In [72]:
```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
import tensorflow as tf
import seaborn as sn
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report
```

In [27]:
```
df = pd.read_csv("D:\Backend\data.csv")
```

In [29]:
```
clf = DecisionTreeClassifier()
```

In [30]:
```
df.Income.replace(["Low","Med","High"],[0,1,2],inplace=True)
```

In [49]:
```
df.Marital_status.replace(["Single","Married"],[0,1],inplace=True)
df.Gender.replace(["M","F"],[0,1],inplace=True)
df.Age.replace(["<21","21-35",">35"],[0,1,2],inplace=True)
df.Buys.replace(["Y","N"],[0,1],inplace=True)
```

In [50]:
```
df.Buys
```

Out[50]:
```
0     1
1     1
2     0
3     0
4     0
5     1
6     0
7     1
8     0
9     0
10    0
11    0
12    0
13    1
Name: Buys, dtype: int64
```

In [32]:
```
features = df[["Age","Income","Gender","Marital_status"]]
```

In [33]:
```
features
```

Out[33]:

	Age	Income	Gender	Marital_status
0	0	2	0	0
1	0	2	0	1
2	1	2	0	0
3	2	1	0	0
4	2	0	1	0
5	2	0	1	1
6	1	0	1	1
7	0	1	0	0
8	0	0	1	1
9	2	1	1	0
10	0	1	1	1
11	1	1	0	1
12	1	2	1	0
13	2	1	0	1

In [51]:
```
labels = df[["Buys"]]
```

In [52]:
```
df
```

Out[52]:

	Age	Income	Gender	Marital_status	Buys
0	0	2	0	0	1
1	0	2	0	1	1
2	1	2	0	0	0
3	2	1	0	0	0
4	2	0	1	0	0
5	2	0	1	1	1
6	1	0	1	1	0
7	0	1	0	0	1
8	0	0	1	1	0
9	2	1	1	0	0
10	0	1	1	1	0
11	1	1	0	1	0
12	1	2	1	0	0
13	2	1	0	1	1

In [53]:
```
clf.fit(features,labels)
```

Out[53]:
```
DecisionTreeClassifier()
```

In [54]:
```
x=df[['Age','Income' ,'Gender','Marital_status']]
```

In [55]:
```
x
```

Out[55]:

	Age	Income	Gender	Marital_status
0	0	2	0	0
1	0	2	0	1
2	1	2	0	0
3	2	1	0	0
4	2	0	1	0
5	2	0	1	1
6	1	0	1	1
7	0	1	0	0
8	0	0	1	1
9	2	1	1	0
10	0	1	1	1
11	1	1	0	1
12	1	2	1	0
13	2	1	0	1

In [56]:
```
y=clf.predict(x)
```

In [57]:
```
y
```

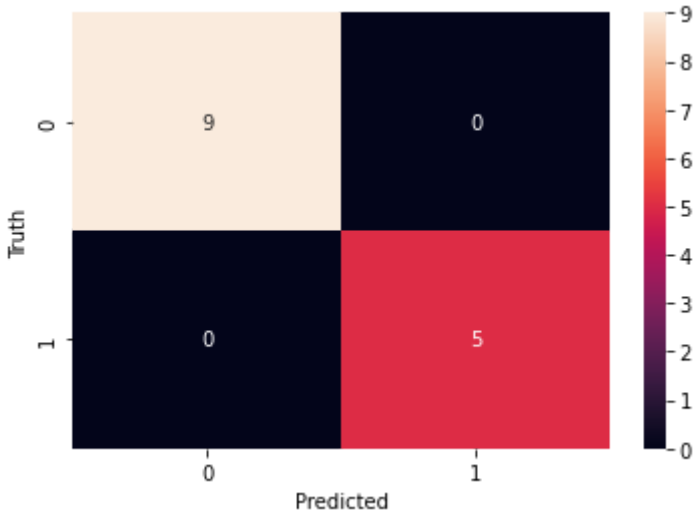
Out[57]:
```
array([1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1], dtype=int64)
```

In [58]:
```
cm=tf.math.confusion_matrix(labels=df.Buys, predictions= y)
cm
```

Out[58]:
```
<tf.Tensor: shape=(2, 2), dtype=int32, numpy=
array([[9, 0],
       [0, 5]])>
```

In [71]:
```
sn.heatmap(cm,annot=True)
plt.xlabel("Predicted")
plt.ylabel("Truth")
```

Out[71]:
```
Text(33.0, 0.5, 'Truth')
```



In [73]:
```
print(classification_report(y_true=df.Buys , y_pred=y))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         9
           1       1.00      1.00      1.00         5

    accuracy                           1.00        14
   macro avg       1.00      1.00      1.00        14
weighted avg       1.00      1.00      1.00        14
```