In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
dataset=pd.read_csv(r"D:\ML_Course\Works_on_python\Decision tree & Random Forest Calssification\bikebuyer1.csv")
```

In [3]:
```
dataset.head()
```

Out[3]:

| | ID | Marital Status | Gender | Yearly Income | Children | Education | Occupation | Home Owner | Cars | Commute Distance | Region | Age | Bike Buyer |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 22711.0 | Single | Male | 30000 | 0.0 | Partial College | Clerical | No | 1 | 1.0 | Europe | 33 | Yes |
| **1** | 13555.0 | Married | Female | 40000 | 0.0 | Graduate Degree | Clerical | Yes | 0 | 1.0 | Europe | 37 | Yes |
| **2** | NaN | Married | Male | 160000 | 5.0 | Partial College | Professional | No | 3 | 2.0 | Europe | 55 | No |
| **3** | 2.0 | Single | Male | 160000 | 0.0 | Graduate Degree | Management | Yes | 2 | 5.0 | Pacific | 47 | No |
| **4** | 25410.0 | NaN | Female | 70000 | 2.0 | Bachelors | Skilled Manual | No | 1 | 1.0 | North America | 38 | Yes |

In [4]:
```
dataset.isnull().any()
```

Out[4]:
```
ID                   True
Marital Status       True
Gender               True
Yearly Income       False
Children             True
Education           False
Occupation          False
Home Owner          False
Cars                False
Commute Distance     True
Region              False
Age                 False
Bike Buyer          False
dtype: bool
```

In [5]:
```
dataset["Marital Status"].fillna(dataset["Marital Status"].mode()[0],inplace=True)
dataset["Gender"].fillna(dataset["Gender"].mode()[0],inplace=True)
dataset["Children"].fillna(dataset["Children"].median(),inplace=True)
dataset["Commute Distance"].fillna(dataset["Commute Distance"].mode()[0],inplace=True)
```

In [6]:
```
dataset["Marital Status"].unique()
```

Out[6]:
```
array(['Single', 'Married'], dtype=object)
```

In [7]:
```
dataset["Gender"].unique()
```

Out[7]:
```
array(['Male', 'Female'], dtype=object)
```

In [8]:
```
dataset["Children"].unique()
```

Out[8]:
```
array([0., 5., 2., 1., 4., 3.])
```

In [9]:
```
dataset["Education"].unique()
```

Out[9]:
```
array(['Partial College', 'Graduate Degree', 'Bachelors', 'High School',
       'Partial High School'], dtype=object)
```

In [10]:
```
dataset["Occupation"].unique()
```

Out[10]:
```
array(['Clerical', 'Professional', 'Management', 'Skilled Manual',
       'Manual'], dtype=object)
```

In [11]:
```
dataset["Home Owner"].unique()
```

Out[11]:
```
array(['No', 'Yes'], dtype=object)
```

In [12]:
```
dataset["Region"].unique()
```

Out[12]:
```
array(['Europe', 'Pacific', 'North America'], dtype=object)
```

In [13]:

```
dataset.isnull().any()
```

Out[13]:

```
ID                   True
Marital Status      False
Gender              False
Yearly Income       False
Children            False
Education           False
Occupation          False
Home Owner          False
Cars                False
Commute Distance    False
Region              False
Age                 False
Bike Buyer          False
dtype: bool
```

In [ ]:

```
import seaborn as sns
sns.pairplot(dataset,hue="Bike Buyer")
```

In [ ]:

```
sns.catplot(x="Bike Buyer",y="Yearly Income",data=dataset)
```

In [ ]:

```
sns.catplot(x="Marital Status",y="Age",data=dataset)
```

In [14]:

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
dataset["Marital Status"]=le.fit_transform(dataset["Marital Status"])
dataset["Region"]=le.fit_transform(dataset["Region"])
dataset["Home Owner"]=le.fit_transform(dataset["Home Owner"])
dataset["Occupation"]=le.fit_transform(dataset["Occupation"])
dataset["Education"]=le.fit_transform(dataset["Education"])
dataset["Gender"]=le.fit_transform(dataset["Gender"])
dataset["Bike Buyer"]=le.fit_transform(dataset["Bike Buyer"])
```

In [15]:

```
dataset.isnull().any()
```

Out[15]:

```
ID                   True
Marital Status      False
Gender              False
Yearly Income       False
Children            False
Education           False
Occupation          False
Home Owner          False
Cars                False
Commute Distance    False
Region              False
Age                 False
Bike Buyer          False
dtype: bool
```

In [16]:
```
dataset.corr()
```

Out[16]:

| | ID | Marital Status | Gender | Yearly Income | Children | Education | Occupation | Home Owner | Cars | Commute Distance | Region | Age | Bike Buyer |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **ID** | 1.000000 | 0.050063 | 0.004949 | -0.006802 | 0.011751 | 0.058976 | 0.005225 | -0.076878 | 0.068315 | 0.012638 | 0.000143 | 0.021135 | 0.210813 |
| **Marital Status** | 0.050063 | 1.000000 | -0.060753 | -0.159823 | -0.107973 | 0.123656 | -0.032806 | -0.254684 | -0.049793 | 0.003927 | -0.117683 | -0.283900 | 0.073119 |
| **Gender** | 0.004949 | -0.060753 | 1.000000 | -0.002706 | -0.005853 | -0.007174 | 0.005131 | -0.007835 | -0.001182 | -0.016584 | 0.014955 | -0.002175 | -0.024682 |
| **Yearly Income** | -0.006802 | -0.159823 | -0.002706 | 1.000000 | 0.474231 | -0.249386 | 0.067132 | 0.089856 | 0.472089 | 0.008711 | 0.256088 | 0.195351 | 0.018456 |
| **Children** | 0.011751 | -0.107973 | -0.005853 | 0.474231 | 1.000000 | -0.033362 | -0.028387 | 0.141689 | 0.448015 | 0.026407 | 0.056532 | -0.000245 | -0.053034 |
| **Education** | 0.058976 | 0.123656 | -0.007174 | -0.249386 | -0.033362 | 1.000000 | 0.018243 | -0.138411 | 0.103111 | 0.017554 | -0.210062 | -0.123425 | -0.049899 |
| **Occupation** | 0.005225 | -0.032806 | 0.005131 | 0.067132 | -0.028387 | 0.018243 | 1.000000 | -0.013761 | -0.032003 | 0.010645 | 0.215309 | -0.212152 | -0.007937 |
| **Home Owner** | -0.076878 | -0.254684 | -0.007835 | 0.089856 | 0.141689 | -0.138411 | -0.013761 | 1.000000 | -0.049189 | -0.008284 | 0.013076 | 0.148014 | -0.001434 |
| **Cars** | 0.068315 | -0.049793 | -0.001182 | 0.472089 | 0.448015 | 0.103111 | -0.032003 | -0.049189 | 1.000000 | 0.019413 | 0.202122 | 0.141093 | -0.095730 |
| **Commute Distance** | 0.012638 | 0.003927 | -0.016584 | 0.008711 | 0.026407 | 0.017554 | 0.010645 | -0.008284 | 0.019413 | 1.000000 | -0.007597 | -0.025420 | -0.004575 |
| **Region** | 0.000143 | -0.117683 | 0.014955 | 0.256088 | 0.056532 | -0.210062 | 0.215309 | 0.013076 | 0.202122 | -0.007597 | 1.000000 | -0.004970 | 0.054137 |
| **Age** | 0.021135 | -0.283900 | -0.002175 | 0.195351 | -0.000245 | -0.123425 | -0.212152 | 0.148014 | 0.141093 | -0.025420 | -0.004970 | 1.000000 | -0.055391 |
| **Bike Buyer** | 0.210813 | 0.073119 | -0.024682 | 0.018456 | -0.053034 | -0.049899 | -0.007937 | -0.001434 | -0.095730 | -0.004575 | 0.054137 | -0.055391 | 1.000000 |

use catplot and check x="" which one is need to use with hue bike buyer and y=yealy income

In [ ]:
```
sns.catplot(x="Occupation",y="Yearly Income",hue="Bike Buyer",data=dataset)
```

In [ ]:
```
sns.countplot(x="Bike Buyer",data=dataset)
```

In [17]:
```
dataset.head(1)
```

Out[17]:

| | ID | Marital Status | Gender | Yearly Income | Children | Education | Occupation | Home Owner | Cars | Commute Distance | Region | Age | Bike Buyer |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 22711.0 | 1 | 1 | 30000 | 0.0 | 3 | 0 | 0 | 1 | 1.0 | 0 | 33 | 1 |

In [19]:

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-19-9f2b259887ef> in <module>
----> 1 x.shape

NameError: name 'x' is not defined
```

In [21]:

```
x=dataset.iloc[:,[1,2,3,4,8,9,11]].values
y=dataset.iloc[:,12:13].values
```

In [22]:

```
x.shape
```

Out[22]:

```
(6997, 7)
```

In [18]:

```
from sklearn.preprocessing import OneHotEncoder
one=OneHotEncoder()
z=one.fit_transform[:,0:1].toarray()
x=one.fit_transform[:,1:2]
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-18-89061fe004ae> in <module>
      1 from sklearn.preprocessing import OneHotEncoder
      2 one=OneHotEncoder()
----> 3 z=one.fit_transform[:,0:1].toarray()

TypeError: 'method' object is not subscriptable
```

In [ ]: