LAPORAN AKHIR PRAKTIKUM

Mata Praktikum	: Kecerdasan Buatan
Kelas	: 3IA13
Praktikum ke-	: 2
Tanggal	: 6 Desember 2022
Materi	: Image Recognition dan Convolutional Neural Networks
NPM	: 51420249
Nama	: Tiara Puspita
Ketua Asisten	: David
Jumlah Lembar	: 5

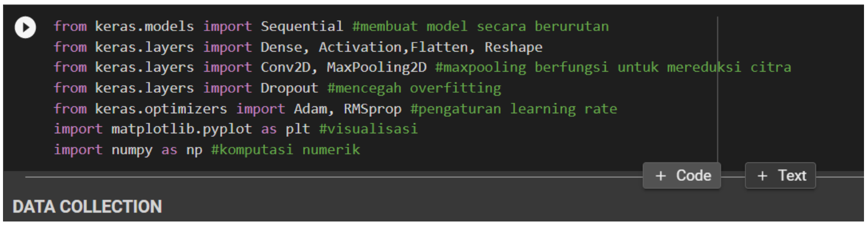
LABORATORIUM TEKNIK INFORMATIKA

UNIVERSITAS GUNADARMA

2022

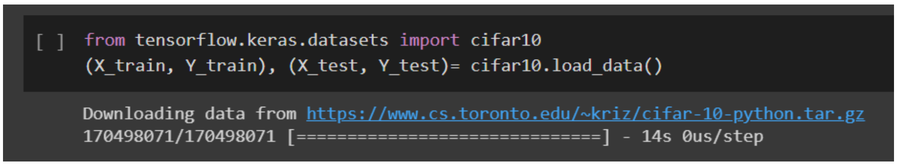
Isi Laporan

1. Blok kode yang digunakan untuk mengimport library yang dibutuhkan



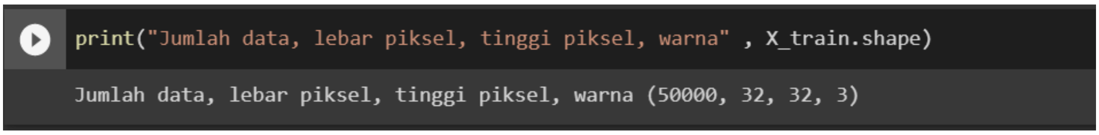
```
from keras.models import Sequential #membuat model secara berurutan
from keras.layers import Dense, Activation,Flatten, Reshape
from keras.layers import Conv2D, MaxPooling2D #maxpooling berfungsi untuk mereduksi citra
from keras.layers import Dropout #mencegah overfitting
from keras.optimizers import Adam, RMSprop #pengaturan learning rate
import matplotlib.pyplot as plt #visualisasi
import numpy as np #komputasi numerik
```

2. Load data dari tensorflow keras



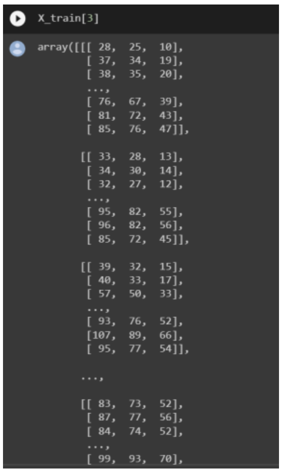
```
[ ] from tensorflow.keras.datasets import cifar10
    (X_train, Y_train), (X_test, Y_test)= cifar10.load_data()

    Downloading data from https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
    170498071/170498071 [==============================] - 14s 0us/step
```

3. Menampilkan dimensi dari data x_train

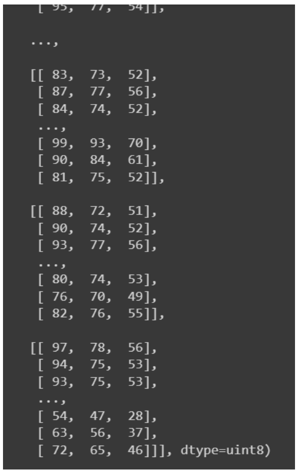


```
print("Jumlah data, lebar piksel, tinggi piksel, warna" , X_train.shape)

Jumlah data, lebar piksel, tinggi piksel, warna (50000, 32, 32, 3)
```

4. Push data x_train dengan index 3

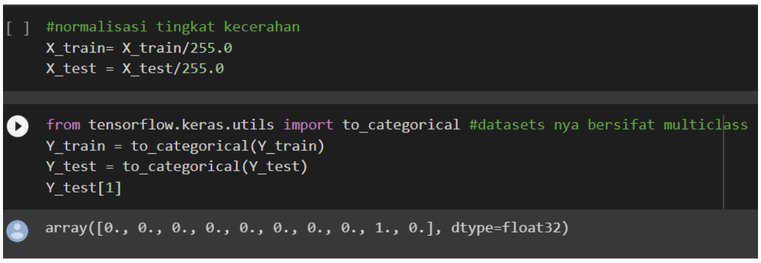


```
X_train[3]

array([[[ 28,  25,  10],
        [ 37,  34,  19],
        [ 38,  35,  20],
        ...,
        [ 76,  67,  39],
        [ 81,  72,  43],
        [ 85,  76,  47]],

       [[ 33,  28,  13],
        [ 34,  30,  14],
        [ 32,  27,  12],
        ...,
        [ 95,  82,  55],
        [ 96,  82,  56],
        [ 85,  72,  45]],

       [[ 39,  32,  15],
        [ 40,  33,  17],
        [ 57,  50,  33],
        ...,
        [ 93,  76,  52],
        [107,  89,  66],
        [ 95,  77,  54]],

       ...,

       [[ 83,  73,  52],
        [ 87,  77,  56],
        [ 84,  74,  52],
        ...,
        [ 99,  93,  70],
```



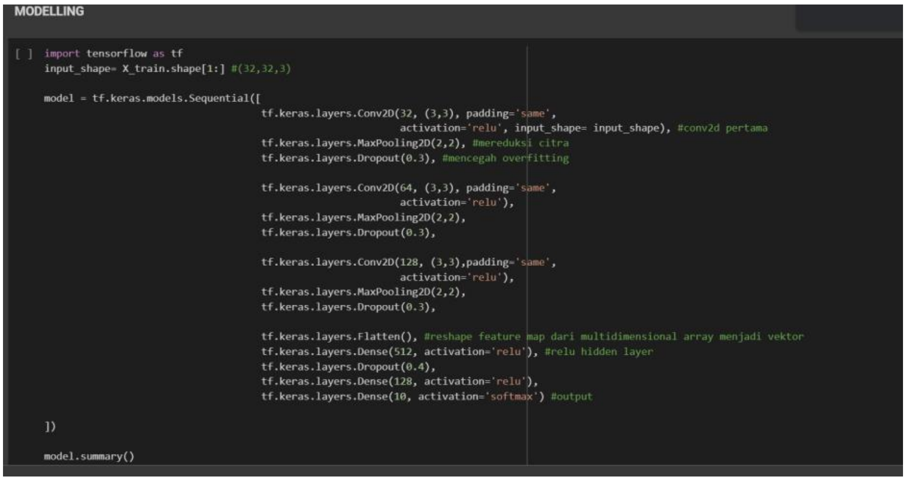
```
        [ 95,  77,  54]],

       ...,

       [[ 83,  73,  52],
        [ 87,  77,  56],
        [ 84,  74,  52],
        ...,
        [ 99,  93,  70],
        [ 90,  84,  61],
        [ 81,  75,  52]],

       [[ 88,  72,  51],
        [ 90,  74,  52],
        [ 93,  77,  56],
        ...,
        [ 80,  74,  53],
        [ 76,  70,  49],
        [ 82,  76,  55]],

       [[ 97,  78,  56],
        [ 94,  75,  53],
        [ 93,  75,  53],
        ...,
        [ 54,  47,  28],
        [ 63,  56,  37],
        [ 72,  65,  46]]], dtype=uint8)
```

5. Data preprocecing data dengan menggunakan to_categorical yang di import dari python keras.



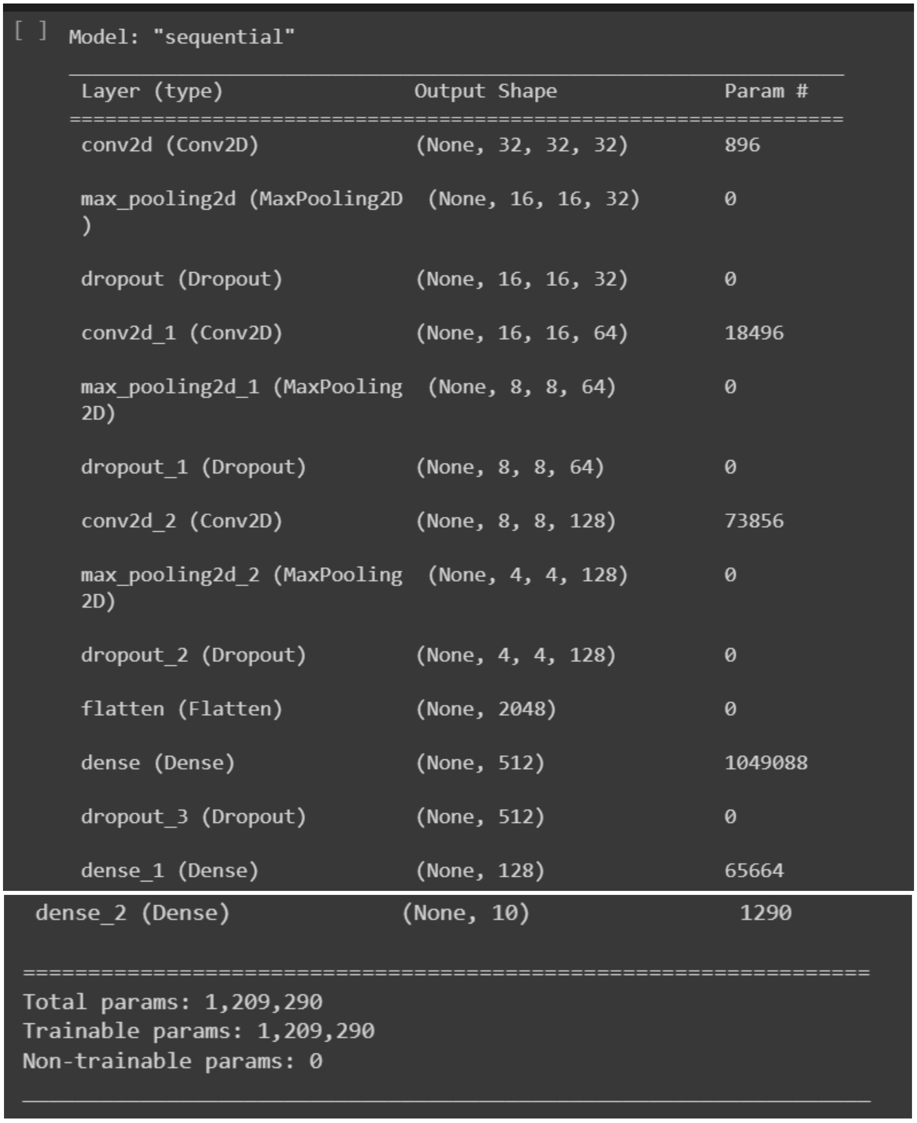
```
[ ] #normalisasi tingkat kecerahan
    X_train= X_train/255.0
    X_test = X_test/255.0

    from tensorflow.keras.utils import to_categorical #datasets nya bersifat multiclass
    Y_train = to_categorical(Y_train)
    Y_test = to_categorical(Y_test)
    Y_test[1]

    array([0., 0., 0., 0., 0., 0., 0., 0., 1., 0.], dtype=float32)
```

6. Data Modelling dengan tensorflow

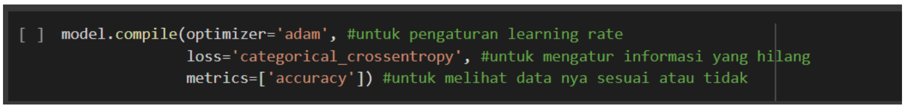


MODELLING

```
[ ] import tensorflow as tf
    input_shape= X_train.shape[1:] #(32,32,3)

    model = tf.keras.models.Sequential([
                                    tf.keras.layers.Conv2D(32, (3,3), padding='same',
                                                           activation='relu', input_shape= input_shape), #conv2d pertama
                                    tf.keras.layers.MaxPooling2D(2,2), #mereduksi citra
                                    tf.keras.layers.Dropout(0.3), #mencegah overfitting

                                    tf.keras.layers.Conv2D(64, (3,3), padding='same',
                                                           activation='relu'),
                                    tf.keras.layers.MaxPooling2D(2,2),
                                    tf.keras.layers.Dropout(0.3),

                                    tf.keras.layers.Conv2D(128, (3,3),padding='same',
                                                           activation='relu'),
                                    tf.keras.layers.MaxPooling2D(2,2),
                                    tf.keras.layers.Dropout(0.3),

                                    tf.keras.layers.Flatten(), #reshape feature map dari multidimensional array menjadi vektor
                                    tf.keras.layers.Dense(512, activation='relu'), #relu hidden layer
                                    tf.keras.layers.Dropout(0.4),
                                    tf.keras.layers.Dense(128, activation='relu'),
                                    tf.keras.layers.Dense(10, activation='softmax') #output

    ])

    model.summary()
```

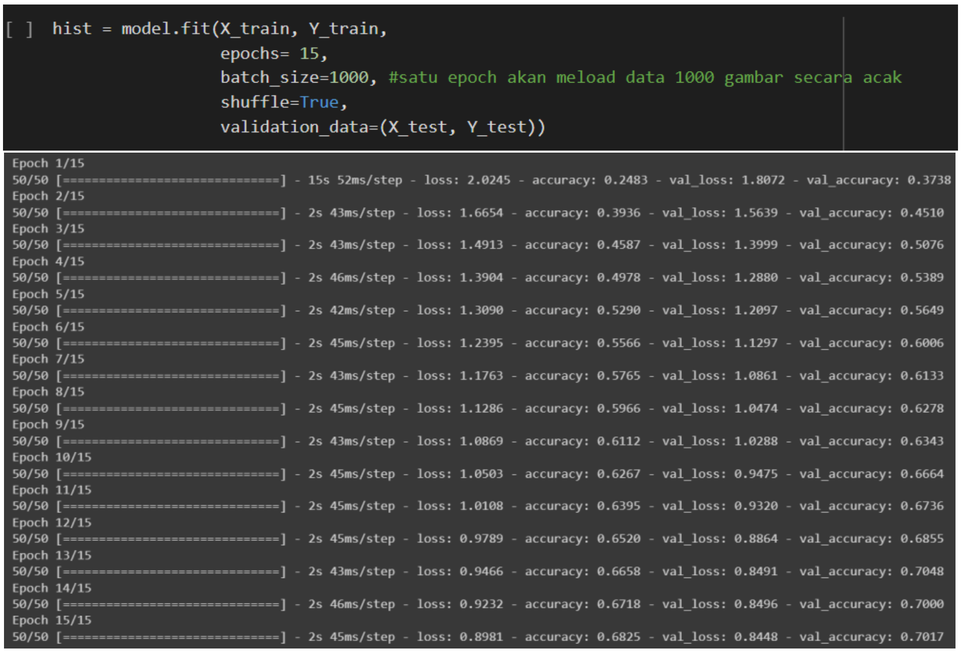


```
[ ] Model: "sequential"
    _________________________________________________________________
     Layer (type)                Output Shape              Param #
    =================================================================
     conv2d (Conv2D)             (None, 32, 32, 32)        896

     max_pooling2d (MaxPooling2D  (None, 16, 16, 32)       0
     )

     dropout (Dropout)           (None, 16, 16, 32)        0

     conv2d_1 (Conv2D)           (None, 16, 16, 64)        18496

     max_pooling2d_1 (MaxPooling  (None, 8, 8, 64)         0
     2D)

     dropout_1 (Dropout)         (None, 8, 8, 64)          0

     conv2d_2 (Conv2D)           (None, 8, 8, 128)         73856

     max_pooling2d_2 (MaxPooling  (None, 4, 4, 128)        0
     2D)

     dropout_2 (Dropout)         (None, 4, 4, 128)         0

     flatten (Flatten)           (None, 2048)              0

     dense (Dense)               (None, 512)               1049088

     dropout_3 (Dropout)         (None, 512)               0

     dense_1 (Dense)             (None, 128)               65664

     dense_2 (Dense)             (None, 10)                1290

    =================================================================
    Total params: 1,209,290
    Trainable params: 1,209,290
    Non-trainable params: 0
    _________________________________________________________________
```



```
[ ] model.compile(optimizer='adam', #untuk pengaturan learning rate
              loss='categorical_crossentropy', #untuk mengatur informasi yang hilang
              metrics=['accuracy']) #untuk melihat data nya sesuai atau tidak
```

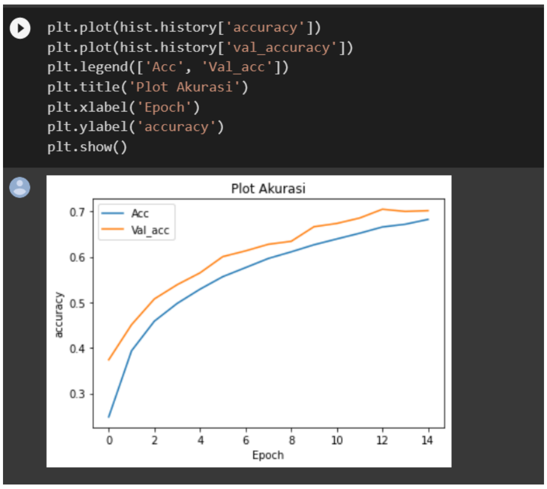
7. Training Data



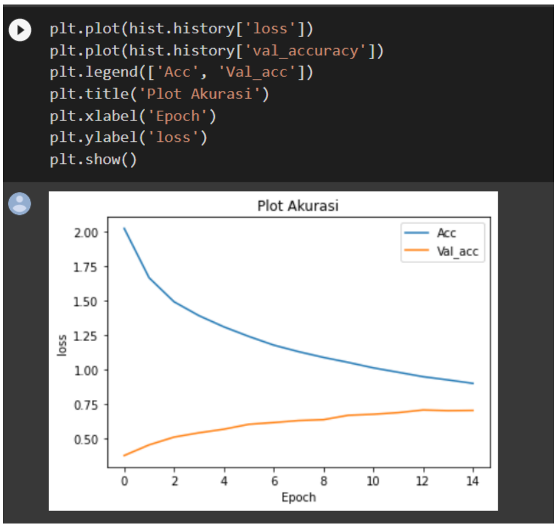
```
[ ]  hist = model.fit(X_train, Y_train,
                      epochs= 15,
                      batch_size=1000, #satu epoch akan meload data 1000 gambar secara acak
                      shuffle=True,
                      validation_data=(X_test, Y_test))
```

```
Epoch 1/15
50/50 [==============================] - 15s 52ms/step - loss: 2.0245 - accuracy: 0.2483 - val_loss: 1.8072 - val_accuracy: 0.3738
Epoch 2/15
50/50 [==============================] - 2s 43ms/step - loss: 1.6654 - accuracy: 0.3936 - val_loss: 1.5639 - val_accuracy: 0.4510
Epoch 3/15
50/50 [==============================] - 2s 43ms/step - loss: 1.4913 - accuracy: 0.4587 - val_loss: 1.3999 - val_accuracy: 0.5076
Epoch 4/15
50/50 [==============================] - 2s 46ms/step - loss: 1.3904 - accuracy: 0.4978 - val_loss: 1.2880 - val_accuracy: 0.5389
Epoch 5/15
50/50 [==============================] - 2s 42ms/step - loss: 1.3090 - accuracy: 0.5290 - val_loss: 1.2097 - val_accuracy: 0.5649
Epoch 6/15
50/50 [==============================] - 2s 45ms/step - loss: 1.2395 - accuracy: 0.5566 - val_loss: 1.1297 - val_accuracy: 0.6006
Epoch 7/15
50/50 [==============================] - 2s 43ms/step - loss: 1.1763 - accuracy: 0.5765 - val_loss: 1.0861 - val_accuracy: 0.6133
Epoch 8/15
50/50 [==============================] - 2s 45ms/step - loss: 1.1286 - accuracy: 0.5966 - val_loss: 1.0474 - val_accuracy: 0.6278
Epoch 9/15
50/50 [==============================] - 2s 43ms/step - loss: 1.0869 - accuracy: 0.6112 - val_loss: 1.0288 - val_accuracy: 0.6343
Epoch 10/15
50/50 [==============================] - 2s 45ms/step - loss: 1.0503 - accuracy: 0.6267 - val_loss: 0.9475 - val_accuracy: 0.6664
Epoch 11/15
50/50 [==============================] - 2s 45ms/step - loss: 1.0108 - accuracy: 0.6395 - val_loss: 0.9320 - val_accuracy: 0.6736
Epoch 12/15
50/50 [==============================] - 2s 45ms/step - loss: 0.9789 - accuracy: 0.6520 - val_loss: 0.8864 - val_accuracy: 0.6855
Epoch 13/15
50/50 [==============================] - 2s 43ms/step - loss: 0.9466 - accuracy: 0.6658 - val_loss: 0.8491 - val_accuracy: 0.7048
Epoch 14/15
50/50 [==============================] - 2s 46ms/step - loss: 0.9232 - accuracy: 0.6718 - val_loss: 0.8496 - val_accuracy: 0.7000
Epoch 15/15
50/50 [==============================] - 2s 45ms/step - loss: 0.8981 - accuracy: 0.6825 - val_loss: 0.8448 - val_accuracy: 0.7017
```

8. Visualisasikan data dengan line plot seperti gambar dibawah ini

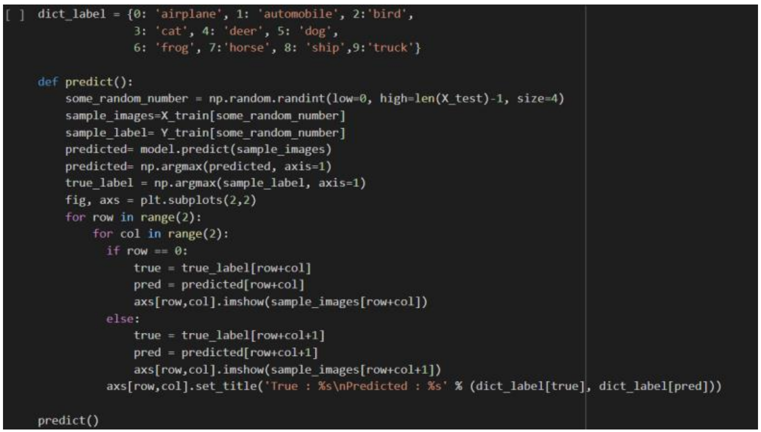


```
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.legend(['Acc', 'Val_acc'])
plt.title('Plot Akurasi')
plt.xlabel('Epoch')
plt.ylabel('accuracy')
plt.show()
```

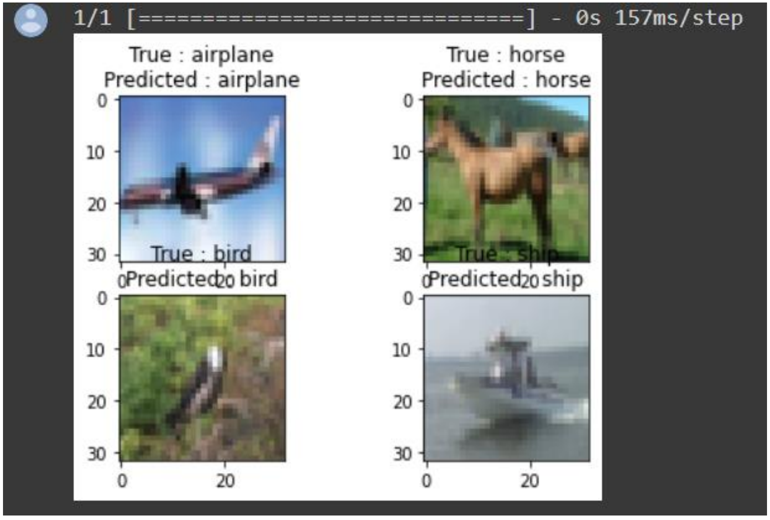


```
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_accuracy'])
plt.legend(['Acc', 'Val_acc'])
plt.title('Plot Akurasi')
plt.xlabel('Epoch')
plt.ylabel('loss')
plt.show()
```

9. Melakukan prediksi



```
[ ] dict_label = {0: 'airplane', 1: 'automobile', 2:'bird',
                  3: 'cat', 4: 'deer', 5: 'dog',
                  6: 'frog', 7:'horse', 8: 'ship',9:'truck'}

    def predict():
        some_random_number = np.random.randint(low=0, high=len(X_test)-1, size=4)
        sample_images=X_train[some_random_number]
        sample_label= Y_train[some_random_number]
        predicted= model.predict(sample_images)
        predicted= np.argmax(predicted, axis=1)
        true_label = np.argmax(sample_label, axis=1)
        fig, axs = plt.subplots(2,2)
        for row in range(2):
            for col in range(2):
              if row == 0:
                  true = true_label[row+col]
                  pred = predicted[row+col]
                  axs[row,col].imshow(sample_images[row+col])
              else:
                  true = true_label[row+col+1]
                  pred = predicted[row+col+1]
                  axs[row,col].imshow(sample_images[row+col+1])
              axs[row,col].set_title('True : %s\nPredicted : %s' % (dict_label[true], dict_label[pred]))

    predict()
```



```
1/1 [==============================] - 0s 157ms/step
```