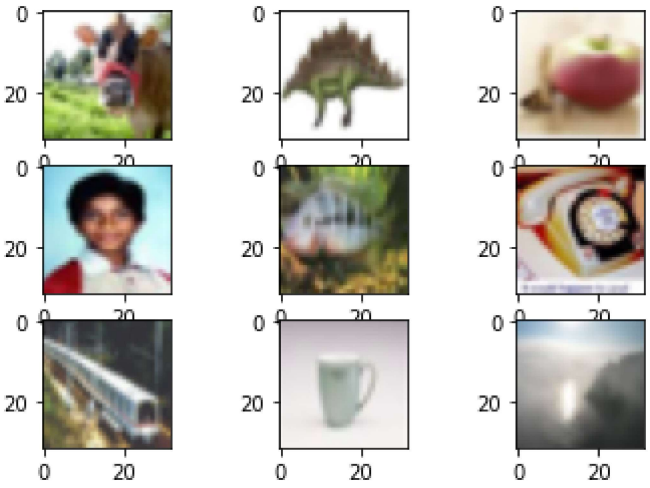
```
#importing important libraries
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, BatchNormalization, Activation
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.utils import np_utils
from keras.datasets import cifar100
import PIL
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
from tensorflow import keras


(x_train, y_train), (x_test, y_test) = cifar100.load_data()


import matplotlib.pyplot as plt
for i in range(9):
  plt.subplot(330+i+1)
  plt.imshow(x_train[i])
plt.show()
```

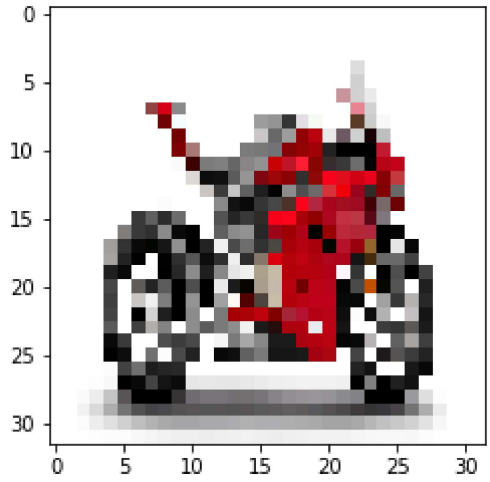


```
y_train = np_utils.to_categorical(y_train,100)
y_test = np_utils.to_categorical(y_test,100)


from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, BatchNormalization, Activation
from tensorflow.keras.models import Sequential
model=Sequential()
model.add(Conv2D(32,(3,3),activation='relu',kernel_initializer='he_uniform',padding='same',in
model.add(Conv2D(32,(3,3),activation='relu',kernel_initializer='he_uniform',padding='same'))
model.add(MaxPooling2D(2,2))
```

```
model.add(MaxPooling2D(2,2))


model.add(Flatten())
model.add(Dense(256, activation = 'relu', kernel_initializer = 'he_uniform'))
model.add(Dense(100, activation = 'softmax'))


from tensorflow.keras.optimizers import SGD


opt = SGD(lr = 0.01, momentum = 0.9)
```

```
/usr/local/lib/python3.7/dist-packages/keras/optimizer_v2/gradient_descent.py:102: UserW
  super(SGD, self).__init__(name, **kwargs)
```

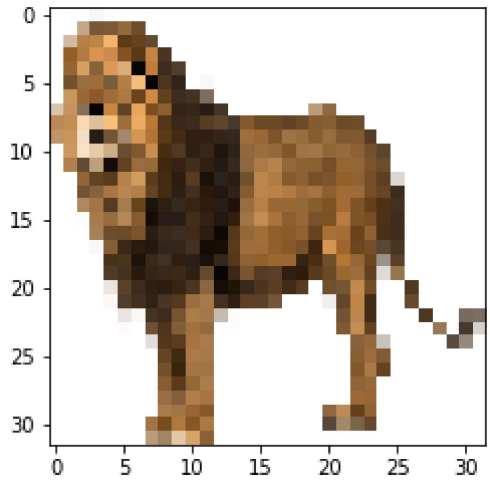
```
from keras.backend import categorical_crossentropy
model.compile(optimizer = opt, loss = 'categorical_crossentropy', metrics = ['accuracy'])


history = model.fit(x_train, y_train, epochs = 1000, batch_size = 256, validation_data = (x_t
# from tensorflow.keras.models import load_model
# model5 = load_model('cifar100_CNN .h5')


model.save('cifar100_CNN.h5')


#model.save('cifar100_CNN.h5')
objects = ['apple', 'aquarium_fish', 'baby', 'bear','beaver', 'bed', 'bee', 'beetle', 'bicycl
'bridge', 'bus', 'butterfly', 'camel','can', 'castle', 'caterpillar', 'cattle', 'chair','chim
'cloud', 'cockroach', 'couch','crab', 'crocodile', 'cup', 'dinosaur', 'dolphin','elephant', '
'house', 'kangaroo', 'keyboard', 'lamp','lawn_mower', 'leopard', 'lion', 'lizard', 'lobster',
'man', 'maple_tree', 'motorcycle', 'mountain','mouse', 'mushroom', 'oak_tree', 'orange', 'orc
'otter', 'palm_tree', 'pear', 'pickup_truck','pine_tree', 'plain', 'plate', 'poppy', 'porcupi
'possum', 'rabbit', 'raccoon', 'ray', 'road', 'rocket','rose', 'sea', 'seal', 'shark', 'shrew
'skyscraper', 'snail', 'snake', 'spider', 'squirrel','streetcar', 'sunflower', 'sweet_pepper'
'tank', 'telephone', 'television', 'tiger', 'tractor','train', 'trout', 'tulip', 'turtle', 'w
'willow_tree', 'wolf', 'woman', 'worm']


from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
img = load_img('motor.jpg', target_size=(32,32))
plt.imshow(img)
img = img_to_array(img)
img = img.reshape(1,32,32,3)
img = img.astype('float32')
img = img/255
```



```
print(objects[np.argmax(model.predict(img))])
```

```
motorcycle
```

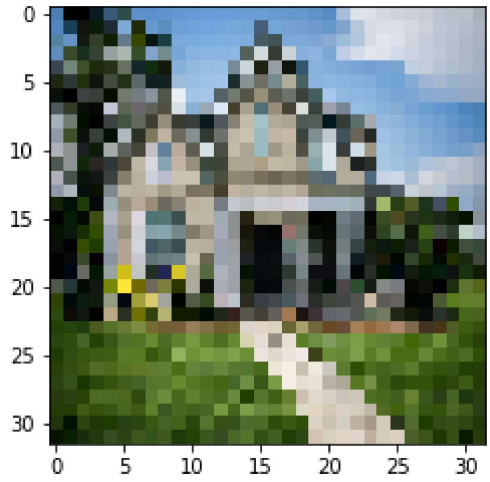
```
img = load_img('lion.jpg', target_size=(32,32))
plt.imshow(img)
img = img_to_array(img)
img = img.reshape(1,32,32,3)
img = img.astype('float32')
img = img/255
```



```
print(objects[np.argmax(model.predict(img))])
```

```
bee
```

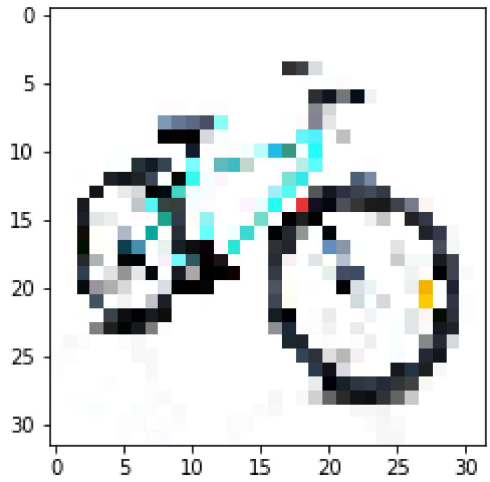
```
img = load_img('house.jpg', target_size=(32,32))
plt.imshow(img)
img = img_to_array(img)
img = img.reshape(1,32,32,3)
img = img.astype('float32')
img = img/255
```



```
print(objects[np.argmax(model.predict(img))])
```

```
house
```

```
img = load_img('bicycle.jpg', target_size=(32,32))
plt.imshow(img)
img = img_to_array(img)
img = img.reshape(1,32,32,3)
img = img.astype('float32')
img = img/255
```



```
print(objects[np.argmax(model.predict(img))])
```

```
bicycle
```

```
from google.colab import drive
drive.mount('/content/drive')
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('Cifar100_CNN.ipynb')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mour
File ‘colab_pdf.py’ already there; not retrieving.


WARNING: apt does not have a stable CLI interface. Use with caution in scripts.


WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

[NbConvertApp] Converting notebook /content/drive/MyDrive/Colab Notebooks/Cifar100_CNN.i
[NbConvertApp] Support files will be in Cifar100_CNN_files/
[NbConvertApp] Making directory ./Cifar100_CNN_files
[NbConvertApp] Making directory ./Cifar100_CNN_files
[NbConvertApp] Making directory ./Cifar100_CNN_files
[NbConvertApp] Making directory ./Cifar100_CNN_files
[NbConvertApp] Making directory ./Cifar100_CNN_files
[NbConvertApp] Writing 200280 bytes to ./notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', './notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', './notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no citatior
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 172821 bytes to /content/drive/My Drive/Cifar100_CNN.pdf
'File ready to be Downloaded and Saved to Drive'
```