Name: CHAN King Yeung
SID: 1155119394
SEEM2460 Lab Assignment 1

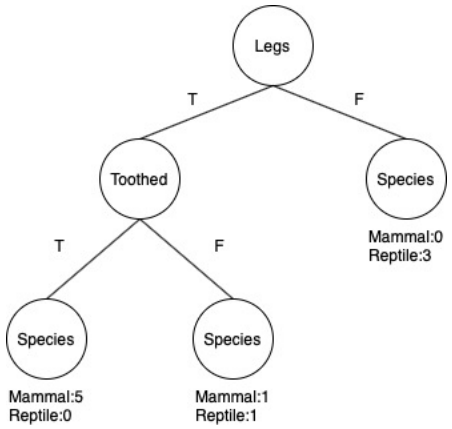
Question 1

a) $Gini_{split\text{“}Toothed\text{”}} = \frac{8}{10}\left[1-\left(\frac{5}{8}\right)^2-\left(\frac{3}{8}\right)^2\right]+\frac{2}{10}\left[1-\left(\frac{1}{2}\right)^2-\left(\frac{1}{2}\right)^2\right] \approx 0.475$

$Gini_{split\text{“}Legs\text{”}} = \frac{7}{10}\left[1-\left(\frac{6}{7}\right)^2-\left(\frac{1}{7}\right)^2\right]+\frac{3}{10}\left[1-\left(\frac{3}{3}\right)^2-\left(\frac{0}{3}\right)^2\right] \approx 0.1714$

Since $Gini_{split\text{“}Toothed\text{”}} > Gini_{split\text{“}Legs\text{”}}$, splitting using “Legs” has lower impurity in the children nodes. Thus, “Legs” has been chosen as the first splitting attribute

b) The following is the two-level decision tree



c) The following is the result in ‘classifier output’ window

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     data-weka.filters.unsupervised.attribute.Discretize-F-B10-M-1.0-Rfirst-
last-precision6-weka.filters.unsupervised.attribute.Discretize-B10-M-1.0-Rfirst-last-
precision6
Instances:    10
Attributes:   3
              Toothed
              Legs
              Species
Test mode:    evaluate on training data

=== Classifier model (full training set) ===

J48 pruned tree
------------------

Legs = T: Mammal (7.0/1.0)
Legs = F: Reptile (3.0)

Number of Leaves  :   2

Size of the tree :        3


Time taken to build model: 0.01 seconds
```

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances	9	90 %
Incorrectly Classified Instances	1	10 %
Kappa statistic	0.7826	
Mean absolute error	0.1714	
Root mean squared error	0.2928	
Relative absolute error	35.468 %	
Root relative squared error	59.7269 %	
Total Number of Instances	10	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.250	0.857	1.000	0.923	0.802	0.875	0.857	Mammal
	0.750	0.000	1.000	0.750	0.857	0.802	0.875	0.850	Reptile
Weighted Avg.	0.900	0.150	0.914	0.900	0.897	0.802	0.875	0.854	

=== Confusion Matrix ===

a	b	<-- classified as
6	0	a = Mammal
1	3	b = Reptile

Question 2

The following is the source code

```
import numpy as np
import cvxpy as cp

A = np.array([[0, -1, 0, -1 ,1 ,0],
              [-2, 1, 0, 2, 0, -1],
              [0, 1, 0, 0, 0, 1],
              [0, 0, 1, 0, -1, 2]])

b = np.array([2, 1, 1, -3])

x = cp.Variable(shape = 6)

constraints = [A * x == b]
obj = cp.Minimize(cp.norm(x[np.array([0, 1])], 1))

prob = cp.Problem(obj, constraints)
prob.solve()

# solution
print(x.value)
```

The above coding return `[-0.000000 0.000000 -2.000000 1.000000 3.000000 1.000000]`

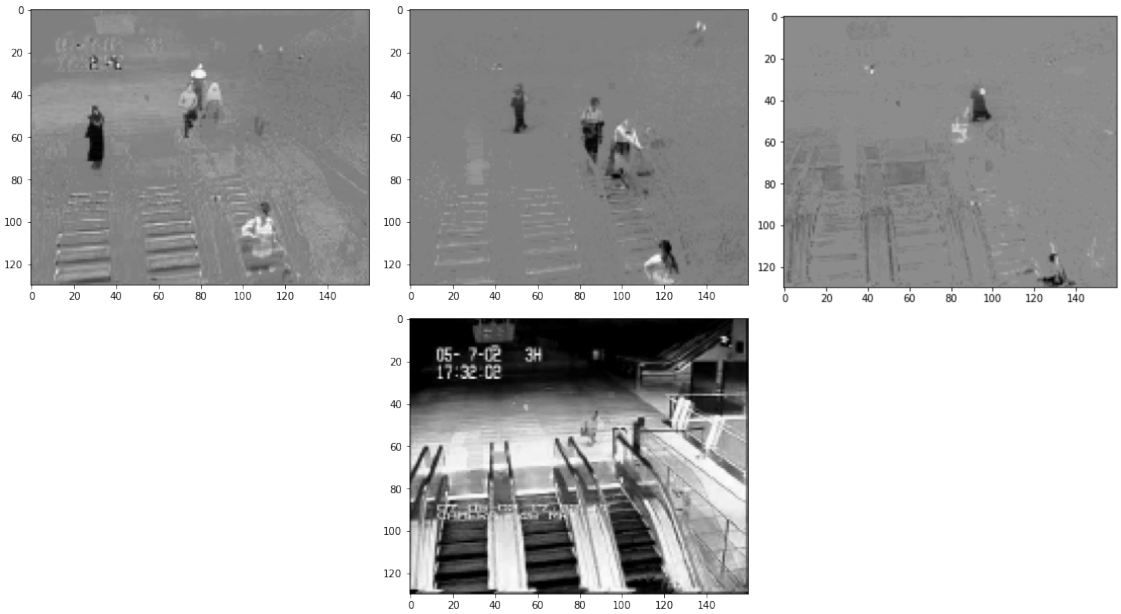
Thus, we obtain $\boldsymbol{x} = (0, 0, -2, 1, 3, 1)'$

Question 3

a) The following is the source code

```
import numpy as np
import cv2
import cvxpy as cp
from cvxpy import *
import matplotlib.pyplot as plt

im1 = cv2.imread('/content/Figure1.png',cv2.IMREAD_GRAYSCALE)
im2 = cv2.imread('/content/Figure2.png',cv2.IMREAD_GRAYSCALE)
im3 = cv2.imread('/content/Figure3.png',cv2.IMREAD_GRAYSCALE)

M_size = im1.shape
size_a = M_size[0]
size_b = M_size[1]
n = size_a*size_b

M1 = im1.reshape(n,-1)
M2 = im2.reshape(n,-1)
M3 = im3.reshape(n,-1)

w = cp.Variable((n,1))

# Please trying to implementing you code here:
################################################

obj = 0
for i in range(3):
  obj += norm((M1 - w), 1) + norm((M2 - w), 1) + norm((M3 - w), 1)

prob = cp.Problem(cp.Minimize(obj))
prob.solve()

################################################
# End of modification here

plt.figure(figsize=(6,6))
plt.imshow((M1 - w.value).reshape(size_a, size_b), cmap='gray')
plt.figure(figsize=(6,6))

plt.imshow((M2 - w.value).reshape(size_a, size_b), cmap='gray')
plt.figure(figsize=(6,6))

plt.imshow((M3 - w.value).reshape(size_a, size_b), cmap='gray')
plt.figure(figsize=(6,6))

plt.imshow((w.value).reshape(size_a, size_b), cmap='gray')
```

The above code return the following graphs



b) Since I am out off relatively static figures, I grab some figures from Github
Source: https://github.com/sayibet/fight-detection-surv-dataset

The following are figures that being inputted into the programme

The following are the output

We can see that there is still some afterglow on the final output, the extracted background. The result is not perfect at all, but it is still acceptable to see how the original background looks like.

My source code for you reference
https://colab.research.google.com/drive/1PIu2pvpGYR8VnzCtS6vmt626AmRPkwun