Date-18-12-2023

In [1]:

```
# Import packages
# read the data

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

In [6]:

```
file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df
```

Out[6]:

	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 machine learning algotithms will devolpos model by using maths
- maths allow only numbers
- so it is very importent , you need to pass numerical data only
- so we neeed to convert categorical data to numerical data
- for that we have encoding methods
- Encoding
    - Label Encoder
        - map method
        - np.where
        - Lable encoder packages from sklearn
    - one head encoder
        - pd.get_dummies()

Map Method

$method - 1$

- Read any categorical column: case_ststus
- check how many unique labels are there
- Create a dictionary with those unique lables as keys by providing a number as values

In [ ]:

```
############# read the data ################################

file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df

###############################Map#################################

visa_df['case_status'].unique() # 2
dict1={'Certified':0,'Denied':1}
visa_df['case_status'].map(dict1)

# do you want overwrite existed column
# do you want create a new column
```

In [8]:

```
visa_df['case_status'].unique()
dict1={'Certified':0,'Denied':1}
```

In [12]:

```
visa_df['case_status'].map(dict1)
```

Out[12]:

```
0        1
1        0
2        1
3        1
4        0
        ..
25475    0
25476    0
25477    0
25478    0
25479    0
Name: case_status, Length: 25480, dtype: int64
```

create a new column

In [13]:

```
############# read the data ################################

file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df

####################### map #######################################

visa_df['case_status'].unique()
dict1={'Certified':0,'Denied':1}
visa_df['case_status_num']=visa_df['case_status'].map(dict1)
```

In [14]:
```
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 × 13 columns

Drop the column

In [16]:
```
visa_df.drop('case_status_num',axis=1,inplace=True) # drop 'case_data_num'
```

In [17]:
```
visa_df
```

Out[17]:

	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

Overwrite the same column(preferable)

In [18]:

```
file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df
```

Out[18]:

	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 [22]:

```
############################### map ###############################3

visa_df['case_status'].unique()
dict1={'Certified':0,'Denied':1}
visa_df['case_status']=visa_df['case_status'].map(dict1)



# in the maP method inplace = True is not there
```

In [23]:

```
visa_df
```

Out[23]:

	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 [26]:

```
visa_df['continent'].unique()
```

Out[26]:

```
array(['Asia', 'Africa', 'North America', 'Europe', 'South America',
       'Oceania'], dtype=object)
```

In [30]:

```
visa_df['continent'].unique()
{'Certified':0,'Denied':1}
{'Asia':0,'Africa':1,'North America':2,'Europe':3,'south America':4,'Oceani
```

Out[30]:

```
{'Asia': 0,
 'Africa': 1,
 'North America': 2,
 'Europe': 3,
 'south America': 4,
 'Oceania': 5}
```

In [32]:

```
num=len(visa_df['continent'].unique())
dict1={}
for i in range(num):
    print(visa_df['continent'].unique()[i],i)
```

```
Asia 0
Africa 1
North America 2
Europe 3
South America 4
Oceania 5
```

In [33]:

```
# dict method

labels=visa_df['continent'].unique()
for i in range(num):
    dict1[labels[i]]=i
dict1
```

Out[33]:

```
{'Asia': 0,
 'Africa': 1,
 'North America': 2,
 'Europe': 3,
 'South America': 4,
 'Oceania': 5}
```

In [34]:

```
# comprihention

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

Out[34]:

```
{'Asia': 0,
 'Africa': 1,
 'North America': 2,
 'Europe': 3,
 'South America': 4,
 'Oceania': 5}
```

$$Method - 2$$

np.where()

In [35]:

```
################################# Read the data ##########################

file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df
```

Out[35]:

	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

- np.where is aplicable for binary condition
- which means is aplicabel only for two lables
- np.where (,,)
- for example case_status has two labels
- condition: == 'Certified'
- True value:replace all certified with 0
- False value: replace all denied values with 1

In [36]:

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

In [38]:

```
visa_df.head()
```

Out[38]:

	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	
3	EZYV04	Asia	Bachelor's	N	N	
4	EZYV05	Africa	Master's	Y	N	

Label Encoder

- labelencoder is a method from sklearn
- Under sklearn we have sub modules
- One of the submodule: preprocessing
- Any sklearn packages we have only 3 steps
- step-1: read the packages
- step-2: save the packages
- step-3: apply fit transform

In [9]:
```
################################ Read the data ##########################

file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
print(visa_df[['continent','case_status']].head(10))

################################ LableEncoder ########################

# step-1

from sklearn.preprocessing import LabelEncoder

# step-2

le=LabelEncoder()

# step-3

visa_df['case_status']=le.fit_transform(visa_df['case_status'])
visa_df['continent']=le.fit_transform(visa_df['continent'])
print(visa_df[['continent','case_status']].head(10))
```

```
       continent case_status
0           Asia      Denied
1           Asia   Certified
2           Asia      Denied
3           Asia      Denied
4         Africa   Certified
5           Asia   Certified
6           Asia   Certified
7  North America      Denied
8           Asia   Certified
9         Europe   Certified
   continent  case_status
0          1            1
1          1            0
2          1            1
3          1            1
4          0            0
5          1            0
6          1            0
7          3            1
8          1            0
9          2            0
```

In [10]:

```
print(visa_df['continent'][:5])
le.inverse_transform(visa_df['continent'])
```

```
0    1
1    1
2    1
3    1
4    0
Name: continent, dtype: int32
```

Out[10]:

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

fit-transform:

- fit and transform two diffrent definations
- age= 1, 2, 3 , 4 , 5
- new age: by adding each observation with mean value: x+mean
- mean= 1+2+3+4+5/5 = 3 ===========> fit
- new age: =========================> Transform 1+3 2+3 3+3 4+3 5+3

In [11]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
le.fit_transform(visa_df['case_status']=le.fit_transform(visa_df['case_stat
visa_df
```

```
  Cell In[11], line 3
    le.fit_transform(visa_df['case_status']=le.fit_transform(visa_df['case
_status']))
                     ^
SyntaxError: expression cannot contain assignment, perhaps you meant "=="?
```

In [2]:

```
# Import packages
# read the data

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

In [3]:

```
file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df
```

Out[3]:

	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

one hot encoder

In [ ]:

```
- one hot encoder means at a time only one will be ON(1/True), others are O
- suppose case status has two unique

|case_status|certified|denied|
|-----------|---------|------|
|certified|1|0|
|denied|0|1|


- one hot encoder new column are othrogonal each other

- orthogonality means 90 degree phase shift


**Draw back**


- Assume that you have 100

- this is **curse of dimentionality**
```

pd.get_dummies method

In [4]:
```
file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df

pd.get_dummies(visa_df,columns=['case_status'])
```

Out[4]:

	no_of_employees	yr_of_estab	prevailing_wage	case_id_EZYV01	case_id_EZYV02	ca
0	14513	2007	592.2029	True	False	
1	2412	2002	83425.6500	False	True	
2	44444	2008	122996.8600	False	False	
3	98	1897	83434.0300	False	False	
4	1082	2005	149907.3900	False	False	
...	...	...	...	...	...	
25475	2601	2008	77092.5700	False	False	
25476	3274	2006	279174.7900	False	False	
25477	1121	1910	146298.8500	False	False	
25478	1918	1887	86154.7700	False	False	
25479	3195	1960	70876.9100	False	False	

25480 rows × 25510 columns

In [5]:
```
file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df

pd.get_dummies(visa_df,columns=['case_status'])
```

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 × 13 columns

In [6]:

```
file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df

pd.get_dummies(visa_df,columns=['continent'])
```

Out[6]:

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

25480 rows × 17 columns

In [7]:

```
file_path="C:\\Users\\kurre\\OneDrive\\Documents\\Naresh IT\\datafiles\\Vis
visa_df=pd.read_csv(file_path)
visa_df

pd.get_dummies(visa_df,columns=['case_status'],dtype='int')
```

Out[7]:

	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 × 13 columns

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: