In [ ]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
import pickle
from sklearn.metrics import accuracy_score, recall_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix

for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

In [3]:

```
data = pd.read_csv('zoo.csv').drop(['class_type', 'animal_name'], axis = 1)
classes = pd.read_csv('class.csv')
```

In [4]:

```
def load(filename):
    #load the model
    loaded_model =  pickle.load(open(filename, 'rb'))
    print(loaded_model)
    return loaded_model
```

In [5]:

```
# For single input
```

def predict(inp, model): d = pd.DataFrame(inp, columns = data.columns) pred = model.predict(d).squeeze() return classes.loc[(classes.Class_Number == pred)].Class_Type.values

In [7]:

```
def predictModel():
    inpNew = [[v1.get(),v2.get(),v3.get(),v4.get(),v5.get(),v6.get(),v7.get(),v8.get(),v9
.get(),v10.get(),v11.get(),v12.get(),int(c13.get()),v14.get(),v15.get(),v16.get()]]
    predictModelHelper(inpNew)
```

In [8]:

```
def predictModelHelper(inp):
    model = load('knnsav/knn.sav')
    #add function to take input as a list. Example given below
    #pass input to a function to predict the output.
    str = 'Predicted class is:' + predict(inp, model)[0];
    labelPredict = Label(root, text=str)
    labelPredict.grid(row=12, column=0, columnspan=2)
```

In [9]:

```
def predict_file(f, model):
    data = pd.read_csv(f)
    X = data.drop(['class_type', 'animal_name'], axis = 1)
    y = data['class_type']
    pred = model.predict(X)
    cm = confusion_matrix(y, pred, labels = range(1, 8)) # output is a 2dim list
    return cm
```

In [10]:

```
filename=''
def browseFiles():
 global filename
```

```
 filename = filedialog.askopenfilename(initialdir="/home/swapnil/Downloads", title="Selec
t File",
  filetypes=(("CSV files", "*.csv*"), ("all files", "*.*")))
```

In [11]:

```
def show():
 label_file_name = Label(root, text = "File Opened: " + filename)
 label_file_name.grid(row=20,column=0,columnspan=2)
```

In [12]:

```
def showCM():
    model = load('knnsav/knn.sav')
    print(predict_file(filename,model))
```

In [13]:

```
from tkinter import *
from tkinter import filedialog
```

In [ ]:

```

```

In [14]:

```
root = Tk()

#Custom Input

v1 = IntVar()
v2 = IntVar()
v3 = IntVar()
v4 = IntVar()
v5 = IntVar()
v6 = IntVar()
v7 = IntVar()
v8 = IntVar()
v9 = IntVar()
v10 = IntVar()
v11 = IntVar()
v12 = IntVar()
v14 = IntVar()
v15 = IntVar()
v16 = IntVar()

c1 = Checkbutton(root, text = "Has Hair?", variable = v1, \
                 onvalue = 1, offvalue = 0)
c2 = Checkbutton(root, text = "Has Feathers?", variable = v2, \
                 onvalue = 1, offvalue = 0)
c3 = Checkbutton(root, text = "Lays Eggs?", variable = v3, \
                 onvalue = 1, offvalue = 0)
c4 = Checkbutton(root, text = "Produces Milk?", variable = v4, \
                 onvalue = 1, offvalue = 0)
c5 = Checkbutton(root, text = "Does it Fly?", variable = v5, \
                 onvalue = 1, offvalue = 0)
c6 = Checkbutton(root, text = "Is Aquatic?", variable = v6, \
                 onvalue = 1, offvalue = 0)
c7 = Checkbutton(root, text = "Is Predator?", variable = v7, \
                 onvalue = 1, offvalue = 0)
c8 = Checkbutton(root, text = "Has Teeth?", variable = v8, \
                 onvalue = 1, offvalue = 0)
c9 = Checkbutton(root, text = "Has Backbone?", variable = v9, \
                 onvalue = 1, offvalue = 0)

c10 = Checkbutton(root, text = "Does it Breathe?", variable = v10, \
                 onvalue = 1, offvalue = 0)
c11 = Checkbutton(root, text = "Is Venomous?", variable = v11, \
                 onvalue = 1, offvalue = 0)
c12 = Checkbutton(root, text = "Has Fins?", variable = v12, \
```

```
                  onvalue = 1, offvalue = 0)
c14 = Checkbutton(root, text = "Has Tail?", variable = v14, \
                  onvalue = 1, offvalue = 0)
c15 = Checkbutton(root, text = "Is Domestic?", variable = v15, \
                  onvalue = 1, offvalue = 0)
c16 = Checkbutton(root, text = "Is Catsize?", variable = v16, \
                  onvalue = 1, offvalue = 0)
c13 = Entry(root)
c13.insert(0, 'Number of Legs')

c1.grid(row=3,column=0)
c2.grid(row=3,column=1)
c3.grid(row=4,column=0)
c4.grid(row=4,column=1)

c5.grid(row=5,column=0)
c6.grid(row=5,column=1)
c7.grid(row=6,column=0)
c8.grid(row=6,column=1)

c9.grid(row=7,column=0)
c10.grid(row=7,column=1)
c11.grid(row=8,column=0)
c12.grid(row=8,column=1)

c14.grid(row=9,column=0)
c15.grid(row=9,column=1)
c16.grid(row=10,column=0)
c13.grid(row=10,column=1, columnspan=1)

myButton3 = Button(root, text="Predict", command=predictModel)
myButton3.grid(row=11,column=0, columnspan=2)

# File Input
button_file_open = Button(root, text="Browse File", command=browseFiles)
button_file_open.grid(row=19,column=0)

button_file_open2 = Button(root, text="Show FileName", command=show)
button_file_open2.grid(row=19,column=1)

button_predict = Button(root, text="Show Confusion Matrix", command=showCM)
button_predict.grid(row=21,column=0,columnspan=2)

root.mainloop()
```

In [39]:

```
```

In [ ]:

```
```