```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import preprocessing
```

```
data = pd.read_csv('/content/bank.csv')
print(data.shape)
data.head()
```

```
(11162, 17)
```

	age	job	marital	education	default	balance	housing	loan	contact	day	mo
0	59	admin.	married	secondary	no	2343	yes	no	unknown	5	r
1	56	admin.	married	secondary	no	45	no	no	unknown	5	r
2	41	technician	married	secondary	no	1270	yes	no	unknown	5	r
3	55	services	married	secondary	no	2476	yes	no	unknown	5	r
4	54	admin.	married	tertiary	no	184	no	no	unknown	5	r

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11162 entries, 0 to 11161
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        11162 non-null  int64 
 1   job        11162 non-null  object
 2   marital    11162 non-null  object
 3   education  11162 non-null  object
 4   default    11162 non-null  object
 5   balance    11162 non-null  int64 
 6   housing    11162 non-null  object
 7   loan       11162 non-null  object
 8   contact    11162 non-null  object
 9   day        11162 non-null  int64 
 10  month      11162 non-null  object
 11  duration   11162 non-null  int64 
 12  campaign   11162 non-null  int64 
 13  pdays      11162 non-null  int64 
 14  previous   11162 non-null  int64 
 15  poutcome   11162 non-null  object
 16  deposit    11162 non-null  object
```

```
    dtypes: int64(7), object(10)
    memory usage: 1.4+ MB
```

```
data.isnull().any()
```

```
    age          False
    job          False
    marital      False
    education    False
    default      False
    balance      False
    housing      False
    loan         False
    contact      False
    day          False
    month        False
    duration     False
    campaign     False
    pdays        False
    previous     False
    poutcome     False
    deposit      False
    dtype: bool
```

```
data.describe()
```

	age	balance	day	duration	campaign	pdays
count	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000	11162.000000
mean	41.231948	1528.538524	15.658036	371.993818	2.508421	51.330407
std	11.913369	3225.413326	8.420740	347.128386	2.722077	108.758282
min	18.000000	-6847.000000	1.000000	2.000000	1.000000	-1.000000
25%	32.000000	122.000000	8.000000	138.000000	1.000000	-1.000000
50%	39.000000	550.000000	15.000000	255.000000	2.000000	-1.000000
75%	49.000000	1708.000000	22.000000	496.000000	3.000000	20.750000
max	95.000000	81204.000000	31.000000	3881.000000	63.000000	854.000000

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data['job'].unique()
```

```
    array(['admin.', 'technician', 'services', 'management', 'retired',
           'blue-collar', 'unemployed', 'entrepreneur', 'housemaid',
           'unknown', 'self-employed', 'student'], dtype=object)
```

```
data['marital'].unique()
```

```
     array(['married', 'single', 'divorced'], dtype=object)
```

```
data['deposit'].unique()
```

```
     array(['yes', 'no'], dtype=object)
```

```
data['education'].unique()
```

```
     array(['secondary', 'tertiary', 'primary', 'unknown'], dtype=object)
```

```
data['default'].unique()
```

```
     array(['no', 'yes'], dtype=object)
```

```
data['housing'].unique()
```

```
     array(['yes', 'no'], dtype=object)
```

```
data['loan'].unique()
```

```
     array(['no', 'yes'], dtype=object)
```

```
data['contact'].unique()
```

```
     array(['unknown', 'cellular', 'telephone'], dtype=object)
```

```
data['month'].unique()
```

```
     array(['may', 'jun', 'jul', 'aug', 'oct', 'nov', 'dec', 'jan', 'feb',
            'mar', 'apr', 'sep'], dtype=object)
```

```
data['poutcome'].unique()
```

```
     array(['unknown', 'other', 'failure', 'success'], dtype=object)
```

```
#Label Encoding for all Textual Columns
data['job'] = le.fit_transform(data['job'])
data['marital'] = le.fit_transform(data['marital'])
data['education'] = le.fit_transform(data['education'])
data['default'] = le.fit_transform(data['default'])
data['housing'] = le.fit_transform(data['housing'])
data['loan'] = le.fit_transform(data['loan'])
data['contact'] = le.fit_transform(data['contact'])
data['month'] = le.fit_transform(data['month'])
data['poutcome'] = le.fit_transform(data['poutcome'])
data['deposit'] = le.fit_transform(data['deposit'])
```

```
data
```

	age	job	marital	education	default	balance	housing	loan	contact	day	mont
0	59	0	1	1	0	2343	1	0	2	5	
1	56	0	1	1	0	45	0	0	2	5	
2	41	9	1	1	0	1270	1	0	2	5	
3	55	7	1	1	0	2476	1	0	2	5	
4	54	0	1	2	0	184	0	0	2	5	
...	...	...	...	...	...	...	...	...	...	...	.
11157	33	1	2	0	0	1	1	0	0	20	
11158	39	7	1	1	0	733	0	0	2	16	
11159	32	9	2	1	0	29	0	0	0	19	
11160	43	9	1	1	0	0	0	1	0	8	
11161	34	9	1	1	0	0	0	0	0	9	

11162 rows × 17 columns

```
#Need one hot encoding for poutcome, month, contact, education, marital, job split into x and
x=data.iloc[:,:15].values
x1=x
y=data.iloc[:,15:].values
```

```
x.shape
```

```
(11162, 15)
```

```
y.shape
```

```
(11162, 2)
```

```
type(x)
```

```
numpy.ndarray
```

```
#One hot encoding for job,marital,education
from sklearn.preprocessing import OneHotEncoder
one = OneHotEncoder()
```

```
z= one.fit_transform(x[:,1:4]).toarray()
z.shape

     (11162, 19)
```

```
x = np.delete(x,1,axis=1)
x = np.delete(x,2,axis=1)
x = np.delete(x,3,axis=1)
x = np.concatenate((x,z),axis=1)
x

     array([[59.,  1.,  0., ...,  1.,  0.,  0.],
            [56.,  1.,  0., ...,  1.,  0.,  0.],
            [41.,  1.,  0., ...,  1.,  0.,  0.],
            ...,
            [32.,  2.,  0., ...,  1.,  0.,  0.],
            [43.,  1.,  0., ...,  1.,  0.,  0.],
            [34.,  1.,  0., ...,  1.,  0.,  0.]])
```

```
x[:,7]

     array([8., 8., 8., ..., 1., 8., 5.])
```

```
#One hot encoding for month
z= one.fit_transform(x[:,7:8]).toarray()
x = np.delete(x,7,axis=1)
x = np.concatenate((x,z),axis=1)
x.shape

     (11162, 42)
```

```
#One hot encoding for contact
z= one.fit_transform(x[:,5:6]).toarray()
x = np.delete(x,5,axis=1)
x = np.concatenate((x,z),axis=1)
x.shape

     (11162, 74)
```

```
y.shape

     (11162, 2)
```

```
#One hot encoding for poutcome
z= one.fit_transform(y[:,:1]).toarray()
y = np.delete(y,0,axis=1)
y = np.concatenate((y,z),axis=1)
y.shape

     (11162, 5)
```

```
#Splitting train-test
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test= train_test_split(x,y,test_size=0.2,random_state=0)
#0.2 means 20% data will be test
print(x_train.shape)
print(x_test.shape)
```

```
(8929, 74)
(2233, 74)
```

```
#Feature scaling
#Standard scaling = (x-mean)/std. dev
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.fit_transform(x_test)
```

```
x_train
```

```
array([[ 0.64941897, -0.30765462, -0.12621015, ..., -0.18321763,
        -0.2131383 , -0.1127064 ],
       [ 1.48737591, -1.90849849, -0.12621015, ..., -0.18321763,
        -0.2131383 , -0.1127064 ],
       [-0.69131213, -0.30765462, -0.12621015, ..., -0.18321763,
        -0.2131383 , -0.1127064 ],
       ...,
       [-0.43992505, -0.30765462, -0.12621015, ..., -0.18321763,
        -0.2131383 , -0.1127064 ],
       [-0.85890352, -0.30765462, -0.12621015, ..., -0.18321763,
        -0.2131383 , -0.1127064 ],
       [ 1.57117161, -0.30765462, -0.12621015, ..., -0.18321763,
        -0.2131383 , -0.1127064 ]])
```

```
x_test
```

```
array([[-0.01351615, -0.36290363, -0.11268723, ..., -0.21424668,
        -0.20493218, -0.11268723],
       [ 1.25461704, -0.36290363, -0.11268723, ..., -0.21424668,
        -0.20493218, -0.11268723],
       [-0.94348049,  1.22916673, -0.11268723, ..., -0.21424668,
        -0.20493218, -0.11268723],
       ...,
       [-0.52076943, -0.36290363, -0.11268723, ..., -0.21424668,
        -0.20493218, -0.11268723],
       [ 1.0009904 ,  1.22916673, -0.11268723, ..., -0.21424668,
        -0.20493218, -0.11268723],
       [-0.09805837, -0.36290363, -0.11268723, ..., -0.21424668,
        -0.20493218, -0.11268723]])
```

● ×