Test 4
95 % accuracy

In [81]:

```
# Importing all necessary libraries
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from matplotlib import pyplot
import matplotlib.pyplot as plt
from keras.layers import Conv2D, MaxPooling2D
from sklearn.metrics import classification_report, confusion_matrix
from keras.layers import Activation, Dropout, Flatten, Dense
from keras import backend as K
import numpy as np
from keras.models import load_model   #1 load_model to load the model
import keras
from keras.callbacks import ModelCheckpoint # 2 It record the best accuracy
#Image Size
img_width, img_height = 224, 224


# Reading training data from the directory
train_data_dir = '/content/drive/MyDrive/Car Vs Plan Data Sets/train'
# Reading validation data from the directory
validation_data_dir = '/content/drive/MyDrive/Car Vs Plan Data Sets/test'
# Total train instances
nb_train_samples =400
# Total Test instances
nb_validation_samples = 100
batch_size = 8 #change batch size to so it can learn better


 # To check if the shape of input image is correct
if K.image_data_format() ==
  'channels_first': input_shape = (3,
  img_width, img_height)
else:
  input_shape = (img_width, img_height, 3)


# Defining Sequential CNN model
model = Sequential()
model.add(Conv2D(32, (3,3),
input_shape=input_shape))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(32, (3,
3)))
model.add(Activation('relu'
))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(64, (3,
3)))
model.add(Activation('relu'
))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())

model.add(Dense(64))
```

```
model.add(Activation('relu'))

model.add(Dropout(0.5))
model.add(Dense(1))
model.add(Activation('sigmoid'))
```

```
# compile model
model.compile(loss='binary_crossentropy',
optimizer='rmsprop',metrics=['accuracy'])


checkpoint = ModelCheckpoint("best_model.hdf5", monitor='val_accuracy',
    verbose=1, save_best_only=True, mode='auto', period=1)


#cearting virtual datastore for training data set
#Better Data Augmentation with
roataion range shift range and
parameter tuning
train_datagen =
ImageDataGenerator( rotation_ra
nge=40,
width_shift_range=0.2,
height_shift_range=0.2,
rescale=1. / 255,
shear_range=0.2,
zoom_range=0.3,
horizontal_flip=True,
fill_mode='nearest')

#cearting virtual datastore for testing data set
test_datagen = ImageDataGenerator(rescale=1. / 255)


# Fetching training data from folder
train_generator =
train_datagen.flow_from_directory( train_data_
dir,
target_size=(img_width, img_height),
batch_size=batch_size,
class_mode='binary')



# Fetching testing data from folder
validation_generator =
test_datagen.flow_from_directory( validation_data_
dir,
target_size=(img_width, img_height),
batch_size=batch_size,
class_mode='binary')



# Fitting model
history =
model.fit_generator(train_generator,steps_per_epoch=len(train_generator),
validation_data=validation_generator,validation_steps=len(validation_generator),
epochs model.save_weights('first_try.h5')     # 5 always save your weights after
training or during


# list all data in history
print(history.history.keys())
```

```
WARNING:tensorflow:`period` argument is deprecated. Please use `save_freq` to
```

```
specify th e frequency in number of batches seen.
Found 400 images belonging to 2
classes. Found 100 images belonging
to 2 classes.
/usr/local/lib/python3.6/dist-packages/tensorflow/python/keras/engine/
training.py:1844: UserWarning: `Model.fit_generator` is deprecated and will be
removed in a future versio
n. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated
and ' Epoch 1/50
```