Student Nane: Dhaniswar B.K. And StudentID: NP03A190318

Applying Logistic Regression on Zoo to predict Class type of Animal

In [2]:
```
import pandas as pd
import numpy as np
```

importing pandas library and numpy library

In [3]:
```
data = pd.read_csv("zoo.csv")
data
```

Out[3]:

	animal_name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	venomous	fins	legs	tail	domestic	cat
0	aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	
1	antelope	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	
2	bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	
3	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	
4	boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
96	wallaby	1	0	0	1	0	0	0	1	1	1	0	0	2	1	0	
97	wasp	1	0	1	0	1	0	0	0	0	1	1	0	6	0	0	
98	wolf	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	
99	worm	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	
100	wren	0	1	1	0	1	0	0	0	1	1	0	0	2	1	0	

101 rows × 18 columns

Reading data and uploading data on programme from csv file using pandas library

In [4]:
```
X = data.iloc[ : , :-1]
y = data.iloc[ : , 17: ]
```

Dividing Data on explanetory and responce variable. i.e. eccessing the specific columns using iloc method

In [5]:
```
X
```

Out[5]:

	animal_name	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	venomous	fins	legs	tail	domestic	cat
0	aardvark	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	
1	antelope	1	0	0	1	0	0	0	1	1	1	0	0	4	1	0	
2	bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	
3	bear	1	0	0	1	0	0	1	1	1	1	0	0	4	0	0	
4	boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
96	wallaby	1	0	0	1	0	0	0	1	1	1	0	0	2	1	0	
97	wasp	1	0	1	0	1	0	0	0	0	1	1	0	6	0	0	
98	wolf	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	
99	worm	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	
100	wren	0	1	1	0	1	0	0	0	1	1	0	0	2	1	0	

101 rows × 17 columns

Displaying the explanetory variable i.e. X data

In [6]:
```
y
```

Out[6]:

	class_type
0	1
1	1
2	4
3	1

	class_type
4	1
...	...
96	1
97	6
98	1
99	7
100	2

101 rows × 1 columns

Displaying the explanetory variable i.e. X data

In [7]:
```
Animal_name =pd.get_dummies(X['animal_name'])
```

In above dataset, animal_name column have all String value so, String values are coverting into numeric value using pandas.get_dummies method.

In [8]:
```
X = X.drop("animal_name", axis=1)
```

After coverting into numeric value we have to delete Animal_name column using drop() method.

In [9]:
```
X = pd.concat([X,Animal_name], axis=1)
```

After dropping the Animal_column we have to concat replace colimn name with X

In [10]:
```
X
```

Out[10]:

	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	...	tuatara	tuna	vampire	vole	vulture	wallaby	wasp
0	1	0	0	1	0	0	1	1	1	1	...	0	0	0	0	0	0	0
1	1	0	0	1	0	0	0	1	1	1	...	0	0	0	0	0	0	0
2	0	0	1	0	0	1	1	1	1	0	...	0	0	0	0	0	0	0
3	1	0	0	1	0	0	1	1	1	1	...	0	0	0	0	0	0	0
4	1	0	0	1	0	0	1	1	1	1	...	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

	hair	feathers	eggs	milk	airborne	aquatic	predator	toothed	backbone	breathes	...	tuatara	tuna	vampire	vole	vulture	wallaby	wasp
96	1	0	0	1	0	0	0	1	1	1	...	0	0	0	0	0	1	0
97	1	0	1	0	1	0	0	0	0	1	...	0	0	0	0	0	0	1
98	1	0	0	1	0	0	1	1	1	1	...	0	0	0	0	0	0	0
99	0	0	1	0	0	0	0	0	0	1	...	0	0	0	0	0	0	0
100	0	1	1	0	1	0	0	0	1	1	...	0	0	0	0	0	0	0

101 rows × 116 columns

After converting String column into numeric displaying the explanetory variable i.e. X data

In [11]:
```
X.shape
```

Out[11]:
```
(101, 116)
```

Displaying the total row of the X data using shape method

In [12]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100000,
                   multi_class='multinomial', n_jobs=1, penalty='l2',
                   random_state=None, solver='sag', tol=0.0001, verbose=0,
                   warm_start=False)
```

impotring logistic regression from sklearn.linear_model and multi_class='ovr' replace with multi_class='multinomial' because prediction class is more than three and solver='liblinear' is replace with solver='sag' because liblinear is use for binary classification and sag is used for multiclass classification and run time scale of 'sag' is better than liblinear and others.

In [13]:
```
from sklearn.model_selection import train_test_split
```

In [14]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,train_size=0.2, random_state = 20)
```

spliting the data into rtain and test importing train_test_split from sklearn.model_selection

In [15]:
```
model.fit(X_train, y_train.values.ravel())
```

Out[15]:
```
LogisticRegression(max_iter=100000, multi_class='multinomial', n_jobs=1,
                   solver='sag')
```

To run the LogisticRegression we have to use model.fit method

In [17]:
```
y_predicted = model.predict(X_test)
y_predicted
```

Out[17]:
```
array([1, 1, 4, 1, 1, 4, 4, 2, 1, 7, 4, 1, 1, 7, 1, 2, 1, 1, 2, 6, 1, 1,
       7, 1, 1, 6, 2, 1, 1, 1, 1, 7, 1, 2, 4, 1, 4, 4, 1, 1, 1, 3, 2, 6,
       6, 1, 6, 7, 7, 2, 2, 1, 7, 7, 1, 1, 1, 1, 2, 7, 2, 2, 1, 1, 4, 2,
       1, 1, 4, 2, 1, 1, 2, 4, 1, 5, 2, 4, 7, 4, 1], dtype=int64)
```

Displaying the predicted by our model model

In [18]:
```
model.score(X_test, y_test)
```

Out[18]:
```
0.9012345679012346
```

Clculating the accuracy of the model which is 90 % using score method

In [24]:
```
from sklearn.metrics import confusion_matrix
fi = confusion_matrix(y_test, y_predicted)
```

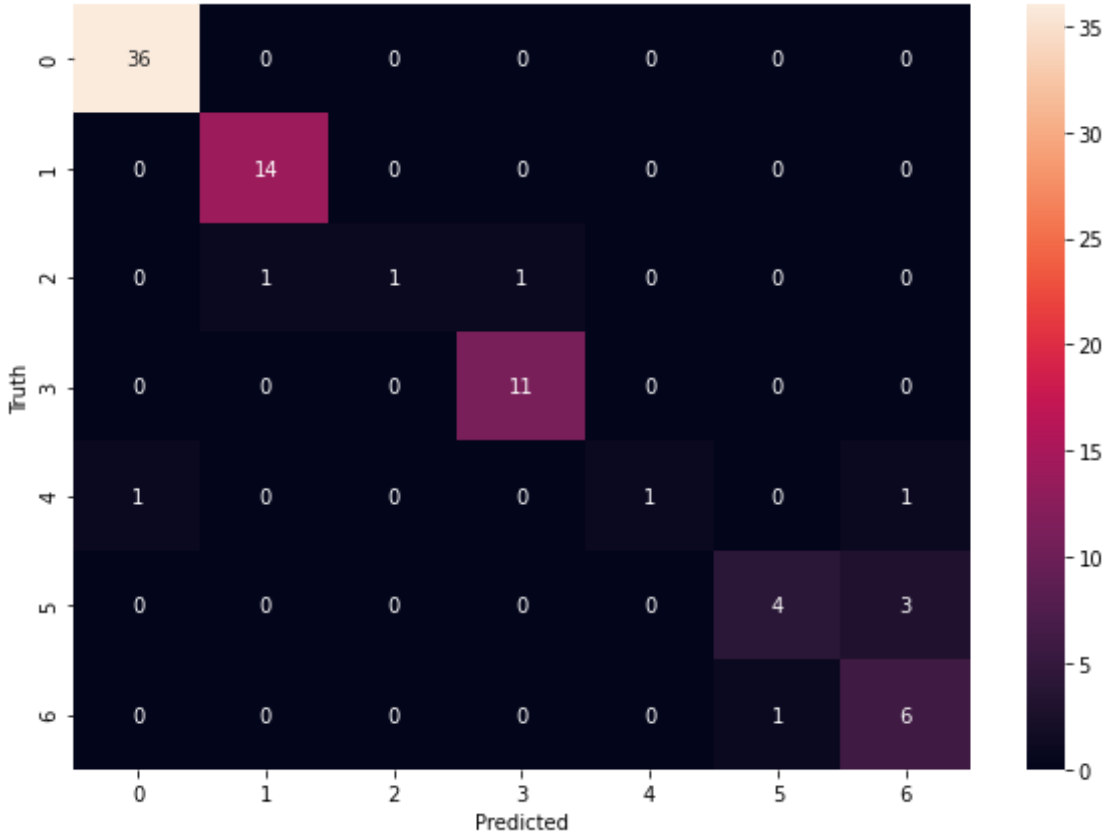
In [25]:
```
fi
```

Out[25]:
```
array([[36,  0,  0,  0,  0,  0,  0],
       [ 0, 14,  0,  0,  0,  0,  0],
       [ 0,  1,  1,  1,  0,  0,  0],
       [ 0,  0,  0, 11,  0,  0,  0],
       [ 1,  0,  0,  0,  1,  0,  1],
       [ 0,  0,  0,  0,  0,  4,  3],
       [ 0,  0,  0,  0,  0,  1,  6]], dtype=int64)
```

importing confusing_matrix from sklearn.metrics to displaying the data in confusing form

In [21]:
```
from matplotlib import pyplot as plt
%matplotlib inline
import seaborn as sn
plt.figure(figsize = (10,7))
sn.heatmap(fi, annot=True)
plt.xlabel("Predicted")
plt.ylabel("Truth")
```

Out[21]:
```
Text(69.0, 0.5, 'Truth')
```



plating Predicted vs Truth value by using matplotlib and seaborn

In [22]:
```
print (pd.DataFrame(confusion_matrix(y_test, y_predict), columns=['Mammal=0','Bird=1','Reptile=2','Fish=3','Amphibia=
```

	Mammal=0	Bird=1	Reptile=2	Fish=3	Amphibia=4	Bug=5	Invertebrate=6
0	36	0	0	0	0	0	0
1	0	14	0	0	0	0	0
2	0	1	1	1	0	0	0
3	0	0	0	11	0	0	0
4	1	0	0	0	1	0	1
5	0	0	0	0	0	4	3
6	0	0	0	0	0	1	6

Displaying confusing matrix with class name

In [23]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_predict))
```

```
              precision    recall  f1-score   support

           1       0.97      1.00      0.99        36
           2       0.93      1.00      0.97        14
           3       1.00      0.33      0.50         3
           4       0.92      1.00      0.96        11
           5       1.00      0.33      0.50         3
           6       0.80      0.57      0.67         7
           7       0.60      0.86      0.71         7

    accuracy                           0.90        81
   macro avg       0.89      0.73      0.75        81
weighted avg       0.91      0.90      0.89        81
```

Calculating the precision, recall, f1-score, accuracy, macro avg and weighted avg using classification_repotr from sklearn.metrics