MULTIDIMENSIONAL DATA ANALYSIS USING CATEGORICAL DATA ANALYSIS IN PYTHON

Aim:

- Multi-dimensional data analysis is an informative analysis of data which takes many relationships into account.
- Some basic techniques are used for analysing multidimensional/multivariate data using open-source libraries written in Python.
- A dataset called “zoo_data.csv” is taken to perform these techniques.
- The type of data we have here is typically categorical.
- The techniques used in this case study for categorical data analysis are very basic ones which are simple to understand, interpret and implement.
- These include cluster analysis, correlation analysis, PCA (Principal component analysis) and EDA (Exploratory Data Analysis) analysis.

Procedure:

Cluster Analysis:

- As the data we have is based on the characteristics of different types of animals, we can classify animals into different groups(clusters) or subgroups using some well-known clustering techniques namely K-Means clustering, DBscan, Hierarchical clustering & KNN (K-Nearest Neighbours) clustering.
- For sake of simplicity, K-Means clustering ought to be a better option in this case.

- Clustering data using K-means clustering technique can be achieved using K-Means module of cluster class of sklearn library as follows:

In [6]:
```
from sklearn.cluster import KMeans
clusters = 7

kmeans = KMeans(n_clusters = clusters)
kmeans.fit(zoo_data)

print(kmeans.labels_)
```

```
C:\Users\jai\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:1334: Us
o have a memory leak on Windows with MKL, when there are less chunks than av
id it by setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

```
[0 0 4 0 0 0 0 4 4 0 0 3 4 1 5 2 3 0 4 4 3 3 0 3 2 5 5 6 0 6 2 0 6 3 4 0 0
 3 4 2 2 3 2 3 0 0 2 0 0 0 0 2 5 2 0 0 3 3 3 3 4 4 1 0 0 0 4 0 0 0 0 3 2 4
 4 6 4 1 3 3 1 1 4 3 6 5 4 3 2 5 5 5 4 6 0 3 6 2 0 1 3]
```

In [5]:
```
import pandas as pd

zoo_data = pd.read_csv(r"C:\Users\jai\Desktop\zoo_data-1.csv", encoding = 'utf-8',index_col = ["animal

# print first 5 rows of zoo data
print(zoo_data.head())
```

animal_name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	\
aardvark	1	0	0	1	0	0	1	1	
antelope	1	0	0	1	0	0	0	1	
bass	0	0	1	0	0	1	1	1	
bear	1	0	0	1	0	0	1	1	
boar	1	0	0	1	0	0	1	1	

animal_name	backbone	breathes	venomous	fins	legs	tail	domestic	catsize
aardvark	1	1	0	0	4	0	0	1
antelope	1	1	0	0	4	1	0	1
bass	1	0	0	1	0	1	0	0
bear	1	1	0	0	4	0	0	1
boar	1	1	0	0	4	1	0	1

Here, overall cluster inertia comes out to be **119.70392382759556**. This value is stored in kmeans.inertia_ variable.

EDA Analysis:

- To perform EDA analysis, we need to reduce dimensionality of multivariate data we have to trivariate/bivariate(2D/3D) data.
- We can achieve this task using PCA(Principal Component Analysis).
- PCA can be carried out using PCA module of class decomposition of library sklearn as follows:

```
In [7]: from sklearn.decomposition import PCA

        pca = PCA(3)
        pca.fit(zoo_data)

        pca_data = pd.DataFrame(pca.transform(zoo_data))

        print(pca_data.head())
```

```
          0         1         2
0  1.351029 -1.058533  0.314103
1  1.306634 -1.208344 -0.289405
2 -3.131655  0.252200  0.929419
3  1.351029 -1.058533  0.314103
4  1.277296 -1.225750  0.126239
```

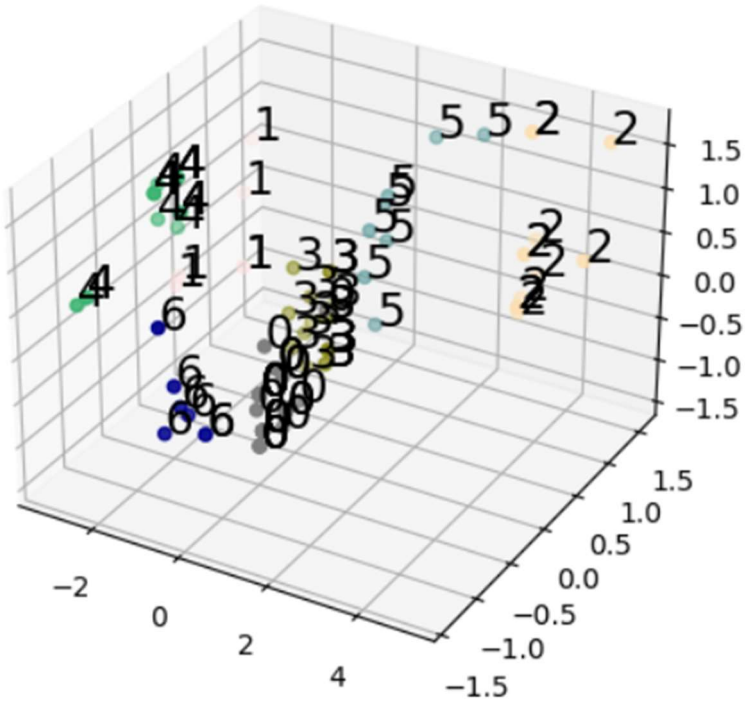
- Data output above represents reduced trivariate(3D) data on which we can perform EDA analysis.
- Reduced Data produced by PCA can be used indirectly for performing various analysis but is not directly human interpretable.

Using Data Visualization for better understanding:

- Scatter plot is a 2D/3D plot which is helpful in analysis of various clusters in 2D/3D data.
- Scatter plot of 3D reduced data we produced earlier can be plotted as follows:
  The code below is a **Pythonic** code which generates an array of colors(where number of colors are approximately equal to number of clusters) sorted in order of their hue, value and saturation values.
- Here each color is associated with a single cluster and will be used to denote an animal as a 3D point while plotting it in a 3D plot/space(Scatter Plot in this case).

In [8]:
```
from matplotlib import colors as mcolors
import math

''' Generating different colors in ascending order
                                of their hsv values '''
colors = list(zip(*sorted((
                tuple(mcolors.rgb_to_hsv(
                    mcolors.to_rgba(color)[:3])), name)
                for name, color in dict(
                    mcolors.BASE_COLORS, **mcolors.CSS4_COLORS
                                                ).items())))[1]


# number of steps to taken generate n(clusters) colors
skips = math.floor(len(colors[5 : -5])/clusters)
cluster_colors = colors[5 : -5 : skips]
```

In [9]:
```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt

fig = plt.figure()
ax = fig.add_subplot(111, projection = '3d')
ax.scatter(pca_data[0], pca_data[1], pca_data[2],
        c = list(map(lambda label : cluster_colors[label],
                                    kmeans.labels_)))

str_labels = list(map(lambda label:'% s' % label, kmeans.labels_))

list(map(lambda data1, data2, data3, str_label:
        ax.text(data1, data2, data3, s = str_label, size = 16.5,
        zorder = 20, color = 'k'), pca_data[0], pca_data[1],
        pca_data[2], str_labels))

plt.show()
```

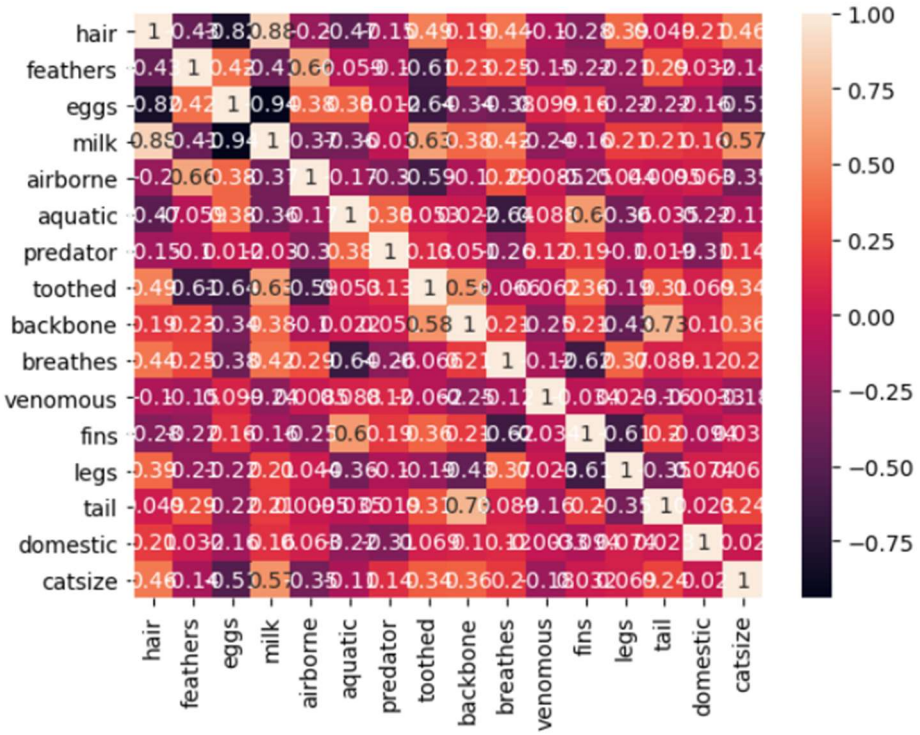


- Closely analysing the scatter plot can lead to hypothesis that the clusters formed using the initial data doesn't have good enough explanatory power.

- To solve this issue, we need to bring down our set of features to a more useful set of features using which we can generate useful clusters.
- One way of producing such a set of features is to carry out correlation analysis.
- This can be done by plotting heatmaps and trisurface plots as follows:

Heatmap:

```
In [10]: import seaborn as sns

         # generating correlation heatmap
         sns.heatmap(zoo_data.corr(), annot = True)

         # posting correlation heatmap to output console
         plt.show()
```



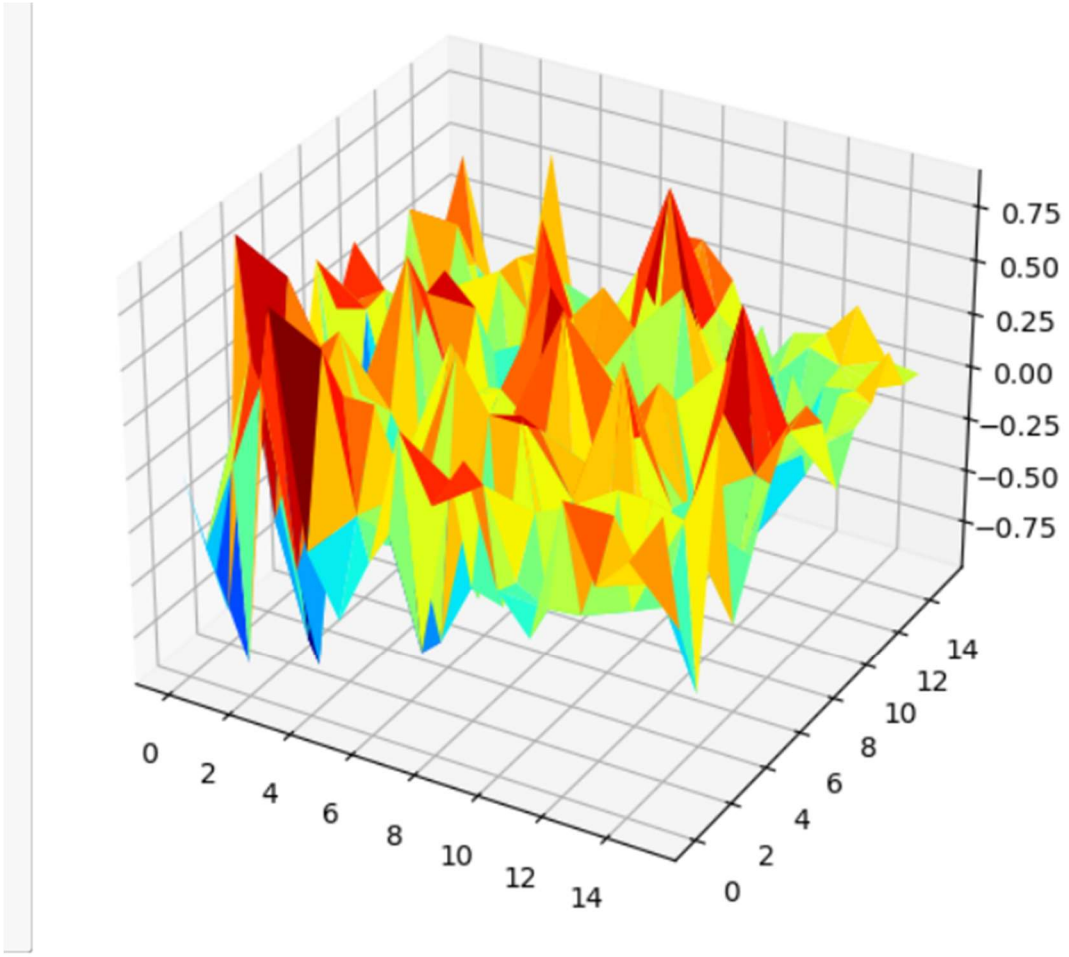
Following code is used to generate a trisurface plot of correlation matrix by making a list of tuples where a tuple contains

coordinates and correlation value in order of animal names.
Pseudocode for above explanation:

```
# PseudoCode
tuple -> (position_in_dataframe(feature1),
          position_in_dataframe(feature2),
          correlation(feature1, feature2))
```

Code for generating trisurface plot for correlation matrix:

In [11]:
```
from matplotlib import cm

# generating correlation data
df = zoo_data.corr()
df.index = range(0, len(df))
df.rename(columns = dict(zip(df.columns, df.index)), inplace = True)
df = df.astype(object)

''' Generating coordinates with
corresponding correlation values '''
for i in range(0, len(df)):
    for j in range(0, len(df)):
        if i != j:
            df.iloc[i, j] = (i, j, df.iloc[i, j])
        else :
            df.iloc[i, j] = (i, j, 0)

df_list = []

# flattening dataframe values
for sub_list in df.values:
    df_list.extend(sub_list)

# converting list of tuples into trivariate dataframe
plot_df = pd.DataFrame(df_list)

fig = plt.figure()
ax = Axes3D(fig)

# plotting 3D trisurface plot
ax.plot_trisurf(plot_df[0], plot_df[1], plot_df[2],
                cmap = cm.jet, linewidth = 0.2)

plt.show()
```



- Using heatmap and trisurface plot, we can make some inferences on how to select a smaller set of features used for performing cluster analysis.
- Generally, feature pairs with extreme correlation values carry high explanatory power and can be used for further analysis.

Conclusion:

So, this is how creation and visualization “Multidimensional Data Analysis Using Categorical Data Analysis” in Python is done.