SAKARYA ÜNİVERSİTESİ

HAŞİM GÜRDAMAR BİLGİSAYAR BİLİŞİM BİLİMLERİ FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞ

DERİN ÖĞRENME VE ERİŞİMLİ SİNİR AĞLARI

Ödev 2

Ad-soyad: Wajeeh Albasha

Öğrenci no: G181210552

1.Eğitimde Kullanılan Sınıflar :

beetle:

can:

house:

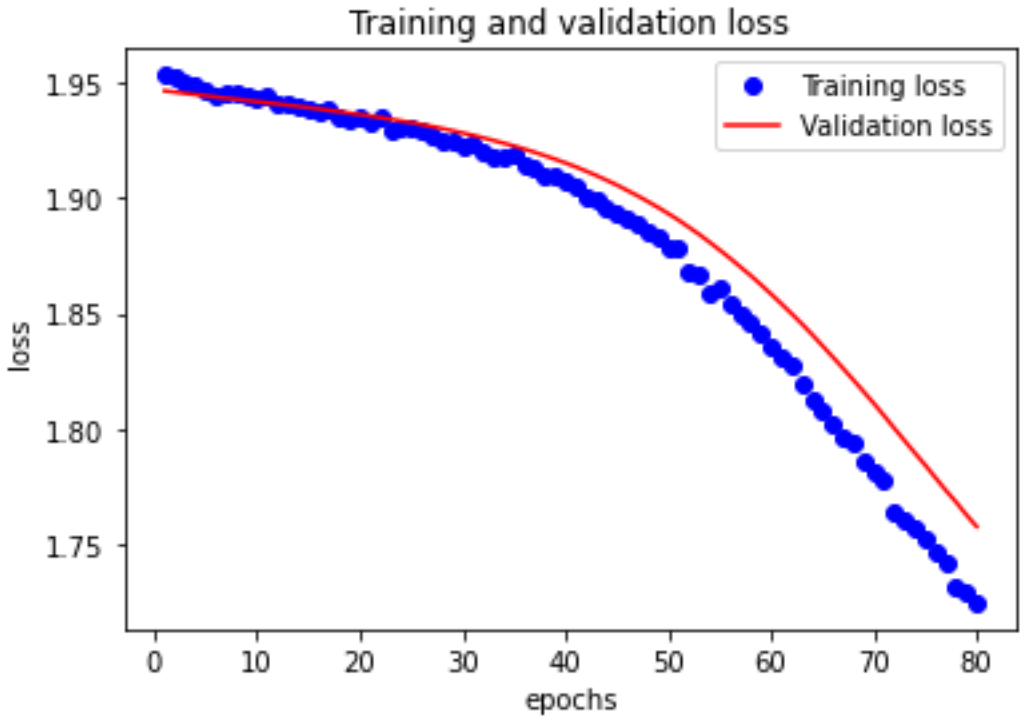
mouse:

pickup_truck:

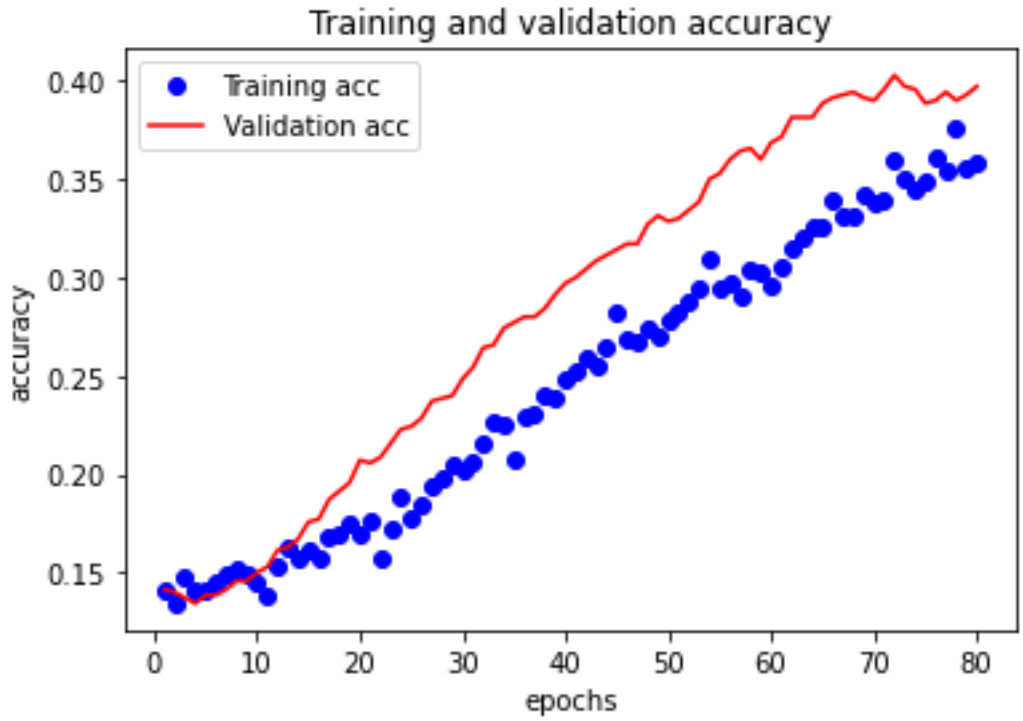
shark:

sweet_pepper:

Kayıp Grafiği:



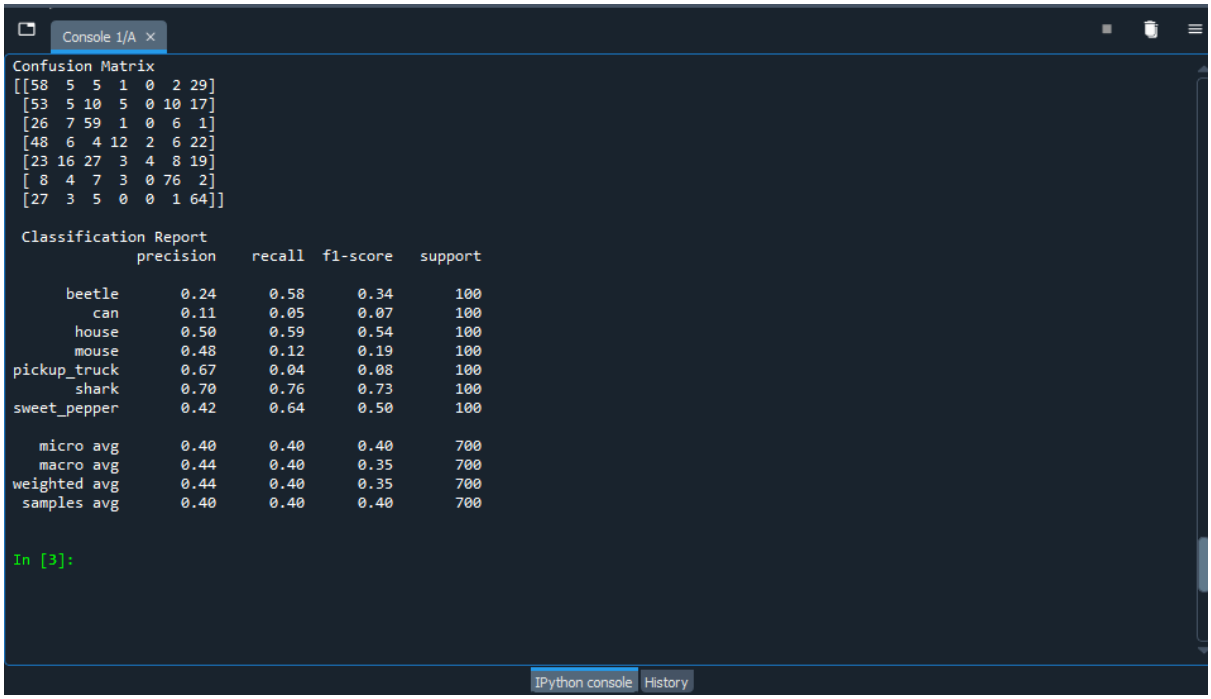
Accuracy Grafiği:



Epoch, Loss ve Accuracy:

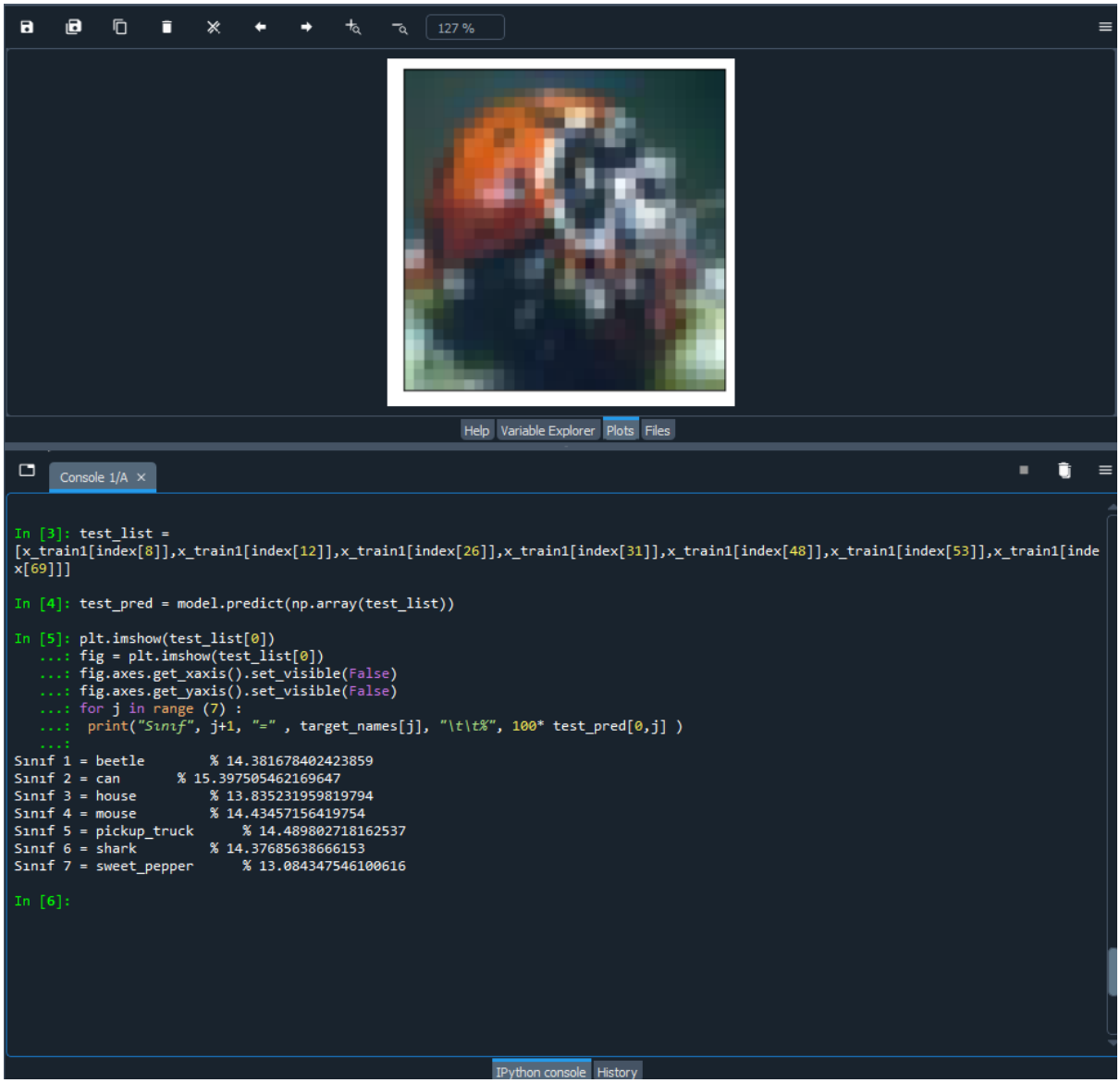
```
Console 1/A
110/110 [==============================] - 6s 53ms/step - loss: 1.8022 - acc: 0.3391 - val_loss: 1.8311 - val_acc: 0.3914
Epoch 67/80
110/110 [==============================] - 6s 54ms/step - loss: 1.7969 - acc: 0.3309 - val_loss: 1.8262 - val_acc: 0.3929
Epoch 68/80
110/110 [==============================] - 6s 53ms/step - loss: 1.7936 - acc: 0.3309 - val_loss: 1.8212 - val_acc: 0.3943
Epoch 69/80
110/110 [==============================] - 6s 55ms/step - loss: 1.7858 - acc: 0.3420 - val_loss: 1.8162 - val_acc: 0.3914
Epoch 70/80
110/110 [==============================] - 6s 54ms/step - loss: 1.7808 - acc: 0.3386 - val_loss: 1.8110 - val_acc: 0.3900
Epoch 71/80
110/110 [==============================] - 6s 53ms/step - loss: 1.7774 - acc: 0.3397 - val_loss: 1.8057 - val_acc: 0.3957
Epoch 72/80
110/110 [==============================] - 6s 53ms/step - loss: 1.7640 - acc: 0.3600 - val_loss: 1.8002 - val_acc: 0.4029
Epoch 73/80
110/110 [==============================] - 6s 52ms/step - loss: 1.7601 - acc: 0.3506 - val_loss: 1.7949 - val_acc: 0.3971
Epoch 74/80
110/110 [==============================] - 6s 52ms/step - loss: 1.7575 - acc: 0.3446 - val_loss: 1.7895 - val_acc: 0.3957
Epoch 75/80
110/110 [==============================] - 6s 53ms/step - loss: 1.7521 - acc: 0.3491 - val_loss: 1.7843 - val_acc: 0.3886
Epoch 76/80
110/110 [==============================] - 6s 52ms/step - loss: 1.7467 - acc: 0.3611 - val_loss: 1.7788 - val_acc: 0.3900
Epoch 77/80
110/110 [==============================] - 6s 52ms/step - loss: 1.7418 - acc: 0.3540 - val_loss: 1.7734 - val_acc: 0.3943
Epoch 78/80
110/110 [==============================] - 6s 53ms/step - loss: 1.7313 - acc: 0.3763 - val_loss: 1.7683 - val_acc: 0.3900
Epoch 79/80
110/110 [==============================] - 6s 54ms/step - loss: 1.7288 - acc: 0.3557 - val_loss: 1.7628 - val_acc: 0.3929
Epoch 80/80
110/110 [==============================] - 6s 52ms/step - loss: 1.7248 - acc: 0.3589 - val_loss: 1.7577 - val_acc: 0.3971
22/22 [==============================] - 0s 13ms/step - loss: 1.7577 - acc: 0.3971
 Loss = % 175.77122449874878
 Accuracy = % 39.71428573131561
IPython console  History
```

Confusion Matrix :

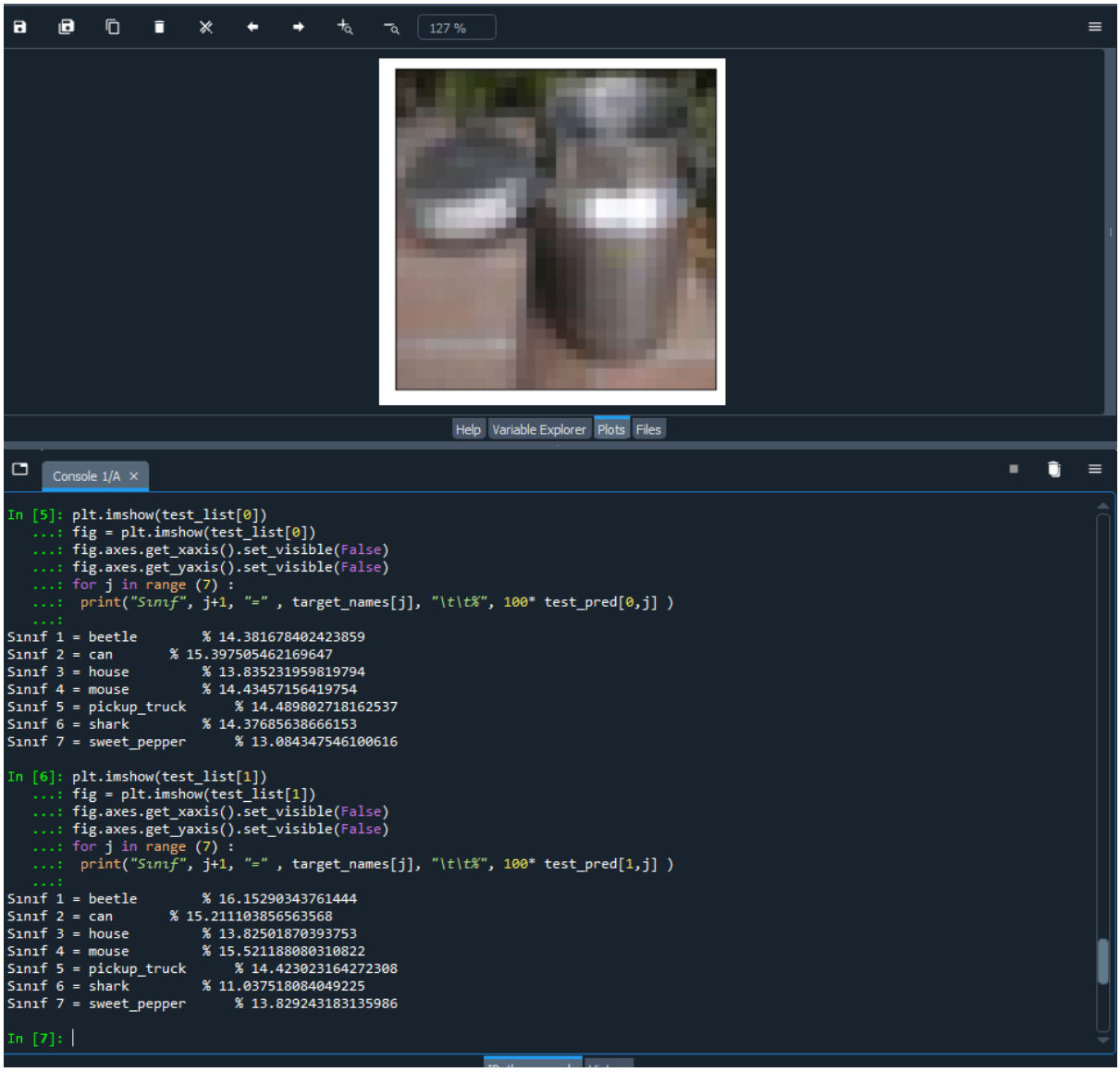


Sınıf örnekleri:

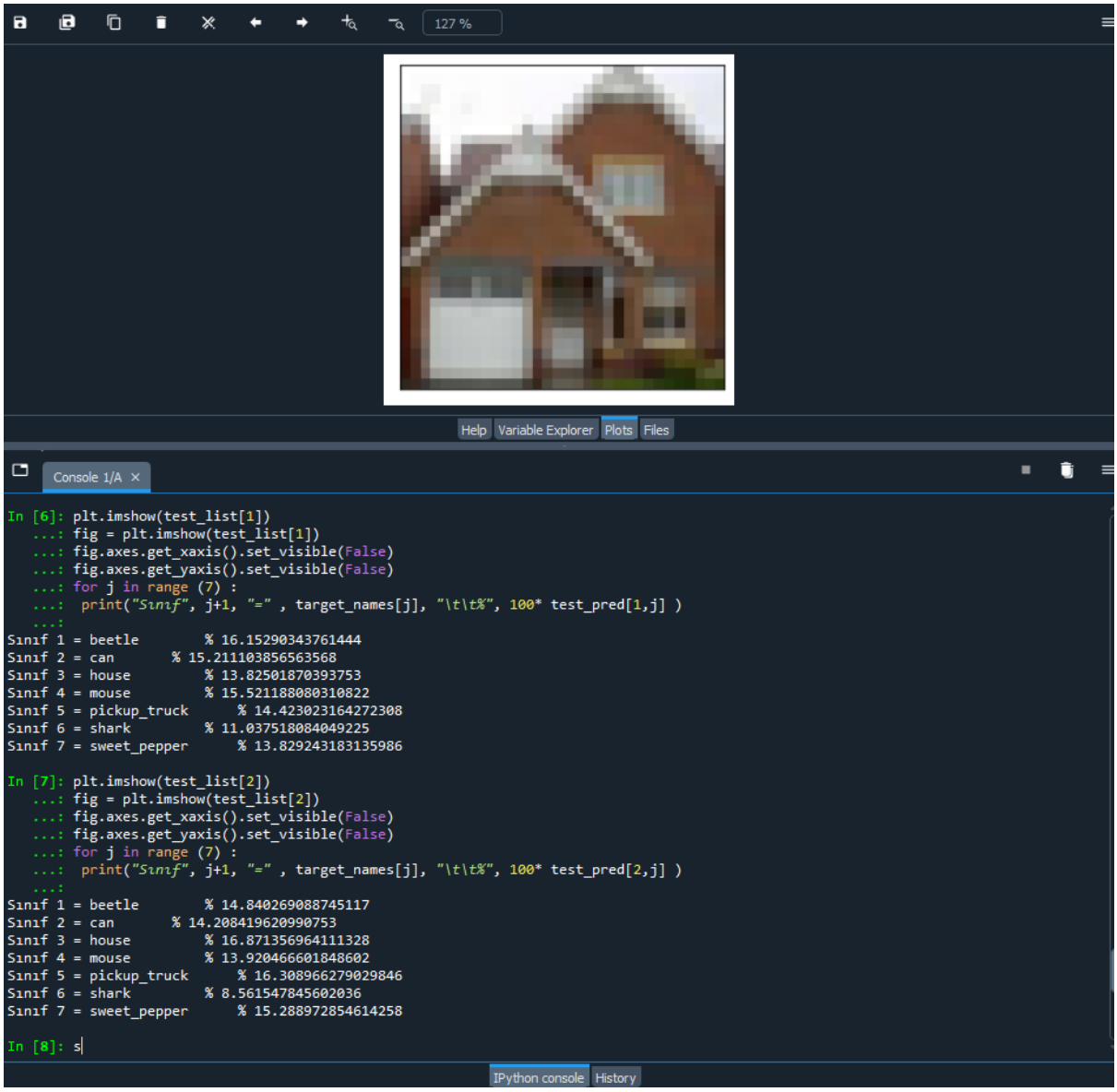
Beetle :


127 %
Help
Variable Explorer
Plots
Files
Console 1/A
In [3]: test_list =
[x_train1[index[8]],x_train1[index[12]],x_train1[index[26]],x_train1[index[31]],x_train1[index[48]],x_train1[index[53]],x_train1[index[69]]]
In [4]: test_pred = model.predict(np.array(test_list))
In [5]: plt.imshow(test_list[0])
...: fig = plt.imshow(test_list[0])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[0,j])
...:
Sınıf 1 = beetle % 14.381678402423859
Sınıf 2 = can % 15.397505462169647
Sınıf 3 = house % 13.835231959819794
Sınıf 4 = mouse % 14.43457156419754
Sınıf 5 = pickup_truck % 14.489802718162537
Sınıf 6 = shark % 14.37685638666153
Sınıf 7 = sweet_pepper % 13.084347546100616
In [6]:
IPython console
History

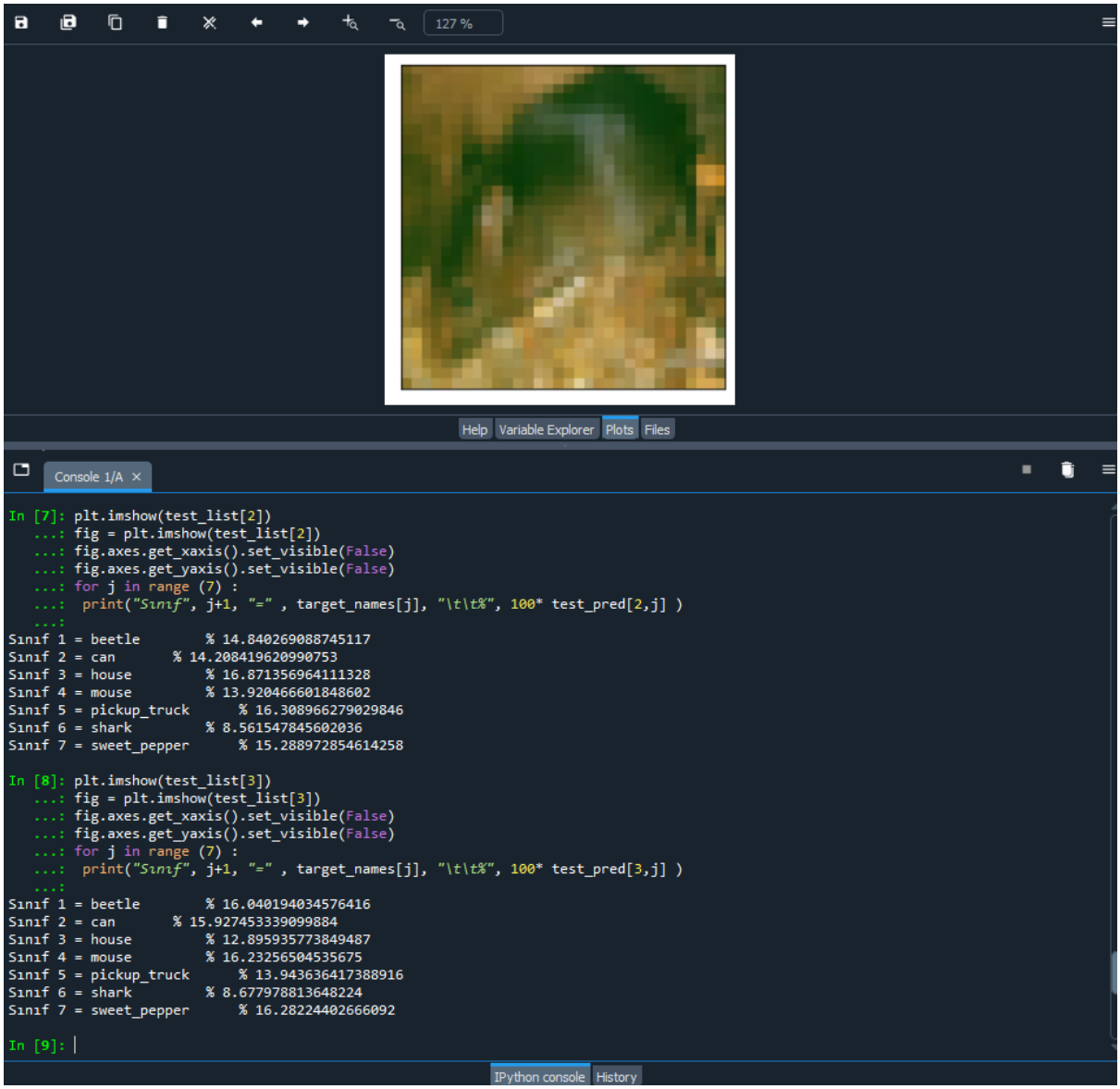
Can :


127 %
Help
Variable Explorer
Plots
Files
Console 1/A
In [5]: plt.imshow(test_list[0])
...: fig = plt.imshow(test_list[0])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[0,j])
...:
Sınıf 1 = beetle % 14.381678402423859
Sınıf 2 = can % 15.397505462169647
Sınıf 3 = house % 13.835231959819794
Sınıf 4 = mouse % 14.43457156419754
Sınıf 5 = pickup_truck % 14.489802718162537
Sınıf 6 = shark % 14.37685638666153
Sınıf 7 = sweet_pepper % 13.084347546100616
In [6]: plt.imshow(test_list[1])
...: fig = plt.imshow(test_list[1])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[1,j])
...:
Sınıf 1 = beetle % 16.15290343761444
Sınıf 2 = can % 15.211103856563568
Sınıf 3 = house % 13.82501870393753
Sınıf 4 = mouse % 15.521188080310822
Sınıf 5 = pickup_truck % 14.423023164272308
Sınıf 6 = shark % 11.037518084049225
Sınıf 7 = sweet_pepper % 13.829243183135986
In [7]:

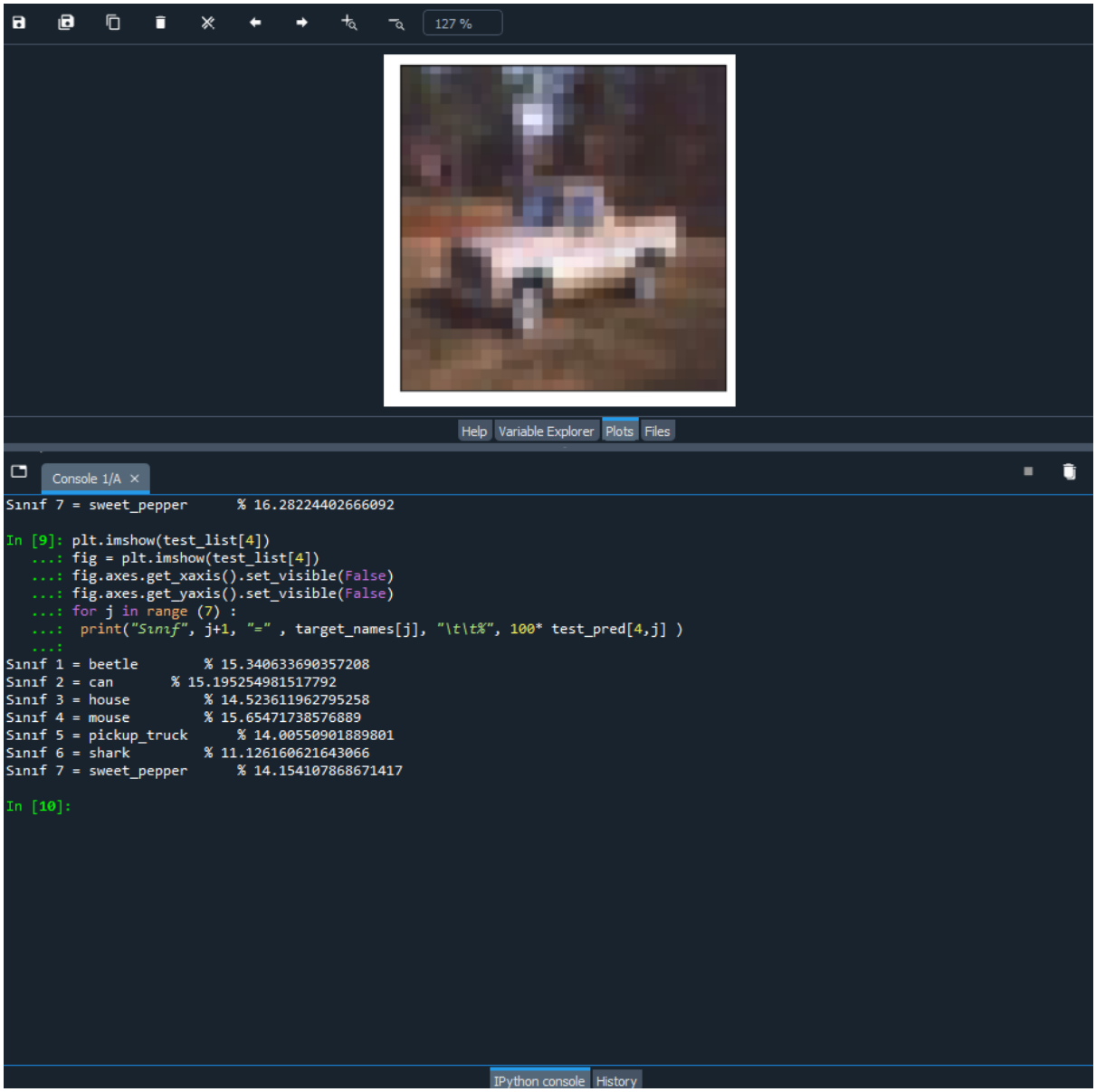
House :


127 %
Help
Variable Explorer
Plots
Files
Console 1/A
In [6]: plt.imshow(test_list[1])
...: fig = plt.imshow(test_list[1])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[1,j])
...:
Sınıf 1 = beetle % 16.15290343761444
Sınıf 2 = can % 15.211103856563568
Sınıf 3 = house % 13.82501870393753
Sınıf 4 = mouse % 15.521188080310822
Sınıf 5 = pickup_truck % 14.423023164272308
Sınıf 6 = shark % 11.037518084049225
Sınıf 7 = sweet_pepper % 13.829243183135986
In [7]: plt.imshow(test_list[2])
...: fig = plt.imshow(test_list[2])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[2,j])
...:
Sınıf 1 = beetle % 14.840269088745117
Sınıf 2 = can % 14.208419620990753
Sınıf 3 = house % 16.871356964111328
Sınıf 4 = mouse % 13.920466601848602
Sınıf 5 = pickup_truck % 16.308966279029846
Sınıf 6 = shark % 8.561547845602036
Sınıf 7 = sweet_pepper % 15.288972854614258
In [8]: s
IPython console
History

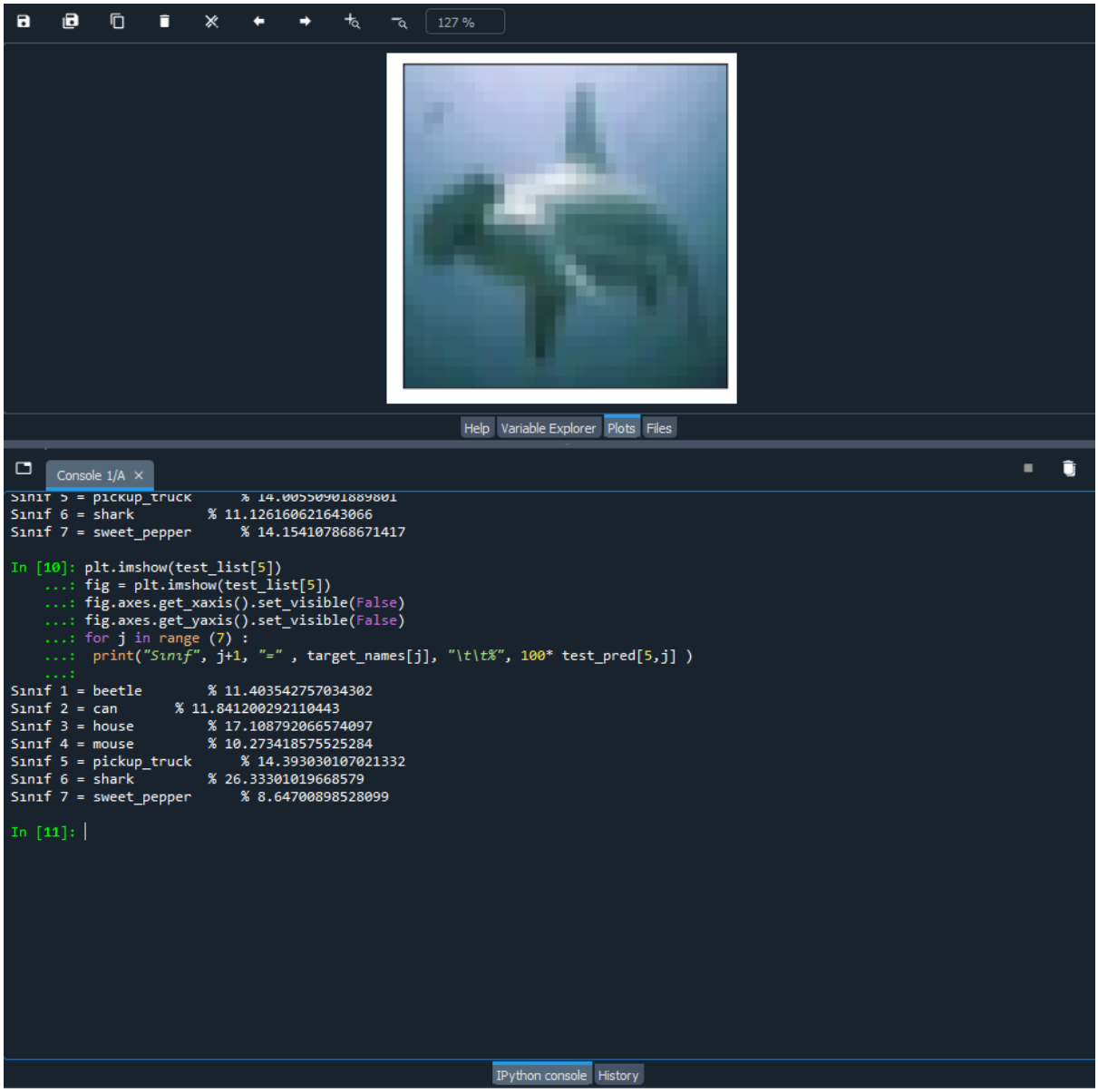
Mouse :


127 %
Help
Variable Explorer
Plots
Files
Console 1/A
In [7]: plt.imshow(test_list[2])
...: fig = plt.imshow(test_list[2])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[2,j])
...:
Sınıf 1 = beetle % 14.840269088745117
Sınıf 2 = can % 14.208419620990753
Sınıf 3 = house % 16.871356964111328
Sınıf 4 = mouse % 13.920466601848602
Sınıf 5 = pickup_truck % 16.308966279029846
Sınıf 6 = shark % 8.561547845602036
Sınıf 7 = sweet_pepper % 15.288972854614258
In [8]: plt.imshow(test_list[3])
...: fig = plt.imshow(test_list[3])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[3,j])
...:
Sınıf 1 = beetle % 16.040194034576416
Sınıf 2 = can % 15.927453339099884
Sınıf 3 = house % 12.895935773849487
Sınıf 4 = mouse % 16.23256504535675
Sınıf 5 = pickup_truck % 13.943636417388916
Sınıf 6 = shark % 8.677978813648224
Sınıf 7 = sweet_pepper % 16.28224402666092
In [9]:
IPython console
History

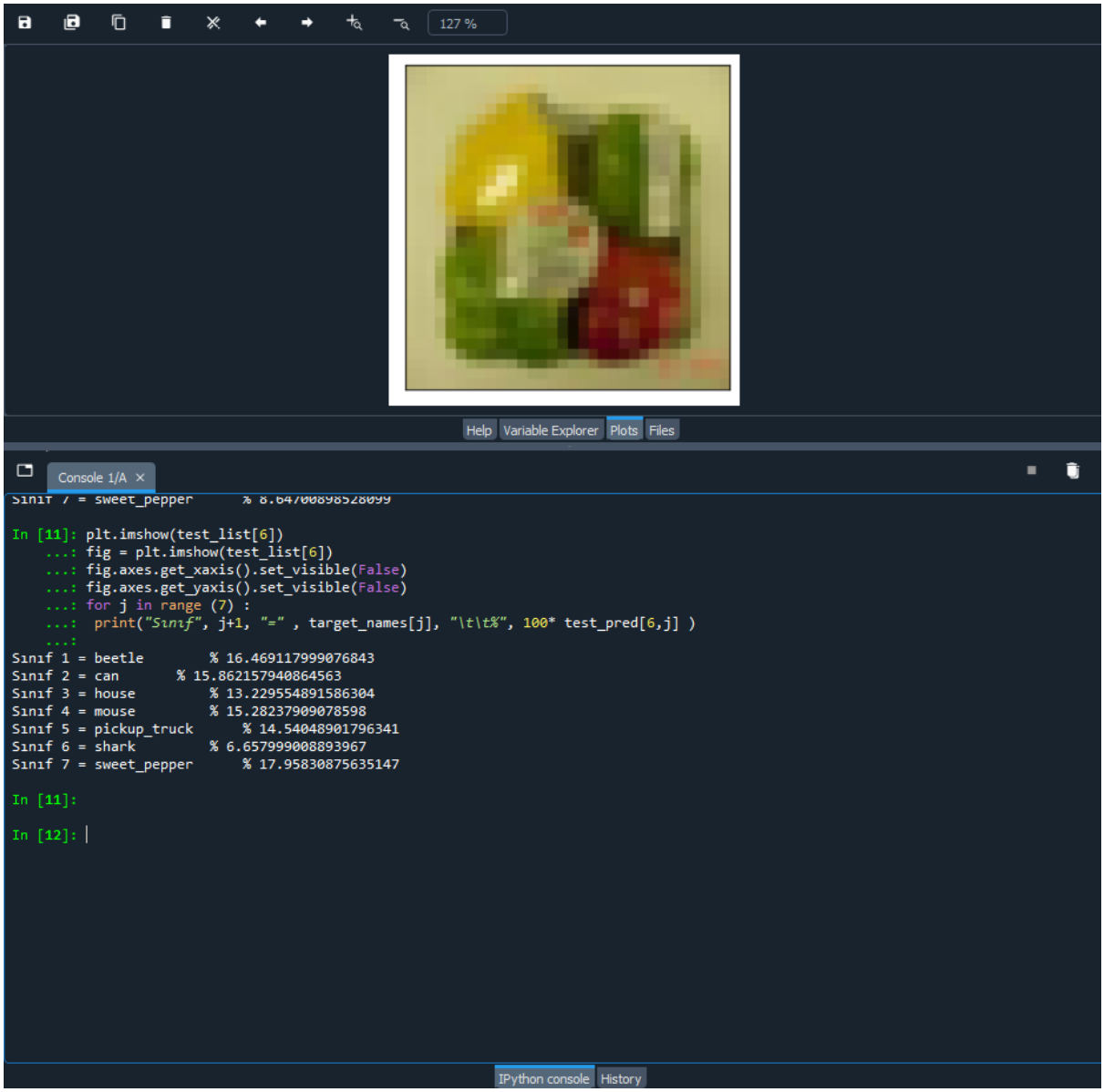
pickup_truck :



Shark :


127 %
Help
Variable Explorer
Plots
Files
Console 1/A
Sınıf 5 = pickup_truck % 14.005509018889801
Sınıf 6 = shark % 11.126160621643066
Sınıf 7 = sweet_pepper % 14.154107868671417
In [10]: plt.imshow(test_list[5])
...: fig = plt.imshow(test_list[5])
...: fig.axes.get_xaxis().set_visible(False)
...: fig.axes.get_yaxis().set_visible(False)
...: for j in range (7) :
...: print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100* test_pred[5,j])
...:
Sınıf 1 = beetle % 11.403542757034302
Sınıf 2 = can % 11.841200292110443
Sınıf 3 = house % 17.108792066574097
Sınıf 4 = mouse % 10.273418575525284
Sınıf 5 = pickup_truck % 14.393030107021332
Sınıf 6 = shark % 26.33301019668579
Sınıf 7 = sweet_pepper % 8.64700898528099
In [11]:
IPython console
History

sweet_pepper :



Kynak Kodu(.py dosyası) :

```
# -*- coding: utf-8 -*-

from keras.datasets import cifar100
from keras.models import Sequential
from keras import layers
import matplotlib.pyplot as plt
from tensorflow.keras.utils import to_categorical
import numpy as np

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

y_train1 = []
x_train1 = []
y_test1 = []
x_test1 = []

for i in range (50000):
    if y_train[i] == 7 :
        y_train1.append(0)
        x_train1.append(x_train[i])
```

```
elif  y_train[i] == 16 :
    y_train1.append(1)
    x_train1.append(x_train[i])

elif  y_train[i] == 37 :
    y_train1.append(2)
    x_train1.append(x_train[i])

elif  y_train[i] == 50 :
    y_train1.append(3)
    x_train1.append(x_train[i])

elif  y_train[i] == 58 :
    y_train1.append(4)
    x_train1.append(x_train[i])

elif  y_train[i] == 73 :
    y_train1.append(5)
    x_train1.append(x_train[i])

elif  y_train[i] == 83 :
    y_train1.append(6)
    x_train1.append(x_train[i])
```

```
y_train1= np.array(y_train1)
x_train1= np.array(x_train1)

for i in range (10000):
    if y_test[i] == 7 :
        y_test1.append(0)
        x_test1.append(x_test[i])

    elif  y_test[i] == 16 :
        y_test1.append(1)
        x_test1.append(x_test[i])

    elif  y_test[i] == 37 :
        y_test1.append(2)
        x_test1.append(x_test[i])

    elif  y_test[i] == 50 :
        y_test1.append(3)
        x_test1.append(x_test[i])

    elif  y_test[i] == 58 :
        y_test1.append(4)
        x_test1.append(x_test[i])
```

```
    elif  y_test[i] == 73 :
        y_test1.append(5)
        x_test1.append(x_test[i])

    elif  y_test[i] == 83 :
        y_test1.append(6)
        x_test1.append(x_test[i])

y_test1= np.array(y_test1)
x_test1= np.array(x_test1)

index=[]
enum=0
for i in range (2500):
    if y_train1[i] == 0 and enum < 10:
        index.append(i)
        enum=enum+1
        if enum == 10 : i = 0
    elif y_train1[i] == 1 and enum >= 10 and enum < 20:
        index.append(i)
        enum=enum+1
        if enum == 20 : i = 0
    elif y_train1[i] == 2 and enum >= 20 and enum < 30:
        index.append(i)
```

```
        enum=enum+1
        if enum == 30 : i = 0
    elif y_train1[i] == 3 and enum >= 30 and enum < 40:
        index.append(i)
        enum=enum+1
        if enum == 40 : i = 0
    elif y_train1[i] == 4 and enum >= 40 and enum < 50:
        index.append(i)
        enum=enum+1

        if enum == 50 : i = 0
    elif y_train1[i] == 5 and enum >= 50 and enum < 60:
        index.append(i)
        enum=enum+1

        if enum == 60 : i = 0
    elif y_train1[i] == 6 and enum >= 60 and enum < 70:
        index.append(i)
        enum=enum+1

for i in range(0,70):
    plt.subplot(7,10,i+1)
    plt.imshow(x_train1[index[i]])
    fig = plt.imshow(x_train1[index[i]])
```

```
    fig.axes.get_xaxis().set_visible(False)
    fig.axes.get_yaxis().set_visible(False)

x_train1=x_train1.astype('float32')/255.0
x_test1=x_test1.astype('float32')/255

y_train1=to_categorical(y_train1,7)
y_test1=to_categorical(y_test1,7)

model=Sequential()
model.add(layers.Conv2D(32,(3,3),activation='relu',padding='same',i
nput_shape= (32, 32, 3)))
model.add(layers.Conv2D(32,(3,3),padding='same',activation='relu'))
model.add(layers.MaxPool2D())
model.add(layers.Dropout(0.25))
model.add(layers.Conv2D(64,(3,3),padding='same',activation='relu'))
model.add(layers.MaxPool2D())
model.add(layers.Dropout(0.25))
model.add(layers.Conv2D(64,(3,3),padding='same',activation='relu'))
model.add(layers.MaxPool2D())
model.add(layers.Dropout(0.4))
model.add(layers.Flatten())
model.add(layers.Dense(512,activation='relu'))
model.add(layers.Dense(7,activation='softmax'))
```

```
model.summary()

from tensorflow.keras import optimizers
from keras import losses
model.compile(loss=losses.CategoricalCrossentropy(),
              optimizer=optimizers.Adam(lr=1e-6),
              metrics=['acc'])

history=model.fit(x_train1,
                  y_train1,
                  epochs=80,
                  validation_data=(x_test1,y_test1))

test_loss, test_acc = model.evaluate(x_test1,y_test1)
print(" Loss = %",100*test_loss,"\n","Accuracy = %",100*test_acc)

acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(1,len(acc)+1)

import matplotlib.pyplot as plt
```

```
plt.figure()

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'r', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('epochs')
plt.ylabel('accuracy')
plt.legend()

plt.show()

plt.figure()

plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('epochs')
plt.ylabel('loss')
plt.legend()

plt.show()

pred=model.predict(x_test1)
```

```
for i in range(0,700):
    maxindex= np.argmax(pred[i], axis=None, out=None)
    for y in range(0,7):
        if y == maxindex:
            pred[i,y] = 1
        else :
            pred[i,y] = 0

from sklearn.metrics import confusion_matrix, classification_report



print('Confusion Matrix')
print(confusion_matrix(y_test1.argmax(axis=1),
pred.argmax(axis=1)))
print("\n",'Classification Report')
target_names = ["beetle", "can", "house", "mouse", "pickup_truck",
"shark", "sweet_pepper"]
print(classification_report(y_test1, pred,
target_names=target_names))


# test kodu :
#    test_list =
[x_train1[index[8]],x_train1[index[12]],x_train1[index[26]],x_train1[i
```

```
ndex[31]],x_train1[index[48]],x_train1[index[53]],x_train1[index[69]]
]


#    test_pred = model.predict(np.array(test_list))


#    plt.imshow(test_list[0])
#    fig = plt.imshow(test_list[0])
#    fig.axes.get_xaxis().set_visible(False)
#    fig.axes.get_yaxis().set_visible(False)
#    for j in range (7) :
#    print("Sınıf", j+1, "=" , target_names[j], "\t\t%", 100*
test_pred[0,j] )
```