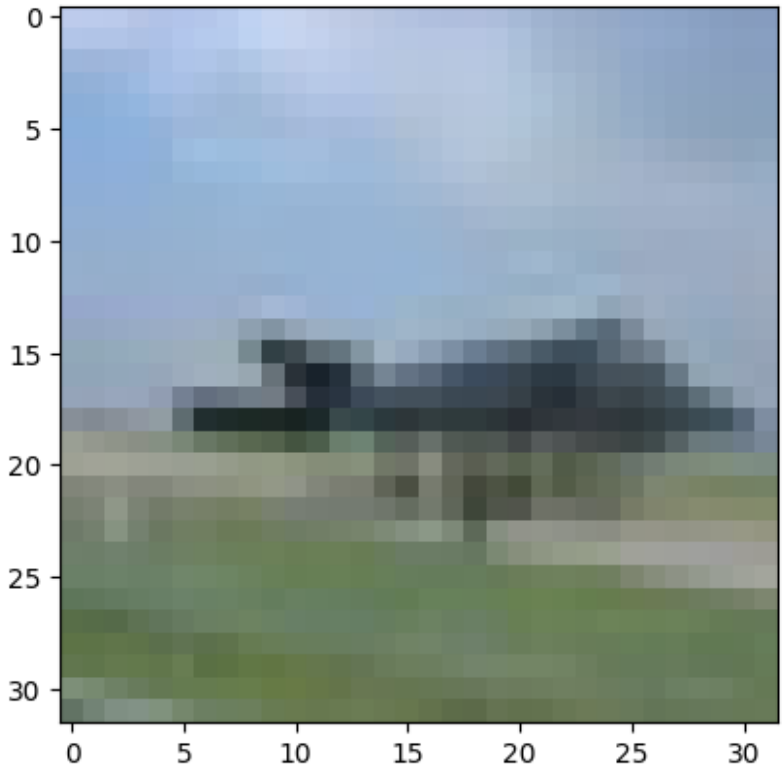
```
import pandas as pd
import numpy as np
import sklearn
import keras
import tensorflow as tf
import seaborn
import matplotlib.pyplot as plt
from tensorflow.keras import datasets
from sklearn.model_selection import train_test_split

(X_train,y_train),(X_test,y_test)=datasets.cifar10.load_data()

print("the shape of xtrain",X_train.shape)
print("the shape of ytrain",y_train.shape)
print("the shape of xtest",X_test.shape)
print("the shape of ytest",y_test.shape)
```

```
the shape of xtrain (50000, 32, 32, 3)
the shape of ytrain (50000, 1)
the shape of xtest (10000, 32, 32, 3)
the shape of ytest (10000, 1)
```

```
plt.imshow(X_train[600],cmap='gray')
plt.show()
print(y_train[600])
```



```
[0]
```

```
input_shape = (32,3,3)
epochs=15

X_train = X_train.astype("float32")/255
X_test = X_test.astype("float32")/255

X_train = np.expand_dims(X_train,-1)
X_test = np.expand_dims(X_test,-1)

print(X_train.shape)
print(X_test.shape)
```

```
(50000, 32, 32, 3, 1)
(10000, 32, 32, 3, 1)
```

```
y_train = tf.keras.utils.to_categorical(y_train,10)
y_test = tf.keras.utils.to_categorical(y_test,10)

model = tf.keras.models.Sequential ()

model.add(tf.keras.layers.Conv2D(32,
(3,3),input_shape=(32,32,3),activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2,2)))

model.add(tf.keras.layers.Conv2D(64,
(3,3),input_shape=(32,32,3),activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2,2)))

model.add(tf.keras.layers.Conv2D(128,
(3,3),input_shape=(32,32,3),activation='relu'))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2,2)))

model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(256, activation='relu'))
model.add(tf.keras.layers.Dense(128, activation='relu'))
model.add(tf.keras.layers.Dense(0, activation='softmax'))

model.summary()
```

```
Model: "sequential_14"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_32 (Conv2D)          (None, 30, 30, 32)        896

 max_pooling2d_32 (MaxPoolin  (None, 15, 15, 32)       0
 g2D)

 conv2d_33 (Conv2D)          (None, 13, 13, 64)        18496
```

```
 max_pooling2d_33 (MaxPoolin  (None, 6, 6, 64)         0
 g2D)

 conv2d_34 (Conv2D)          (None, 4, 4, 128)         73856

 max_pooling2d_34 (MaxPoolin  (None, 2, 2, 128)        0
 g2D)

 flatten_9 (Flatten)         (None, 512)               0

 dense_22 (Dense)            (None, 256)               131328

 dense_23 (Dense)            (None, 128)               32896

 dense_24 (Dense)            (None, 0)                 0

=================================================================
Total params: 257,472
Trainable params: 257,472
Non-trainable params: 0
_________________________________________________________________
```

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(50000, 32, 32, 3, 1)
(10000, 32, 32, 3, 1)
(50000, 10)
(10000, 10)
```

compilation

```
model.compile(loss='categorical_crossentropy',optimizer='Adam',metrics
=['accuracy'])

r =
model.fit(X_train,y_train,epochs=15,validation_data=(X_test,y_test),ba
tch_size=125)
```

```
Epoch 1/15

---------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
Cell In [133], line 1
----> 1 r =
model.fit(X_train,y_train,epochs=15,validation_data=(X_test,y_test),ba
```

```
tch_size=125)

File ~\AppData\Local\Programs\Python\Python39\lib\site-packages\keras\
utils\traceback_utils.py:70, in
filter_traceback.<locals>.error_handler(*args, **kwargs)
     67     filtered_tb = _process_traceback_frames(e.__traceback__)
     68     # To get the full stack trace, call:
     69     # `tf.debugging.disable_traceback_filtering()`
---> 70     raise e.with_traceback(filtered_tb) from None
     71 finally:
     72     del filtered_tb

File ~\AppData\Local\Temp\__autograph_generated_filetwxz_xsl.py:15, in
outer_factory.<locals>.inner_factory.<locals>.tf__train_function(itera
tor)
     13 try:
     14     do_return = True
---> 15     retval_ = ag__.converted_call(ag__.ld(step_function),
(ag__.ld(self), ag__.ld(iterator)), None, fscope)
     16 except:
     17     do_return = False

ValueError: in user code:

    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\engine\training.py", line 1284, in train_function
*
        return step_function(self, iterator)
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\engine\training.py", line 1268, in step_function
**
        outputs = model.distribute_strategy.run(run_step,
args=(data,))
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\engine\training.py", line 1249, in run_step  **
        outputs = model.train_step(data)
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\engine\training.py", line 1051, in train_step
        loss = self.compute_loss(x, y, y_pred, sample_weight)
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\engine\training.py", line 1109, in compute_loss
        return self.compiled_loss(
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\engine\compile_utils.py", line 265, in __call__
        loss_value = loss_obj(y_t, y_p, sample_weight=sw)
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\losses.py", line 142, in __call__
        losses = call_fn(y_true, y_pred)
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\losses.py", line 268, in call  **
```

```
        return ag_fn(y_true, y_pred, **self._fn_kwargs)
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\losses.py", line 1984, in categorical_crossentropy
        return backend.categorical_crossentropy(
    File "C:\Users\kushp\AppData\Local\Programs\Python\Python39\lib\
site-packages\keras\backend.py", line 5559, in
categorical_crossentropy
        target.shape.assert_is_compatible_with(output.shape)

    ValueError: Shapes (125, 10) and (125, 0) are incompatible
```