

In [17]:

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

In [2]:

```
zoo_data=pd.read_csv('Zoo.csv')
zoo_data.head()
```

Out[2]:

	animal name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes
0	aardvark	1	0	0	1	0	0	1	1	1	
1	antelope	1	0	0	1	0	0	0	1	1	
2	bass	0	0	1	0	0	1	1	1	1	
3	bear	1	0	0	1	0	0	1	1	1	
4	boar	1	0	0	1	0	0	1	1	1	

Initial investigation

In [3]:

```
zoo_data.shape
```

Out[3]:

(101, 18)

In [4]:

```
zoo_data.isnull().sum()
```

Out[4]:

```
animal name    0
hair           0
feathers       0
eggs           0
milk           0
airborne       0
aquatic        0
predator       0
toothed        0
backbone       0
breathes       0
venomous       0
fins           0
legs           0
tail           0
domestic       0
catsize        0
type           0
dtype: int64
```

In [5]:

```
zoo_data.dtypes
```

Out[5]:

```
animal name    object
hair           int64
feathers       int64
eggs           int64
milk           int64
airborne       int64
aquatic        int64
predator       int64
toothed        int64
backbone       int64
breathes       int64
venomous       int64
fins           int64
legs           int64
tail           int64
domestic       int64
catsize        int64
type           int64
dtype: object
```

Number of features and records in the given data set is 11 and 400 respectively

There is no null values in the data set

In [6]:

```
zoo_data['animal name'].nunique()
```

Out[6]:

100

In [7]:

```
zoo_data['type'].unique()
```

Out[7]:

```
array([1, 4, 2, 7, 6, 5, 3], dtype=int64)
```

Model building

In [8]:

```
x=zoo_data.iloc[:,1:-1]  
y=zoo_data.iloc[:, -1:]
```

In [9]:

```
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)
```

Model training

In [10]:

```
import warnings  
warnings.filterwarnings('ignore')
```

In [11]:

```
from sklearn.neighbors import KNeighborsClassifier  
knn_model=KNeighborsClassifier(n_neighbors=11)
```

In [12]:

```
knn_model.fit(x_train,y_train)
```

Out[12]:

```
KNeighborsClassifier(n_neighbors=11)
```

Model testing

In [13]:

```
y_pred=knn_model.predict(x_test)
```

Model evalution

In [14]:

```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

In [15]:

```
print(accuracy_score(y_test, y_pred))
```

```
0.6190476190476191
```

In [16]:

```
print(confusion_matrix(y_test, y_pred))
```

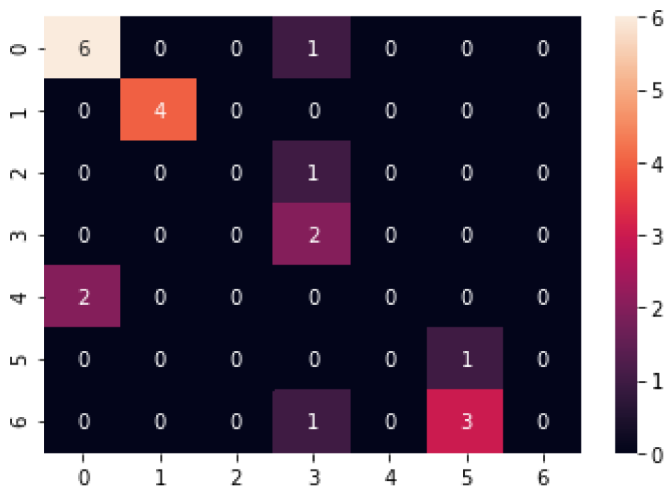
```
[[6 0 0 1 0 0 0]
 [0 4 0 0 0 0 0]
 [0 0 0 1 0 0 0]
 [0 0 0 2 0 0 0]
 [2 0 0 0 0 0 0]
 [0 0 0 0 0 1 0]
 [0 0 0 1 0 3 0]]
```

In [18]:

```
confusion_matrix_test = confusion_matrix(y_test, y_pred)
sns.heatmap(confusion_matrix_test, annot=True)
```

Out[18]:

<AxesSubplot:>



In [19]:

```
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
1	0.75	0.86	0.80	7
2	1.00	1.00	1.00	4
3	0.00	0.00	0.00	1
4	0.40	1.00	0.57	2
5	0.00	0.00	0.00	2
6	0.25	1.00	0.40	1
7	0.00	0.00	0.00	4
accuracy			0.62	21
macro avg	0.34	0.55	0.40	21
weighted avg	0.49	0.62	0.53	21

Tweaking K value

In [20]:

```
for i in range(1,50,2):  
    knn_model=KNeighborsClassifier(n_neighbors=i).fit(x_train,y_train)  
    y_pred=knn_model.predict(x_test)  
    print('K value is: ',i)  
    print(accuracy_score(y_test,y_pred))
```

```
K value is:  1  
1.0  
K value is:  3  
0.8095238095238095  
K value is:  5  
0.7619047619047619  
K value is:  7  
0.6666666666666666  
K value is:  9  
0.6190476190476191  
K value is: 11  
0.6190476190476191  
K value is: 13  
0.6190476190476191  
K value is: 15  
0.6190476190476191  
K value is: 17  
0.6190476190476191  
K value is: 19  
0.6190476190476191  
K value is: 21  
0.5714285714285714  
K value is: 23  
0.5714285714285714  
K value is: 25  
0.5714285714285714  
K value is: 27  
0.5714285714285714  
K value is: 29  
0.5714285714285714  
K value is: 31  
0.47619047619047616  
K value is: 33  
0.47619047619047616  
K value is: 35  
0.47619047619047616  
K value is: 37  
0.47619047619047616  
K value is: 39  
0.47619047619047616  
K value is: 41  
0.42857142857142855  
K value is: 43  
0.42857142857142855  
K value is: 45  
0.42857142857142855  
K value is: 47  
0.42857142857142855  
K value is: 49  
0.42857142857142855
```

By comparing the above obtained accuracy, the model behaves good when the value of K is 3 or 5

In [21]:

```
knn_model=KNeighborsClassifier(n_neighbors=i).fit(x_train,y_train)
y_pred=knn_model.predict(x_test)
print('K value is: ',3)
print(accuracy_score(y_test,y_pred))
```

K value is: 3
0.42857142857142855

In [22]:

```
knn_model=KNeighborsClassifier(n_neighbors=i).fit(x_train,y_train)
y_pred=knn_model.predict(x_test)
print('K value is: ',5)
print(accuracy_score(y_test,y_pred))
```

K value is: 5
0.42857142857142855