1.Import Library

```
from keras.datasets import cifar
from keras.layers import Dense,Flatten,Conv2D,MaxPooling2D,Dropout
import tensorflow as tf
from keras.layers import Flatten,Dense,Conv2D,MaxPooling2D
from keras.models import Sequential
import matplotlib.pyplot as plt
import numpy as np
```

2. Import Datasets

```
(x_train, y_train), (x_test, y_test)=tf.keras.datasets.cifar10.load_data()
```

```
    Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
    170500096/170498071 [==============================] - 11s 0us/step
    170508288/170498071 [==============================] - 11s 0us/step
```

3.Data Undestanding

```
x_train=x_train.astype("float")
x_test=x_test.astype('float')
```

```
x_train=x_train/255
x_test=x_test/255
```

```
x_train.shape,x_test.shape
```

```
    ((50000, 32, 32, 3), (10000, 32, 32, 3))
```

```
y_train.shape,y_test.shape
```

```
    ((50000, 1), (10000, 1))
```

MODEL Building

```
model=Sequential()
model.add(Conv2D(input_shape=(32,32,3),kernel_size=(5,5),strides=1,activation='relu',filters=
model.add(MaxPooling2D(pool_size=2, strides=2))
model.add(Conv2D(kernel_size=(5,5),strides=1,activation='relu',filters=16))
model.add(MaxPooling2D(pool_size=2 ,strides=2))
model.add(Conv2D(kernel_size=(5,5),strides=1,activation='relu',filters=120))
model.add(Flatten())
model.add(Dense(84,activation='relu'))
model.add(Dense(1000,activation='softmax'))
model.summary()
```

```
    Model: "sequential"
    _________________________________________________________________
     Layer (type)                Output Shape              Param #
    =================================================================
     conv2d (Conv2D)             (None, 28, 28, 6)         456

     max_pooling2d (MaxPooling2D  (None, 14, 14, 6)        0
     )

     conv2d_1 (Conv2D)           (None, 10, 10, 16)        2416

     max_pooling2d_1 (MaxPooling  (None, 5, 5, 16)         0
     2D)

     conv2d_2 (Conv2D)           (None, 1, 1, 120)         48120

     flatten (Flatten)           (None, 120)               0

     dense (Dense)               (None, 84)                10164

     dense_1 (Dense)             (None, 1000)              85000

    =================================================================
    Total params: 146,156
    Trainable params: 146,156
    Non-trainable params: 0
    _________________________________________________________________
```

```
model.compile(optimizer='sgd',metrics="sparse_categorical_accuracy",loss='sparse_categorical_
```

```
model_training=model.fit(x=x_train, y=y_train, epochs=10,verbose=1,batch_size=32,validation_d
```

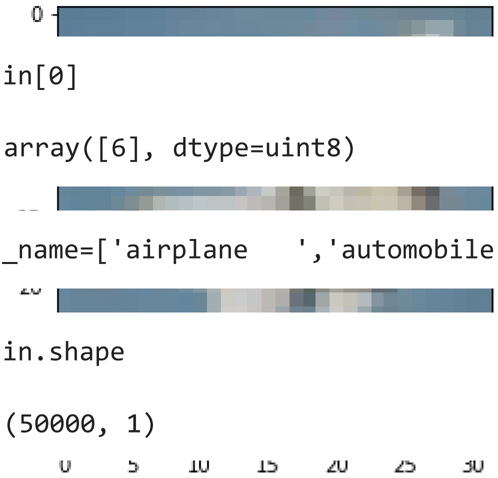
```
Epoch 1/10
1563/1563 [==============================] - 10s 7ms/step - loss: 1.1449 - sparse_catego
Epoch 2/10
1563/1563 [==============================] - 10s 7ms/step - loss: 1.1006 - sparse_catego
Epoch 3/10
1563/1563 [==============================] - 10s 6ms/step - loss: 1.0646 - sparse_catego
Epoch 4/10
1563/1563 [==============================] - 10s 7ms/step - loss: 1.0342 - sparse_catego
Epoch 5/10
1563/1563 [==============================] - 10s 7ms/step - loss: 1.0091 - sparse_catego
Epoch 6/10
1563/1563 [==============================] - 11s 7ms/step - loss: 0.9866 - sparse_catego
Epoch 7/10
1563/1563 [==============================] - 10s 7ms/step - loss: 0.9615 - sparse_catego
Epoch 8/10
1563/1563 [==============================] - 10s 7ms/step - loss: 0.9449 - sparse_catego
Epoch 9/10
1563/1563 [==============================] - 11s 7ms/step - loss: 0.9243 - sparse_catego
Epoch 10/10
1563/1563 [==============================] - 10s 7ms/step - loss: 0.9067 - sparse_catego
```

```
model.evaluate(x_test,y_test)
```

```
313/313 [==============================] - 1s 4ms/step - loss: 1.1104 - sparse_categori
[1.1104278564453125, 0.6195999979972839]
```

```
plt.imshow(x_test[10])
```

```
    <matplotlib.image.AxesImage at 0x7ff67cbd5d10>
```



```
y_train[0]
```

```
    array([6], dtype=uint8)
```

```
class_name=['airplane	','automobile	','bird','cat','deer','dog',"frog",'horse',"ship","tr
```

```
y_train.shape
```

```
    (50000, 1)
```

```
y_train=np.reshape(y_train,newshape=-1)
```

```
y_test=np.reshape(y_test,newshape=-1)
```

```
y_train.shape
```

```
    (50000,)
```

```
class_name[y_test[10]]
```

```
    'airplane\t'
```

Model Testing

```
y_pred_test=model.predict(x_test)
y_pred_train=model.predict(x_train)
```

```
y_pred_test.shape,y_pred_train.shape
```

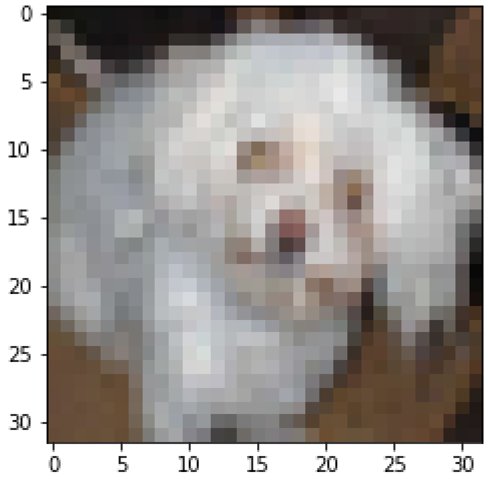
```
    ((10000, 1000), (50000, 1000))
```

```
y_pred_test.shape,y_pred_train.shape
```

```
    ((10000, 1000), (50000, 1000))
```

```
plt.imshow(x_test[1000])
```

```
    <matplotlib.image.AxesImage at 0x7ff67dca8e50>
```



```
np.argmax(y_test[1000])
```

```
    0
```

! 0s completed at 12:24 AM ● ✕