11_ML - One Hot Encoding and Dummy Variables

- To convert categorical data into numerical data, as ML use mathemetical formulas in its model so all string data should be converted into numerical data
- It is normally used when number of data is lees

In [1]:

```
import pandas as pd
```

In [2]:

```
dataset = pd.read_csv('loan.csv')
```

In [3]:

```
dataset.head(3)
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome
0	LP001002	Male	No	0	Graduate	No	5849
1	LP001003	Male	Yes	1	Graduate	No	4583
2	LP001005	Male	Yes	0	Graduate	Yes	3000

11.1 Find Missing Values and Handle it

In [4]:

```
dataset.isnull().sum()
```

Out[4]:

```
Loan_ID               0
Gender               13
Married               3
Dependents           15
Education             0
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

In [8]:

```
dataset['Gender'].mode()[0]
```

Out[8]:

```
'Male'
```

In [11]:

```
# Gender column contains missing values so we will fill it using mode method
dataset['Gender'].fillna(dataset['Gender'].mode()[0],inplace=True)
```

In [12]:

```
dataset.isnull().sum()
```

Out[12]:

```
Loan_ID               0
Gender                0
Married               3
Dependents           15
Education             0
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

In [13]:

```
# Married column contains missing values so we will fill it using mode method
dataset['Married'].fillna(dataset['Married'].mode()[0],inplace=True)
```

In [14]:

```
dataset.isnull().sum()
```

Out[14]:

```
Loan_ID               0
Gender                0
Married               0
Dependents           15
Education             0
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           22
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

11.2 Use One Hot Code to handle missing values

First separate Gender and Married data to perform encoding

In [16]:

```
en_data = dataset[['Gender', 'Married']]
en_data
```

Out[16]:

	Gender	Married
0	Male	No
1	Male	Yes
2	Male	Yes
3	Male	Yes
4	Male	No
...	...	...
609	Female	No
610	Male	Yes
611	Male	Yes
612	Male	Yes
613	Female	No

614 rows × 2 columns

In [17]:

```
pd.get_dummies(en_data)
```

Out[17]:

	Gender_Female	Gender_Male	Married_No	Married_Yes
0	0	1	1	0
1	0	1	0	1
2	0	1	0	1
3	0	1	0	1
4	0	1	1	0
...	...	...	...	...
609	1	0	1	0
610	0	1	0	1
611	0	1	0	1
612	0	1	0	1
613	1	0	1	0

614 rows × 4 columns

In [18]:

```
pd.get_dummies(en_data).info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Gender_Female  614 non-null    uint8
 1   Gender_Male    614 non-null    uint8
 2   Married_No     614 non-null    uint8
 3   Married_Yes    614 non-null    uint8
dtypes: uint8(4)
memory usage: 2.5 KB
```

In [19]:

```
from sklearn.preprocessing import OneHotEncoder
```

In [20]:

```
# fit_transfrom() converts categorical data into numerical data
ohe = OneHotEncoder()
ohe.fit_transform(en_data)
```

Out[20]:

```
<614x4 sparse matrix of type '<class 'numpy.float64'>'
        with 1228 stored elements in Compressed Sparse Row format>
```

sparse matrix contains data in 0 and 1 form

In [25]:

```
ohe = OneHotEncoder()
ar = ohe.fit_transform(en_data).toarray()
ar
```

Out[25]:

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

In [27]:

```
# Convert the array data into dataframe

pd.DataFrame(ar, columns=['Gender_Female', 'Gender_Male', 'Married_No', 'Married_Ye
```

Out[27]:

	Gender_Female	Gender_Male	Married_No	Married_Yes
0	0.0	1.0	1.0	0.0
1	0.0	1.0	0.0	1.0
2	0.0	1.0	0.0	1.0
3	0.0	1.0	0.0	1.0
4	0.0	1.0	1.0	0.0
...	...	...	...	...
609	1.0	0.0	1.0	0.0
610	0.0	1.0	0.0	1.0
611	0.0	1.0	0.0	1.0
612	0.0	1.0	0.0	1.0
613	1.0	0.0	1.0	0.0

614 rows × 4 columns

You can see out of 2 column 4 column are produced, so to avoid this, we will use 'drop first' to delete first column after encoding i.e. Gender_Female and Married_No, so use it as follows:

In [28]:

```python
ohe = OneHotEncoder(drop='first')
ar = ohe.fit_transform(en_data).toarray()
ar
```

Out[28]:

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

In [29]:

```python
# Convert the array data into dataframe

pd.DataFrame(ar, columns=['Gender_Male','Married_Yes'])
```

Out[29]:

	Gender_Male	Married_Yes
0	1.0	0.0
1	1.0	1.0
2	1.0	1.0
3	1.0	1.0
4	1.0	0.0
...	...	...
609	0.0	0.0
610	1.0	1.0
611	1.0	1.0
612	1.0	1.0
613	0.0	0.0

614 rows × 2 columns

In [ ]:

In [ ]: