In [1]:
```
# Import the packages
# read the data
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\Vi
visa_df=pd.read_csv(path)
visa_df.head(3)
```

Out[1]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training	no_
0	EZYV01	Asia	High School	N	N	
1	EZYV02	Asia	Master's	Y	N	
2	EZYV03	Asia	Bachelor's	N	Y	

- In Machine learning it is very important to convert categorical data to numerical data
- Machine learning models develop by Maths
- Machine learning takes the input in the form of Numbers only
- To convert the we have some encoding techniques
- Label Encoder
  - map
  - np.where
  - using sklearn package: LabelEncoder
- One hot encoder
  - using pandas package: pd.get_dummies

map

- Before applying map method first get the unique labels of the column
- For example case_status is a categorical column
- It has two unique labels are there
  - Denied
  - Certified
- Create a dictionary key as label, value as number
- d={'Certified':0,'Denied':1}
- This dictionary we need to map the case_status column

In [2]:
```
visa_df['case_status'].unique()
```

Out[2]:
```
array(['Denied', 'Certified'], dtype=object)
```

In [3]:
```
d={'Certified':0,'Denied':1}
visa_df['case_status']=visa_df['case_status'].map(d)
```

In [4]:
```
visa_df
```

Out[4]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

In [5]:
```
# when ever you go the error
# run all together
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
visa_df=pd.read_csv(path)
d={'Certified':0,'Denied':1}
visa_df['case_status']=visa_df['case_status'].map(d)
visa_df
```

Out[5]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

In [10]:
```python
d={}
labels=visa_df['continent'].unique()
for i in range(len(labels)):
    d[labels[i]]=i

visa_df['continent']=visa_df['continent'].map(d)
visa_df
```

Out[10]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
0	EZYV01	0	High School	N	
1	EZYV02	0	Master's	Y	
2	EZYV03	0	Bachelor's	N	
3	EZYV04	0	Bachelor's	N	
4	EZYV05	1	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	0	Bachelor's	Y	
25476	EZYV25477	0	High School	Y	
25477	EZYV25478	0	Master's	Y	
25478	EZYV25479	0	Master's	Y	
25479	EZYV25480	0	Bachelor's	Y	

25480 rows × 12 columns

In [14]:
```python
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
visa_df=pd.read_csv(path)
visa_df
```

Out[14]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

```
In [ ]: # case_id  cate
        # labels  25480
        # for i in range(25480):
        # d[case_id[1]]=
```

```
In [25]: # Read the data
         path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
         visa_df=pd.read_csv(path)
         cat_cols=visa_df.select_dtypes(include='object').columns

         d={}
         for j in cat_cols[1:]:  # j= column
             labels=visa_df[j].unique()
             for i in range(len(labels)): # i =number
                 d[labels[i]]=i
             visa_df[j]=visa_df[j].map(d)

         visa_df
```

Out[25]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_traini
0	EZYV01	0	0	0	
1	EZYV02	0	1	1	
2	EZYV03	0	2	0	
3	EZYV04	0	2	0	
4	EZYV05	1	1	1	
...	...	...	...	...	
25475	EZYV25476	0	2	1	
25476	EZYV25477	0	0	1	
25477	EZYV25478	0	1	1	
25478	EZYV25479	0	1	1	
25479	EZYV25480	0	2	1	

25480 rows × 12 columns

```
In [16]: cat_cols=visa_df.select_dtypes(include='object').columns
         cat_cols[1:]
```

```
Out[16]: Index(['continent', 'education_of_employee', 'has_job_experience',
                'requires_job_training', 'region_of_employment', 'unit_of_wage',
                'full_time_position', 'case_status'],
               dtype='object')
```

```
In [ ]: # we always drop the id columns
        # id columns never provide any information
```

LabelEncoder

- LabelEncoder is pacakge avialabel in sklearn
- Sickit learn heart of ML
- Read the package

- Save the package
- Apply fit transform

In [6]:
```
# Read the data again
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\Vi
visa_df=pd.read_csv(path)
```

In [7]:
```
from sklearn.preprocessing import LabelEncoder # read the pacakge
le=LabelEncoder()
visa_df['case_status']=le.fit_transform(visa_df['case_status'])
visa_df
```

Out[7]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_trainii
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

In [8]:
```
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\Vi
visa_df=pd.read_csv(path)
cat_cols= visa_df.select_dtypes(include='object').columns
cat_cols # avoid
```

Out[8]:
```
Index(['case_id', 'continent', 'education_of_employee', 'has_job_experienc
e',
       'requires_job_training', 'region_of_employment', 'unit_of_wage',
       'full_time_position', 'case_status'],
      dtype='object')
```

In [17]:
```
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
visa_df=pd.read_csv(path)
cat_cols= visa_df.select_dtypes(include='object').columns
cat_cols # avoid

from sklearn.preprocessing import LabelEncoder # read the pacakge
le=LabelEncoder()
for i in cat_cols:
    visa_df[i]=le.fit_transform(visa_df[i])
```

In [33]:
```
visa_df
```

Out[33]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_training
0	0	1	2	0	0
1	1	1	3	1	0
2	2	1	0	0	1
3	3	1	0	0	0
4	4	0	3	1	0
...	...	...	...	...	...
25475	17204	1	0	1	1
25476	17205	1	2	1	0
25477	17206	1	3	1	0
25478	17207	1	3	1	1
25479	17209	1	0	1	0

25480 rows × 12 columns

In [35]:
```
visa_df['continent']
```

Out[35]:
```
0        1
1        1
2        1
3        1
4        0
        ..
25475    1
25476    1
25477    1
25478    1
25479    1
Name: continent, Length: 25480, dtype: int32
```

In [37]:
```
visa_df['continent'].value_counts()
```

Out[37]:
```
continent
1    16861
2     3732
3     3292
5      852
0      551
4      192
Name: count, dtype: int64
```

In [39]:
```
col=visa_df['continent']
```

In [41]:
```
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
visa_df=pd.read_csv(path)
cat_cols= visa_df.select_dtypes(include='object').columns
cat_cols # avoid

from sklearn.preprocessing import LabelEncoder # read the pacakge
le=LabelEncoder()
visa_df['continent']=le.fit_transform(visa_df['continent'])
visa_df
```

Out[41]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_trainii
0	EZYV01	1	High School	N	
1	EZYV02	1	Master's	Y	
2	EZYV03	1	Bachelor's	N	
3	EZYV04	1	Bachelor's	N	
4	EZYV05	0	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	1	Bachelor's	Y	
25476	EZYV25477	1	High School	Y	
25477	EZYV25478	1	Master's	Y	
25478	EZYV25479	1	Master's	Y	
25479	EZYV25480	1	Bachelor's	Y	

25480 rows × 12 columns

In [42]:
```
le.inverse_transform(visa_df['continent'])
```

Out[42]:
```
array(['Asia', 'Asia', 'Asia', ..., 'Asia', 'Asia', 'Asia'], dtype=object)
```

```
In [49]: path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
         visa_df=pd.read_csv(path)
         cat_cols= visa_df.select_dtypes(include='object').columns
         cat_cols # avoid

         from sklearn.preprocessing import LabelEncoder # read the pacakge
         le=LabelEncoder()
         for i in cat_cols:
             visa_df[i]=le.fit_transform(visa_df[i])

         # last one executed in visa df : continent: 0,1,2,3,4,5
```

```
In [45]: cat_cols

Out[45]: Index(['case_id', 'continent', 'education_of_employee', 'has_job_experienc
         e',
                'requires_job_training', 'region_of_employment', 'unit_of_wage',
                'full_time_position', 'case_status'],
               dtype='object')
```

```
In [50]: le.inverse_transform(visa_df['continent'])

         ---------------------------------------------------------------------------
         -
         ValueError                                Traceback (most recent call las
         t)
         Cell In[50], line 1
         ----> 1 le.inverse_transform(visa_df['continent'])

         File ~\anaconda3\Lib\site-packages\sklearn\preprocessing\_label.py:160, in
         LabelEncoder.inverse_transform(self, y)
             158 diff = np.setdiff1d(y, np.arange(len(self.classes_)))
             159 if len(diff):
         --> 160     raise ValueError("y contains previously unseen labels: %s" % s
         tr(diff))
             161 y = np.asarray(y)
             162 return self.classes_[y]

         ValueError: y contains previously unseen labels: [2 3 4 5]
```

```
In [48]: for i in range(1,10):
             a=i

         a

Out[48]: 9
```

```
In [52]: path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
         visa_df=pd.read_csv(path)

         from sklearn.preprocessing import LabelEncoder # read the pacakge
         le=LabelEncoder()
         visa_df['continent']=le.fit_transform(visa_df['continent'])
```

In [53]:
```
le.inverse_transform(visa_df['continent'])
```
Out[53]:
```
array(['Asia', 'Asia', 'Asia', ..., 'Asia', 'Asia', 'Asia'], dtype=object)
```

$np.where$

- np.where required 3 arguments
- condition
- True
- False
- It is applicable only for binary labels
- case status has only two labels Certified and Denied
- if case_status==Certified replace that as 0, otherwise 1

In [54]:
```
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
visa_df=pd.read_csv(path)
```

In [56]:
```
con=visa_df['case_status']=='Certified'
visa_df['case_status']=np.where(con,0,1)
visa_df
```

Out[56]:

	case_id	continent	education_of_employee	has_job_experience	requires_job_trainii
0	EZYV01	Asia	High School	N	
1	EZYV02	Asia	Master's	Y	
2	EZYV03	Asia	Bachelor's	N	
3	EZYV04	Asia	Bachelor's	N	
4	EZYV05	Africa	Master's	Y	
...	...	...	...	...	
25475	EZYV25476	Asia	Bachelor's	Y	
25476	EZYV25477	Asia	High School	Y	
25477	EZYV25478	Asia	Master's	Y	
25478	EZYV25479	Asia	Master's	Y	
25479	EZYV25480	Asia	Bachelor's	Y	

25480 rows × 12 columns

One hot encoder

- one hot encoder name says at a time one will On and other will Off
- For example case status has two labels
  - Certified
  - Denied
- When you apply one hot encoding on case status , it creates two more extra columns
  - case_status_Certified
  - case_status_Denied

case_status	case_status_certified	case_status_denied
Certified	1	0
Denied	0	1

Advantages

- When you develop ML model it is very impoartnt the columns should be independent each other
- So here case status creating two extra columns
- Which are independent each other, which means the row values at a time only one column has 1
- Columns are independent each other
- Whcih means 90 degrees phase shift
- Whcih means perpendicular each other
- Whcih mean orthoganal each other

Disadvantage

- The Distavantage is if a column has 100 unique lables , 100 new columns will be created
- The data will become sparse , which means huge
- Columns are more means, Dimesnions are more
- The processing time is more
- The memory consumption is more
- **Curse of Dimensionality**

pd.get_dummies

In [57]:
```
# Read the data
path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\Vi
visa_df=pd.read_csv(path)
```

```
In [62]: path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V
         visa_df=pd.read_csv(path)
         pd.get_dummies(visa_df,
                        columns=['education_of_employee','case_status'],
                       dtype='int')
```

Out[62]:

	case_id	continent	has_job_experience	requires_job_training	no_of_employees	yr
0	EZYV01	Asia	N	N	14513	
1	EZYV02	Asia	Y	N	2412	
2	EZYV03	Asia	N	Y	44444	
3	EZYV04	Asia	N	N	98	
4	EZYV05	Africa	Y	N	1082	
...	...	...	...	...	...	
25475	EZYV25476	Asia	Y	Y	2601	
25476	EZYV25477	Asia	Y	N	3274	
25477	EZYV25478	Asia	Y	N	1121	
25478	EZYV25479	Asia	Y	Y	1918	
25479	EZYV25480	Asia	Y	N	3195	

25480 rows × 16 columns

```
In [63]: path=r"C:\Users\omkar\OneDrive\Documents\Data science\Naresh IT\Datafiles\V:
         visa_df=pd.read_csv(path)
         # make sure drop the id column
         visa_df.drop('case_id',axis=1,inplace=True)
         # When you dont provide the column , it takes all the columns
         pd.get_dummies(visa_df,
                        dtype='int')
```

Out[63]:

	no_of_employees	yr_of_estab	prevailing_wage	continent_Africa	continent_Asia	cont
0	14513	2007	592.2029	0	1	
1	2412	2002	83425.6500	0	1	
2	44444	2008	122996.8600	0	1	
3	98	1897	83434.0300	0	1	
4	1082	2005	149907.3900	1	0	
...	...	...	...	...	...	
25475	2601	2008	77092.5700	0	1	
25476	3274	2006	279174.7900	0	1	
25477	1121	1910	146298.8500	0	1	
25478	1918	1887	86154.7700	0	1	
25479	3195	1960	70876.9100	0	1	

25480 rows × 30 columns

```
In [ ]:
```