94, 87, 84, 61, 9, 4, 0, 4, 8, 1, 1, 1, 8, 0,
        0, 1, 0, 0, 4, 4, 0, 8, 0, 0, 21, 1, 0, 1, 0, 0, 0, 1, 0],
       [78, 62, 73, 60, 71, 70, 73, 84, 91, 12, 7, 1, 2, 3, 1, 0, 1, 5,
        2, 0, 0, 2, 1, 7, 0, 1, 4, 1, 1, 5, 1, 1, 0, 1, 0, 0, 0, 1],
       [71, 86, 91, 87, 61, 81, 72, 72, 94, 11, 1, 4, 1, 3, 1, 0, 1, 0,
        7, 0, 1, 2, 0, 6, 2, 0, 5, 1, 1, 29, 0, 0, 1, 0, 1, 0, 0, 1],
       [76, 87, 60, 84, 89, 73, 62, 88, 69, 7, 1, 1, 2, 5, 0, 1, 0, 6, 1,
        1, 0, 1, 0, 7, 5, 0, 7, 0, 0, 23, 0, 1, 0, 0, 1, 1, 1, 1],
       [92, 62, 90, 67, 71, 89, 73, 71, 73, 4, 5, 4, 6, 3, 0, 0, 0, 0, 0,
        0, 0, 2, 0, 0, 5, 0, 9, 0, 1, 7, 1, 0, 1, 0, 1, 0, 1, 1]],
      dtype=object)
```

```
from sklearn.preprocessing import Normalizer
```

```
data1=data[:,:14]
```

```
normalized_data = Normalizer().fit_transform(data1)
print(normalized_data.shape)
```

```
(20000, 14)
```

```
data2=data[:,14:]
data2.shape
```

```
(20000, 24)
```

```
df1 = np.append(normalized_data,data2,axis=1)
```

```
df1.shape
```

```
(20000, 38)
```

> #adding headers

```
X1 = pd.DataFrame(df1,columns=['Acedamic percentage in Operating Systems', 'percentage in
      'Percentage in Programming Concepts',
      'Percentage in Software Engineering', 'Percentage in Computer Networks',
      'Percentage in Electronics Subjects',
      'Percentage in Computer Architecture', 'Percentage in Mathematics',
      'Percentage in Communication skills', 'Hours working per day',
      'Logical quotient rating', 'hackathons', 'coding skills rating',
      'public speaking points', 'can work long time before system?',
      'self-learning capability?', 'Extra-courses did', 'certifications',
      'workshops', 'talenttests taken?', 'olympiads',
      'reading and writing skills', 'memory capability score',
      'Interested subjects', 'interested career area ', 'Job/Higher Studies?',
      'Type of company want to settle in?',
      'Taken inputs from seniors or elders', 'interested in games',
      'Interested Type of Books', 'Salary Range Expected',
      'In a Realtionship?', 'Gentle or Tuff behaviour?',
      'Management or Technical', 'Salary/work', 'hard/smart worker',
      'worked in teams ever?', 'Introvert'])
```

```
X1.head()
```

	Acedamic percentage in Operating Systems	percentage in Algorithms	Percentage in Programming Concepts	Percentage in Software Engineering	Percentage in Computer Networks	Percentage in Electronics Subjects	Perc in Co Archit
0	0.28509	0.260299	0.322276	0.359461	0.388383	0.388383	0.
1	0.34998	0.278189	0.327545	0.269215	0.318571	0.314085	0.
2	0.295012	0.357339	0.378115	0.361494	0.253461	0.336563	0.
3	0.328025	0.375503	0.258967	0.362554	0.384135	0.315077	
4	0.397157	0.267649	0.388523	0.289234	0.306502	0.384206	0.

5 rows × 38 columns

```
label = labelencoder.fit_transform(label)
print(len(label))
```

```
20000
```

```
y=pd.DataFrame(label,columns=["Suggested Job Role"])
y.head()
```

	Suggested Job Role
0	7
1	18
2	18
3	28
4	2

Decision Tree Classifier

```
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn import preprocessing
from sklearn.metrics import accuracy_score
```

```
X_train,X_test,y_train,y_test=train_test_split(X1,y,test_size=0.2,random_state=10)
```

```
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train, y_train)
```

```
from sklearn.metrics import confusion_matrix,accuracy_score
```

```
y_pred = clf.predict(X_test)
```

```
y_pred
```

```
array([29, 29,  6, ...,  2,  3, 28])
```

```
cm = confusion_matrix(y_test,y_pred)
accuracy = accuracy_score(y_test,y_pred)
```

```
print("confusion matrics=",cm)
print("  ")
print("accuracy=",accuracy*100)
```

```
confusion matrics= [[2 9 0 ... 5 2 4]
 [3 4 1 ... 1 5 4]
 [3 2 3 ... 2 3 4]
 ...
 [5 4 4 ... 3 1 5]
 [3 4 1 ... 7 5 5]
```

```
 [3 6 2 ... 4 1 2]]

accuracy= 2.65
```

Decision Tree with Entropy

```
clf_entropy = tree.DecisionTreeClassifier(criterion = "entropy", random_state = 10)
clf_entropy.fit(X_train, y_train)
```

```
DecisionTreeClassifier(criterion='entropy', random_state=10)
```

```
entropy_y_pred=clf_entropy.predict(X_test)
```

```
cm_entopy = confusion_matrix(y_test,entropy_y_pred)
```

```
entropy_accuracy = accuracy_score(y_test,entropy_y_pred)
```

```
print("confusion matrics=",cm_entopy)
print("  ")
print("accuracy=",entropy_accuracy*100)
```

```
confusion matrics= [[1 3 7 ... 3 2 2]
 [2 4 3 ... 3 2 4]
 [1 2 4 ... 0 3 1]
 ...
 [1 6 3 ... 0 5 2]
 [3 2 4 ... 2 2 4]
 [5 1 6 ... 6 6 4]]

accuracy= 2.7
```

SVM Classifier

```
from sklearn import svm
```

```
clf = svm.SVC()
clf.fit(X_train, y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:993: DataConversi
  y = column_or_1d(y, warn=True)
SVC()
```

```
svm_y_pred = clf.predict(X_test)
```

```
svm_cm = confusion_matrix(y_test,svm_y_pred)
svm_accuracy = accuracy_score(y_test,svm_y_pred)
print("confusion matrics=",svm_cm)
```

```
print("  ")
print("accuracy=",svm_accuracy*100)
```

```
confusion matrics= [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]

accuracy= 5.6000000000000005
```

XGBoost

```
X_train,X_test,y_train,y_test=train_test_split(X1,y,test_size=0.3,random_state=10)
```

```
X_train.shape
```

```
(14000, 38)
```

```
X_train=pd.to_numeric(X_train.values.flatten())
```

```
X_train=X_train.reshape((14000,38))
```

```
from xgboost import XGBClassifier
```

```
model = XGBClassifier()
model.fit(X_train, y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/preprocessing/_label.py:98: DataConve
  y = column_or_1d(y, warn=True)
/usr/local/lib/python3.7/dist-packages/sklearn/preprocessing/_label.py:133: DataConv
  y = column_or_1d(y, warn=True)
XGBClassifier(objective='multi:softprob')
```

```
xgb_y_pred = clf.predict(X_test)
```

```
xgb_cm = confusion_matrix(y_test,xgb_y_pred)
xgb_accuracy = accuracy_score(y_test,xgb_y_pred)
print("confusion matrics=",xgb_cm)
print("  ")
print("accuracy=",xgb_accuracy*100)
```

```
confusion matrics= [[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
```

```
 [0 0 0 ... 0 0 0]]

accuracy= 5.75
```