In [68]:

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

In [69]:

```
df = pd.read_csv("Lab1.csv")
df.head()
```

Out[69]:

	country	age	salary	purchased
0	France	NaN	7200	no
1	Spain	27.0	4800	yes
2	Germany	30.0	5400	yes
3	UK	49.0	98000	no

In [70]:

```
df.tail()
```

Out[70]:

	country	age	salary	purchased
0	France	NaN	7200	no
1	Spain	27.0	4800	yes
2	Germany	30.0	5400	yes
3	UK	49.0	98000	no

In [71]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   country    4 non-null      object 
 1   age        3 non-null      float64
 2   salary     4 non-null      int64  
 3   purchased  4 non-null      object 
dtypes: float64(1), int64(1), object(2)
memory usage: 256.0+ bytes
```

In [72]:

```
df.describe()
```

Out[72]:

	age	salary
count	3.000000	4.000000
mean	35.333333	28850.000000
std	11.930353	46111.278447
min	27.000000	4800.000000
25%	28.500000	5250.000000
50%	30.000000	6300.000000
75%	39.500000	29900.000000
max	49.000000	98000.000000

In [73]:

```
df.isnull().sum()
```

Out[73]:

```
country      0
age          1
salary       0
purchased    0
dtype: int64
```

In [74]:

```
df['age'].fillna(df['age'].mean(), inplace = True)
df['salary'].fillna(df['salary'].mean(), inplace=True)
```

In [75]:

```
df.isnull().sum()
```

Out[75]:

```
country      0
age          0
salary       0
purchased    0
dtype: int64
```

In [76]:

```
from sklearn.impute import SimpleImputer
x = df.iloc[:,:-1].values
x
```

Out[76]:

```
array([['France', 35.333333333333336, 7200],
       ['Spain', 27.0, 4800],
       ['Germany', 30.0, 5400],
       ['UK', 49.0, 98000]], dtype=object)
```

In [77]:

```
y = df.iloc[:,3: ].values
y
```

Out[77]:

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

In [78]:

```
imp = SimpleImputer(missing_values =np.nan, strategy = "mean")
x[:, 1:3] = imp.fit_transform(x[:, 1:3])
x
```

Out[78]:

```
array([['France', 35.333333333333336, 7200.0],
       ['Spain', 27.0, 4800.0],
       ['Germany', 30.0, 5400.0],
       ['UK', 49.0, 98000.0]], dtype=object)
```

In [79]:

```
from sklearn.preprocessing import LabelEncoder
```

In [80]:

```
le = LabelEncoder()
h = le.fit_transform(x[:,0])
h
```

Out[80]:

```
array([0, 2, 1, 3])
```

In [81]:

```
y = le.fit_transform(y)
y
```

```
C:\Users\25LAB-2BCA\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[81]:

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

In [82]:

```
from sklearn.utils import column_or_1d
y = column_or_1d(y, warn = True)
y
```

Out[82]:

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

In [83]:

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
```

In [84]:

```
transform = ColumnTransformer([('norm1', OneHotEncoder(), [0])], remainder ="passthrough")
x = transform.fit_transform(x)
x
```

Out[84]:

```
array([[1.0, 0.0, 0.0, 0.0, 35.333333333333336, 7200.0],
       [0.0, 0.0, 1.0, 0.0, 27.0, 4800.0],
       [0.0, 1.0, 0.0, 0.0, 30.0, 5400.0],
       [0.0, 0.0, 0.0, 1.0, 49.0, 98000.0]], dtype=object)
```

In [85]:

```
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)
```

In [86]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
```

In [87]:

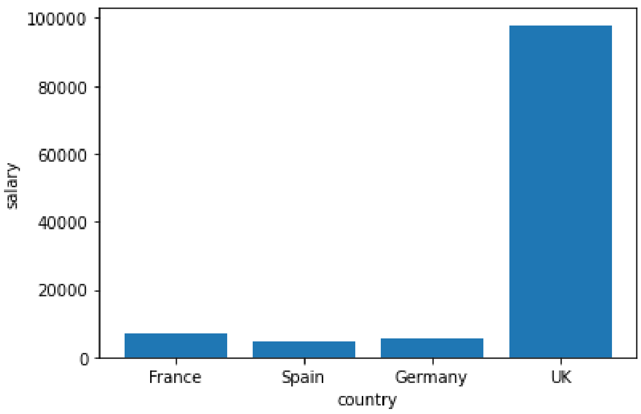
```
x_train[:,4:6] = sc.fit_transform(x_train[:, 4:6])
x_train
```

Out[87]:

```
array([[0.0, 0.0, 0.0, 1.0, 1.3109359202840398, 1.4138527953056175],
       [0.0, 0.0, 1.0, 0.0, -1.1149081191200718, -0.7345887349522664],
       [1.0, 0.0, 0.0, 0.0, -0.1960278011639687, -0.679264060353351]],
      dtype=object)
```

In [91]:

```
plt.bar(df['country '],df['salary'])
plt.xlabel('country')
plt.ylabel('salary')
plt.show()
```

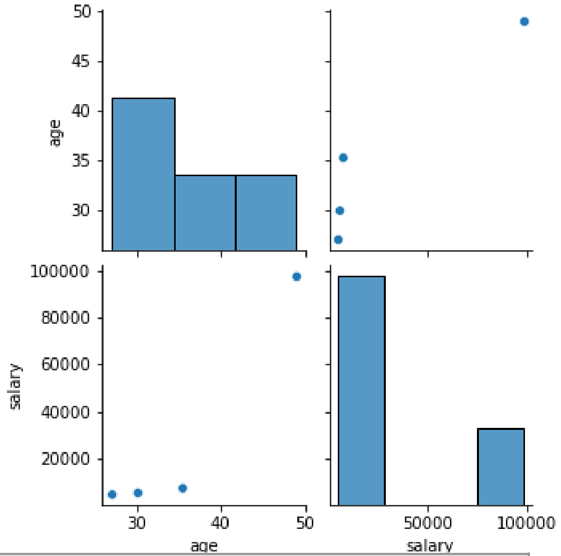


In [92]:

```
import seaborn as sns
sns.pairplot(df)
```

Out[92]:

```
<seaborn.axisgrid.PairGrid at 0x18b7dac21f0>
```



Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js