

In [1]:

```

1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 from IPython import get_ipython
6 import warnings
7 warnings.filterwarnings("ignore")

```

In [2]:

```

1 data1 = pd.read_csv('zoo2.csv')
2 data2 = pd.read_csv('zoo3.csv')

```

In [3]:

```
1 data1.head()
```

Out[3]:

	animal_name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	bre
0	turtle	0	0	1	0	0	1	0	0	1	
1	chameleon	0	0	1	0	0	0	0	1	1	
2	iguana	0	0	1	0	0	0	1	1	1	
3	lizard	0	0	1	0	0	0	1	1	1	
4	gecko	0	0	1	0	0	0	0	1	1	

In [4]:

```
1 data1.tail()
```

Out[4]:

	animal_name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	bre
38	spider	0	0	1	0	0	0	1	1	0	
39	snail	0	0	1	0	0	0	0	0	0	
40	silkworm	0	0	1	0	0	0	0	0	0	
41	jellyfish	0	0	1	0	0	1	0	0	0	
42	squid	0	0	1	0	0	1	0	0	0	

In [5]:



```
1 data2.head()
```

Out[5]:

	animal_name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	bre
0	anta	1	0	0	1	0	0	0	1	1	
1	ariranha	1	0	0	1	0	1	1	1	1	
2	boto-cor-de-rosa	0	0	0	1	0	1	1	1	1	
3	bugio	1	0	0	1	0	0	0	1	1	
4	cachorro-vinagre	1	0	0	1	0	0	1	1	1	

In [6]:



```
1 data2.tail()
```

Out[6]:

	animal_name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	bre
65	vespa	0	0	1	0	1	0	1	0	0	
66	bicho-pau	0	0	1	0	0	0	0	0	0	
67	caracol-da-mata-atlantica	0	0	1	0	0	0	0	0	0	
68	caranguejeira	1	0	1	0	0	0	1	0	0	
69	sauva-limao	1	0	1	0	0	0	1	0	0	

In [7]:



```
1 data1.shape
```

Out[7]:

(43, 18)

In [8]:



```
1 data2.shape
```

Out[8]:

(70, 18)

In [9]:



```
1 data1.columns
```

Out[9]:

```
Index(['animal_name', 'hair', 'feathers', 'eggs', 'milk', 'airborne',  
      'aquatic', 'predator', 'toothed', 'backbone', 'breathes', 'venomou  
s',  
      'fins', 'legs', 'tail', 'domestic', 'catsize', 'class_type'],  
      dtype='object')
```

In [10]:



```
1 data2.columns
```

Out[10]:

```
Index(['animal_name', 'hair', 'feathers', 'eggs', 'milk', 'airborne',  
      'aquatic', 'predator', 'toothed', 'backbone', 'breathes', 'venomou  
s',  
      'fins', 'legs', 'tail', 'domestic', 'catsize', 'class_type'],  
      dtype='object')
```

In [11]:



```
1 data1.duplicated().sum()
```

Out[11]:

```
0
```

In [12]:



```
1 data2.duplicated().sum()
```

Out[12]:

```
0
```

In [13]:



```
1 data1.isnull().sum()
```

Out[13]:

```
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
class_type     0
dtype: int64
```

In [14]:



```
1 data2.isnull().sum()
```

Out[14]:

```
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
class_type     0
dtype: int64
```

In [15]:



```
1 data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 43 entries, 0 to 42
Data columns (total 18 columns):
#   Column          Non-Null Count  Dtype
---  -
0   animal_name     43 non-null    object
1   hair            43 non-null    int64
2   feathers        43 non-null    int64
3   eggs            43 non-null    int64
4   milk            43 non-null    int64
5   airborne        43 non-null    int64
6   aquatic         43 non-null    int64
7   predator        43 non-null    int64
8   toothed         43 non-null    int64
9   backbone        43 non-null    int64
10  breathes        43 non-null    int64
11  venomous        43 non-null    int64
12  fins            43 non-null    int64
13  legs            43 non-null    int64
14  tail            43 non-null    int64
15  domestic        43 non-null    int64
16  catsize         43 non-null    int64
17  class_type      43 non-null    int64
dtypes: int64(17), object(1)
memory usage: 6.2+ KB
```

In [16]:



```
1 data1.describe()
```

Out[16]:

	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone
count	43.000000	43.0	43.0	43.0	43.000000	43.000000	43.000000	43.000000	43.000000
mean	0.023256	0.0	1.0	0.0	0.162791	0.465116	0.302326	0.441860	0.581395
std	0.152499	0.0	0.0	0.0	0.373544	0.504685	0.464701	0.502486	0.499169
min	0.000000	0.0	1.0	0.0	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.0	1.0	0.0	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.0	1.0	0.0	0.000000	0.000000	0.000000	0.000000	1.000000
75%	0.000000	0.0	1.0	0.0	0.000000	1.000000	1.000000	1.000000	1.000000
max	1.000000	0.0	1.0	0.0	1.000000	1.000000	1.000000	1.000000	1.000000

In [17]:

```
1 data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 70 entries, 0 to 69
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  -
0   animal_name           70 non-null    object
1   hair                  70 non-null    int64
2   feathers              70 non-null    int64
3   eggs                 70 non-null    int64
4   milk                 70 non-null    int64
5   airborne             70 non-null    int64
6   aquatic              70 non-null    int64
7   predator             70 non-null    int64
8   toothed              70 non-null    int64
9   backbone             70 non-null    int64
10  breathes             70 non-null    int64
11  venomous             70 non-null    int64
12  fins                 70 non-null    int64
13  legs                 70 non-null    int64
14  tail                 70 non-null    int64
15  domestic             70 non-null    int64
16  catsize              70 non-null    int64
17  class_type           70 non-null    int64
dtypes: int64(17), object(1)
memory usage: 10.0+ KB
```

In [18]:

```
1 data2.describe()
```

Out[18]:

	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed
count	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000
mean	0.314286	0.285714	0.728571	0.271429	0.314286	0.328571	0.442857	0.485714
std	0.467583	0.455016	0.447907	0.447907	0.467583	0.473085	0.500310	0.503401
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000
75%	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

In [19]:



```
1 data1.nunique()
```

Out[19]:

```
animal_name    43
hair            2
feathers        1
eggs            1
milk            1
airborne        2
aquatic         2
predator        2
toothed         2
backbone        2
breathes        2
venomous        2
fins            2
legs            5
tail            2
domestic        2
catsize         2
class_type      5
dtype: int64
```

In [20]:



```
1 data2.nunique()
```

Out[20]:

```
animal_name    70
hair            2
feathers        2
eggs            2
milk            2
airborne        2
aquatic         2
predator        2
toothed         2
backbone        2
breathes        2
venomous        2
fins            2
legs            5
tail            2
domestic        2
catsize         2
class_type      7
dtype: int64
```

In [21]:



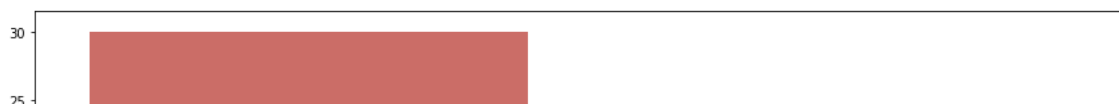
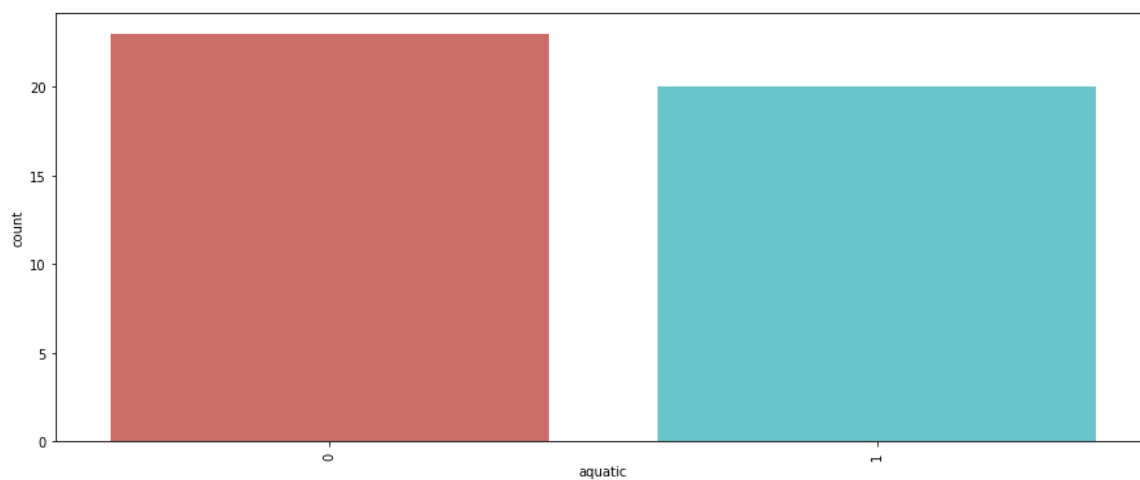
```
1 data11 = data1[['hair', 'feathers', 'eggs', 'milk', 'airborne',
2               'aquatic', 'predator', 'toothed', 'backbone', 'breathes', 'venomous',
3               'fins', 'legs', 'tail', 'domestic', 'catsize', 'class_type']]
```

In [22]:

```
1 data22 = data2[['hair', 'feathers', 'eggs', 'milk', 'airborne',  
2               'aquatic', 'predator', 'toothed', 'backbone', 'breathes', 'venomous',  
3               'fins', 'legs', 'tail', 'domestic', 'catsize', 'class_type']]
```

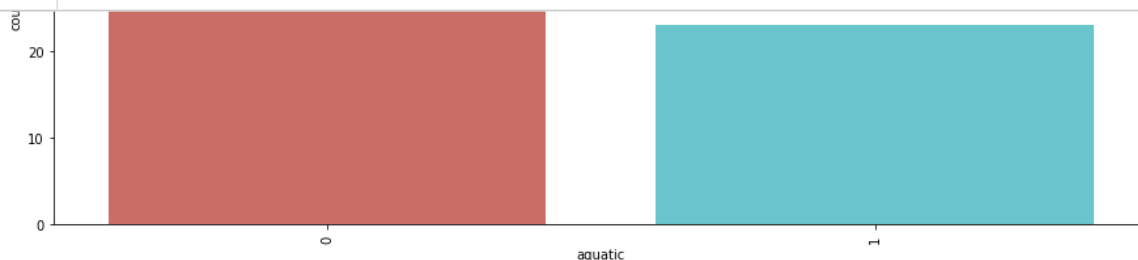
In [23]:

```
1 for i in data11.columns:  
2     plt.figure(figsize=(15,6))  
3     sns.countplot(data11[i], data = data11,  
4                   palette='hls')  
5     plt.xticks(rotation = 90)  
6     plt.show()
```



In [24]:

```
1 for i in data22.columns:  
2     plt.figure(figsize=(15,6))  
3     sns.countplot(data22[i], data = data22,  
4                   palette='hls')  
5     plt.xticks(rotation = 90)  
6     plt.show()
```



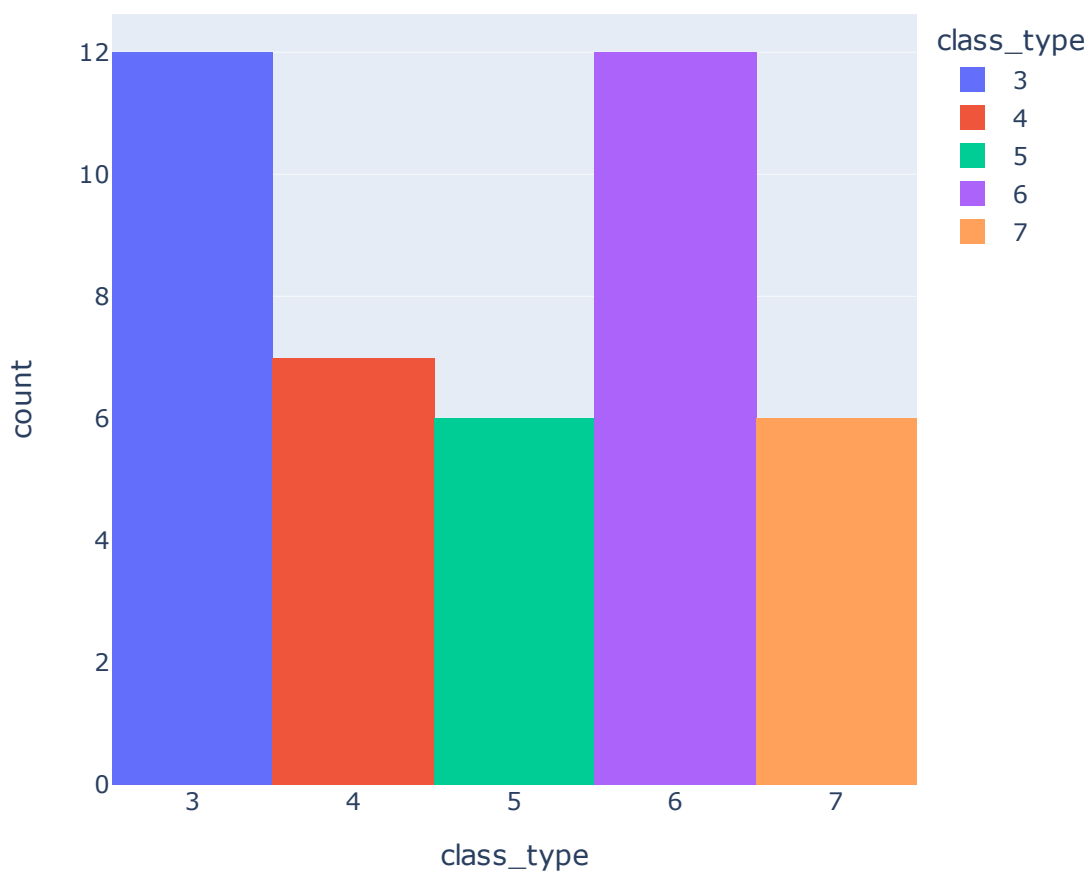
In [25]:

```
1 import plotly.express as px
```

In [26]:

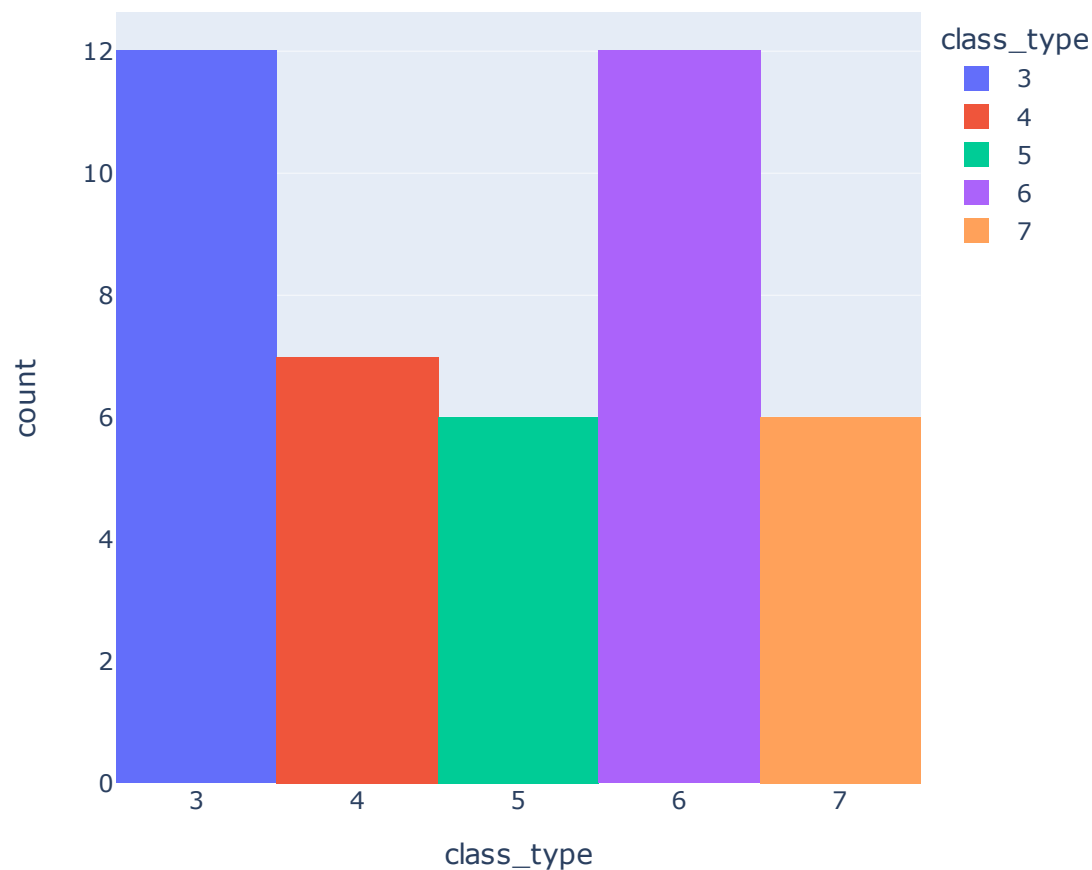


```
1 fig1 = px.histogram(data1, x = 'class_type', color = 'class_type')
2 fig1.show()
```



In [27]:

```
1 fig2 = px.histogram(data2, x = 'class_type', color = 'class_type')
2 fig1.show()
```



In [28]:

```
1 data1 = data1.drop(columns = ['animal_name', 'class_type'])
```

In [29]:

```
1 data1.head()
```

Out[29]:

	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	venomo
0	0	0	1	0	0	1	0	0	1	1	
1	0	0	1	0	0	0	0	1	1	1	
2	0	0	1	0	0	0	1	1	1	1	
3	0	0	1	0	0	0	1	1	1	1	
4	0	0	1	0	0	0	0	1	1	1	

In [30]:



```
1 X = data1.iloc[:,:]
2 print(X)
```

	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	\
0	0	0	1	0	0	1	0	0	
1	0	0	1	0	0	0	0	1	
2	0	0	1	0	0	0	1	1	
3	0	0	1	0	0	0	1	1	
4	0	0	1	0	0	0	0	1	
5	0	0	1	0	0	0	1	1	
6	0	0	1	0	0	0	1	1	
7	0	0	1	0	0	0	1	1	
8	0	0	1	0	0	1	1	1	
9	0	0	1	0	0	1	1	1	
10	0	0	1	0	0	1	1	1	
11	0	0	1	0	0	0	0	1	
12	0	0	1	0	0	1	0	0	
13	0	0	1	0	0	1	0	0	
14	0	0	1	0	0	1	0	0	
15	0	0	1	0	0	1	0	0	
16	0	0	1	0	0	1	1	1	
17	0	0	1	0	0	1	0	0	
18	0	0	1	0	0	1	0	0	
19	0	0	1	0	0	1	0	1	
20	0	0	1	0	0	1	0	1	
21	0	0	1	0	0	1	0	1	
22	0	0	1	0	0	1	0	1	
23	0	0	1	0	0	1	0	1	
24	0	0	1	0	0	1	0	1	
25	0	0	1	0	1	0	0	0	
26	0	0	1	0	1	0	1	0	
27	0	0	1	0	0	0	0	0	
28	0	0	1	0	0	0	0	0	
29	1	0	1	0	1	0	0	0	
30	0	0	1	0	1	0	1	0	
31	0	0	1	0	0	0	0	0	
32	0	0	1	0	0	0	1	0	
33	0	0	1	0	1	0	0	0	
34	0	0	1	0	0	0	0	0	
35	0	0	1	0	1	0	0	0	
36	0	0	1	0	1	0	0	0	
37	0	0	1	0	0	1	0	0	
38	0	0	1	0	0	0	1	1	
39	0	0	1	0	0	0	0	0	
40	0	0	1	0	0	0	0	0	
41	0	0	1	0	0	1	0	0	
42	0	0	1	0	0	1	0	0	

	backbone	breathes	venomous	fins	legs	tail	domestic	catsize
0	1	1	0	0	4	1	1	1
1	1	1	0	0	4	1	1	0
2	1	1	0	0	4	1	1	1
3	1	1	0	0	4	1	0	0
4	1	1	0	0	4	1	1	0
5	1	1	1	0	0	1	0	1
6	1	1	0	0	0	1	0	1
7	1	1	1	0	0	1	0	1
8	1	1	0	0	4	1	0	1

9	1	1	0	0	4	1	0	1
10	1	1	0	0	4	1	0	1
11	1	1	0	0	4	1	0	0
12	1	0	0	1	0	1	0	0
13	1	0	0	1	0	1	0	1
14	1	0	0	1	0	1	0	1
15	1	0	0	1	0	1	0	1
16	1	0	0	1	0	1	0	1
17	1	0	0	1	0	1	0	1
18	1	0	0	1	0	1	0	1
19	1	1	0	0	4	1	1	0
20	1	1	0	0	2	1	0	0
21	1	1	0	0	4	0	0	0
22	1	1	0	0	4	0	0	0
23	1	1	0	0	4	0	0	0
24	1	1	0	0	4	0	0	0
25	0	1	0	0	6	0	0	0
26	0	1	1	0	6	0	0	0
27	0	1	0	0	6	0	0	0
28	0	1	0	0	6	0	0	0
29	0	1	0	0	6	0	0	0
30	0	1	0	0	6	0	0	0
31	0	1	0	0	6	0	0	0
32	0	1	0	0	6	0	0	0
33	0	1	0	0	6	0	0	0
34	0	1	0	0	6	0	0	0
35	0	1	0	0	6	0	0	0
36	0	1	0	0	6	0	0	0
37	0	0	0	0	0	0	0	0
38	0	1	1	0	8	0	0	0
39	0	1	0	0	0	0	0	0
40	0	1	0	0	0	0	0	0
41	0	0	1	0	0	0	0	1
42	0	0	0	0	0	0	0	1

In [34]:



```

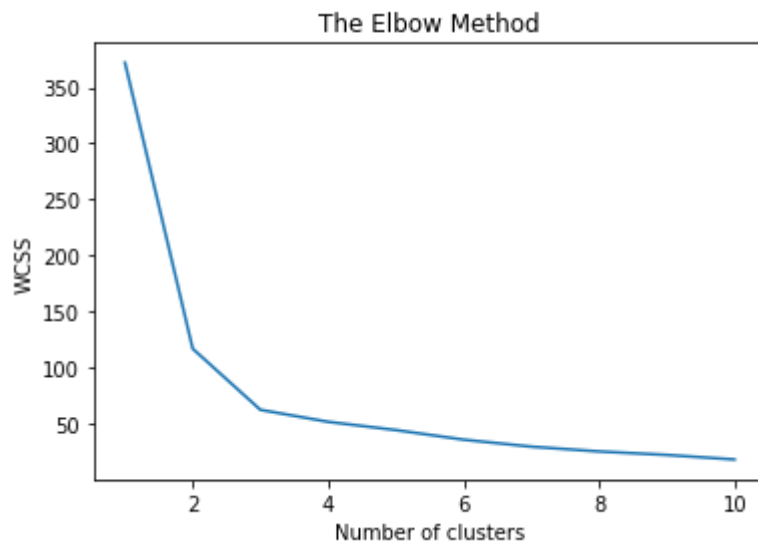
1 from sklearn.cluster import KMeans
2 from yellowbrick.cluster import KElbowVisualizer

```

In [33]:



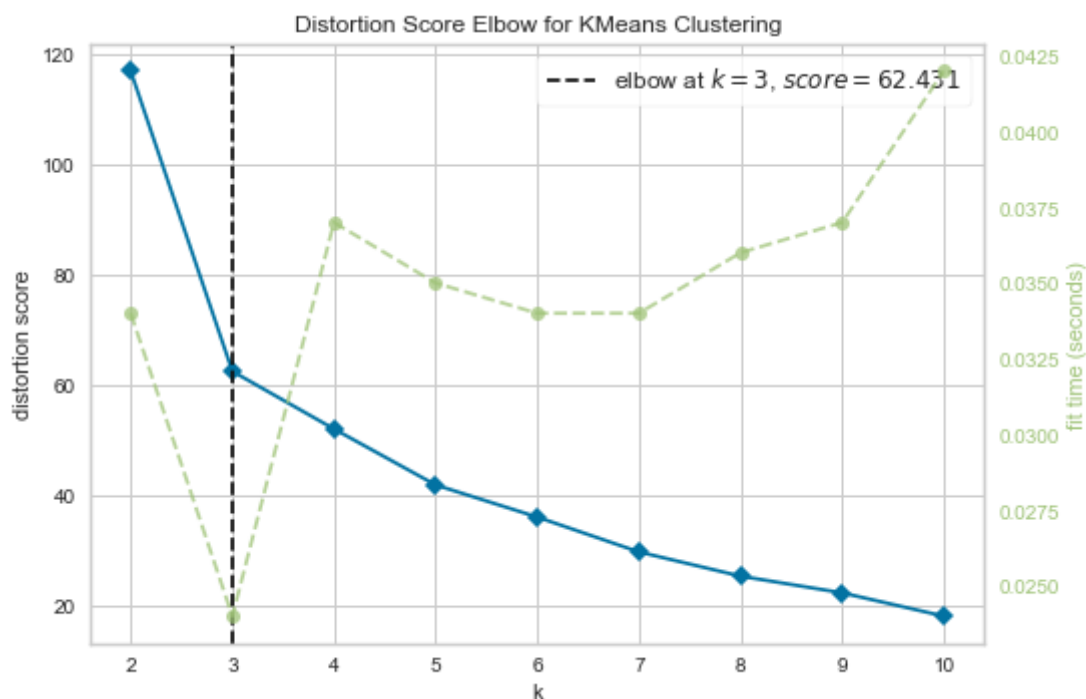
```
1 wcss = []
2 for i in range(1, 11):
3     kmeans = KMeans(n_clusters = i, init = 'k-means++',
4                     random_state = 42)
5     kmeans.fit(X)
6     wcss.append(kmeans.inertia_)
7 plt.plot(range(1, 11), wcss)
8 plt.title('The Elbow Method')
9 plt.xlabel('Number of clusters')
10 plt.ylabel('WCSS')
11 plt.show()
```



In [35]:

```
1 # Quick examination of elbow method to find numbers of clusters to make.
2 print('Elbow Method to determine the number of clusters to be formed:')
3 Elbow_M = KElbowVisualizer(KMeans(), k=10)
4 Elbow_M.fit(X)
5 Elbow_M.show()
```

Elbow Method to determine the number of clusters to be formed:



Out[35]:

```
<AxesSubplot:title={'center': 'Distortion Score Elbow for KMeans Clusterin
g'}, xlabel='k', ylabel='distortion score'>
```

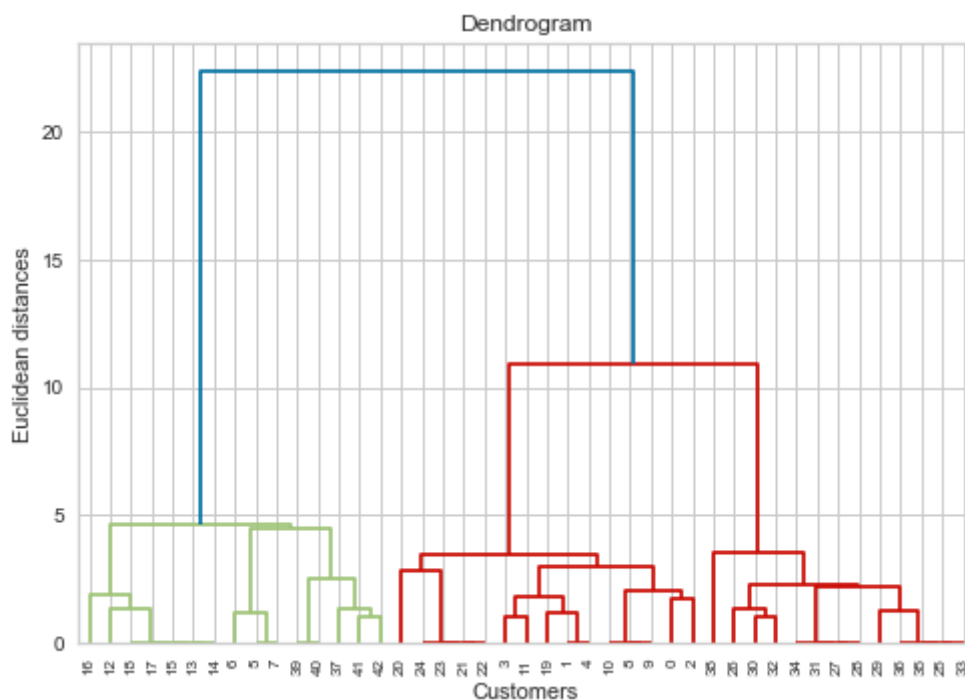
In [36]:



```

1 import scipy.cluster.hierarchy as sch
2 dendrogram = sch.dendrogram(sch.linkage(X, method = 'ward'))
3 plt.title('Dendrogram')
4 plt.xlabel('Customers')
5 plt.ylabel('Euclidean distances')
6 plt.show()

```



In [38]:



```

1 kmeans = KMeans(
2     n_clusters=2, init="k-means++",
3     n_init=10,
4     tol=1e-04, random_state=42
5 )
6 kmeans.fit(X)
7 clusters=pd.DataFrame(X,columns=data1.columns)
8 clusters['label']=kmeans.labels_
9 polar=clusters.groupby("label").mean().reset_index()
10 polar=pd.melt(polar,id_vars=["label"])

```

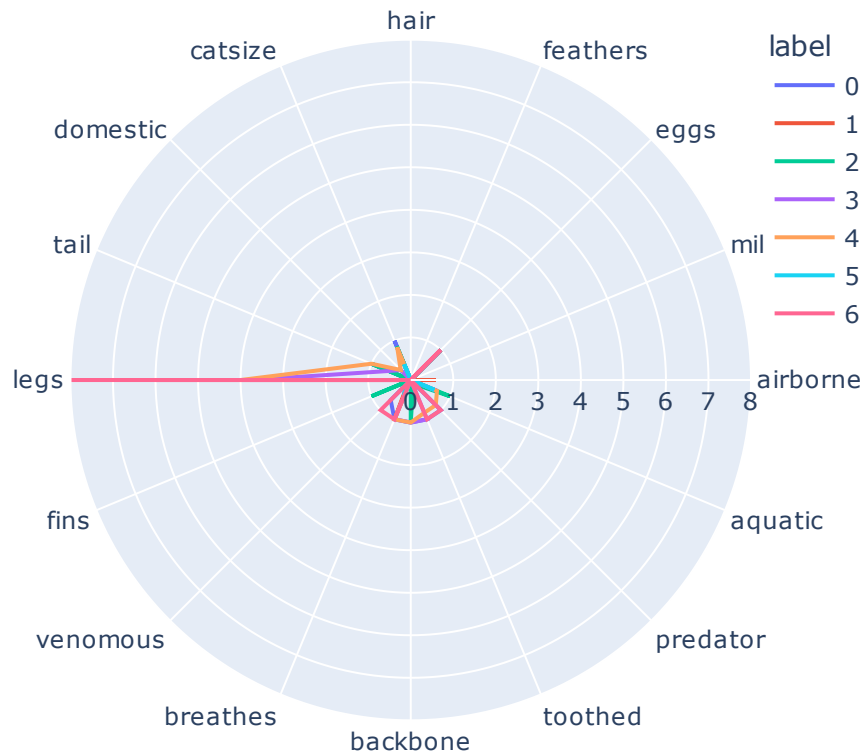

In [45]:



```

1 fig3 = px.line_polar(polar, r="value", theta="variable",
2                       color="label", line_close=True,height=500,
3                       width=500)
4 fig3.show()

```



In [40]:



```

1 kmeans = KMeans(
2     n_clusters=5, init="k-means++",
3     n_init=10,
4     tol=1e-04, random_state=42
5 )
6 kmeans.fit(X)
7 clusters=pd.DataFrame(X,columns=data1.columns)
8 clusters['label']=kmeans.labels_
9 polar=clusters.groupby("label").mean().reset_index()
10 polar=pd.melt(polar,id_vars=["label"])

```

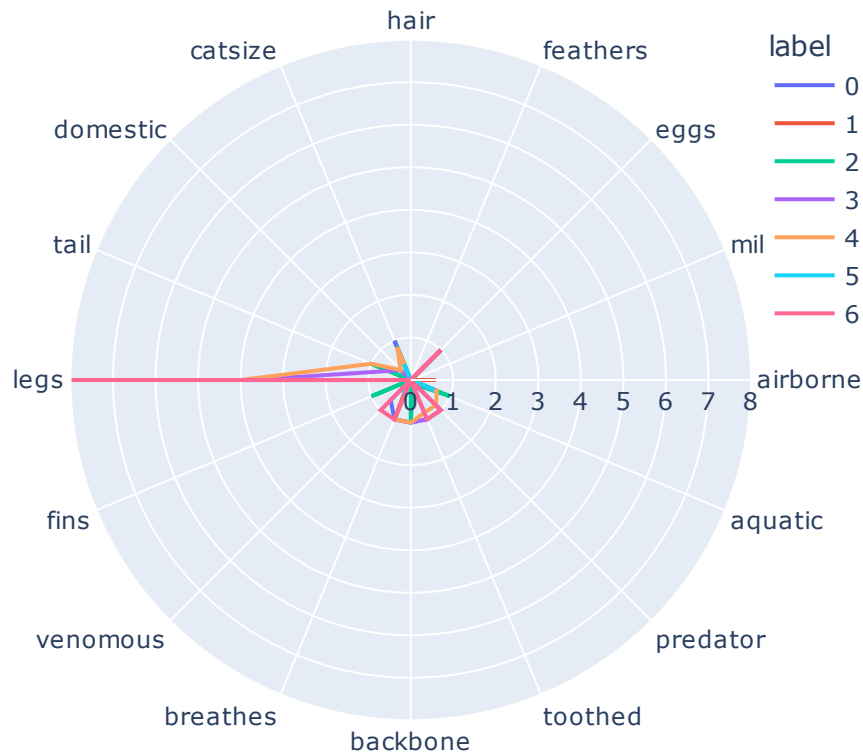
In [46]:



```

1 fig4 = px.line_polar(polar, r="value", theta="variable",
2                       color="label", line_close=True,height=500,
3                       width=500)
4 fig4.show()

```



In [42]:



```

1 kmeans = KMeans(
2     n_clusters=7, init="k-means++",
3     n_init=10,
4     tol=1e-04, random_state=42
5 )
6 kmeans.fit(X)
7 clusters=pd.DataFrame(X,columns=data1.columns)
8 clusters['label']=kmeans.labels_
9 polar=clusters.groupby("label").mean().reset_index()
10 polar=pd.melt(polar,id_vars=["label"])

```

In [47]:

```
1 fig5 = px.line_polar(polar, r="value", theta="variable",  
2                       color="label", line_close=True,height=500,  
3                       width=500)  
4 fig5.show()
```

